Intel® 82580 Quad/Dual Gigabit Ethernet LAN Controller Datasheet

LAN Access Division (LAD)

FEATURES

External Interfaces Provided:

- PCIe v2.0 (5Gbps and 2.5Gbps) x4/x2/x1; called PCIe in this document.
- MDI (Copper) standard IEEE 802.3 Ethernet interface for 1000BASE-T, 100BASE-TX, and 10BASE-T applications (802.3, 802.3u, and 802.3ab)
- Serializer-Deserializer (SERDES) to support 1000Base-SX/LX (optical fiber)
- Serializer-Deserializer (SERDES) to support 1000BASE-KX and 1000BASE-BX for Gigabit backplane applications
- SGMII interface for SFP/external PHY connections
- NC-SI or SMBus for Manageability connection to MC
- IEEE 1149.6 JTAG

Performance Enhancements:

- Intel® I/O Acceleration Technology v3.0 supported:
- Stateless offloads (Header split, RSS)
- Direct Cache Access
- PCIe v2.1 TLP Processing Hints (TPH)
- UDP, TCP and IP Checksum offload
- UDP and TCP Transmit Segmentation Offload (TSO)
- SCTP receive and transmit checksum offload

Virtualization Ready:

- Enhanced VMDq1 support:
- Queues per port: 8 TX and 8 RX queues
- Support of up to 8 VMs per port (1 queue allocated to each VM)

Power Saving Features:

- Advanced Configuration and Power Interface (ACPI) power management states and wake-up capability
- Advanced Power Management (APM) wake-up functionality
- Low power link-disconnect state
- PCIe v2.1 LTR (Latency Tolerance Reporting)
- DMA Coalescing for improved system power management

IEEE802.1AS - Timing and Synchronization:

- IEEE 1588 Precision Time Protocol support
- Per-packet timestamp

Total Cost Of Ownership (TCO):

- IPMI MC pass-thru; multi-drop NC-SI

Additional Product Details:

- 17x17 PBGA package
- Estimated power: 2.8W (max) in dual port mode and 4.2W (max) in quad port mode
- Full data path Parity or ECC protection

321027-012EN
Revision: 2.4
March 2010

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. Intel products are not intended for use in medical, life saving, life sustaining, critical control or safety systems, or in nuclear facility applications.

Intel may make changes to specifications and product descriptions at any time, without notice.

Intel Corporation may have patents or pending patent applications, trademarks, copyrights, or other intellectual property rights that relate to the presented subject matter. The furnishing of documents and other materials and information does not provide any license, express or implied, by estoppel or otherwise, to any such patents, trademarks, copyrights, or other intellectual property rights.

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.

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. See http://www.intel.com/products/processor_number for details.

Hyper-Threading Technology requires a computer system with an Intel® Pentium® 4 processor supporting HT Technology and a HT Technology enabled chipset, BIOS and operating system. Performance will vary depending on the specific hardware and software you use. See http://www.intel.com/products/ht/Hyperthreading_more.htm for additional information.

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 website at http://www.intel.com.

Intel and Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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

Revisions

Rev	Date		Notes
0.30	Dec 2008		Initial public release of early materials.
0.31	Jan 2009		Chapter 2.0. Signals connected to the E14, F14, N12, R1, R2 and T1 corrected according to the latest ballout. Section 11.4. Updated power consumption estimates. Table 11-246. Corrected packaging information in the table. Now listed consistently as 17x17 PBGA package.
0.5	2 April 2009		Updated EAS source used as base.
1.0	12 June 2009		Updated design information chapter added; supports Samples.
1.1	1 Oct 2009		Editorial Changes.
1.2	23 Oct 2009		Chapter 12.0, Design Guidelines - 1.9V is no longer needed at the center tap. Language expressing that requirement has been removed. Figure 11-1 and Table 11-227 updated to correct errors.
1.3	5 Jan 2010		• New EAS core added to Datasheet text. • Datasheet title updated to reflect dual and quad core capabilities. • Datasheet title changed to cover Intel® 82580 Quad/Dual GbE LAN Controller. 'Dual' added. • Section 1.0, Introduction language updated to indicate dual core support. • Table 2-17, SERDES/SGMII Pins updated; now includes dual port exclusions. See asterisks. • Table 2-18, SFP Pins updated; now includes dual port exclusions. See asterisks. • Table 2-19, LED Output Pins updated; now includes dual port exclusions. See asterisks. • Table 2-20, Analog Pins updated; now includes dual port exclusions. See asterisks. • Table 2-21, Testability Pins updated; now includes dual port exclusions. See asterisks. • Table 2-26, Pin List in Alphabetical Order updated; now summarizes all dual port exclusions. See asterisks. • Table 4-70, PCI Functions Mapping (Legacy Mode) updated; information expanded. • Table 6-84, EEPROM Top Level Partitioning updated; now includes dual port exclusions. See asterisks.
2.0	15 Jan 2010		• Section 7.8.2.4, Size Filtering added. • Figure 12-4, Recommended Crystal Placement and Layout on page 671 updated. • Chapter 13.0, Thermal Management - Thermal management chapter added. added.
2.1	15 Jan 2010		• Test data updated.
2.2	26 Feb 2010		• Figure 12-5, Oscillator Solution on page 672 updated. • Table 12-251, Oscillator Manufacturers and Part Numbers updated. • Confidential stamp removed from document for posting on Developer.
2.3	5 Mar 2010		• In Section 13.4.4, Package Thermal Characteristics ; Table 13-259 and Table 13-259 have been provided with updated data. • Appendix A., Changes from the 82576; this appendix was added to the Datasheet.

Rev	Date		Notes
2.4	29 Mar 2010		- In Section 6.2.5, Device ID (LAN Base Address + Offset 0x0D), the device ID was indicated as TBD because of a poorly set build variable. That has been corrected (Device ID = 1509). - In Section 10.3.2.1.3, Request Status Command, the descriptve paragraph has been updated for clarity. - In Section 11.7.1, Mechanical; ball, solder, and pad information has been added to the section.

Contents

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

1.0 Introduction

The Intel® 82580 Quad/Dual GbE LAN Controller is a single, compact, low power component that supports quad and dual port gigabit Ethernet designs. The device offers four fully-integrated gigabit Ethernet media access control (MAC), physical layer (PHY) ports and four SGMII/SerDes ports that can be connected to an external PHY. The 82580 supports PCI Express* (PCIe v2.0 (5Gbps and 2.5Gbps)).

The device enables two-port or four port 1000BASE-T implementations using integrated PHY's. It can be used for server system configurations such as rack mounted or pedestal servers, in an add-on NIC or LAN on Motherboard (LOM) design. Another possible system configuration is for blade servers. Here, the 82580 can support up to 4 SerDes ports as LOM or mezzanine card. It can also be used in embedded applications such as switch add-on cards and network appliances.

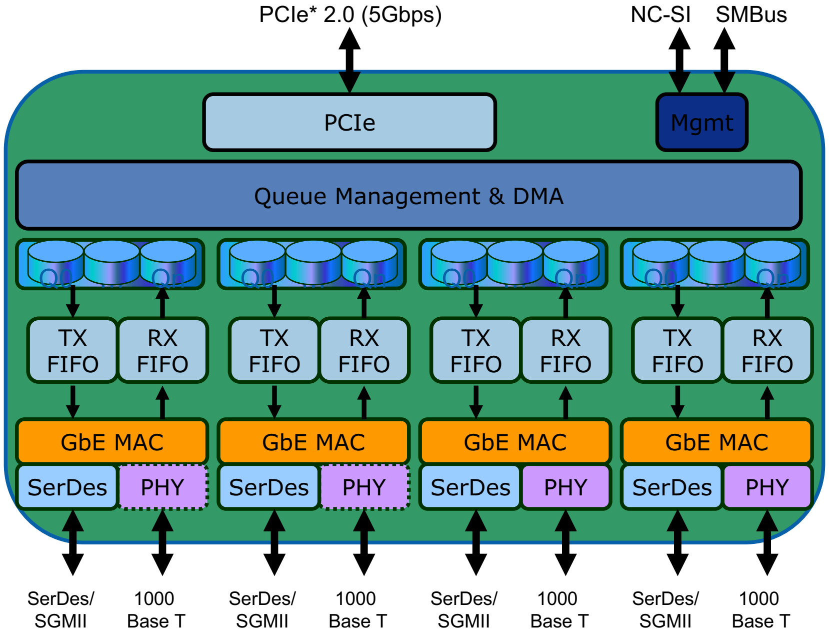


Figure 1-1. Intel® 82580 Quad/Dual GbE LAN Controller

1.1 Scope

This document provides the external architecture (including device operation, pin descriptions, register definitions, etc.) for the 82580.

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

1.2 Terminology and Acronyms

Table 1-1. Glossary

Definition	Meaning
1000BASE-BX	1000BASE-BX is the PICMG 3.1 electrical specification for transmission of 1 Gb/s Ethernet or 1 Gb/s fibre channel encoded data over the backplane.
1000BASE-KX	1000BASE-KX is the IEEE802.3ap electrical specification for transmission of 1 Gb/s Ethernet over the backplane.
1000BASE-CX	1000BASE-X over specialty shielded 150 Ω balanced copper jumper cable assemblies as specified in IEEE 802.3 Clause 39.
1000BASE-T	1000BASE-T is the specification for 1 Gb/s Ethernet over category 5e twisted pair cables as defined in IEEE 802.3 clause 40.
b/w	Bandwidth.
BIOS	Basic Input/Output System.
BMC	Baseboard Management Controller.
BT	Bit Time.
DCA	Direct Cache Access.
DFT	Design for Testability.
DQ	Descriptor Queue.
DW	Double word (4 bytes).
EEPROM	Electrically Erasable Programmable Memory. A non-volatile memory located on the LAN controller that is directly accessible from the host.
EOP	End of Packet.
FC	Flow Control.
Firmware (FW)	Embedded code on the LAN controller that is responsible for the implementation of the NC-SI protocol and pass through functionality.
Host Interface	RAM on the LAN controller that is shared between the firmware and the host. RAM is used to pass commands from the host to firmware and responses from the firmware to the host.
HPC	High - Performance Computing.
IPC	Inter Processor Communication.
IPG	Inter Packet Gap.
LAN (auxiliary Power-Up)	The event of connecting the LAN controller to a power source (occurs even before system power-up).
LOM	LAN on Motherboard.
LTR	Latency Tolerance Reporting (PCIe protocol)
LSO	Large Send Offload.
MAC	Media Access Control.
MDIO	Management Data Input/Output Interface over MDC/MDIO lines.
MIFS/MIPG	Minimum Inter Frame Spacing/Minimum Inter Packet Gap.
MMW	Maximum Memory Window.
MSS	Maximum Segment Size. Largest amount of data, in a packet (without headers) that can be transmitted. Specified in Bytes.

Table 1-1. Glossary (Continued)

Definition	Meaning
MPS	Maximum Payload Size in PCIe specification.
MTU	Maximum Transmit Unit. Largest packet size (headers and data) that can be transmitted. Specified in Bytes.
NC-SI	Network Controller Sideband Interface DMTF Specification
NIC	Network Interface Controller.
TPH	TLP Process Hints (PCIe protocol).
PCS	Physical Coding Sub layer.
PHY	Physical Layer Device.
PMA	Physical Medium Attachment.
PMD	Physical Medium Dependent.
RMII	Reduced Media Independent Interface (Reduced MII).
SA	Source Address.
SDP	Software Defined Pins.
SerDes	Serializer and De-Serializer Circuit.
SFD	Start Frame Delimiter.
SGMII	Serialized Gigabit Media Independent Interface.
SMBus	System Management Bus. A bus that carries various manageability components, including the LAN controller, BIOS, sensors and remote-control devices.
TCO	Total Cost of Ownership (TCO) System Management.
TLP	Transaction Layer Packet in the PCI Express specification.
TSO	Transmit Segmentation offload - A mode in which a large TCP/UDP I/O is handled to the device and the device segments it to L2 packets according to the requested MSS.
VPD	Vital Product Data (PCI protocol).

1.2.1 External Specification and Documents

The 82580 implements features from the following specifications.

1.2.1.1 Network Interface documents

1. IEEE standard 802.3, 2006 Edition (Ethernet). Incorporates various IEEE Standards previously published separately. Institute of Electrical and Electronic Engineers (IEEE).
2. IEEE standard 1149.1, 2001 Edition (JTAG). Institute of Electrical and Electronics Engineers (IEEE)
3. IEEE Std 1149.6-2003, IEEE Standard for Boundary-Scan Testing of Advanced Digital Networks, IEEE, 2003.
4. IEEE standard 802.1Q for VLAN
5. PICMG3.1 Ethernet/Fibre Channel Over PICMG 3.0 Draft Specification, January 14, 2003, Version D1.0
6. Serial-GMII Specification, Cisco Systems document ENG-46158, Revision 1.7
7. INF-8074i Specification for SFP (Small Form factor Pluggable) Transceiver (ftp://ftp.seagate.com/sff)
8. IEEE Std 802.3ap-2007
9. IEEE 1588^{TM} Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, November 8 2002

10. IEEE 802.1AS Timing and Synchronization for Time- Sensitive Applications in Bridged Local Area Networks Draft 2.0, February 22, 2008

1.2.1.2 Host Interface Documents

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

1.2.1.3 Networking Protocol documents

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

1.2.1.4 Manageability documents

1. DMTF Network Controller Sideband Interface (NC-SI) Specification rev 1.0.0a, June 2007
2. System Management Bus (SMBus) Specification, SBS Implementers Forum, Ver. 2.0, August 2000

1.3 Product Overview

The 82580 supports 4 SerDes or SGMII ports for MAC to MAC blade server connections or MAC to external PHY connections. Alternatively, the four internal 1000BASE-T PHY's can be used to implement a quad port NIC or LOM design.

The 82580 targets server system configurations such as rack mounted or pedestal servers, where the 82580 can be used as add-on NIC or LAN on Motherboard (LOM) design. Another system configuration is blade servers, where it can be used on Mezzanine card or LOM. The 82580 can also be used in embedded applications such as switch add-on cards and network appliances.

1.4 External Interface

1.4.1 PCIe Interface

The PCIe v2.0 (5Gbps) Interface is used by the 82580 as a host interface. The interface supports both PCIe v2.0 (2.5Gbps) and PCIe v2.0 (5Gbps) rates and can be configured to x4, x2 and x1. The maximum aggregated raw bandwidth for a typical x4 PCIe v2.0 (5Gbps) configuration is 16 Gb/s in each direction. See Section 2.1.1 for full pin description. The timing characteristics of this interface are defined in PCI Express Card Electromechanical Specification rev 2.0 and in the PCIe v2.0 (5Gbps and 2.5Gbps) specification.

1.4.2 Network interfaces

Four independent interfaces are used to connect the four 82580 ports to external devices. The following protocols are supported:

- MDI (Copper) support for standard IEEE 802.3 Ethernet interface for 1000BASE-T, 100BASE-TX, and 10BASE-T applications (802.3, 802.3u, and 802.3ab)
- SerDes interface to connect over a backplane to another SerDes compliant device or to an Optical module. The 82580 supports both 1000BASE-BX and 1000BASE-KX (Without IEEE802.3ap Backplane Auto-Negotiation)
- SGMII interface to attach to an external PHY, either on board or via an SFP module. The SGMII interface shares the same pins as the SerDes

See Section 2.1.8.2 and Section 2.1.6 for full pin description; Section 11.6.3 and Section 11.6.4 for timing characteristics of this interface.

1.4.3 EEPROM Interface

The 82580 uses an EEPROM device for storing product configuration information. Several words of the EEPROM are accessed automatically by the 82580 after reset in order to provide pre-boot configuration data that must be available to the 82580 before it is accessed by host software. The remainder of the stored information is accessed by various software modules used to report product configuration, serial number, etc.

The 82580 is intended for use with a SPI (4-wire) serial EEPROM device such as an AT25040AN or compatible EEPROM device (See Section 11.8.2 for full list of supported EEPROM devices). See Section 2.1.2 for full pin description and Section 11.6.2.5 for timing characteristics of this interface.

1.4.4 Serial Flash Interface

The 82580 provides an external SPI serial interface to a Flash or Boot ROM device such as the Atmel* AT25F1024 or compatible Flash device (See Section 11.8.1 for full list of supported Flash devices). The 82580 supports serial Flash devices with up to 64 Mbit (8 MByte) of memory. The size of the Flash used by the 82580 can be configured by the EEPROM. See Section 2.1.2 for full pin description and Section 11.6.2.4 for timing characteristics of this interface.

Note: Though the 82580 supports devices with up to 8 MB of memory, bigger devices can also be used. Accesses to memory beyond the Flash device size results in access wrapping as only the lower address bits are used by the Flash device.

1.4.5 SMBus Interface

SMBus is an optional interface for pass-through and/or configuration traffic between a BMC and the 82580.

The 82580's SMBus interface can be configured to support both slow and fast timing modes. See Section 2.1.3 for full pin description and Section 11.6.2.2 for timing characteristics of this interface.

1.4.6 NC-SI Interface

NC-SI and SMBus interfaces are optional for pass-through and/or configuration traffic between a BMC and the 82580. The NC-SI interface meets the DMTF NC-SI Specification, Rev. 1.0.0a as an integrated Network Controller (NC) device.

See Chapter 2.1.4 for full pin description and Chapter 11.6.2.6 for timing characteristics of this interface.

1.4.7 MDIO/I^{2}C 2 wires Interfaces

The 82580 implements four management Interfaces for control of an optional external PHY. Each interface can be either a 2-wire Standard-mode I^{2}C interface used to control an SFP module or an MII Management Interface (also known as the Management Data Input/Output or MDIO Interface) for control plane connection between the MAC and PHY devices (master side). This interface provides the MAC and software with the ability to monitor and control the state of the external PHY. The 82580 supports the data formats defined in IEEE 802.3 clause 22.

The 82580 supports shared MDIO operation and separate MDIO connection. When configured via the *MDICNFG* register to separate MDIO operation each MDIO interface should be connected to the relevant PHY. When configured via the *MDICNFG* register to shared MDIO operation the MDC/MDIO interface of LAN port 0 can be shared by all ports to support connection to a multi-port PHY with a single MDC/MDIO interface. See Section 2.1.7 for full pin description, Section 11.6.2.8 for MDIO timing characteristics and Section 11.6.2.3 for I^{2}C timing characteristics of this interface.

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

The 82580 has four software-defined pins (SDP pins) per port that can be used for miscellaneous hardware or software-control purposes. These pins can be individually configurable to act as either standard inputs, general-purpose interrupt (GPI) inputs or output pins. The default direction of each pin is configurable via the EEPROM (see Section 6.2.18, Section 8.2.1 and Section 8.2.3), as well as the default value of all pins configured as outputs. Further information on SDP usage can be found in Section 3.4 and Section 7.9.4. See Section 2.1.5 for pin description of this interface.

1.4.9 LEDs Interface

The 82580 implements four output drivers per port intended for driving external LED circuits. 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. Furthermore, the hardware-default configuration for all LED outputs can be specified via EEPROM fields (see Section 6.2.15 and Section 6.2.17), thereby supporting LED displays configurable to a particular OEM preference.

See Section 2.1.8.1 for full pin description of this interface.

See Section 7.5 for more detailed description of LED behavior.

1.5 Features

Table 1-2 to Table 1-7 list the 82580's features and compares them to other LAD products .

Table 1-2. 82580 Network Features

Feature	82580	82599	82575	82576
Half duplex at 10/100 Mb/s operation and full duplex operation at all supported speeds	Y	100 Mb/s full duplex	Y	Y
10/100/1000 Copper PHY integrated on-chip	4 ports	N	2 ports	2 ports
Jumbo frames supported	Y	Y	Y	Y
Size of jumbo frames supported	9.5 KB	16 KB	9.5 KB	9.5 KB
Flow control support: send/receive PAUSE frames and receive FIFO thresholds	Y	Y	Y	Y
Statistics for management and RMON	Y	Y	Y	Y
802.1q VLAN support	Y	Y	Y	Y
SerDes interface for external PHY connection or system interconnect	4 ports	2 ports	2 ports	2 ports
1000BASE-KX interface for Blade Server Backplane connections	Y	Y	N	N
802.3ap Backplane Auto-negotiation	N	Y	N	N
SGMII interface for external 1000BASE-T PHY connection	4 ports	2 ports	2 ports	2 ports
Fiber/copper auto-sense	4 ports	N/A	2 ports	2 ports
SerDes support of non-Auto-Negotiation partner	Y	Y	Y	Y
SerDes signal detect	Y	N	Y	Y
External PHY control I/F MDC/MDIO 2 wire I/F	Shared or per function Per function	Per function Per function	Per function Per function	Per function Per function

Table 1-3. 82580 Host Interface Features (Sheet 1 of 2)

Feature	82580	82599	82575	82576
PCIe revision	2.0 (5 Gbps or 2.5 Gbps)	2.0 (5 Gbps or 2.5 Gbps)	2.0 (2.5 Gbps)	2.0 (2.5 Gbps)
PCIe physical layer	Gen 2	Gen 2	Gen 1	Gen 1
Bus width	x1, x2, x4	x1, x4, x8	x1, x2, x4	x1, x2, x4
64-bit address support for systems using more than 4 GB of physical memory	Y	Y	N	Y
Outstanding requests for Tx buffers per port	24 Per port and for all ports	16	4	4
Outstanding requests for Tx descriptors per port	4 Per port and for all ports	8	1	1
Outstanding requests for Rx descriptors per port	4 Per port and for all ports	4	1	1
Credits for posted writes	4	8	2	2
Max payload size supported	512 B	512 B	256 B	512 B
Max request size supported	2 KB	2 KB	512 B	512 B

Table 1-3. 82580 Host Interface Features (Sheet 2 of 2)

Link layer retry buffer size	3.2 KB	3.2 KB	2 KB	2 KB
Vital Product Data (VPD)	Y	Y	N	Y
End to End CRC (ECRC)	Y	Y	N	N
LTR (Latency Tolerance Reporting)	Y	N	N	N
TPH	Y	N	N	N
CSR access via Configuration space	Y	N	N	N

Table 1-4. 82580 LAN Functions Features

Feature	82580	82599	82575	82576
Programmable host memory receive buffers	Y	Y	Y	Y
Descriptor ring management hardware for transmit and receive	Y	Y	Y	Y
ACPI register set and power down functionality supporting D0 & D3 states	Y	Y	Y	Y
Software controlled global reset bit (resets everything except the configuration registers)	Y	Y	Y	Y
Software Definable Pins (SDP) - per port	4	8	4	4
Four SDP pins can be configured as general purpose interrupts	Y	Y	Y	Y
Wake up	Y	Y	Y	Y
Flexible wake-up filters	8	6	4	6
Flexible filters for queue assignment in normal operation	8	N	N	N
IPv6 wake-up filters	Y	Y	Y	Y
Default configuration by the EEPROM for all LEDs for pre-driver functionality	4 LEDs	4 LEDs	4 LEDs	4 LEDs
LAN function disable capability	Y	Y	Y	Y
Programmable memory transmit buffers	Y	Y	Y	Y
Double VLAN	Y	Y	Y	Y
IEEE 1588	Y	Y	N	Y
Per-Packet Timestamp	Y	N	N	N

Table 1-5. 82580 LAN Performance Features

Feature	82580	82599	82575	82576
TCP segmentation offload Up to 256 KB	Y	Y	Y	Y
iSCSI TCP segmentation offload (CRC)	N	Y	N	N
IPv6 support for IP/TCP and IP/UDP receive checksum offload	Y	Y	Y	Y
Fragmented UDP checksum offload for packet reassembly	Y	Y	Y	Y
Message Signaled Interrupts (MSI)	Y	Y	Y	Y
Message Signaled Interrupts (MSI-X) number of vectors	10	256	10	25
Packet interrupt coalescing timers (packet timers) and absolute-delay interrupt timers for both transmit and receive operation	Y	Y	N	Y
Interrupt throttling control to limit maximum interrupt rate and improve CPU utilization	Y	Y	Y	Y
Rx packet split header	Y	Y	Y	Y
Receive Side Scaling (RSS) number of queues per port	Up to 8	Up to 16	4	Up to 16
Total number of Rx queues per port	8	128	4	16
Total number of TX queues per port	8	128	4	16

Table 1-5. 82580 LAN Performance Features (Continued)

Feature	82580	82599	82575	82576
RX header replication Low latency interrupt DCA support TCP timer interrupts No snoop Relax ordering	Yes to all	Yes to all	Yes to all	Yes to all
TSO interleaving for reduced latency	Y	Y	N	Y
Receive side coalescing	N	Y	N	N
SCTP receive and transmit checksum offload	Y	Y	N	Y
UDP TSO	Y	Y	N	Y
IPSec offload	N	Y	N	Y

Table 1-6. 82580 Virtualization Features

Feature	82580	82599	82575	82576
Support for Virtual Machines Device queues (VMDq) per port	8 pools (single queue)	16/32/64 pools	4 pools	8 pools
L2 MAC address filters (unicast and multicast)	24	128	16	24
L2 VLAN filters	Per pool	64	-	Per pool
PCI-SIG SR-IOV	N	16/32/64 VF	N	8 VF
Multicast/Broadcast Packet replication	Y on Receive	Y	N	Y
VM to VM Packet forwarding (Packet Loopback)	N	Y	N	Y
RSS replication	N	Y	N	N
Traffic shaping	N	Y	N	Y
MAC and VLAN anti-spoofing	N	Y	N	Y
Malicious driver detection	Y	N	N	N
Per-pool statistics	Y	Y	N	Y
Per-pool off loads	Y	Y	Partial	Y
Per-pool jumbo support	Y	Y	N	Y
Mirroring rules	4	4	0	4

Table 1-7. 82580 Manageability Features

Feature	82580	82576	82599
Advanced pass-through-compatible management packet transmit/ receive support	Y	Y	Y
Managed ports on SMBus interface to external BMC	4	2	2
Fail-over support over SMBus	N	Y	Y
Auto-ARP reply over SMBus	N	Y	Y
NC-SI Interface to an External BMC	Y	Y	Y
DMTF NC-SI protocol standards support	Y	Y	Y
L2 address filters	2	4	4
VLAN L2 filters	8	8	8
EtherType filters	4	4	4
Flex L4 port filters	8	16	16

Table 1-7. 82580 Manageability Features (Continued)

Feature	82580	82576	82599
Flex TCO filters	1	4	4
L3 address filters (IPv4)	4	4	4
L3 address filters (IPv6)	4	4	4

1.6 Overview of Changes Compared to the 82576

The following section describes modifications done in the 82580 compared to the 82576.

1.6.1 Network Interface

1.6.1.1 Quad port

The 82580 supports 4 LAN ports (10/100/1000BASE-T on Serdes) and associated MAC and DMA queues enabling reduction of BOM cost and board space when implementing quad port Ethernet NICs or LOM designs.

1.6.1.2 Shared MDIO support

The 82580 enables sharing of the MDIO interface on LAN port 0 by all ports connected to an external 1000BASE-T PHY to support connection to an external multi-port PHY device.

To abstract actual MDIO interface configuration from Software driver, MDIO setup (PHY Address, External or Internal PHY and Shared MDIO) is loaded into the MDICNFG register from the EEPROM following reset. Driver can issue read or write operation to the internal PHY registers using the MDIC register, without a need to know actual MDIO configuration. Further information can be found in Section 3.5.2.2 and Section 8.2.5.

1.6.1.3 I^{2}C Clock Stretching

There are situations where an I^{2}C slave can not support the clock speed or needs to delay an access initiated by the master. Delaying the access is done by a mechanism referred to as clock stretching. An I^{2}C slave is allowed to hold down the clock if it needs to delay an access. The master is required to read back the clock signal after releasing it to a high-z state and wait until the line has actually gone high as a result of an external Pull-up resistor.

When configured to operate in I^{2}C mode, the 82580 supports clock stretching on the I2C_CLK pin. When accessing PHY registers via the I^{2}C interface the PHY can delay the access or reduce clock speed if required.

1.6.1.4 1000BASE-KX Backplane Ethernet Interface

The 82580 can interface up to four 1Gbps 1000BASE-KX lanes for Blade Server backplane interconnect without need for additional glue logic. The 82580 supports only parallel detection of 1000BASE-KX signaling and does not support the full Auto-Negotiation for Backplane Ethernet protocol.

1.6.2 HOST Interface

1.6.2.1 PCIe v2.0 (5Gbps and 2.5Gbps) Link speed

The 82580 supports X4, X2 and X1 PCIe v2.0 (5Gbps and 2.5Gbps) port configurations.

1.6.2.2 ECRC

The 82580 supports generation and checking of ECRC on PCI-Express traffic.

1.6.2.3 64 bit BARs support

The 82580 64 bit BAR support modified to be similar to the 82599.

1.6.2.4 MSI-X Support

Number of MSI-X vectors supported by the 82580 changed to 10.

1.6.2.5 Outstanding Requests and Credits

Following changes to the outstanding request count and credits are implemented in the 82580:

- Outstanding requests for TX buffers increased to 24 to improve small packet TX performance. Each function can generate up to 24 TX buffer outstanding requests, but sum of outstanding TX Buffer requests generated by all functions can not pass 24.
- Outstanding requests for TX descriptor fetch increased to 4. Each function can generate up to 4 TX descriptor fetch outstanding requests, but sum of TX descriptor fetch outstanding requests generated by all functions can not pass 4.
- Outstanding requests for RX descriptor fetch increased to 4. Each function can generate up to 4 RX descriptor fetch outstanding requests, but sum of RX descriptor fetch outstanding requests generated by all functions can not pass 4.
- Number of Credits for posted writes has been increased from 2 to 4.

1.6.2.6 CSR Access Via Configuration Space

Server systems run into scaling issues when adding a large number of PCIe cards that require IO address resources. IO addresses on the PCIe interface are limited to 64K of addresses for all PCIe devices by legacy definition. Furthermore, although the 82580 only needs two 32-bit registers (8 bytes), the device must allocate 32 bytes per-function in IO space. Lastly, the PCI Bridge specification requires a granularity of 4K allocation per PCI segment for IO space (dividing the 64K address space into at most 16 segments).

The BIOS may further reduce the available IO space by allocating IO space in a sub-optimal manner (e.g. lots of space in one 16KB chunk, but exhausted in another) due to need to support required legacy IO offsets such as mouse/keyboard or POST code ports.

Networking products only require IO space to support 'legacy' 16-bit operating environments where memory-mapped PCIe device CSRs (e.g. big 32-bit or 64-bit address offsets) can't be mapped arbitrarily into the 16-bit address space of the executable environment. These legacy environments happen to include pre-boot BIOS environments - such as what PXE and 16-bit iSCSI boot are designed for modern platforms using EFI BIOS components do not need IO mode. Modern operating systems also do not need IO mode.

To support legacy pre-boot 16-bit operating environments without requiring IO address space, the 82580 enables accessing CSRs via configuration address space using the IOADDR (Configuration address 0x98) and IODATA (Configuration address 0x9C). See Section 8.1.1.6 and Section 9.5.4.1 for further information.

1.6.3 Boundary Scan

In addition to support of the IEEE 1149.1 boundary scan standard the 82580 supports the IEEE 1149.6 boundary scan standard to enable testing of it's high speed differential interfaces.

1.6.4 Performance Features

1.6.4.1 TLP Process Hints (TPH) Support

The 82580 supports the TPH capability defined in the PCI Express specification. It can provide hints to the root complex about the destination cache of DMA memory writes or instruct the root complex to read specific type of traffic directly from the cache. See Section 7.7.2 for more information.

1.6.5 Receive and Transmit Queues

The number of Receive and Transmit queues supported by the 82580 for RSS filtering and Virtualization is 8 per port.

1.6.5.1 Unused Receive and Transmit Ports Buffer Sharing

The 82580 supports up to 4 Gigabit Ethernet ports. When device is configured to work only as a single port or dual port device, available internal receive and transmit buffer memory for the remaining active ports, can be increased by four for the single port case and by two for the dual port case. See Section 7.1.3.2 and Section 7.2.1.2 for more information.

1.6.6 Virtualization

The 82580 does not support the following virtualization features previously supported by the 82576:

- PCIe SR-IOV
- VM to VM packet forwarding
- Anti-Spoof protection

The 82580 supports 8 queues per port that can be shared between eight virtual machines (VMDq1). By associating receive and transmit packets to separate queues dedicated to a VM, performance of the VMM (Virtual Machine Monitor) is improved. The 82580 also supports the following hardware Virtualization features:

- Per VM interrupt assignment
- Enabling read of statistic counters by VMs without initiating a clear by read
- Copy of received multicast and broadcast packets to multiple queues
- Per VM Offload
- Per Pool statistics
- Mirroring
- Storm Control

1.6.7 Malicious Driver Detection

Malicious behavior exhibited by the driver can be a result of incorrect activation of the network controller or a virus on a certain VM. The 82580 contains internal circuitry to protect from an attack on one virtual machine disrupting operation of other virtual machines. When malicious driver behavior is detected on a certain queue, the 82580 disables activity of the queue and sends notification to the VMM. See Section 7.8.2.8.2 for details.

1.6.8 2-tuple filtering

The 82580 supports only 2-tuple filtering compared to previous products that supported 5-tuple filtering. The 82580 enables queuing, generating immediate interrupts and time stamping according to TCP/UDP port destination address and L4 protocol fields. See Section 7.1.1 and Section 7.1.1.5 for more information.

1.6.9 Security Offload

The 82580 does not support the IPSec and LinkSec offload features that were previously supported by the 82576.

1.6.10 Quality of Service

1.6.10.1 DCE and BCN

The 82580 does not support the Reedtown (Data Center Ethernet) and Backward Congestion Notification (BCN) features that were previously supported by 82576.

1.6.10.2 Transmit Queue Prioritization

The 82580 supports strict Transmit queue prioritization to enable transmission of high priority real-time packets ahead of low priority packets. A Transmit queue is either a high priority queue or low priority queue, with the high priority queues always served ahead of the low priority queues. See Section 7.2.2.5 for information.

1.6.11 Manageability

1.6.11.1 SMBus Interface to External BMC

The 82580 supports 4 managed ports on SMBus interface to external BMC.

1.6.11.1.1 Teaming and Fail-over on SMBus

The LAN ports act independently of each other and each port has a separate SMBus address. As a result, fail-over where manageability traffic is routed to an active port if any of the ports fail, is not supported internally. The BMC is responsible for teaming and fail-over.

1.6.11.1.2 Auto-ARP Reply on SMBus

The 82580 can not be programmed for auto-ARP reply on reception of ARP request packets and does not support sending of gratuitous ARP packets to reduce the traffic over the BMC interconnect.

1.6.11.2 NC-SI Interface to External BMC

The 82580 supports 4 managed ports on NC-SI interface to external BMC.

The 82580 supports the DMTF NC-SI Specification, revision 1.0.0a, as a PHY-side device based on NC-SI specification, revision 1.2, similar to the 82576. However, the 82580 does not support the MII protocol over the NC-SI pins.

1.6.11.2.1 NC-SI Commands.

Support for the NC-SI Get Controller Packet Statistics command was added in the 82580.

Support for filtering related NC-SI commands and NC-SI flow control were removed. See Section 10.2.1 and Section 10.3.1 for supported NC-SI commands.

1.6.11.2.2 NC-SI Hardware Arbitration

The 82580 does not support NC-SI HW arbitration between different Network Controller packages. Arbitration is done by the BMC Software. Only one package can be in the Selected state at a time. Otherwise, hardware electrical buffer conflicts will occur between packages.

1.6.11.3 Management Filtering

Quantity of management filters per port in the 82580 is reduced compared to the 82576. Changes in amount of management filters is shown in Table 1-8.

Table 1-8. Management Filters Reduction Per Port

Filter Type	82576 Filters	82580 Filters
Destination MAC address (6-bytes)	4	2
UDP/TCP port (16-bit)	16	8
Flexible (128-byte)	4	1

Description of management filters supported in the 82580 can be found in Section 8.22 and Section 10.4.

1.6.11.4 Exclusive Management Filtering

In previous network controller chips the BMC needed to define which packets that were destined to management should also be forwarded to the host. In the 82580, decisions regarding forwarding of packets to the host and to the BMC are separate and are configured through two sets of registers. However, the BMC may define some types of traffic as exclusive. This traffic will be forwarded only to the BMC, even if it passes the filtering process of the host. These types of traffic are defined using the MNGONLY register. Further information can be found in Section 7.1.2.3 and Section 10.4.1.

1.6.12 Embedded Features

1.6.12.1 Flex Filters

Two additional Flex filters were added to the 82580, making a total of 8 flexible filters. In addition to Wake-on-Lan (WoL) activity, flex filters in the 82580 can also be used for queuing decisions in normal (D0) operating mode. See Section 5.6.3.2 and Section 7.1.1.6 for more information.

1.6.12.2 Time SYNC (IEEE1588 and IEEE 802.1AS)

1.6.12.2.1 Per Packet Timestamp

The 82580 supports adding an optional tailored header before the MAC header of the packet in the receive buffer. The 128 bit tailored header contains a 64 bit timestamp (MSB bits) and 64 reserved bits (LSB bits).

The timestamp is sampled on the MAC/PHY interface, on detection of a received packet that meets certain criteria. Time stamping the received packet on the PHY/MAC interface avoids incurring additional delays due to PCIe latencies. The timestamp is placed in the receive buffer ahead of the actual received packet. See Section 7.1.10 for more information.

1.6.12.2.2 Improved System Time Accuracy

System time accuracy has been increased in the 82580. System time Corrections with a resolution of 2^{-32} ns can be entered using the *TIMINCA.Incvalue* field (see Section 7.9.3 for further information). The structure of system-related registers has been modified to enable improved accuracy.

1.6.12.2.3 SYSTIM Synchronized Pulse Generation on SDP pins

Capability to generate a pulse synchronized to system time has been added to the 82580. See Section 7.9.4.1.2 for more information.

1.6.12.2.4 Time SYNC Interrupts

Additional capability to generate interrupts on occurrence of Time Sync events has been added to the 82580. See Section 7.9.5 for more information.

1.6.13 Power Saving

1.6.13.1 DMA Coalescing

The 82580 supports DMA Coalescing to enable synchronizing device activity and optimize power management of memory, CPU and RC internal circuitry. By synchronizing PCIe activity between ports and shaping traffic on PCIe link to increase duration of idle periods, system can stay in lower power states for a longer duration and reduce overall power consumption.

When in DMA Coalescing (Buffer Fill) operating mode, PCIe link is optionally placed in L1 power saving state and DMA activity is placed on hold. The 82580 moves into Buffer Flush mode when internal receive buffers pass a pre-determined threshold value, a watchdog timer expires, or the PCIe interface invokes a move out of DMA Coalescing state. See Section 5.7.

1.6.13.2 Latency Tolerance Reporting (LTR)

The 82580 supports PCIe LTR messages to enable report of service latency and bandwidth requirements for memory reads and writes to the Root Complex. LTR messaging support enables central platform resources (such as main memory, RC internal interconnects, snoop resources, and such) to be power managed without impacting Endpoint functionality and performance. See Section 5.8.

1.7 Device Data Flows

1.7.1 Transmit Data Flow

Table 1-9 provides a high level description of all data/control transformation steps needed for sending Ethernet packets to the line.

Table 1-9. Transmit Data Flow

Step	Description
1	The host creates a descriptor ring and configures one of the 82580's transmit queues with the address location, length, head and tail pointers of the ring (one of 8 available Tx queues).
2	The host is requested by the TCP/IP stack to transmit a packet, it gets the packet data within one or more data buffers.
3	The host initializes descriptor(s) that point to the data buffer(s) and have additional control parameters that describe the needed hardware functionality. The host places that descriptor in the correct location at the appropriate Tx ring.
4	The host updates the appropriate queue tail pointer (TDT)
5	The 82580's DMA senses a change of a specific TDT and as a result sends a PCIe request to fetch the descriptor(s) from host memory.
6	The descriptor(s) content is received in a PCIe read completion and is written to the appropriate location in the descriptor queue internal cache.
7	The DMA fetches the next descriptor from the internal cache and processes its content. As a result, the DMA sends PCIe requests to fetch the packet data from system memory.
8	The packet data is received from PCIe completions and passes through the transmit DMA that performs all programmed data manipulations (various CPU off loading tasks as checksum off load, TSO off load, etc.) on the packet data on the fly.
9	While the packet is passing through the DMA, it is stored into the transmit FIFO. After the entire packet is stored in the transmit FIFO, it is forwarded to the transmit switch module.
10	The transmit switch arbitrates between host and management packets and eventually forwards the packet to the MAC.
11	The MAC appends the L2 CRC to the packet and sends the packet to the line using a pre-configured interface.
12	When all the PCIe completions for a given packet are done, the DMA updates the appropriate descriptor(s).
13	After enough descriptors are gathered for write back or the interrupt moderation timer expires, the descriptors are written back to host memory using PCIe posted writes. Alternatively, the head pointer can only be written back.
14	After the interrupt moderation timer expires, an interrupt is generated to notify the host device driver that the specific packet has been read to the 82580 and the driver can release the buffers.

1.7.2 Receive Data Flow

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

Table 1-10. Receive Data Flow

Step	Description
1	The host creates a descriptor ring and configures one of the 82580's receive queues with the address location, length, head, and tail pointers of the ring (one of 8 available Rx queues).
2	The host initializes descriptors that point to empty data buffers. The host places these descriptors in the correct location at the appropriate Rx ring.
3	The host updates the appropriate queue tail pointer (RDT).
4	The 82580's DMA senses a change of a specific RDT and as a result sends a PCIe request to fetch the descriptors from host memory.

Table 1-10. Receive Data Flow (Continued)

Step	Description
5	The descriptors content is received in a PCIe read completion and is written to the appropriate location in the descriptor queue internal cache.
6	A packet enters the Rx MAC. The RX MAC checks the CRC of the packet.
7	The MAC forwards the packet to an Rx filter
8	If the packet matches the pre-programmed criteria of the Rx filtering, it is forwarded to the Rx FIFO. VLAN and CRC are optionally stripped from the packet and L3/L4 checksum are checked and the destination queue is fixed.
9	The receive DMA fetches the next descriptor from the internal cache of the appropriate queue to be used for the next received packet.
10	After the entire packet is placed into the Rx FIFO, the receive DMA posts the packet data to the location indicated by the descriptor through the PCIe interface. If the packet size is greater than the buffer size, more descriptors are fetched and their buffers are used for the received packet.
11	When the packet is placed into host memory, the receive DMA updates all the descriptor(s) that were used by packet data.
12	After enough descriptors are gathered for write back or the interrupt moderation timer expires or the packet requires immediate forwarding, the receive DMA writes back the descriptor content along with status bits that indicate the packet information including what off loads were done on that packet.
13	After the interrupt moderation timer completes or an immediate packet is received, the 82580 initiates an interrupt to the host to indicate that a new received packet is already in host memory.
14	Host reads the packet data and sends it to the TCP/IP stack for further processing. The host releases the associated buffers and descriptors once they are no longer in use.

§ §

NOTE: This page intentionally left blank.

2.0 Pin Interface

2.1 Pin Assignment

The 82580 is packaged in a 17x17 PBGA package with 1.0 mm ball pitch. The pin count (or ball count) is 256.

Table 2-1. Signal Type Definition

Type	Description	DC specification
In	LVTTL input-only signal.	See Section 11.6.1.1
Out	LVTTL Output active driver.	See Section 11.6.1.1 and Section 11.6.1.2
T/S	LVTTL bi-directional, tri-state input/output pin.	See Section 11.6.1.1
O/D	Open Drain allows multiple devices to share line using wired-OR configuration.	See Section 11.6.1.3
NC-SI-in	NC-SI compliant input signal	See Section 11.6.1.4
NC-SI-out	NC-SI compliant output signal	See Section 11.6.1.4
A	Analog signals	See Section 11.6.4
A-in	Analog input signals	See Section 11.6.3
A-out	Analog output signals	See Section 11.6.3
B	Input bias	See Section 11.6.6 and Section 11.6.7
PS	Power Supply	

2.1.1 PCIe

The AC specification for these pins is described in Section 11.6.2.10.

Table 2-2. PCIe Pins

Reserved	Symbol	Ball #	Type	Name and Function
	PE_CLK_p PE_CLK_n	A16 A15	A-in	PCIe Differential Reference Clock in: A 100MHz 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	A12 B12	A-out	PCIe Serial Data output Lane 0: A serial differential output pair running at a bit rate of 2.5Gb/s or 5Gb/s.
	PET_1_p PET_1_n	A11 B11	A-out	PCIe Serial Data output Lane 1: A serial differential output pair running at a bit rate of 2.5Gb/s or 5Gb/s.
	PET_2_p PET_2_n	A6 B6	A-out	PCIe Serial Data output Lane 2: A serial differential output pair running at a bit rate of 2.5 Gb/s or 5Gb/s.

Table 2-2. PCIe Pins (Continued)

Reserved	Symbol	Ball #	Type	Name and Function
	PET_3_p PET_3_n	A5 B5	A-out	PCIe Serial Data output Lane 3: A serial differential output pair running at a bit rate of 2.5Gb/s or 5Gb/s.
	PER_0_p PER_0_n	A14 B14	A-in	PCIe Serial Data input Lane 0: A Serial differential input pair running at a bit rate of 2.5Gb/s or 5Gb/s.
	PER_1_p PER_1_n	A9 B9	A-in	PCIe Serial Data input Lane 1: A Serial differential input pair running at a bit rate of 2.5Gb/s or 5Gb/s.
	PER_2_p PER_2_n	A8 B8	A-in	PCIe Serial Data input Lane 2: A Serial differential input pair running at a bit rate of 2.5Gb/s or 5Gb/s.
	PER_3_p PER_3_n	A3 B3	A-in	PCIe Serial Data input Lane 3: A Serial differential input pair running at a bit rate of 2.5Gb/s or 5Gb/s.
	PE_WAKE_N	D16	O/D	WAKE#: Active low signal pulled to '0' to indicate that a Power Management Event (PME) is pending and the PCI Express link should be restored. Defined in the PCI Express CEM specification.
	PE_RST_N	B1	In	PERST#: Active low PCI Express fundamental reset input. When pulled to '0' resets chip and when de-asserted (set to '1') indicates that power and PCI Express reference clock are within specified values. Defined in the PCI Express specification. On exit from reset all registers and state machines are set to their initialization values.
	PE_TXVTERM1 PE_TXVTERM2 PE_TXVTERM3 PE_TXVTERM4	C6 C8 C9 C11	A-in	Should be connected to 1.8V power supply for termination
	PE_TRIM1 PE_TRIM2	A2 A1	A	PCIe Trimming A 1.5KΩ 1% resistor connected between these pins.

2.1.2 Flash and EEPROM Ports

The AC specification for these pins is described in Section 11.6.2.4 to Section 11.6.2.5.

Table 2-3. Flash and EEPROM Ports Pins

Reserved	Symbol	Ball #	Type	Name and Function
	FLSH_SI	B15	T/S	Serial Data output to the Flash
	FLSH_SO	C15	In	Serial Data input from the Flash
	FLSH_SCK	B16	T/S	Flash serial clock Operates at 15.625MHz.
	FLSH_CE_N	C16	T/S	Flash chip select Output
	EE_DI	E15	T/S	Data output to EEPROM
	EE_DO	F15	In	Data input from EEPROM
	EE_SK	E16	T/S	EEPROM serial clock output Operates at ~2 MHz.
	EE_CS_N	F16	T/S	EEPROM chip select Output

2.1.3 System Management Bus (SMB) Interface

The AC specification for these pins is described in Section 11.6.2.2.

Table 2-4. SMB Interface Pins

Reserved	Symbol	Ball #	Type	Name and Function
	SMBD	G3	T/S, O/D	SMB Data. Stable during the high period of the clock (unless it is a start or stop condition).
	SMBCLK	E5	T/S, O/D	SMB Clock. One clock pulse is generated for each data bit transferred.
	SMBALRT_N	G4	T/S, O/D	SMB Alert: acts as an Interrupt pin of a slave device on the SMB

2.1.4 NC-SI Interface Pins

The AC specification for these pins is described in Section 11.6.2.6.

Table 2-5. NC-SI Interface Pins

Reserved	Symbol	Ball #	Type	Name and Function
	NCSI_CLK_IN	H1	NC-SI-In	NC-SI Reference Clock Input – Synchronous clock reference for receive, transmit and control interface. It is a 50MHz clock /- 50 ppm. Note: When the 82580 drives the NC-SI clock NCSI_CLK_IN should be connected to NCSI_CLK_OUT pin on-board.
	NCSI_CLK_OUT	H2	NC-SI-Out	NC-SI Reference Clock Output – Synchronous clock reference for receive, transmit and control interface. It is a 50MHz clock /- 50 ppm. Serves as a clock source to the BMC and the 82580 (when configured so).
	NCSI_CRS_DV	H3	NC-SI-Out	CRS/DV – Carrier Sense / Receive Data Valid.
	NCSI_RXD_1 NCSI_RXD_0	J1 H4	NC-SI-Out	Receive Data – Data signals from the 82580 to BMC.
	NCSI_TX_EN	J2	NC-SI-In	Transmit Enable.
	NCSI_TXD_1 NCSI_TXD_0	J4 J3	NC-SI-In	Transmit Data – Data signals from BMC to the 82580.

2.1.5 Miscellaneous Pins

The AC specification for the XTAL pins is described in sections 11.6.5.

Table 2-6. Miscellaneous Pins

Reserved	Symbol	Ball #	Type	Name and Function
	SDP0_0 SDP0_1 SDP0_2 SDP0_3	K1 K2 K3 K4	T/S	SW Defined Pins for port 0: These pins are reserved pins that are software programmable write/read input/output capability. These default to inputs upon power up, but may have their direction and output values defined in the EEPROM. The SDP bits may be mapped to the General Purpose Interrupt bits when configured as inputs. The SDP0[0] pin can be used as a watchdog output indication. All the SDP pins can be used as SFP sideband signals (TxDisable, present & TxFault). The 82580 does not use these signals; it is available for SW control over SFP.
	SDP1_0 SDP1_1 SDP1_2 SDP1_3	L1 L2 L3 L4	T/S	SW Defined Pins for port 1: Reserved pins that are software programmable write/read input/output capability. These default to inputs upon power up, but may have their direction and output values defined in the EEPROM. The SDP bits may be mapped to the General Purpose Interrupt bits when configured as inputs. The SDP1[0] pin can be used as a watchdog output indication. All the SDP pins can be used as SFP sideband signals (TxDisable, present & TxFault). The 82580 does not generate these signals; it is available for SW control over SFP.
	SDP2_0 SDP2_1 SDP2_2 SDP2_3	M1 M2 M3 M4	T/S	SW Defined Pins for port 2: These pins are reserved pins that are software programmable write/read input/output capability. These default to inputs upon power up, but may have their direction and output values defined in the EEPROM. The SDP bits may be mapped to the General Purpose Interrupt bits when configured as inputs. The SDP2[0] pin can be used as a watchdog output indication. All the SDP pins can be used as SFP sideband signals (TxDisable, present & TxFault). The 82580 does not generate these signals; it is available for SW control over SFP.
	SDP3_0 SDP3_1 SDP3_2 SDP3_3	N1 N2 N3 N4	T/S	SW Defined Pins for port 3: These pins are reserved pins that are software programmable write/read input/output capability. These default to inputs upon power up, but may have their direction and output values defined in the EEPROM. The SDP bits may be mapped to the General Purpose Interrupt bits when configured as inputs. The SDP3[0] pin can be used as a watchdog output indication. All the SDP pins can be used as SFP sideband signals (TxDisable, present & TxFault). The 82580 does not generate these signals; it is available for SW control over SFP.
	TSENSP	D5	A-out	Thermal Sensor output; Can be used to measure the 82580 on-die temperature.
	TSENSZ	C5	GND	Thermal Sensor Ground.
	LAN_PWR_GOOD	D4	In	LAN Power Good: A 3.3v input signal. A transition from low to high initializes the device into operation. If the internal Power-on-Reset circuit is used to trigger device power-up, this signal should be connected to VCC3P3.

Table 2-6. Miscellaneous Pins

Reserved	Symbol	Ball #	Type	Name and Function
	MAIN_PWR_OK	B2	In	Main Power OK – Indicates that platform main power is up. Must be connected externally to main core 3.3V power.
	DEV_OFF_N	C4	In	Device Off: Assertion of DEV_OFF_N puts the device in Device Disable mode. This pin is asynchronous and is sampled once the EEPROM is ready to be read following power-up. The DEV_OFF_N pin should always be connected to VCC3P3 to enable device operation.
	XTAL1 XTAL2	P1 P2	A-In A-out	Reference Clock / XTAL: These pins may be driven by an external 25MHz crystal or driven by a single ended external CMOS compliant 25MHz oscillator.

2.1.6 SERDES/SGMII Pins

The AC specification for these pins is described in Section 11.6.3.

Table 2-7. SERDES/SGMII Pins

Reserved	Symbol	Ball #	Type	Name and Function
	SER0_p SER0_n	P16 P15	A-in	SERDES/SGMII Serial Data input Port 0: Differential SERDES Receive interface. A Serial differential input pair running at 1.25Gb/s. An embedded clock present in this input is recovered along with the data.
	SET0_p SET0_n	R16 R15	A-out	SERDES/SGMII Serial Data output Port 0: Differential SERDES Transmit interface. A serial differential output pair running at 1.25Gb/s. This output carries both data and an embedded 1.25GHz clock that is recovered along with data at the receiving end.
	SRDS_0_SIG_DET	N13	In	Port 0 Signal Detect: Indicates that signal (light) is detected from the Fiber. High for signal detect, Low otherwise. Polarity of Signal Detect pin is controlled by *CTRL.ILOS* bit. For non-fiber serdes applications link indication is internal and pin should be connected to a pull-up resistor.
	SER1_p SER1_n	M16 M15	A-in	SERDES/SGMII Serial Data input Port 1: Differential fiber SERDES Receive interface. A Serial differential input pair running at 1.25Gb/s. An embedded clock present in this input is recovered along with the data.
	SET1_p SET1_n	N16 N15	A-out	SERDES/SGMII Serial Data output Port 1: Differential fiber SERDES Transmit interface. A serial differential output pair running at 1.25Gb/s. This output carries both data and an embedded 1.25GHz clock that is recovered along with data at the receiving end.

Table 2-7. SERDES/SGMII Pins (Continued)

Reserved	Symbol	Ball #	Type	Name and Function
	SRDS_1_SIG_DET	P14	In	Port 1 Signal Detect: Indicates that signal (light) is detected from the fiber. High for signal detect, Low otherwise. Polarity of Signal Detect pin is controlled by *CTRL.ILOS* bit. For non-fiber serdes applications link indication is internal and pin should be connected to a pull-up resistor.
	SER2_p* SER2_n*	K16 K15	A-in	SERDES/SGMII Serial Data input Port 2: Differential SERDES Receive interface. A Serial differential input pair running at 1.25Gb/s. An embedded clock present in this input is recovered along with the data. * This port can be left unconnected in the dual port 82580DB.
	SET2_p* SET2_n*	L16 L15	A-out	SERDES/SGMII Serial Data output Port 2: Differential SERDES Transmit interface. A serial differential output pair running at 1.25Gb/s. This output carries both data and an embedded 1.25GHz clock that is recovered along with data at the receiving end. * This port can be left unconnected in the dual port 82580DB.
	SRDS_2_SIG_DET*	T15	In	Port 2 Signal Detect: Indicates that signal (light) is detected from the Fiber. High for signal detect, Low otherwise. Polarity of Signal Detect pin is controlled by *CTRL.ILOS* bit. For non-fiber serdes applications link indication is internal and pin should be connected to a pull-up resistor. * This port can be left unconnected in the dual port 82580DB.
	SER3_p* SER3_n*	H16 H15	A-in	SERDES/SGMII Serial Data input Port 3: Differential SERDES Receive interface. A Serial differential input pair running at 1.25Gb/s. An embedded clock present in this input is recovered along with the data. * This port can be left unconnected in the dual port 82580DB.

Table 2-7. SERDES/SGMII Pins (Continued)

Reserved	Symbol	Ball #	Type	Name and Function
	SET3_p* SET3_n*	J16 J15	A-out	SERDES/SGMII Serial Data output Port 3: Differential SERDES Transmit interface. A serial differential output pair running at 1.25Gb/s. This output carries both data and an embedded 1.25GHz clock that is recovered along with data at the receiving end. * This port can be left unconnected in the dual port 82580DB.
	SRDS_3_SIG_DET*	T16	In	Port 3 Signal Detect: Indicates that signal (light) is detected from the Fiber. High for signal detect, Low otherwise. Polarity of Signal Detect pin is controlled by *CTRL.ILOS* bit. For non-fiber serdes applications link indication is internal and pin should be connected to a pull-up resistor. * This port can be left unconnected in the dual port 82580DB.
	SE_RSET	K13	B	SerDes Bias Connect 2.37KΩ 1% resistor between pin and ground.

2.1.7 SFP Pins

The AC specification for these pins is described in Section 11.6.2.9.

Table 2-8. SFP Pins

Reserved	Symbol	Ball #	Type	Name and Function
	SFP0_I2C_CLK	M13	Out, O/D	Port 0 SFP 2 wire interface clock – connects to Mod-Def1 input of SFP (O/D). Can also be used as MDC pin (Out).
	SFP0_I2C_DATA	J13	T/S, O/D	Port 0 SFP 2 wire interface data – connects to Mod-Def2 pin of SFP (O/D). Can also be used as MDIO pin (T/S).
	SFP1_I2C_CLK	M14	Out, O/D	Port 1 SFP 2 wire interface clock – connects to Mod-Def1 input of SFP (O/D). Can also be used as MDC pin (Out).
	SFP1_I2C_DATA	N14	T/S, O/D	Port 1 SFP 2 wire interface data – connects to Mod-Def2 pin of SFP (O/D). Can also be used as MDIO pin (T/S).
	SFP2_I2C_CLK*	J14	Out, O/D	Port 2 SFP 2 wire interface clock – connects to Mod-Def1 input of SFP (O/D). Can also be used as MDC pin (Out). * This port can be left unconnected in the dual port 82580DB.

Table 2-8. SFP Pins (Continued)

Reserved	Symbol	Ball #	Type	Name and Function
	SFP2_I2C_DATA*	H13	T/S, O/D	Port 2 SFP 2 wire interface data – connects to Mod-Def2 pin of SFP (O/D). Can also be used as MDIO pin (T/S). * This port can be left unconnected in the dual port 82580DB.
	SFP3_I2C_CLK*	H14	Out, O/D	Port 3 SFP 2 wire interface clock – connects to Mod-Def1 input of SFP (O/D). Can also be used as MDC pin (Out). * This port can be left unconnected in the dual port 82580DB.
	SFP3_I2C_DATA*	G16	T/S, O/D	Port 3 SFP 2 wire interface data – connects to Mod-Def2 pin of SFP (O/D). Can also be used as MDIO pin (T/S). * This port can be left unconnected in the dual port 82580DB.

2.1.8 PHY Pins

2.1.8.1 LED's

The table below describes the functionality of the LED output pins. Default activity of the LED may be modified in the EEPROM word offsets 1Ch and 1Fh from start of relevant LAN Port section. The LED functionality is reflected and can be further modified in the configuration registers *LEDCTL*.

Table 2-9. LED Output Pins

Reserved	Symbol	Ball #	Type	Name and Function
	LED0_0	C1	Out	Port 0 LED0. Programmable LED which indicates by default Link Up. ***Note:*** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.17) or *LEDCTL* register (See Section 8.2.9).
	LED0_1	C2	Out	Port 0 LED1. Programmable LED which indicates by default activity (when packets are transmitted or received that match MAC filtering). ***Note:*** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.15) or *LEDCTL* register (See Section 8.2.9).
	LED0_2	C3	Out	Port 0 LED2. Programmable LED which indicates by default a 100Mbps Link. ***Note:*** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.17) or *LEDCTL* register (See Section 8.2.9).
	LED0_3	E4	Out	Port 0 LED3. Programmable LED which indicates by default a 1000Mbps Link. ***Note:*** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.15) or *LEDCTL* register (See Section 8.2.9).

Table 2-9. LED Output Pins (Continued)

Reserved	Symbol	Ball #	Type	Name and Function
	LED1_0	D1	Out	Port 1 LED0. Programmable LED which indicates by default Link up. **Note:** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.17) or *LEDCTL* register (See Section 8.2.9).
	LED1_1	D2	Out	Port 1 LED1. Programmable LED which indicates by default activity (when packets are transmitted or received that match MAC filtering). **Note:** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.15) or *LEDCTL* register (See Section 8.2.9).
	LED1_2	D3	Out	Port 1 LED2. Programmable LED which indicates by default a 100Mbps Link. **Note:** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.17) or *LEDCTL* register (See Section 8.2.9).
	LED1_3	F4	Out	Port 1 LED3. Programmable LED which indicates by default a 1000Mbps Link. **Note:** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.15) or *LEDCTL* register (See Section 8.2.9).
	LED2_0*	E1	Out	Port 2 LED0. Programmable LED which indicates by default Link up. **Note:** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.17) or *LEDCTL* register (See Section 8.2.9). * This port can be left unconnected in the dual port 82580DB.
	LED2_1*	E2	Out	Port 2 LED1. Programmable LED which indicates by default activity (when packets are transmitted or received that match MAC filtering). **Note:** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.15) or *LEDCTL* register (See Section 8.2.9). * This port can be left unconnected in the dual port 82580DB.
	LED2_2*	E3	Out	Port 2 LED2. Programmable LED which indicates by default a 100Mbps Link. **Note:** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.17) or *LEDCTL* register (See Section 8.2.9). * This port can be left unconnected in the dual port 82580DB.
	LED2_3*	G2	Out	Port 2 LED3. Programmable LED which indicates by default a 1000Mbps Link. **Note:** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.15) or *LEDCTL* register (See Section 8.2.9). * This port can be left unconnected in the dual port 82580DB.

Table 2-9. LED Output Pins (Continued)

Reserved	Symbol	Ball #	Type	Name and Function
	LED3_0*	F1	Out	Port 3 LED0. Programmable LED which indicates by default Link up. ***Note:*** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.17) or *LEDCTL* register (See Section 8.2.9). * This port can be left unconnected in the dual port 82580DB.
	LED3_1*	F2	Out	Port 3 LED1. Programmable LED which indicates by default activity (when packets are transmitted or received that match MAC filtering). ***Note:*** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.15) or *LEDCTL* register (See Section 8.2.9). * This port can be left unconnected in the dual port 82580DB.
	LED3_2*	F3	Out	Port 3 LED2. Programmable LED which indicates by default a 100Mbps Link. ***Note:*** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.17) or *LEDCTL* register (See Section 8.2.9). * This port can be left unconnected in the dual port 82580DB.
	LED3_3*	G1	Out	Port 3 LED3. Programmable LED which indicates by default a 1000Mbps Link. ***Note:*** Pin is active low by default, can be programmed via EEPROM (See Section 6.2.15) or *LEDCTL* register (See Section 8.2.9). * This port can be left unconnected in the dual port 82580DB.

2.1.8.2 PHY Analog Pins

The AC specification for these pins is described in sections Section 11.6.4.

Table 2-10. Analog Pins

Reserved	Symbol	Ball #	Type	Name and Function
	MDI0_0_p MDI0_0_n MDI1_0_p MDI1_0_n MDI2_0_p* MDI2_0_n* MDI3_0_p* MDI3_0_n*	T3 R3 T6 R6 T9 R9 T12 R12	A	Media Dependent Interface[0] for port 0, port 1, Port 2 and port 3 accordingly: • 1000BASE-T: In MDI configuration, MDI[0]+/- corresponds to BI_DA+/- and in MDIX configuration MDI[0]+/- corresponds to BI_DB+/-. • 100BASE-TX: In MDI configuration, MDI[0]+/- is used for the transmit pair and in MDIX configuration MDI[0]+/- is used for the receive pair. • 10BASE-T: In MDI configuration, MDI[0]+/- is used for the transmit pair and in MDIX configuration MDI[0]+/- is used for the receive pair. * This port can be left unconnected in the dual port 82580DB.
	MDI0_1_p MDI0_1_n MDI1_1_p MDI1_1_n MDI2_1_p* MDI2_1_n* MDI3_1_p* MDI3_1_n*	P4 P5 P6 P7 T10 R10 T13 R13	A	Media Dependent Interface[1] for port 0, port 1, port 2 and port 3 accordingly: • 1000BASE-T: In MDI configuration, MDI[1]+/- corresponds to BI_DB+/- and in MDIX configuration MDI[1]+/- corresponds to BI_DA+/-. • 100BASE-TX: In MDI configuration, MDI[1]+/- is used for the receive pair and in MDIX configuration MDI[1]+/- is used for the transmit pair. • 10BASE-T: In MDI configuration, MDI[1]+/- is used for the receive pair and in MDIX configuration MDI[1]+/- is used for the transmit pair. * This port can be left unconnected in the dual port 82580DB.
	MDI0_2_p MDI0_2_n MDI1_2_p MDI1_2_n MDI2_2_p* MDI2_2_n* MDI3_2_p* MDI3_2_n*	T4 R4 T7 R7 T11 R11 T14 R14	A	Media Dependent Interface[2] for port 0, port 1 port 2 and port 3: • 1000BASE-T: In MDI configuration, MDI[2]+/- corresponds to BI_DC+/- and in MDIX configuration MDI[2]+/- corresponds to BI_DD+/-. • 100BASE-TX: Unused. • 10BASE-T: Unused. * This port can be left unconnected in the dual port 82580DB.

Table 2-10. Analog Pins (Continued)

Reserved	Symbol	Ball #	Type	Name and Function
	MDI0_3_p MDI0_3_n MDI1_3_p MDI1_3_n MDI2_3_p* MDI2_3_n* MDI3_3_p* MDI3_3_n*	T5 R5 T8 R8 P9 P10 P12 P13	A	Media Dependent Interface[3] for port 0, port 1, port 2 and port 3: • 1000BASE-T: In MDI configuration, MDI[3]+/- corresponds to BI_DD+/- and in MDIX configuration MDI[3]+/- corresponds to BI_DC+/-. • 100BASE-TX: Unused. • 10BASE-T: Unused. *This port can be left unconnected in the dual port 82580DB.
	GE_REXT3K	T2	B	PHY Bias Connect 3.01KΩ 1% resistor between this pin and ground.
	RSVD_TX_TCLK	R1	Out	Transmit 1.25 MHz clock for IEEE testing. Shared for the 4 ports. Not connected in normal operation
	RSVD_ATST_P RSVD_ATST_N	R2 T1	A-out	Analog differential test pins. Shared for the 4 ports. Not connected in normal operation.

2.1.9 Testability Pins

Table 2-11. Testability Pins

Reserved	Symbol	Ball #	Type	Name and Function
	RSVD_TE_VSS	C12	In	Enables test mode. When high test pins are multiplexed on functional signals. In functional mode, must be connected to ground.
	RSVD_TP_5 RSVD_TP_6 RSVD_TP_7 RSVD_TP_8	C13 D12 G14 G15	T/S	Test pins for production testing. In functional mode should not be connected.
	JTCK	F13	In	JTAG Clock Input
	JTDI	E12	In	JTAG TDI Input
	JTDO	D13	T/S, O/D	JTAG TDO Output
	JTMS	G13	In	JTAG TMS Input
	RSRVD_JRST_3P3	E13	In	JTAG Reset Input
	AUX_PWR	D15	T/S	Auxiliary Power Available: This pin is a strapping option pin, latched at the rising edge of PE_RST# or In-Band PCIe Reset. This pin has an internal weak pull-up resistor. In case this pin is driven high during init time it indicates that Auxiliary Power is available and the device should support D3COLD power state if enabled to do so. This pin is also used for testing and scan.

Table 2-11. Testability Pins (Continued)

Reserved	Symbol	Ball #	Type	Name and Function
	LAN0_DIS_N	F14	T/S	This pin is a strapping option pin, latched at the rising edge of PE_RST# or In-Band PCIe Reset. This pin has an internal weak pull-up resistor. In case this pin is not connected or driven high during init time, LAN 0 is enabled. In case this pin is driven low during init time, LAN 0 function is disabled. This pin is also used for testing and scan.
	LAN1_DIS_N	E14	T/S	This pin is a strapping option pin, latched at the rising edge of PE_RST# or In-Band PCIe Reset. This pin has an internal weak pull-up resistor. In case this pin is not connected or driven high during init time, LAN 1 is enabled. In case this pin is driven low during init time, LAN 1 function is disabled. This pin is also used for testing and scan.
	LAN2_DIS_N *	D14	T/S	This pin is a strapping option pin, latched at the rising edge of PE_RST# or In-Band PCIe Reset. This pin has an internal weak pull-up resistor. In case this pin is not connected or driven high during init time, LAN 2 is enabled. In case this pin is driven low during init time, LAN 2 function is disabled. This pin is also used for testing and scan. * Ignored during INIT when using the dual port 82580DB.
	LAN3_DIS_N*	C14	T/S	This pin is a strapping option pin, latched at the rising edge of PE_RST# or In-Band PCIe Reset. This pin has an internal weak pull-up resistor. In case this pin is not connected or driven high during init time, LAN 3 is enabled. In case this pin is driven low during init time, LAN 3 function is disabled. This pin is also used for testing and scan. * Ignored during INIT when using the dual port 82580DB.

2.1.10 Power Supply and Ground Pins

Table 2-12. Power Supply Pins

Reserved	Symbol	Ball #	Type	Name and Function
	VCC3P3	K5	3.3V	3.3V Periphery power supply
	VCC3P3	F5, H5	3.3V	3.3V Periphery power supply
	VCC3P3	F12, H12, K12, L12	3.3V	3.3V Periphery power supply
	VCC1P0	E6,G6, H6, J6,E11, G11, H11, J11, K11, M11, N12	1.0V	1.0V digital power supply
	VCC1P0	K6	1.0V	1.0V digital power supply
	VCC1P0_APE	D6, D8, D9, D11	1.0V	1.0V PCIe Analog Power Supply
	VCC1P0_ASE	L13, K14, L14	1.0V	1.0V SerDes Analog power supply
	VCC1P0_AGE	L7, L8, L9, L10	1.0V	1.0V PHY analog power supply
	VCC3P3_A	M6, M7, M8, M9, M10, P8, P11	3.3V	3.3V PHY analog power supply
	VCC3P3_AGE	L5	3.3V	3.3V PHY analog power supply
	VCC1P8_PE_1	C7	1.8V	PCIe VCO Analog power supply connected to 1.8V.
	VCC1P8_PE_2	C10	1.8V	PCIe VCO Analog power supply connected to 1.8V.

Table 2-13. VSS Listings

Signal	Pin
VSS	A4, A7, A10, A13, B4, B7, B10, B13, D7, D10, E7, E8, E9, E10, F6, F7, F8, F9, F10, F11, G5, G7, G8, G9, G10, G12, H7, H8, H9, H10, J5, J7, J8, J9, J10, J12, K7, K8, K9, K10, L6, L11, M5, M12, N5, N6, N7, N8, N9, N10, N11, P3

2.2 Pullups/Pulldowns

The table below lists internal & external pull-up resistors and their functionality in different device states. Each internal PUP has a nominal value of 100KΩ, ranging from 50KΩ to 150KΩ

The device states are defined as follows:

- Power-up = while 3.3V is stable, yet 1.0V isn’t
- Active = normal mode (not power up or disable)
- Disable = device disable (a.k.a. dynamic IDDQ – see See Section 4.5)

Table 2-14. Pull-Up Resistors

Signal Name	Power up[5]		Active		Disable[6]		External
	PUP	Comments	PUP	Comments	PUP	Comments	
LAN_PWR_GOOD	N		N	Must be connected	N		PU
PE_WAKE_N	N		N		N		Y
PE_RST_N	N		N		N		N
FLSH_SI	Y		N		Y		N
FLSH_SO	Y		Y		Y		N
FLSH_SCK	Y		N		Y		N
FLSH_CE_N	Y		N		Y		N
EE_DI	Y		N		Y		N
EE_DO	Y		Y		Y		N
EE_SK	Y		N		Y		N
EE_CS_N	Y		N		Y		N
SMBD	N		N		N		Y
SMBCLK	N		N		N		Y
SMBALRT_N	N		N		N		Y
NCSI_CLK_IN	N	HiZ	N		N		PD (Note 1)
NCSI_CLK_OUT	N		N		N	If active, stable output	N
NCSI_CRS_DV	N	HiZ	N		N		Y (Note 2)
NCSI_RXD[1:0]	N	HiZ	N		N		Y (Note 2)
NCSI_TX_EN	N	HiZ	N		N		PD (Note 1)
NCSI_TXD[1:0]	N	HiZ	N		N		PD (Note 1)
SDP0[3:0]	Y		Y	Until EEPROM done	N	May keep state by EEPROM control	N
SDP1[3:0]	Y		Y	Until EEPROM done	N	May keep state by EEPROM control	N
SDP2[3:0]	Y		Y	Until EEPROM done	N	May keep state by EEPROM control	N
SDP3[3:0]	Y		Y	Until EEPROM done	N	May keep state by EEPROM control	N
DEV_OFF_N	Y		N		N		Must be connected on board
MAIN_PWR_OK	Y		Y		N		Must be connected on board
SRDS_0_SIG_DET	Y		N		N		Must be connected externally
SRDS_1_SIG_DET	Y		N		N		Must be connected externally
SRDS_2_SIG_DET	Y		N		N		Must be connected externally

Table 2-14. Pull-Up Resistors (Continued)

Signal Name	Power up[5]		Active		Disable[6]		External
	PUP	Comments	PUP	Comments	PUP	Comments	
SRDS_3_SIG_DET	Y		N		N		Must be connected externally
SFP0_I2C_CLK	Y		Y	Until EEPROM done or if *I2C disable* set in EEPROM	N		Y if I2C
SFP0_I2C_DATA	Y		Y	Until EEPROM done or if *I2C disable* set in EEPROM	N		Y
SFP1_I2C_CLK	Y		Y	Until EEPROM done or if *I2C disable* set in EEPROM	N		Y if I2C
SFP1_I2C_DATA	Y		Y	Until EEPROM done or if *I2C disable* set in EEPROM	N		Y
SFP2_I2C_CLK	Y		Y	Until EEPROM done or if *I2C disable* set in EEPROM	N		Y if I2C
SFP2_I2C_DATA	Y		Y	Until EEPROM done or if *I2C disable* set in EEPROM	N		Y
SFP3_I2C_CLK	Y		Y	Until EEPROM done or if *I2C disable* set in EEPROM	N		Y if I2C
SFP3_I2C_DATA	Y		Y	Until EEPROM done or if *I2C disable* set in EEPROM	N		Y
LED0_0	Y		N		N	HiZ	
LED0_1	Y		N		N	HiZ	
LED0_2	Y		N		N	HiZ	
LED0_3	Y		N		N	HiZ	
LED1_0	Y		N		N	HiZ	
LED1_1	Y		N		N	HiZ	
LED1_2	Y		N		N	HiZ	
LED1_3	Y		N		N	HiZ	
LED2_0	Y		N		N	HiZ	
LED2_1	Y		N		N	HiZ	
LED2_2	Y		N		N	HiZ	
LED2_3	Y		N		N	HiZ	
LED3_0	Y		N		N	HiZ	
LED3_1	Y		N		N	HiZ	
LED3_2	Y		N		N	HiZ	
LED3_3	Y		N		N	HiZ	
RSVD_TE_VSS	N		N		N		Connect to ground

Table 2-14. Pull-Up Resistors (Continued)

Signal Name	Power up[5]		Active		Disable[6]		External
	PUP	Comments	PUP	Comments	PUP	Comments	
RSVD_TP_8:5	Y		Y When input		Y		
JTCK	N		N		N		Y- Connect PD
JTDI	N		N	PU in Test mode	N		Y
JTDO	N		N		N		Y
JTMS	N		N	PU in Test mode	N		Y
RSRVD_JRST_3P3	N		N	PU in Test mode	N		Y- Connect PU
AUX_PWR	Y		N		N		PU or PD (note 3)
LAN0_DIS_N	Y		Y when input		Y		PU or PD (note 4)
LAN1_DIS_N	Y		Y when input		Y		PU or PD (note 4)
LAN2_DIS_N	Y		Y when input		Y		PU or PD (note 4)
LAN3_DIS_N	Y		Y when input		Y		PU or PD (note 4)

Notes:

1. Should be pulled down if NC-SI interface is disabled.
2. Only if NC-SI is unused or set to multi drop configuration.
3. If Aux power is connected, should be pulled up, else should be pulled down.
4. If the specific function is disabled, should be pulled down, else should be pulled up.
5. Power up - LAN_PWR_GOOD = 0
6. See Section 5.2.6 for description of Disable state.

2.3 Strapping

The following signals are used for static configuration. Unless otherwise stated, strapping options are latched on the rising edge of LAN_PWR_GOOD, at power up, at in-band PCI Express reset and at PE_RST_N assertion. At other times, they revert to their standard usage.

Table 2-15. Strapping Options

Purpose	Pin	Polarity	Pull-up / Pull-down
LAN0 Disable	LAN0_DIS_N	0b – LAN0 is disabled 1b – LAN0 is enabled	Internal pull-up
LAN1 Disable	LAN1_DIS_N	0b – LAN1 is disabled 1b – LAN1 is enabled	Internal pull-up
LAN2 Disable	LAN2_DIS_N	0b – LAN2 is disabled 1b – LAN2 is enabled	Internal pull-up
LAN3 Disable	LAN3_DIS_N	0b – LAN3 is disabled 1b – LAN3 is enabled	Internal pull-up
AUX_PWR	AUX_PWR	0b – AUX power is not available 1b – AUX power is available	None

2.4 Interface Diagram

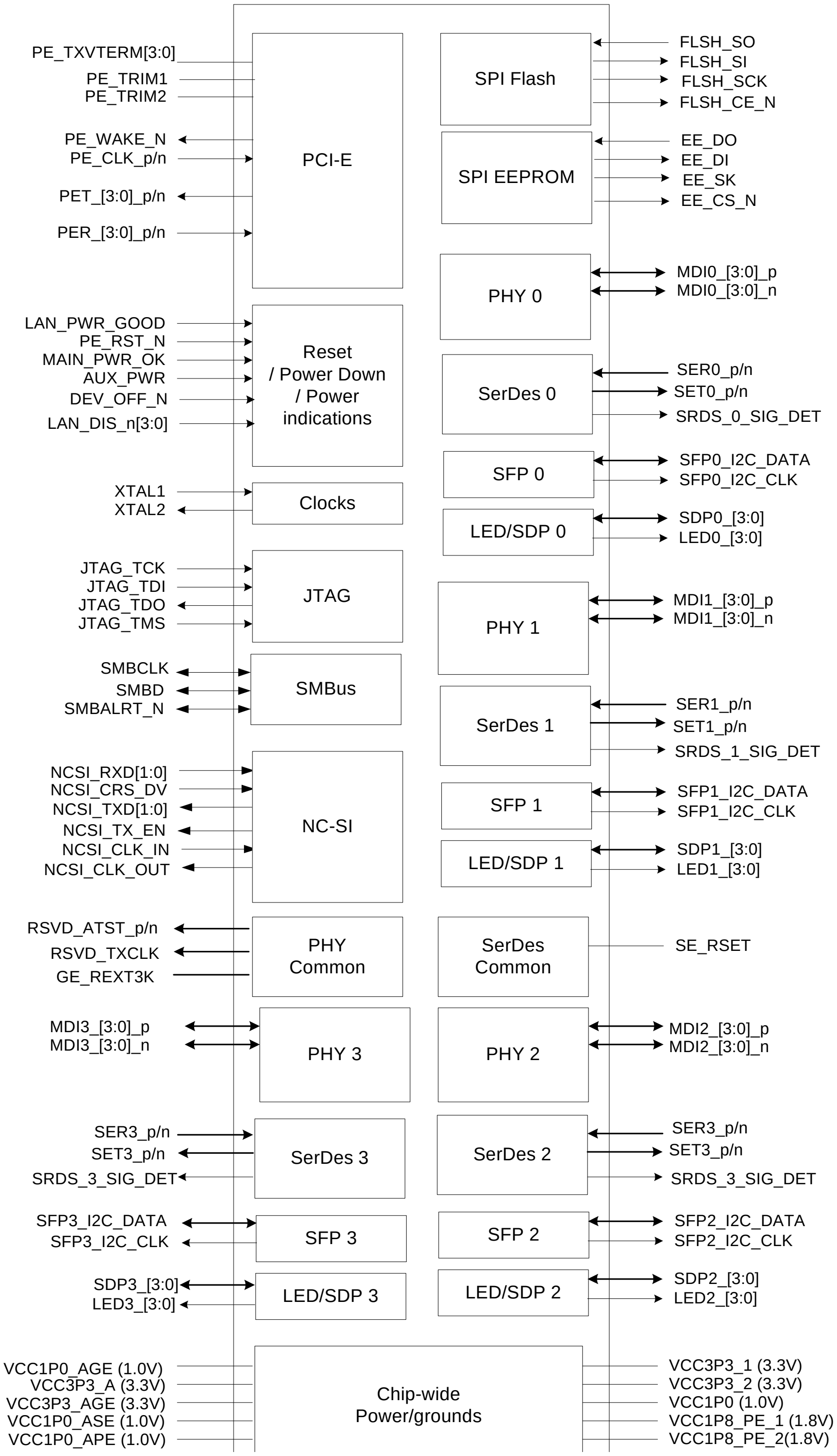


Figure 2-1. 82580 Interface Diagram

2.5 Pin List (Alphabetical)

Table 2-16 lists the pins and signals in ball alphabetical order.

Table 2-16. Pin List in Alphabetical Order

* NOTE: This port can be left unconnected in the dual port 82580DB.

Signal	Ball	Signal	Ball	Signal	Ball
PE_TRIM2	A1	TSENSZ	C5	VSS	E9
PE_TRIM1	A2	PE_TXVTERM1	C6	VSS	E10
PER_3_p	A3	VCC1P8_PE_1	C7	VCC1P0	E11
VSS	A4	PE_TXVTERM2	C8	JTDI	E12
PET_3_p	A5	PE_TXVTERM3	C9	RSVD_JRST_3P3	E13
PET_2_p	A6	VCC1P8_PE_2	C10	LAN1_DIS_N	E14
VSS	A7	PE_TXVTERM4	C11	EE_DI	E15
PER_2_p	A8	RSVD_TE_VSS	C12	EE_SK	E16
PER_1_p	A9	RSVD_TP_5	C13	LED3_0*	F1
VSS	A10	LAN3_DIS_N*	C14	LED3_1*	F2
PET_1_p	A11	FLSH_SO	C15	LED3_2*	F3
PET_0_p	A12	FLSH_CE_N	C16	LED1_3*	F4
VSS	A13	LED1_0	D1	VCC3P3	F5
PER_0_p	A14	LED1_1	D2	VSS	F6
PE_CLK_n	A15	LED1_2	D3	VSS	F7
PE_CLK_p	A16	LAN_PWR_GOOD	D4	VSS	F8
PE_RST_N	B1	TSENSP	D5	VSS	F9
MAIN_PWR_OK	B2	VCC1P0_APE	D6	VSS	F10
PER_3_n	B3	VSS	D7	VSS	F11
VSS	B4	VCC1P0_APE	D8	VCC3P3	F12
PET_3_n	B5	VCC1P0_APE	D9	JTCK	F13
PET_2_n	B6	VSS	D10	LAN0_DIS_N	F14
VSS	B7	VCC1P0_APE	D11	EE_DO	F15
PER_2_n	B8	RSVD_TP_6	D12	EE_CS_N	F16
PER_1_n	B9	JTDO	D13	LED3_3	G1
VSS	B10	LAN2_DIS_N*	D14	LED2_3	G2
PET_1_n	B11	AUX_PWR	D15	SMBD	G3
PET_0_n	B12	PE_WAKE_N	D16	SMBALRT_N	G4
VSS	B13	LED2_0*	E1	VSS	G5
PER_0_n	B14	LED2_1*	E2	VCC1P0	G6
FLSH_SI	B15	LED2_2*	E3	VSS	G7
FLSH_SCK	B16	LED0_3*	E4	VSS	G8
LED0_0	C1	SMBCLK	E5	VSS	G9
LED0_1	C2	VCC1P0	E6	VSS	G10
LED0_2	C3	VSS	E7	VCC1P0	G11
DEVICE_OFF_N	C4	VSS	E8	VSS	G12
JTMS	G13	VCC1P0	K6	SER1_n	M15

Table 2-16. Pin List in Alphabetical Order

* NOTE: This port can be left unconnected in the dual port 82580DB.

Signal	Ball	Signal	Ball	Signal	Ball
RSVD_TP_7	G14	VSS	K7	SER1_p	M16
RSVD_TP_8	G15	VSS	K8	SDP3_0	N1
SFP3_I2C_DATA	G16	VSS	K9	SDP3_1	N2
NCSI_CLK_IN	H1	VSS	K10	SDP3_2	N3
NCSI_CLK_OUT	H2	VCC1P0	K11	SDP3_3	N4
NCSI_CRS_DV	H3	VCC3P3	K12	VSS	N5
NCSI_RXD_0	H4	SE_RSET	K13	VSS	N6
VCC3P3	H5	VCC1P0_ASE	K14	VSS	N7
VCC1P0	H6	SER2_n*	K15	VSS	N8
VSS	H7	SER2_p*	K16	VSS	N9
VSS	H8	SDP1_0	L1	VSS	N10
VSS	H9	SDP1_1	L2	VSS	N11
VSS	H10	SDP1_2	L3	VCC1P0	N12
VCC1P0	H11	SDP1_3	L4	SRDS_0_SIG_DET	N13
VCC3P3	H12	VCC3P3_AGE	L5	SFP1_I2C_DATA	N14
SFP2_I2C_DATA*	H13	VSS	L6	SET1_n	N15
SFP3_I2C_CLK*	H14	VCC1P0_AGE	L7	SET1_p	N16
SER3_n*	H15	VCC1P0_AGE	L8	XTAL_CLK_I	P1
SER3_p*	H16	VCC1P0_AGE	L9	XTAL_CLK_O	P2
NCSI_RXD_1	J1	VCC1P0_AGE	L10	VSS	P3
NCSI_TX_EN	J2	VSS	L11	MDI0_1_p	P4
NCSI_TXD_0	J3	VCC3P3	L12	MDI0_1_n	P5
NCSI_TXD_1	J4	VCC1P0_ASE	L13	MDI1_1_p	P6
VSS	J5	VCC1P0_ASE	L14	MDI1_1_n	P7
VCC1P0	J6	SET2_n*	L15	VCC3P3_A	P8
VSS	J7	SET2_p*	L16	MDI2_3_p*	P9
VSS	J8	SDP2_0	M1	MDI2_3_n*	P10
VSS	J9	SDP2_1	M2	VCC3P3_A	P11
VSS	J10	SDP2_2	M3	MDI3_3_p*	P12
VCC1P0	J11	SDP2_3	M4	MDI3_3_n*	P13
VSS	J12	VSS	M5	SRDS_1_SIG_DET	P14
SFP0_I2C_DATA	J13	VCC3P3_A	M6	SER0_n	P15
SFP2_I2C_CLK	J14	VCC3P3_A	M7	SER0_p	P16
SET3_n*	J15	VCC3P3_A	M8	RSVD_TX_TCLK	R1
SET3_p*	J16	VCC3P3_A	M9	RSVD_ATST_P	R2
SDP0_0	K1	VCC3P3_A	M10	MDI0_0_n	R3
SDP0_1	K2	VCC1P0	M11	MDI0_2_n	R4
SDP0_2	K3	VSS	M12	MDI0_3_n	R5
SDP0_3	K4	SFP0_I2C_CLK	M13	MDI1_0_n	R6
VCC3P3	K5	SFP1_I2C_CLK	M14	MDI1_2_n	R7
MDI1_3_n	R8	RSVD_ATST_N	T1	MDI2_1_p*	T10
MDI2_0_n*	R9	GE_REXT3K	T2	MDI2_2_p*	T11

Table 2-16. Pin List in Alphabetical Order

* NOTE: This port can be left unconnected in the dual port 82580DB.

Signal	Ball	Signal	Ball	Signal	Ball
MDI2_1_n*	R10	MDI0_0_p	T3	MDI3_0_p*	T12
MDI2_2_n*	R11	MDI0_2_p	T4	MDI3_1_p*	T13
MDI3_0_n*	R12	MDI0_3_p	T5	MDI3_2_p*	T14
MDI3_1_n*	R13	MDI1_0_p	T6	SRDS_2_SIG_DET*	T15
MDI3_2_n*	R14	MDI1_2_p	T7	SRDS_3_SIG_DET*	T16
SET0_n	R15	MDI1_3_p	T8		
SET0_p	R16	MDI2_0_p*	T9		

2.6 Ball-Out

Figure 2-2 depicts a top view ball map of the 82580, in a 17x17 PBGA package. See Section 11.7.3 for locating A1 corner ball on package.

*	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A	PE_TRIM2	PE_TRIM1	PER_3_p	VSS	PET_3_p	PET_2_p	VSS	PER_2_p	PER_1_p	VSS	PET_1_p	PET_0_p	VSS	PER_0_p	PE_CLK_n	PE_CLK_p
B	PE_RST_N	MAIN_PWR_OK	PER_3_n	VSS	PET_3_n	PET_2_n	VSS	PER_2_n	PER_1_n	VSS	PET_1_n	PET_0_n	VSS	PER_0_n	FLSH_SI	FLSH_SCK
C	LED0_0	LED0_1	LED0_2	DEVICE_OFF_N	TSENSZ	PE_TXVTERM1	VCC1P8_PE_1	PE_TXVTERM2	PE_TXVTERM3	VCC1P8_PE_2	PE_TXVTERM4	RSVD_TE_VSS	RSVD_TP_5	LAN3_DIS_N	FLSH_SO	FLSH_CE_N
D	LED1_0	LED1_1	LED1_2	LAN_PWR_GOOD	TSENSP	VCC1P0_APE	VSS	VCC1P0_APE	VCC1P0_APE	VSS	VCC1P0_APE	RSVD_TP_6	JTDO	LAN2_DIS_N	AUX_PWR	PE_WAKE_N
E	LED2_0	LED2_1	LED2_2	LED0_3	SMBCLK	VCC1P0	VSS	VSS	VSS	VSS	VCC1P0	JTDI	RSVD_JRST_3P3	LAN1_DIS_N	EE_DI	EE_SK
F	LED3_0	LED3_1	LED3_2	LED1_3	VCC3P3	VSS	VSS	VSS	VSS	VSS	VSS	VCC3P3	JTCK	LAN0_DIS_N	EE_DO	EE_CS_N
G	LED3_3	LED2_3	SMBD	SMBALRT_N	VSS	VCC1P0	VSS	VSS	VSS	VSS	VCC1P0	VSS	JTMS	RSVD_TP_7	RSVD_TP_8	SFP3_I2C_DATA
H	NCSI_CLK_IN	NCSI_CLK_OUT	NCSI_CRS_DV	NCSI_RXD_0	VCC3P3	VCC1P0	VSS	VSS	VSS	VSS	VCC1P0	VCC3P3	SFP2_I2C_DATA	SFP3_I2C_CLK	SER3_n	SER3_p
J	NCSI_RXD_1	NCSI_TX_EN	NCSI_TXD_0	NCSI_TXD_1	VSS	VCC1P0	VSS	VSS	VSS	VSS	VCC1P0	VSS	SFP0_I2C_DATA	SFP2_I2C_CLK	SETn_3	SET3_p
K	SDP0_0	SDP0_1	SDP0_2	SDP0_3	VCC3P3	VCC1P0	VSS	VSS	VSS	VSS	VCC1P0	VCC3P3	SE_RSET	VCC1P0_ASE	SER2_n	SER2_p
L	SDP1_0	SDP1_1	SDP1_2	SDP1_3	VCC3P3_AGE	VSS	VCC1P0_AGE	VCC1P0_AGE	VCC1P0_AGE	VCC1P0_AGE	VSS	VCC3P3	VCC1P0_ASE	VCC1P0_ASE	SET2_n	SET2_p
M	SDP2_0	SDP2_1	SDP2_2	SDP2_3	VSS	VCC3P3_A	VCC3P3_A	VCC3P3_A	VCC3P3_A	VCC3P3_A	VCC1P0	VSS	SFP0_I2C_CLK	SFP1_I2C_CLK	SER1_n	SER1_p
N	SDP3_0	SDP3_1	SDP3_2	SDP3_3	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VCC1P0	SRDS_0_SIG_DET	SFP1_I2C_DATA	SET1_n	SET1_p
P	XTAL_CLK_I	XTAL_CLK_O	VSS	MDI0_1_p	MDI0_1_n	MDI1_1_p	MDI1_1_n	VCC3P3_A	MDI2_3_p	MDI2_3_n	VCC3P3_A	MDI3_3_p	MDI3_3_n	SRDS_1_SIG_DET	SER0_n	SER0_p
R	RSVD_TX_TCLK	RSVD_ATST_P	MDI0_0_n	MDI0_2_n	MDI0_3_n	MDI1_0_n	MDI1_2_n	MDI1_3_n	MDI2_0_n	MDI2_1_n	MDI2_2_n	MDI3_0_n	MDI3_1_n	MDI3_2_n	SET0_n	SET0_p
T	RSVD_ATST_N	GE_REXT3K	MDI0_0_p	MDI0_2_p	MDI0_3_p	MDI1_0_p	MDI1_2_p	MDI1_3_p	MDI2_0_p	MDI2_1_p	MDI2_2_p	MDI3_0_p	MDI3_1_p	MDI3_2_p	SRDS_2_SIG_DET	SRDS_3_SIG_DET

Figure 2-2. Ball-out

§ §

3.0 Interconnects

3.1 PCIe

3.1.1 PCIe Overview

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

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

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

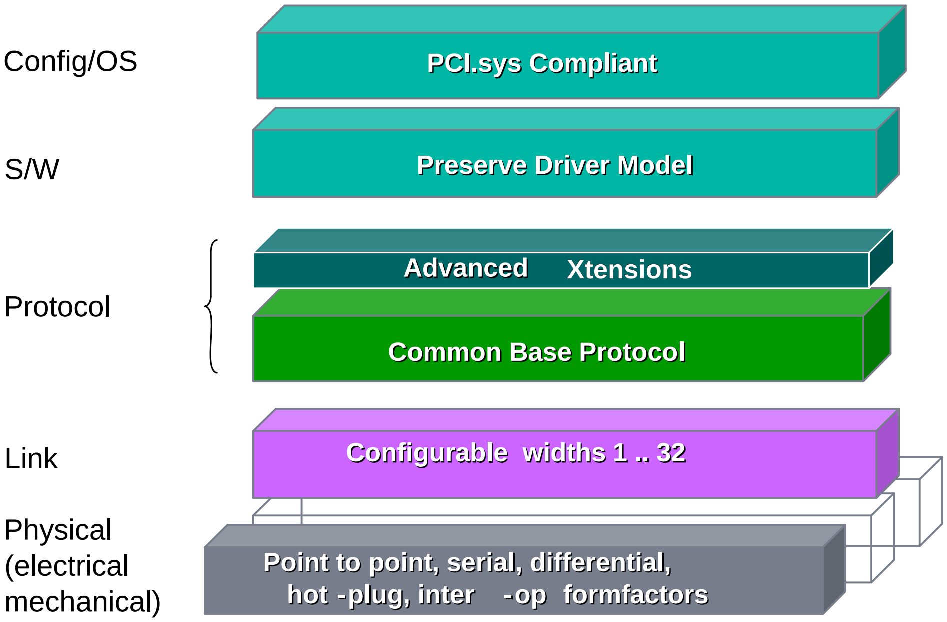

Config/OS
PCI.sys Compliant
S/W
Preserve Driver Model
Protocol
Advanced Xtensions
Common Base Protocol
Link
Configurable widths 1 .. 32
Physical (electrical mechanical)
Point to point, serial, differential,
hot -plug, inter -op formfactors

Figure 3-1. PCIe Stack Structure

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

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

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

3.1.1.1 Architecture, Transaction and Link Layer Properties

- Split transaction, packet-based protocol
- Common flat address space for load/store access (such as PCI addressing model)
  - Memory address space of 32-bits to allow compact packet header (must be used to access addresses below 4 GB)
  - Memory address space of 64-bit using extended packet header
- Transaction layer mechanisms:
  - PCI-X style relaxed ordering
  - Optimizations for no-snoop transactions
- Credit-based flow control
- Packet sizes/formats:
  - Maximum upstream (write) payload size of 512 Bytes
  - Maximum downstream (read) payload size of 2 KBytes
- Reset/initialization:
  - Frequency/width/profile negotiation performed by hardware
- Data integrity support
  - Using CRC-32 for transaction layer packets
- Link layer retry for recovery following error detection
  - Using CRC-16 for link layer messages
- No retry following error detection
  - 8b/10b encoding with running disparity
- Software configuration mechanism:
  - Uses PCI configuration and bus enumeration model
  - PCIe-specific configuration registers mapped via PCI extended capability mechanism
- Baseline messaging:
  - In-band messaging of formerly side-band legacy signals (such as interrupts, etc.)
  - System-level power management supported via messages
- Power management:
  - Full support for PCI-PM
  - Wake capability from D3cold state
  - Compliant with ACPI, PCI-PM software model

  - Active state power management
- Support for PCIe rev 2.0
  - Support for completion time out
  - Support for additional registers in the PCIe capability structure.

3.1.1.2 Physical Interface Properties

- Point to point interconnect
  - Full-duplex; no arbitration
- Signaling technology:
  - Low Voltage Differential (LVD)
  - Embedded clock signaling using 8b/10b encoding scheme
- Serial frequency of operation: 5 Gbps (Gen2) or 2.5Gbps (Gen1).
- Interface width of x4, x2, or x1.
- DFT and DFM support for high volume manufacturing

3.1.1.3 Advanced Extensions

PCIe defines a set of optional features to enhance platform capabilities for specific usage modes. The 82580 supports the following optional features:

- Extended error reporting - messaging support to communicate multiple types/severity of errors.
- Device serial number.
- Completion timeout control.
- TLP Processing Hints (TPH) - provides hints on a per transaction basis to facilitate optimized processing of transactions that target Memory Space.
- Latency Tolerance Reporting (LTR) - messaging support to communicate service latency requirements for Memory Reads and Writes to the Root Complex.

3.1.2 Functionality - General

3.1.2.1 Native/Legacy

All the 82580 PCI functions are native PCIe functions.

3.1.2.2 Locked Transactions

The 82580 does not support locked requests as target or master.

3.1.3 Host Interface

3.1.3.1 Tag IDs

PCIe device numbers identify logical devices within the physical device (the 82580 is a physical device). The 82580 implements a single logical device with up to four separate PCI functions: LAN 0, LAN 1, LAN2 and LAN3. The device number is captured from each type 0 configuration write transaction.

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

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

3.1.3.1.1 TAG ID Allocation for Read Transactions

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

Table 3-1. IDs in Read Transactions

Tag ID	Description	Comment
0x0	Data request 0	
0x1	Data request 1	
0x2	Data request 2	
0x3	Data request 3	
0x4	Data request 4	
0x5	Data request 5	
0x6	Data request 6	
0x7	Data request 7	
0x8	Data request 8	
0x9	Data request 9	
0xA	Data request 10	
0xB	Data request 11	
0xC	Data request 12	
0xD	Data request 13	
0xE	Data request 14	
0xF	Data request 15	
0x10	Data request 16	
0x11	Data request 17	
0x12	Data request 18	
0x13	Data request 19	
0x14	Data request 20	
0x15	Data request 21	
0x16	Data request 22	
0x17	Data request 23	
0x18	Descriptor Tx 0	
0x19	Descriptor Tx 1	
0x1A	Descriptor Tx 2	
0x1B	Descriptor Tx 3	
0x1C	Descriptor Rx 0	
0x1D	Descriptor Rx 1	
0x1E	Descriptor Rx 2	
0x1F	Descriptor Rx 3	

3.1.3.1.2 TAG ID Allocation for Write Transactions

Request tag allocation depends on these system parameters:

- DCA supported/not supported in the system (*DCA_CTRL.DCA_DIS* - see Section 8.13.4 for details)
- TPH enabled in the system.
- DCA enabled/disabled for each type of traffic (*TXCTL.TX Descriptor DCA EN, RXCTL.RX Descriptor DCA EN, RXCTL.RX Header DCA EN, RXCTL.Rx Payload DCA EN*).
- TPH enabled or disabled for the specific type of traffic carried by the TLP (*TXCTL.TX Descriptor TPH EN, RXCTL.RX Descriptor TPH EN, RXCTL.RX Header TPH EN, RXCTL.Rx Payload TPH EN*).
- System type: Legacy DCA vs. DCA 1.0 (*DCA_CTRL.DCA_MODE* - see Section 8.13.4 for details).
- CPU ID (*RXCTL.CPUID* or *TXCTL.CPUID*).

I/OAT 1I/OAT 2/3

3.1.3.1.2.1 Case 1 - DCA Disabled in the System:

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

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

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

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

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

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

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

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

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

- The request tag (all 8 bits) is taken from the CPUID field of the adequate register or context as described in Table 7-58.

3.1.3.2 Completion Timeout Mechanism

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

The completion timeout mechanism is activated for each request that requires one or more completions when the request is transmitted. The 82580 provides a programmable range for the completion timeout, as well as the ability to disable the completion timeout altogether. The completion timeout is programmed through an extension of the PCIe capability structure (See Section 9.5.6.12).

The 82580's reaction in case of a completion timeout is defined in Table 3-12.

The 82580 controls the following aspects of completion timeout:

- Disabling or enabling completion timeout.
- Disabling or enabling re-send of a request on completion timeout.
- A programmable range of re-sends on completion timeout, if re-send enabled.
- A programmable range of timeout values.
- Programming the behavior of completion timeout is summarized in Table 3-3. System Software may configure completion timeout independently per each LAN function.

Table 3-3. Completion Timeout Programming

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

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

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

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

3.1.3.2.1 Completion Timeout Period

Programmed through the *PCI Device Control 2* configuration register (See Section 9.5.6.12). The 82580 supports all ranges defined by PCIe v2.0 (5Gbps and 2.5Gbps).

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

Note: The completion timeout value must be programmed correctly in PCIe configuration space (in Device Control 2 Register); the value must be set above the expected maximum latency for completions in the system in which the 82580 is installed. This will ensure that the 82580 receives the completions for the requests it sends out, avoiding a completion timeout scenario. It is expected that the system BIOS will set this value appropriately for the system.

3.1.4 Transaction Layer

The upper layer of the PCIe architecture is the transaction Layer. The transaction layer connects to the 82580 core using an implementation specific protocol. Through this core-to-transaction-layer protocol, the application-specific parts of the 82580 interact with the PCIe subsystem and transmit and receive requests to or from the remote PCIe agent, respectively.

3.1.4.1 Transaction Types Accepted by the 82580

Table 3-4. 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
IO Read Request	NPH	CPLH + CPLD	Requester ID, TAG, Attribute	CSR
IO Write Request	NPH + NPD	CPLH	Requester ID, TAG, Attribute	CSR
Read completions	CPLH + CPLD	-	-	DMA
Message	PH	-	-	Message Unit / PM

Flow control types:

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

3.1.4.1.1 Configuration Request Retry Status

PCIe supports devices requiring a lengthy self-initialization sequence to complete before they are able to service configuration requests. This is the case for the 82580 were initialization is long due to the EEPROM read operation following reset.

If the read of the PCIe section in the EEPROM was not completed and the 82580 receives a configuration request, the 82580 responds with a configuration request retry completion status to terminate the request, and thus effectively stall the configuration request until the subsystem has completed local initialization and is ready to communicate with the host.

3.1.4.1.2 Partial Memory Read and Write Requests

The 82580 has limited support of read and write requests when only part of the byte enable bits are set as described later in this section.

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

Zero-length writes have no internal impact (nothing written, no effect such as clear-by-write). The transaction is treated as a successful operation (no error event).

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

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

3.1.4.2 Transaction Types Initiated by the 82580

Table 3-5. Transaction Types Initiated by the Transaction Layer

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

Note: MAX_PAYLOAD_SIZE supported is loaded from EEPROM (128 bytes, 256 bytes or 512 bytes). Effective MAX_PAYLOAD_SIZE is defined for each PCI function according to configuration space register of this function.

3.1.4.2.1 Data Alignment

Requests must never specify an address/length combination that causes a memory space access to cross a 4 KB boundary. The 82580 breaks requests into 4 KB-aligned requests (if needed). This does not pose any requirement on software. However, if software allocates a buffer across a 4 KB boundary, hardware issues multiple requests for the buffer. Software should consider limiting buffer sizes and base addresses to comply with a 4 KB boundary in cases where it improves performance.

The general rules for packet alignment are as follows:

1. The length of a single request should not exceed the PCIe limit of MAX_PAYLOAD_SIZE for write and MAX_READ_REQ for read.
2. The length of a single request does not exceed the 82580's internal limitation.
3. A single request should not span across different memory pages as noted by the 4 KB boundary previously mentioned.

Note: The rules apply to all the 82580 requests (read/write, snoop and no snoop).

If a request can be sent as a single PCIe packet and still meet rules 1-3, then it is not broken at a cache-line boundary (as defined in the PCIe Cache line size configuration word), but rather, sent as a single packet (motivation is that the chipset might break the request along cache-line boundaries, but the 82580 should still benefit from better PCIe utilization). However, if rules 1-3 require that the request is broken into two or more packets, then the request is broken at a cache-line boundary.

3.1.4.2.2 Multiple Tx Data Read Requests (MULR)

The 82580 supports 24 pipelined requests for transmit data on all ports. In general, the 24 requests might belong to the same packet or to consecutive packets to be transmitted on a single LAN port or on multiple LAN ports. However, the following restriction applies:

- All requests for a packet are issued before a request is issued for a consecutive packet

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

The PCIe specification does not ensure that completions for separate requests return in-order. Read completions for concurrent requests are not required to return in the order issued. The 82580 handles completions that arrive in any order. Once all completions arrive for a given request, the 82580 might issue the next pending read data request.

- The 82580 incorporates a re-order buffer to support re-ordering of completions for all requests. Each request/completion can be up to 2 KBytes long. The maximum size of a read request is defined as the minimum {2KB, Max_Read_Request_Size}.

In addition to the 24 pipeline requests for transmit data, the 82580 can issue up to 4 read requests for all ports (either for a single port or for multiple LAN ports) to fetch transmit descriptors and 4 read requests for all ports (either for a single LAN port or for multiple LAN ports) to fetch receive descriptors. The requests for transmit data, transmit descriptors, and receive descriptors are independently issued. Each descriptor read request can fetch up to 16 descriptors for reception and 24 descriptors for transmission.

3.1.4.3 Messages

3.1.4.3.1 Message Handling by the 82580 (as a Receiver)

Message packets are special packets that carry a message code.

The upstream device transmits special messages to the 82580 by using this mechanism.

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

Table 3-6. Supported Messages in the 82580 (as a Receiver)

Message code [7:0]	Routing r2r1r0	Message	Device later response
0x14	100	PM_Active_State_NAK	Internal signal set
0x19	011	PME_Turn_Off	Internal signal set
0x50	100	Slot power limit support (has one Dword data)	Silently drop
0x7E	010,011,100	Vendor_defined type 0 no data	Unsupported request -- NEC*

Table 3-6. Supported Messages in the 82580 (as a Receiver) (Continued)

Message code [7:0]	Routing r2r1r0	Message	Device later response
0x7E	010,011,100	Vendor_defined type 0 data	Unsupported request -- NEC*
0x7F	010,011,100	Vendor_defined type 1 no data	Silently drop
0x7F	010,011,100	Vendor_defined type 1 data	Silently drop
0x00	011	Unlock	Silently drop

3.1.4.3.2 Message Handling by the 82580 (as a Transmitter)

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

Table 3-7. Supported Message in the 82580 (as a Transmitter)

Message code [7:0]	Routing r2r1r0	Message
0x20	100	Assert INT A
0x21	100	Assert INT B
0x22	100	Assert INT C
0x23	100	Assert INT D
0x24	100	De-assert INT A
0x25	100	De-assert INT B
0x26	100	De-assert INT C
0x27	100	De-Assert INT D
0x30	000	ERR_COR
0x31	000	ERR_NONFATAL
0x33	000	ERR_FATAL
0x18	000	PM_PME
0x1B	101	PME_TO_ACK
0x10	100	Latency Tolerance Reporting (LTR)

3.1.4.4 Ordering Rules

The 82580 meets the PCIe ordering rules (PCI-X rules) by following the PCI simple device model:

- Deadlock avoidance - Master and target accesses are independent - The response to a target access does not depend on the status of a master request to the bus. If master requests are blocked, such as due to no credits, target completions might still proceed (if credits are available).
- Descriptor/data ordering - The 82580 does not proceed with some internal actions until respective data writes have ended on the PCIe link:
  - — The 82580 does not update an internal header pointer until the descriptors that the header pointer relates to are written to the PCIe link.
  - — The 82580 does not issue a descriptor write until the data that the descriptor relates to is written to the PCIe link.

The 82580 might issue the following master read request from each of the following clients:

- Rx Descriptor Read (up to 4 for all LAN ports)
- Tx Descriptor Read (up to 4 for all LAN ports)
- Tx Data Read (up to 24 for all LAN ports)

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

3.1.4.4.1 Out of Order Completion Handling

In a split transaction protocol, when using multiple read requests in a multi processor environment, there is a risk that completions arrive from the host memory out of order and interleaved. In this case, the 82580 sorts the request completions and transfers them to the Ethernet in the correct order.

3.1.4.5 Transaction Definition and Attributes

3.1.4.5.1 Max Payload Size

The 82580 policy to determine Max Payload Size (MPS) is as follows:

- Master requests initiated by the 82580 (including completions) limits MPS to the value defined for the function issuing the request.
- Target write accesses to the 82580 are accepted only with a size of one Dword or two Dwords. Write accesses in the range of (three Dwords, MPS, etc.) are flagged as UR. Write accesses above MPS are flagged as malformed.

See section 2.2.2 - TLPs with Data Payloads - Rules of the PCIe base specification.

3.1.4.5.2 Relaxed Ordering

The 82580 takes advantage of the relaxed ordering rules in PCIe. By setting the relaxed ordering bit in the packet header, the 82580 enables the system to optimize performance in the following cases:

- Relaxed ordering for descriptor and data reads: When the 82580 emits a read transaction, its split completion has no ordering relationship with the writes from the CPUs (same direction). It should be allowed to bypass the writes from the CPUs.
- Relaxed ordering for receiving data writes: When the 82580 issues receive DMA data writes, it also enables them to bypass each other in the path to system memory because software does not process this data until their associated descriptor writes complete.
- The 82580 cannot relax ordering for descriptor writes, MSI/MSI-X writes or PCIe messages.

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

Relaxed ordering is enabled in the 82580 by clearing the *RO_DIS* bit in the CTRL_EXT register. Actual setting of relaxed ordering is done for LAN traffic by the host through the DCA registers.

3.1.4.5.3 Snoop Not Required

The 82580 sets the *Snoop Not Required* attribute bit for master data writes. System logic might provide a separate path into system memory for non-coherent traffic. The non-coherent path to system memory provides higher, more uniform, bandwidth for write requests.

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

Global no-snoop support is enabled in the 82580 by clearing the *NS_DIS* bit in the *CTRL_EXT* register. Actual setting of no snoop is done for LAN traffic by the host through the DCA registers.

3.1.4.5.4 No Snoop and Relaxed Ordering for LAN Traffic

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

Table 3-8 lists the default behavior for the *No-Snoop* and *Relaxed Ordering* bits for LAN traffic when I/OAT 2 is enabled.

Table 3-8. LAN Traffic Attributes

Transaction	No-Snoop Default	Relaxed Ordering Default	Comments
Rx Descriptor Read	N	Y	
Rx Descriptor Write-Back	N	N	Relaxed ordering must never be used for this traffic.
Rx Data Write	Y	Y	See the following note and Section 3.1.4.5.4.1
Rx Replicated Header	N	Y	
Tx Descriptor Read	N	Y	
Tx Descriptor Write-Back	N	Y	
Tx TSO Header Read	N	Y	
Tx Data Read	N	Y	

Note: Rx payload no-snoop is also conditioned by the *NSE* bit in the receive descriptor. See Section 3.1.4.5.4.1.

3.1.4.5.4.1 No-Snoop Option for Payload

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

In this case, software should be able to set a bit in the receive descriptor indicating that the 82580 should perform a no-snoop DMA transfer when it eventually writes a packet to this buffer.

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

This is triggered by the *NSE* bit in the receive descriptor. See Section 7.1.5.

3.1.4.5.5 TLP processing Hint (TPH)

The TPH bit can be set to provide information to the root complex about the cache in which the data should be stored or from which the data should be read as described in Section 7.7.2.

TPH is enabled via the TPH Requester Enable field in the TPH control register of the configuration space (Section 9.6.3.3). Setting of the TPH bit for different type of traffic is described in Table 7-58.

3.1.4.6 Flow Control

3.1.4.6.1 82580 Flow Control Rules

The 82580 implements only the default Virtual Channel (VC0). A single set of credits is maintained for VC0.

Table 3-9. Allocation of FC Credits

Credit Type	Operations	Number Of Credits
Posted Request Header (PH)	Target Write (one unit) Message (one unit)	Four units (to enable concurrent accesses to all LAN ports).
Posted Request Data (PD)	Target Write (Length/16 bytes=1) Message (one unit)	MAX_PAYLOAD_SIZE/16
Non-Posted Request Header (NPH)	Target Read (one unit) Configuration Read (one unit) Configuration Write (one unit)	Four units (to enable concurrent target accesses to all LAN ports).
Non-Posted Request Data (NPD)	Configuration Write (one unit)	Four units.
Completion Header (CPLH)	Read Completion (N/A)	Infinite (accepted immediately).
Completion Data (CPLD)	Read Completion (N/A)	Infinite (accepted immediately).

Rules for FC updates:

- The 82580 maintains four credits for NPD at any given time. It increments the credit by one after the credit is consumed and sends an UpdateFC packet as soon as possible. UpdateFC packets are scheduled immediately after a resource is available.
- The 82580 provides four credits for PH (such as for four concurrent target writes) and four credits for NPH (such as for four concurrent target reads). UpdateFC packets are scheduled immediately after a resource becomes available.
- The 82580 follows the PCIe recommendations for frequency of UpdateFC FCPs.

3.1.4.6.2 Upstream Flow Control Tracking

The 82580 issues a master transaction only when the required FC credits are available. Credits are tracked for posted, non-posted, and completions (the later to operate with a switch).

3.1.4.6.3 Flow Control Update Frequency

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

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

3.1.4.6.4 Flow Control Timeout Mechanism

The 82580 implements the optional FC update timeout mechanism.

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

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

3.1.4.7 Error Forwarding

If a TLP is received with an error-forwarding trailer (Poisoned TLP received), the transaction may either be resent or dropped and not delivered to its destination, depending on the *GCR.Completion Timeout resend enable* bit and the *GCR.Number of resends* field. If the re-sends were unsuccessful or if re-send is disabled, The 82580 does not initiate any additional master requests for that PCI function until it detects an internal reset or a software reset for the associated LAN. Software is able to access device registers after such a fault.

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

3.1.5 Data Link Layer

3.1.5.1 ACK/NAK Scheme

The 82580 will send ACK/NAK immediately in the following cases:

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

In all other cases The 82580 will schedule ACK transmission according to time-outs specified in the PCIe specification (Depends on link speed, link width, and max_payload_size).

3.1.5.2 Supported DLLPs

The following DLLPs are supported by the 82580 as a receiver:

Table 3-10. DLLPs Received by the 82580

DLLP type	Remarks
Ack	
Nak	
PM_Request_Ack	
InitFC1-P	Virtual Channel 0 only
InitFC1-NP	Virtual Channel 0 only
InitFC1-Cpl	Virtual Channel 0 only
InitFC2-P	Virtual Channel 0 only
InitFC2-NP	Virtual Channel 0 only
InitFC2-Cpl	Virtual Channel 0 only
UpdateFC-P	Virtual Channel 0 only
UpdateFC-NP	Virtual Channel 0 only
UpdateFC-Cpl	Virtual Channel 0 only

The following DLLPs are supported by the 82580 as a transmitter:

Table 3-11. DLLPs Initiated by the 82580

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

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

3.1.5.3 Transmit EDB Nullifying

In case of a necessity to re-train, there is a need to guarantee that no abrupt termination of the Tx packet happens. For this reason, early termination of the transmitted packet is possible. This is done by appending an EDB (EnD Bad symbol) to the packet.

3.1.6 Physical Layer

3.1.6.1 Link Speed

The 82580 supports 2.5GT/s and 5GT/s link speeds. The following PCIe configuration bits define the link speed:

- *Max Link Speed* bit in the *Link CAP* register — Indicates the link speed supported by The 82580 as determined by the *Disable PCIe Gen 2* bit in the *PCIe PHY Auto Configuration* EEPROM section.
- *Link Speed* bit in the *Link Status register* — Indicates the negotiated Link speed.
- *Target Link Speed* bit in the *Link Control 2* register — used to set the target compliance mode speed when software is using the Enter Compliance bit to force a link into compliance mode. The default value is determined by the *Disable PCIe Gen 2* bit in the *PCIe PHY Auto Configuration* EEPROM section.

The 82580 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 82580 supports entering compliance mode at the speed indicated in the *Target Link Speed* field in the *PCIe Link Control 2* register. Compliance mode functionality is controlled via the *Enter Compliance* bit in the *PCIe Link Control 2* register.

3.1.6.2 Link Width

The 82580 supports a maximum link width of x4, x2, or x1 as determined by the *Disable Lane* bits in the *PCIe PHY Auto Configuration* EEPROM section.

The max link width is loaded into the *Maximum Link Width* field of the *PCIe Capability* register (*LCAP[11:6]*). The hardware default is x4 link.

During link configuration, the platform and the 82580 negotiate on a common link width. The link width must be one of the supported PCIe link widths (x1, x2, x4), such that:

- If *Maximum Link Width* = x4, then the 82580 negotiates to either x4, x2 or x1.[1]
- If *Maximum Link Width* = x2, then the 82580 negotiates to either x2 or x1.
- If *Maximum Link Width* = x1, then the 82580 only negotiates to x1.

3.1.6.3 Polarity Inversion

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

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

3.1.6.4 L0s Exit latency

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

3.1.6.5 Lane-to-Lane De-Skew

A multi-lane link might 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 (400/200 ps for 2.5/5 Gbps), or full symbol time units (4 ns) of skew caused by the re-timing repeaters' insert/delete operations. Receivers use TS1 or TS2 or Skip Ordered Sets (SOS) to perform link de-skew functions.

The 82580 supports de-skew of up to 12 symbol times (48 ns for 2.5 GbpS link rate and 24 ns for 5Gbps link rate).

3.1.6.6 Lane Reversal

The following lane reversal modes are supported (see Figure 3-2):

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

Note: The restriction requires that a x2 interface to the 82580 must connect to lanes 0 and 1 on the 82580. The PCIe Card Electromechanical specification does not allow to route a x2 link to a wider connector. Therefore, a system designer is not allowed to connect a x2 link to lanes 2 and 3 of a PCIe connector. It is also recommended that when used in x2 mode on a NIC, the 82580 is connected to lanes 0 and 1 of the NIC.

[1] See restriction in Section 3.1.6.6.

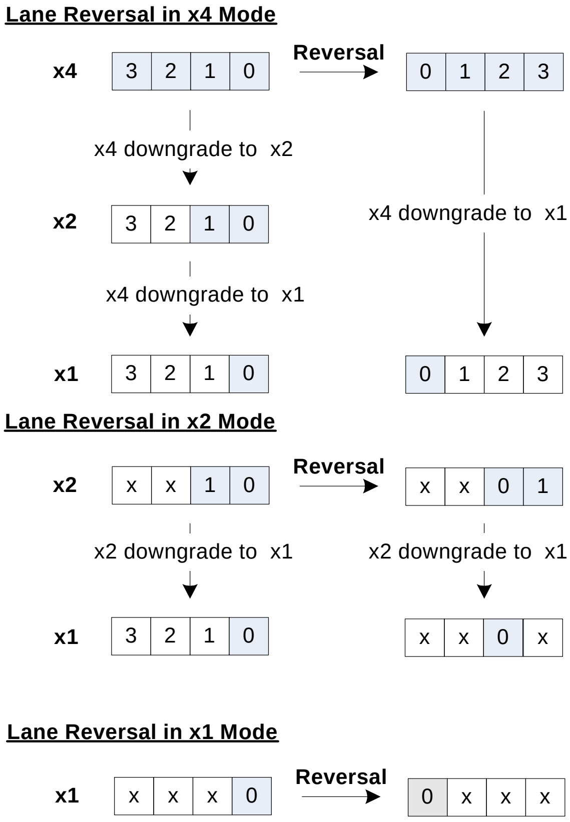


Figure 3-2. Lane Reversal Supported Modes

3.1.6.7 Reset

The PCIe PHY can supply core reset to the 82580. The reset can be caused by two sources:

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

3.1.6.8 Scrambler Disable

The scrambler/de-scrambler functionality in the 82580 can be eliminated by upstream according to the PCIe specification.

3.1.7 Error Events and Error Reporting

3.1.7.1 Mechanism in General

PCIe defines two error reporting paradigms: the baseline capability and the Advanced Error Reporting (AER) capability. The baseline error reporting capabilities are required of all PCIe devices and define the minimum error reporting requirements. The AER capability is defined for more robust error reporting and is implemented with a specific PCIe capability structure.

Both mechanisms are supported by the 82580.

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

In a multi-Function device, PCI Express errors that are not related to any specific Function within the device, are logged in the corresponding status and logging registers of all Functions in that device. These include the following cases of Unsupported Request (UR):

- A memory or I/O access that does not match any BAR for any function
- Messages.
- Configuration accesses to a non-existent function.

3.1.7.2 Error Events

Table 3-12 lists the error events identified by the 82580 and the response in terms of logging, reporting, and actions taken. Consult the PCIe specification for the effect on the PCI Status register.

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

Error Name	Error Events	Default Severity	Action
PHY errors			
Receiver error	8b/10b decode errors Packet framing error	Correctable. Send ERR_CORR	TLP to initiate NAK and drop data. DLLP to drop.
Data link errors			
Bad TLP	• Bad CRC • Not legal EDB • Wrong sequence number	Correctable. Send ERR_CORR	TLP to initiate NAK and drop data.
Bad DLLP	• Bad CRC	Correctable. Send ERR_CORR	DLLP to drop.
Replay timer timeout	• REPLAY_TIMER expiration	Correctable. Send ERR_CORR	Follow LL rules.
REPLAY NUM rollover	• REPLAY NUM rollover	Correctable. Send ERR_CORR	Follow LL rules.
Data link layer protocol error	• Violations of Flow Control Initialization Protocol • Reception of NACK/ACK with no corresponding TLP	Uncorrectable. Send ERR_FATAL	Follow LL rules.
TLP errors			
Poisoned TLP received	• TLP with error forwarding	Uncorrectable. ERR_NONFATAL Log header	A poisoned completion is ignored and the request can be retried after timeout. If enabled, the error is reported.

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

Error Name	Error Events	Default Severity	Action
Unsupported Request (UR)	• Wrong config access • MRdLk • Configuration request type 1 • Unsupported vendor Defined type 0 message • Not valid MSG code • Not supported TLP type • Wrong function number • Received TLP outside address range	Uncorrectable. ERR_NONFATAL Log header	Send completion with UR.
Completion timeout	• Completion timeout timer expired	Uncorrectable. ERR_NONFATAL	Error is non-fatal (default case) • Send error message if advisory • Retry the request once and send advisory error message on each failure • If fails, send uncorrectable error message Error is defined as fatal • Send uncorrectable error message
Completer abort	• Received target access with data size > 64-bit	Uncorrectable. ERR_NONFATAL Log header	Send completion with CA.
Unexpected completion	• Received completion without a request for it (tag, ID, etc.)	Uncorrectable. ERR_NONFATAL Log header	Discard TLP.
Receiver overflow	• Received TLP beyond allocated credits	Uncorrectable. ERR_FATAL	Receiver behavior is undefined.
Flow control protocol error	• Minimum initial flow control advertisements • Flow control update for infinite credit advertisement	Uncorrectable. ERR_FATAL	Receiver behavior is undefined. The 82580 doesn't report violations of Flow Control initialization protocol

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

Error Name	Error Events	Default Severity	Action
Malformed TLP (MP)	• Data payload exceed Max_Payload_Size • Received TLP data size does not match length field • TD field value does not correspond with the observed size • Power management messages that doesn't use TC0. • Usage of unsupported VC.	Uncorrectable. ERR_FATAL Log header	Drop the packet and free FC credits.
Completion with unsuccessful completion status		No action (already done by originator of completion).	Free FC credits.
Byte count integrity in completion process.	When byte count isn't compatible with the length field and the actual expected completion length. For example, length field is 10 (in Dword), actual length is 40, but the byte count field that indicates how many bytes are still expected is smaller than 40, which is not reasonable.	No action	The 82580 doesn't check for this error and accepts these packets. This may cause a completion timeout condition.

3.1.7.3 Error Forwarding (TLP poisoning)

If a TLP is received with an error-forwarding trailer, the transaction can be re-sent a number of times as programmed in the GCR register. If transaction still fails the packet is dropped and is not delivered to its destination. The 82580 then reacts as described in Table 3-12.

The 82580 does not initiate any 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.

3.1.7.4 ECRC

The 82580 supports End to End CRC (ECRC) as defined in the PCIe spec. The following functionality is provided:

- Insertion of ECRC in all transmitted TLPs
  - The 82580 indicates support for insertion of ECRC in the ECRC Generation Capable bit of the PCIe configuration registers. This bit is loaded from the "ECRC Generation" EEPROM bit.
  - Insertion of ECRC is enabled by the ECRC Generation Enable bit of the PCIe configuration registers.
- ECRC is checked on all incoming TLPs. A packet received with an ECRC error is dropped. Note that for completions, a completion timeout will occur later (if enabled), which would result in re-issuing the request.
  - The 82580 indicates support for ECRC checking in the ECRC Check Capable bit of the PCIe configuration registers. This bit is loaded from the "ECRC Check" EEPROM bit.
  - Checking of ECRC is enabled by the ECRC Check Enable bit of the PCIe configuration registers.

- ECRC errors are reported
- System SW may configure ECRC independently per each LAN function

3.1.7.5 Partial Read and Write Requests

Partial memory accesses

The 82580 has limited support of reads and writes requests with only part of the byte enable bits set:

- Partial writes with at least one byte enabled are silently dropped.
- Zero-length writes has no internal impact (nothing written, no effect such as clear-by-write). The transaction is treated as a successful operation (no error event).
- Partial reads with at least one byte enabled are handled as a full read. Any side effect of the full read (such as clear by read) is also applicable to partial reads.
- Zero-length reads generate a completion, but the register is not accessed and undefined data is returned.

The 82580 does not generate an error indication in response to any of the above events.

Partial I/O accesses

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

3.1.7.6 Error Pollution

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

Similarly, when the data link layer detects an error, subsequent errors that occur for the same packet are not signaled at the transaction layer.

3.1.7.7 Completion with Unsuccessful Completion Status

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

3.1.7.8 Error Reporting Changes

The Rev. 1.1 specification defines two changes to advanced error reporting. A new *Role-Based Error Reporting* bit in the Device Capabilities register is set to 1b to indicate that these changes are supported by the 82580.

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

2. Changes in the response to some uncorrectable non-fatal errors, detected in non-posted requests to the 82580. These are called advisory Non-fatal error cases. For each of the errors that follow, the following behavior is defined:
    a. The *Advisory Non-Fatal Error Status* bit is set in the Correctable Error Status register to indicate the occurrence of the advisory error and the *Advisory Non-Fatal Error Mask* corresponding bit in the Correctable Error Mask register is checked to determine whether to proceed further with logging and signaling.
    b. If the *Advisory Non-Fatal Error Mask* bit is clear, logging proceeds by setting the corresponding bit in the Uncorrectable Error Status register, based upon the specific uncorrectable error that's being reported as an advisory error. If the corresponding uncorrectable error bit in the Uncorrectable Error Mask register is clear, the First Error Pointer and Header Log registers are updated to log the error, assuming they are not still occupied by a previously unserviced error.
    c. An ERR_COR message is sent if the *Correctable Error Reporting Enable* bit is set in the Device Control register. An ERROR_NONFATAL message is not sent for this error.

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

- A completion with an Unsupported Request or Completer Abort (UR/CA) status that signals an uncorrectable error for a non-posted request. If the severity of the UR/CA error is non-fatal, the completer must handle this case as an advisory non-fatal error.
- When the requester of a non-posted request times out while waiting for the associated completion, the requester is permitted to attempt to recover from the error by issuing a separate subsequent request, or to signal the error without attempting recovery. The requester is permitted to attempt recovery zero, one, or multiple (finite) times, but must signal the error (if enabled) with an uncorrectable error message if no further recovery attempts are made. If the severity of the completion timeout is non-fatal and the requester elects to attempt recovery by issuing a new request, the requester must first handle the current error case as an advisory non-fatal error.
- Reception of a poisoned TLP. See Section 3.1.7.3.
- When a receiver receives an unexpected completion and the severity of the unexpected completion error is non-fatal, the receiver must handle this case as an advisory non-fatal error.

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

A DMA master transaction ending with an Unsupported Request (UR) completion or a Completer Abort (CA) completion will cause all PCIe Master transactions to stop, *PICAUSE.ABR* bit is set and an interrupt is generated if the appropriate Mask bits are set. To enable PCIe master transactions following reception of an UR or CA completion Software should issue a software reset (*CTRL.RST*).

3.1.8 PCIe Power Management

Described in Section 5.4.1 - Power Management.

3.1.9 PCIe Programming Interface

Described in Section 9.0 - PCIe Programming Interface

3.2 Management Interfaces

The 82580 contains twopossible interfaces to an external BMC.

- SMBus

NC-SISince the manageability sideband throughput is lower than the network link throughput, the 82580 allocates a 2 KB internal buffer for incoming network packets prior to being sent over the sideband interface. the 82580 also allocates a 2 KB internal buffer for outgoing network packets prior to being sent over the Ethernet link.

Note: Allocated buffer size is a function of number of active LAN ports. If only 2 LAN ports are enabled, internal Buffer size is 4 KB. If only a single LAN port is enabled, internal buffer size is 8 KB. Enabled LAN ports are defined in the *Manageability Capability/Manageability Enable* EEPROM word (Word 0x54).

3.2.1 SMBus

SMBus is an optional interface for pass-through and/or configuration traffic between an external BMC and the 82580. The SMBus commands used to configure or read status from the 82580 are described in Chapter 10.0.

3.2.1.1 Channel Behavior

3.2.1.1.1 SMBus Addressing

The SMBus is presented as four SMBus devices on the SMBus were each device has a different SMBus addresses. All pass-through functionality is duplicated per SMBus address, where each SMBus address is connected to a different LAN port.

Note: DO NOT configure ports to the same address. When a LAN function is disabled, the corresponding SMBus address is not presented to the external BMC.

The SMBus addresses are set by *SMBus Address 0*, *SMBus Address 1*, *SMBus Address 2* and *SMBus Address 3* words in the EEPROM.

The SMBus addresses (those that are enabled from the EEPROM) can be re-assigned using the SMBus ARP protocol.

Besides the SMBus address values, all the previously stated parameters of the SMBus (SMBus channel selection, address mode, address enable) can be set only through EEPROM configuration. The EEPROM is read by the 82580 at power-up, resets, and other cases described in Section 4.2.

All SMBus addresses should be in Network Byte Order (NBO); most significant byte first.

3.2.1.1.2 SMBus Notification Methods

The 82580 supports three methods of informing the external BMC that it has information that is needed to be read by an external BMC:

- SMBus alert.
- Asynchronous notify.
- Direct receive.

The notification method that is used by the 82580 can be configured from the SMBus using the Receive Enable command. The default method is set from the EEPROM in the *Notification Method* field.

The following events cause the 82580 to send a notification event to the external BMC:

- Receiving a LAN packet that was designated to the BMC.

- Receiving a request status command from the BMC initiates a status response (see Section 10.3.2.1.2).
- Status change has occurred and the 82580 is configured to notify the external BMC upon one of the status changes. The following event triggers a notification to the BMC:
  - — A change in any of the *Status Data 1* bits of the Read Status command (see Section 10.3.2.2.3 for description of this command).

There might be cases where the external BMC is hung and is unable to respond to the SMBus notification. The 82580 has a time-out value defined in the EEPROM (see Section 6.10) to avoid hanging while waiting for the notification response. If the BMC does not respond until the timeout expires, the notification is de-asserted.

3.2.1.1.2.1 SMBus Alert and Alert Response Method

The SMBus Alert# signal is an additional SMBus signal that acts as an asynchronous interrupt signal to an external SMBus master. The 82580 asserts this signal each time it has a message that it needs the external BMC to read and if the chosen notification method is the SMBus-alert method. Note that the SMBus alert is an open-drain signal, which means that other devices besides the 82580 can be connected on the same alert pin and the external BMC needs a mechanism to distinguish between the alert sources as described:

The external BMC can respond to the alert by issuing an ARA (Alert Response Address) cycle (see Figure 3-13) to detect the alert source device. The 82580 responds to the ARA cycle (if it was the SMBus alert source) and de-asserts the alert when the ARA cycle completes. Following the ARA cycle, the external BMC issues a Read command to retrieve the 82580 message.

Some BMCs do not implement ARA cycle transactions. These BMCs respond to an alert by issuing a Read command to all active the 82580 ports (0xC0/0xD0 or 0xDE). The 82580 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 82580 is the source of the SMBus alert, it replies to the read transaction and de-asserts the alert after the command byte of the read transaction.

The ARA cycle is an SMBus receive byte transaction to SMBus Address 0001-100b. Note that the ARA transaction does not support PEC. The ARA transaction format is as follows:

Table 3-13. SMBus ARA Cycle Format

1	7	1	1	7	1	1	1
S	Alert Response Address	Rd	A	Slave Device Address		A	P
	0001 100	1	0	Manageability Slave SMBus Address	0	1	

Note: Since the master-receiver (BMC receiver) is involved in the transaction it must signal the end of data to the 82580 by generating a NACK (a '1' in the ACK bit position) on the Slave Device Address byte that was clocked out by the 82580. The 82580 releases the data line to allow the master to generate a STOP condition.

3.2.1.1.2.2 Asynchronous Notify Method

When configured to asynchronous notify method, the 82580 acts as SMBus master and notifies the external 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 method is not protected by a PEC byte.

Table 3-14. Asynchronous Notify Command Format

1	7	1	1	7	1	1	
S	Target Address	Wr	A	Sending Device Address		A	• • •
	BMC Slave Address	0	0	Manageability Slave SMBus Address	0	0	

8	1	8	1	1
Data Byte Low	A	Data Byte High	A	P
Interface	0	Alert Value	0	

The target address and data byte low/high is taken from the Receive Enable command (see Section 10.3.2.1.3) or EEPROM configuration (See Section 6.10).

3.2.1.1.2.3 Direct Receive Method

If configured, the 82580 has the capability to send the 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.

Table 3-15 shows the message format when receiving a LAN packet that was designated to the BMC. Note that the "F", "L" and command fields in the message are the same as the op-code returned by the 82580 in response to a BMC *Receive TCO Packet* Block read command (See Section 10.3.2.2.1). The rules for the "F" and "L" flags are also the same as used in the *Receive TCO Packet* Block Read command.

Table 3-15. Direct Receive - LAN Packet Receive Transaction Format

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	

Table 3-16 shows the message format when status change has occurred and the 82580 is configured to notify the external BMC upon a status change. Note that the op-code and Status data fields returned by the 82580 are the same as in response to a BMC *Read Status* command (see Section 10.3.2.2.3).

Table 3-16. Direct Receive - Status Change Transaction Format

1	7	1	1	8	1	
S	Target Address	Wr	A	Command	A	• • •
	BMC Slave Address	0	0	Read status op-code 0xDD	0	

8	1	8	1	8	1	1
Byte Count	A	Data 2 (Status Data 1)	A	Data 3 (Status Data 2)	A	P
0x02	0		0		0	

3.2.1.1.3 Receive TCO Flow

The 82580 is used as a channel for receiving packets from the network link and passing them to the external BMC. The BMC can configure the 82580 to pass specific packets to the BMC as described in Section 10.3.2.1.6. Once a full packet is received from the link and identified as a manageability packet that should be transferred to the BMC, the 82580 starts the receive TCO transaction flow to the BMC.

The maximum SMBus fragment length is defined in the EERPOM (See Section 6.10.3). The 82580 uses the SMBus notification method to notify the BMC that it has data to deliver. The packet is divided into fragments, where the 82580 uses the maximum fragment size allowed in each fragment. The last fragment of the packet transfer is always the status of the packet (see Section 10.3.2.2.2). As a result, the packet is transferred in at least two fragments. The data of the packet is transferred in the Receive TCO LAN packet transaction as described in Section 10.3.2.2.1.

When SMBus alert is selected as the BMC notification method, the 82580 notifies the BMC on each fragment of a multi-fragment packet. When asynchronous notify is selected as the BMC notification method, the 82580 notifies the BMC only on the first fragment of a received packet. It is BMC's responsibility to read the full packet including all the fragments.

Any timeout on SMBus notification results in discarding the entire packet. Any NACK by the BMC on one of the 82580 receive bytes also causes the packet to be silently discarded.

If a SMBus time-out occurs during reception of a packet from the network to the BMC, the 82580 silently discards the packet.

The maximum size of the received packet is limited by the 82580 hardware to 1536 bytes. Packets larger then 1536 bytes are silently discarded. Any packet smaller than 1536 bytes is processed by the 82580.

Note: When the *RCV_EN* bit is cleared (see Section 10.3.2.1.3), all receive TCO functionality is disabled, not just the packets that are directed to the BMC.

3.2.1.1.4 Transmit TCO Flow

The 82580 is used as a channel for transmitting packets from the external BMC to the network link. The network packet is transferred from the external BMC over the SMBus, and then, when fully received by the 82580, is transmitted over the network link.

Each SMBus address is connected to a different LAN port. When a packet is received in SMBus transactions using *SMBus Address 0*, *SMBus Address 1*, *SMBus Address 2* or *SMBus Address 3* it is transmitted to the network using LAN port 0, LAN port 1, LAN port 2 or LAN port 3 respectively.

The 82580 supports packets up to the Ethernet packet length (1536 bytes). SMBus transactions can be up to 240 bytes in length, which means that packets can be transferred over the SMBus in more than one fragment. In each command byte there are the *F* and *L* bits. When the *F* bit is set, it means that this is the first fragment of the packet; *L* means that it is the last fragment of the packet.

Note: When both flags 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 82580 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 external BMC (the 82580 does not change any field in the transmitted packet, besides adding padding and CRC bytes).

Note: If the packet sent by the BMC is larger than 1536 bytes, then the packet is discarded by the 82580 and Abort is asserted.

The minimum packet length defined by the 802.3 specification is 64 bytes. The 82580 pads packets that are less than 64 bytes to meet the specification requirements. There is one exception, when the packet sent over the SMBus is less than 32 bytes, the external BMC must pad it for at least 32 bytes. The passing bytes value should be zero.

Note: Packets that are smaller then 32 bytes (including padding), then the packet is discarded by the 82580 and Abort is asserted.

If the network link goes down at anytime while the 82580 is receiving a packet from the BMC for transmission on the network, it silently discards the packet. Note that any link down event during the transfer of a packet to the BMC over the SMBus (after it is fully received from the network), does not stop the operation.

Note: If a SMBus time-out occurs during transmission of a packet from the BMC to the network the 82580 silently discards the packet.

The transmit SMBus transactions are described in Section 10.3.2.1.

Transmit Errors in Sequence Handling

Once a packet is transferred over the SMBus from the BMC to the 82580, 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.

The following table lists the different options regarding the flags in transmit packet transactions:

Table 3-17. Flags in Transmit Packet Transactions

Previous	Current	Action/Notes
Last	First	Accepts both.
Last	Not First	Error for current transaction. Current transaction is discarded and an abort status is asserted.
Not Last	First	Error for previous transaction. Previous transaction (until previous first) is discarded. Current packet is processed. No abort status is asserted.
Not Last	Not First	Processes the current transaction.

Note that since every other Block Write command in the TCO protocol has both *F* and *L* flags off, they cause flushing any pending transmit fragments that were previously received. In other words, when running the TCO transmit flow, no other block write transactions are allowed in between the fragments.

3.2.1.1.5 TCO Command Aborted Flow

Bit 6 in first byte of the status returned from the 82580 to the external BMC indicates that there was a problem with previous SMBus transactions or with the completion of the operation requested in previous transaction.

An abort can be asserted for any of the following reasons:

- Any error in the SMBus protocol (NACK, SMBus timeout).
- If the BMC does not respond until the notification timeout (programmed in the EEPROM) expires.
- Any error in compatibility between required protocols to specific functionality (Receive Enable command with byte count not 1/14 as defined in the command specification).
- If the 82580 does not have space to store the transmit packet from the BMC (in its internal buffer before sending it to the link). In this case, the entire transaction completes, but the packet is discarded and the BMC is notified about it through the *Abort* bit.
- Error in the *F/L* bit sequence during multi-fragment transactions.
- If the packet sent by the BMC is larger than 1536 bytes.
- If the packet sent by the BMC is smaller than 32 bytes (including padding).
- An internal reset to the 82580 manageability unit occurred.
- Following an unsuccessful internal register access when the *MCSR_TO_RETRY* EEPROM bit is cleared.

Note: An abort in the status does not always imply that the last transaction of the sequence was incorrect. There is a gap between the time the status is read from the 82580 and the time the transaction occurred.

3.2.1.1.6 Concurrent SMBus Transactions

Concurrent SMBus transactions (receive, transmit and configuration read/write) are allowed between the four addresses supported by the 82580. Transmit fragments can be sent between receive fragments and configuration Read/Write commands can also be issued between receive and transmit fragments.

3.2.1.1.7 SMBus ARP Functionality

The 82580 supports SMBus ARP protocol as defined in the SMBus 2.0 specification. The 82580 is a persistent slave address device meaning that its SMBus address is valid after power-up and loaded from the EEPROM. The 82580 supports all SMBus ARP commands defined in the SMBus specification, both general and directed.

Note: SMBus ARP can be disabled through EEPROM configuration (See Section 6.10.4).

SMBus-ARP transactions are described in Section 10.3.2.3.

3.2.1.1.7.1 SMBus ARP Response Behavior

The 82580 responds as four SMBus devices, meaning that it has four sets of *AR/AV* flags (one for each port). The 82580 responds four times to the SMBus-ARP master, one time for each port. All SMBus addresses are taken from the SMBus ARP addresses word of the EEPROM. The UDID is different between the four ports in the Vendor Specific ID field, which represent the MAC address, which is different between the four ports. The 82580 answers first as port 0, and only when the address is assigned, starts answering as port 1, 2 and 3 to the Get UDID command.

3.2.1.1.7.2 SMBus ARP Flow

SMBus-ARP flow is based on the status of two flags:

- *AV* - Address Valid - This flag is set when the 82580 has a valid SMBus address.
- *AR* - Address Resolved - This flag is set when the 82580's SMBus address is resolved (SMBus address was assigned by the SMBus-ARP process).

Note: These flags are internal the 82580 flags and not shown to external SMBus devices.

Since the 82580 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 the *AV* flag is always set, the 82580 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 the devices on the SMBus by issuing either Prepare to ARP or Reset Device commands. When the 82580 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).

The meaning of an *AR* flag cleared is that the 82580 answers the following SMBus ARP transactions that are issued by the master. The SMBus master then issues a Get UDID command (general or directed), to identify the devices on the SMBus. The 82580 responds to the directed command all the time and to the general command only if its *AR* flag is not set. After the Get UDID command, the master assigns the 82580's SMBus address by issuing an Assign Address command. The 82580 checks whether the UDID matches its own UDID, and if they match, 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 on (as long as the *AR* flag is set), the 82580 does not respond to the Get UDID general command, while all other commands should be processed even if the *AR* flag is set. The 82580 stores the SMBus address that was assigned in the SMBus-ARP process in its EEPROM, so after the next power-up, it returns to its assigned SMBus address.

Figure 3-3 shows the SMBus-ARP behavior of the 82580.

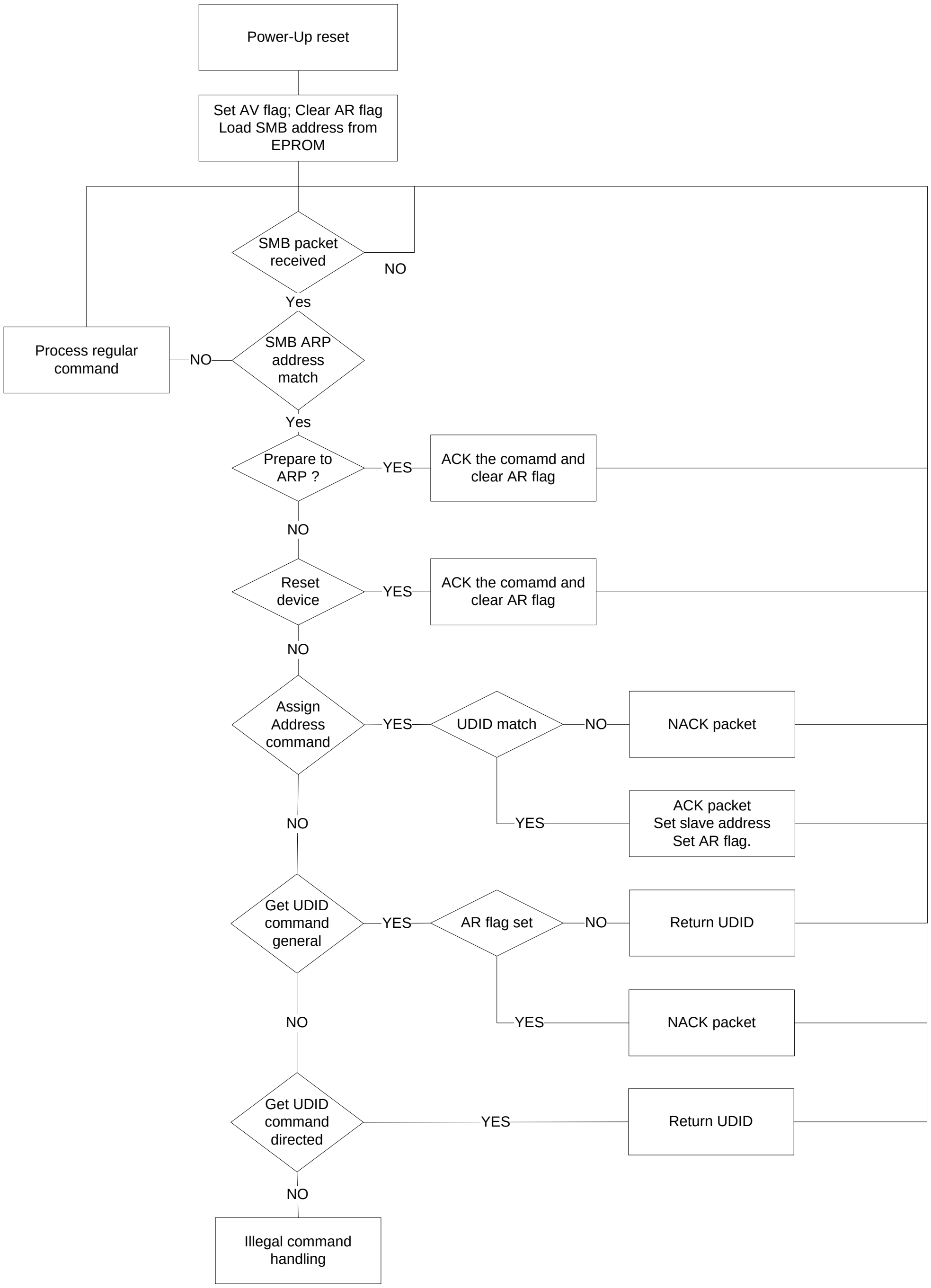


Figure 3-3. SMBus ARP Flow

SMBus ARP UDID Content

The Unique Device Identifier (UDID) provides a mechanism to isolate each device for the purpose of address assignment. Each device has a unique identifier. The 128-bit number is comprised of the following fields:

Table 3-18. Unique Device Identifier (UDID)

1 Byte	1 Byte	2 Bytes	2 Bytes	2 Bytes	2 Bytes	2 Bytes	4 Bytes
Device Capabilities	**Version / Revision**	**Vendor ID**	**Device ID**	**Interface**	**Sub-system Vendor ID**	**Sub- system Device ID**	**Vendor Specific ID**
See below	See below	0x8086	PCIe Dev ID[1]	0x0004	0x0000	0x0000	See below
MSB							LSB

[1] Device ID defined in Section 9.4.2.

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: See Section 9.4.2.
- Interface - Identifies the protocol layer interfaces supported over the SMBus connection by the device - In this case, SMBus Version 2.0 - Constant value: 0x0004.
- Sub-system Fields - These fields are not supported and return zeros.

Device Capabilities: Dynamic and Persistent Address, PEC Support bit:

Table 3-19. 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:

Table 3-20. 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:

Table 3-21. Silicon Revision ID

Silicon version	Revision ID
A1	001b

Vendor Specific ID - Four LSB bytes of the 82580's Ethernet MAC address. The 82580 Ethernet address is taken from offsets 0 to 2 from start of the relevant sections in the EEPROM. Note that in the 82580 there are four MAC addresses (one for each port).

Table 3-22. Vendor Specific ID

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

3.2.2 NC-SI

The NC-SI interface in the 82580 is a connection to an external BMC defined by the DMTF NC-SI protocol. It operates as a single interface with an external BMC, where all traffic between the 82580 and the BMC flows through the interface.

3.2.2.1 Electrical Characteristics

The 82580 complies with the electrical characteristics defined in the NC-SI specification.

The 82580 NC-SI behavior is configured on power-up in the following manner:

- The 82580 provides an NC-SI clock output if defined by the *NC-SI Output Clock Disable* EEPROM bit (Section 6.2.19). The default value is to use an external clock source as defined in the NC-SI specification.
- The output driver strength for the NC-SI_CLK_OUT pad is configured by the *NC-SI Clock Pad Drive Strength* bit (default = 0b) in the *Functions Control* EEPROM word (Section 6.2.19).
- The output driver strength for the NC-SI output signals (NC-SI_DV & NC-SI_RX) is configured by the EEPROM *NC-SI Data Pad Drive Strength* bit (default = 0b see Section 6.2.19).
- The *Multi-Drop NC-SI EEPROM* bit (Section 6.6.7) defines the NC-SI topology (point-to-point or multi-drop; the default is point-to-point).

The 82580 can provide an NC-SI clock output as previously mentioned. The NC-SI clock input (NC-SI_CLK_IN) serves as an NC-SI input clock in either case. That is, if the 82580 provides an NC-SI output clock, the platform is required to route it back through the NC-SI clock input with the correct latency. See the Electrical chapter for more details.

The 82580 dynamically drives its NC-SI output signals (NC-SI_DV and NC-SI_RX) as required by the sideband protocol:

- On power-up, the 82580 floats the NC-SI outputs except for NCSI_CLK_OUT.
- If the 82580 operates in point-to-point mode, then the 82580 starts driving the NC-SI outputs some time following power-up.
- If the 82580 operates in a multi-drop mode, the 82580 drives the NC-SI outputs as configured by the BMC.

3.2.2.2 NC-SI Transactions

The NC-SI link supports both pass-through traffic between the BMC and the 82580 LAN functions, as well as configuration traffic between the BMC and the 82580 internal units as defined in the NC-SI protocol. See Section 10.2.1 and Section 10.3.1 in this manual.

3.3 Flash / EEPROM

3.3.1 EEPROM Interface

3.3.1.1 General Overview

The 82580 uses an EEPROM device for storing product configuration information. The EEPROM is divided into three general regions:

- Hardware accessed - Loaded by the 82580 after power-up, PCI reset de-assertion, D3 ->D0 transition, or a software-commanded EEPROM read (CTRL_EXT.EE_RST).
- Manageability firmware accessed - Loaded by the 82580 in pass-through mode after power-up or firmware reset.
- Software accessed - Used only by software. The meaning of these registers, as listed here, is a convention for software only and is ignored by the 82580.

Table 3-23 lists the structure of the EEPROM image in the 82580.

Table 3-23. EEPROM Structure

Address	Content
0x0 – 0x9	LAN 0 MAC address and software area
0xA – 0x2F	LAN 0 and Common hardware area
0x30 – 0x3E	PXE area
0x3F	Software Checksum, for Words 0x00 - 0x3E
0x40 – 0x4F	Software area
0x50 – 0x7F	FW pointers
0x80 -0xBF	LAN 1 hardware area (with SW checksum in 0xBF)
0xC0 - 0xFF	LAN 2 hardware area (with SW checksum in 0xFF)
0x100 - 0x13F	LAN 3 hardware area (with SW checksum in 0x13F)
…	
	Firmware structures
…	
	VPD area
…	
	CSR and Analog configuration (PCIe/PHY/PLL/SerDes structures)

The EEPROM mapping is described in Chapter 6.0.

3.3.1.2 EEPROM Device

The EEPROM interface supports an SPI interface and expects the EEPROM to be capable of 2 MHz operation.

The 82580 is compatible with various sizes of 4-wire serial EEPROM devices. If pass-through mode functionality is desired, up to 256 Kbits serial SPI compatible EEPROM can be used. If no manageability mode is desired, a 128 Kbits (16 Kbytes) serial SPI compatible EEPROM can be used. All EEPROM's are accessed in 16-bit words although the EEPROM is designed to also accept 8-bit data accesses.

The 82580 automatically determines the address size to be used with the SPI EEPROM it is connected to and sets the EEPROM address size field of the *EEPROM/FLASH Control and Data* register (*EEC.EE_ADDR_SIZE*) field appropriately. Software can use this size to determine how to access the EEPROM. The exact size of the EEPROM is determined within one of the EEPROM words.

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

3.3.1.3 HW initial load process.

Upon power on reset or PCIe reset, the 82580 reads the global device parameters from the EEPROM including all the parameters impacting the content of the PCIe configuration space. Upon a software reset to one of the ports (*CTRL.RST* set to 1), a partial load is done of the parameters relevant to the port were the software reset occurred. Upon a software reset to all ports (*CTRL.DEV_RST* = 1) a partial load is done of the parameters relevant to all ports. Table 3-24 lists the words read in each EEPROM auto-read sequence. During full load after power-on all hardware related EEPROM words are loaded. Following a software reset only a subset of the hardware related EEPROM words are loaded. For details of the content of each word - see Chapter 6.0.

Note: LANx_start parameter in Table 3-24 relates to start of LAN related EEPROM section where:

- — LAN0_start = 0x0
- — LAN1_start = 0x80
- — LAN2_start = 0xC0
- — LAN3_start = 0x100

Table 3-24. EEPROM Auto-Load Sequence

EEPROM Word	EEPROM Word Address	Full Load (Power-up)	Full Load No MGMT (PCI RST)	SW[1] reset port 0 Load	SW[1] reset port 1 Load	SW[1] reset port 2 Load	SW[1] reset port 3 Load
EEPROM sizing and protected fields	0x12	Y	Y	Y	Y	Y	Y
CSR Auto Configuration Power-Up LAN0	0x27	Y					
CSR Auto Configuration Power-Up LAN1	LAN1_start + 0x27	Y					
CSR Auto Configuration Power-Up LAN2	LAN2_start + 0x27	Y					
CSR Auto Configuration Power-Up LAN3	LAN3_start + 0x27	Y					
PCIe PHY Auto Configuration Pointer and PCIe PHY Auto Configuration structures.	0x10	Y					
Init Control 1	0x0A	Y	Y	Y	Y	Y	Y
PCIe init configuration 1	0x18	Y	Y				
PCIe init configuration 2	0x19	Y	Y				
PCIe init configuration 3	0x1A	Y	Y				
PCIe control 1	0x1B	Y	Y				
PCIe control 2	0x28	Y	Y				
PCIe control 3	0x29	Y	Y				
HW Section							
Functions control	0x21	Y	Y				

Table 3-24. EEPROM Auto-Load Sequence (Continued)

EEPROM Word	EEPROM Word Address	Full Load (Power-up)	Full Load No MGMT (PCI RST)	SW[1] reset port 0 Load	SW[1] reset port 1 Load	SW[1] reset port 2 Load	SW[1] reset port 3 Load
Device Rev ID	0x1E	Y	Y				
PCIe L1 Exit latencies	0x14	Y	Y				
NC-SI and PCIe completion timeout configuration	0x15	Y	Y				
Subsystem ID[2]	0x0B	Y	Y				
Subsystem Vendor ID[2]	0x0C	Y	Y				
Device ID - LAN 0[3]	0x0D	Y	Y				
Device ID - LAN 1[3]	LAN1_start + 0x0D	Y	Y				
Device ID - LAN 2[3]	LAN2_start + 0x0D	Y	Y				
Device ID - LAN 3[3]	LAN3_start + 0x0D	Y	Y				
Vendor ID - LAN 0[3]	0x0E	Y	Y				
Dummy function device ID[3]	0x1D	Y	Y				
MSI-X configuration LAN 0	0x16	Y	Y				
MSI-X configuration LAN 1	LAN1_start + 0x16	Y	Y				
MSI-X configuration LAN 2	LAN2_start + 0x16	Y	Y				
MSI-X configuration LAN 3	LAN3_start + 0x16	Y	Y				
LAN power consumption	0x22	Y	Y				
CSR Auto Configuration Pointer and CSR Auto Configuration structures - LAN0	0x17	Y	Y	Y			
CSR Auto Configuration Pointer and CSR Auto Configuration structures - LAN1	LAN1_start + 0x17	Y	Y		Y		
CSR Auto Configuration Pointer and CSR Auto Configuration structures - LAN2	LAN2_start + 0x17	Y	Y			Y	
CSR Auto Configuration Pointer and CSR Auto Configuration structures - LAN3	LAN3_start + 0x17	Y	Y				Y
VPD Pointer to table	0x2F	Y	Y				
VPD table entry ID TAG	ID STRING	Y	Y				
VPD read or write area TAG	VPD TAG 1	Y	Y				
VPD read or write area length	VPD TAG 1 LENGTH	Y	Y				
VPD read or write area TAG	VPD TAG 2	Y	Y				
VPD read or write area length	VPD TAG 2 LENGTH	Y	Y				
VPD end TAG	VPD END	Y	Y				
Init Control 3 LAN 0	0x24	Y	Y				
Init Control 3 LAN 1	LAN1_start + 0x24	Y	Y				
Init Control 3 LAN 2	LAN2_start + 0x24	Y	Y				
Init Control 3 LAN 3	LAN3_start + 0x24	Y	Y				
Init Control 4 LAN 0	0x13	Y	Y	Y			
Init Control 4 LAN 1	LAN1_start + 0x13	Y	Y		Y		
Init Control 4 LAN 2	LAN2_start + 0x13	Y	Y			Y	
Init Control 4 LAN 3	LAN3_start + 0x13	Y	Y				Y
LEDCTL 1 3 default LAN 0	0x1C	Y	Y				

Table 3-24. EEPROM Auto-Load Sequence (Continued)

EEPROM Word	EEPROM Word Address	Full Load (Power-up)	Full Load No MGMT (PCI RST)	SW[1] reset port 0 Load	SW[1] reset port 1 Load	SW[1] reset port 2 Load	SW[1] reset port 3 Load
LEDCTL 0 2 default LAN 0	0x1F	Y	Y				
LEDCTL 1 3 default LAN 1	LAN1_start + 0x1C	Y	Y				
LEDCTL 0 2 default LAN 1	LAN1_start + 0x1F	Y	Y				
LEDCTL 1 3 default LAN 2	LAN2_start + 0x1C	Y	Y				
LEDCTL 0 2 default LAN 2	LAN2_start + 0x1F	Y	Y				
LEDCTL 1 3 default LAN 3	LAN3_start + 0x1C	Y	Y				
LEDCTL 0 2 default LAN 3	LAN3_start + 0x1F	Y	Y				
End of read only area	0x2C	Y	Y				
Start of read only area	0x2D	Y	Y				
Init Control 2	0x0F	Y	Y	Y	Y	Y	Y
Ethernet address byte 2-1 - LAN 0	0x00	Y	Y	Y			
Ethernet address byte 4-3 - LAN 0	0x01	Y	Y	Y			
Ethernet address byte 6-5 - LAN 0	0x02	Y	Y	Y			
Ethernet address byte 2-1 - LAN 1	LAN1_start + 0x00	Y	Y		Y		
Ethernet address byte 4-3 - LAN 1	LAN1_start + 0x01	Y	Y		Y		
Ethernet address byte 6-5 - LAN 1	LAN1_start + 0x02	Y	Y		Y		
Ethernet address byte 2-1 - LAN 2	LAN2_start + 0x00	Y	Y			Y	
Ethernet address byte 4-3 - LAN 2	LAN2_start + 0x01	Y	Y			Y	
Ethernet address byte 6-5 - LAN 2	LAN2_start + 0x02	Y	Y			Y	
Ethernet address byte 2-1 - LAN 3	LAN3_start + 0x00	Y	Y				Y
Ethernet address byte 4-3 - LAN 3	LAN3_start + 0x01	Y	Y				Y
Ethernet address byte 6-5 - LAN 3	LAN3_start + 0x02	Y	Y				Y
Software defined pins control - LAN0	0x20	Y	Y	Y			
Software defined pins control - LAN1	LAN1_start + 0x20	Y	Y		Y		
Software defined pins control - LAN2	LAN2_start + 0x20	Y	Y			Y	
Software defined pins control - LAN3	LAN3_start + 0x20	Y	Y				Y
Management Section[4]							
Pass Through LAN Configuration Pointer LAN0	0x11	Y					
Pass Through LAN Configuration Pointer LAN1	LAN1_start + 0x11	Y					
Pass Through LAN Configuration Pointer LAN2	LAN2_start + 0x11	Y					
Pass Through LAN Configuration Pointer LAN3	LAN3_start + 0x11	Y					
Management Hardware Config Control	0x23	Y					
MNG Capabilities	0x54	Y					
Sideband Configuration Pointer	0x57	Y					
Reserved	0x5E	Y					

Table 3-24. EEPROM Auto-Load Sequence (Continued)

EEPROM Word	EEPROM Word Address	Full Load (Power-up)	Full Load No MGMT (PCI RST)	SW[1] reset port 0 Load	SW[1] reset port 1 Load	SW[1] reset port 2 Load	SW[1] reset port 3 Load
Firmware patch Pointer	0x51	Y					
HW Section Continued							
Watchdog configuration	0x2E[5]	Y	Y	Y	Y	Y	Y

1. Upon assertion of *CTRL.DEV_RST* by software partial load of parameters relevant to all ports is done. Assertion of *CTRL_EXT.EE_RST* causes load of per port parameters similar to *CTRL.RST*.
2. Loaded only if load subsystem ID bit is set
3. Loaded only if load device ID bit is set
4. EEPROM words listed under Management Section are also loaded following Firmware Reset
5. Word also loaded following Firmware reset.

3.3.1.4 Software Accesses

The 82580 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.

In addition, the VPD area of the EEPROM can be accessed via the VPD capability structure of the PCIe.

Software can use the EEPROM Read (*EERD*) register to cause the 82580 to read a word from the EEPROM that the software can then use. To do this, software writes the address to read to the *Read Address* (*EERD.ADDR*) field and simultaneously writes a 1b to the *Start Read* bit (*EERD.START*). The 82580 reads the word from the EEPROM, sets the *Read Done* bit (*EERD.DONE*), and places the data in the *Read Data* field (*EERD.DATA*). Software can poll the EEPROM Read register until it sees the *Read Done* bit set and then uses the data from the *Read Data* field. Any words read this way are not written to the 82580's internal registers.

Software can also directly access the EEPROM's 4-wire interface through the EEPROM/Flash Control (*EEC*) register. 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 Reques*t bit (*EEC.EE_REQ*).
2. Read the *EEPROM Grant* bit (*EEC.EE_GNT*) 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 and Data (*EEC*) register. 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*) to enable EEPROM access by other drivers.

Note: If direct access via the EEPROM's 4-wire interface to a read protected area is attempted, the 82580 blocks the access and sets the *EEC.EE_BLOCKED* bit.

Finally, software can cause the 82580 to re-read the per-function hardware accessed fields of the EEPROM (setting the 82580's internal registers appropriately similar to software reset) by writing a 1b to the *EEPROM Reset* bit of the Extended Device Control register (*CTRL_EXT.EE_RST*).

Note: If the EEPROM does not contain a valid signature (see Section 3.3.1.5), the 82580 assumes 16-bit addressing. In order to access an EEPROM that requires 8-bit addressing, software must use the direct access mode.

3.3.1.5 Signature Field

The 82580 determines if an EEPROM is present by attempting to read it. The 82580 first reads the *EEPROM Sizing and Protected Fields* word at address 0x12. It checks the signature value for bits 15 and 14. If bit 15 is 0b and bit 14 is 1b, it considers the EEPROM to be present and valid and reads additional EEPROM words and then programs its internal registers based on the values read. Otherwise, it ignores the values it reads from that location and does not read any other words as part of the auto-read process. However, the EEPROM is still accessible to software.

Note: If bit 15 in the *EEPROM Sizing and Protected Fields* word is read as 1b, The 82580 assumes that there is no EEPROM connected. It does not attempt any further auto-reads of the EEPROM. if a valid image is later programmed, The 82580 will not attempt an auto-read until a full power cycle is performed, i.e. until the assertion of LAN_PWR_GOOD.

3.3.1.6 Protected EEPROM Space

The 82580 provides a mechanism for a hidden area in the EEPROM to the host. The hidden area cannot be accessed (read or written to) via the EEPROM registers in the CSR space. It can be accessed only by the manageability subsystem. This area is located at the end of the EEPROM memory. It’s size is defined by the HEPSize field in EEPROM word 0x12. Note that the current the 82580 manageability firmware does not use the HEPSize mechanism.

A mechanism to protect part of the EEPROM from host writes is also provided. This mechanism is controlled by word 0x2D and 0x2C that controls the start and the end of the read-only area.

Note: If the VPD area is mapped to the hidden EEPROM space, Software can write to the hidden EEPROM space via the VPD capability structure in the PCIe configuration space. To avoid such an occurrence the VPD area pointer (EEPROM word 0x2F) should be placed in the read only area, defined by words 0x2C and 0x2D.

3.3.1.6.1 Initial EEPROM Programming

In most applications, initial EEPROM programming is done directly on the EEPROM pins. Nevertheless, it is desired to enable existing software utilities (accessing the EEPROM via the host interface) to initially program the entire EEPROM without breaking the protection mechanism. Following a power-up sequence, the 82580 reads the hardware initialization words in the EEPROM. If the signature in word 0x12 does not equal 01b, the EEPROM is assumed as non-programmed. There are two effects of a non-valid signature:

- The 82580 does not read any further EEPROM data and sets the relevant registers to default.
- The 82580 enables access to any location in the EEPROM via the EEPROM CSR registers.

3.3.1.6.2 Activating the Protection Mechanism

Following initialization, the 82580 reads the EEPROM and turns on the protection mechanism if word 0x12 contains a valid signature (equals 01b) and word 0x12, bit 4 is set (enable protection). Once the protection mechanism is turned on, words 0x12, 0x2C and 0x2D become write-protected, the area that is defined by word 0x12 becomes hidden (such as read/write protected) and the area defined by words 0x2C and 0x2D become write protected.

- No matter what is designated as the read only protected area, words 0x30:0x3F (used by PXE driver) are writable, unless it is defined as hidden.

3.3.1.6.3 Non Permitted Accessing to Protected Areas in the EEPROM

This paragraph refers to EEPROM accesses via the EEC (bit banging) or EERD (parallel read access) registers. Following a write access to the protected areas in the EEPROM, hardware responds properly on the PCIe interface but does not initiate any access to the EEPROM. Following a read access to the hidden area in the EEPROM (as defined by word 0x12), hardware does not access the EEPROM and returns meaningless data to the host.

Note: Using bit banging, the SPI EEPROM can be accessed in a burst mode. For example, providing op-code, address, and then read or write data for multiple bytes. Hardware inhibits any attempt to access the protected EEPROM locations even in burst accesses.

Note: 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.

3.3.1.7 EEPROM Recovery

The EEPROM contains fields that if programmed incorrectly might affect the functionality of the 82580. The impact can range from an incorrect setting of some function (such as LED programming), via disabling of entire features (such as no manageability) and link disconnection, to the inability to access the 82580 via the regular PCIe interface.

The 82580 implements a mechanism that enables recovery from a faulty EEPROM no matter what the impact is, using an SMBus message that instructs 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, firmware clears the signature of the EEPROM in word 0x12 (bits 15/14 to 00b). Afterwards, the BIOS/ operating system initiates a reset to force an EEPROM auto-load process that fails in order to enable access to the 82580.

Firmware is programmed to receive such a command only from a PCIe reset until one of the functions changes it’s status from D0u to D0a. Once one of the functions moves to D0a, it can be safely assumed that the 82580 is accessible to the host and there is no further need for this function. This reduces the possibility of malicious software using this command as a back door and limits the time 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 the SMBus Block write as follows:

Table 3-25. Command Format

Function	Command	Byte Count	Data Byte
Release EEPROM	0xC7	0x01	0xAA

Note: This solution requires a controllable SMBus connection to the 82580.

If more than one the 82580 part is in a state to accept this command, all of the devices in this state will respond with an ACK to this command and accept it. A device with one of it’s ports in D0a should not respond with an ACK to this command if not in D0u state.

The 82580 is guaranteed to accept the command on the SMBus interface and on address 0xC8 but it might be accepted on other configured interfaces and addresses as well.

If the command SMBus address is different from 0xC8 or if one of the functions is in D0 state, the 82580 will not accept the command and will return a NACK.

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

3.3.1.7.1 Access to the EEPROM Controlled Feature

The EEARBC register enables access to registers that are not accessible via regular CSR access (such as PCIe configuration read-only registers) by emulating the auto-read process. EEARBC contains five strobe fields that emulate the internal strobes of the internal auto-read process. This register is common to all functions and should be accessed only after verifying that it’s not being accessed by other functions.

Table 3-26 lists the strobe to be used when emulating a read of a specific word of the EEPROM auto-read feature.

Table 3-26. Strobes for EEARBC Auto-Read Emulation

EEPROM Word Emulated (In Hex)	Content	Port 0 Strobe	Port 1 Strobe	Port 2 Strobe	Port 3 Strobe
0:2	MAC address	VALID_CORE0	N/A	N/A	N/A
LAN1_start + 0:2		N/A	VALID_CORE1	N/A	N/A
LAN2_start + 0:2		N/A	N/A	VALID_CORE2	N/A
LAN3_start + 0:2		N/A	N/A	N/A	VALID_CORE3
0A/0F	Init control 1/2	VALID_CORE0	VALID_CORE1	VALID_CORE2	VALID_CORE3
0B/0C/0E[1]	Sub-system device and vendor	VALID_COMMON	VALID_COMMON	VALID_COMMON	VALID_COMMON
1E/1D[2]	Dummy device ID, Rev ID	VALID_COMMON	VALID_COMMON	VALID_COMMON	VALID_COMMON
21	Function control	VALID_COMMON	VALID_COMMON	VALID_COMMON	VALID_COMMON
0D[2]	Device ID port 0	VALID_CORE0	N/A	N/A	N/A
LAN1_start + 0D[2]	Device ID port 1	N/A	VALID_CORE1	N/A	N/A
LAN2_start + 0D[2]	Device ID port 2	N/A	N/A	VALID_CORE2	N/A
LAN3_start + 0D[2]	Device ID port 3	N/A	N/A	N/A	VALID_CORE3
20	SDP control LAN 0	VALID_CORE0	N/A	N/A	N/A
LAN1_start + 20	SDP control LAN 1	N/A	VALID_CORE1	N/A	N/A
LAN2_start + 20	SDP control LAN 2	N/A	N/A	VALID_CORE2	N/A
LAN3_start + 20	SDP control LAN 3	N/A	N/A	N/A	VALID_CORE3
24/13	Init control 3/4	VALID_CORE0	N/A	N/A	N/A
LAN1_start + 24/13	Init control 3/4	N/A	VALID_CORE1	N/A	N/A
LAN2_start + 24/13	Init control 3/4	N/A	N/A	VALID_CORE2	N/A
LAN3_start + 24/13	Init control 3/4	N/A	N/A	N/A	VALID_CORE3
14/15/16/18/19/1A/1B/22/28/29/2A/2B	PCIe and NC-SI configuration	VALID_COMMON	VALID_COMMON	VALID_COMMON	VALID_COMMON
1C/1F[3]	LED control port 0	VALID_CORE0	N/A	N/A	N/A
LAN1_start + 1C/1F[3]	LED control port 1	N/A	VALID_CORE1	N/A	N/A
LAN2_start + 1C/1F[3]	LED control port 2	N/A	N/A	VALID_CORE2	N/A
LAN3_start + 1C/1F[3]	LED control port 3	N/A	N/A	N/A	VALID_CORE3
2E[3]	Watchdog configuration	VALID_CORE0	VALID_CORE1	VALID_CORE2	VALID_CORE3
2F	VPD area	N/A	N/A	N/A	N/A

1. If word 0xA was accessed before the subsystem or subvendor ID are set, care must be taken that the load Subsystem IDs bit in word 0xA is set.
2. If word 0xA was accessed before one of the device IDs is set, care must be taken that the load Device IDs bit in word 0xA is set.
3. Part of the parameters that can be configured through the EEARBC register can be directly set through regular registers and thus usage of this mechanism is not needed for them. Specifically, words 0x1C, 0x1F and 0x2E controls only parameters that can be set through regular registers.

3.3.2 Shared EEPROM

The 82580 uses a single EEPROM device to configure hardware default parameters for all LAN devices, including Ethernet Individual Addresses (IA), LED behaviors, receive packet filters for manageability, wake-up capability, etc. Certain EEPROM words are used to specify hardware parameters that are LAN device-independent (such as those that affect circuit behavior). Other EEPROM words are associated with a specific LAN device. All LAN devices access the EEPROM to obtain their respective configuration settings.

3.3.2.1 EEPROM Deadlock Avoidance

The EEPROM is a shared resource between the following clients:

- Hardware auto-read.
- Port 0 LAN driver accesses.
- Port 1 LAN driver accesses.
- Port 2 LAN driver accesses.
- Port 3 LAN driver accesses.
- Firmware accesses.

All clients can access the EEPROM using parallel access, where hardware implements the actual access to the EEPROM. Hardware can schedule these accesses so that all clients get served without starvation.

However, software and hardware clients can access the EEPROM using bit banging. In this case, there is a request/grant mechanism that locks the EEPROM to the exclusive usage of one client. If this client is stuck (without releasing the lock), the other clients are not able to access the EEPROM. In order to avoid this, the 82580 implements a timeout mechanism, which releases the grant from a client that didn't toggle the EEPROM bit-bang interface for more than two seconds. The EEPROM deadlock avoidance mechanism is enabled when the *Deadlock Timeout Enable* bit in the *Initialization Control Word 1* EEPROM word is set to 1.

Notes:

1. If an agent that was granted access to the EEPROM for bit-bang access didn't toggle the bit bang interface for 500 ms, it should check if it still owns the interface before continuing the bit-banging.
2. Bit bang access to both Flash and EEPROM should not be done concurrently. SW should take ownership of both EEPROM and Flash Semaphore bits as described in Section 4.7.1 before doing a read or write bit bang operation to the EEPROM.

3.3.2.2 EEPROM Map Shared Words

The EEPROM map in Section 6.1 identifies those words configuring either LAN devices or the entire 82580 component as “all”. Those words configuring a specific LAN device parameter are identified by their LAN number.

The following EEPROM words warrant additional notes specifically related to quad-LAN support:

Table 3-27. Notes on EEPROM Words

Initialization Control 1, Initialization Control 2 (shared between LANs)	These EEPROM words specify hardware-default values for parameters that apply a single value to all LAN devices, such as link configuration parameters required for auto-negotiation, wake-up settings, PCIe bus advertised capabilities, etc.
Initialization Control 3, Initialization Control 4 (unique to each LAN)	This EEPROM word configures default values associated with each LAN device's hardware connections, including which link mode (internal PHY, SGMII, SerDes/1000BASE-BX, 1000BASE-KX) is used with this LAN device. Because a separate EEPROM word configures the defaults for each LAN, extra care must be taken to ensure that the EEPROM image does not specify a resource conflict.

3.3.3 Vital Product Data (VPD) Support

The EEPROM image might contain an area for VPD. This area is managed by the OEM vendor and doesn't influence the behavior of hardware. Word 0x2F of the EEPROM image contains a pointer to the VPD area in the EEPROM. A value of 0xFFFF means VPD is not supported and the VPD capability doesn't appear in the configuration space.

The VPD area should be aligned to a Dword boundary in the EEPROM.

The maximum area size is 256 bytes but can be smaller. The VPD block is built from a list of resources. A resource can be either large or small. The structure of these resources is listed in the following tables.

Table 3-28. Small Resource Structure

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

Table 3-29. Large Resource Structure

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

The 82580 parses the VPD structure during the auto-load process (power up and PCIe reset or warm reset) in order to detect the read-only and read/write area boundaries. The 82580 assumes the following VPD structure:

Table 3-30. VPD Structure

Tag	Structure Type	Length (Bytes)	Data	Resource Description
0x82	Large	Length of identifier string	Identifier	Identifier string.
0x90	Large	Length of RO area	RO data	VPD-R list containing one or more VPD keywords This part is optional and might not appear.
0x91	Large	Length of R/W area	RW data	VPD-W list containing one or more VPD keywords. This part is optional and might not appear.
0x78	Small	N/A	N/A	End tag.

Note: The VPD-R and VPD-W structures can be in any order.

If the 82580 doesn’t detect a value of 0x82 in the first byte of the VPD area, or the structure doesn’t follow the description listed in Table 3-30, it assumes the area is not programmed and the entire 256 bytes area is read only. If a VPD-W tag is found after the VPD-R tag, the area defined by it’s size is writable via the VPD structure. Refer to the PCI 3.0 specification (Appendix I) for details of the different tags.

In any case, the VPD area is accessible for read and write via the regular EEPROM mechanisms pending the EEPROM protection capabilities enabled. For example, if VPD is in the protected area, the VPD area is not accessible to the software device driver (parallel or serial), but accessible through the VPD mechanism. If the VPD area is not in the protected area, then the software device driver can access all of it for read and write.

The VPD area can be accessed through the PCIe configuration space VPD capability structure described in Section 9.5.5. Write accesses to a read-only area or any access outside of the VPD area via this structure are ignored.

Note: Write access to Dwords, which are only partially in the read/write area, are ignored. It is responsibility of VPD software to make the right alignment to enable a write to the entire area.

3.3.4 Flash Interface

3.3.4.1 Flash Interface Operation

The 82580 provides two different methods for software access to the Flash.

Using the legacy Flash transactions, the Flash is read from or written to each time the host CPU performs a read or a write operation to a memory location that is within the Flash address mapping or after a re-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 device 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 CPU accesses to a memory window defined in either:

1. The 82580's Flash Base Address register (PCIe Control register at offset 0x10 and 0x14. See Section 9.4.11).
2. The Expansion ROM Base Address register (PCIe Control register at offset 0x30. See Section 9.4.15).

The 82580 controls accesses to the Flash when it decodes a valid access.

Note: Flash read accesses must always be assembled by the 82580 each time the access is greater than a byte-wide access.

The 82580 byte reads or writes to the Flash take on the order of 2 μs. The 82580 will delay issuing PCIe credits during this time.

The 82580 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 (FLA) register. It can use this for reads, writes, or other Flash operations (accessing the Flash status register, erase, etc.).

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

1. Write a 1b to the *Flash Request* bit (*FLA.FL_REQ*).

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

3.3.4.2 Flash Write Control

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

After sending 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 *FLA* register, If bit (*FLA.FL_BUSY*) in this register is set, the current write did not finish. If bit (*FLA.FL_BUSY*) is clear then software can continue and write the next byte to the Flash.

3.3.4.3 Flash Erase Control

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

Hardware gets this command and sends the Erase command to the Flash. The erase process finishes by itself. Software should wait for the end of the erase process before any further access to the Flash. This can be checked by using the Flash write control mechanism previously described in Section 3.3.4.2.

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

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

3.3.5 Shared FLASH

The 82580 provides an interface to an external serial Flash/ROM memory device, as described in Section 2.1.2. This Flash/ROM device can be mapped into memory and/or IO address space for each LAN device through the use of Base Address Registers (BARs).

Clearing the *Flash Size* and *CSR_Size* fields in *Initialization Control Word 2* EEPROM word (Word 0x0F) to 0, disables Flash mapping to PCI space of all LAN ports via the *Flash Base Address* register. Setting the *LAN Boot Disable* bit in the per LAN port *Initialization Control 3* EEPROM word, disables Flash mapping to PCI space for LAN 0, LAN1, LAN2 and LAN 3 respectively, via the *Expansion ROM Base Address* register.

3.3.5.1 Flash Access Contention

The 82580 implements internal arbitration between Flash accesses initiated from the LAN 0, LAN 1, LAN 2 and LAN 3 devices. If accesses from these LAN devices are initiated during the same window, The first one is served first and only then the following devices are served in a Round Robin fashion.

Note: The 82580 does not synchronize between the entities accessing the Flash. Contentions caused by one entity reading and the other modifying the same location is possible.

To avoid this contention, accesses from the 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 by the nature of the software sequence.

3.3.5.2 Flash Deadlock Avoidance

The Flash is a shared resource between the following clients:

- Port 0 LAN driver accesses.
- Port 1 LAN driver accesses.
- Port 2 LAN driver accesses.
- Port 3 LAN driver accesses.
- BIOS parallel access via expansion ROM mechanism.
- Firmware accesses.

All clients can access the flash using parallel access, where hardware implements the actual access to the Flash. Hardware can schedule these accesses so that all the clients get served without starvation.

However, the driver and firmware clients can access the serial Flash using bit banging. In this case, there is a request/grant mechanism that locks the serial Flash to the exclusive usage of one client. If this client is stuck without releasing the lock, the other clients are unable to access the Flash. In order to avoid this, the 82580 implements a time-out mechanism that releases the grant from a client that doesn’t toggle the Flash bit-bang interface for more than two seconds.

Notes:

1. If an agent that was granted access to the Flash for bit-bang access doesn’t toggle the bit-bang interface for 500 ms, it should check that it still owns the interface before continuing the bit banging.
2. Bit bang access to both Flash and EEPROM should not be done concurrently. Software should take ownership of both EEPROM and Flash Semaphore bits as described in Section 4.7.1 before doing a read or write bit bang operation to the Flash.

This mode is enabled by bit five in word 0xA of the EEPROM.

3.4 Configurable I/O Pins

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

The 82580 has four software-defined pins (SDP pins) 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. For example, eight SDP pins cannot be associated with a single LAN device. These pins can each be individually configurable to act as either input or output pins. The default direction of each of the four pins is configurable via the EEPROM as well as the default value of any pins configured as outputs. To avoid signal contention, all four pins are set as input pins until after the EEPROM configuration has been loaded.

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

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

The SDPs can be used for special purpose mechanisms such as watch dog indication (see Section 3.4.2 for details) or IEEE 1588 support (see Section 7.9.4 for details).

3.4.2 Software Watchdog

In some situations it might be useful to give an indication to manageability firmware or to external devices that the 82580 hardware or the software device driver is not functional. For example, in a pass-through NIC, the 82580 might be bypassed if it is not functional. In order to provide this functionality, a watchdog mechanism is used. This mechanism can be enabled by default, according to EEPROM configuration.

Once the host driver is up and it determines that hardware is functional, it might reset the watchdog timer to indicate that the 82580 is functional. The software device driver should then re-arm the timer periodically. If the timer is not re-armed after pre-programmed timeout, an interrupt is sent to firmware and a pre-programmed SDPx_0 pin (either SDP0_0, SDP1_0, SDP2_0 or SDP3_0) is asserted. Note that the SDP indication is shared between the ports. Additionally the *ICR.Software WD* bit can be set to give an interrupt to the driver when the timeout is reached.

The SDPx_0 pin on which the watchdog timeout is indicated, is defined via the *CTRL.SDP0_WDE bit* on the relevant port. In this mode the *CTRL.SDP0_IODIR* should be set to output. The *CTRL.SDP0_DATA* bit indicates the polarity of the indication. Setting the *CTRL.SDP0_WDE* bit in one of the ports causes the watchdog timeout indication of all ports to be routed to this SDPx_0 pin.

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

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

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

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

3.4.2.1 Watchdog Rearm

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

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

3.4.3 LEDs

The 82580 provides four LEDs per port that can be used to indicate different statuses of the traffic. The default setup of the LEDs is done via EEPROM word offsets 0x1C and 0x1F from start of relevant LAN port section (LAN port 0, 1, 2 and 3). This setup is reflected in the *LEDCTL* register of each port. Each software device driver can change its setup individually. For each of the LEDs the following parameters can be defined:

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

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

3.5 Network Interfaces

3.5.1 Overview

The 82580 MAC provides a complete CSMA/CD function supporting IEEE 802.3 (10 Mb/s), 802.3u (100 Mb/s), 802.3z and 802.3ab (1000 Mb/s) implementations. The 82580 performs all of the functions required for transmission, reception, and collision handling called out in the standards.

Each 82580 MAC can be configured to use a different media interface. While the most likely application is expected to be based on use of the internal copper PHY, The 82580 supports the following potential configurations:

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

Selection between the various configurations is programmable via each MAC's Extended Device Control register (*CTRL_EXT.LINK_MODE* bits) and default is set via EEPROM settings. Table 3-31 lists the encoding on the *LINK_MODE* field for each of the modes.

Table 3-31. Link Mode Encoding

Link Mode	82580 Mode
00b	Internal PHY
01b	1000BASE-KX
10b	SGMII
11b	SerDes/1000BASE-BX

The GMII/MII interface, used to communicate between the MAC and the internal PHY or the SGMII PCS, supports 10/100/1000 Mb/s operation, with both half- and full-duplex operation at 10/100 Mb/s, and only full-duplex operation at 1000 Mb/s.

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

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

- No Tx clock
- AC coupling only

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

3.5.2 MAC Functionality

3.5.2.1 Internal GMII/MII Interface

The 82580’s MAC and PHY/PCS communicate through an internal GMII/MII interface that can be configured for either 1000 Mb/s operation (GMII) or 10/100 Mb/s (MII) mode of operation. For proper network operation, both the MAC and PHY must be properly configured (either explicitly via software or via hardware auto-negotiation) to identical speed and duplex settings.

All MAC configuration is performed using Device Control registers mapped into system memory or I/O space; an internal MDIO/MDC interface, accessible via software, is used to configure the Internal PHY. In addition an external MDIO/MDC interface is available to configure external PHY’s that are connected to the 82580 via the SGMII interface.

3.5.2.2 MDIO/MDC PHY Management Interface

The 82580 implements an IEEE 802.3 MII Management Interface (also known as the Management Data Input/Output or MDIO Interface) between the MAC and a PHY. This interface provides the MAC and software the ability to monitor and control the state of the PHY. The MDIO interface defines a physical connection, a special protocol that runs across the connection, and an internal set of addressable registers. The interface consists of a data line (MDIO) and clock line (MDC), which are accessible by software via the MAC register space.

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

Software can use MDIO accesses to read or write registers of the internal PHY or an external SGMII PHY, by accessing the 82580's *MDIC* register (see Section 8.2.4). MDIO configuration setup (Internal/External PHY, PHY Address and Shared MDIO) is defined in the *MDICNFG* register (see Section 8.2.5).

When working in SGMII/SerDes mode, the external PHY (if it exists) can be accessed either through MDC/MDIO as previously described, or via a two wire I^{2}C interface bus using the *I2CCMD* register (see Section 8.18.8). The two wire interface bus or the MDC/MDIO bus are connected via the same pins, and thus are mutually exclusive. In order to be able to control an external device, either by I^{2}C or MDC/MDIO, the 2 wires *SFP Enable* bit in *Initialization Control 3* EEPROM word, that’s loaded into the *CTRL_EXT.I2C Enabled* register bit, should be set.

As the MDC/MDIO command can be targeted either to the internal PHY or to an external bus, the *MDICNFG.destination* bit is used to define the target of the transaction. Following reset the value of the *MDICNFG.destination* bit is loaded from the *External MDIO* bit in the *Initialization Control 3* EEPROM word. When the *MDICNFG.destination* is clear the MDIO access is always to the internal PHY and the PHY address is ignored.

Each port has its own MDC/MDIO or two wire interface bus. However, the MDC/MDIO bus of LAN port 0 may be shared by all ports configured to external PHY operation (*MDICNFG.destination* set to 1), to allow control of a multi PHY chip with a single MDC/MDIO bus.

MDIO operation using a shared bus or a separate bus is controlled by the *MDICNFG.Com_MDIO* bit that's loaded from *Initialization Control 3* EEPROM word following reset. The external port PHY Address is written in the *MDICNFG.PHYADD* register field and is loaded from the *Initialization Control 4* EEPROM word following reset.

3.5.2.2.1 Detection of External I^{2}C or MDIO Connection

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

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

3.5.2.2.2 MDIC and MDICNFG register usage

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

1. If default MDICNFG register values loaded from EEPROM need to be updated. The processor performs a PCIe write access to the *MDICNFG* register to define the:
   - — PHYADD = Address of external PHY.
   - — Destination = Internal or external PHY.
   - — Com_MDIO = Shared or separate MDIO external PHY connection.
2. The processor performs a PCIe write cycle to the MDIC register with:
   - — Ready = 0b
   - — Interrupt Enable set to 1b or 0b
   - — Opcode = 10b (read)
   - — REGADD = Register address of the specific register to be accessed (0 through 31).
3. The MAC applies the following sequence on the MDIO signal to the PHY:

   <PREAMBLE><01><10><PHYADD><REGADD><Z> where Z stands for the MAC tri-stating the MDIO signal.
4. The PHY returns the following sequence on the MDIO signal:

   <0><DATA><IDLE>.
5. The MAC discards the leading bit and places the following 16 data bits in the MII register.
6. The 82580 asserts an interrupt indicating MDIO "Done" if the *Interrupt Enable* bit was set.
7. The 82580 sets the *Ready* bit in the MDIC register indicating the Read is complete.
8. The processor might read the data from the MDIC register and issue a new MDIO command.

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

1. If default MDICNFG register values loaded from EEPROM need to be updated. The processor performs a PCIe write cycle to the MDICNFG register to define the:
   - PHYADD = Address of external PHY.
   - Destination = Internal or external PHY.
   - Com_MDIO = Shared or separate MDIO external PHY connection.
2. The processor performs a PCIe write cycle to the MDIC register with:
   - Ready = 0b.
   - Interrupt Enable set to 1b or 0b.
   - Opcode = 01b (write).
   - REGADD = Register address of the specific register to be accessed (0 through 31).
   - Data = Specific data for desired control of the PHY.
3. The MAC applies the following sequence on the MDIO signal to the PHY:

   <PREAMBLE><01><01><PHYADD><REGADD><10><DATA><IDLE>
4. The 82580 asserts an interrupt indicating MDIO "Done" if the *Interrupt Enable* bit was set.
5. The 82580 sets the *Ready* bit in the MDIC register to indicate that the write operation completed.
6. The CPU might issue a new MDIO command.

Note: A MDIO read or write might take as long as 64 μs from the processor write to the *Ready* bit assertion. When a shared MDC/MDIO bus is used, each transaction can take up to 256 μs to complete if other ports are using the bus concurrently.

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

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

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

3.5.2.3 Duplex Operation with Copper PHY

The 82580 supports half-duplex and full-duplex 10/100 Mb/s MII mode either through the internal copper PHY or SGMII interface. However, only full-duplex mode is supported when SerDes, 1000BASE-BX or 1000BASE-KX modes are used or in any 1000 Mb/s connection.

Configuration of the duplex operation of the 82580 can either be forced or determined via the auto-negotiation process. See Section 3.5.4.4 for details on link configuration setup and resolution.

3.5.2.3.1 Full Duplex

All aspects of the IEEE 802.3, 802.3u, 802.3z, and 802.3ab specifications are supported in full-duplex operation. Full-duplex operation is enabled by several mechanisms, depending on the speed configuration of the 82580 and the specific capabilities of the link partner used in the application. During full-duplex operation, the 82580 can transmit and receive packets simultaneously across the link interface.

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

3.5.2.3.2 Half Duplex

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

In the case of a collision, the PHY/SGMII detects the collision and asserts the COL signal to the MAC. Frame transmission stops within four link clock times and then the 82580 sends a JAM sequence onto the link. After the end of a collided transmission, the 82580 backs off and attempts to re-transmit per the standard CSMA/CD method.

Note: The re-transmissions are done from the data stored internally in the 82580 MAC transmit packet buffer (no re-access to the data in host memory is performed).

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

In the case of a successful transmission, the 82580 is ready to transmit any other frame(s) queued in the MAC's transmit FIFO, after the minimum inter-frame spacing (IFS) of the link has elapsed.

During transmit, the PHY is expected to signal a carrier-sense (assert the CRS signal) back to the MAC before one slot time has elapsed. The transmission completes successfully even if the PHY fails to indicate CRS within the slot time window. If this situation occurs, the PHY can either be configured incorrectly or be in a link down situation. Such an event is counted in the Transmit without CRS statistic register (see Section 8.19.12).

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

The 82580 can be configured to follow either SGMII, SerDes/1000BASE-BX or 1000BASE-KX standards. When in SGMII mode, the 82580 can be configured to operate in 1 Gb/s, 100 Mb/s or 10 Mb/s speeds. When in the 10/100 Mb/s speed, the 82580 can be configured to half-duplex mode of operation. When configured for SerDes/1000BASE-BX or 1000BASE-KX operation, the port supports only 1 Gb/s, full-duplex operation. Since the serial interfaces are defined as differential signals, internally the hardware has analog and digital blocks. Following is the initialization/configuration sequence for the analog and digital blocks.

3.5.3.1 SerDes Analog Block

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

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

The link setup for SerDes/1000BASE-BX, 1000BASE-KX and SGMII are described in sections 3.5.4.1, 3.5.4.2 and 3.5.4.3 respectively.

3.5.3.3 GbE Physical Coding Sub-Layer (PCS)

The 82580 integrates the 802.3z PCS function on-chip. The on-chip PCS circuitry is used when the link interface is configured for SerDes/1000BASE-BX, 1000BASE-KX or SGMII operation and is bypassed for internal PHY mode.

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

3.5.3.3.1 8B10B Encoding/Decoding

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

3.5.3.3.2 Code Groups and Ordered Sets

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

Table 3-32. Brief Description of Defined Ordered Sets

Code	Ordered_Set	# of Code Groups	Usage
/C/	Configuration	4	General reference to configuration ordered sets, either /C1/ or /C2/, which is used during auto-negotiation to advertise and negotiate link operation information between link partners. Last 2 code groups contain configuration base and next page registers.
/C1/	Configuration 1	4	See /C/. Differs from /C2/ in 2nd code group for maintaining proper signaling disparity[1].
/C2/	Configuration 2	4	See /C/. Differs from /C1/ in 2nd code group for maintaining proper signaling disparity[1].
/I/	IDLE	2	General reference to idle ordered sets. Idle characters are continually transmitted by the end stations and are replaced by encapsulated packet data. The transitions in the idle stream enable the SerDes to maintain clock and symbol synchronization between link partners.
/I1/	IDLE 1	2	See /I/. Differs from /I2/ in 2nd code group for maintaining proper signaling disparity[1].
/I2/	IDLE 2	2	See /I/. Differs from /I1/ in 2nd code group for maintaining proper signaling disparity[1].
/R/	Carrier_Extend	1	This ordered set is used to indicate carrier extension to the receiving PCS. It is also used as part of the end_of_packet encapsulation delimiter as well as IPG for packets in a burst of packets.

Table 3-32. Brief Description of Defined Ordered Sets (Continued)

Code	Ordered_Set	# of Code Groups	Usage
/S/	Start_of_Packet	1	The SPD (start_of_packet delimiter) ordered set is used to indicate the starting boundary of a packet transmission. This symbol replaces the last byte of the preamble received from the MAC layer.
/T/	End_of_Packet	1	The EPD (end_of_packet delimiter) is comprised of three ordered sets. The /T/ symbol is always the first of these and indicates the ending boundary of a packet.
/V/	Error_Propagation	1	The /V/ ordered set is used by the PCS to indicate error propagation between stations. This is normally intended to be used by repeaters to indicate collisions.

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

3.5.4 Auto-Negotiation and Link Setup Features

The method for configuring the link between two link partners is highly dependent on the mode of operation as well as the functionality provided by the specific physical layer device (PHY or SerDes). In SerDes mode, the 82580 provides the complete PCS and Auto-negotiation functionality as defined in IEEE802.3 clause 36 and clause 37. In internal PHY mode, the PCS and IEEE802.3 clause 28 and clause 40 auto-negotiation functions are maintained within the PHY. In SGMII mode, the 82580 supports the SGMII link auto-negotiation process, whereas the link auto-negotiation, as defined in IEEE802.3 clause 28 and clause 40, is done by the external PHY. In 1000BASE-KX mode, the 82580 supports only parallel detect of 1000BASE-KX signaling and does not support the full Auto-Negotiation for Backplane Ethernet protocol as defined in IEEE802.3ap clause 73.

Configuring the link can be accomplished by several methods ranging from software forcing link settings, software-controlled negotiation, MAC-controlled auto-negotiation, to auto-negotiation initiated by a PHY. The following sections describe processes of bringing the link up including configuration of the 82580 and the transceiver, as well as the various methods of determining duplex and speed configuration.

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

When operating in SerDes/1000BASE-BX mode, the PCS layer performs auto-negotiation per clause 37 of the 802.3z standard. The transceiver used in this mode does not participate in the auto-negotiation process as all aspects of auto-negotiation are controlled by the 82580.

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

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

When operating in 1000BASE-KX mode the 82580 performs parallel detect of 1000BASE-KX operation but does not implement the full Auto-Negotiation for Backplane Ethernet sequence as defined in IEEE802.3ap clause 73.

3.5.4.1 SerDes/1000BASE-BX Link Configuration

When using SerDes link mode, link mode configuration can be performed using the PCS function in the 82580. The hardware supports both hardware and software auto-negotiation methods for determining the link configuration, as well as allowing for a manual configuration to force the link. Hardware auto-negotiation is the preferred method.

3.5.4.1.1 Signal Detect Indication

When the *CONNSW.ENRGSRC* bit is set to 1, the SRDS_0/1/2/3_SIG_DET pins can be connected to a Signal Detect or loss-of-signal (LOS) output that indicates when no laser light is being received when the 82580 is used in a 1000BASE-SX or -LX implementation (SerDes operation). It prevents false carrier cases occurring when transmission by a non connected port couples in to the input. Unfortunately, there is no standard polarity for this signal coming from different manufacturers. The *CTRL.ILOS* bit provides for inversion of the signal from different external optical module vendors, and should be set when the external optical module provides a negative-true loss-of-signal.

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

3.5.4.1.2 MAC Link Speed

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

3.5.4.1.3 SerDes Mode Auto-Negotiation

In SerDes mode, after power up or the 82580 reset via PE_RST_N, the 82580 initiates IEEE802.3 clause 37 auto-negotiation based on the default settings in the device control and transmit configuration or PCS Link Control Word registers, as well as settings read from the EEPROM. If enabled in the EEPROM, the 82580 immediately performs auto-negotiation.

TBI mode auto-negotiation, as defined in clause 37 of the IEEE 802.3z standard, provides a protocol for two devices to advertise and negotiate a common operational mode across a GbE link. The 82580 fully supports the IEEE 802.3z auto-negotiation function when using the on-chip PCS and internal SerDes.

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

- Duplex resolution (even though the 82580 MAC only supports full-duplex in SerDes mode).
- Flow control configuration.

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

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

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

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

Auto-negotiation can be initiated by the following:

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

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

Software must configure the *PCS_ANADV* fields to the desired advertised base page. The bits in the Device Control register are not mapped to the *txConfigWord* field in hardware until after auto-negotiation completes. Table 3-33 lists the mapping of the *PCS_ANADV* fields to the Config_reg Base Page encoding per clause 37 of the standard.

Table 3-33. 802.3z Advertised Base Page Mapping

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

The partner advertisement can be seen in the *PCS_ LPAB* and *PCS_ LPABNP* registers.

3.5.4.1.4 Forcing Link-up in SerDes Mode

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

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

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

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

3.5.4.1.5 HW Detection of Non-Auto-Negotiation Partner

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

3.5.4.2 1000BASE-KX Link Configuration

When using 1000BASE-KX link mode, link mode configuration is forced by software since the 82580 does not support IEEE802.3 clause 73 backplane auto-negotiation.

3.5.4.2.1 MAC Link Speed

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

3.5.4.2.2 1000BASE-KX Auto-Negotiation

The 82580 only supports parallel detection of the 1000BASE-KX link and does not support the full IEEE802.3ap clause 73 backplane auto-negotiation protocol.

3.5.4.2.3 Forcing Link-up in 1000BASE-KX Mode

In 1000BASE-KX mode (*EXT_CTRL.LINK_MODE* = 001b) the 82580 should always operates in force link mode (*CTRL.SLU* bit is set to 1). The MAC PCS logic is placed in a link-up state once energy indication is received, implying that a valid signal is being received by the 1000BASE-KX circuitry. When in the link-up state PCS logic can lock on incoming characters.

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

3.5.4.2.4 1000BASE-KX HW Detection of Link Partner

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

3.5.4.3 SGMII Link Configuration

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

3.5.4.3.1 SGMII Auto-Negotiation

This auto-negotiation process is not dependent on the SRDS_[n]_SIG_DET signal, as this signal indicates optical module signal detection and is not relevant in SGMII mode.

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

- Link status
- Speed
- Duplex

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

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

Auto-negotiation can be initiated by the following:

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

Auto-negotiation determines the 82580 operation with respect to speed and duplex settings. These negotiated capabilities override advertised and software controlled device configuration.

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

3.5.4.3.2 Forcing Link in SGMII mode

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

3.5.4.3.3 MAC Speed Resolution

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

3.5.4.4 Copper PHY Link Configuration

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

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

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

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

- Speed: - 10/100/1000 Mb/s
- Duplex: - Full or half
- Flow control operation

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

Note: If flow control is enabled in the 82580, the settings for the desired flow control behavior must be set by software in the PHY registers and auto-negotiation restarted. After auto-negotiation completes, the software device driver must read the PHY registers to determine the resolved flow control behavior of the link and reflect these in the MAC register settings (*CTRL.TFCE* and *CTRL.RFCE*).

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

3.5.4.4.2 MAC Speed Resolution

For proper link operation, both the MAC and PHY must be configured for the same speed of link operation. The speed of the link can be determined and set by several methods with the 82580. These include:

- Software-forced configuration of the MAC speed setting based on PHY indications, which might be determined as follows:
  - Software reads of PHY registers directly to determine the PHY's auto-negotiated speed
  - Software reads the PHY's internal PHY-to-MAC speed indication (SPD_IND) using the MAC *STATUS.SPEED* register
- Software asks the MAC to attempt to auto-detect the PHY speed from the PHY-to-MAC RX_CLK, then programs the MAC speed accordingly
- MAC automatically detects and sets the link speed of the MAC based on PHY indications by using the PHY's internal PHY-to-MAC speed indication (SPD_IND)

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

3.5.4.4.2.1 Forcing MAC Speed

There might be circumstances when the software device driver must forcibly set the link speed of the MAC. This can occur when the link is manually configured. To force the MAC speed, the software device driver must set the *CTRL.FRCSPD* (force-speed) bit to 1b and then write the speed bits in the Device Control register (*CTRL.SPEED*) to the desired speed setting. See Section 8.2.1 for details.

Note: Forcing the MAC speed using *CTRL.FRCSPD* overrides all other mechanisms for configuring the MAC speed and can yield non-functional links if the MAC and PHY are not operating at the same speed/configuration.

When forcing the 82580 to a specific speed configuration, the software device driver must also ensure the PHY is configured to a speed setting consistent with MAC speed settings. This implies that software must access the PHY registers to either force the PHY speed or to read the PHY status register bits that indicate link speed of the PHY.

Note: Forcing speed settings by *CTRL.SPEED* can also be accomplished by setting the *CTRL_EXT.SPD_BYPS* bit. This bit bypasses the MAC's internal clock switching logic and enables the software device driver complete control of when the speed setting takes place. The *CTRL.FRCSPD* bit uses the MAC's internal clock switching logic, which does delay the affect of the speed change.

3.5.4.4.2.2 Using Internal PHY Direct Link-Speed Indication

The 82580’s internal PHY provides a direct internal indication of its speed to the MAC (SPD_IND). When using the internal PHY, the most direct method for determining the PHY link speed and either manually or automatically configuring the MAC speed is based on these direct speed indications.

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

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

3.5.4.4.3 MAC Full-/Half- Duplex Resolution

The duplex configuration of the link is also resolved by the PHY during the auto-negotiation process. The 82580's internal PHY provides an internal indication to the MAC of the resolved duplex configuration using an internal full-duplex indication (FDX).

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

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

3.5.4.4.4 Using PHY Registers

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

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

The set of PHY management registers required for all PHY devices can be found in the IEEE P802.3ab standard. The registers for the 82580 PHY are described in Section 3.5.8.

3.5.4.4.5 Comments Regarding Forcing Link

Forcing link in GMII/MII mode (internal PHY) requires the software device driver to configure both the MAC and PHY in a consistent manner with respect to each other as well as the link partner. After initialization, the software device driver configures the desired modes in the MAC, then accesses the PHY registers to set the PHY to the same configuration.

Before enabling the link, the speed and duplex settings of the MAC can be forced by software using the *CTRL.FRCSPD*, *CTRL.FRCDPX*, *CTRL.SPEED*, and *CTRL.FD* bits. After the PHY and MAC have both been configured, the software device driver should write a 1b to the *CTRL.SLU* bit.

3.5.4.5 Loss of Signal/Link Status Indication

For all modes of operation, an LOS/LINK signal provides an indication of physical link status to the MAC. When the MAC is configured for optical SerDes mode, the input reflects loss-of-signal connection from the optics. In backplane mode, where there is no LOS external indication, an internal indication from the SerDes receiver can be used. In SFP systems the LOS indication from the SFP can be used. In internal PHY mode, this signal from the PHY indicates whether the link is up or down; typically indicated after successful auto-negotiation. Assuming that the MAC has been configured with *CTRL.SLU*=1b, the MAC status bit *STATUS.LU*, when read, generally reflects whether the PHY or SerDes has link (except under forced-link setup where even the PHY link indication might have been forced).

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

3.5.5 Ethernet Flow Control (FC)

The 82580 supports flow control as defined in 802.3x as well as the specific operation of asymmetrical flow control defined by 802.3z.

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

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

The following registers are defined for the implementation of flow control:

- *CTRL.RFCE* field is used to enable reception of legacy flow control packets and reaction to them.
- *CTRL.TFCE* field is used to enable transmission of legacy flow control packets.
- Flow Control Address Low, High (*FCAL/H*) - 6-byte flow control multicast address
- Flow Control Type (*FCT*) 16-bit field to indicate flow control type
- Flow Control bits in Device Control (*CTRL*) register - Enables flow control modes.
- Discard PAUSE Frames (*DPF*) and Pass MAC Control Frames (*PMCF*) in *RCTL* - controls the forwarding of control packets to the host.
- Flow Control Receive Threshold High (*FCRTH0*) - A 13-bit high watermark indicating receive buffer fullness. A single watermark is used in link FC mode.
- DMA Coalescing Receive Threshold High (*FCRTC*) - A 13-bit high watermark indicating receive buffer fullness when in DMA coalescing and TX buffer is empty. Value in this register can be higher than value placed in the *FCRTH0* register since watermark needs to be set to allow for reception of only a maximum sized RX packet before XOFF flow control takes effect and reception is stopped (See Table 3-37 for information on flow control threshold calculation).
- Flow Control Receive Threshold Low (*FCRTL0*) - A 13-bit low watermark indicating receive buffer emptiness. A single watermark is used in link FC mode.
- Flow Control Transmit Timer Value (*FCTTV*) - a set of 16-bit timer values to include in transmitted PAUSE frame. A single timer is used in Link FC mode.
- Flow Control Refresh Threshold Value (*FCRTV*) - 16-bit PAUSE refresh threshold value

3.5.5.1 MAC Control Frames and Receiving Flow Control Packets

3.5.5.1.1 Structure of 802.3X FC Packets

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

1. A match on the 6-byte multicast address for MAC control frames or to the station address of the 82580 (Receive Address Register 0).
2. A match on the type field.
3. A comparison of the MAC *Control Op-Code* field.

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

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

The final check for a valid PAUSE frame is the MAC control op-code. At this time only the PAUSE control frame op-code is defined. It has a value of 0x0001.

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

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

Table 3-34 lists the structure of a 802.3X FC packet

Table 3-34. 802.3X Packet Format

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

3.5.5.1.2 Operation and Rules

The 82580 operates in Link FC.

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

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

Once the receiver has validated receiving an XOFF, or PAUSE frame, the 82580 performs the following:

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

Resumption of transmission might occur under the following conditions:

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

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

3.5.5.1.3 Timing Considerations

When operating at 1 Gb/s line speed, the 82580 must not begin to transmit a (new) frame more than two pause-quantum-bit times after receiving a valid link XOFF frame, as measured at the wires. A pause quantum is 512-bit times.

When operating in full duplex at 100 Mb/s or 1 Gb/s line speeds, the 82580 must not begin to transmit a (new) frame more than 576-bit times after receiving a valid link XOFF frame, as measured at the wire.

3.5.5.2 PAUSE and MAC Control Frames Forwarding

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

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

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

Table 3-35 lists the behavior of the *DPF* bit.

Table 3-35. Forwarding of PAUSE Packet to Host (*DPF* Bit)

RFCE	DPF	Are FC Packets Forwarded to Host?
0	X	Yes. Packets needs to pass the L2 filters (see Section 7.1.2.1).[1]
1	0	Yes. Packets needs to pass the L2 filters (see Section 7.1.2.1).

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

Table 3-36. Transfer of Non-PAUSE Control Packets to Host (*PMCF* Bit)

RFCE	PMCF	Are Non-FC MAC Control Packets Forwarded to Host?
0	X	Yes. Packets needs to pass the L2 filters (see Section 7.1.2.1).
X	0	Yes. Packets needs to pass the L2 filters (see Section 7.1.2.1).

3.5.5.3 Transmission of PAUSE Frames

The 82580 generates PAUSE packets to ensure there is enough space in its receive packet buffers to avoid packet drop. The 82580 monitors the fullness of its receive packet buffers and compares it with the contents of a programmable threshold. When the threshold is reached, the 82580 sends a PAUSE frame. The 82580 also supports the sending of link Flow Control (FC).

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

The transmission of flow control frames should only be enabled in full-duplex mode per the IEEE 802.3 standard. Software should ensure that the transmission of flow control packets is disabled when the 82580 is operating in half-duplex mode.

3.5.5.3.1 Operation and Rules

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

the 82580 sends a PAUSE frame when Rx packet buffer is full above the high threshold defined in the Flow Control Receive Threshold High (*FCRT0.RTH*) register field. When the threshold is reached, the 82580 sends a PAUSE frame with its pause time field equal to *FCTTV*. The threshold should be large enough to overcome the worst case latency from the time that crossing the threshold is sensed till packets are not received from the link partner. The Flow Control Receive Threshold High value should be calculated as follows:

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

Parameter values to be used for calculating the *FCRTH0.RTH* value can be found in Table 3-37.

Table 3-37. Flow Control Receive Threshold High (*FCRTH0.RTH*) Value Calculation

Latency Parameter	Affected by	Parameter Value
Threshold Cross to XOFF Transmission	Max packet size	Max packet Size * 1.25
XOFF Reception to Link Partner response	Max packet size	Max packet size
Round trip latency	The latencies on the wire and the LAN devices at both sides of the wire	320 Byte (for 1000Base-T operation).

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

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

The *FCRTC.RTH_Coal* is used as the high watermark to generate XOFF flow control packets when the internal TX buffer is empty and the 82580 is executing DMA coalescing. In this case no delay to transmission of flow control packet exists so it's possible to increase level of watermark before issuing a XOFF flow control frame.

After transmitting a PAUSE frame the 82580 activates an internal shadow counter that reflects the link partner pause timeout counter. When the counter reaches the value indicated in the *FCRTV* register, then, if the PAUSE condition is still valid (meaning that the buffer fullness is still above the high watermark), a XOFF message is sent again.

Once the receive buffer fullness reaches the low water mark, the 82580 sends a XON message (a PAUSE frame with a timer value of zero). Software enables this capability with the *XONE* field of the *FCRTL*.

The 82580 sends an additional PAUSE frame if it has previously sent one and the packet buffer overflows. This is intended to minimize the amount of packets dropped if the first PAUSE frame did not reach its target.

3.5.5.3.2 Software Initiated PAUSE Frame Transmission

The 82580 has the added capability to transmit an XOFF frame via software. This is accomplished by software writing a 1b to the *SWXOFF* bit of the Transmit Control register. Once this bit is set, hardware initiates the transmission of a PAUSE frame in a manner similar to that automatically generated by hardware.

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

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

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

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

XOFF transmission is not supported in 802.3x for half-duplex links. Software should not initiate an XOFF or XON transmission if the 82580 is configured for half-duplex operation.

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

3.5.5.4 IPG Control and Pacing

The 82580 supports the following modes of controlling IPG duration:

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

3.5.5.4.1 Fixed IPG Extension

The 82580 allows controlling of the IPG duration. The IPGT configuration field enables an extension of IPG in 4-byte increments. One possible use of this capability is to allow the insertion of bytes into the transmit packet after it has been transmitted by the 82580 without violating the minimum IPG requirements. For example, a security device connected in series to the 82580 might add security headers to transmit packets before the packets are transmitted on the network.

3.5.6 Loopback Support

3.5.6.1 General

The 82580 supports the following types of internal loopback in the LAN interfaces:

- MAC Loopback (Point 1)
- PHY Loopback (Point 2)
- SerDes, SGMII or 1000BASE-KX Loopback (Point 3)
- External PHY Loopback (Point 4)

By setting the device to loopback mode, packets that are transmitted towards the line will be looped back to the host. The 82580 is fully functional in these modes, just not transmitting data over the lines. Figure 3-4 shows the points of loopback.

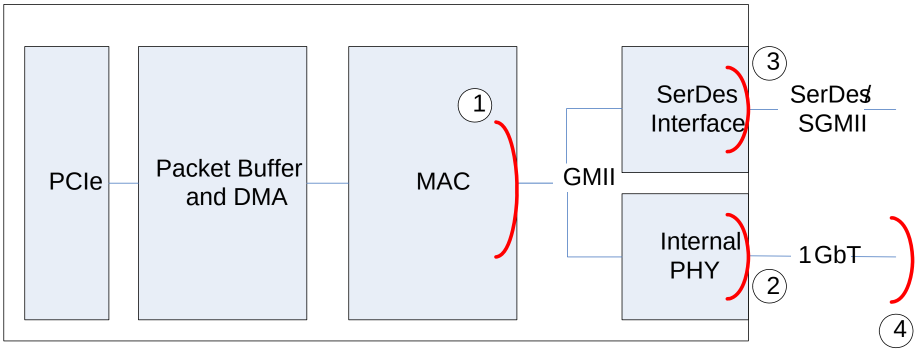


Figure 3-4. 82580 Loopback Modes

3.5.6.2 MAC Loopback

In MAC loopback, the PHY and SerDes blocks are not functional and data is looped back before these blocks. MAC loopback is operational only when working in PHY mode (*CTRL_EXT.LINK_MODE* = 000b).

3.5.6.2.1 Setting the 82580 to MAC loopback Mode

The following procedure should be used to put the 82580 in MAC loopback mode:

- Set *RCTL.LBM* to 2'b01 (bits 7:6)
- Set *CTRL.SLU* (bit 6, should be set by default)
- Set *CTRL.FRCSPD* & *FRCDPLX* (bits 11&12)
- Set *CTRL.SPEED* to 2'b10 (1G) and *CTRL.FD*
- Set *CTRL.ILOS*
- Disable Auto-Negotiation in the PHY control register (Address 0 in the PHY):
  — Clear Auto Neg enable bit (Bit 12)
- Set CTRL_EXT.LINK_MODE = 11b

Filter configuration and other TX/RX processes are the same as in normal mode.

Note: This configuration is for a case were there is no link in the PHY. If there is a link, *ILOS* bit should be cleared.

3.5.6.3 Internal PHY Loopback

In PHY loopback the SerDes block is not functional and data is looped back at the end of the PHY functionality. This means all the design that is functional in copper mode, is involved in the loopback

3.5.6.3.1 Setting the 82580 to Internal PHY loopback Mode

The following procedure should be used to place the 82580 in PHY loopback mode on LAN port 0,1,2 or 3:

- Set Link mode to Internal PHY: *CTRL_EXT.LINK_MODE* = 00b.
- In PHY control register (*PCTRL* - Address 0 in the PHY):

  - — Set Duplex mode (bit 8)
  - — Set Loopback bit (Bit 14)
  - — Clear Auto Neg enable bit (Bit 12)
  - — Set speed using bits 6 and 13 as described in EAS.
  - — Register value should be:

    For 10 Mbps 0x4100

    For 100 Mbps 0x6100

    For 1000 Mbps 0x4140.

- Determine the exact type of loopback using the loopback control register (*PHLBKC* - address 19d, See Section 8.24.2.14).

Note: While in MII loopback mode *PHLBKC.Force Link Status should* be set to 1 to receive valid link state and be able to Transmit and Receive normally.

Note: Make sure a Configure command is re-issued (loopback bits set to 00b) to cancel the loopback mode.

3.5.6.4 SerDes, SGMII and 1000BASE-KX Loopback

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

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

3.5.6.4.1 Setting the 82580 to SerDes/1000BASE-BX, SGMII or 1000BASE-KX loopback Mode

The following procedure should be used to place the 82580 in SerDes loopback mode:

- Set Link mode to either SerDes, SGMII or 1000BASE-KX by:
  - — 1000BASE-KX: *CTRL_EXT.LINK_MODE* = 01b
  - — SGMII: *CTRL_EXT.LINK_MODE* = 10b
  - — SerDes/1000BASE-BX: *CTRL_EXT.LINK_MODE* = 11b
- Configure SERDES to loopback: *RCTL.LBM* = 11b
- Move to Force mode by setting the following bits:
  - — *CTRL.FD* (CSR 0x0 bit 0) = 1
  - — *CTRL.SLU* (CSR 0x0 bit 6) = 1
  - — *CTRL.RFCE* (CSR 0x0 bit 27) = 0
  - — *CTRL.TFCE* (CSR 0x0 bit 28) = 0
  - — *PCS_LCTL.FORCE_LINK* (CSR 0X4208 bit 5) = 1
  - — *PCS_LCTL.FSD* (CSR 0X4208 bit 4) = 1
  - — *PCS_LCTL.FDV* (CSR 0X4208 bit 3) = 1
  - — *PCS_LCTL.FLV* (CSR 0X4208 bit 0) = 1
  - — *PCS_LCTL.AN_ENABLE* (CSR 0X4208 bit 16) = 0
  - —

3.5.6.5 External PHY Loopback

In External PHY loopback the SerDes block is not functional and data is sent through the MDI interface and looped back using an external loopback plug. This means all the design that is functional in copper mode, is involved in the loopback. If connected at 10/100 Mbps the loopback will work without any special setup. For 1000 Mbps operation, the following flow should be used:

3.5.6.5.1 Setting the 82580 Internal PHY to External Loopback Mode

The following procedure should be used to put the 82580 internal PHY into external loopback mode:

- Set Link mode to PHY: *CTRL_EXT.LINK_MODE* = 00b
- In PHY control register (Address 0 in the PHY):
  - Set Duplex mode (bit 8)
  - Clear Loopback bit (Bit 14)
  - Set Auto Neg enable bit (Bit 12)
- Restart auto-negotiation (Set bit 9)
- Reset the PHY (Set bit 15).
- Wait for auto-negotiation to complete, then transmit and receive normally.

3.5.7 Integrated Copper PHY Functionality

The register set used to control the PHY functionality (PHYREG) is described in Section 8.24. the registers can be programmed using the MDIC register (See Section 8.2.4).

3.5.7.1 Determining Link State

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

- Auto-negotiation
- Parallel detection
- Forced operation

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

Figure 3-5 provides an overview of link establishment. First the PHY checks if auto-negotiation is enabled. By default, the PHY supports auto-negotiation, see PHY Register 0, bit 12. If not, the PHY forces operation as directed. If auto-negotiation is enabled, the PHY begins transmitting Fast Link Pulses (FLPs) and receiving FLPs from its link partner. If FLPs are received by the PHY, auto-negotiation proceeds. It also can receive 100BASE-TX MLT3 and 10BASE-T Normal Link Pulses (NLPs). If either MLT3 or NLPs are received, it aborts FLP transmission and immediately brings up the corresponding half-duplex link.

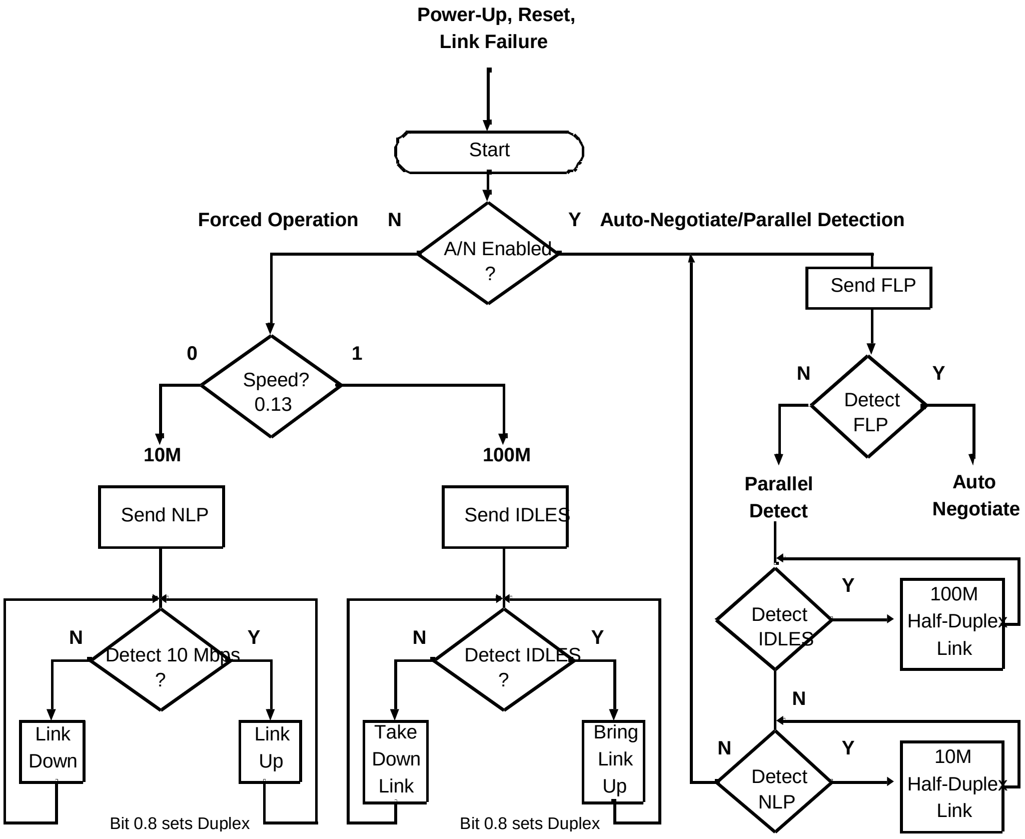


Figure 3-5. Overview of Link Establishment

3.5.7.1.1 False Link

The PHY does not falsely establish link with a partner operating at a different speed. For example, the PHY does not establish a 1 Gb/s or 10 Mb/s link with a 100 MB/s link partner.

When the PHY is first powered on, reset, or encounters a link down state, it must determine the line speed and operating conditions to use for the network link.

The PHY first checks the MDIO registers (initialized via the hardware control interface or written by software) for operating instructions. Using these mechanisms, designers can command the PHY to do one of the following:

- Force twisted-pair link operation to:
  - — 1000T, full duplex
  - — 1000T, half duplex
  - — 100TX, full duplex

- — 100TX, half duplex
- — 10BASE-T, full duplex
- — 10BASE-T, half duplex
- Allow auto-negotiation/parallel-detection.

In the first six cases (forced operation), the PHY immediately begins operating the network interface as commanded. In the last case, the PHY begins the auto-negotiation/parallel-detection process.

3.5.7.1.2 Forced Operation

Forced operation can be used to establish 10 Mb/s and 100 Mb/s links, and 1000 Mb/s links for test purposes. In this method, auto-negotiation is disabled completely and the link state of the PHY is determined by MII Register 0.

Note: When speed is forced, the auto cross-over feature is not functional.

In forced operation, the designer sets the link speed (10, 100, or 1000 MB/s) and duplex state (full or half). For Gigabit (1000 MB/s) links, designers must explicitly designate one side as the master and the other as the slave.

Note: The paradox (per the standard): If one side of the link is forced to full-duplex operation and the other side has auto-negotiation enabled, the auto-negotiating partner parallel-detects to a half-duplex link while the forced side operates as directed in full-duplex mode. The result is spurious, unexpected collisions on the side configured to auto-negotiate.

Table 3-38 lists link establishment procedures.

Table 3-38. Determining Duplex State Via Parallel Detection

Configuration	Result
Both sides set for auto-negotiate	Link is established via auto-negotiation.
Both sides set for forced operation	No problem as long as duplex settings match.
One side set for auto-negotiation and the other for forced, half-duplex	Link is established via parallel detect.
One side set for auto-negotiation and the other for forced full-duplex	Link is established; however, sides disagree, resulting in transmission problems (Forced side is full-duplex, auto-negotiation side is half-duplex.).

3.5.7.1.3 Auto Negotiation

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

3.5.7.1.4 Parallel Detection

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

With parallel detection, it is impossible to determine the true duplex state of the link partner and the IEEE standard requires the PHY to assume a half-duplex link. Parallel detection also does not allow exchange of flow-control ability (PAUSE and ASM_DIR) or the master/slave relationship required by 1000BASE-T. This is why parallel detection cannot be used to establish GbE links.

3.5.7.1.5 Auto Cross-Over

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

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

For 10/100/1000 Mb/s links and during auto-negotiation, pair usage is determined by bits 9 and 10 in the *PHCTRL2* register (PHYREG18). The PHY activates an automatic cross-over detection function. If bit *PHCTRL2.Automatic MDI/MDI-X (*18.10) = 1b, the PHY automatically detects which application is being used and configures itself accordingly.

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

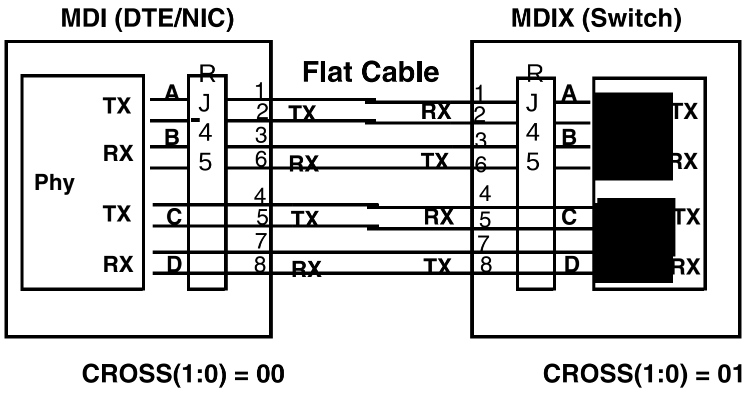


Figure 3-6. Cross-Over Function

3.5.7.1.6 10/100 MB/s Mismatch Resolution

It is a common occurrence that a link partner (such as a switch) is configured for forced full-duplex (FDX) 10/100 Mb/s operation. The normal auto-negotiation sequence would result in the other end settling for half-duplex (HDX) 10/100 Mb/s operation. The mechanism described in this section resolves the mismatch automatically and transitions the 82580 into FDX mode, enabling it to operate with a partner configured for FDX operation.

The 82580 enables the system software device driver to detect the mismatch event previously described and sets its duplex mode to the appropriate value without a need to go through another auto-negotiation sequence or breaking link. Once software detects a possible mismatch, it might instruct the 82580 to change its duplex setting to either HDX or FDX mode. Software sets the *Duplex_manual_set* bit to indicate that duplex setting should be changed to the value indicated by the *Duplex Mode* bit in PHY Register 0. Any change in the value of the *Duplex Mode* bit in PHY Register 0 while the *Duplex_manual_set* bit is set to 1b would also cause a change in the device duplex setting.

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

3.5.7.1.7 Link Criteria

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

3.5.7.1.7.1 1000BASE-T

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

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

3.5.7.1.7.2 100BASE-TX

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

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

3.5.7.1.7.3 10BASE-T

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

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

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

3.5.7.2 Link Enhancements

The PHY offers two enhanced link functions, each of which are discussed in the sections that follow:

- SmartSpeed
- Flow control

3.5.7.2.1 SmartSpeed

SmartSpeed is an enhancement to auto-negotiation that enables the PHY to react intelligently to network conditions that prohibit establishment of a 1000BASE-T link, such as cable problems. Such problems might allow auto-negotiation to complete, but then inhibit completion of the training phase. Normally, if a 1000BASE-T link fails, the PHY returns to the auto-negotiation state with the same speed settings indefinitely. With SmartSpeed enabled by programming the *PHCNFG.Automatic Speed Downshift Mode* field (See Section 8.24.2.17), after a configurable number of failed attempts, as configured in the *PHCTRL1* register (bits 12:10 - See Section 8.24.2.18) the PHY automatically downgrades the highest ability it advertises to the next lower speed: from 1000 to 100 to 10 Mb/s. Once a link is established, and if it is later broken, the PHY automatically upgrades the capabilities advertised to the original setting. This enables the PHY to automatically recover once the cable plant is repaired.

3.5.7.2.1.1 Using SmartSpeed

SmartSpeed is enabled by programming the *PHCNFG.Automatic Speed Downshift Mode* field (See Section 8.24.2.17). When SmartSpeed downgrades the PHY advertised capabilities, it sets bit *PHINT.Automatic Speed Downshift* (*PHYREG.25.1* - See Section 8.24.2.20). When link is established, its speed is indicated in the *PHSTAT.Speed Status* field (PHYREG.26.9:8 - See Section 8.24.2.21). SmartSpeed automatically resets the highest-level auto-negotiation abilities advertised, if link is established and then lost.

The number of failed attempts allowed is configured in the *PHCTRL1* register (bits 12:10 - See Section 8.24.2.18).

Note: SmartSpeed and M/S fault - When SmartSpeed is enabled, the M/S (Master-Slave) number of Attempts Before Downshift is programmed to be less than 7, resolution is not given seven attempts to try to resolve M/S status (see IEEE 802.3 clause 40.5.2).

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

3.5.7.3 Flow Control

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

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

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

Table 3-39 lists the intended operation for the various settings of *ASM_DIR* and *PAUSE*. This information is provided for reference only; it is the responsibility of the MAC to implement the correct function. The PHY merely enables the two MACs to communicate their abilities to each other.

Table 3-39. Pause And Asymmetric Pause Settings

ASM_DIR settings Local (PHYREG.4.10) and Remote (PHYREG.5.10)	Pause Setting - Local (PHYREG.4.9)	Pause Setting - Remote (PHYREG.5.9)	Result
Both ASM_DIR = 1b	1	1	Symmetric - Either side can flow control the other
	1	0	Asymmetric - Remote can flow control local only
	0	1	Asymmetric - Local can flow control remote
	0	0	No flow control
Either or both ASM_DIR = 0b	1	1	Symmetric - Either side can flow control the other
	Either or both = 0		No flow control

3.5.7.4 Management Data Interface

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

The PHY supports the core 16-bit MDIO registers. Registers 0-10 and 15 are required and their functions are specified by the IEEE 802.3 specification. Additional registers are included for expanded functionality. Specific bits in the registers are referenced using an PHY REG X.Y notation, where X is the register number (0-31) and Y is the bit number (0-15). See the software interface chapter (See Section 8.24).

3.5.7.5 Internal PHY Low Power Operation and Power Management

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

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

3.5.7.5.1 Power Down via the PHY Register

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

3.5.7.5.2 Power Management State

The internal PHY is aware of the power management state. If the PHY is not in a power down state, then PHY behavior regarding several features are different depending on the power state, see Section 3.5.7.5.4.

3.5.7.5.3 Disable High Speed Power Saving Options

The 82580 supports disabling 1000 Mb/s or both 1000 Mb/s and 100 Mb/s advertisement by the internal PHY regardless of the values programmed in the PHY ANA Register (address - 4d) and the PHY *GCON* Register (address - 9d).

This is for cases where the system doesn't support working in 1000 Mb/s or 100 Mb/s due to power limitations.

This option is enabled in the following *PHPM* register bits:

- *PHPM.Disable 1000 in non-D0a* - disable 1000 Mb/s when in non-D0a states only.
- *PHPM.Disable 100 in non-D0a* - disable 1000 Mb/s and 100 Mb/s when in non-D0a states only.
- *PHPM.Disable 1000* - disable 1000 Mb/s always.

Note: When Value of *PHPM.Disable 1000* bit is changed, PHY initiates Auto-negotiation without direct driver command.

3.5.7.5.4 Low Power Link Up - Link Speed Control

Normal Internal PHY speed negotiation drives to establish a link at the highest possible speed. The 82580 supports an additional mode of operation, where the PHY drives to establish a link at a low speed. The link-up process enables a link to come up at the lowest 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.

Table 3-40 lists link speed as function of power management state, link speed control, and GbE speed enabling:

Table 3-40. Link Speed vs. Power State

Power Management State	Low Power Link Up (*PHPM.1, PHPM.2*)	GbE Disable Bits		100M Disable Bit	PHY Speed Negotiation
		Disable 1000 (*PHPM.6*)	Disable 1000 in non-D0a (*PHPM.3*)	Disable 100 in non-D0a (*PHPM.9*)	
D0a	0, Xb	0b	X	X	PHY negotiates to highest speed advertised (normal operation).
		1b	X	X	PHY negotiates to highest speed advertised (normal operation), excluding 1000 Mb/s.
	1, Xb	0b	X	X	PHY goes through Low Power Link Up (LPLU) procedure, starting with advertised values.
		1b	X	X	PHY goes through LPLU procedure, starting with advertised values. Does not advertise 1000 Mb/s.

Table 3-40. Link Speed vs. Power State

Power Management State	Low Power Link Up (*PHPM.1*, *PHPM.2*)	GbE Disable Bits		100M Disable Bit	PHY Speed Negotiation
		Disable 1000 (*PHPM.6*)	Disable 1000 in non-D0a (*PHPM.3*)	Disable 100 in non-D0a (*PHPM.9*)	
Non-D0a	X, 0b	0b	0b	0b	PHY negotiates to highest speed advertised.
		0b	1b	0b	PHY negotiates to highest speed advertised, excluding 1000 Mb/s.
		1b	X	0b	
		X	X	1b	PHY negotiates and advertises only 10 Mb/s
	X, 1b	0b	0b	0b	PHY goes through LPLU procedure, starting at 10 Mb/s.
		0b	1b	0b	PHY goes through LPLU procedure, starting at 10 Mb/s. Does not advertise 1000 Mb/s.
		X	X	1b	PHY negotiates and advertises only 10 Mb/s

The Internal PHY initiates auto-negotiation without a direct driver command in the following cases:

- When the *PHPM.Disable 1000 in non-D0a* bit is set and 1000 Mb/s is disabled on D3 or Dr entry (but not in D0a), the PHY auto-negotiates on entry.
- When the *PHPM.Disable 100 in non-D0a* is set and 1000 Mb/s and 100 Mb/s are disabled on D3 or Dr entry (but not in D0a), the PHY auto-negotiates on entry.
- When *PHPM.LPLU* changes state with a change in a power management state. For example, on transition from D0a without *PHPM.LPLU* to D3 with *PHPM.LPLU*. Or, on transition from D3 with *PHPM.LPLU* to D0 without *LPLU*.
- On a transition from D0a state to a non-D0a state, or from a non-D0a state to D0a state, and *PHPM.LPLU* is set.

Notes:

- The Low-Power Link-Up (LPLU) feature previously described should be disabled (in both D0a state and non-D0a states) when the intended advertisement is anything other than 10 Mb/s only, 10/100 Mb/s only, or 10/100/1000 Mb/s. This is to avoid reaching (through the LPLU procedure) a link speed that is not advertised by the user.
- When the LAN PCIe Function is disabled via the *LAN_PCI_DIS* bit in the *Software Defined Pins Control* EEPROM word, the relevant Function is in a Non-D0a state. As a result Management might operate with reduced link speed if the *LPLU*, *Disable 1000 in Non-D0a* or *Disable 100 in Non-D0a* EEPROM bits are set.

3.5.7.5.4.1 D0a State

A power-managed link speed control lowers link speed (and power) when highest link performance is not required. When enabled (D0 Low Power Link Up mode), any link negotiation tries to establish a low-link speed, starting with an initial advertisement defined by software.

The D0LPLU configuration bit enables *D0 Low Power Link Up*. Before enabling this feature, software must advertise to one of the following speed combinations: 10 Mb/s only, 10/100 Mb/s only, or 10/100/1000 Mb/s.

When speed negotiation starts, the PHY tries to negotiate at a speed based on the currently advertised values. If link establishment fails, the PHY tries to negotiate with different speeds; it enables all speeds up to the lowest speed supported by the partner. For example, PHY advertises 10 Mb/s only, and the partner supports 1000 Mb/s only. After the first try fails, the PHY enables 10/100/1000 Mb/s and tries again. The PHY continues to try and establish a link until it succeeds or until it is instructed otherwise. In the second step (adjusting to partner speed), the PHY also enables parallel detect, if needed. Automatic MDI/MDI-X resolution is done during the first auto-negotiation stage.

3.5.7.5.4.2 Non-D0a State

The PHY might negotiate to a low speed while in non-D0a states (Dr, D0u, D3). This applies only when the link is required by one of the following: SMBus manageability, APM Wake, or PME. Otherwise, the PHY is disabled during the non-D0 state.

The *Low Power on Link-Up* (Register *PHPM.LPLU*, is also loaded from EEPROM) bit enables reduction in link speed:

- At power-up entry to Dr state, the PHY advertises supports for 10 Mb/s only and goes through the link up process.
- At any entry to a non-D0a state (Dr, D0u, D3), the PHY advertises support for 10 Mb/s only and goes through the link up process.
- While in a non-D0 state, if auto-negotiation is required, the PHY advertises support for 10 Mb/s only and goes through the link up process.

Link negotiation begins with the PHY trying to negotiate at 10 Mb/s speed only regardless of user auto-negotiation advertisement. If link establishment fails, the PHY tries to negotiate at additional speeds; it enables all speeds up to the lowest speed supported by the partner. For example, the PHY advertises 10 Mb/s only and the partner supports 1000 Mb/s only. After the first try fails, PHY enables 10/100/1000 Mb/s and tries again. The PHY continues to try and establish a link until it succeeds or until it is instructed otherwise. In the second step (adjusting to partner speed), the PHY also enables parallel detect, if needed. Automatic MDI/MDI-X resolution is done during the first auto-negotiation stage.

3.5.7.5.5 Internal PHY Smart Power-Down (SPD)

Smart power-down is a link-disconnect capability applicable to all power management states. Smart Power-down combines a power saving mechanism with the fact that the link might disappear and resume.

Smart power-down is enabled by *PHPM.SPD_EN* or by *SPD Enable* bit in the EEPROM if the following conditions are met:

1. Auto-negotiation is enabled.
2. PHY detects link loss.

While in the smart power-down state, the PHY powers down circuits and clocks that are not required for detection of link activity. The PHY is still be able to detect link pulses (including parallel detect) and wake-up to engage in link negotiation. The PHY does not send link pulses (NLP) while in SPD state; however, register accesses are still possible.

When the Internal PHY is in smart power-down mode and detects link activity, it re-negotiates link speed based on the power state and the *Low Power Link Up* bits as defined by the *PHPM.D0LPLU* and *PHPM.LPLU* bits.

Note: The PHY does not enter the SPD state unless auto-negotiation is enabled.

While in the SPD state, the PHY powers down all circuits not required for detection of link activity. The PHY must still be able to detect link pulses (including parallel detect) and wake up to engage in link negotiation. The PHY does not send link pulses (NLP) while in SPD state.

Notes: While in the link-disconnect state, the PHY must allow software access to its registers.

The link-disconnect state applies to all power management states (Dr, D0u, D0a, D3).

The link might change status, that is go up or go down, while in any of these states.

3.5.7.5.5.1 Internal PHY **Back-to-Back Smart Power-Down**

While in link disconnect, the 82580 monitors the link for link pulses to identify when a link is re-connected. The 82580 also periodically transmits pulses (every 100 ms) to resolve the case of two the 82580 devices (or devices with the 82580-like behavior) connected to each other across the link. Otherwise, two such devices might be locked in Smart power-down mode, not capable of identifying that a link was re-connected.

Back-to-back smart power-down is enabled by the *SPD_B2B_EN* bit in the *PHPM* register. The default value is enabled. The *Enable* bit applies to smart power-down mode.

Note: This bit should not be altered by software once the 82580 was set in smart power-down mode. If software requires changing the back-to-back status, it first needs to transition the PHY out of smart power-down mode and only then change the back-to-back bit to the required state.

3.5.7.5.6 Internal PHY Link Energy Detect

The 82580 asserts the *Link Energy Detect* bit (*PHPM.Link Energy Detect*) each time energy is not detected on the link. This bit provides an indication of a cable becoming plugged or unplugged.

This bit is valid only if the *PHPM.Go Link disconnect* bit is set.

In order to correctly deduce that there is no energy, the bit must read 0b for three consecutive reads each second.

3.5.7.5.7 Internal PHY Power-Down State

The 82580 ports 0 to 3 enter a power-down state when none of the port's clients are enabled and therefore the internal PHY has no need to maintain a link. This can happen in one of the following cases, if the Internal PHY power-down functionality is enabled through the EEPROM PHY *Power Down Enable* bit.

1. D3/Dr state: Each Internal PHY enters a low-power state if the following conditions are met:
   a. The LAN function associated with this PHY is in a non-D0 state
   b. APM WOL is inactive
   c. Manageability doesn't use this port.
   d. ACPI PME is disabled for this port.
   e. The PHY *Power Down Enable* EEPROM bit is set (*Initialization Control Word 2*, word 0xF, bit 6).
2. SerDes mode: Each Internal PHY is disabled when its LAN function is configured to SerDes mode.
3. LAN disable: Each Internal PHY can 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 might 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. The 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 Internal PHY enters power down.

Note: Setting the EEPROM *LAN_PCI_DIS* bit does not move the internal PHY into power down. However if the *LPLU*, *Disable 1000 in Non-D0a* or the *Disable 100 in Non-D0a* EEPROM bits are set Management may operate with reduced link speed since the Function is in a Non-D0a (uninitialized) state.

3.5.7.6 Advanced Diagnostics

The 82580 Integrated PHY incorporates hardware support for advanced diagnostics.

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

The current diagnostics supported are:

3.5.7.6.1 TDR - Time Domain Reflectometry

By sending a pulse onto the twisted pair and observing the retuned signal, the following can be deduced:

1. Is there a short?
2. Is there an open?
3. Is there an impedance mismatch?
4. What is the length to any of these faults?

3.5.7.6.2 Channel Frequency Response

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

3.5.7.7 1000 Mb/s Operation

3.5.7.7.1 Introduction

Figure 3-7 shows an overview of 1000BASE-T functions, followed by discussion and review of the internal functional blocks.

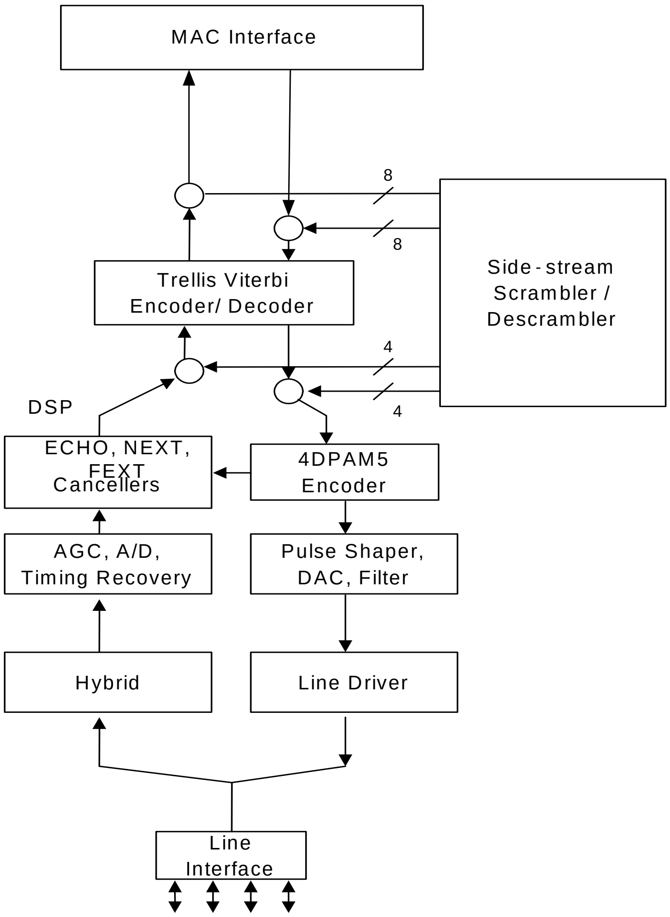

MAC Interface
8
8
Side-stream Scrambler / Descrambler
Trellis Viterbi Encoder/ Decoder
4
4
DSP
ECHO, NEXT, FEXT Cancellers
4DPAM5 Encoder
AGC, A/D, Timing Recovery
Pulse Shaper, DAC, Filter
Hybrid
Line Driver
Line Interface

Figure 3-7. 1000BASE-T Functions Overview

3.5.7.7.2 Transmit Functions

This section describes functions used when the Media Access Controller (MAC) transmits data through the PHY and out onto the twisted-pair connection (see Figure 3-7).

3.5.7.7.2.1 Scrambler

The scrambler randomizes the transmitted data. The purpose of scrambling is twofold:

1. Scrambling eliminates repeating data patterns (also known as spectral lines) from the 4DPAM5 waveform in order to reduce EMI.
2. Each channel (A, B, C, D) has a unique signature that the receiver uses for identification.

The scrambler is driven by a 33-bit Linear Feedback Shift Register (LFSR), which is randomly loaded at power up. The LFSR function used by the master differs from that used by the slave, giving each direction its own unique signature. The LFSR, in turn, generates twelve mutually uncorrelated outputs. Eight of these are used to randomize the inputs to the 4DPAM5 and Trellis encoders. The remaining four outputs randomize the sign of the 4DPAM5 outputs.

3.5.7.7.2.2 Transmit FIFO

The transmit FIFO re-synchronizes data transmitted by the MAC to the transmit reference used by the PHY. The FIFO is large enough to support a frequency differential of up to +/- 1000 ppm over a packet size of 10 KB (jumbo frame).

3.5.7.7.2.3 Transmit Phase-Locked Loop PLL

This function generates the 125 MHz timing reference used by the PHY to transmit 4DPAM5 symbols. When the PHY is the master side of the link, the XI input is the reference for the transmit PLL. When the PHY is the slave side of the link, the recovered receive clock is the reference for the transmit PLL.

3.5.7.7.2.4 Trellis Encoder

The Trellis encoder uses the two high-order bits of data and its previous output to generate a ninth bit, which determines if the next 4DPAM5 pattern should be even or odd.

For data, this function is:

Trellisn = Data7n-1 XOR Data6n-2 XOR Trellisn-3

This provides forward error correction and enhances the Signal-To-Noise (SNR) ratio by a factor of 6 dB.

3.5.7.7.2.5 4DPAM5 Encoder

The 4DPAM5 encoder translates 8-byte codes transmitted by the MAC into 4DPAM5 symbols. The encoder operates at 125 MHz, which is both the frequency of the MAC interface and the baud rate used by 1000BASE-T.

Each 8-byte code represents one of 28 or 256 data patterns. Each 4DPAM5 symbol consists of one of five signal levels (-2,-1,0,1,2) on each of the four twisted pair (A,B,C,D) representing 54 or 625 possible patterns per baud period. Of these, 113 patterns are reserved for control codes, leaving 512 patterns for data. These data patterns are divided into two groups of 256 even and 256 odd data patterns. Thus, each 8-byte octet has two possible 4DPAM5 representations: one even and one odd pattern.

3.5.7.7.2.6 Spectral Shaper

This function causes the 4DPAM5 waveform to have a spectral signature that is very close to that of the MLT3 waveform used by 100BASE-TX. This enables 1000BASE-T to take advantage of infrastructure (cables, magnetics) designed for 100BASE-TX.

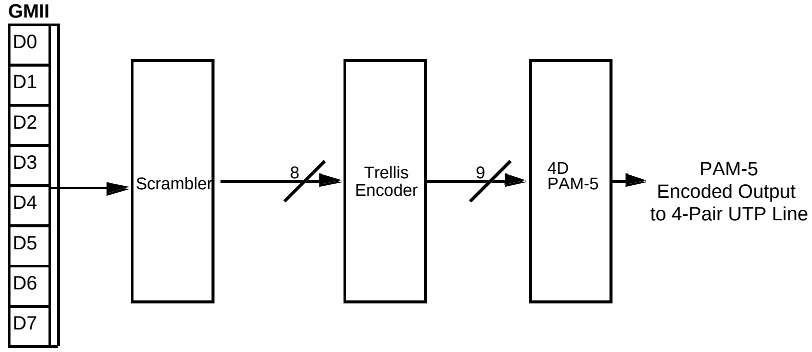
The shaper works by transmitting 75% of a 4DPAM5 code in the current baud period, and adding the remaining 25% into the next baud period.

3.5.7.7.2.7 Low-Pass Filter

To aid with EMI, this filter attenuates signal components more than 180 MHz. In 1000BASE-T, the fundamental symbol rate is 125 MHz.

3.5.7.7.2.8 Line Driver

The line driver drives the 4DPAM5 waveforms onto the four twisted-pair channels (A, B, C, D), adding them onto the waveforms that are simultaneously being received from the link partner.



Scrambler Polynomials:

$1 + x^{13} + x^{33}$ (Master PHY Mode)

$1 + x^{20} + x^{33}$ (Slave PHY Mode)

Figure 3-8. 1000BASE-T Transmit Flow And Line Coding Scheme

Figure 3-9. Transmit/Receive Flow

3.5.7.7.3 Receive Functions

This section describes function blocks that are used when the PHY receives data from the twisted pair interface and passes it back to the MAC (see Figure 3-9).

3.5.7.7.3.1 Hybrid

The hybrid subtracts the transmitted signal from the input signal, enabling the use of simple 100BASE-TX compatible magnetics.

3.5.7.7.3.2 Automatic Gain Control (AGC)

AGC normalizes the amplitude of the received signal, adjusting for the attenuation produced by the cable.

3.5.7.7.3.3 Timing Recovery

This function re-generates a receive clock from the incoming data stream which is used to sample the data. On the slave side of the link, this clock is also used to drive the transmitter.

3.5.7.7.3.4 Analog-to-Digital Converter (ADC)

The ADC function converts the incoming data stream from an analog waveform to digitized samples for processing by the DSP core.

3.5.7.7.3.5 Digital Signal Processor (DSP)

DSP provides per-channel adaptive filtering, which eliminates various signal impairments including:

- Inter-symbol interference (equalization)
- Echo caused by impedance mismatch of the cable
- Near-end crosstalk (NEXT) between adjacent channels (A, B, C, D)
- Far-end crosstalk (FEXT)
- Propagation delay variations between channels of up to 120 ns
- Extraneous tones that have been coupled into the receive path

The adaptive filter coefficients are initially set during the training phase. They are continuously adjusted (adaptive equalization) during operation through the decision-feedback loop.

3.5.7.7.3.6 Descrambler

The descrambler identifies each channel by its characteristic signature, removing the signature and re-routing the channel internally. In this way, the receiver can correct for channel swaps and polarity reversals. The descrambler uses the same base 33-bit LFSR used by the transmitter on the other side of the link.

The descrambler automatically loads the seed value from the incoming stream of scrambled idle symbols. The descrambler requires approximately 15 μs to lock, normally accomplished during the training phase.

3.5.7.7.3.7 Viterbi Decoder/Decision Feedback Equalizer (DFE)

The Viterbi decoder generates clean 4DPAM5 symbols from the output of the DSP. The decoder includes a Trellis encoder identical to the one used by the transmitter. The Viterbi decoder simultaneously looks at the received data over several baud periods. For each baud period, it predicts whether the symbol received should be even or odd, and compares that to the actual symbol received. The 4DPAM5 code is organized in such a way that a single level error on any channel changes an even code to an odd one and vice versa. In this way, the Viterbi decoder can detect single-level coding errors, effectively improving the signal-to-noise (SNR) ratio by a factor of 6 dB. When an error occurs, this information is quickly fed back into the equalizer to prevent future errors.

3.5.7.7.3.8 4DPAM5 Decoder

The 4DPAM5 decoder generates 8-byte data from the output of the Viterbi decoder.

3.5.7.7.3.9 100 Mb/s Operation

The MAC passes data to the PHY over the MII. The PHY encodes and scrambles the data, then transmits it using MLT-3 for 100TX over copper. The PHY de-scrambles and decodes MLT-3 data received from the network. When the MAC is not actively transmitting data, the PHY sends out idle symbols on the line.

3.5.7.7.3.10 10 Mb/s Operation

The PHY operates as a standard 10 Mb/s transceiver. Data transmitted by the MAC as 4-bit nibbles is serialized, Manchester-encoded, and transmitted on the MDI[0]+/- outputs. Received data is decoded, de-serialized into 4-bit nibbles and passed to the MAC across the internal MII. The PHY supports all the standard 10 Mb/s functions.

3.5.7.7.3.11 Link Test

In 10 Mb/s mode, the PHY always transmits link pulses. If link test function is enabled, it monitors the connection for link pulses. Once it detects two to seven link pulses, data transmission are enabled and remain enabled as long as the link pulses or data reception continues. If the link pulses stop, the data transmission is disabled.

If the link test function is disabled, the PHY might transmit packets regardless of detected link pulses. Setting the Port Configuration register bit (PHYREG.16.14) can disable the link test function.

3.5.7.7.3.12 10Base-T Link Failure Criteria and Override

Link failure occurs if link test is enabled and link pulses stop being received. If this condition occurs, the PHY returns to the auto-negotiation phase, if auto-negotiation is enabled. Setting the Port Configuration register bit (PHYREG.16.14) disables the link integrity test function, then the PHY transmits packets, regardless of link status.

3.5.7.7.3.13 Jabber

If the MAC begins a transmission that exceeds the jabber timer, the PHY disables the transmit and loopback functions and asserts collision indication to the MAC. The PHY automatically exits jabber mode after 250-750 ms. This function can be disabled by setting bit PHYREG.16.10 = 1b.

3.5.7.7.3.14 Polarity Correction

The PHY automatically detects and corrects for the condition where the receive signal (MDI_PLUS[0]/MDI_MINUS[0]) is inverted. Reversed polarity is detected if eight inverted link pulses or four inverted end-of-frame markers are received consecutively. If link pulses or data are not received for 96-130 ms, the polarity state is reset to a non-inverted state.

Automatic polarity correction can be disabled by setting bit PHYREG.27.5.

3.5.7.7.3.15 Dribble Bits

The PHY handles dribble bits for all of its modes. If between one and four dribble bits are received, the nibble is passed across the interface. The data passed across is padded with 1's if necessary. If between five and seven dribble bits are received, the second nibble is not sent onto the internal MII bus to the MAC. This ensures that dribble bits between 1-7 do not cause the MAC to discard the frame due to a CRC error.

3.5.7.7.3.16 PHY Address

The External PHY MDIO Address is defined in the *MDICNFG.PHYADD* field and is loaded at power-up from EEPROM. If the *MDICNFG.Destination* bit is cleared (Internal PHY), MDIO access is always to the internal PHY.

3.5.8 Media Auto Sense

The 82580 provides a significant amount of flexibility in pairing a LAN device with a particular type of media (such as copper or fiber-optic) as well as the specific transceiver/interface used to communicate with the media. Each MAC, representing a distinct LAN device, can be coupled with an internal copper PHY or SerDes/SGMII/1000BASE-KX interface independently. The link configuration specified for each LAN device can be specified in the *LINK_MODE* field of the Extended Device Control (*CTRL_EXT*) register and initialized from the EEPROM Initialization Control Word 3 associated with each LAN Port.

In some applications, software might need to be aware of the presence of a link on the media not currently active. In order to supply such an indication, any of the 82580 ports can set the *AUTOSENSE_EN* bit in the CONNSW register (address 0x0034) in order to enable sensing of the non active media activity.

Note: When in SerDes/SGMII/1000BASE-KX detect mode, software should define which indication is used to detect the energy change on the SerDes/SGMII/1000BASE-KX media. It can be either the external signal detect pin or the internal signal detect. This is done using the *CONNSW.ENRGSRC* bit. The signal detect pin is normally used when connecting in SerDes mode to optical media where the receive LED provide such an indication.

Software can then enable the *OMED* interrupt in *ICR* in order to get an indication on any detection of energy in the non active media.

Note: The auto-sense capability can be used in either port independent of the usage of the other port.

The following sections describes the procedures that should be followed in order to enable the auto-sense mode

3.5.8.1 Auto sense setup

3.5.8.1.1 SerDes/SGMII/1000BASE-KX Detect Mode (PHY is Active)

1. Set *CONNSW.ENRGSRC* to determine the sources for the signal detect indication (1b = external SIG_DET, 0b = internal SerDes electrical idle). The default of this bit is set by EEPROM.
2. Set *CONNSW.AUTOSENSE_EN*.
3. When link is detected on the SerDes/SGMII/1000BASE-KX media, the 82580 sets the interrupt bit *OMED* in ICR and if enabled, issues an interrupt. The *CONNSW.AUTOSENSE_EN* bit is cleared .

3.5.8.1.2 PHY Detect Mode (SerDes/SGMII/1000BASE-KX is active)

1. Set *CONNSW.AUTOSENSE_CONF* = 1b.

2. Reset the PHY as described in Section 4.3.2.5.
3. Enable PHY to move to low power mode when cable is disconnected by setting the *PHPM.SPD_EN* bit.
4. Set *CONNSW.AUTOSENSE_EN* = 1b and then clear *CONNSW.AUTOSENSE_CONF*.
5. When signal is detected on the PHY media, the 82580 sets the *ICR.OMED* interrupt bit and issues an interrupt if enabled.
6. The 82580 puts the PHY in power down mode.

According to the result of the interrupt, software can then decide to switch to the other media.

Note: Assertion of *ICR.OMED* PHY is a one time event. To re-enable Auto-detect after cable is unplugged Software should clear the *CONNSW.AUTOSENSE_EN* bit and the procedure defined above should be executed again.

3.5.8.2 Switching between medias.

The 82580's link mode is controlled by the *CTRL_EXT.LINK_MODE* field. The default value for the *LINK_MODE* setting is directly mapped from the EEPROM's *initialization Control Word 3 Link Mode* field. Software can modify the *LINK_MODE* indication by writing the corresponding value into this register.

Notes: Before dynamically switching between medias, the software should ensure that the current mode of operation is not in the process of transmitting or receiving data. This is achieved by disabling the transmitter and receiver, waiting until the 82580 is in an idle state, and then beginning the process for changing the link mode.

The mode switch in this method is only valid until the next hardware reset of the 82580. After a hardware reset, the link mode is restored to the default setting by the EEPROM. To get a permanent change of the link mode, the default in the EEPROM should be changed.

The following procedures need to be followed to actually switch between the two modes.

3.5.8.2.1 Transition to SerDes/1000BASE-KX/SGMII Modes

1. Disable the receiver by clearing *RCTL.RXEN*.
2. Disable the transmitter by clearing *TCTL.EN*.
3. Verify the 82580 has stopped processing outstanding cycles and is idle.
4. Modify LINK mode to SerDes, 1000BASE-KX or SGMII by setting *CTRL_EXT.LINK_MODE* to 011b, 001b or 010b respectively.
5. Set up the link as described in Section 4.6.7.3,Section 4.6.7.4 or Section 4.6.7.5.
6. Set up Tx and Rx queues and enable Tx and Rx processes.

3.5.8.2.2 Transition to Internal PHY Mode

1. Disable the receiver by clearing *RCTL.RXEN*.
2. Disable the transmitter by clearing *TCTL.EN*.
3. Verify the 82580 has stopped processing outstanding cycles and is idle.
4. Modify LINK mode to PHY mode by setting *CTRL_EXT.LINK_MODE* to 000b.
5. Set link-up indication by setting *CTRL.SLU*.
6. Reset the PHY as described in Section 4.3.2.5.

7. Set up the link as described in Section 4.6.7.2.
8. Set up the Tx and Rx queues and enable the Tx and Rx processes.

§ §

4.0 Initialization

4.1 Power Up

4.1.1 Power-Up Sequence

Figure 4-1 shows the power-up sequence from power ramp up and to when the 82580 is ready to accept host commands.

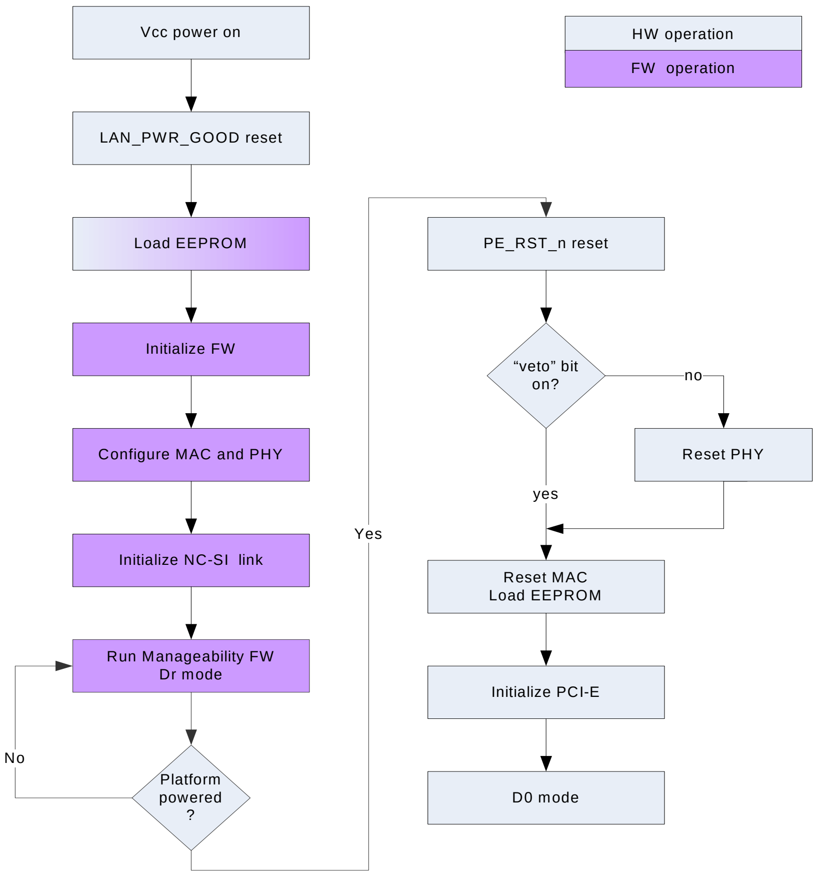


Figure 4-1. Power-Up - General Flow

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

4.1.2 Power-Up Timing Diagram

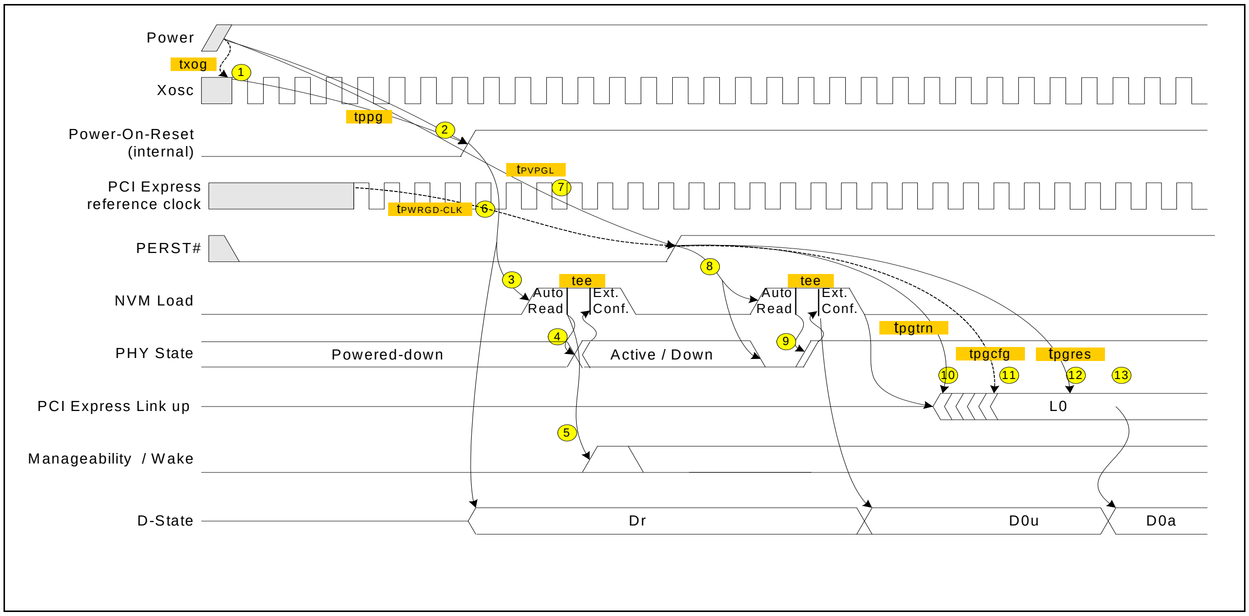


Figure 4-2. Power-Up Timing Diagram

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

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

4.2 Reset Operation

4.2.1 Reset Sources

The 82580 reset sources are described below:

4.2.1.1 LAN_PWR_GOOD

The 82580 has an internal mechanism for sensing the power pins. Once the power is up and stable, the 82580 creates an internal reset. This reset acts as a master reset of the entire chip. It is level sensitive, and while it is zero holds all of the registers in reset. LAN_PWR_GOOD is interpreted to be an indication that device power supplies are all stable. LAN_PWR_GOOD changes state during system power-up.

4.2.1.2 PE_RST_N

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

4.2.1.3 In-Band PCIe Reset

The 82580 generates an internal reset in response to a Physical layer message from the PCIe or when the PCIe link goes down (entry to Polling or Detect state). This reset is equivalent to PCI reset in previous (PCI) gigabit LAN controllers.

4.2.1.4 D3hot to D0 Transition

This is also known as ACPI Reset. The 82580 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 transitions from D3hot to D0.

Note: Software drivers should implement the handshake mechanism defined in Section 5.2.3.3 to verify that all pending PCIe completions are done, before moving the 82580 to D3.

4.2.1.5 Function Level Reset (FLR)

A FLR reset to a function is issued, by setting bit *15* in the *Device Control* configuration register (see Section 9.5.6.5), is equivalent to a D0 ⇨ D3 ⇨ D0 transition. The only difference is that this reset does not require driver intervention in order to stop the master transactions of this function. This reset is per function and resets only the function without effecting activity of other functions or Lan ports.

The EEPROM is partially reloaded after an FLR reset. The words read from EEPROM at FLR are the same as read following a full software reset. A list of these words can be found in Section 3.3.1.3.

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

Note: If software initiates a FLR when the *Transactions Pending* bit in the *Device Status* configuration register is set to 1b (see Section 9.5.6.6), then software must not initialize the function until allowing time for any associated completions to arrive. The *Transactions Pending* bit is cleared upon completion of the FLR.

4.3 Software Reset

4.3.1 Port Software Reset (RST)

Software can reset a port in the 82580 by setting the Port Software Reset (*CTRL.RST)* in the Device Control Register. The port software reset (*CTRL.RST*) is per function and resets only the function that received the software reset. Following the reset the PCI configuration space (configuration and mapping) of the device is unaffected. Prior to issuing software reset the driver needs to operate the master disable algorithm as defined in Section 5.2.3.3.

The *CTRL.RST* bit is provided primarily to recover from an indeterminate or suspected Port hung hardware state. Most registers (receive, transmit, interrupt, statistics, etc.) and state machines in the port are set to their power-on reset values, approximating the state following a power-on or PCIe reset. However, PCIe configuration registers and logic common to all ports is not reset, leaving the device mapped into system memory space and accessible by a driver.

Note: To ensure that software reset has fully completed and that the 82580 responds correctly to subsequent accesses, the driver should wait at least 3 milliseconds after setting *CTRL.RST* before attempting to check if the bit was cleared or before attempting to access any other register.

When asserting the *CTRL.RST* software reset bit, only some EEPROM bits related to the specific function are re-read (See Section 3.3.1.3). Bits re-read from EEPROM are reset to default values.

Fields controlled by the LED, SDP and Init3 words of the EEPROM are not reset and not re-read after a software reset. For the list of words read from EEPROM at full software reset, see Section 3.3.1.3.

4.3.2 Device Software Reset (DEV_RST)

Software can reset all the 82580 ports by setting the Device Reset bit (*CTRL.DEV_RST*) in the Device Control Register. The Device Reset (*CTRL.DEV_RST)* resets all functions and common logic. PCI configuration space (configuration and mapping) of the device is unaffected.

Device Reset (*CTRL.DEV_RST*) can be used to globally reset the entire component, if the *DEV_RST_EN* bit in *Initialization Control 4* EEPROM word is set. This bit is provided as a last-ditch software mechanism to recover from an indeterminate or suspected hardware hung state that could not be resolved by setting the *CTRL.RST* bit. When setting *CTRL.DEV_RST*, most registers (receive, transmit, interrupt, statistics, etc.) and state machines on ports are set to power-on reset values, approximating the state following a power-on or PCI reset. However, PCIe configuration registers are not reset, leaving the device mapped into system memory space and accessible by the drivers.

When *CTRL.DEV_RST* is asserted by software on a LAN port, all LAN ports (including LAN ports that didn't initiate the reset) are placed in a reset state. To notify software device drivers on all ports that *CTRL.DEV_RST* has been asserted, an interrupt is generated and the *ICR.DRSTA* bit is set on all ports that didn't initiate the Device reset. In addition the *STATUS.DEV_RST_SET* bit is set on all ports to indicate that Device reset was asserted.

Prior to issuing Device reset the driver needs to:

1. Get ownership of the Device reset functionality by sending message via the mailbox mechanism described in section Section 4.7.3 and receiving acknowledge message from other drivers.

2. Initiate the master disable algorithm as defined in Section 5.2.3.3.

Note: To ensure that Device reset has fully completed and that the 82580 responds correctly to subsequent accesses, wait at least 3 milliseconds after setting *CTRL.DEV_RST* before attempting to check if the bit was cleared or before attempting to access any other register.

Following Device Reset assertion or reception of Device Reset interrupt (*ICR.DRSTA*) software should initiate the following steps to re-initialize the port:

1. Wait for the *GCR.DEV_RST in progress* bit to be cleared.
2. Read *STATUS.DEV_RST_SET* bit and clear bit by write 1.
3. Re-initialize the port.
4. Check *STATUS.DEV_RST_SET* bit and verify that bit is still 0. If bit is set, return to 1. and re-start initialization process.
5. Driver that initiated the Device reset should release ownership of Device Reset and Mailbox using the flow described in Section 4.7.3.

When setting the Device reset bit (*CTRL.DEV_RST*), EEPROM bits related to all ports are re-read (See Section 3.3.1.3). Bits re-read from EEPROM are reset to default values.

Fields controlled by the LED, SDP and Init3 words of the EEPROM are not reset and not re-read after a software reset. For the list of words read from EEPROM at full software reset, see Section 3.3.1.3.

4.3.2.1 BME (Bus Master Enable)

Disabling Bus Master activity of a function by clearing the Configuration *Command register.BME* bit to 0, resets all DMA activities and MSI/MSIx operations related to the port. The Master disable is per function and resets only the DMA activities related to this function without effecting activity of other functions or LAN ports. Configuration accesses and target accesses to the function are still enabled and BMC can still transmit and receive packets on the port.

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

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

4.3.2.2 Force TCO

This reset is generated when manageability logic is enabled and BMC detects that the 82580 does not receive or transmit data correctly. Force TCO reset is enabled if the *Reset on Force TCO* bit in the *Management Control* EEPROM word is set 1.

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

4.3.2.3 Firmware Reset

This reset is activated by writing a 1 to the FWR bit in the HOST Interface Control Register (HICR) in CSR address 0x8F00.

4.3.2.4 EEPROM Reset

Writing a 1 to the EEPROM Reset bit of the Extended Device Control Register (*CTRL_EXT.EE_RST*) causes the 82580 to re-read the per-function configuration from the EEPROM, setting the appropriate bits in the registers loaded by the EEPROM.

4.3.2.5 PHY Reset

Software can write a 1 to the PHY Reset bit of the Device Control Register (*CTRL.PHY_RST*) to reset the internal PHY. The PHY is internally configured after a PHY reset.

Note: The internal PHY should not be reset using PHYREG 0 bit 15 (*PCTRL.Reset*), since in this case the internal PHY configuration process is bypassed and there is no guarantee the PHY will operate correctly.

As the PHY may be accessed by the internal firmware and the driver software, the driver software should coordinate any PHY reset with the firmware using the following procedure:

1. Check that *MANC.BLK_Phy_Rst_On_IDE* (offset 0x5820 bit 18) is cleared. If it is set, the BMC requires a stable link and thus the PHY should not be reset at this stage. The driver may skip the PHY reset if not mandatory or wait for *MANC.BLK_Phy_Rst_On_IDE* to clear. See Section 4.3.4 for more details.
2. Take ownership of the relevant PHY (on port 0,1,2 or 3) using the following flow:
    a. Get ownership of the software/software semaphore *SWSM.SMBI* bit (offset 0x5B50 bit 0).
        - Read the *SWSM* register.
        - If *SWSM.SMBI* is read as zero, the semaphore was taken.
        - Otherwise, go back to step a.
        - This step assure that other software will not access the shared resources register (*SW_FW_SYNC*).

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

    c. Software reads the Software-Firmware Synchronization Register (*SW_FW_SYNC)* and checks both bits in the pair of bits that control the PHY it wishes to own.
        - If both bits are cleared (both firmware and other software does not own the PHY), software sets the software bit in the pair of bits that control the resource it wishes to own.
        - If one of the bits is set (firmware or other software owns the PHY), software tries again later.

    d. Release ownership of the software/firmware semaphore by clearing the *SWSM.SWESMBI* bit.
3. Drive PHY reset bit in *CTRL* bit 31.
4. Wait 100 µs.
5. Release PHY reset in *CTRL* bit 31.
6. Release ownership of the relevant PHY to the FW using the following flow:
    a. Get ownership of the software/firmware semaphore *SWSM.SWESMBI* (offset 0x5B50 bit 1):
        - Set the *SWSM.SWESMBI* bit.

- Read SWSM.
- If *SWSM.SWESMBI* was successfully set - the semaphore was acquired - otherwise, go back to step a.
- Clear the bit in *SW_FW_SYNC* that control the software ownership of the resource to indicate this resource is free.
- Release ownership of the software/firmware semaphore by clearing the *SWSM.SWESMBI* bit.

7. Wait for the relevant *CFG_DONE* bit (*EEMNGCTL.CFG_DONE0*, *EEMNGCTL.CFG_DONE1*, *EEMNGCTL.CFG_DONE2* or *EEMNGCTL.CFG_DONE3*).
8. Take ownership of the relevant PHY using the following flow:
   a. Get ownership of the software/firmware semaphore *SWSM.SWESMBI* (offset 0x5B50 bit 1):
      - Set the *SWSM.SWESMBI* bit.
      - Read SWSM.
      - If *SWSM.SWESMBI* was successfully set - the semaphore was acquired - otherwise, go back to step a.
      - This step assure that the internal firmware will not access the shared resources register (*SW_FW_SYNC*).

   b. Software reads the Software-Firmware Synchronization Register (*SW_FW_SYNC*) and checks both bits in the pair of bits that control the PHY it wishes to own.
      - If both bits are cleared (both firmware and other software does not own the PHY), software sets the software bit in the pair of bits that control the resource it wishes to own.
      - If one of the bits is set (firmware or other software owns the PHY), software tries again later.

   c. Release ownership of the software/software semaphore and the software/firmware semaphore by clearing *SWSM.SMBI* and *SWSM.SWESMBI* bits.
9. Configure the PHY.
10. Release ownership of the relevant PHY using the flow described in Section 4.7.2.

Note: Software PHY ownership should not exceed 100 mS. If Software takes PHY ownership for a longer duration, Firmware may implement a timeout mechanism and take ownership of the PHY.

4.3.3 Reset Effects

The resets affect the following registers and logic:

Table 4-2. 82580 Reset Effects - Common Resets

Reset Activation	LAN_PWR_GOOD	PE_ RST_N	SW CTRL. DEV_RST	In-Band PCIe Reset	FW Reset	Notes
LTSSM (PCIe back to detect/ polling)	X	X		X		
PCIe Link data path	X	X		X		
Read EEPROM (Per Function)			X			18.
Read EEPROM (Complete Load)	X	X		X		
PCI Configuration Registers- non sticky	X	X		X		3.
PCI Configuration Registers - sticky	X	X		X		4.
PCIe local registers	X	X		X		5.

Table 4-2. 82580 Reset Effects - Common Resets (Continued)

Reset Activation	LAN_PWR_GOOD	PE_RST_N	SW CTRL. DEV_RST	In-Band PCIe Reset	FW Reset	Notes
Data path	X	X	X	X		
On-die memories	X	X	X	X		15.
MAC, PCS, Auto Negotiation and other port related logic	X	X	X	X		
DMA	X	X	X	X		
Functions queue enable	X	X	X	X		
Function interrupt & statistics registers	X	X	X	X		
Wake Up (PM) Context	X	1.				7.
Wake Up Control Register	X					8.
Wake Up Status Registers	X					9.
Manageability Control Registers	X					10.
MMS Unit	X				X	
Wake-Up Management Registers	X	X	X	X		3.,11.
Memory Configuration Registers	X	X	X	X		3.
EEPROM and flash request	X		X			16.
PHY/SERDES PHY	X	X		X		2.
Strapping Pins	X	X		X		

Table 4-3. 82580 Reset Effects - Per Function Resets

Reset Activation	D3hot → D0	FLR	Port SW Reset (CTRL.RST)	Force TCO	EE Reset	PHY Reset	Notes
Read EEPROM (Per Function)	X	X	X	X	X		
PCI Configuration Registers RO							3.
PCI Configuration Registers MSI-X	X	X					6.
PCI Configuration Registers RW							
PCIe local registers							5.
Data path	X	X	X	X			
On-die memories	X	X	X	X			15.
MAC, PCS, Auto Negotiation and other port related logic	X	X	X	X			
DMA	X	X					17.
Wake Up (PM) Context							7.
Wake Up Control Register							8.
Wake Up Status Registers							9.
Manageability Control Registers							10.
Function queue enable	X	X	X	X			

Table 4-3. 82580 Reset Effects - Per Function Resets (Continued)

Reset Activation	D3hot → D0	FLR	Port SW Reset (CTRL.RST)	Force TCO	EE Reset	PHY Reset	Notes
Function interrupt & statistics registers	X	X	X				
Wake-Up Management Registers	X	X	X	X			3.,11.
Memory Configuration Registers	X	X	X	X			3.
EEPROM and flash request	X	X					16.
PHY/SERDES PHY	X	X		X		X	2.
Strapping Pins							

Notes:

1. If AUX_POWER = 0b the Wakeup Context is reset (*PME_Status* and *PME_En* bits should be 0b at reset if the 82580 does not support PME from D3cold).
2. The MMS unit must configure the PHY after any PHY reset.
3. The following register fields do not follow the general rules above:
   a. "*CTRL.SDP0_IODIR*, *CTRL.SDP1_IODIR*, *CTRL_EXT.SDP2_IODIR*, *CTRL_EXT.SDP3_IODIR*, *CONNSW.ENRGSRC* field, *CTRL_EXT.SFP_Enable*, *CTRL_EXT.LINK_MODE*, *CTRL_EXT.EXT_VLAN* and LED configuration registers are reset on LAN_PWR_GOOD only. Any EEPROM read resets these fields to the values in the EEPROM.
   b. The Aux Power Detected bit in the PCIe Device Status register is reset on LAN_PWR_GOOD and PE_RST_N (PCIe reset) assertion only.
   c. FLA - reset on LAN_PWR_GOODInternal Power only.
   d. The bits mentioned in the next note.
4. The following registers are part of this group:
   a. VPD registers
   b. Max payload size field in PCIe Capability Control register (offset 0xA8).
   c. Active State Link PM Control field, Common Clock Configuration field and Extended Synch field in PCIe Capability Link Control register (Offset 0xB0).
   d. Read Completion Boundary in the PCIe Link Control register (Offset 0xB0).
5. The following registers are part of this group:
   a. SWSM
   b. *GCR* (only part of the bits - see register description for details)
   c. FUNCTAG
   d. GSCL_1/2/3/4
   e. GSCN_0/1/2/3
   f. *SW_FW_SYNC* - only part of the bits - see register description for details.
6. The following registers are part of this group:
   a. *MSIX control* register, *MSIX PBA* and *MSIX per vector mask*.
7. The Wake Up Context is defined in the PCI Bus Power Management Interface Specification (Sticky bits). It includes:

- — *PME_En* bit of the Power Management Control/Status Register (*PMCSR*).
- — *PME_Status* bit of the Power Management Control/Status Register (*PMCSR*).
- — *Aux_En* in the PCIe registers
- — The device Requester ID (since it is required for the PM_PME TLP).

The shadow copies of these bits in the Wakeup Control Register are treated identically.

8. Refers to bits in the Wake Up Control Register that are not part of the Wake-Up Context (the *PME_En* and *PME_Status* bits).
9. The Wake Up Status Registers include the following:
   a. Wake Up Status Register
   b. Wake Up Packet Length.
   c. Wake Up Packet Memory.
10. The manageability control registers refer to the following registers:
   a. *MANC* 0x5820
   b. *MFUTP0-7* 0x5030 - 0x504C
   c. *MNGONLY* 0x5864
   d. *MAVTV0-7* 0x5010 - 0x502C
   e. *MDEF0-7* 0x5890 - 0x58AC
   f. *MDEF_EXT* 0x5930 - 0x594C
   g. *METF0-3* 0x5060 - 0x506C
   h. *MIPAF0-15* 0x58B0 - 0x58EC
   i. *MMAH/MMAL0*-1 0x5910 - 0x591C
   j. FWSM
11. The Wake-up Management Registers include the following:
   a. Wake Up Filter Control
   b. IP Address Valid
   c. IPv4 Address Table
   d. IPv6 Address Table
   e. Flexible Filter Length Table
   f. Flexible Filter Mask Table
12. The Other Configuration Registers include:
   a. General Registers
   b. Interrupt Registers
   c. Receive Registers
   d. Transmit Registers
   e. Statistics Registers
   f. Diagnostic Registers

Of these registers, *MTA[n]*, *VFTA[n]*, *WUPM[n]*, *FTFT[n]*, *FHFT[n]*, *FHFT_EXT[n]*, *TDBAH/TDBAL*, and *RDBAH/RDVAL* registers have no default value. If the functions associated with the registers are enabled they must be programmed by software. Once programmed, their value is preserved through all resets as long as power is applied to the 82580.

Note: In situations where the device is reset using the software reset *CTRL.RST* or *CTRL.DEV_RST* the transmit data lines are forced to all zeros. This causes a substantial number of symbol errors to be detected by the link partner. In TBI mode, if the duration is long enough, the link partner might restart the Auto-Negotiation process by sending "break-link" (/C/ codes with the configuration register value set to all zeros).

13. These registers include:
    a. MSI/MSI-X enable bits
    b. BME
    c. Error indications
14. These registers include:
    a. RXDCTL.Enable
15. The contents of the following memories are cleared to support the requirements of PCIe FLR:
    a. The Tx packet buffers
    b. The Rx packet buffers
16. Includes *EEC.REQ*, *EEC.GNT*, *FLA.REQ* and *FLA.GNT* fields.
17. The following DMA Registers are cleared only by LAN_PWR_GOOD, PCIe Reset or *CTRL.DEV_RST*: *DMCTLX*, *DTPARS*, *DRPARS* and *DDPARS.*
18. *CTRL.DEV_RST* assertion causes read of function related sections for all ports

4.3.4 PHY Behavior During a Manageability Session

During some manageability sessions (e.g. an IDER or SoL session as initiated by an external BMC), the platform is reset so that it boots from a remote media. This reset must not cause the Ethernet link to drop since the manageability session is lost. Also, the Ethernet link should be kept on continuously during the session for the same reasons. The 82580 therefore limits the cases in which the internal PHY would restart the link, by masking two types of events from the internal PHY:

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

The capability described in this section is disabled by default on LAN Power Good reset. The *Keep_PHY_Link_Up_En* bit in the EEPROM must be set to '1' to enable it. Once enabled, the feature is enabled until the next LAN Power Good (i.e., the 82580 does not revert to the hardware default value on PE_RST#, PCIe reset or any other reset but LAN Power Good).

When the Keep_PHY_Link_Up bit (also known as "veto bit") in the MANC Register is set, the following behaviors are disabled:

- The PHY is not reset on PE_RST# and PCIe resets (in-band and link drop). Other reset events are not affected - LAN Power Good reset, Device Disable, Force TCO, and PHY reset by software.
- The PHY does not change its power state. As a result link speed does not change.
- The 82580 does not initiate configuration of the PHY to avoid losing link.

The keep_PHY_link_up bit is set by the BMC through the Management Control command (See Section 10.3.2.1.5 for SMBus command and Section 10.3.1.6.1 for NC-SI command) on the sideband interface. It is cleared by the external BMC (again, through a command on the sideband interface) when the manageability session ends. Once the keep_PHY_link_up bit is cleared, the PHY updates its Dx state and acts accordingly (e.g. negotiates its speed).

The Keep_PHY_Link_Up bit is also cleared on de-assertion of the MAIN_PWR_OK input pin. MAIN_PWR_OK must be de-asserted at least 1 msec before power drops below its 90% value. This allows enough time to respond before auxiliary power takes over.

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

- On LAN Power Good
- When the BMC resets or initializes it
- On de-assertion of the MAIN_PWR_OK input pin. The BMC should set the bit again if it wishes to maintain speed on exit from Dr state.

4.4 Function Disable

4.4.1 General

For a LOM (Lan on Motherboard) design, it might be desirable for the system to provide BIOS-setup capability for selectively enabling or disabling LAN functions. It allows the end-user more control over system resource-management and avoid conflicts with add-in NIC solutions. The 82580 provides support for selectively enabling or disabling one or more LAN device(s) in the system.

4.4.2 Overview

Device presence (or non-presence) must be established early during BIOS execution, in order to ensure that BIOS resource-allocation (of interrupts, of memory or IO regions) is done according to devices that are present only. This is frequently accomplished using a BIOS CVDR (Configuration Values Driven on Reset) mechanism. The 82580 LAN-disable mechanism is implemented in order to be compatible with such a solution.

The 82580 provides two mechanisms to disable its LAN ports:

- Four pins (LANx_DIS_N, one per LAN port) are sampled on reset to determine the LAN-enable configuration
- All LAN ports except the first (LAN Port 0) might be disabled using EEPROM configuration, to avoid case were all LAN ports are disabled in the EEPROM and the EEPROM can't be accessed.
  - The LAN ports can be disabled by setting to 1 either the *LAN_DIS* or the *LAN PCI Disable* bit in the *Software Defined Pins Control* EEPROM word.

Disabling a LAN port affects the PCIe function it resides on. When function 0 is disabled (either LAN0, LAN1, LAN2 or LAN3), two different behaviors are possible:

- Dummy Function mode — In some system, it is required to keep all the functions at their respective location, even when other functions are disabled. In Dummy Function mode, if function #0 (either LAN0, LAN1, LAN2 or LAN3) 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 0x10A6, Class Code 0xFF0000). In addition, the function does not require any memory or I/O space, and does not require an interrupt line.

- Legacy mode - when function 0 is disabled, then the port residing on the next existing function moves to reside on function 0. All other ports keeps their respective locations.

The disabled LAN port is still available for manageability purposes if it was disabled using the *LAN_PCI_DIS* bit of the *Software Defined Pins Control* word in the EEPROM or if it was disabled through the pin mechanism and the *PHY_in_LAN_Disable* bit in the *Software Defined Pins Control* word in the EEPROM is cleared.

Mapping between function and LAN ports as a function of the LAN0_DIS_N, LAN1_DIS_N, LAN2_DIS_N and LAN3_DIS_N pins and the *FACTPS.LAN Function Sel* bit is summarized in the following tables.

Note: PCIe Functions Mapping of dual port SKU is like 4 port SKU with Port 2 and Port 3 disabled.

Table 4-4. PCI Functions Mapping (Legacy Mode)

Configuration Options:					PCIe Function to LAN Port Mapping			
LAN Port 0 enabled	**LAN Port1 enabled**	**LAN Port2 enabled**	**LAN Port 3 enabled**	**FACTPS.LAN Function Sel**	**PCIe Function 0**	**PCIe Function 1**	**PCIe Function 2**	**PCIe Function 3**
		LAN Port 2 and 3 are never enabled in 82580DB SKU						
Yes	Yes	Yes	Yes	0	LAN Port 0	LAN Port 1	LAN Port 2	LAN Port 3
Yes	Yes	Yes	No	0	LAN Port 0	LAN Port 1	LAN Port 2	Disabled
Yes	Yes	No	Yes	0	LAN Port 0	LAN Port 1	Disabled	LAN Port 3
Yes	Yes	No	No	0	LAN Port 0	LAN Port 1	Disabled	Disabled
Yes	No	Yes	Yes	0	LAN Port 0	Disabled	LAN Port 2	LAN Port 3
Yes	No	Yes	No	0	LAN Port 0	Disabled	LAN Port 2	Disabled
Yes	No	No	Yes	0	LAN Port 0	Disabled	Disabled	LAN Port 3
Yes	No	No	No	0	LAN Port 0	Disabled	Disabled	Disabled
No	Yes	Yes	Yes	0	LAN Port 1	Disabled	LAN Port 2	LAN Port 3
No	Yes	Yes	No	0	LAN Port 1	Disabled	LAN Port 2	Disabled
No	Yes	No	Yes	0	LAN Port 1	Disabled	Disabled	LAN Port 3
No	Yes	No	No	0	LAN Port 1	Disabled	Disabled	Disabled
No	No	Yes	Yes	0	LAN Port 2	Disabled	Disabled	LAN Port 3
No	No	Yes	No	0	LAN Port 2	Disabled	Disabled	Disabled
No	No	No	Yes	0	LAN Port 3	Disabled	Disabled	Disabled
Yes	Yes	Yes	Yes	1	LAN Port 3	LAN Port 2	LAN Port 1	LAN Port 0
Yes	Yes	No	Yes	1	LAN Port 3	Disabled	LAN Port 1	LAN Port 0
Yes	No	Yes	Yes	1	LAN Port 3	LAN Port 2	Disabled	LAN Port 0
Yes	No	No	Yes	1	LAN Port 3	Disabled	Disabled	LAN Port 0
No	Yes	Yes	Yes	1	LAN Port 3	LAN Port 2	LAN Port 1	Disabled
No	Yes	No	Yes	1	LAN Port 3	Disabled	LAN Port 1	Disabled
No	No	No	Yes	1	LAN Port 3	Disabled	Disabled	Disabled
Yes	Yes	Yes	No	1	LAN Port 2	Disabled	LAN Port 1	LAN Port 0
Yes	No	Yes	No	1	LAN Port 2	Disabled	Disabled	LAN Port 0
No	Yes	Yes	No	1	LAN Port 2	Disabled	LAN Port 1	Disabled

Table 4-4. PCI Functions Mapping (Legacy Mode)

Configuration Options:					PCIe Function to LAN Port Mapping			
LAN Port 0 enabled	LAN Port1 enabled	LAN Port2 enabled	LAN Port 3 enabled	FACTPS.LAN Function Sel	PCIe Function 0	PCIe Function 1	PCIe Function 2	PCIe Function 3
		LAN Port 2 and 3 are never enabled in 82580DB SKU						
No	No	Yes	No	1	LAN Port 2	Disabled	Disabled	Disabled
Yes	Yes	No	No	1	LAN Port 1	Disabled	Disabled	LAN Port 0
No	Yes	No	No	1	LAN Port 1	Disabled	Disabled	Disabled
Yes	No	No	No	1	LAN Port 0	Disabled	Disabled	Disabled
No	No	No	No	0 or 1	All PCI functions are disabled. Device is in low power mode unless used by manageability			

The following EEPROM bits control Function Disable:

- PCI functions 1 to 3 can be enabled or disabled according to the EEPROM “LAN PCIe Function Disable” bit in the *Software Defined Pins Control* EEPROM word.

Note: PCI function 0 can not be disabled via EEPROM.

- The *LAN Disable* EEPROM bit in the *Software Defined Pins Control* EEPROM word, indicates which function and LAN port are disabled. When this bit is set port is not available to manageability channel.
- The *LAN Function Sel* EEPROM bit in the *Functions Control* EEPROM word, defines the correspondence between LAN Port and PCI function
- The *Dummy Function Enable* EEPROM bit enables the Dummy Function mode for function 0. Default value is disabled.
- The *PHY_in_LAN_disable* EEPROM bit in the *Software Defined Pins Control* EEPROM word, controls the availability of the disabled function to manageability channel.

When a particular LAN is fully disabled, all internal clocks to that LAN are disabled, the device is held in reset, and the internal PHY for that LAN is powered-down. In both modes, the device does not respond to PCI configuration cycles. Effectively, the LAN device becomes invisible to the system from both a configuration and power-consumption standpoint.

4.4.3 Control Options

The functions have a separate enabling Mechanism. Any function that is not enabled does not function and does not expose its PCI configuration registers.

Table 4-5. Strapping for Control Options

Function	Control Options
LAN 0	Strapping Option + EEPROM offset 0x20 bit 13 (full/PCI only disable in case of strap)
LAN 1	Strapping Option + EEPROM offset 0x20 bit 13 (full/PCI only disable in case of strap)/ EEPROM offset 0x20 bit 11 (full disable) / EEPROM offset 0x20 bit 10 (PCI only disable)
LAN 2	Strapping Option + EEPROM offset 0x20 bit 13 (full/PCI only disable in case of strap)/ EEPROM offset 0x20 bit 11 (full disable) / EEPROM offset 0x20 bit 10 (PCI only disable)
LAN 3	Strapping Option + EEPROM offset 0x20 bit 13 (full/PCI only disable in case of strap)/ EEPROM offset 0x20 bit 11 (full disable) / EEPROM offset 0x20 bit 10 (PCI only disable)

The 82580 strapping option for LAN Disable feature are:

Table 4-6. Strapping for LAN Disable

Symbol	Ball #	Name and Function
LAN0_DIS_N	F14	This pin is a strapping option pin always active. This pin has an internal weak pull-up resistor. In case this pin is not connected or driven hi during init time, LAN 0 is enabled. In case this pin is driven low during init time, LAN 0 is disabled. This pin is also used for testing and scan. When used for testing or scan, the LAN disable functionality is not active.
LAN1_DIS_N	E14	This pin is a strapping option pin always active. This pin has an internal weak pull-up resistor. In case this pin is not connected or driven hi during init time, LAN 1 is enabled. In case this pin is driven low during init time, LAN 1 function is disabled. This pin is also used for testing and scan. When used for testing or scan, the LAN disable functionality is not active.
LAN2_DIS_N	D14	This pin is a strapping option pin always active. This pin has an internal weak pull-up resistor. In case this pin is not connected or driven hi during init time, LAN 2 is enabled. In case this pin is driven low during init time, LAN 2 is disabled. This pin is also used for testing and scan. When used for testing or scan, the LAN disable functionality is not active.
LAN3_DIS_N	C14	This pin is a strapping option pin always active. This pin has an internal weak pull-up resistor. In case this pin is not connected or driven hi during init time, LAN 3 is enabled. In case this pin is driven low during init time, LAN 3 is disabled. This pin is also used for testing and scan. When used for testing or scan, the LAN disable functionality is not active.

4.4.4 Event Flow for Enable/Disable Functions

This section describes the driving levels and event sequence for device functionality. Following a Power on Reset / LAN_PWR_GOOD / PE_RST_N/ In-Band reset the LANx_DIS_N signals should be driven hi (or left unconnected) for nominal operation. If any of the LAN functions are not required statically its associated Disable strapping pin can be tied statically to low.

Case A - BIOS Disables the LAN Function at boot time by using strapping:

1. Assume that following power up sequence LANx_DIS_N signals are driven high.
2. The PCIe link is established following the PE_RST_N.
3. BIOS recognizes that a LAN function in the 82580 should be disabled.
4. The BIOS drives the LANx_DIS_N signal to low level.
5. The BIOS should assert the PCIe reset, either in-band or via PE_RST_N.
6. As a result, the 82580 samples the LANx_DIS_N signals and 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 or changed to dummy function).
8. Proceed with Nominal operation.
9. Re-enable could be done by driving the LANx_DIS_N signal high and then request the user to issue a warm boot that generate bus enumeration.

4.4.4.1 Multi-Function Advertisement

If all but one of the LAN devices are disabled, the 82580 is no longer a multi-function device. The 82580 normally reports a 0x80 in the PCI Configuration Header field Header Type, indicating multi-function capability. However, if only a single LAN is enabled, the 82580 reports a 0x0 in this field to signify single-function capability.

4.4.4.2 Legacy Interrupts Utilization

When more than one LAN device is enabled, the 82580 can utilize the INTA# to INTC# interrupts for interrupt reporting. The EEPROM *Initialization Control Word 3* (bits 12:11) associated with each LAN device controls which of these interrupts are used for each LAN device.   The specific interrupt pin utilized is reported in the PCI Configuration Header Interrupt Pin field associated with each LAN device.

However, if only one LAN device is enabled, then the INTA# must be used for this LAN device, regardless of the EEPROM configuration. Under these circumstances, the Interrupt Pin field of the PCI Header always reports a value of 0x1, indicating INTA# usage.

4.4.4.3 Power Reporting

When more than one LAN function is enabled, the PCI Power Management Register Block has the capability of reporting a "Common Power" value. The Common Power value is reflected in the Data field of the PCI Power Management registers. The value reported as Common Power is specified via the *LAN Power Consumption* EEPROM word (word 0x22), and is reflected in the Data field whenever the Data_Select field has a value of 0x8 (0x8 = Common Power Value Select).

When only one LAN is enabled, the 82580 appears as a single-function device, the Common Power value, if selected, reports 0x0 (undefined value), as Common Power is undefined for a single-function device.

4.5 Device Disable

For a LOM design, it might be desirable for the system to provide BIOS-setup capability for selectively enabling or disabling LOM devices. This might allow the end-user more control over system resource-management; avoid conflicts with add-in NIC solutions, etc. The 82580 provides support for selectively enabling or disabling it.

Note: If the 82580 is configured to provide a 50MHz NC-SI clock (via the NC-SI Output Clock EEPROM bit), then the device should not be disabled.

Device Disable is initiated by assertion of the asynchronous DEV_OFF_N pin. The DEV_OFF_N pin should always be connected to enable correct device operation.

The EEPROM *Power Down Enable* bit (see Section 6.2.16) enables device disable mode (Hardware default is that this mode is disabled).

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

Note: Behavior of SDP pins in device disable mode is controlled by the *SDP_IDDQ_EN* EEPROM bit (See Section 6.2.2).

Assertion or deassertion of PCIe PE_RST_N does not have any effect while the device is in device disable mode (i.e., the device stays in the respective mode as long as DEV_OFF_N is asserted). However, the device might momentarily exit the device disable mode from the time PCIe PE_RST_N is de-asserted again and until the EEPROM is read.

During power-up, the DEV_OFF_N pin is ignored until the EEPROM is read. From that point, the device might enter Device Disable if DEV_OFF_N is asserted.

Note: De-assertion of the DEV_OFF_N pin causes a fundamental reset to the 82580.

Note to system designer: The DEV_OFF_N pin should maintain its state during system reset and system sleep states. It should also insure the proper default value on system power-up. For example, one could use a GPIO pin that defaults to '1' (enable) and is on system suspend power (i.e., it maintains state in S0-S5 ACPI states).

4.5.1 BIOS Handling of Device Disable

1. Assume that following power up sequence the DEV_OFF_N signal is driven high (else it is already disabled).
2. The PCIe is established following the PE_RST_N.
3. BIOS recognize that the whole Device should be disabled.
4. The BIOS drive the DEV_OFF_N signal to the low level.
5. As a result, the 82580 samples the DEV_OFF_N signal and enters the device disable mode.
6. The BIOS places the Link in the Electrical IDLE state (at the other end of the PCIe link) by clearing the LINK Disable bit in the Link Control Register.
7. BIOS might start with the Device enumeration procedure (all of the Device functions are invisible).
8. Proceed with Nominal operation.
9. Re-enable could be done by driving the DEV_OFF_N signal high followed later by bus enumeration.

4.6 Software Initialization and Diagnostics

4.6.1 Introduction

This chapter discusses general software notes for the 82580, especially initialization steps. This includes general hardware, power-up state, basic device configuration, initialization of transmit and receive operation, link configuration, software reset capability, statistics, and diagnostic hints.

4.6.2 Power Up State

When the 82580 powers up it reads the EEPROM. The EEPROM contains sufficient information to bring the link up and configure the 82580 for manageability and/or APM wakeup. However, software initialization is required for normal operation.

The power-up sequence, as well as transitions between power states, are described in section 4.1.1. The detailed timing is given in Section 5.5. The next section gives more details on configuration requirements.

4.6.3 Initialization Sequence

The following sequence of commands is typically issued to the device by the software device driver in order to initialize the 82580 to normal operation. The major initialization steps are:

- Disable Interrupts - see Interrupts during initialization.
- Issue Global Reset and perform General Configuration - see Global Reset and General Configuration.
- Setup the PHY and the link - see Link Setup Mechanisms and Control/Status Bit Summary.
- Initialize all statistical counters - see Initialization of Statistics.
- Initialize Receive - see Receive Initialization.
- Initialize Transmit - see Transmit Initialization.

- Enable Interrupts - see Interrupts during initialization.

4.6.4 Interrupts During Initialization

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

After the initialization is done, a typical driver enables the desired interrupts by writing to the *EIMS* (Extended Interrupt Mask Set) register.

4.6.5 Global Reset and General Configuration

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

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

- *FD* bit should be set per interface negotiation (if done in software), or is set by the hardware if the interface is Auto-Negotiating. This is reflected in the *Device Status* Register in the Auto-Negotiation case.
- Speed is determined via Auto-Negotiation by the PHY, Auto-Negotiation by the PCS layer in SGMII/SerDes mode, or forced by software if the link is forced. Status information for speed is also readable in the *STATUS* register.
- *ILOS* bit should normally be set to 0.

4.6.6 Flow Control Setup

If flow control is enabled, program the *FCRTL0*, *FCRTH0*, *FCTTV* and *FCRTV* registers. In order to avoid packet losses, *FCRTH* should be set to a value equal to at least two max size packet below the receive buffer size. E.g. Assuming a packet buffer size of 32 KB and expected max size packet of 9.5K, the *FCRTH0* value should be set to 32 - 2 * 9.5 = 17KB i.e. *FCRTH0.RTH* should be set to 0x440.

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

4.6.7 Link Setup Mechanisms and Control/Status Bit Summary

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

4.6.7.1 PHY Initialization

Refer to the PHY documentation for the initialization and link setup steps. The device driver uses the *MDIC* register to initialize the PHY and setup the link. Section 3.5.4.4 describes the link setup for the internal copper PHY. Section 3.5.2.2 Section describes the usage of the *MDIC* register.

4.6.7.2 MAC/PHY Link Setup (CTRL_EXT.LINK_MODE = 00b)

This section summarizes the various means of establishing proper MAC/PHY link setups, differences in MAC CTRL register settings for each mechanism, and the relevant MAC status bits. The methods are ordered in terms of preference (the first mechanism being the most preferred).

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

CTRL.FD Don't care; duplex setting is established from PHY's internal indication to the MAC (FDX) after PHY has auto-negotiated a successful link-up.

CTRL.SLU Must be set to 1 by software to enable communications between MAC and PHY.

CTRL.RFCE Must be set to 1 by software after reading capabilities from PHY registers and resolving the desired setting.

CTRL.TFCE Must be set to 1 by software after reading capabilities from PHY registers and resolving the desired setting.

CTRL.SPEED Don't care; speed setting is established from PHY's internal indication to the MAC (*SPD_IND*) after PHY has auto-negotiated a successful link-up.

STATUS.FD Reflects the actual duplex setting (FDX) negotiated by the PHY and indicated to MAC.

STATUS.LU Reflects link indication (LINK) from PHY qualified with *CTRL.SLU* (set to 1).

STATUS.SPEED Reflects actual speed setting negotiated by the PHY and indicated to the MAC (*SPD_IND*).

4.6.7.2.2 MAC Duplex and Speed Settings Forced by Software Based on Resolution of PHY (CTRL.FRCDPLX = 1b, CTRL.FRCSPD = 1b)

CTRL.FD Set by software based on reading PHY status register after PHY has auto-negotiated a successful link-up.

CTRL.SLU Must be set to 1 by software to enable communications between MAC and PHY.

CTRL.RFCE Must be set to 1 by software after reading capabilities from PHY registers and resolving the desired setting.

CTRL.TFCE Must be set to 1 by software after reading capabilities from PHY registers and resolving the desired setting.

CTRL.SPEED Set by software based on reading PHY status register after PHY has auto-negotiated a successful link-up.

STATUS.FD Reflects the MAC forced duplex setting written to *CTRL.FD*.

STATUS.LU Reflects link indication (LINK) from PHY qualified with *CTRL.SLU* (set to 1).

STATUS.SPEED Reflects MAC forced speed setting written in *CTRL.SPEED*.

4.6.7.2.3 MAC/PHY Duplex and Speed Settings Both Forced by Software (Fully-Forced Link Setup) (CTRL.FRCDPLX = 1b, CTRL.FRCSPD = 1b, CTRL.SLU = 1b)

CTRL.FD Set by software to desired full/half duplex operation (must match duplex setting of PHY).

CTRL.SLU Must be set to 1 by software to enable communications between MAC and PHY. PHY must also be forced/configured to indicate positive link indication (LINK) to the MAC.

CTRL.RFCE Must be set to 1 by software to desired flow-control operation (must match flow-control settings of PHY).

CTRL.TFCE Must be set to 1 by software to desired flow-control operation (must match flow-control settings of PHY).

CTRL.SPEED Set by software to desired link speed (must match speed setting of PHY).

STATUS.FD Reflects the MAC duplex setting written by software to *CTRL.FD*.

STATUS.LU Reflects 1 (positive link indication LINK from PHY qualified with *CTRL.SLU*). Note that since both *CTRL.SLU* and the PHY link indication LINK are forced, this bit set does not guarantee that operation of the link has been truly established.

STATUS.SPEED Reflects MAC forced speed setting written in *CTRL.SPEED*.

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

Link setup procedures using an external SERDES interface mode:

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

CTRL.FD Ignored; duplex is set by priority resolution of *PCS_ANDV* and *PCS_LPAB*.

CTRL.SLU Must be set to 1 by software to enable communications to the SerDes.

CTRL.RFCE Set by Hardware according to auto negotiation resolution[1].

CTRL.TFCE Set by Hardware according to auto negotiation resolution[1].

CTRL.SPEED Ignored; speed always 1000Mb/s when using SerDes mode communications.

STATUS.FD Reflects hardware-negotiated priority resolution.

STATUS.LU Reflects *PCS_LSTS.AN COMPLETE* (Auto-Negotiation complete).

STATUS.SPEED Reflects 1000Mb/s speed, reporting fixed value of (10)b.

PCS_LCTL.FSD Must be zero.

PCS_LCTL.Force Flow Control Must be zero[1].

PCS_LCTL.FSV Must be set to 10b. Only 1000 Mb/s is supported in SerDes mode.

PCS_LCTL.FDV Ignored; duplex is set by priority resolution of *PCS_ANDV* and *PCS_LPAB*.

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

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

CTRL.FD Must be set to 1b. - only full duplex is supported in SerDes mode.

CTRL.SLU Must be set to 1b by software to enable communications to the SerDes.

CTRL.RFCE Must be 0b (No Auto-negotiation).

CTRL.TFCE Must be 0b (No Auto-negotiation).

CTRL.SPEED Must be set to 10b. Only 1000 Mb/s is supported in SerDes mode.

STATUS.FD Reflects the value written by software to *CTRL.FD*.

STATUS.LU Reflects whether the PCS detected comma symbols, qualified with *CTRL.SLU* (set to 1b).

STATUS.SPEED Reflects 1000Mb/s speed, reporting fixed value of (10)b.

PCS_LCTL.FSD Must be set to 1b by software to enable communications to the SerDes.

PCS_LCTL.Force Flow Control Must be set to 1b.

PCS_LCTL.FSV Must be set to 10b. Only 1000 Mb/s is supported in SerDes mode.

PCS_LCTL.FDV Must be set to 1b - only full duplex is supported in SerDes mode.

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

Link setup procedures using an external SGMII interface mode:

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

CTRL.FD Ignored; duplex is set by priority resolution of PCS_ANDV and PCS_LPAB.

CTRL.SLU Must be set to 1b by software to enable communications to the SerDes.

CTRL.RFCE Must be set by software after reading flow control resolution from PCS registers.

CTRL.TFCE Must be set by software after reading flow control resolution from PCS registers.

CTRL.SPEED Ignored; speed setting is established from SGMII's internal indication to the MAC after SGMII PHY has auto-negotiated a successful link-up.

STATUS.FD Reflects hardware-negotiated priority resolution.

STATUS.LU Reflects *PCS_LSTS.Link OK*

STATUS.SPEED Reflects actual speed setting negotiated by the SGMII and indicated to the MAC.

PCS_LCTL.Force Flow Control Ignored.

PCS_LCTL.FSD Should be set to zero.

PCS_LCTL.FSV Ignored; speed is set by priority resolution of *PCS_ANDV* and *PCS_LPAB*.

PCS_LCTL.FDV Ignored; duplex is set by priority resolution of *PCS_ANDV* and *PCS_LPAB*.

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

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

Link setup procedures using an external 1000BASE-KX Server Backplane interface mode:

CTRL.FD Must be set to 1b. 1000BASE-KX always in full duplex mode.

CTRL.SLU Must be set to 1b by software to enable communications to the SerDes.

CTRL.RFCE Must be 0b (No Auto-negotiation).

CTRL.TFCE Must be 0b (No Auto-negotiation).

CTRL.SPEED Must be set to 10b. Only 1000 Mb/s is supported in 1000BASE-KX mode.

STATUS.FD Reflects the value written by software to *CTRL.FD*.

STATUS.LU Reflects whether the PCS detected comma symbols, qualified with *CTRL.SLU* (set to 1b).

STATUS.SPEED Reflects 1000Mb/s speed, reporting fixed value of (10b).

PCS_LCTL.FSD Must be set to 1b by software to enable communications to the 1000BASE-KX SerDes.

PCS_LCTL.Force Flow Control Must be set to 1b.

PCS_LCTL.FSV Must be set to 10b. Only 1000 Mb/s is supported in 1000BASE-KX mode.

PCS_LCTL.FDV Must be set to 1b - only full duplex is supported in 1000BASE-KX mode.

4.6.8 Initialization of Statistics

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

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

4.6.9 Receive Initialization

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

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

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

The following should be done once per receive queue needed:

- 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* of the queue according to the size of the buffers, the required header handling and the drop policy.
- If header split or header replication is required for this queue, program the *PSRTYPE* register according to the required headers.
- Enable the queue by setting *RXDCTL.ENABLE*. In the case of queue zero, the enable bit is set by default - so the ring parameters should be set before *RCTL.RXEN* is set.
- Poll the *RXDCTL* register until the *ENABLE* bit is set. The tail should not be bumped before this bit was read as one.
- Program the direction of packets to this queue according to the mode selected in the *MRQC* register. Packets directed to a disabled queue are dropped.

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

4.6.9.1 Initialize the Receive Control Register

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

4.6.9.2 Dynamic Enabling and Disabling of Receive Queues

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

Enabling:

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

Note: If there are still packets in the packet buffer directed to this queue according to previous settings, they are received after the queue is re-enabled. In order to avoid this condition, the software might poll the *PBRWAC* register. Once a an empty condition of the relevant packet buffer is detected or 2 wrap around occurrences are detected the queue can be re-enabled.

Disabling:

- Disable the direction of packets to this queue.
- Disable the queue by clearing *RXDCTL.ENABLE*. The 82580 stops fetching and writing back descriptors from this queue immediately. The 82580 eventually completes the storage of one buffer allocated to this queue. Any further packet directed to this queue is dropped. If the currently processed packet is spread over more than one buffer, all subsequent buffers are not written.
- The 82580 clears *RXDCTL.ENABLE* only after all pending memory accesses to the descriptor ring or to the buffers are done. The driver should poll this bit before releasing the memory allocated to this queue.

The RX path might be disabled only after all Rx queues are disabled.

4.6.10 Transmit Initialization

- Program the *TCTL* register according to the MAC behavior needed.

If work in half duplex mode is expected, program the *TCTL_EXT.COLD* field. For internal PHY mode the default value of 0x42 is OK. For SGMII mode, a value reflecting the 82580 and the PHY SGMII delays should be used. A suggested value for a typical PHY is 0x46 for 10 Mbps and 0x4C for 100 Mbps.

The following should be done once per transmit queue:

- Allocate a region of memory for the transmit descriptor list.
- Program the descriptor base address with the address of the region.
- Set the length register to the size of the descriptor ring.
- Program the *TXDCTL* register with the desired TX descriptor write back policy. Suggested values are:
  - — *WTHRESH* = 1b
  - — All other fields 0b.
- If needed, set the *TDWBAL*/*TWDBAH* to enable head write back
- Enable the queue using *TXDCTL.ENABLE* (queue zero is enabled by default).
- Poll the *TXDCT*L register until the *ENABLE* bit is set.

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

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

4.6.10.1 Dynamic Queue Enabling and Disabling

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

Enabling:

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

Disabling:

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

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

4.6.11 Virtualization Initialization Flow

4.6.11.1 VMDq Mode

4.6.11.1.1 Global Filtering and Offload Capabilities

- Select the VMDQ pooling method - MAC/VLAN filtering for pool selection. *MRQC.Multiple Receive Queues Enable* = 011b.
- Set the *RPLOLR* and *RPLPSRTYPE* registers to define the behavior of replicated packets.

- Configure *VT_CTL.DEF_PL* to define the default pool. If packets with no pools should be dropped, set *VT_CTL.Dis_def_Pool* field.
- If needed, enable padding of small packets via the *RCTL.PSP*

4.6.11.1.2 Mirroring rules

For each mirroring rule to be activated:

a. Set the type of traffic to be mirrored in the *VMRCTL[n]* register.

b. Set the mirror pool in the *VMRCTL[n].MP*

c. For pool mirroring, set the *VMRVM[n]* register with the pools to be mirrored.

d. For VLAN mirroring, set the *VMVRLAN[n]* with the indexes from the *VLVF* registers of the VLANs to be mirrored.

4.6.11.1.3 Per Pool Settings

As soon as a pool of queues is associated to a VM the software should set the following parameters:

1. Address filtering:
   a. The unicast MAC address of the VM by enabling the pool in the *RAH/RAL* registers.
   b. If all the MAC addresses are used, the unicast hash table (*UTA*) can be used. Pools servicing VMs whose address is in the hash table should be declared as so by setting the *VMOLR.ROPE*. Packets received according to this method didn’t pass perfect filtering and are indicated as such.
   c. Enable the pool in all the *RAH*/*RA*L registers representing the multicast MAC addresses this VM belongs to.
   d. If all the MAC addresses are used, the multicast hash table (*MTA*) can be used. Pools servicing VMs using multicast addresses in the hash table should be declared as so by setting the *VMOLR.ROMPE*. Packets received according to this method didn’t pass perfect filtering and are indicated as such.
   e. Define whether this VM should get all multicast/broadcast packets in the same VLAN via the *VMOLR.MPE* and *VMOLR.BAM fields*
   f. Enable the pool in each *VLVF* register representing a VLAN this VM belongs to.
   g. Define whether the pool belongs to the default VLAN and should accept untagged packets via the *VMOLR.AUPE* field
2. Offloads
   a. Define whether VLAN header should be stripped from the packet (defined by *VMOLR.strvlan*).
   b. Set which header split is required via the *PSRTYPE* register.
   c. Set whether larger than standard packet are allowed by the VM and what is the largest packet allowed (jumbo packets support) via *VMOLR.*RLPML & *VMOLR.RLE*.
3. Queues
   a. Enable Rx & Tx queues as described in Section 4.6.9 & Section 4.6.10
   b. For each Rx queue a drop/no drop flag can be set in *SRRCTL.DROP_EN*, controlling the behavior in cases no receive buffers are available in the queue to receive packets. The usual behavior is to allow drops in order to avoid head of line blocking, unless a no-drop behavior is needed for some type of traffic (e.g. storage).

4.6.11.1.4 Security Features

4.6.11.1.4.1 Storm control

The driver may set limits to the broadcast or multicast traffic it can receive.

1. It should set the how many 64 byte chunks of Broadcast and Multicast traffic are acceptable per interval via the *BSCTRH* and *MSCTRH* respectively.
2. It should then set the interval to be used via the *SCCRL.Interval* field and which action to take when the broadcast or multicast traffic crosses the programmed threshold via the *SCCRL.BDIPW*, *SCCRL.BDICW*, *SCCRL.MDIPW*, and *SCCRL.MDICW* fields.
3. The driver may be notified of storm control events through the *ICR.SCE* interrupt cause.

4.7 Access to shared resources

Part of the resources in the 82580 are shared between several software entities - namely the drivers of the four ports and the internal firmware. In order to avoid contentions, a driver that needs to access one of these resources should use the flow described in Section 4.7.1 in order to acquire ownership of this resource and use the flow described in Section 4.7.2 in order to relinquish ownership of this resource.

The shared resources are:

1. EEPROM.
2. All PHYs or SerDes ports.
3. CSRs accessed by the internal firmware after the initialization process. Currently there are no such CSRs.
4. The flash.
5. Software to Software Mailbox

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

4.7.1 Acquiring ownership over a shared resource

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

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

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

   This step assure that the internal firmware will not access the shared resources register (*SW_FW_SYNC*).
3. Software reads the Software-Firmware Synchronization Register (SW_FW_SYNC) and checks both bits in the pair of bits that control the resource it wishes to own.
   a. If both bits are cleared (both firmware and other software does not own the resource), software sets the software bit in the pair of bits that control the resource it wishes to own.
   b. If one of the bits is set (firmware or other software owns the resource), software tries again later.

4. Release ownership of the software/software semaphore and the software/firmware semaphore by clearing SWSM.SMBI and SWSM.SWESMBI bits.
5. At this stage, the shared resources is owned by the driver and it may access it. The *SWSM* and *SW_FW_SYNC* registers can now be used to take ownership of another shared resources.

Notes:

- Software ownership of *SWSM.SWESMBI* bit should not exceed 100 mS. If Software takes ownership for a longer duration, Firmware may implement a timeout mechanism and take ownership of the *SWSM.SWESMBI* bit.
- Software ownership of bits in *SW_FW_SYNC* register should not exceed 1 Second. If Software takes ownership for a longer duration, Firmware may implement a timeout mechanism and take ownership of the relevant *SW_FW_SYNC* bits.

4.7.2 Releasing ownership over a shared resource

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

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

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

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

4.7.3 Software to Software Mailbox

In order to allow different the 82580 drivers to coordinate activities, a simple mailbox mechanism is defined. This mechanism allows each driver to send a broadcast message to all the other drivers on the same device.

In order to send a message the following flow should be used:

1. The Driver that wants to send the message should acquire the mailbox semaphore (*SW_FW_SYNC.SW_MB_SM*) using the flow described in the previous sections.
2. The Driver should then write the message in the *SWMBWR* register.
3. All the drivers will then receive an interrupt (except for the driver that initiated the message) via the *SWMB* cause in the *ICR* registers.

4. All the drivers will read the *SWMB0*, *SWMB1*, *SWMB2 and SWMB3* registers to understand which driver sent a message.

Note: The mapping of *SWMB0*, *SWMB1*, *SWMB2 and SWMB3* registers is according to the physical ports. A function can detect which physical port it is mapped to, by reading the *STATUS.LAN ID* field.

5. If the message requires an acknowledgment from the other drivers, each driver may write an acknowledge message through their *SWMBWR* register.
6. The driver that sent the original message can then poll the *SWMB0*, *SWMB1*, *SWMB2 and SWMB3 registers* for acknowledge messages.
7. After the message was acknowledged, the driver that sent the original message should:
    a. clear the value in the *SWMBWR* register to avoid confusion when future messages are sent.
    b. Release the Software mailbox semaphore (*SW_FW_SYNC.SW_MB_SM*) using the flow described in the previous section.

§ §

5.0 Power Management

This section describes how power management is implemented in the 82580. The 82580 supports the Advanced Configuration and Power Interface (ACPI) specification as well as Advanced Power Management (APM).

Note: Power management can be disabled via the *power management* bit in the *Initialization Control Word 1* EERPROM word (see Section 6.2.2).

5.1 General Power State Information

5.1.1 PCI Device Power States

The PCIe Specification defines function power states (D-states) that enable the platform to establish and control power states for the 82580 ranging from fully on to fully off (drawing no power) and various in-between levels of power-saving states, annotated as D0-D3. Similarly, PCIe defines a series of link power states (L-states) that work specifically within the link layer between the 82580 and its upstream PCIe port (typically in the host chipset).

Since the 82580 is a multi-port device, each of its PCI functions may be in a different state at any given moment. The device power state is defined by the most active function. For example, if function 0 is in D0 state and all other functions are in D3 state, device state is D0. Link state follows the device state. For a given device D-state, only certain L-states are possible as follows:

- D0 (fully on): The 82580 is completely active and responsive during this D-state. The link can be in either L0 or a low-latency idle state referred to as L0s. Minimizing L0s exit latency is paramount for enabling frequent entry into L0s while facilitating performance needs via a fast exit. A deeper link power state, L1 state, is supported as well.
- D1 and D2: These modes are not supported by the 82580.
- D3 (off): Two sub-states of D3 are supported:
  - D3hot, where primary power is maintained.
  - D3cold, where primary power is removed.

  Link states are mapped into device states as follows:
  - D3hot maps to L1 to support clock removal on mobile platforms
  - D3cold maps to L2 if auxiliary power is supported on 82580 with wake-capable logic, or to L3 if no power is delivered to 82580. A sideband PE_WAKE_N mechanism is supported to interface wake-enabled logic on mobile platforms during the L2 state.

5.1.2 PCIe Link Power States

Configuring the 82580 into a D-state automatically causes the PCIe link to transition to the appropriate L-state.

- L2/L3 Ready: This link state prepares the PCIe link for the removal of power and clock. The 82580 is in the D3hot state and is preparing to enter D3cold. The power-saving opportunities for this state include, but are not limited to, clock gating of all PCIe architecture logic, shutdown of the PLL, and shutdown of all transceiver circuitry.
- L2: This link state is intended to comprehend D3cold with auxiliary power support. Note that sideband PE_WAKE_N signaling exists to cause wake-capable devices to exit this state. The power-saving opportunities for this state include, but are not limited to, shutdown of all transceiver circuitry except detection circuitry to support exit, clock gating of all PCIe logic, and shutdown of the PLL as well as appropriate platform voltage and clock generators.
- L3 (link off): Power and clock are removed in this link state, and there is no auxiliary power available. To bring the 82580 and its link back up, the platform must go through a boot sequence where power, clock, and reset are reapplied appropriately.

5.2 Power States

The 82580 supports the D0 and D3 architectural power states as described earlier. Internally, The 82580 supports the following power states:

- D0u (D0 un-initialized) - an architectural sub-state of D0
- D0a (D0 active) - an architectural sub-state of D0
- D3 - architecture state D3hot
- Dr - internal state that contains the architecture D3cold state. Dr state is entered when PE_RST_N is asserted or a PCIe in-band reset is received

Figure 5-1 shows the power states and transitions between them.

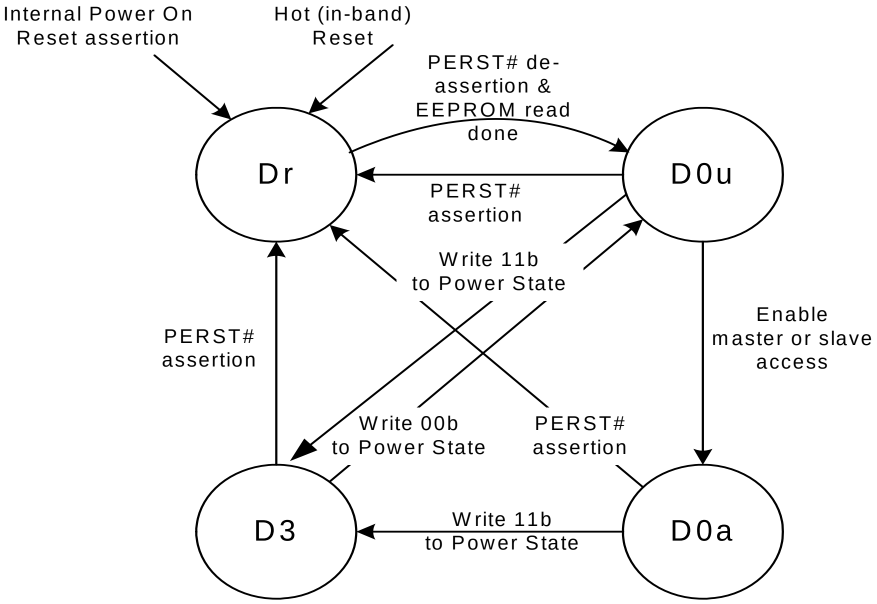


Figure 5-1. Power Management State Diagram

5.2.1 D0 Uninitialized State (D0u)

The D0u state is an architectural low-power state.

When entering D0u, the 82580:

- Asserts a reset to the PHY while the EEPROM is being read
- Disables wake up. However, if the *APM Mode* bit in the EEPROM's *Initialization Control Word 2* is set, then APM wake up is enabled.

5.2.1.1 Entry into D0u state

D0u is reached from either the Dr state (on de-assertion of *PE_RST_N*) or the D3hot state (by configuration software writing a value of 00b to the *Power State* field of the PCI PM registers).

5.2.1.2 Exit from D0u state

De-asserting *PE_RST_N* means that the entire state of the 82580 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 82580.

On a transition from D3 to D0u state, the 82580 PCI configuration space is not reset. However, the 82580 requires that software perform a full re-initialization of the function including its PCI configuration space.

5.2.2 D0active State

Once memory space is enabled, the 82580 enters the D0 active state. It can transmit and receive packets if properly configured by the software device driver. The PHY is enabled or re-enabled by the software device driver to operate/auto-negotiate to full line speed/power if not already operating at full capability. Any APM wake up previously active remains active. The software device driver can deactivate APM wake up 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.

5.2.2.1 Entry to D0a state

D0a is entered from the D0u state by writing a 1b to the *Memory Access Enable* or the *I/O Access Enable* bit of the PCI Command register. The DMA, MAC, and PHY of the appropriate LAN function are also enabled.

5.2.3 D3 State (PCI-PM D3hot)

The 82580 transitions to D3 when the system writes a 11b to the Power State field of the *Power Management Control/Status Register (PMCSR)*. Any wake-up filter settings that were enabled before entering this state are maintained. Upon completion or during the transition to D3 state, the 82580 clears the *Memory Access Enable* and *I/O Access Enable* bits of the PCI Command register, which disables memory access decode. While in D3, the 82580 does not generate master cycles.

Configuration and message requests are the only TLPs accepted by a function in the D3hot state. All other received requests must be handled as unsupported requests, and all received completions are handled as unexpected completions. If an error caused by a received TLP (such as an unsupported request) is detected while in D3hot, and reporting is enabled, the link must be returned to L0 if it is not already in L0 and an error message must be sent. See section 5.3.1.4.1 in The PCIe Base Specification

5.2.3.1 Entry to D3 State

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

Prior to transition from D0 to the D3 state, the software device driver disables scheduling of further tasks to the 82580; it masks all interrupts and does not write to the Transmit Descriptor Tail (*TDT*) register or to the Receive Descriptor Tail (*RDT*) register and operates the master disable algorithm as defined in Section 5.2.3.3.

If wake up capability is needed, system should enable wake capability by setting to 1b the *PME_En* bit in the *Power Management Control / Status Register (PMCSR)*. After Wake capability has been enabled Software device driver should set up the appropriate wake up registers prior to the D3 transition.

Note: If operation during $D3_{cold}$ is required, even when Wake capability is not required (e.g. for manageability operation), system should also set the *Auxiliary (AUX) Power PM Enable* bit in the PCIe *Device Control register*.

As a response to being programmed into D3 state, the 82580 transitions its PCIe link into the L1 link state. As part of the transition into L1 state, the 82580 suspends scheduling of new TLPs and waits for the completion of all previous TLPs it has sent. The 82580 clears the *Memory Access Enable* and *I/O Access Enable* bits of the PCI Command register, which disables memory access decode. Any receive packets that have not been transferred into system memory are kept in the 82580 (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 order to reduce power consumption, if the link is still needed for manageability or wake-up functionality, the PHY auto-negotiates to a lower link speed on D3 entry (See Section 3.5.7.5.4).

5.2.3.2 Exit from D3 State

A D3 state is followed by either a D0u state (in preparation for a D0a state) or by a transition to Dr state (PCI-PM D3cold state). To transition back to D0u, the system writes a 00b to the *Power State* field of the *Power Management Control/Status* Register (*PMCSR*). Transition to Dr state is through *PE_RST_N* assertion.

the 82580 always sets the *No_Soft_Reset* bit in the *PCIe Power Management Control / Status* Register (*PMCSR*) to 0b to indicate that The 82580 performs an internal reset on transition from D3hot to D0. Configuration context is lost when performing the soft reset. After transition from the D3hot to the D0 state, full re-initialization sequence is needed to return The 82580 to D0 Initialized.

5.2.3.3 Master Disable Via CTRL Register

System software can disable master accesses on the PCIe link by either clearing the PCI *Bus Master* bit or by bringing the function into a D3 state. From that time on, the 82580 must not issue master accesses for this function. Due to the full-duplex nature of PCIe, and the pipelined design in the 82580, 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 82580 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 82580 blocks new master requests by this function. The 82580 then proceeds to issue any pending requests by this function. This bit is cleared on master reset (LAN_PWR_GOOD, PCIe reset and software reset) to enable master accesses.
- *GIO Master Enable Status* bit in the Device Status (*STATUS*) register - Cleared by the 82580 when the *GIO Master Disable* bit is set and no master requests are pending by the relevant function and is set otherwise. Indicates that no master requests are issued by this function as long as the *GIO Master Disable* bit is set. The following activities must end before the 82580 clears the *GIO Master Enable Status* bit:
    - Master requests by the transmit and receive engines (for both data and MSI/MSIx interrupts).
    - All pending completions to the 82580 are received.

In the event of a PCIe Master disable (Configuration *Command register.BME* set to 0) on a certain function or LAN port or if the function is moved into D3 state during a DMA access, the 82580 generates an internal reset to the function and stops all port DMA accesses and interrupts related to the function. Following move to normal operating mode software driver should re-initialize the receive and transmit queues of the relevant port.

Notes: The software device driver sets the *GIO Master Disable* bit when notified of a pending master disable (or D3 entry). The 82580 then blocks new requests and proceeds to issue any pending requests by this function. The software device driver then polls the *GIO Master Enable Status* bit. Once the bit is cleared, it is guaranteed that no requests are pending from this function. The software device driver might time out if the *GIO Master Enable Status* bit is not cleared within a given time.

The *GIO Master Disable* bit must be cleared to enable a master request to the PCIe link. This can be done either through reset or by the software device driver.

5.2.4 Dr State (D3cold)

Transition to Dr state is initiated on several occasions:

- On system power up - Dr state begins with the assertion of the internal power detection circuit and ends with de-assertion of *PE_RST_N*.
- On transition from a D0a state - During operation the system might assert *PE_RST_N* at any time. In an ACPI system, a system transition to the G2/S5 state causes a transition from D0a to Dr state.
- On transition from a D3 state - The system transitions the 82580 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 82580 might enter one of several modes with different levels of functionality and power consumption. The lower-power modes are achieved when the 82580 is not required to maintain any functionality (see Section 5.2.4.1).

Note: If the 82580 is configured to provide a 50 MHz NC-SI clock (via the *NC-SI Output Clock* EEPROM bit), then the NC-SI clock must be provided in Dr state as well.

5.2.4.1 Dr Disable Mode

The 82580 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 82580 (all PCI functions) is in Dr state
- APM WOL is inactive for all LAN functions
- Pass-through manageability is disabled
- ACPI PME is disabled for all PCI functions
- The 82580 *Power Down Enable* EEPROM bit (word 0x1E, bit 15) is set (default hardware value is disabled).
- The *PHY Power Down Enable* EEPROM bit is set (word 0xF, bit 6).

Entering Dr disable mode is usually done by asserting PCIe *PE_RST_N*. It might also be possible to enter Dr disable mode by reading the 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 82580 enters Dr state on power-up, the EEPROM is read. If the EEPROM contents determine that the conditions to enter Dr disable mode are met, the 82580 then enters this mode (assuming that PCIe *PE_RST_N* is still asserted).

The 82580 exits Dr disable mode when Dr state is exited (See Figure 5-1 for conditions to exit Dr state).

5.2.4.2 Entry to Dr State

Dr entry on platform power-up begins with the assertion of the internal power detection circuit. The EEPROM is read and determines 82580 configuration. If the *APM Enable* bit in the EEPROM's *Initialization Control Word 3* is set, then APM wake up is enabled. PHY and MAC states are redetermined by the state of manageability and APM wake. To reduce power consumption, if manageability or APM wake is enabled, the PHY auto-negotiates to a lower link speed on Dr entry (See Section 3.5.7.5.4). The PCIe link is not enabled in Dr state following system power up (since *PE_RST_N* is asserted).

Entering Dr state from D0a state is done by asserting *PE_RST_N*. An ACPI transition to the G2/S5 state is reflected in the 82580 transition from D0a to Dr state. The transition can be orderly (such as user selecting the shut down option), in which case the software device driver might have a chance to intervene. Or, it might be an emergency transition (such as power button override), in which case, the software device driver is not notified.

To reduce power consumption, if any of manageability, APM wake or PCI-PM PME[1] is enabled, the PHY auto-negotiates to a lower link speed on D0a to Dr transition (see Section 3.5.7.5.4).

Transition from D3 (hot) state to Dr state is done by asserting *PE_RST_N*. Prior to that, the system initiates a transition of the PCIe link from L1 state to either the L2 or L3 state (assuming all functions were already in D3 state). The link enters L2 state if PCI-PM PME is enabled.

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

5.2.4.3 Auxiliary Power Usage

The EEPROM *D3COLD_WAKEUP_ADVEN* bit and the *AUX_PWR strapping pin determine when D3cold PME is supported:*

- D3COLD_WAKEUP_ADVEN denotes that PME wake should be supported
- AUX_PWR strapping pin indicates that auxiliary power is provided

D3cold PME is supported as follows:

- If the *D3COLD_WAKEUP_ADVEN is set to '1' and* the AUX_PWR strapping is set to '1', then *D3cold PME is supported*
- Else *D3cold PME is not supported*

The amount of power required for the function (including the entire NIC) is advertised in the *Power Management Data* register, which is loaded from the 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).

5.2.5 Link Disconnect

In any of D0u, D0a, D3, or Dr power states, the 82580 enters a link-disconnect state if it detects a link-disconnect condition on the Ethernet link. Note that the link-disconnect state in the internal PHY is invisible to software (other than the *PHPM.Link Energy Detect* bit state). In particular, while in D0 state, software might be able to access any of the 82580 registers as in a link-connect state.

5.2.6 Device Power-Down State

The 82580 enters a global power-down state if all of the following conditions are met:

- The 82580 *Power Down Enable* EEPROM bit (word 0x1E bit 15) was set (default hardware value is disabled).
- The 82580 is in Dr state.
- The link connections of all ports (PHY or SerDes) are in power down mode.

The 82580 also enters a power-down state when the DEV_OFF_N pin is asserted and the relevant EEPROM bits were configured as previously described (see Section 4.5 for more details on DEV_OFF_N functionality).

5.3 Power Limits by Certain Form Factors

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

Table 5-1. Power Limits by Form-Factor

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

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

The 82580 exceeds the allocated auxiliary power in some configurations (such as all ports running at 1000 Mb/s speed). The 82580 must therefore be configured to meet the previously mentionedcertain requirements. To do so, the 82580 implements three EEPROM bits to disable operation in certain cases:

1. The *PHPM.Disable_1000* PHY register bit disables 1000 Mb/s operation under all conditions.
2. The *PHPM.Disable 1000 in non-D0a* PHY CSR bit disables 1000 Mb/s operation in non-D0a states[1]. If *PHPM.Disable 1000 in non-D0a* is set, and the 82580 is at 1000 Mb/s speed on entry to a non-D0a state, then the 82580 removes advertisement for 1000 Mb/s and auto-negotiates.
3. The *PHPM.Disable 100 in non-D0a* PHY CSR bit disables 1000 Mb/s and 100 Mb/s operation in non-D0a states. If *PHPM.Disable 100 in non-D0a* is set, and the 82580 is at 1000 Mb/s or 100 Mb/s speeds on entry to a non-D0a state, then the 82580 removes advertisement for 1000 Mb/s and 100 Mb/s and auto-negotiates.

Note that the 82580 restarts link auto-negotiation each time it transitions from a state where 1000 Mb/s or 100 Mb/s speed is enabled to a state where 1000 Mb/s or 100 Mb/s speed is disabled, or vice versa. For example, if *PHPM.Disable 1000 in non-D0a* is set but *PHPM.Disable_1000* is cleared, the 82580 restarts link auto-negotiation on transition from D0 state to D3 or Dr states.

5.4 Interconnects Power Management

This section describes the power reduction techniques employed by the 82580 main interconnects.

5.4.1 PCIe Link Power Management

The PCIe link state follows the power management state of the 82580. Since the 82580 incorporates multiple PCI functions, its power management state is defined as the power management state of the most awake function (see Figure 5-2):

- If any function is in D0 state (either D0a or D0u), the PCIe link assumes the 82580 is in D0 state. Else,
- If the functions are in D3 state, the PCIe link assumes the 82580 is in D3 state. Else,
- The 82580 is in Dr state (PE_RST_N is asserted to all functions).

The 82580 supports all PCIe power management link states:

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

[1] The restriction is defined for all non-D0a states to have compatible behavior with previous products.

- The L1 state is also used in D0a and D0u states when idle conditions apply for a longer period of time. The L1 state is also used in the D3 state.
- The L2 state is used in the Dr state following a transition from a D3 state if *PCI-PM PME* is enabled.
- The L3 state is used in the Dr state following power up, on transition from D0a, and if *PME* is not enabled in other Dr transitions.

The 82580 support for active state link power management is reported via the PCIe *Active State Link PM Support* register and is loaded from the EEPROM.

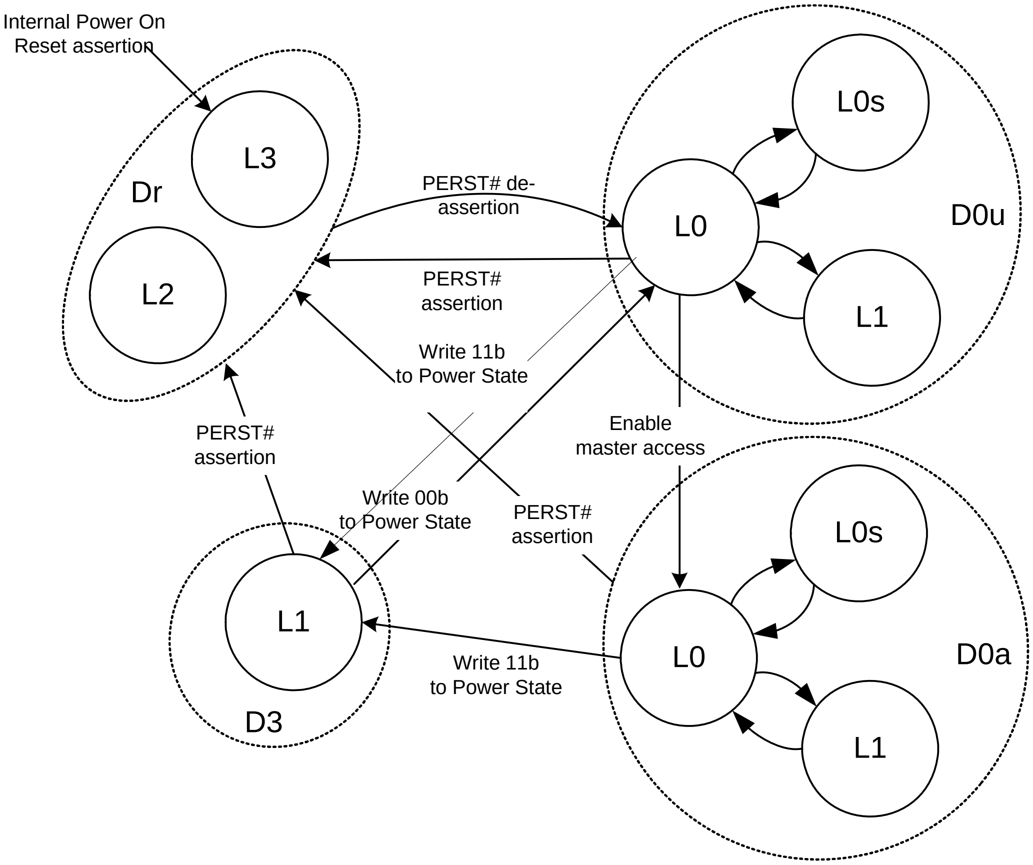


Figure 5-2. Link Power Management State Diagram

While in L0 state, the 82580 transitions the transmit lane(s) into L0s state once the idle conditions are met for a period of time as follows:

L0s configuration fields are:

- L0s enable - The default value of the *Active State Link PM Control* field in the PCIe *Link Control* Register is set to 00b (both L0s and L1 disabled). System software may later write a different value into the Link Control Register. The default value is loaded on any reset of the PCI configuration registers.

- L0s exit latency (as published in the L0s Exit Latency field of the Link Capabilities Register) is loaded from EEPROM. Separate values are loaded when the 82580 shares the same reference PCIe clock with its partner across the link, and when the 82580 uses a different reference clock than its partner across the link. The 82580 reports whether it uses the slot clock configuration through the PCIe Slot Clock Configuration bit loaded from the *Slot_Clock_Cfg* bit in the *PCIe Init Configuration 3* EEPROM Word.
- L0s Acceptable Latency (as published in the Endpoint L0s Acceptable Latency field of the Device Capabilities Register) is loaded from EEPROM.

If link is in L0s state, the 82580 transitions the link into L1 state once the transmit lanes or both directions of the link have been in L0s state for a period of time defined in the *Latency_To_Enter_L1* field in the PCIe PHY Auto Configuration EEPROM section.

The following EEPROM fields control L1 behavior:

- *Act_Stat_PM_Sup* - Indicates support for ASPM L1 in the PCIe configuration space (loaded into the Active State Link PM Support field)
- *L1_Act_Ext_Latency* - Defines L1 active exit latency
- *L1_Act_Acc_Latency* - Defines L1 active acceptable exit latency
- *Latency_To_Enter_L1* - Defines the period (in the L0s state) before the transition into L1 state

5.4.2 NC-SI Clock Control

The 82580 can be configured to provide a 50 MHz output clock to its NC-SI interface and other platform devices. When enabled, the NC-SI clock is provided in all power states without exception.

5.4.3 Internal PHY Power-Management

The PHY power management features are described in Section 3.5.7.5.

5.5 Timing of Power-State Transitions

The following sections give detailed timing for the state transitions. In the diagrams the dotted connecting lines represent the 82580 requirements, while the solid connecting lines represent the 82580 guarantees.

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

5.5.1 Power Up (Off to Dup to D0u to D0a

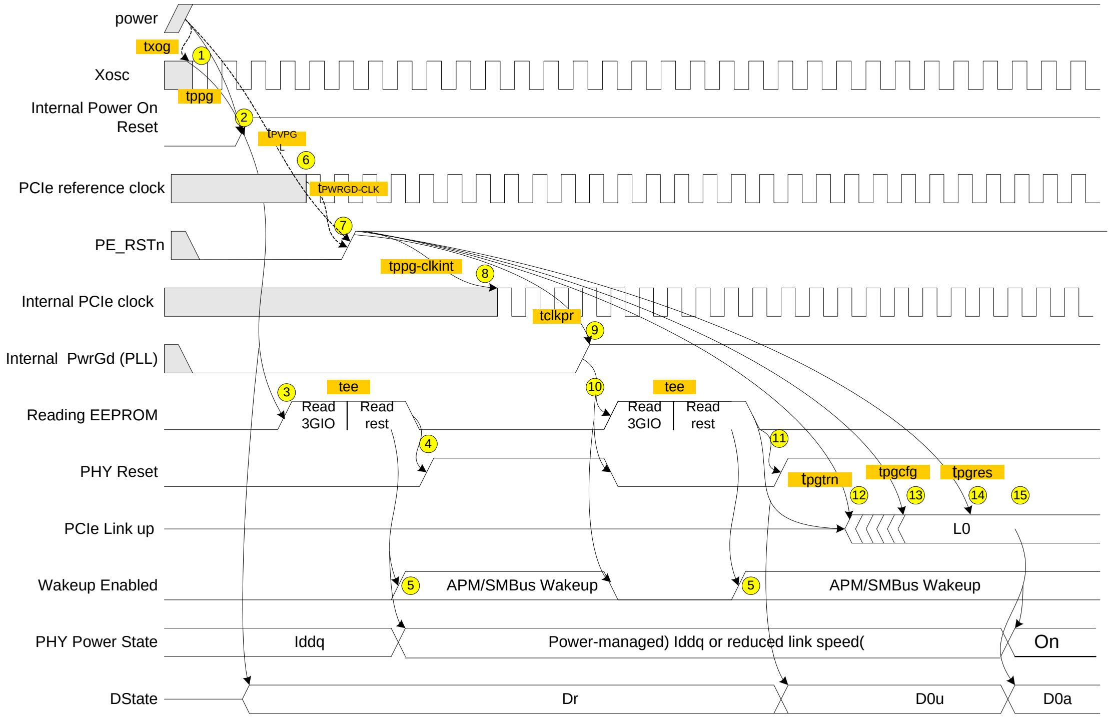


Figure 5-3. Power Up (Off to Dup to D0u to D0a)

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

Note	Description
1	Xosc is stable t_{xog} after power is stable.
2	LAN_PWR_GOOD is asserted after all power supplies are good and t_{ppg} after Xosc is stable.
3	An EEPROM read starts on the rising edge of LAN_PWR_GOOD.
4	After reading the EEPROM, PHY reset is de-asserted.
5	APM wake-up mode can be enabled based on what is read from the EEPROM.
6	The PCIe reference clock is valid $t_{PE\_RST-CLK}$ before de-asserting PE_RST_N (according to PCIe specification).
7	PE_RST_N is de-asserted t_{PVPGL} after power is stable (according to PCIe specification).
8	The internal PCIe clock is valid and stable $t_{ppg\text{-}clkint}$ from PE_RST_N de-assertion.
9	The PCIe internal PWRGD signal is asserted t_{clkpr} after the external PE_RST_N signal.
10	Asserting internal PCIe PWRGD causes the EEPROM to be re-read, asserts PHY reset, and disables wake up.
11	After reading the EEPROM, PHY reset is de-asserted.
12	Link training starts after t_{pgtrn} from PE_RST_N de-assertion.

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

Note	Description
13	A first PCIe configuration access might arrive after t_{pgcfg} from PE_RST_N de-assertion.
14	A first PCI configuration response can be sent after t_{pgres} from PE_RST_N de-assertion.
15	Writing a 1b to the *Memory Access Enable* bit in the PCI Command Register transitions the 82580 from D0u to D0. state.

5.5.2 Transition from D0a to D3 and Back Without PE_RST_N

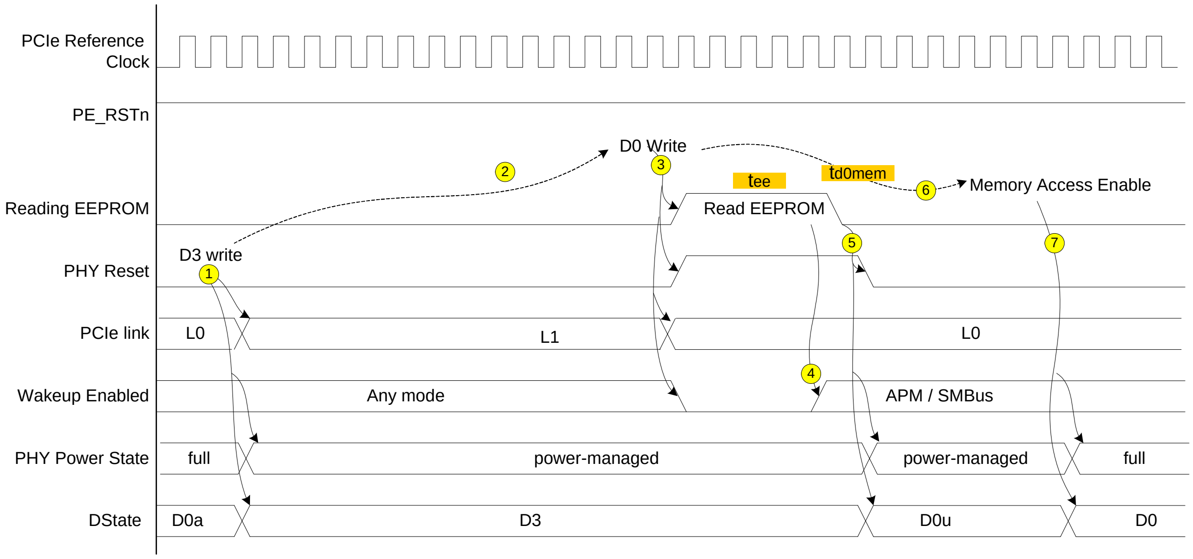


Figure 5-4. Transition from D0a to D3 and Back Without PE_RST_N

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

Note	Description
1	Writing 11b to the *Power State* field of the Power Management Control/Status Register (PMCSR) transitions the 82580 to D3.
2	The system can keep the 82580 in D3 state for an arbitrary amount of time.
3	To exit D3 state, the system writes 00b to the *Power State* field of the PMCSR.
4	APM wake-up or SMBus mode might be enabled based on what is read in the EEPROM.
5	After reading the EEPROM, reset to the PHY is de-asserted. The PHY operates at reduced-speed if APM wake up or SMBus is enabled, else powered-down.
6	The system can delay an arbitrary time before enabling memory access.
7	Writing a 1b to the *Memory Access Enable* bit or to the *I/O Access Enable* bit in the PCI Command Register transitions the 82580 from D0u to D0 state and returns the PHY to full-power/speed operation.

5.5.3 Transition From D0a to D3 and Back With PE_RST_N

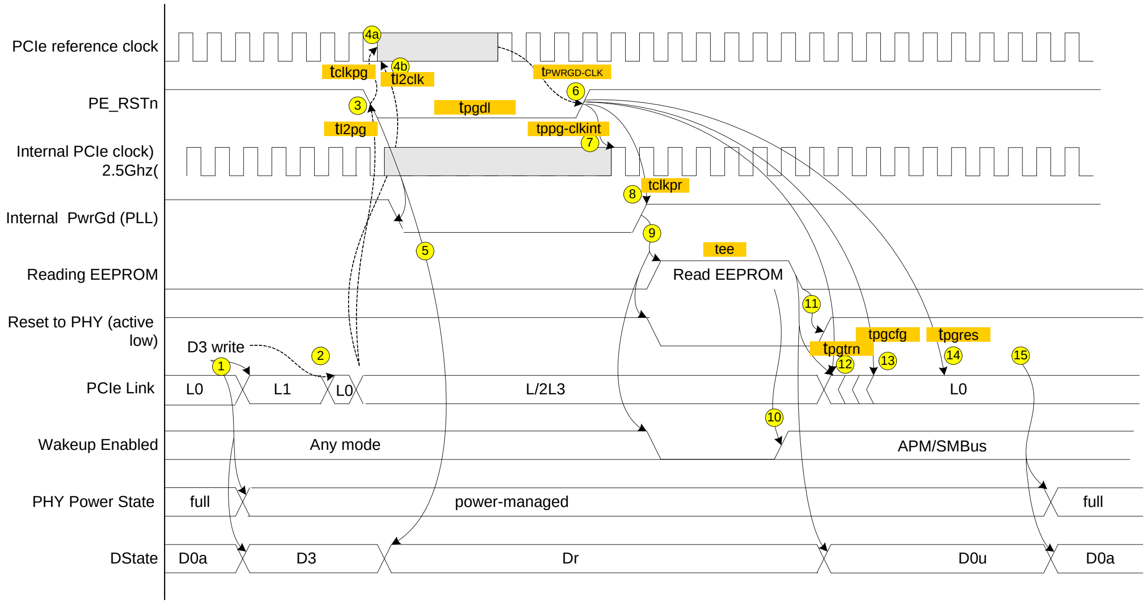


Figure 5-5. Transition From D0a to D3 and Back With PE_RST_N

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

Note	Description
1	Writing 11b to the *Power State* field of the PMCSR transitions the 82580 to D3. PCIe link transitions to L1 state.
2	The system can delay an arbitrary amount of time between setting D3 mode and transition of 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 82580 transitions to Dr state.
6	The system starts the PCIe reference clock $t_{PE\_RST-CLK}$ before de-assertion PE_RST_N.
7	The internal PCIe clock is valid and stable $t_{ppg-clkint}$ from PE_RST_N de-assertion.
8	The PCIe internal PWRGD signal is asserted t_{clkpr} after the external PE_RST_N signal.
9	Asserting internal PCIe PWRGD causes the EEPROM to be re-read, asserts PHY reset, and disables wake up.
10	APM wake-up mode might be enabled based on what is read from the EEPROM.
11	After reading the EEPROM, PHY reset is de-asserted.
12	Link training starts after t_{pgtrn} from PE_RST_N de-assertion.
13	A first PCIe configuration access might arrive after t_{pgcfg} from PE_RST_N de-assertion.
14	A first PCI configuration response can be sent after t_{pgres} from PE_RST_N de-assertion.
15	Writing a 1b to the *Memory Access Enable* bit in the PCI Command Register transitions the 82580 from D0u to D0 state.

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

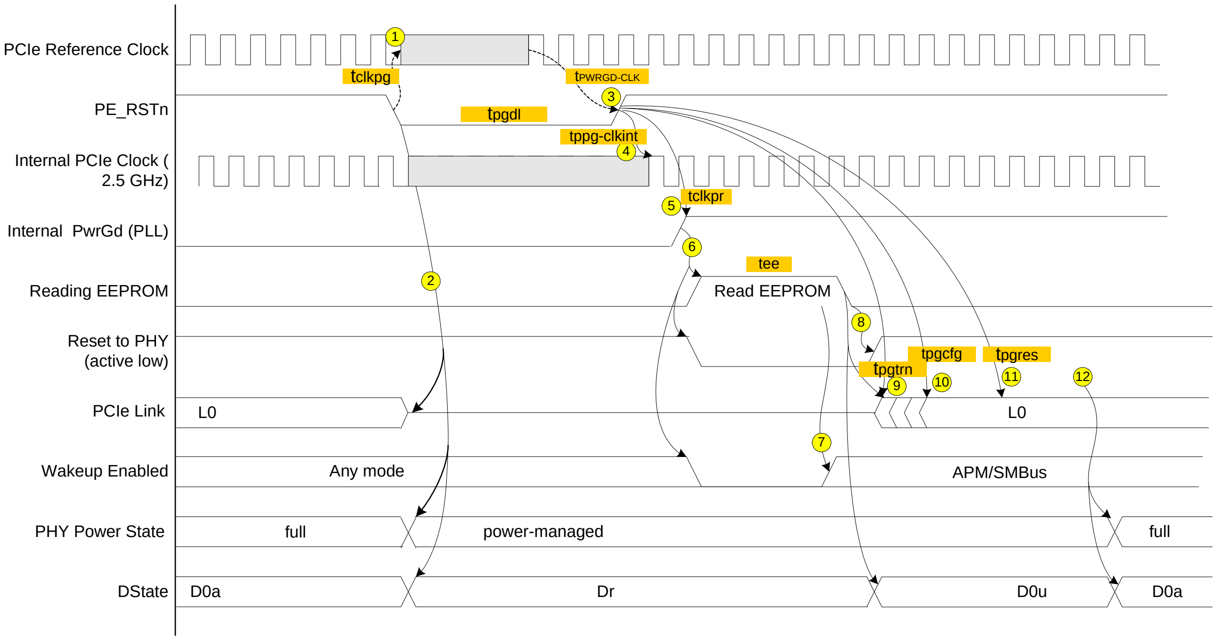


Figure 5-6. Transition From D0a to Dr and Back Without Transition to D3

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

Note	Description
1	The system must assert PE_RST_N before stopping the PCIe reference clock. It must also wait t_{l2clk} after link transition to L2/L3 before stopping the reference clock.
2	On assertion of PE_RST_N, the 82580 transitions to Dr state and the PCIe link transition to electrical idle.
3	The system starts the PCIe reference clock $t_{PE\_RST-CLK}$ before de-assertion PE_RST_N.
4	The internal PCIe clock is valid and stable $t_{ppg-clkint}$ from PE_RST_N de-assertion.
5	The PCIe internal PWRGD signal is asserted t_{clkpr} after the external PE_RST_N signal.
6	Asserting internal PCIe PWRGD causes the EEPROM to be re-read, asserts PHY reset, and disables wake up.
7	APM wake-up mode might be enabled based on what is read from the EEPROM.
8	After reading the EEPROM, PHY reset is de-asserted.
9	Link training starts after t_{pgtrn} from PE_RST_N de-assertion.
10	A first PCIe configuration access might arrive after t_{pgcfg} from PE_RST_N de-assertion.
11	A first PCI configuration response can be sent after t_{pgres} from PE_RST_N de-assertion.
12	Writing a 1b to the *Memory Access Enable* bit in the PCI Command Register transitions the 82580 from D0u to D0 state.

5.6 Wake Up

The 82580 supports two modes of wake-up management:

1. Advanced Power Management (APM) wake up
2. ACPI/PCIe defined wake up

The usual model is to activate one mode at a time but not both modes together. If both modes are activated, the 82580 might wake up the system in unexpected events. For example, if APM is enabled together with PCIe PME, a magic packet might wake up the system even if APMPME is disabled. Alternatively, if APM is enabled together with some PCIe filters, packets matching these filters might wake up the system even if PCIe PME is disabled.

5.6.1 Advanced Power Management Wake Up

Advanced Power Management Wake Up or APM Wakeup (also known as Wake on LAN) is a feature that existed in earlier 10/100 Mb/s NICs. This functionality was designed to receive a broadcast or unicast packet with an explicit data pattern, and then assert a subsequent signal to wake up the system. This was accomplished by using a special signal that ran across a cable to a defined connector on the motherboard. The NIC would assert the signal for approximately 50 ms to signal a wake up. The 82580 now uses (if configured) an in-band PM_PME message for this functionality.

On power up, the 82580 reads the *APM Enable* bits from the EEPROM *Initialization Control Word 3* into the *APM Enable (APME)* bits of the *Wakeup Control (WUC)* register. These bits control enabling of APM wake up.

When APM wake up is enabled, the 82580 checks all incoming packets for Magic Packets. See Section 5.6.3.1.4 for a definition of Magic Packets.

Once the 82580 receives a matching magic packet, and if the *Assert PME On APM Wakeup* (*WUC.APMPME*) bit is set in the Wake Up Control (*WUC*) register, it:

- Sets the *PME_Status* bit in the *PMCSR* register and issues a PM_PME message (in some cases, this might require asserting the PE_WAKE_N signal first to resume power and clock to the PCIe interface).
- Stores the first 128 bytes of the packet in the Wake Up Packet Memory (*WUPM*) register.
- Sets the *Magic Packet Received* bit in the Wake Up Status (*WUS*) register.
- Sets the packet length in the Wake Up Packet Length (*WUPL*) register.

The 82580 maintains the first Magic Packet received in the Wake Up Packet Memory (*WUPM*) register until the software device driver writes a 1b to the *Magic Packet Received MAG* bit in the Wake Up Status (*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 software explicitly writes a 0b to the *APM Wake Up* (APME) bit of the *WUC* register.

Note: When *WUC.APMPME* is set PE_WAKE_N is asserted and a PM_PME message is issued even if *PMCSR.PME_En* is cleared. To enable disabling of system Wake-up when *PMCSR.PME_En* is cleared, Software driver should clear the *WUC.APMPME* bit after power-up or PCIe reset.

5.6.2 PCIe Power Management Wake Up

The 82580 supports PCIe power management based wake ups. It can generate system wake-up events from three sources:

- Reception of a Magic Packet.
- Reception of a network wakeup packet.
- Detection of a link change of state.

Activating PCIe power management wake up requires the following:

- 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*) and the Flexible Host Filter Table (*FHFT*). It can also set the *Link Status Change Wake Up Enable* (*LNKC*) bit in the Wake Up Filter Control (*WUFC*) register to cause wake up when the link changes state.
- The operating system (at configuration time) writes a 1b to the *PME_En* bit of the Power Management Control/Status (*PMCSR.8*) register.

Normally, after enabling wake up, the operating system writes 11b to the lower two bits of the *PMCSR* register to place the 82580 into low-power mode.

Once wake up is enabled, the 82580 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 wakeup filters, the 82580:

- Sets the *PME_Status* bit in the *PMCSR*.
- Asserts PE_WAKE_N (if the *PME_En* bit in the PMCSR is set).
- Stores the first 128 bytes of the packet in the *Wakeup Packet Memory (WUPM)* register.
- Sets one or more of the received bits in the *Wake Up Status (WUS)* register. Note that the 82580 sets more than one bit if a packet matches more than one filter.
- Sets the packet length in the *Wake Up Packet Length (WUPL)* register.

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

The 82580 supports the following change described in the PCIe Base Specification, Rev. 1.1RD (section 5.3.3.4) - On receiving a PME_Turn_Off message, the 82580 must block the transmission of PM_PME messages and transmit a PME_TO_Ack message upstream. The 82580 is permitted to send a PM_PME message after the Link is returned to an L0 state through LDn.

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 wake-up packet, the 82580 ignores any subsequent wake-up packets until the software device driver clears all of the received bits in the Wake Up Status (WUS) register. It also ignores link change events until the software device driver clears the *Link Status Changed (LNKC)* bit in the *Wake Up Status (WUS)* register.

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

5.6.3 Wake-Up Packets

The 82580 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 Wake Up Filter Control (*WUFC*) register is set to 1b.

5.6.3.1 Pre-Defined Filters

The following packets are supported by the 82580'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 *Wakeup Filter Control (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.

Note: Both VLAN frames and LLC/SNAP can increase the given offsets if they are present.

5.6.3.1.1 Directed Exact Packet

The 82580 generates a wake-up event after receiving any packet whose destination address matches one of the 24 valid programmed receive addresses, if the *Directed Exact Wake Up Enable* bit is set in the Wake Up Filter Control (*WUFC.EX*) register.

5.6.3.1.2 Directed Multicast Packet

For multicast packets, the upper bits of the incoming packet's destination address index a bit vector, the Multicast Table Array (*MTA*) that indicates whether to accept the packet. If the *Directed Multicast Wake Up Enable* bit set in the Wake Up Filter Control (*WUFC.MC*) register and the indexed bit in the vector is one, then the 82580 generates a wake-up event. The exact bits used in the comparison are programmed by software in the *Multicast Offset* field of the Receive Control (*RCTL.MO*) register.

5.6.3.1.3 Broadcast

If the Broadcast Wake Up Enable bit in the Wake Up Filter Control (WUFC.BC) register is set, the 82580 generates a wake-up event when it receives a broadcast packet.

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address	FF*6	Compare	

5.6.3.1.4 Magic Packet

Magic packets are defined in:

http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/20213.pdf as:

Once the LAN controller has been put into the Magic Packet mode, it scans all incoming frames addressed to the node for a specific data sequence.This sequence indicates to the controller that this is a Magic Packet frame. A Magic Packet frame must also meet the basic requirements for the LAN technology chosen, such as SOURCE ADDRESS, DESTINATION ADDRESS (which may be the receiving station's IEEE address or a MULTICAST address which includes the BROADCAST address), and CRC. The specific data sequence consists of 16 repetitions of the IEEE address of this node, with no breaks or interruptions. This sequence can be located anywhere within the packet, but must be preceded by a synchronization stream. The synchronization stream allows the scanning state machine to be much simpler. The synchronization stream is defined as 6 bytes of 0xFF. The device will also accept a BROADCAST frame, as long as the 16 repetitions of the IEEE address match the address of the machine to be awakened."

The 82580 expects the destination address to either:

- Be the broadcast address (FF.FF.FF.FF.FF.FF)
- Match the value in Receive Address 0 (*RAH0*, *RAL0*) register. This is initially loaded from the EEPROM but can be changed by the software device driver.
- Match any other address filtering (*RAH*[n], *RAL*[n]) enabled by the software device driver.

The 82580 searches for the contents of Receive Address 0 (*RAH0*, *RAL0*) register as the embedded IEEE address. It considers any non-0xFF byte after a series of at least 6 0xFFs to be the start of the IEEE address for comparison purposes. For example, it catches the case of 7 0xFFs followed by the IEEE address). As soon as one of the first 96 bytes after a string of 0xFFs don't match, it continues to search for another set of at least 6 0xFFs followed by the 16 copies of the IEEE address later in the packet. Note that this definition precludes the first byte of the destination address from being FF.

A Magic Packet's destination address must match the address filtering enabled in the configuration registers with the exception that broadcast packets are considered to match even if the *Broadcast Accept* bit of the *Receive Control (RCTL.BAM)* register is 0b. If APM wake up (wake up by a Magic Packet) is enabled in the EEPROM, the 82580 starts up with the Receive Address 0 (*RAH0*, *RAL0*) register loaded from the EEPROM. This enables the 82580 to accept packets with the matching IEEE address before the software device driver loads.

Table 5-6. Magic Packet Structure

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC header – processed by main address filter.
6	6	Source Address		Skip	
12	S=(0/4)	Possible VLAN Tag		Skip	
12 + S	D=(0/8)	Possible Length + LLC/SNAP Header		Skip	
12 + S + D	2	Type		Skip	
Any	6	Synchronizing Stream	FF*6+	Compare	
any+6	96	16 copies of Node Address	A*16	Compare	Compared to Receive Address 0 (RAH0, RAL0) register.

5.6.3.1.5 ARP/IPv4 Request Packet

The 82580 supports receiving ARP request packets for wake up if the *ARP* bit is set in the Wake Up Filter Control (*WUFC*) register. Four IPv4 addresses are supported, which are programmed in the IPv4 Address Table (*IP4AT*). A successfully matched packet must contain a broadcast MAC address, a protocol type of 0x0806, an ARP op-code of 0x01, and one of the four programmed IPv4 addresses. The 82580 also handles ARP request packets that have VLAN tagging on both Ethernet II and Ethernet SNAP types.

Table 5-7. ARP Packet Structure and Processing

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC header – processed by main address filter.
6	6	Source Address		Skip	
12	S=(0/4)	Possible VLAN Tag		Compare	Processed by main address filter.
12 + S	D=(0/8)	Possible Length + LLC/SNAP Header		Skip	
12 + S + D	2	Type	0x0806	Compare	ARP
14 + S + D	2	HW Type	0x0001	Compare	
16 + S + D	2	Protocol Type	0x0800	Compare	
18 + S + D	1	Hardware Size	0x06	Compare	
19 + S + D	1	Protocol Address Length	0x04	Compare	
20 + S + D	2	Operation	0x0001	Compare	
22 + S + D	6	Sender HW Address	-	Ignore	
28 + S + D	4	Sender IP Address	-	Ignore	
32 + S + D	6	Target HW Address	-	Ignore	
38 + S + D	4	Target IP Address	IP4AT	Compare	Compare if the *Directed ARP* bit is set to 1b. May match any of four values in *IP4AT*.

5.6.3.1.6 Directed Ipv4 Packet

The 82580 supports receiving directed IPv4 packets for wake up if the *IPV4* bit is set in the Wake Up Filter Control (*WUFC*) register. Four IPv4 addresses are supported, which are programmed in the IPv4 Address Table (*IP4AT*). A successfully matched packet must contain the station's MAC address, a protocol type of 0x0800, and one of the four programmed IPv4 addresses. The 82580 also handles directed IPv4 packets that have VLAN tagging on both Ethernet II and Ethernet SNAP types.

Table 5-8. IPv4 Packet Structure and Processing

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC header – processed by main address filter.
6	6	Source Address		Skip	
12	S=(0/4)	Possible VLAN Tag		Compare	Processed by main address filter.
12 + S	D=(0/8)	Possible Length + LLC/SNAP Header		Skip	
12 + S + D	2	Type	0x0800	Compare	IP
14 + S + D	1	Version/ HDR length	0x4X	Compare	Check IPv4
15 + S + D	1	Type of Service	-	Ignore	
16 + S + D	2	Packet Length	-	Ignore	
18 + S + D	2	Identification	-	Ignore	
20 + S + D	2	Fragment Info	-	Ignore	

Table 5-8. IPv4 Packet Structure and Processing (Continued)

Offset	# of bytes	Field	Value	Action	Comment
22 + S + D	1	Time to live	-	Ignore	
23 + S + D	1	Protocol	-	Ignore	
24 + S + D	2	Header Checksum	-	Ignore	
26 + S + D	4	Source IP Address	-	Ignore	
30 + S + D	4	Destination IP Address	IP4AT	Compare	May match any of four values in *IP4AT*.

5.6.3.1.7 Directed IPv6 Packet

The 82580 supports receiving directed IPv6 packets for wake up if the *IPV6* bit is set in the *Wake Up Filter Control (WUFC)* register. One IPv6 address is supported and is programmed in the *IPv6 Address Table (IP6AT)*. A successfully matched packet must contain the station's MAC address, a protocol type of 0x86DD, and the programmed IPv6 address. In addition, the *IPAV.V60* bit should be set. The 82580 also handles directed IPv6 packets that have VLAN tagging on both Ethernet II and Ethernet SNAP types.

Table 5-9. IPv6 Packet Structure and Processing

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC header – processed by main address filter.
6	6	Source Address		Skip	
12	S=(0/4)	Possible VLAN Tag		Compare	Processed by main address filter.
12+ S	D=(0/8)	Possible Length + LLC/SNAP Header		Skip	
12 + S + D	2	Type	0x86DD	Compare	IP
14 + S + D	1	Version/ Priority	0x6X	Compare	Check IPv6
15 + S + D	3	Flow Label	-	Ignore	
18 + S + D	2	Payload Length	-	Ignore	
20 + S + D	1	Next Header	-	Ignore	
21 + S + D	1	Hop Limit	-	Ignore	
22 + S + D	16	Source IP Address	-	Ignore	
38 + S + D	16	Destination IP Address	IP6AT	Compare	Match value in *IP6AT*.

5.6.3.2 Flexible Filters

The 82580 supports a total of 8 flexible filters. Each filter can be configured to recognize any arbitrary pattern within the first 128 bytes of the packet. To configure the flexible filters, software programs the mask values (required values and the minimum packet length), into the Flexible Host Filter Table (FHFT and FHFT_EXT). These 8 flexible filters contain separate values for each filter. Software must also enable the filters in the Wake Up Filter Control (*WUFC*) register, and enable the overall wake up functionality. The overall wake up functionality must be enabled by setting *PME_En* in the PMCSR or the Wake Up Control (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 value programmed in the Flexible Host Filter Table (*FHFT* or *FHFT_EXT*), then the filter fails that packet. If the filter reaches the required length without failing the packet, it passes the packet and generates a wake-up event. It ignores any mask bits set to one beyond the required length.

Note: The flex filters are temporarily disabled when read from or written to by the host. Any packet received during a read or write operation is dropped. Filter operation resumes once the read or write access completes.

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

5.6.3.2.1 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 might include LLC/SNAP headers and VLAN tags. Since filtering this packet relies on the flexible filters, which use offsets specified by the operating system directly, the operating system must account for the extra offset LLC/SNAP headers and VLAN tags.

Table 5-10. IPX Diagnostic Responder Request Packet Structure and Processing

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	
6	6	Source Address		Skip	
12	S=(0/4)	Possible VLAN Tag		Skip	
12+ S	D=(0/8)	Possible Length + LLC/ SNAP Header		Skip	
12 + S + D	2	Type	0x8137	Compare	IPX
14 + S + D	16	Some IPX Stuff	-	Ignore	
30 + S + D	2	IPX Diagnostic Socket	0x0456	Compare	

5.6.3.2.2 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 is equal to the station's MAC address. It might include LLC/SNAP headers and VLAN tags. Since filtering this packet relies on the flexible filters, which use offsets specified by the operating system directly, the operating system must account for the extra offset LLC/SNAP headers and VLAN tags.

Table 5-11. IPX Packet Structure and Processing

Offset	# of bytes	Field	Value	Action	Comment
0	6	Destination Address		Compare	MAC header – processed by main address filter.
6	6	Source Address		Skip	
12	S=(0/4)	Possible VLAN Tag		Skip	
12+ S	D=(0/8)	Possible Length + LLC/SNAP Header		Skip	

Table 5-11. IPX Packet Structure and Processing

12 + S + D	2	Type	0x8137	Compare	IPX
14 + S + D	10	Some IPX Info	-	Ignore	
24 + S + D	6	IPX Node Address	Receive Address 0	Compare	Must match receive address 0.

5.6.3.2.3 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 neighborhood discovery packet.

5.6.3.2.4 Utilizing Flex Wake-Up Filters In Normal Operation

The 82580 enables utilizing the WoL Flex filters in normal operation, when in D0 power management state, for queuing decisions. Further information can be found in Section 7.1.1.6.

5.6.3.3 Wake Up Packet Storage

The 82580 saves the first 128 bytes of the wake-up packet in its internal buffer, which can be read through the *Wake Up Packet Memory (WUPM)* register after the system wakes up.

5.7 DMA Coalescing

The 82580 supports DMA Coalescing to enable synchronizing port activity and optimize power management of memory, CPU and RC internal circuitry.

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

1. *DMACR.DMACTHR* field to set the receive threshold that causes move out of DMA Coalescing operating mode. Receive watermark programmed should take into account latency tolerance reported (See Section 5.8) and L1 to L0 latency to avoid Receive Buffer overflow when DMA Coalescing is enabled.
2. *DMCTXTH.DMCTTHR* field to set transmit threshold that causes move out of DMA Coalescing operating mode. Transmit watermark programmed should take into account latency tolerance reported (See Section 5.8) and L1 to L0 latency to allow transmission of back to back packets when DMA Coalescing is enabled.
3. *DMACR.DMACWT* field defines a maximum time for a receive packet to be stored in the internal receive buffer before the 82580 moves packet to Host memory, even if the *DMACR.DMACTHR* watermark is not passed.
4. *DMCTLX.*TTLX timer field to define, the time between detection of DMA idle status to actual move into low power link state (L0s or L1). Value programmed in this register reduces amount of entries into low power PCIe link state when traffic rate is high.
5. *DMCRTRH.UTRESH* low rate threshold field to define the upper limit to data arrival rate, were DMA coalescing is not entered. This field avoids moving into DMA coalescing mode when traffic is sparse and effectiveness of DMA coalescing on system power saving is limited. The time interval were data rate is measured is defined by the Storm Control *SCCRL.INTERVAL* field that uses the time units defined in the *SCBI* register (See Section 7.8.2.8.3.2 for description of how to set the time interval for rate measurement).
6. *FCRTC.RTH_Coal* field that defines a flow control receive high watermark for sending flow control packets. The 82580 uses the *FCRTC.RTH_Coal* threshold when:
   - Flow control is enabled by setting the *CTRL.RFCE* bit.
   - The 82580 is in DMA Coalescing mode.

— Internal transmit buffer is empty.

7. *SRRCTL[n].DMACQ_Dis* bit to define high priority queues. When a received packet is forwarded to a queue with the *SRRCTL[n].DMACQ_Dis* bit set, the 82580 moves immediately out of Buffer Fill mode and executes a DMA operation to store the packet in host memory.
8. *DMACR.DMAC_EN* bit should be set to 1 to enable activation of DMA Coalescing operating mode.
9. *DMACR.DMAC_Lx* to 10b to define that L1 low power PCIe link state is entered in DMA Coalescing operation .

Note: The values of *DMACR.DMACTHR* and *FCRTC.RTH_Coal* should be set so that XOFF packet generation is avoided. In DMA Coalescing mode, when transmit buffer is empty, the XOFF flow control threshold (*FCRTC.RTH_Coal*) value can be increased by maximum jumbo frame size compared to normal operation, were high threshold is set by the *FCRTH0* register.

When entering DMA coalescing mode, the value written in the *FCRTH0* register is used to generate XOFF flow control frames until the internal transmit buffer is empty. Once the internal transmit buffer is empty the value written in the *FCRTC.RTH_Coal* field is used as a watermark for generation of XOFF frames.

5.7.1 Entering DMA Coalescing Operating Mode

Enabling DMA Coalescing operation by setting the *DMACR.DMAC_EN* bit to 1, enables alignment of bus master traffic and interrupts from all ports. Power saving is achieved since synchronizing PCIe accesses between ports increases the occurrence of idle intervals on the PCIe bus and also increases the duration of these idle intervals. Power Management Unit on platform can utilize these Idle intervals to reduce system power.

5.7.1.1 Conditions to Enter DMA Coalescing

The 82580 enters DMA Coalescing state, were access to the PCIe bus is delayed and received packets are buffered in internal receive memory until appropriate conditions occur, when all of the following conditions exist:

- Internal receive buffers are empty.
- There are no pending DMA operations.
- None of the conditions defined in Section 5.7.2 to move out of DMA Coalescing exist.

Before entering the DMA coalescing power saving mode, the 82580 will:

- Flush all pending interrupts that were delayed due to the Interrupt Throttling (ITR) mechanism.
- The 82580 will flush all pending receive descriptor write backs and pre-fetch available receive descriptors to internal cache.

Note: The 82580 enters DMA Coalescing mode only when above conditions exist on all active functions.

5.7.2 Conditions to Exit DMA Coalescing

When in DMA Coalescing operating mode, PCIe link is placed in a link state as defined in the *DMACR.*DMAC_Lx field and no PCIe activity is initiated by the 82580. In this mode generation of flow control packets is defined by the *FCRTC.RTH_Coal* threshold field. DMA Coalescing mode is exited when one of the following events occur:

1. PCIe link is active (L0 link power state).
2. Amount of data in internal receive buffers passed the *DMACR.DMACTHR* threshold.

3. Empty space in internal transmit buffer is above the value defined in *DMCTXTH.DMCTTHR* field and available transmit descriptors exist.
4. A high priority packet was received (See Section 7.3.6 for definition of high priority packets). A high priority packet is defined as packet that generates an immediate interrupt when received.
5. A received packet destined to a high priority queue (*SRRCTL[n].DMACQ_Dis* =1b) was detected.
6. An Interrupt is pending (Link disconnect, TCP timer …).
7. DMA coalescing Watchdog timer defined in the *DMACR.DMACWT* field expires as a result of a received packet not being serviced for a long duration.
8. Packet rate lower than defined in *DMCRTRH.UTRESH* field is detected.
9. DMA Coalescing is disabled (*DMACR.DMAC_EN* = 0).
10. Another the 82580 function initiated a transaction on the PCIe bus.

Note: Even when conditions for DMA Coalescing do not exist, the 82580 will continue to be in low power PCIe link state (L0s or L1) if there is no requirement for PCIe access.

5.8 Latency Tolerance Reporting (LTR)

The 82580 generates PCIe LTR messages to report service latency requirements for memory reads and writes to the Root Complex for system power management.

The 82580 will report either minimum latency tolerance, maximum latency tolerance or no latency tolerance requirements as a function of link, LAN port and function status. Minimum and maximum latency tolerance values are programmed in the *LTRMINV* and *LTRMAXV* registers respectively by the software driver to optimize power consumption without incurring packet loss due to receive buffer overflow. The 82580 sends LTR messages according to the following algorithm when the capability is enabled:

1. When Links on all ports are disconnected or all LAN ports are disabled (transmit and receive activity not enabled) and the *LTRC.LNKDLS_EN* and *LTRC.PDLS_EN* bits are set respectively, the 82580 will send a LTR PCIe message with LTR Requirement bits cleared, to indicate that no Latency tolerance requirements exists.
2. When software sets *LTRC.LTR_MAX*, the 82580 will send a LTR message with the value placed in the *LTRMAXV* register.
3. Otherwise, the 82580 will send a LTR message with a minimum value.

Note: In all cases maximum LTR Value sent by the 82580 do not exceed the maximum latency values in the *Max No-Snoop Latency* and *Max Snoop Latency* Registers in the Latency Tolerance Reporting (LTR) Capability structure of Function 0.

Figure 5-7 describes the 82580 LTR message generation flow.

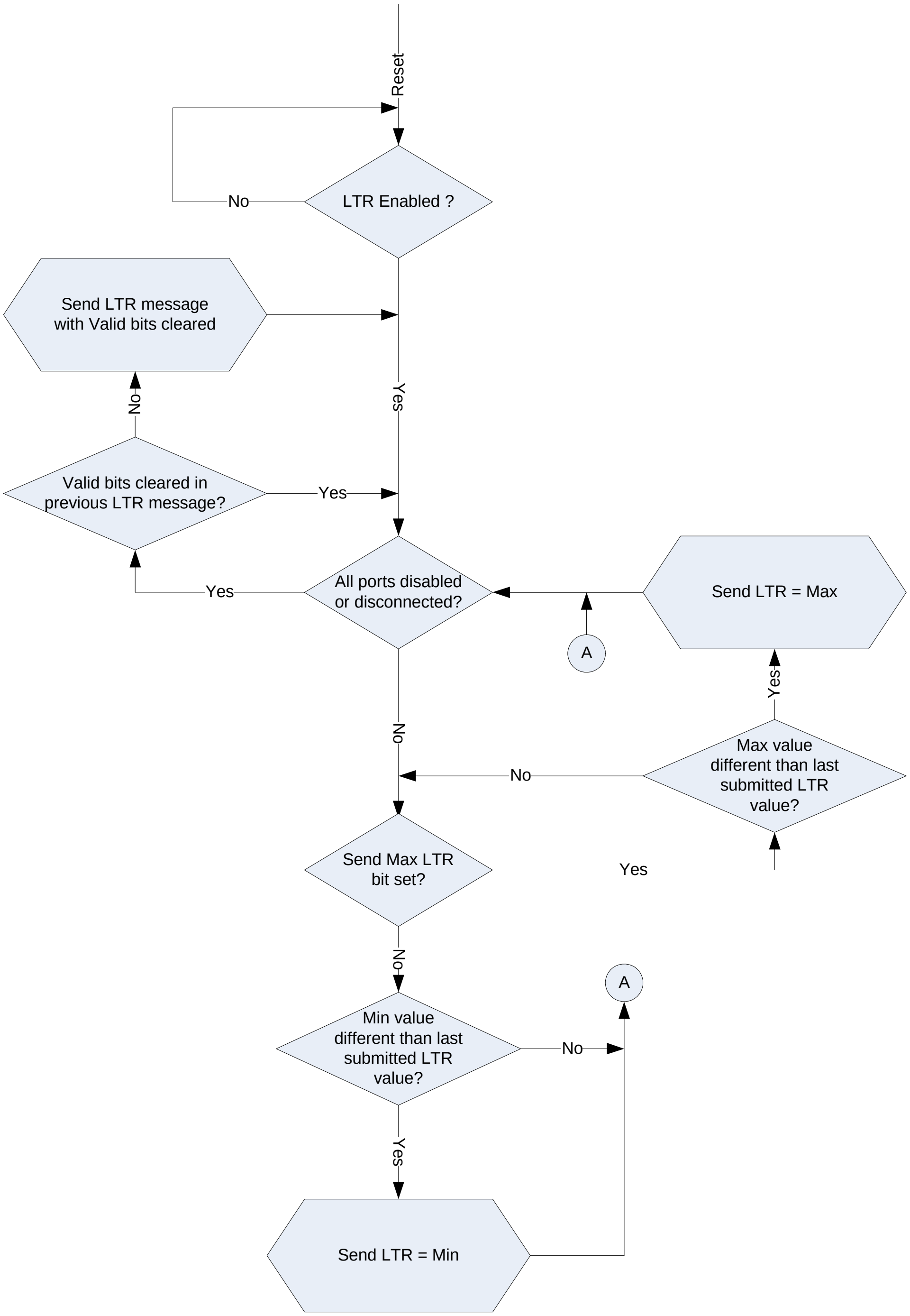

Reset
LTR Enabled ?
No
Send LTR message with Valid bits cleared
Yes
No
Valid bits cleared in previous LTR message?
Yes
All ports disabled or disconnected?
Yes
Send LTR = Max
A
Yes
No
Max value different than last submitted LTR value?
No
Send Max LTR bit set?
Yes
No
A
Min value different than last submitted LTR value?
No
Yes
Send LTR = Min

Figure 5-7. PCIe LTR Message Generation Flow per Function

5.8.1 Latency Tolerance Reporting Per Function

Internal to the 82580 each function can request to generate a Minimum value LTR, a Maximum value LTR and a LTR message with the Requirement bits cleared. The 82580 conglomerates latency requirements from the functions that have LTR messaging enabled and sends a single LTR message in the following manner:

- The acceptable latency values for the message sent upstream by the 82580 must reflect the lowest latency tolerance values associated with any function.
  - It is permitted that the Snooped and Non-Snooped values reported in the conglomerated message are associated with different functions.
  - If none of the functions have a Latency requirement for a certain type of traffic (Snoop/Non-snoop), the message sent by the 82580 will not have the Requirement bit corresponding to that type of traffic set.
- The 82580 transmits a new LTR message upstream when the capability is enabled and when any function changes the values it has reported internally in such a way as to change the conglomerated value reported previously by the 82580.

Each function in the 82580 reports support of LTR messaging in the configuration space by:

- Setting the *LTR Mechanism Supported* bit in the PCIe *Device Capabilities 2* configuration register (Support defined by *LTR_EN* bit in *Initialization Control Word 1* EEPROM word, that controls enabling of the LTR structures).
- Supporting the *Latency Tolerance Requirement Reporting (LTR) Capability* structure in the PCIe configuration space.

To enable generation of LTR messages by a function the *LTR Mechanism Enable* bit in the *Device Control 2* configuration register of function 0 should be set.

Note: A function that does not have LTR messaging enabled is considered a function that does not have any Latency Tolerance requirements.

5.8.1.1 Conditions for Generating LTR Message with the Requirement Bits Cleared

When LTR messaging is enabled the 82580 functions will send a LTR message with the Requirement bits cleared in the following cases:

1. Following PE_RST_N assertion (PCIe reset) after LTR capability is enabled.
2. LAN port is disabled (both *RCTRL.RXEN* and *TCTL.EN* are cleared), receive buffer is empty and *LTRC.PDLS_EN* is set.
3. LAN port is disconnected, BMC to Host traffic is disabled (*MANC.EN_BMC2HOST* = 0) and *LTRC.LNKDLS_EN* is set.
4. Function is not in D0a state.
5. When the LSNP and LNSNP bits are cleared in the LTRMINV register and minimum LTR value needs to be sent.
6. When the LSNP and LNSNP bits are cleared in the LTRMAXV register and maximum LTR value needs to be sent.

Note: A disabled function does not generate Latency Tolerance requirements.

5.8.1.2 Conditions for Generating LTR Message with Maximum LTR Value

When LTR messaging is enabled and conditions to send LTR Message with Valid bits cleared do not exist, the 82580 functions will send a maximum value LTR message, with the values programmed in the *LTRMAXV* register in the following cases:

1. Following a software write 1 operation to *LTRC.LTR_MAX* bit.
2. When updated data was written to the *LTRMAXV* register and condition defined in 1. to send a LTR message with a maximum value exists.

Note: When the *LTRC.LTR_MAX* bit is cleared, the 82580 will send a LTR message with the value placed in the *LTRMINV* register, if the value is smaller than the value placed in the *LTRMAXV* register.

5.8.1.3 Conditions for Generating LTR Message with Minimum LTR Value

When LTR messaging is enabled, the 82580 functions will send a minimum value LTR message, with the values programmed in the *LTRMINV* register in the following cases:

1. Following a software write 1 operation to the *LTRC.LTR_MIN* bit.
2. When updated data was written to the *LTRMINV* register and conditions to send LTR message with Valid bits cleared or maximum value LTR do not exist.

§ §

NOTE: This page intentionally left blank.

6.0 Non-Volatile Memory Map - EEPROM

6.1 EEPROM General Map

The 82580 EEPROM is partitioned into 4 main blocks followed by Firmware and PXE structures. First 128 word section is allocated to words common to all LAN ports, Firmware, Software, PXE and LAN port 0. Following 64 word sections are allocated to Lan port 1, Lan port 2 and Lan port 3 as shown in Table 6-84.

A detailed list of EEPROM words loaded by Hardware following Power-up, Hardware reset or software generated resets (*CTRL.RST*, *CTRL_EXT.EE_RS*T or *CTRL.DEV_RST*) can be found in the auto load sequence table in Section 3.3.1.3.

Table 6-1. EEPROM Top Level Partitioning

EEPROM Word Offsets	Partition
0x00 to 0x7F	Common Words (PCIe, PXE, SW and FW) and LAN port 0 words - see Table 6-85
0x80 to 0xBF	LAN Port 1 words - see Table 6-86
0xC0 to 0xFF	LAN Port 2 words* - see Table 6-86
0x100 to 0x13F	LAN Port 3 words* - see Table 6-86
0x140...	PXE and Firmware Structures

* Ignored on the dual port 82580DB.

Table 6-85 lists the 82580 EEPROM word map for Common words and Lan Port 0.

Table 6-2. Common and Lan Port 0 EEPROM Map

EEPROM Word Offsets	Used By/In	High Byte	Low Byte	Which LAN
0x00	HW	00*	AA or A0	LAN 0
0x01	HW	00 or C9	variable	LAN 0
0x02	HW	variable	variable	LAN 0
		*Section 6.2.1, Ethernet Address (LAN Base Address + Offsets 0x00-0x02)		
0x03	SW	Compatibility High	Compatibility Low	All
0x04	SW	Compatibility High	Compatibility Low	All
0x05	SW	Compatibility High	Compatibility Low	All
0x06	SW	Compatibility High	Compatibility Low	All
0x07	SW	Compatibility High	Compatibility Low	All
0x08	SW	Section 6.11.5, PBA Number (Word 0x08, 0x09)		All
0x09	SW			All
0x0A	HW	Section 6.2.2, Initialization Control Word 1 (word 0x0A)		All

Table 6-2. Common and Lan Port 0 EEPROM Map (Continued)

EEPROM Word Offsets	Used By/In	High Byte	Low Byte	Which LAN
0x0B	HW	Section 6.2.3, Subsystem ID (Word 0x0B)		All
0x0C	HW	Section 6.2.4, Subsystem Vendor ID (Word 0x0C)		All
0x0D	HW	Section 6.2.5, Device ID (LAN Base Address + Offset 0x0D)		LAN 0
0x0E	HW	Section 6.2.4, Subsystem Vendor ID (Word 0x0C)		All
0x0F	HW	Section 6.2.8, Initialization Control Word 2 (Word 0x0F)		All
0x10	HW	Section 6.5, PCIe PHY Auto Configuration Pointer (Word 0x10)		All
0x11	HW (MNG HW)	Section 6.6.1, Pass Through LAN Configuration Pointer (LAN Base Address + Offset 0x11)		LAN 0 (MNG HW)
0x12	HW	Section 6.2.9, EEPROM Sizing and Protected Fields (Word 0x12)		All
0x13	HW	Section 6.2.10, Initialization Control 4 (LAN Base Address + Offset 0x13)		LAN 0
0x14	HW	Section 6.2.11, PCIe L1 Exit latencies (Word 0x14)		All
0x15	HW	Section 6.2.12, PCIe Completion Timeout Configuration (Word 0x15)		All
0x16	HW	Section 6.2.13, MSI-X Configuration (LAN Base Address + Offset 0x16)		LAN 0
0x17	HW	Section 6.3, CSR Auto Configuration Pointer (LAN Base Address + Offset 0x17)		LAN 0
0x1B	HW	Section 6.2.14, PCIe Control 1 (Word 0x1B)		All
0x1C	HW	Section 6.2.15, LED 1,3 Configuration Defaults (LAN Base Address + Offset 0x1C)		LAN 0
0x1D	HW	Section 6.2.7, Dummy Device ID (Word 0x1D)		All
0x1E	HW	Section 6.2.16, Device Rev ID (Word 0x1E)		All
0x1F	HW	Section 6.2.17, LED 0,2 Configuration Defaults (LAN Base Address + Offset 0x1F)		LAN 0
0x20	HW	Section 6.2.18, Software Defined Pins Control (LAN Base Address + Offset 0x20)		LAN 0
0x21	HW	Section 6.2.19, Functions Control (Word 0x21)		All
0x22	HW	Section 6.2.20, LAN Power Consumption (Word 0x22)		All
0x23	HW	Section 6.6.7, Management HW Config Control (Word 0x23)		MNG HW
0x24	HW	Section 6.2.21, Initialization Control 3 (LAN Base Address + Offset 0x24)		LAN 0
0x25	HW	Reserved		
0x26	HW	Reserved		
0x27	HW	Reserved		LAN0
0x27	HW	Section 6.4, CSR Auto Configuration Power-Up Pointer (LAN Base Address + Offset 0x27)		LAN0
0x28	HW	Section 6.2.22, PCIe Control 2 (Word 0x28)		All
0x29	HW	Section 6.2.23, PCIe Control 3 (Word 0x29)		All
0x2A	HW	Reserved		All
0x2B	HW	Reserved		All
0x2C	HW	Reserved		All
0x2D	HW	Section 6.2.25, Start of RO Area (Word 0x2D)		All
0x2E	HW	Section 6.2.26, Watchdog Configuration (Word 0x2E)		All
0x2F	OEM	Section 6.2.27, VPD Pointer (Word 0x2F)		
0x30	PXE	Section 6.11.6.1, Setup Options PCI Function 0 (Word 0x30)		
0x31	PXE	Section 6.11.6.2, Configuration Customization Options PCI Function 0 (Word 0x31)		
0x32	PXE	Section 6.11.6.3, PXE Version (Word 0x32)		
0x33	PXE	Section 6.11.6.4, IBA Capabilities (Word 0x33)		

Table 6-2. Common and Lan Port 0 EEPROM Map (Continued)

EEPROM Word Offsets	Used By/In	High Byte	Low Byte	Which LAN
0x34	PXE	Section 6.11.6.5, Setup Options PCI Function 1 (Word 0x34)		
0x35	PXE	Section 6.11.6.6, Configuration Customization Options PCI Function 1 (Word 0x35)		
0x38	PXE	Section 6.11.6.7, Setup Options PCI Function 2 (Word 0x38)		
0x39	PXE	Section 6.11.6.8, Configuration Customization Options PCI Function 2 (Word 0x39)		
0x3A	PXE	Section 6.11.6.9, Setup Options PCI Function 3 (Word 0x3A)		
0x3B	PXE	Section 6.11.6.10, Configuration Customization Options PCI Function 3 (Word 0x3B)		
0x3D	PXE	Section 6.11.7, iSCSI Boot Configuration Pointer (Word 0x3D)		
0x3E	PXE	Reserved		
0x3F	SW	Section 6.11.9, Checksum Word (Offset 0x3F)		
0x40:0x41	SW	Reserved		
0x42	SW	Section 6.11.10, Image Unique ID (Word 0x42, 0x43)		
0x43	SW	Section 6.11.10, Image Unique ID (Word 0x42, 0x43)		
0x44:0x4F	SW	Reserved		
0x50	FW	Section 6.6.2, PHY Configuration Pointer (Word 0x50)		MNG
0x51	FW	Section 6.6.3, Firmware Patch Pointer (Word 0x51)		MNG Code
0x52:0x53	FW	Reserved		MNG
0x54	FW	Section 6.6.6, Manageability Capability/Manageability Enable (Word 0x54)		MNG HW
0x55:0x56	FW	Reserved		MNG
0x57	FW	Section 6.6.4, Sideband Configuration Pointer (Word 0x57)		MNG HW
0x58:0x5D	FW	Reserved		MNG
0x5E	FW	Reserved		MNG
0x5F:0x7F	FW	Reserved		MNG
0x80:0xBF	LAN Port 1 words - see Table 6-86			
0xC0:0xFF	LAN Port 2 words - see Table 6-86			
0x100:0x13F	LAN Port 3 words - see Table 6-86			
0x140...	PXE and Firmware Structures			

Table 6-86 maps the 82580 EEPROM words that can hold different content for LAN Ports 0, 1, 2 and 3. Addresses listed in the table are an offset from the LAN Base address of the relevant EEPROM LAN section. EEPROM LAN Base addresses of the LAN ports are as follows:

- LAN Port 0 EEPROM section Base Address - 0x0
- LAN Port 1 EEPROM section Base Address – 0x80
- LAN Port 2 EEPROM section Base Address – 0xC0
- LAN Port 3 EEPROM section Base Address – 0x100

Table 6-3. LAN Ports 1, 2 and 3 EEPROM Map

EEPROM Word Offsets	Used By/In	High Byte	Low Byte
0x00	HW	00*	AA or A0
0x01	HW	00 or C9	variable
0x02	HW	variable	variable
		*Section 6.2.1, Ethernet Address (LAN Base Address + Offsets 0x00-0x02)	
0x03:0x0C	HW	Reserved	
0x0D	HW	Section 6.2.5, Device ID (LAN Base Address + Offset 0x0D)	
0x0E:0x10	HW	Reserved	
0x11	HW (MNG HW)	Section 6.6.1, Pass Through LAN Configuration Pointer (LAN Base Address + Offset 0x11)	
0x12	HW	Reserved	
0x13	HW	Section 6.2.10, Initialization Control 4 (LAN Base Address + Offset 0x13)	
0x14:0x15	HW	Reserved	
0x16	HW	Section 6.2.13, MSI-X Configuration (LAN Base Address + Offset 0x16)	
0x17	HW	Section 6.3, CSR Auto Configuration Pointer (LAN Base Address + Offset 0x17)	
0x18:0x1B	HW	Reserved	
0x1C	HW	Section 6.2.15, LED 1,3 Configuration Defaults (LAN Base Address + Offset 0x1C)	
0x1D:0x1E	HW	Reserved	
0x1F	HW	Section 6.2.17, LED 0,2 Configuration Defaults (LAN Base Address + Offset 0x1F)	
0x20	HW	Section 6.2.18, Software Defined Pins Control (LAN Base Address + Offset 0x20)	
0x21:0x23	HW	Reserved	
0x24	HW	Section 6.2.21, Initialization Control 3 (LAN Base Address + Offset 0x24)	
0x25:0x26	HW	Reserved	
0x27	HW	Section 6.4, CSR Auto Configuration Power-Up Pointer (LAN Base Address + Offset 0x27)	
0x28:0x3E	HW	Reserved	
0x3F	SW	Section 6.11.9, Checksum Word (Offset 0x3F)	

6.2 Hardware Accessed Words

This section describes the EEPROM words that are loaded by 82580 hardware. Most of these bits are located in configuration registers. The words are only read and used if the signature field in the *EEPROM Sizing & Protected Fields* EEPROM word (word 0x12) is valid.

Note: When **Word** is mentioned before an EEPROM address, address is the absolute address in the EEPROM. When **Offset** is mentioned before an EEPROM address, the address is relative to the start of the relevant EEPROM section.

6.2.1 Ethernet Address (LAN Base Address + Offsets 0x00-0x02)

The Ethernet Individual Address (IA) is a 6-byte field that must be unique for each NIC, and thus unique for each copy of the EEPROM image. The first three bytes are vendor specific. For example, the IA is equal to [00 AA 00] or [00 A0 C9] for Intel products. The value from this field is loaded into the Receive Address Register 0 (RAL0/RAH0).

For the purpose of this specification, the IA byte numbering convention is indicated as follows:

	IA Byte / Value					
Vendor	**1**	**2**	**3**	**4**	**5**	**6**
Intel Original	00	AA	00	variable	variable	variable
Intel New	00	A0	C9	variable	variable	variable

The Ethernet address is loaded for LAN0 from Addresses 0x0 to 0x02 and for LAN 1, 2 and 3 from offsets 0x0 to 0x2 at the start of the relevant sections.

6.2.2 Initialization Control Word 1 (word 0x0A)

The *Initialization Control Word 1* in the Common section contains initialization values that:

- Set defaults for some internal registers
- Enable/disable specific features
- Determine which PCI configuration space values are loaded from the EEPROM

Bit	Name	Default in EEPROM-less mode	Description
15:14	Reserved		Reserved
13	LTR_EN	1b	LTR capabilities reporting enable. 0 - Do not report LTR support in the PCIe configuration Device Capabilities 2 register. 1 - Report LTR support in the PCIe configuration Device Capabilities 2 register. Defines default setting of LTR capabilities reporting (See Section 9.5.6.11).
12	Reserved		Reserved
11	FRCSPD	0b	Default setting for the *Force Speed* bit in the Device Control register (CTRL[11]). See See Section 8.2.1
10	FD	0b	Default setting for duplex setting. Mapped to CTRL[0]. See See Section 8.2.1
9:7	Reserved		Reserved. Value should be 110b.
6	SDP_IDDQ_EN	0b	When set, SDP IOs keep their value and direction when the 82580 enters dynamic IDDQ mode either due to PCIe entering Dr state or DEV_OFF_N pin being asserted. Otherwise, SDP IOs moves to HighZ + pull-up mode in dynamic IDDQ mode. Reflected in EEDIAG (See Section 8.4.5).
5	Deadlock Timeout Enable	1b	If set, a device granted access to the EEPROM or Flash that does not toggle the interface for more than 2 seconds will have the grant revoked. See Section 3.3.2.1.
4	Reserved	0b	Reserved.
3	Power Management	1b	0b = Power Management registers set to read only. In this mode, the 82580 does not execute a hardware transition to D3 .Reserved 1b = Full support for power management (For normal operation, this bit must be set to 1b). See Section 9.5.1 .
2	Reserved		Reserved

Bit	Name	Default in EEPROM-less mode	Description
1	Load Subsystem IDs	1b	When this bit is set to 1b the 82580 loads its PCIe Subsystem ID and Subsystem Vendor ID from the EEPROM (*Subsystem ID* and *Subsystem Vendor ID* EEPROM words).
0	Load Vendor/Device IDs	1b	When set to 1b the 82580 loads its PCIe Device IDs from the EEPROM (*Device ID* or *Dummy Device ID* EEPROM words) and the PCIe Vendor ID from the EEPROM.

6.2.3 Subsystem ID (Word 0x0B)

If the *Load Subsystem IDs* in *Initialization Control Word 1* EEPROM word is set, the *Subsystem ID* word in the Common section is read in to initialize the PCIe Subsystem ID. Default value is 0x0 (See Section 9.4.14).

6.2.4 Subsystem Vendor ID (Word 0x0C)

If the *Load Subsystem IDs* bit in *Initialization Control Word 1* EEPROM word is set, the Subsystem Vendor ID word in the Common section is read in to initialize the PCIe Subsystem Vendor ID. The default value is 0x8086 (See Section 9.4.13).

6.2.5 Device ID (LAN Base Address + Offset 0x0D)

If the *Load Vendor/Device IDs* bit in *Initialization Control Word 1* is set, the *Device ID* EEPROM word is read in from the Common, LAN 1, LAN 2 and LAN 3 sections to initialize the Device ID of LAN0, LAN1, LAN2 and LAN3 functions, respectively. The default value is 0x1509 (See Section 9.4.2).

6.2.6 Vendor ID (Word 0x0E)

If the *Load Vendor/Device IDs* bit in *Initialization Control Word 1* EEPROM word is set, this word is read in to initialize the PCIe Vendor ID. The default value is 0x8086 (See Section 9.4.1).

Note: If a value of 0xFFFF is placed in the *Vendor ID* EEPROM word, the value in the PCIe *Vendor ID* register will return to the default 0x8086 value. This functionality is implemented to avoid a system hang situation.

6.2.7 Dummy Device ID (Word 0x1D)

If the *Load Vendor/Device IDs* bit in *Initialization Control Word 1* EEPROM word is set, this word is read in to initialize the Device ID of dummy devices. The default value is 0x10A6 (See Section 9.4.2).

6.2.8 Initialization Control Word 2 (Word 0x0F)

The *Initialization Control Word 2* read by the 82580, contains additional initialization values that:

- Set defaults for some internal registers
- Enable/disable specific features

Bit	Name	Default in EEPROM-less mode	Description
15	APM PME# Enable	0b	Initial value of the *Assert PME On APM Wakeup* bit in the Wake Up Control (WUC.APMPME) register. See Section 8.21.1.
14	PCS parallel detect	1b	Enables PCS parallel detect. Mapped to *PCS_LCTL.AN TIMEOUT EN* bit. See See Section 8.18.2 . Note: Bit should be 0b only when port operates in SGMII mode (*CTRL_EXT.LINK_MODE* = 10b).
13:12	Pause Capability	11b	Desired pause capability for advertised configuration base page. Mapped to *PCS_ANADV.ASM*. See Section 8.18.4.
11	ANE	0b	Auto-Negotiation Enable Mapped to *PCS_LCTL.AN_ENABLE*. See See Section 8.18.2 . Note: Bit should be 0b only when port operates in internal copper PHY mode (*CTRL_EXT.LINK_MODE* = 00b).
10:8	Flash Size	000b	Indicates Flash size according to the following equation: Size = 64 KB * 2**(Flash Size field). From 64 KB up to 8 MB in powers of 2. The Flash size impacts the requested memory space for the Flash and expansion ROM BARs in PCIe configuration space (See CSRSize and FLSize fields in the BARCTRL register in Section 8.6.12). Note: When CSR_Size and Flash_size fields in the EEPROM are set to 0, Flash access BAR in the PCI configuration space is disabled.
7	MAC clock gating enable	1b	Enables MAC clock gating power saving mode. Mapped to *STATUS[31]*. This bit is relevant only if the *Enable Dynamic MAC Clock Gating* bit is set. See Section 8.2.2.
6	PHY Power Down Enable	1b	When set, enables the Internal PHY to enter a low-power state (See Section 3.5.7.5). This bit is mapped to CTRL_EXT[20] (See Section 8.2.3).
5	CSR_Size	0b	The CSR_Size and FLASH_Size fields define the usable FLASH size and CSR mapping window size as shown in BARCTRL register description (See Section 8.6.12). Note: When CSR_Size and Flash_size fields in the EEPROM are set to 0, Flash access BAR in the PCI configuration space is disabled.
4	LAN PLL Shutdown Enable	0b	When set, enables shutting down the PHY PLL in low-power states when the Internal PHY is powered down (such as link disconnect). When cleared, the PHY PLL is not shut down in a low-power state. Reflected in EEDIAG (See Section 8.4.5).
3	Enable Dynamic MAC Clock Gating	0b	When set, enables dynamic MAC clock gating mechanism. See Section 8.2.3.
2	SerDes Low Power Enable	0b	When set, enables the SerDes to enter a low power state when the function is in Dr state. See Chapter 5.0 and Section 8.2.3.
1	DMA clock gating Disabled	1b	When set Disables DMA clock gating power saving mode.
0	Reserved		Reserved

6.2.9 EEPROM Sizing and Protected Fields (Word 0x12)

Provides indication on EEPROM size and protection.

Note: If the Enable Protection Bit in this word is set and the signature is valid, the software device driver has read but no write access to this word via the EEC and EERD registers; In this case, write access is possible only via an authenticated firmware interface.

Bit	Name	Default in EEPROM-less mode	Description
15:14	Signature		The *Signature* field indicates to the 82580 that there is a valid EEPROM present. If the signature field is 01b, EEPROM read is performed, otherwise the other bits in this word are ignored, no further EEPROM read is performed, and default values are used for the configuration space IDs.
13:10	EEPROM Size	0010b	These bits indicate the EEPROM's actual size. Mapped to *EEC.EE_SIZE* See (Section 8.4.1). Field Value EEPROM Size EEPROM Address Size 0000b - 0110b Reserved 0111b 16 Kbytes 2 bytes 1000b 32 Kbytes 2 bytes 1001b - 1111b Reserved
9:5	Reserved		Reserved
4	Enable EEPROM Protection	0b	If set, all EEPROM protection schemes are enabled.
3:0	HEPSize	0x0	Hidden EEPROM Block Size This field defines the EEPROM area accessible only by manageability firmware. It can 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 2^ HEPSize bytes else (for example, 2 B, 4 B, …32 KB)

6.2.10 Initialization Control 4 (LAN Base Address + Offset 0x13)

These words control general initialization values of LAN 0, LAN 1, LAN 2 and LAN 3 ports.

Bit	Name	Default in EEPROM-less mode	Description
15:12	TXPbsize	0x0	Transmit internal buffer size: 0x0 - 20 KB 0x1 - 40 KB 0x2 - 80 KB 0x3 - 1 KB 0x4 - 2 KB 0x5 - 4 KB 0x6 - 8 KB 0x7 - 16 KB 0x8 - 19 KB 0x9 - 38 KB 0xA - 76 KB 0xB:0XF reserved. When 4 ports are enabled maximum buffer size is 20KB. When 2 ports are enabled maximum buffer size is 40KB. When only a single port is enabled maximum buffer size is 80KB. Sets value of ITPBS.TXPbsize. See Section 8.3.
11:8	RXPbsize	0x0	Receive internal buffer size: 0x0 - 36 KB 0x1 - 72 KB 0x2 - 144 KB 0x3 - 1 KB 0x4 - 2 KB 0x5 - 4 KB 0x6 - 8 KB 0x7 - 16 KB 0x8 - 35 KB 0x9 - 70 KB 0xA - 140 KB Note: When 4 ports are enabled maximum buffer size is 36 KB. When 2 ports are enabled maximum buffer size is 72 KB. When only a single port is enabled maximum buffer size is 144 KB. Sets value of *IRPBS.RXPbsize*. See Section 8.3.
7	SPD Enable	1b	Smart Power Down – When set, enables Internal PHY Smart Power Down mode (See Section 3.5.7.5.5).
6	LPLU	1b	Low Power Link Up Enables a decrease in link speed in non-D0a states when power policy and power management states dictate it (See Section 3.5.7.5.4).

Bit	Name	Default in EEPROM-less mode	Description
5:1	PHY_ADD	0x00 0x01 0x02 0x03	PHY address. Value loaded to *MDICNFG.PHYADD* field. See Section 8.2.5.
0	DEV_RST_EN	1b	Enable software reset (*CTRL.DEV_RST*) generation to all ports (See Section 4.3).

6.2.11 PCIe L1 Exit latencies (Word 0x14)

Bits	Name	Default in EEPROM-less mode	Description
15	Reserved	1b	Reserved
14:12	L1_Act_Acc_Latency	110b	Loaded to the "Endpoint L1 Acceptable Latency" field in the "Device Capabilities" in the "PCIe configuration registers" at power up.
11:9	L1 G2 Sep exit latency	101	L1 exit latency G2S. Loaded to "Link Capabilities" -> "L1 Exit Latency" at PCIe v2.0 (5Gbps) system in Separate clock setting.
8:6	L1 G2 Com exit latency	011b	L1 exit latency G2C. Loaded to "Link Capabilities" -> "L1 Exit Latency" at PCIe v2.0 (5Gbps) system in Common clock setting.
5:3	L1 G1 Sep exit latency	100b	L1 exit latency G1S. Loaded to "Link Capabilities" -> "L1 Exit Latency" at PCIe v2.0 (2.5Gbps) system in Separate clock setting.
2:0	L1 G1 Com exit latency	010b	L1 exit latency G1C. Loaded to "Link Capabilities" -> "L1 Exit Latency" at PCIe v2.0 (2.5Gbps) system in Common clock setting.

6.2.12 PCIe Completion Timeout Configuration (Word 0x15)

Bit	Name	Default in EEPROM-less mode	Description
15:5	Reserved		Reserved. Set value to 0x2.
4	Completion Timeout Resend	1b	When set, enables to resend a request once the completion timeout expired 0b = Do not re-send request on completion timeout. 1b = Re-send request on completion timeout. See Section 8.6.1.
3::0	Reserved		Reserved

6.2.13 MSI-X Configuration (LAN Base Address + Offset 0x16)

These words configure MSI-X functionality for LAN 0, LAN 1, LAN 2 and LAN 3.

Bit	Name	Default in EEPROM-less mode	Description
15:11	MSI_X_N	0x9	This field specifies the number of entries in MSI-X tables of the relevant LAN. The range is 0-9. MSI_X_N is equal to the number of entries minus one. See Section 9.5.3.3.
10	MSI Mask	1b	MSI per-vector masking setting. This bit is loaded to the masking bit (bit 8) in the *Message Control* word of the *MSI Configuration Capability structure*.
9:0	Reserved		Reserved

6.2.14 PCIe Control 1 (Word 0x1B)

This word is used to configure initial settings for PCIe default functionality.

Bit	Name	Default in EEPROM-less mode	Description
15	Reserved	0b	Reserved - must be zero
14	Dummy Function Enable	0b	Controls the behavior of function 0 when disabled. See Section 4.4.2. 0b - Legacy Mode 1b - Dummy Function Mode
13:8	Reserved		Reserved
7	Reserved	0b	Reserved must be 0b
6:0	Reserved		Reserved

6.2.15 LED 1,3 Configuration Defaults (LAN Base Address + Offset 0x1C)

These EEPROM words specify the hardware defaults for the LEDCTL register fields controlling the LED1 (ACTIVITY indication) and LED3 (LINK_1000 indication) output behavior. Words control LEDs behavior of LAN 0, LAN 1, LAN 2 and LAN 3 ports.

Bit	Name	Default in EEPROM-less mode	Description
3:0	LED1 Mode	0011b	Initial value of the *LED1_MODE* field specifying what event/state/pattern is displayed on LED1 (ACTIVITY) output. A value of 0011b (0x3) indicates the ACTIVITY state. See Section 8.2.9 and Section 7.5.
5:4	Reserved		Reserved
6	LED1 Invert	0b	Initial value of *LED1_IVRT* field. 0b = Active-low output. See Section 8.2.9 and Section 7.5.
7	LED1 Blink	1b	Initial value of *LED1_BLINK* field. 0b = Non-blinking. See Section 8.2.9 and Section 7.5.
11:8	LED3 Mode	0111b	Initial value of the *LED3_MODE* field specifying what event/state/pattern is displayed on LED3 (LINK_1000) output. A value of 0111b (0x7) indicates 1000 Mb/s operation. See Section 8.2.9 and Section 7.5.
13:12	Reserved		Reserved
14	LED3 Invert	0b	Initial value of *LED3_IVRT* field. 0b = Active-low output. See Section 8.2.9 and Section 7.5.
15	LED3 Blink	0b	Initial value of *LED3_BLINK* field. 0b = Non-blinking. See Section 8.2.9 and Section 7.5.

A value of 0x0703 is used to configure default hardware LED behavior equivalent to previous copper adapters (LED0=LINK_UP, LED1=blinking ACTIVITY, LED2=LINK_100, and LED3=LINK_1000).

6.2.16 Device Rev ID (Word 0x1E)

Bit	Name	Default in EEPROM-less mode	Description
15	Power Down Enable	0b	Enable Power down when DEV_OFF_N pin is asserted or PCIe in Dr state. See Section 5.2.4.1 for details. Reflected in EEDIAG (See Section 8.4.5).
14	LAN 3 iSCSI enable	0b	When set, LAN 3 class code is set to 0x010000 (SCSI) When reset, LAN 3 class code is set to 0x020000 (LAN) See Section 9.4.5.
13	LAN 2 iSCSI enable	0b	When set, LAN 2 class code is set to 0x010000 (SCSI) When reset, LAN 2 class code is set to 0x020000 (LAN) See Section 9.4.5.
12	LAN 1 iSCSI enable	0b	When set, LAN 1 class code is set to 0x010000 (SCSI) When reset, LAN 1 class code is set to 0x020000 (LAN) See Section 9.4.5.

Bit	Name	Default in EEPROM-less mode	Description
11	LAN 0 iSCSI enable	0b	When set, LAN 0 class code is set to 0x010000 (SCSI) When reset, LAN 0 class code is set to 0x020000 (LAN) See Section 9.4.5.
10:8	Reserved		Reserved
7:0	DEVREVID	0x0	Device Revision ID The actual device revision ID is the EEPROM value XORed with the hardware default of Rev ID. For the 82580, the default value is zero. See Section 9.4.5.

6.2.17 LED 0,2 Configuration Defaults (LAN Base Address + Offset 0x1F)

These EEPROM words specify the hardware defaults for the LEDCTL register fields controlling the LED0 (LINK_UP) and LED2 (LINK_100) output behaviors. Words control LEDs behavior of LAN 0, LAN 1, LAN 2 and LAN 3 ports.

Bit	Name	Default in EEPROM-less mode	Description
3:0	LED0 Mode	0010b	Initial value of the *LED0_MODE* field specifying what event/state/pattern is displayed on LED0 (LINK_UP) output. A value of 0010b (0x2) indicates the LINK_UP state. See Section 8.2.9 and Section 7.5.
4	Reserved	0b	Reserved. Set to 0b.
5	Global Blink Mode	0b	Global Blink Mode 0b = Blink at 200 ms on and 200ms off. 1b = Blink at 83 ms on and 83 ms off. See Section 8.2.9 and Section 7.5.
6	LED0 Invert	0b	Initial value of *LED0_IVRT* field. 0b = Active-low output. See Section 8.2.9 and Section 7.5.
7	LED0 Blink	0b	Initial value of *LED0_BLINK* field. 0b = Non-blinking. See Section 8.2.9 and Section 7.5.
11:8	LED2 Mode	0110b	Initial value of the *LED2_MODE* field specifying what event/state/pattern is displayed on LED2 (LINK_100) output. A value of 0110b (0x6) indicates 100 Mb/s operation. See Section 8.2.9 and Section 7.5.
12	Reserved	0b	Reserved. Set to 0b.
13	Reserved	0b	Reserved
14	LED2 Invert	0b	Initial value of *LED2_IVRT* field. 0b = Active-low output. See Section 8.2.9 and Section 7.5.
15	LED2 Blink	0b	Initial value of *LED2_BLINK* field. 0b = Non-blinking. See Section 8.2.9 and Section 7.5.

A value of 0x0602 is used to configure default hardware LED behavior equivalent to previous copper adapters (LED0=LINK_UP, LED1=blinking ACTIVITY, LED2=LINK_100, and LED3=LINK_1000).

6.2.18 Software Defined Pins Control (LAN Base Address + Offset 0x20)

These words at offset 0x20 from start of relevant EEPROM section are used to configure initial settings of software defined pins (SDPs) for LAN 0, LAN 1, LAN 2 and LAN 3.

Bit	Name	Default in EEPROM-less mode	Description
15	SDPDIR[3]	0b	SDP3 Pin – Initial Direction This bit configures the initial hardware value of the *SDP3_IODIR* bit in the Extended Device Control (CTRL_EXT) register following power up. See Section 8.2.3.
14	SDPDIR[2]	0b	SDP2 Pin – Initial Direction This bit configures the initial hardware value of the *SDP2_IODIR* bit in the Extended Device Control (CTRL_EXT) register following power up. See See Section 8.2.3 .
13	PHY_in_LAN_disable	0b	Determines the behavior of the MAC and PHY when a LAN port is disabled through an external pin. 0b = MAC and PHY are kept functional in LAN disable (to support manageability). 1b = MAC and PHY are powered down in LAN disable (manageability cannot access the network through this port). Reflected in EEDIAG (See Section 8.4.5).
12	Disable 100 in non-D0a	0b	Disables 1000Mb/s and 100 Mb/s operation in non-D0a states (See Section 3.5.7.5.4). Sets default value of PHPM.Disable 100 in non-D0a bit.
11	LAN_DIS[1]	0b	LAN Disable In LAN ports 1,2 and 3, when set to 1b, the appropriate LAN is disabled (both PCIe function and LAN access for manageability are disabled). Note: Should be cleared to 0 for LAN port 0.
10	LAN_PCI_DIS[1]	0b	LAN PCIe Function Disable In LAN ports 1,2 and 3, when set to 1b, the appropriate LAN PCI function is disabled. For example, in the case were the LAN is functional for manageability operation but is not connected to the host through the PCIe interface. Reflected in EEDIAG (See Section 8.4.5). Note: 1. Bit has no effect on LAN port 0 and is Reserved with a value of 0b. 2. When bit is set, Function is in a Non-D0a (uninitialized) state. As a result if the *LPLU*, *Disable 1000 in Non-D0a* or *Disable 100 in Non-D0a* EEPROM bits are set, Management might operate with reduced link speed.
9	SDPDIR[1]	0b	SDP1 Pin – Initial Direction This bit configures the initial hardware value of the *SDP1_IODIR* bit in the Device Control (CTRL) register following power up. See Section 8.2.1 .
8	SDPDIR[0]	0b	SDP0 Pin – Initial Direction This bit configures the initial hardware value of the *SDP0_IODIR* bit in the Device Control (CTRL) register following power up. See Section 8.2.1 .

Bit	Name	Default in EEPROM-less mode	Description
7	SDPVAL[3]	0b	SDP3 Pin – Initial Output Value This bit configures the initial power-on value output on SDP3 (when configured as an output) by configuring the initial hardware value of the *SDP3_DATA* bit in the Extended Device Control (CTRL_EXT) register after power up. See Section 8.2.3 .
6	SDPVAL[2]	0b	SDP2 Pin – Initial Output Value This bit configures the initial power-on value output on SDP2 (when configured as an output) by configuring the initial hardware value of the *SDP2_DATA* bit in the Extended Device Control (CTRL_EXT) register after power up. See Section 8.2.3 .
5	WD_SDP0	0b	When set, SDP[0] is used as a watchdog timeout indication. When reset, it is used as an SDP (as defined in bits 8 and 0). See Section 8.2.1 .
4	Giga Disable	0b	When set, GbE operation is disabled. A usage example for this bit is to disable GbE operation if system power limits are exceeded (See Section 3.5.7.5.4).
3	Disable 1000 in non-D0a	0b	Disables 1000 Mb/s operation in non-D0a states (See Section 3.5.7.5.4).
2	ADVD3WUC	1b	Controls reporting of D3 Cold wake-up support in the *Power Management Capabilities* (*PMC*) configuration register. See Section 9.5.1.3. In addition bit is loaded to CTRL.ADVD3WUC (See Section 8.2.1). When set, D3Cold wake up capability is advertised based on whether AUX_PWR pin is connected to 3.3V to advertise presence of auxiliary power (yes if AUX_PWR is indicated, no otherwise). When 0b, however, D3Cold wake up capability is not advertised even if AUX_PWR presence is indicated. If full 1Gb/sec. operation in D3 state is desired but the system's power requirements in this mode would exceed the D3Cold Wake up-Enabled specification limit (375mA at 3.3V), this bit can be used to prevent the capability from being advertised to the system.
1	SDPVAL[1]	0b	SDP1 Pin – Initial Output Value This bit configures the initial power-on value output on SDP1 (when configured as an output) by configuring the initial hardware value of the *SDP1_DATA* bit in the Device Control (CTRL) register after power up. See See Section 8.2.1 .
0	SDPVAL[0]	0b	SDP0 Pin – Initial Output Value This bit configures the initial power-on value output on SDP0 (when configured as an output) by configuring the initial hardware value of the *SDP0_DATA* bit in the Device Control (CTRL) register after power up. See See Section 8.2.1 .

1. Function 0 can not be disabled via EEPROM.

6.2.19 Functions Control (Word 0x21)

Bit	Name	Default in EEPROM-less mode	Description
15	NC-SI Clock Pad Drive Strength	0b	Defines the drive strength of the NCSI_CLK_OUT pad. If set, the driving strength is stronger. See Section 11.6.1.4 for details. Reflected in EEDIAG (See Section 8.4.5).
14	NC-SI Data Pad Drive Strength	0b	Defines the drive strength of the NCSI_CRS_DV and NCSI_RXD pads. If set, the driving strength is stronger. See Section 11.6.1.4 for details. Reflected in EEDIAG (See Section 8.4.5).
13	NC-SI Output Clock Disable	0b	If set, the clock source is external. In this case, the NCSI_CLK_OUT pad is kept stable at zero and the NCSI_CLK_IN pad is used as an input source of the clock. If cleared, the 82580 outputs the NC-SI clock through the NCSI_CLK_OUT pad. The NCSI_CLK_IN pad is still used as a NC-SI clock input. If NC-SI is not used, then this bit should be set. If this bit is cleared, the Device Power Down Enable bit in word 0x1E (bit 15) should not be set. Reflected in EEDIAG (See Section 8.4.5).
12	LAN Function Sel	0b	LAN Function Select When all LAN ports are enabled and LAN Function Sel = 0b, LAN 0 is routed to PCI Function 0, LAN 1 is routed to PCI Function 1, etc. If LAN Function Sel = 1b, LAN 0 is routed to PCI Function 3, LAN 1 is routed to PCI Function 2, etc. This bit is Mapped to FACTPS[30]. See Section 8.6.9).
11	Reserved	0b	Reserved
10	BAR32	1b	Bit (loaded to the BARCTRL register) preserves the legacy 32 bit BAR mode when BAR32 is set. When cleared to 0b 64 bit BAR addressing mode is selected. Note: If *PREFBAR* is set the *BAR32* bit should always be 0 (64 bit BAR addressing mode). See Section 9.4.11.
9	PREFBAR	0b	0 = BARs are marked as non prefetchable 1 = BARs are marked as prefetchable Note: The 82580 implements Non-prefetchable space in memory BAR, since it has read side effects. This bit is loaded from the *PREFBAR* bit in the EEPROM. See Section 9.4.11.
8:1	Reserved	00100000b	Reserved
0	NC-SI Slew Rate	1b	Defines the slew rate of the NCSI_CLK_OUT, NCSI_CRS_DV and NCSI_RXD pads. If set, the slew rate is high.

6.2.20 LAN Power Consumption (Word 0x22)

Bit	Name	Default in EEPROM-less mode	Description
15:8	LAN D0 Power	0x0	The value in this field is reflected in the PCI Power Management Data Register of the LAN functions for D0 power consumption and dissipation (*Data_Select* = 0 or 4). Power is defined in 100mW units. The power includes also the external logic required for the LAN function. See Section 9.5.1.4.
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. See Section 9.5.1.4.
4:0	LAN D3 Power	0x0	The value in this field is reflected in the PCI Power Management Data register of the LAN functions for D3 power consumption and dissipation (*Data_Select* = 3 or 7). Power is defined in 100 mW units. The power also includes the external logic required for the LAN function. The MSBs in the data register that reflects the power values are padded with zeros. See Section 9.5.1.4.

6.2.21 Initialization Control 3 (LAN Base Address + Offset 0x24)

These words control general initialization values of LAN 0, LAN 1, LAN 2 and LAN 3 ports.

Bit	Name	Default in EEPROM-less mode	Description
15	SerDes Energy Source	0b	SerDes Energy Source Detection When set to 0b, source is internal SerDes Rx circuitry for electrical idle or link-up indication. When set to 1b, source is external SRDS_[n]_SIG_DET signal for electrical idle or Link-up indication. This bit also indicates the source of the signal detect while establishing a link in SerDes mode. This bit sets the default value of the *CONNSW.ENRGSRC* bit. See Section 8.2.7.
14	2 wires SFP Enable	0b	2 wires SFP interface *enable* - bit is used to enable interfacing an external PHY either VIA the MDIO or I^{2}C interface 0b = Disabled. When disabled, the 2 wires I/F pads are isolated. 1b = Enabled. Used to set the default value of *CTRL_EXT.I2C Enabled*. See See Section 8.2.3 .
13	ILOS	0b	Default setting for the loss-of-signal polarity bit (CTRL[7]). See Section 8.2.1 .
12:11	Interrupt Pin	00b LAN 0 01b LAN 1 10b LAN 2 11b LAN3	Controls the value advertised in the *Interrupt Pin* field of the PCI Configuration header for this device/function. The encoding of this field is as follow: **Value** **INT Line** ***Interrupt Pin* Field Value** 00b INTA 1 01b INTB 2 10b INTC 3 11b INTD 4 If only a single device/function of the 82580 component is enabled, this value is ignored and the *Interrupt Pin* field of the enabled device reports INTA# usage. See Section 9.4.18.
10	APM Enable	0b	Initial value of *Advanced Power Management Wake Up Enable* bit in the Wake Up Control (WUC.APME) register. Mapped to CTRL[6] and to WUC[0]. see See Section 8.2.1 and See Section 8.21.1 .

Bit	Name	Default in EEPROM-less mode	Description
9	Reserved		Reserved
8	Reserved		Reserved
7	LAN Boot Disable	1b	A value of 1b disables the expansion ROM BAR in the PCI configuration space.
5:4	Link Mode	00b	Initial value of *Link Mode* bits of the Extended Device Control (*CTRL_EXT.LINK_MODE*) register, specifying which link interface and protocol is used by the MAC. 00b = MAC operates with internal copper PHY (10/100/1000Base-T). 01b = MAC and SerDes I/F operate in 1000BASE-KX mode. 10b = MAC and SerDes operate in SGMII mode. 11b = MAC and SerDes I/F operate in SerDes (1000BASE-BX) mode. See Section 8.2.3 .
6	Reserved		Reserved
3	Com_MDIO	0b	When this bit is set, access to the MDIO interface on this port is routed to the LAN 0 MDIO interface. In this case the LAN 0 MDIO interface is used as a common interface for MDIO accesses to a Multi PHY device that has a single MDIO interface for all PHYs. 0b - MDIO access routed to the dedicated LAN port MDIO interface. 1b - MDIO accesses on this LAN port are routed to the LAN 0 MDIO interface Used to set the default value of MDICNFG.Com_MDIO bit. See Section 8.2.5.
2	External MDIO	0b	When set PHY management interface is via external MDIO interface. Loaded to MDICNFG.Destination. See Section 8.2.5.
1	EXT_VLAN	0b	Sets the default for CTRL_EXT[26] bit. Indicates that additional VLAN is expected in this system. See Section 8.2.3 .
0	Keep_PHY_Link_ Up_En	0b	Enables *No PHY Reset* when the Baseboard Management Controller (BMC) indicates that the PHY should be kept on. When asserted, this bit prevents the PHY reset signal and the power changes reflected to the PHY according to the *MANC.Keep_PHY_Link_Up* value. Note: This EEPROM bit should be set to the same value for all LAN ports.

6.2.22 PCIe Control 2 (Word 0x28)

This word is used to configure initial settings for the PCIe default functionality.

Bits	Name	Default in EEPROM-less mode	Description
15:13	Reserved		Reserved
12	ECRC Check	1b	Loaded into the ECRC Check Capable bit of the PCIe configuration registers 0b - Function is not capable of checking ECRC 1b - Function is capable of checking ECRC
11	ECRC Generation	1b	Loaded into the ECRC Generation Capable bit of the PCIe configuration registers. 0b - Function is not capable of generating ECRC 1b - Function is capable of generating ECRC
10	FLR capability enable	1b	FLR capability Enable bit is loaded to “PCIe configuration registers” -> “Device Capabilities”.

Bits	Name	Default in EEPROM-less mode	Description
9:6	FLR delay	0x1	Delay in microseconds from D0 to D3 move till reset assertion.
5	FLR delay disable	0b	FLR delay disable. 0 - Add delay to FLR assertion. 1 - Do not add delay to FLR assertion.
4:0	Reserved	0x0	Reserved

6.2.23 PCIe Control 3 (Word 0x29)

This word is used for programming PCIe functionality.

Bits	Name	Default in EEPROM-less mode	Description
15:7	Reserved	0x0	Reserved
6	Reserved	0b	Reserved
5	Wake_pin_enable	0b	Enables the use of the wake pin for a PME event in all non L2 power states.
4	DIS Clock Gating in DISABLE	1b	Disable clock gating when LTSSM is at DISABLE state.
3	DIS Clock Gating in L2	1b	Disable clock gating when LTSSM is at L2 state
2	DIS Clock Gating in L1	1b	Disable clock gating when LTSSM is at L1 state
1:0	Reserved		Reserved

6.2.24 End of Read-Only (RO) Area (Word 0x2C)

Defines the end of the area in the EEPROM that is RO.

Bit	Name	Description
15	Reserved	Reserved
14:0	EORO_area	Defines the end of the area in the EEPROM that is RO. The resolution is one word and can be up to byte address 0xFFFF (0x7FFF words). A value of zero indicates no RO area.

6.2.25 Start of RO Area (Word 0x2D)

Defines the start of the area in the EEPROM that is RO.

Bit	Name	Description
15	Reserved	Reserved
14:0	SORO_area	Defines the start of the area in the EEPROM that is RO. The resolution is one word and can be up to byte address 0xFFFF (0x7FFF words).

6.2.26 Watchdog Configuration (Word 0x2E)

Bit	Name	Default in EEPROM-less mode	Description
15	Watchdog Enable	0b	Enable watchdog interrupt. See Section 8.16.1. ***Note:*** If bit is set to 1b value of EEPROM *Watchdog Timeout* field should be 2 or higher to avoid immediate generation of a watchdog interrupt.
14:11	Watchdog Timeout	0x0	Watchdog timeout period (in seconds). See Section 8.16.1. Note: Loaded to 4 LSB bits of *WDSTP.WD_Timeout* field.
10:0	Reserved		Reserved

6.2.27 VPD Pointer (Word 0x2F)

This word points to the Vital Product Data (VPD) structure. This structure is available for the NIC vendor to store it's own data. A value of 0xFFFF indicates that the structure is not available.

Bit	Name	Description
15:0	VPD offset	Offset to VPD structure in words. Bit 15 must be set to 0b (VPD area must be in the first 32 Kbytes of the EEPROM).

6.3 CSR Auto Configuration Pointer (LAN Base Address + Offset 0x17)

Word points to the CSR auto configuration structures of LAN 0, LAN 1, LAN 2 and LAN3. Sections are loaded during HW auto-load as described in Section 3.3.1.3. If no CSR autoload is required for the specific LAN port, the word shall be set to 0xFFFF.

The CSR Auto Configuration structure format is listed in the following tables.

Table 6-4. CSR Auto Configuration Structure Format

Offset	High Byte[15:8]	Low Byte[7:0]	Section
0x0	Section Length = 3*n (n – number of CSRs to configure)		Section 6.3.1
0x1	Block CRC8		Section 6.3.2
0x2	CSR Address		Section 6.3.3
0x3	Data LSB		Section 6.3.4
0x4	Data MSB		Section 6.3.5
	...		
3*n - 1	CSR Address		Section 6.3.3
3*n	Data LSB		Section 6.3.4
3*n + 1	Data MSB		Section 6.3.5

6.3.1 CSR Configuration Section Length - Offset 0x0

The section length word contains the length of the section in words. Note that section length count does not include the section length word and Block CRC8 word.

Bits	Name	Default	Description
15	Reserved		
14:0	Section_length		Section length in words.

6.3.2 Block CRC8 (Offset 0x1)

Bit	Name	Description
15:8	Reserved	
7:0	CRC8	

6.3.3 CSR Address - (Offset 3*n - 1; [n = 1... Section Length])

Bits	Name	Default	Description
15	Reserved		
14:0	CSR_ADDR		CSR Address in Double Words (4 bytes)

6.3.4 CSR Data LSB - (Offset 3*n; [n = 1... Section Length])

Bits	Name	Default	Description
15:0	CSR_Data_LSB		CSR Data LSB

6.3.5 CSR Data MSB - (Offset 3*n + 1; [n = 1... Section Length])

Bits	Name	Default	Description
15:0	CSR_Data_MSB		CSR Data MSB

6.4 CSR Auto Configuration Power-Up Pointer (LAN Base Address + Offset 0x27)

This word points to the CSR auto configuration Power-Up structures of LAN 0, LAN 1, LAN 2 and LAN3. Sections are loaded during HW auto-load as described in Section 3.3.1.3. If no CSR autoload is required for the specific LAN port, the word shall be set to 0xFFFF.

The CSR Auto Configuration Power-Up structure format is listed in the following tables.

Table 6-5. CSR Auto Configuration Power-Up Structure Format

Offset	High Byte[15:8]	Low Byte[7:0]	Section
0x0	Section Length = 3*n (n – number of CSRs to configure)		Section 6.4.1
0x1	Block CRC8		Section 6.4.2
0x2	CSR Address		Section 6.4.3
0x3	Data LSB		Section 6.4.4
0x4	Data MSB		Section 6.4.5

Table 6-5. CSR Auto Configuration Power-Up Structure Format

Offset	High Byte[15:8]	Low Byte[7:0]	Section
	...		
3*n - 1	CSR Address		Section 6.4.3
3*n	Data LSB		Section 6.4.4
3*n + 1	Data MSB		Section 6.4.5

6.4.1 CSR Configuration Power-Up Section Length - Offset 0x0

The section length word contains the length of the section in words. Note that section length count does not include the section length word and Block CRC8 word.

Bits	Name	Default	Description
15	Reserved		
14:0	Section_length		Section length in words.

6.4.2 Block CRC8 (Offset 0x1)

Bit	Name	Description
15:8	Reserved	
7:0	CRC8	

6.4.3 CSR Address - (Offset 3*n - 1; [n = 1... Section Length])

Bits	Name	Default	Description
15	Reserved		
14:0	CSR_ADDR		CSR Address in Double Words (4 bytes)

6.4.4 CSR Data LSB - (Offset 3*n; [n = 1... Section Length])

Bits	Name	Default	Description
15:0	CSR_Data_LSB		CSR Data LSB

6.4.5 CSR Data MSB - (Offset 3*n + 1; [n = 1... Section Length])

Bits	Name	Default	Description
15:0	CSR_Data_MSB		CSR Data MSB

6.5 PCIe PHY Auto Configuration Pointer (Word 0x10)

This word points to the PCIe PHY auto configuration structure used to configure PCIe PHY related circuitry. Sections are loaded during HW auto-load as described in Section 3.3.1.3. If default values do not need to be modified, the word shall be set to 0xFFFF.

The PCIe PHY Auto Configuration structure format is listed in the following table.

Table 6-6. PCIe PHY Auto Configuration Structure format

Offset	High Byte[15:8]	Low Byte[7:0]	Section
0x0	Section Length = 2*n (n – number of registers to configure)		Section 6.5.1
0x1	Block CRC8		Section 6.5.3
0x2	Register Address		Section 6.5.3
0x3	Data		Section 6.5.4
	...		
2*n	Register Address		Section 6.5.3
2*n + 1	Data		Section 6.5.4

6.5.1 PCIe PHY Configuration Section Length - Offset 0x0

The section length word contains the length of the section in words. Note that section length count does not include the section length word and Block CRC8 word.

Bits	Name	Default	Description
15	Reserved		
14:0	Section_length		Length in words of register write section (2 * registers to be written).

6.5.2 Block CRC8 (Offset 0x1)

Bit	Name	Description
15:8	Reserved	
7:0	CRC8	

6.5.3 Register Address - (Offset 2*n; [n = 1... Number of Registers to be Written])

Bits	Name	Default	Description
15:0	Reg_ADDR		PCIe PHY Register Address in Words (2 bytes)

6.5.4 Register Data - (Offset 2*n + 1; [n = 1... Number of Registers to be Written])

Bits	Name	Default	Description
15:0	Reg_Data		CSR Data

6.5.4.1 Setting Default PCIe Link Width and Link Speed

By default, the 82580 PCIe interface is configured to a maximum Link speed of 5.0 Gb/s and Link Width of 4 lanes. Default link width and speed can be modified by programming the Internal *PCIe Link Configuration Register* using the PCIe PHY Configuration data EEPROM Section.

6.5.4.2 PCIe Link Configuration Register Address - Offset 0x2

Bits	Field	Address	Description
15:0	Reg_ADDR	0x94	PCIe Link Configuration Register Address in Words (2 bytes)

6.5.4.3 PCIe Link Configuration Register Data - Offset 0x3

Placing a value of 0x0 in the *PCIe Link Configuration Register Data* EEPROM word configures the 82580 PCIe interface to PCIe Gen 2 link rates and a link width of 4.

Bits	Field	Default in EEPROM-less mode	Description
15:9	Reserved	0x0	Reserved. Value should be 0.
8	Disable PCIe Gen 2	0b	1b - 2.5 Gb/s Link speed supported 0b - 5.0 Gb/s and 2.5 GT/s Link speeds supported
7:4	Reserved	0x0	Reserved. Value should b 0.
3	Disable Lane 3	0b	0b - Lane 3 enabled 1b - Lane 3 disabled
2	Disable Lane 2	0b	0b - Lane 2 enabled 1b - Lane 2 disabled
1	Disable Lane 1	0b	0b - Lane 1 enabled 1b - Lane 1 disabled
0	Disable Lane 0	0b	0b - Lane 0 enabled 1b - Lane 0 disabled

6.5.4.4 PCIe Link Power Down Register Address - Offset 0x4

Bits	Field	Address	Description
15:0	Reg_ADDR	0x96	PCIe Link Configuration Register Address in Words (2 bytes)

6.5.4.5 PCIe Link Power Down Register Data - Offset 0x5

Bits	Field	Default in EEPROM-less mode	Description
15:12	Reserved	0x0	Reserved. Value should be 0.
11:8	Latency_To_Enter_L0s	0x3	Sets Latency in µs between PCIe Idle detection and entry to L0s when *PCIEMISC.Lx_decision* bit is 0.
7:5	Reserved	0x0	Reserved. Value should be 0.
4:0	Latency_To_Enter_L1	0x6	Sets Latency in µs between entry to L0s and entry to L1 when *PCIEMISC.Lx_decision* bit is 0.

6.6 Firmware Pointers and Control Words

Word 0x11 in each LAN port EEPROM section are used to point to structures specific to Pass through operation. Words 0x54 and 0x23 control some aspects of the Firmware functionality.

Word 0x51 is used to point to the load Firmware patch structure.

A value of 0xFFFF for a pointer indicates that the relevant structure is not present in the EEPROM.

6.6.1 Pass Through LAN Configuration Pointer (LAN Base Address + Offset 0x11)

Bit	Name	Description
15:0	Pointer	Pointer to the PT LAN Configuration Structure. See Section 6.8 for details of the structure.

6.6.2 PHY Configuration Pointer (Word 0x50)

Bit	Name	Description
15:0	Pointer	Pointer to PHY configuration structure. See Section 6.9 for details of the structure. A value of 0xFFFF means the pointer is invalid.

6.6.3 Firmware Patch Pointer (Word 0x51)

Bit	Name	Description
15:0	Pointer	Pointer to loader patch structure. See Section 6.7 for details of the structure.

6.6.4 Sideband Configuration Pointer (Word 0x57)

Bit	Name	Description
15:0	Pointer	Pointer to the Sideband configuration pointer structure. See Section 6.10 for details of the structure.

6.6.5 Reserved (Word 0x5E)

Bit	Name	Description
15:0	Pointer	Value should be 0xFFFF

6.6.6 Manageability Capability/Manageability Enable (Word 0x54)

Bit	Name	Description
15	Reserved	Reserved
14	Redirection Sideband Interface	0b = SMBus. 1b = NC-SI.
13:12	Reserved	Reserved.
11	MCSR_TO_RETRY	0b - On CSR access Timeout return failed access to BMC. 1b - On CSR access Timeout retry access internally and don't return failed access. Default value - 1b
10:8	Manageability Mode	0x0 = None. 0x1 = Reserved. 0x2 = Pass Through (PT) mode. 0x3 = Reserved. 0x4 = Host interface enable only. 0x5:0x7 = Reserved.
7	Port 3 Manageability Capable	0 = Not capable 1 = Bit 3 applicable to port 3.
6	Port 2 Manageability Capable	0 = Not capable 1 = Bit 3 applicable to port 2.
5	Port 1 Manageability Capable	0 = Not capable 1 = Bit 3 applicable to port 1.
4	Port 0 Manageability Capable	0 = Not capable 1 = Bit 3 applicable to port 0.
3	Pass Through Capable	0b = Disable. 1b = Enable.
2:0	Reserved	Reserved

6.6.7 Management HW Config Control (Word 0x23)

This word contains bits that configure special firmware behavior.

Bit	Name	Description
15	LAN3_FTCO_RST_DIS	LAN3 force TCO reset disable (1b disable; 0b enable).
14	LAN2_FTCO_RST_DIS	LAN2 force TCO reset disable (1b disable; 0b enable).
13	LAN1_FTCO_RST_DIS	LAN1 force TCO reset disable (1b disable; 0b enable).
12	LAN0_FTCO_RST_DIS	LAN0 force TCO reset disable (1b disable; 0b enable).
11	Multi-Drop NC-SI	Multi-Drop NC-SI topology. 0 - Point-to-point (default) 1 - Multi-drop When bit is set the NCSI_CRS_DV and NCSI_RXD[1:0] pins are High-Z Following power-up, otherwise the pins are driven.
10	PARITY_ERR_RST_EN	When set enables reset of Management logic and generation of internal Firmware reset as a result of Parity Error detected in Management memories. Note: Bit should be set to 1.

Bit	Name	Description
9	Enable All Phys in D3	0 - Only ports activated for BMC activity will stay active in D3: In NCSI mode – according to enable channel command to the specific port. In SMBUS mode – according to EEPROM port enable bit. 1- All PHYs will stay active in D3.
8:4	Reserved	Reserved
3	LAN3_FTCO_ISOL_DIS	LAN3 force TCO Isolate disable (1b disable; 0b enable).
2	LAN2_FTCO_ISOL_DIS	LAN2 force TCO Isolate disable (1b disable; 0b enable).
1	LAN1_FTCO_ISOL_DIS	LAN1 force TCO Isolate disable (1b disable; 0b enable).
0	LAN0_FTCO_ISOL_DIS	LAN0 force TCO Isolate disable (1b disable; 0b enable).

6.7 Firmware Patch Structure

This structure is used for all FW patches. Firmware patch Pointer (Word 0x51) points to the start (offset 0x0) of this kind of structure. If pointer is 0xFFFF then no structure exists. Structure is loaded during HW EEPROM auto-load as described in Section 3.3.1.3.

6.7.1 Firmware Patch Data Size (Offset 0x0)

The Data Size word contains the length of the section in words. Note that Data Size count does not include the section length word and Block CRC8 word.

Bit	Name	Description
15:0	Data Size (Words)	

6.7.2 Block CRC8 (Offset 0x1)

Bit	Name	Description
15:8	Reserved	
7:0	CRC8	

6.7.3 Patch Ram Address Word (Offset 0x2)

Bit	Name	Description
15:0	Ram address	Ram Address to load patch.

6.7.4 Patch Version 1 Word (Offset 0x3)

Bit	Name	Description
15:8	Patch Generation Hour	
7:0	Patch Generation Minutes	

6.7.5 Patch Version 2 Word (Offset 0x4)

Bit	Name	Description
15:8	Patch Generation Month	
7:0	Patch Generation Day	

6.7.6 Patch Version 3 Word (Offset 0x5)

Bit	Name	Description
15:8	Patch Silicon Version Compatibility	0x00 = A0. 0x01 = A1. 0x10 = B0. 0x11 = B1.
7:0	Patch Generation Year	

6.7.7 Patch Version 4 Word (Offset 0x6)

Bit	Name	Description
15:8	Patch Major Number	
7:0	Patch Minor Number	

6.7.8 Patch Data Words (Offset 0x7, Block Length)

Bit	Name	Description
15:0	Patch Firmware Data	

6.8 PT LAN Configuration Structure

The Pass Through LAN Configuration Pointer (LAN Base Address + Offset 0x11) points to the start (offset 0x0) of this type of structure, to configure manageability filters. If pointer is 0xFFFF then no structure exists. Structure is loaded during HW EEPROM auto-load as described in Section 3.3.1.3.

Note: When interface to BMC is via SMBus the PT LAN Configuration Structure must be used to configure the required filters. When Interface is NC-SI, structure should be empty and value of Pass Through LAN Configuration Pointer (Offset 0x11) should be 0XFFFF.

The PT LAN Configuration Structure format is listed in the following tables.

Table 6-7. PT LAN Configuration Structure Format

Offset	High Byte[15:8]	Low Byte[7:0]	Section
0x0	Section Length = 3*n (n – number of CSRs to configure)		Section 6.8.1
0x1	Block CRC8		Section 6.8.2
0x2	CSR Address		Section 6.8.3
0x3	Data LSB		Section 6.8.4

Table 6-7. PT LAN Configuration Structure Format

Offset	High Byte[15:8]	Low Byte[7:0]	Section
0x4	Data MSB		Section 6.8.5
	...		
3*n - 1	CSR Address		Section 6.8.3
3*n	Data LSB		Section 6.8.4
3*n + 1	Data MSB		Section 6.8.5

6.8.1 PT LAN Configuration Structure Section Length - Offset 0x0

The section length word contains the length of the section in words. Note that section length count does not include the section length word and Block CRC8 word.

Bits	Name	Default	Description
15	Reserved		
14:0	Section_length		Section length in words.

6.8.2 Block CRC8 (Offset 0x1)

Bit	Name	Description
15:8	Reserved	
7:0	CRC8	

6.8.3 CSR Address - (Offset 2*n; [n = 1... Section Length])

Bits	Name	Default	Description
15	Reserved		
14:0	CSR_ADDR		CSR Address in Double Words (4 bytes)

6.8.4 CSR Data LSB - (Offset 0x1 + 2*n; [n = 1... Section Length])

Bits	Name	Default	Description
15:0	CSR_Data_LSB		CSR Data LSB

6.8.5 CSR Data MSB - (Offset 0x2 + 2*n; [n = 1... Section Length])

Bits	Name	Default	Description
15:0	CSR_Data_MSB		CSR Data MSB

6.8.6 Manageability Filters

Following table lists registers that can be programmed via the PT LAN Configuration structure.

Name	Description	Section
MAVTV	Management VLAN TAG Value	Section 8.22.1
MFUTP	Management Flex UDP/TCP Ports	Section 8.22.2
METF	Management Ethernet Type Filters	Section 8.22.3
MNGONLY	Management Only Traffic Register	Section 8.22.5
MDEF	Manageability Decision Filters	Section 8.22.6
MDEF_EXT	Manageability Decision Filters Extension	Section 8.22.7
MIPAF	Manageability IP Address Filter registers (IPv4 or IPV6) Note: Can be used to filter IPV4 Address of ARP packets.	Section 8.22.8
MMAL	Manageability MAC Address Low Registers	Section 8.22.9
MMAH	Manageability MAC Address High Registers	Section 8.22.10
FTFT	Flexible TCO Filter Table registers	Section 8.22.11

6.9 PHY configuration Structure

This section describes the PHY auto configuration structure used to configure PHY related circuitry. The programming in this section is applied after each PHY reset. After the registers in this section are written to the PHY, the relevant *EEMNGCTL.CFG_DONE* bit is set.

The PHY Configuration Pointer (Word 0x50) points to the start (offset 0x0) of this type of structure, to configure PHY registers (Internal and External PHYs). If pointer is 0xFFFF then no structure exists. Structure is loaded during HW EEPROM auto-load as described in Section 3.3.1.3.

Table 6-8. PHY Auto Configuration Structure format

Offset	High Byte[15:8]	Low Byte[7:0]	Section
0x0	Section Length = 2*n (n – number of registers to configure)		Section 6.9.1
0x1	Block CRC8		Section 6.9.2
0x2	PHY number and PHY address		Section 6.9.3
0x3	PHY data (MDIC[15:0] or I2CCMD[15:0])		Section 6.9.4
	…		
2*n	PHY number and PHY address		Section 6.9.3
2*n + 1	PHY data (MDIC[15:0] or I2CCMD[15:0])		Section 6.9.4

6.9.1 PHY Configuration Section Length - Offset 0x0

The section length word contains the length of the section in words. Note that section length count does not include the section length word and Block CRC8 word.

Bits	Name	Default	Description
15	Reserved		
14:0	Section_length		Section length in words.

6.9.2 Block CRC8 (Offset 0x1)

Bit	Name	Description
15:8	Reserved	
7:0	CRC8	

6.9.3 PHY number and PHY address - (Offset 2*n; [n = 1... Section Length])

Bits	Name	Default	Description
15:12	Reserved		Reserved
11	Apply to port 3		If set, apply to programming when the PHY of port three is reset.
10	Apply to port 2		If set, apply to programming when the PHY of port two is reset.
9	Apply to port 1		If set, apply to programming when the PHY of port one is reset.
8	Apply to port 0		If set, apply to programming when the PHY of port zero is reset.
7:0	PHY address		PHY address to which the data is written

6.9.4 PHY data (Offset 2*n + 1; [n = 1... Section Length])

Bits	Name	Default	Description
15:0	Reg_Data		MDIC[15:0]/I2CCMD[15:0] value (Data).

6.10 Sideband Configuration Structure

The Sideband Configuration Pointer (Word 0x57) points to the start (offset 0x0) of this structure. If pointer is 0xFFFF then no structure exists.

6.10.1 Section Length (Offset 0x0)

The section length word contains the length of the section in words. Note that section length count does not include the section length word and Block CRC8 word.

Bits	Name	Default	Description
15	Reserved		
14:0	Section_length		Section length in words.

6.10.2 Block CRC8 (Offset 0x1)

Bit	Name	Description
15:8	Reserved	
7:0	CRC8	

6.10.3 SMBus Max Fragment Size (Offset 0x2)

Bit	Name	Description
15:8	Reserved	Reserved
7:0	SMBus Max Fragment Size (Bytes)	Maximum SMBus Fragment Size sent to BMC. Value should be in the 29 to 237 Byte range. Note: Fragment size should be a multiple of 4 + 1 (e.g. 29, 33, 37...).

6.10.4 SMBus Notification Timeout (Offset 0x3)

Bit	Name	Description
15:0	SMBus Notification Timeout	Timeout until decision to discard a packet that was not read by the external BMC. Resolution of field is in 1.043 mSec units (e.g. a value of 8 results in a timeout value of 8.344 mSec) 0b - No discard.

6.10.5 SMBus Slave Address 0 1 (Offset 0x4)

Bit	Name	Description
15:9	SMBus 1 Slave Address	SMBus Slave Address for port 1.
8	Reserved	Reserved.
7:1	SMBus 0 Slave Address	SMBus Slave Address for port 0.
0	Reserved	Reserved.

6.10.6 SMBus Slave Address 2 3 (Offset 0x5)

Bit	Name	Description
15:9	SMBus 3 Slave Address	SMBus Slave Address for port 3.
8	Reserved	Reserved.
7:1	SMBus 2 Slave Address	SMBus Slave Address for port 2.
0	Reserved	Reserved.

6.10.7 NC-SI Configuration (Offset 0x6)

Bit	Name	Description
15:8	Reserved	Reserved.
7:5	Package ID	
4:0	Reserved	Reserved.

6.10.8 Reserved (Offset 0x7 - 0x8)

Reserved words should be 0x0000.

6.10.9 SMBus Flags (Offset 0x9)

Bit	Name	Description
15:14	Reserved	Reserved
13	RX_P3_PAR_EN	Management RX Buffer parity check enable port 3
12	RX_P2_PAR_EN	Management RX Buffer parity check enable port 2
11	RX_P1_PAR_EN	Management RX Buffer parity check enable port 1
10	RX_P0_PAR_EN	Management RX Buffer parity check enable port 0
9	TX_PAR_EN	Management TX Buffer parity check enable
8	CMD_PAR_EN	Management ARC RAM parity check enable (Command RAM)
7:6	Reserved	Reserved
5	SMBus Block Read Command	0b = Block read command is C0. 1b = Block read command is D0.
4:3	Reserved	Reserved
2	Disable SMBus ARP Functionality	
1	SMBus ARP PEC	
0	Reserved	

6.10.10 LAN Receive Enable 3 (Offset 0xA)

Bit	Name	Description
15:4	Reserved	Reserved
3	Enable BMC Dedicated MAC port 3	Configure BMC dedicated MAC Address on Port 3. 0b = The 82580 will share MAC address for MNG traffic with host MAC address on Port 3. Host MAC address is specified in EEPROM words located at LAN Base Address + offset 0x0-0x2 per port (See Section 6.2.1). 1b = The 82580 will use the BMC dedicated MAC address on port 3 (MAC address 1 (MMAL/H1) see Section 8.22.9) as filter for incoming receive packets. Note: MMAL/H registers can be programmed from the EEPROM using the *PT LAN Configuration structure* (See Section 6.8.1).

Bit	Name	Description
2	Enable BMC Dedicated MAC port 2	Configure BMC dedicated MAC Address on Port 2. 0b = The 82580 will share MAC address for MNG traffic with host MAC address on Port 2. Host MAC address is specified in EEPROM words located at LAN Base Address + offset 0x0-0x2 per port (See Section 6.2.1). 1b = The 82580 will use the BMC dedicated MAC address on port 2(MAC address 1(MMAL/H1) see Section 8.22.9) as filter for incoming receive packets. Note: MMAL/H registers can be programmed from the EEPROM using the *PT LAN Configuration structure* (See Section 6.8.1).
1	Enable BMC Dedicated MAC port 1	Configure BMC dedicated MAC Address on Port 1. 0b = The 82580 will share MAC address for MNG traffic with host MAC address on Port 1. Host MAC address is specified in EEPROM words located at LAN Base Address + offset 0x0-0x2 per port (See Section 6.2.1). 1b = The 82580 will use the BMC dedicated MAC address on port 1(MAC address 1 (MMAL/H1) see Section 8.22.9) as filter for incoming receive packets. Note: MMAL/H registers can be programmed from the EEPROM using the *PT LAN Configuration structure* (See Section 6.8.1).
0	Enable BMC Dedicated MAC port 0	Configure BMC dedicated MAC Address on Port 0. 0b = The 82580 will share MAC address for MNG traffic with host MAC address on Port 0. Host MAC address is specified in EEPROM words located at LAN Base Address + offset 0x0-0x2 per port (See Section 6.2.1). 1b = The 82580 will use the BMC dedicated MAC address on port 3 (MAC address 1 (MMAL/H1) see Section 8.22.9) as filter for incoming receive packets. Note: MMAL/H registers can be programmed from the EEPROM using the *PT LAN Configuration structure* (See Section 6.8.1).

6.10.11 LAN0 MANC Value LSB (Offset 0xB)

This value will be stored in the LSB Portion of the LAN0 Management Control (MANC) register. See Section 8.22.4 for register description.

Bit	Name	Description
15	TCO_Isolate	TCO Isolate command. Note: Value placed in this field is not written to MANC register.
14	FW_RESET	FW Reset occurred. Note: Value placed in this field is not written to MANC register.
13:0	Reserved	Reserved.

6.10.12 LAN0 MANC Value MSB (Offset 0xC)

This value will be stored in the MSB Portion of the LAN0 Management Control (MANC) register. See Section 8.22.4 for register description.

Bit	Name	Description
15:11	Reserved	Reserved.
10	NET_TYPE	NET TYPE: 0b = pass only un-tagged packets. 1b = pass only VLAN tagged packets. Valid only if FIXED_NET_TYPE is set.

Bit	Name	Description
9	FIXED_NET_TYPE	Fixed net type: If set, only packets matching the net type defined by the NET_TYPE field passes to manageability. Otherwise, both tagged and un-tagged packets can be forwarded to the manageability engine.
8	EN_IPv4_FILTER	Enable IPv4 address Filters – when set, the last 128 bits of the MIPAF register are used to store 4 IPv4 addresses for IPv4 filtering. When cleared, these bits store a single IPv6 filter.
7	EN_XSUM_FILTER	Enable checksum filtering to MNG When this bit is set, only packets that pass L3, L4 checksum are sent to the MNG block.
6:4	Reserved	Reserved should be 0
3	Reserved	Reserved should be 0
2	KEEP_PHY_LINK_UP	Block PHY reset and power state changes. When this bit is set the PHY reset and power state changes do not reach to the PHY (PHY is not reset), This bit can not be written to, unless the *Keep_PHY_Link_Up_En* EEPROM bit is set.
1	RCV_TCO_EN	Enable BMC to network and network to BMC traffic 0b - The BMC can not communicate with the network. 1b - The BMC can communicate with the network When cleared the BMC traffic is not forwarded to the network and the network traffic is not forwarded to the BMC even if the decision filters indicates it should. This bit does not impact the OS to BMC traffic.
0	TCO_RESET	TCO Reset occurred Note: Value placed in this field is not written to MANC register.

6.10.13 LAN1 MANC Value LSB (Offset 0xD)

This value will be stored in the LSB Portion of the LAN1 Management Control (MANC) register. See Section 8.22.4 for register description.

Bit	Name	Description
15	TCO_Isolate	TCO Isolate command. Note: Value placed in this field is not written to MANC register.
14	FW_RESET	FW Reset occurred. Note: Value placed in this field is not written to MANC register.
13:0	Reserved	Reserved.

6.10.14 LAN1 MANC Value MSB (Offset 0xE)

This value will be stored in the MSB Portion of the LAN1 Management Control (MANC) register. See Section 8.22.4 for register description.

Bit	Name	Description
15:11	Reserved	Reserved.
10	NET_TYPE	NET TYPE: 0b = pass only un-tagged packets. 1b = pass only VLAN tagged packets. Valid only if FIXED_NET_TYPE is set.
9	FIXED_NET_TYPE	Fixed net type: If set, only packets matching the net type defined by the NET_TYPE field passes to manageability. Otherwise, both tagged and un-tagged packets can be forwarded to the manageability engine.

Bit	Name	Description
8	EN_IPv4_FILTER	Enable IPv4 address Filters – when set, the last 128 bits of the MIPAF register are used to store 4 IPv4 addresses for IPv4 filtering. When cleared, these bits store a single IPv6 filter.
7	EN_XSUM_FILTER	Enable checksum filtering to MNG When this bit is set, only packets that pass L3, L4 checksum are sent to the MNG block.
6:4	Reserved	Reserved should be 0
3	Reserved	Reserved should be 0
2	KEEP_PHY_LINK_UP	Block PHY reset and power state changes. When this bit is set the PHY reset and power state changes do not reach to the PHY (PHY is not reset), This bit can not be written to, unless the *Keep_PHY_Link_Up_En* EEPROM bit is set.
1	RCV_TCO_EN	Enable BMC to network and network to BMC traffic The BMC can not communicate with the network. The BMC can communicate with the network When cleared the BMC traffic is not forwarded to the network and the network traffic is not forwarded to the BMC even if the decision filters indicates it should. This bit does not impact the OS to BMC traffic.
0	TCO_RESET	TCO Reset occurred Note: Value placed in this field is not written to MANC register.

6.10.15 LAN2 MANC Value LSB (Offset 0xF)

This value will be stored in the LSB Portion of the LAN2 Management Control (MANC) register. See Section 8.22.4 for register description.

Bit	Name	Description
15	TCO_Isolate	TCO Isolate command. Note: Value placed in this field is not written to MANC register.
14	FW_RESET	FW Reset occurred. Note: Value placed in this field is not written to MANC register.
13:0	Reserved	Reserved.

6.10.16 LAN2 MANC Value MSB (Offset 0x10)

This value will be stored in the MSB Portion of the LAN2 Management Control (MANC) register. See Section 8.22.4 for register description.

Bit	Name	Description
15:11	Reserved	Reserved.
10	NET_TYPE	NET TYPE: 0b = pass only un-tagged packets. 1b = pass only VLAN tagged packets. Valid only if FIXED_NET_TYPE is set.
9	FIXED_NET_TYPE	Fixed net type: If set, only packets matching the net type defined by the NET_TYPE field passes to manageability. Otherwise, both tagged and un-tagged packets can be forwarded to the manageability engine.
8	EN_IPv4_FILTER	Enable IPv4 address Filters – when set, the last 128 bits of the MIPAF register are used to store 4 IPv4 addresses for IPv4 filtering. When cleared, these bits store a single IPv6 filter.

Bit	Name	Description
7	EN_XSUM_FILTER	Enable checksum filtering to MNG When this bit is set, only packets that pass L3, L4 checksum are sent to the MNG block.
6:4	Reserved	Reserved should be 0
3	Reserved	Reserved should be 0
2	KEEP_PHY_LINK_UP	Block PHY reset and power state changes. When this bit is set the PHY reset and power state changes do not reach to the PHY (PHY is not reset), This bit can not be written to, unless the *Keep_PHY_Link_Up_En* EEPROM bit is set.
1	RCV_TCO_EN	Enable BMC to network and network to BMC traffic The BMC can not communicate with the network. The BMC can communicate with the network When cleared the BMC traffic is not forwarded to the network and the network traffic is not forwarded to the BMC even if the decision filters indicates it should. This bit does not impact the OS to BMC traffic.
0	TCO_RESET	TCO Reset occurred Note: Value placed in this field is not written to MANC register.

6.10.17 LAN3 MANC Value LSB (Offset 0x11)

This value will be stored in the LSB Portion of the LAN3 Management Control (MANC) register. See Section 8.22.4 for register description.

Bit	Name	Description
15	TCO_Isolate	TCO Isolate command. Note: Value placed in this field is not written to MANC register.
14	FW_RESET	FW Reset occurred. Note: Value placed in this field is not written to MANC register.
13:0	Reserved	Reserved.

6.10.18 LAN3 MANC Value MSB (Offset 0x12)

This value will be stored in the MSB Portion of the LAN3 Management Control (MANC) register. See Section 8.22.4 for register description.

Bit	Name	Description
15:11	Reserved	Reserved.
10	NET_TYPE	NET TYPE: 0b = pass only un-tagged packets. 1b = pass only VLAN tagged packets. Valid only if FIXED_NET_TYPE is set.
9	FIXED_NET_TYPE	Fixed net type: If set, only packets matching the net type defined by the NET_TYPE field passes to manageability. Otherwise, both tagged and un-tagged packets can be forwarded to the manageability engine.
8	EN_IPv4_FILTER	Enable IPv4 address Filters – when set, the last 128 bits of the MIPAF register are used to store 4 IPv4 addresses for IPv4 filtering. When cleared, these bits store a single IPv6 filter.
7	EN_XSUM_FILTER	Enable checksum filtering to MNG When this bit is set, only packets that pass L3, L4 checksum are sent to the MNG block.

Bit	Name	Description
6:4	Reserved	Reserved should be 0
3	Reserved	Reserved should be 0
2	KEEP_PHY_LINK_UP	Block PHY reset and power state changes. When this bit is set the PHY reset and power state changes do not reach to the PHY (PHY is not reset), This bit can not be written to, unless the *Keep_PHY_Link_Up_En* EEPROM bit is set.
1	RCV_TCO_EN	Enable BMC to network and network to BMC traffic The BMC can not communicate with the network. The BMC can communicate with the network When cleared the BMC traffic is not forwarded to the network and the network traffic is not forwarded to the BMC even if the decision filters indicates it should. This bit does not impact the OS to BMC traffic.
0	TCO_RESET	TCO Reset occurred Note: Value placed in this field is not written to MANC register.

6.11 Software Accessed Words

Words 0x03 to 0x07 in the EEPROM image are reserved for compatibility information. New bits within these fields will be defined as the need arises for determining software compatibility between various hardware revisions.

Words 0x8 and 0x09 are used to indicate the Printed Board Assembly (PBA) number and words 0x42 and 0x43 identifies the EEPROM image.

Words 0x30 to 0x3E have been reserved for configuration and version values to be used by PXE code. The only exceptions are word 0x3D, which is used for the iSCSI boot configuration and word 0x37 used for Pointer to Alternate MAC address.

6.11.1 Compatibility (Word 0x03)

Bit	Description
15:12	Reserved (set to 0000b).
11	LOM/Not a LOM. 0b = NIC. 1b = LOM.
10	Server/Not a Server NIC. 0b = Client. 1b = Server.
9	Client/Not a Client NIC. 0b = Server. 1b = Client.
8	Retail/OEM. 0b = Retail. 1b = OEM.
7:6	Reserved (set to 00b).
5	Reserved (set to 1b).
4	SMBus Connected. 0b = Not connected. 1b = Connected.
3	Reserved (set to 0b).
2	PCI Bridge/No PCI Bridge. 0b = PCI bridge not present. 1b = PCI bridge present.
1:0	Reserved (set to 00b)

6.11.2 OEM specific (Word 0x04)

Bit	Description
15:12	Control for LED 3. 0001b = Default in STATE1 + Default in STATE2. 0010b = Default in STATE1 + LED is ON in STATE2. 0011b = Default in STATE1 + LED is OFF in STATE2. 0100b = LED is ON in STATE1 + Default in STATE2. 0101b = LED is ON in STATE1 + LED is ON in STATE2. 0110b = LED is ON in STATE1 + LED is OFF in STATE2. 0111b = LED is OFF in STATE1 + Default in STATE2. 1000b = LED is OFF in STATE1 + LED is ON in STATE2. 1001b = LED is OFF in STATE1 + LED is OFF in STATE2.
11:8	Control for LED 2 – same encoding as for LED 3.
7:4	Control for LED 1 – same encoding as for LED 3.
3:0	Control for LED 0 – same encoding as for LED 3.

6.11.3 OEM Specific (Word 0x06, 0x07)

These words are available for OEM use.

6.11.4 EEPROM Image Revision (Word 0x05)

This word is valid only for device starter images and indicates the ID and version of the EEPROM image.

Bit	Description
15:12	EEPROM major version.
11:4	EEPROM minor version.
3:0	EEPROM image ID.

6.11.5 PBA Number (Word 0x08, 0x09)

The nine-digit Printed Board Assembly (PBA) number used for Intel manufactured NICs that are stored in a four-byte field. The dash itself is not stored; neither is the first digit of the 3-digit suffix, as it is always zero for the affected products. Note that through the course of hardware ECOs, the suffix field (byte 4) increments. The purpose of this information is to allow customer support (or any user) to identify the exact revision level of a product. Network driver software should not rely on this field to identify the product or its capabilities.

Note: This PBA number is not related to the MSI-X Pending Bit Array (PBA).

Product	PWA Number	Byte 1	Byte 2	Byte 3	Byte 4
Example	123456-003	12	34	56	03

6.11.6 PXE Configuration Words (Word 0x30:3B)

PXE configuration is controlled by the words.

6.11.6.1 Setup Options PCI Function 0 (Word 0x30)

The main setup options are stored in word 0x30. These options are those that can be changed by the user via the Control-S setup menu. Word 0x30 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.
9	RFU	Reserved. Must be 0.

Bit(s)	Name	Function
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	IBD	iSCSI Boot Disable
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 0x31 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.

6.11.6.2 Configuration Customization Options PCI Function 0 (Word 0x31)

Word 0x31 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.

Bit(s)	Name	Function
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.

6.11.6.3 PXE Version (Word 0x32)

Word 0x32 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.

6.11.6.4 IBA Capabilities (Word 0x33)

Word 0x33 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.

6.11.6.5 Setup Options PCI Function 1 (Word 0x34)

This word is the same as word 0x30, but for function 1 of the device.

6.11.6.6 Configuration Customization Options PCI Function 1 (Word 0x35)

This word is the same as word 0x31, but for function 1 of the device.

6.11.6.7 Setup Options PCI Function 2 (Word 0x38)

This word is the same as word 0x30, but for function 2 of the device.

6.11.6.8 Configuration Customization Options PCI Function 2 (Word 0x39)

This word is the same as word 0x31, but for function 2 of the device.

6.11.6.9 Setup Options PCI Function 3 (Word 0x3A)

This word is the same as word 0x30, but for function 3 of the device.

6.11.6.10 Configuration Customization Options PCI Function 3 (Word 0x3B)

This word is the same as word 0x31, but for function 3 of the device.

6.11.7 iSCSI Boot Configuration Pointer (Word 0x3D)

Bit	Name	Description
15:0	iSCSI Address	iSCSI Configuration Block EEPROM Offset Offset of iSCSI configuration block from the start of the EEPROM. If set to 0000h or FFFFh there is no EEPROM configuration data available for the iSCSI adapter. In this case configuration data must be provided by the BIOS through the SM CLP interface.

6.11.7.1 iSCSI Module Structure

The table below defines the layout of the iSCSI boot configuration block stored in EEPROM. EEPROM word 0x3D described above stores the offset within the EEPROM of the configuration block. Software must first read word 0x3D to determine the offset of the configuration table before attempting to read or write the configuration block.

The strings defined below are stored in UTF-8 encoding and NULL terminated. All data words are stored in little-endian (Intel) byte order.

Configuration Item	Size in Bytes	Comments
iSCSI Boot Signature	2	‘i’, ‘S’
iSCSI Block Size	2	The structure size is stored in this field and is set depending on the amount of free EEPROM space available. The total size of this structure, including variable length fields, must fit within this space.
Structure Version	1	Version of this structure. Should be set to one.
Reserved	1	Reserved for future use.
Initiator Name	255 + 1	iSCSI Initiator Name - This field is optional and can also be built by DHCP.
Reserved	34	Reserved for future use.
Below fields are per port		

Configuration Item	Size in Bytes	Comments
Flags	2	Bit 0 ⇨ Enable DHCP 0 - Use static configurations from this structure 1 - Overrides configurations retrieved from DHCP. Bit 01h ⇨ Enable DHCP for getting iSCSI target information. 0 - Use static target configuration 1 - Use DHCP to get target information. Bit 02h - 03h ⇨ Authentication Type 00 - none 01 - one way chap 02 - mutual chap Bit 04h - 05h ⇨ Ctrl-D setup menu 00 - enabled 03 - disabled Bit 06h - 07h ⇨ Reserved Bit 08h - 09h ⇨ ARP Retries Retry value Bit 0Ah - 0Fh ⇨ ARP Timeout Timeout value for each retry
Initiator IP	4	DHCP flag not set ⇨ This field should contain the configured IP address. DHCP flag set ⇨ If DHCP bit is set this field is ignored.
Initiator Subnet Mask	4	DHCP flag not set ⇨ This field should contain the configured subnet mask. DHCP flag set ⇨ If DHCP bit is set this field is ignored.
Initiator Gateway	4	DHCP flag not set ⇨ This field should contain the configured gateway DHCP flag set ⇨ If DHCP bit is set this field is ignored.
Boot LUN	2	DHCP flag not set ⇨ Target LUN that Initiator will be attached to. DHCP flag set ⇨ If DHCP bit is set this field is ignored.
Target IP	4	DHCP flag not set ⇨ IP address of iSCSI target. DHCP flag set ⇨ If DHCP bit is set this field is ignored.
Target Port	2	DHCP flag not set ⇨ IP port of iSCSI target. Default is 3260. DHCP flag set ⇨ If DHCP bit is set this field is ignored
Target Name	255 + 1	DHCP flag set ⇨ If DHCP bit is set this field is ignored
CHAP Password	16 + 2	The minimum CHAP secret must be 12 octets and maximum CHAP secret size is 16. 1 byte is reserved for alignment padding and 1 byte for null.
CHAP User Name	127 + 1	The user name must be non-null value and maximum size of user name allowed is 127 characters.
Vlan ID	2	Vlan Id to be used for iSCSI boot traffic. a valid Vlan ID is between 1 and 4094
Mutual CHAP Password	16 + 2	The minimum mutual CHAP secret must be 12 octets and maximum CHAP secret size is 16. 1 byte is reserved for alignment padding and 1 byte for null.
Reserved	160	Reserved for future use.

6.11.8 Alternate MAC address pointer (Word 0x37)

This word may point to a location in the EEPROM containing additional MAC addresses used by system management functions. If the additional MAC addresses are not supported, the word shall be set to 0xFFFF. The structure of the alternate MAC address block can be found in Table 6-92.

Table 6-9. Alternate MAC Address Block

Word Offset	Description[1]
0x0... 0x2	Alternate MAC Address for LAN port assigned to PCI function 0
0x3... 0x5	Alternate MAC Address for LAN port assigned to PCI function 1
0x6... 0x8	Alternate MAC Address for LAN port assigned to PCI function 2
0x9... 0xB	Alternate MAC Address for LAN port assigned to PCI function 3

[1] An alternate MAC Address value of 0xFFFF-FFFF-FFFF means that no alternate MAC address is present for the port.

6.11.9 Checksum Word (Offset 0x3F)

The checksum words (Offset 0x3F from start of Common, LAN 1, LAN 2 and LAN 3 sections) are used to ensure that the base EEPROM image is a valid image. The value of this word should be calculated such that after adding all the words (0x00:0x3E), including the checksum word itself, the sum should be 0xBABA. The initial value in the 16-bit summing register should be 0x0000 and the carry bit should be ignored after each addition.

Note: Hardware does not calculate the checksum word during EEPROM write; it must be calculated by software independently and included in the EEPROM write data. Hardware does not compute a checksum over words 0x00:0x3F during EEPROM reads in order to determine validity of the EEPROM image; this field is provided strictly for software verification of EEPROM validity. All hardware configurations based on word 0x00:0x3F content is based on the validity of the *Signature* field of the *EEPROM Sizing & Protected Fields* EEPROM word (*Signature* must be 01b).

6.11.10 Image Unique ID (Word 0x42, 0x43)

These words contain a unique 32-bit ID for each image generated by Intel to enable tracking of images and comparison to the original image if testing a customer EEPROM image.

§ §

7.0 Inline Functions

7.1 Receive Functionality

7.1.1 Receive Queues Assignment

A received packet goes through three stages of filtering as shown in Figure 7-1. Figure 7-1 describes a switch-like structure that is used in virtualization mode to route packets between the network port (top of drawing) and one or more virtual ports (bottom of figure), where each virtual port can be associated with a virtual machine, a VMM or any other software entity.

The first step in queue assignment is to verify that the packet is destined to the port. This is done by a set of L2 filters as described in Section 7.1.2.

The second stage is specific to virtualization environments and defines the virtual ports (called pools) that are the targets for the Rx packet. A packet can be associated with any number of ports/pools using a selection process described in Section 7.1.1.2.

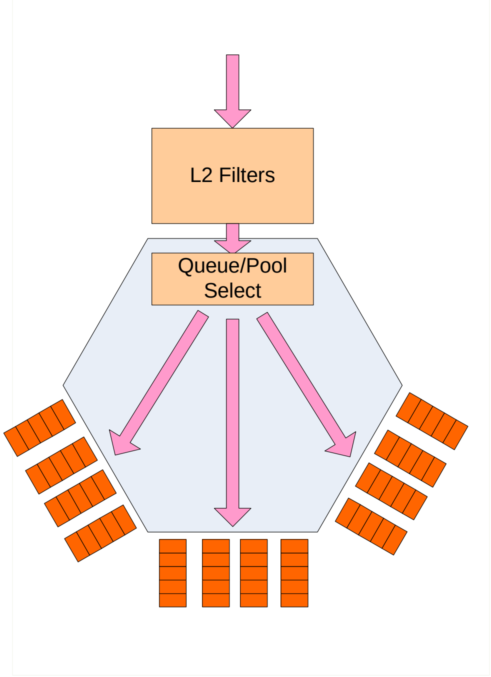


Figure 7-1. Stages in Packet Filtering

In the third stage, a received packet that successfully passed the Rx filters is associated with one or more receive descriptor queues as described in this section.

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

- Virtualization — In a virtualized environment, DMA resources are shared between more than one software entity (operating system and/or software device driver). This is done by allocating receive descriptor queues/pools to virtual partitions (VMM or VMs). Virtualization assigns to each received packet one or more pool indices. Packets are routed to a pool based on their pool index and other considerations. See Section 7.1.1.2 for details on routing for virtualization.
- RSS — Receive Side Scaling distributes packet processing between several processor cores by assigning packets into different descriptor queues. RSS assigns to each received packet an RSS index. Packets are routed to a queue out of a set of Rx queues based on their RSS index and other considerations. See Section 7.1.1.8 for details on RSS.
- L2 Ethertype filters — These filters identify packets by their L2 Ether type and assign them to receive queues. Examples of possible uses are LLDP packets and 802.1X packets. See Section 7.1.1.4 for mode details. The 82580 incorporates 8 Ether-type filters per port.
- 2-tuple filters — These filters identify packets with specific TCP/UDP destination port and/or L4 protocol. Each filter consists of a 2-tuple (protocol and destination TCP/UDP port) and routes packets into one of the Rx queues. The 82580 has 8 such filters per port. See Section 7.1.1.5 for details.
- TCP SYN filters — The 82580 might route TCP packets with their *SYN* flag set into a separate queue. *SYN* packets are often used in *SYN* attacks to load the system with numerous requests for new connections. By filtering such packets to a separate queue, security software can monitor and act on *SYN* attacks. The 82580 has one such filter per port. See Section 7.1.1.7 for more details.
- Flex Filters - These filters can be either used as WoL filters when the 82580 is in D3 state or for queueing in normal operating mode (D0 state). Filters enable queueing according to a match of any 128 Byte sequence at the beginning of a packet. Each one of the 128 bytes can be either compared or masked using a dedicated mask field. The 82580 has 8 such filters per port. See Section 7.1.1.6 for details.

Typically, packet reception consists of recognizing the presence of a packet on the wire, performing address filtering, storing the packet in the receive data FIFO, transferring the data to one of the 8 receive queues in host memory, and updating the state of a receive descriptor.

Note: Maximum supported received-packet size is 9.5 KB (9728 bytes).

A received packet is allocated to a queue based on the previous criteria and the following order:

1. Queue by L2 Ether-type filters (if a match)
2. If RFCTL.SYNQFP is 0b (2-tuple filter and Flex filter have priority), then:
    a. Queue by Flex filter (if a match)
    b. Queue by 2-tuple filter
    c. Queue by SYN filter (if a match)
3. If RFCTL.SYNQFP is 1b (SYN filter has priority), then:
    a. Queue by SYN filter (if a match)
    b. Queue by Flex filter (if a match)
    c. Queue by 2-tuple filter (if a match)
4. Define a pool (if virtualization enabled)
5. Queue by RSS (if RSS enabled).

The tables below describe allocation of queues in each of the modes.

Table 7-1. Queue Allocation[1]

Virtualization	RSS	Queue allocation
Disabled	Disabled	One default queue (MRQC.DEF_Q)
	Enabled	Up to 8 queues by RSS.
Enabled	Disabled	One queue per VM (queues 0-7 for VM 0-7).
	Enabled	Two queues per VM (queues 0, 8; 1, 9; 2, 10; 3, 11; 4, 12; 5, 13; 6, 14; 7; 15 for VM 0-7, respectively). Spread between the queues by RSS.
Disabled	Disabled	One queue per TC (Queue 0 and 8).
	Enabled	Eight queues per TC (queues 0-7 for TC0 and queues 8-15 for TC1). Spread between the queues by RSS.
Enabled	Disabled	One queue per TC per VM (Queues 0, 8; 1, 9; 2, 10; 3, 11; 4, 12; 5, 13; 6, 14; 7; 15 for VM 0-7/TC 0,1 respectively).
	Enabled	Not available

[1] On top of this allocation, the special filters can override the queueing decision.

7.1.1.1 Queuing in a Non-Virtualized Environment

When the *MRQC.Multiple Receive Queues Enable* field equals 010b (Multiple receive queues as defined by filters and RSS for 8 queues) the received packet is assigned to a queue in the following manner:

- L2 Ether-type filters — Each filter identifies one of 8 receive queues.
- SYN filter — Identifies one of 8 receive queues.
- Flex Filter - Each filter identifies one of 8 receive queues.
- 2-tuple filters — Each filter identifies one of 8 receive queues.
- RSS filters - Identifies one of 1 x 8 queues through the RSS index. The following modes are supported:
  - No RSS — The default queue as defined in *MRQC.DEF_Q* is used for packets that do not meet any of the previous conditions.
  - RSS only — A set of 8 queues is allocated for RSS. The queue is identified through the RSS index. Note that it is possible to use a subset of the 8 queues.

When the *MRQC.Multiple Receive Queues Enable* field equals 000b (Multiple receive queues as defined by filters) the received packet is assigned to a queue in the following manner:

- L2 Ether-type filters — Each filter identifies one of 8 receive queues.
- SYN filter — Identifies one of 8 receive queues.
- Flex Filter - Each filter identifies one of 8 receive queues.
- 2-tuple filters — Each filter identifies one of 8 receive queues.

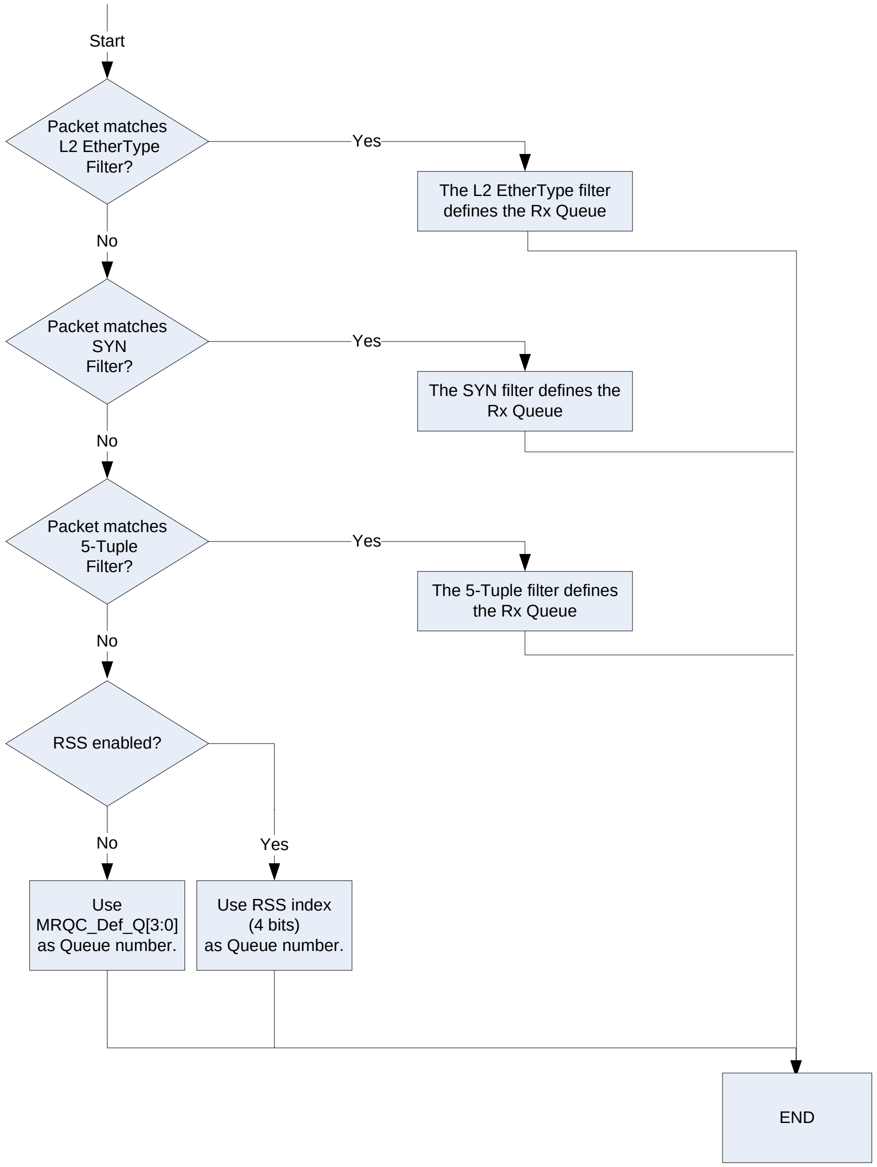

Start
Packet matches L2 EtherType Filter?
Yes
The L2 EtherType filter defines the Rx Queue
No
Packet matches SYN Filter?
Yes
The SYN filter defines the Rx Queue
No
Packet matches 5-Tuple Filter?
Yes
The 5-Tuple filter defines the Rx Queue
No
RSS enabled?
No
Yes
Use MRQC_Def_Q[3:0] as Queue number.
Use RSS index (4 bits) as Queue number.
END

Figure 7-2. Rx Queuing Flow (Non-Virtualized)

7.1.1.2 Receive Queuing in a Virtualized Environment

In VMDq mode, system software allocates the pools to the VMM, an IOVM, or to VMs. When the *MRQC.Multiple Receive Queues Enable* field equals 011b (Multiple receive queues as defined by VMDq), the received packets are allocated to the 8 receive queues/pools in the following manner:

Incoming packets are associated with pools/ queues based on their L2 characteristics as described in Section 7.8.2. This section describes the following stage, where an Rx queue is assigned to each replication of the Rx packet as determined by its pool's association.

A received packet is assigned to a queue/pool in the following manner:

- L2 Ether-type filters — Each filter identifies a specific receive queue (the queue is usually allocated to the VMM or a service operating system).
- SYN filter — Not supported in VT modes.
- 2-tuple filters — Not supported in VT modes.
- Flex filters — Not supported in VT modes.
- RSS filters — Not supported in VT mode.

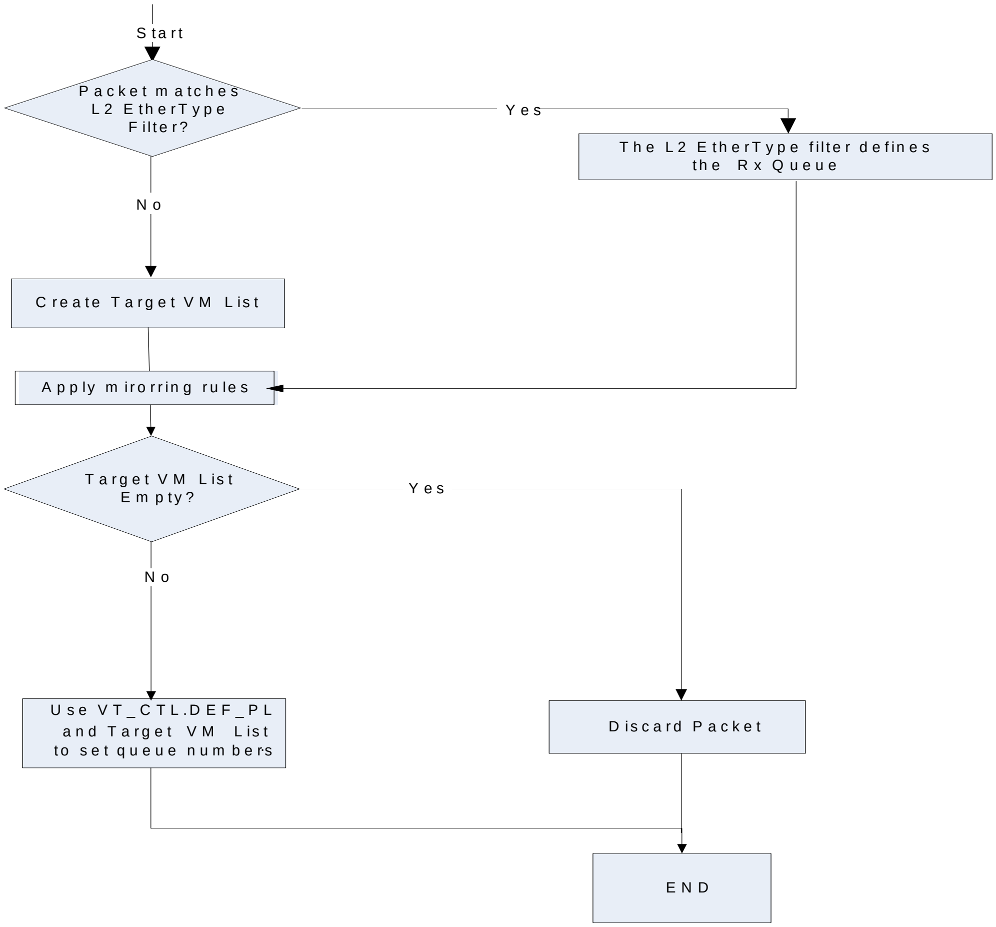

Start
Packet matches L2 EtherType Filter?
Yes
The L2 EtherType filter defines the Rx Queue
No
Create Target VM List
Apply mirorring rules
Target VM List Empty?
Yes
No
Use VT_CTL.DEF_PL and Target VM List to set queue numbers
Discard Packet
END

Figure 7-3. Receive Queuing Flow (Virtualization)

7.1.1.3 Queue Configuration Registers

Configuration registers (CSRs) that control queue operation are replicated per queue (total of 8 copies of each register per port). Each of the replicated registers correspond to a queue such that the queue index equals the serial number of the register (such as register 0 corresponds to queue 0, etc.). Registers included in this category are:

- *RDBAL* and *RDBAH* — Rx Descriptor Base
- *RDLEN* — RX Descriptor Length
- *RDH* — RX Descriptor Head
- *RDT* — RX Descriptor Tail
- *RXDCTL* — Receive Descriptor Control
- *RXCTL* — Rx DCA Control
- *SRRCTL* — Split and Replication Receive Control
- *PSRTYPE* — Packet Split Receive type

7.1.1.4 L2 Ether-Type Filters

These filters identify packets by L2 Ether-type and assign them to a receive queue. The following usages have been identified:

- IEEE 802.1X packets — Extensible Authentication Protocol over LAN (EAPOL).
- Time sync packets (such as IEEE 1588) — Identifies Sync or Delay_Req packets
- IEEE802.1AB LLDP (Link Layer Discovery Protocol) packets.

The 82580 incorporates 8 Ether-type filters.

The *Packet Type* field in the Rx descriptor captures the filter number that matched the L2 Ether-type. See Section 7.1.5 for decoding of the *Packet Type* field.

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

- The *EType* field contains the 16-bit Ether-type compared against all L2 type fields in the Rx packet.
- The *Filter Enable* bit enables identification of Rx packets by Ether-type according to this filter. If this bit is cleared, the filter is ignored for all purposes.
- The *Rx Queue* field contains the absolute destination queue for the packet.
- The *1588 Time Stamp* field indicates that the packet should be time stamped according to the IEEE 1588 specification.
- The *Queue Enable* field enables forwarding Rx packets based on the Ether-type defined in this register.

Note: Software should not assign the same Ether-type value to different ETQF filters with different *Rx Queue* assignments.

Special considerations for Virtualization modes:

- Packets that match an Ether-type filter are diverted from their original pool (the pool identified by the L2 filters) to the pool used as the pool to which the queue in the *Queue* field belongs. In other words, The L2 filters are ignored in determining the pool for such packets.
- The same applies for multi-cast packets. A single copy is posted to the pool defined by the filter.
- Mirroring rules:
  - — If a pool is being mirrored, the pool to which the queue belongs to is used to determine if a packet that matches the filter should be mirrored.
  - — The queue inside the pool is used for both the original pool and the mirroring pool.

7.1.1.5 2-Tuple Filters

These filters identify specific packets destined to a certain TCP/UDP port and implement a specific protocol. Each filter consists of a 2-tuple (protocol and destination TCP/UDP port) and forwards packets into one of the receive queues.

The 82580 incorporates 8 such filters.

The 2-tuple filters are configured via the *TTQF* (See Section 8.11.3), *IMIR* (See Section 8.11.1) and *IMIR_EXT* (See Section 8.11.2) registers as follows (per filter):

- Protocol — Identifies the IP protocol, part of the 2-tuple queue filters. Enabled by a bit in the *TTQF.Mask* field.
- Destination port — Identifies the TCP/UDP destination port, part of the 2-tuple queue filters. Enabled by the *IMIR.PORT_BP* bit.
- Size threshold — Identifies the length of the packet that should trigger the filter. This is the length as received by the host, not including any part of the packet removed by hardware. Enabled by the *IMIREXT.Size_BP* field.
- Control Bits — Identify TCP flags that might be part of the filtering process. Enabled by the *IMIREXT.CtrlBit_BP* field.
- Rx queue — Determines the Rx queue for packets that match this filter:
  - The *TTQF.Rx Queue* field contains the queue serial number.
- Queue enable — Enables forwarding a packet that uses this filter to the queue defined in the *TTQF.Rx Queue* field.
- Mask — A 1-bit field that masks the L4 protocol check. The filter is a logical AND of the non-masked 2-tuple fields. If all 2-tuple fields are masked, the filter is not used for queue forwarding.

Notes:

- If more than one 2-tuple filter with the same priority is matched by the packet, the first filter (lowest ordinal number) is used in order to define the queue destination of this packet.
- The immediate interrupt and 1588 actions are defined by the OR of all the matching filters.

7.1.1.6 Flex Filters

The 82580 supports a total of 8 flexible filters. Each filter can be configured to recognize any arbitrary pattern within the first 128 bytes of the packet. To configure the flexible filters, software programs the mask values (required values and the minimum packet length), into the Flexible Host Filter Table (*FHFT* and *FHFT_EXT*, See Section 8.21.9 and Section 8.21.10). These 8 flexible filters can be used as for wake-up when in D3 state or for queueing when in D0 state. Software must enable the filters in the *Wake Up Filter* Control (*WUFC See* Section 8.21.2) register for operation in D3 or D0 mode. In D0 mode these filters enable forwarding of packets that match up to 128 Bytes defined in the filter to one of the receive queues. In D3 mode these filters can be used for Wake-on-Lan as described in Section 5.6.3.2

Once enabled, the flexible filters scan incoming packets for a match. If the filter encounters any byte in the packet where the mask bit is one and the byte doesn't match the value programmed in the Flexible Host Filter Table (*FHFT* or *FHFT_EXT*), then the filter fails that packet. If the filter reaches the required length without failing the packet, it forwards the packet to the appropriate receive queue. It ignores any mask bits set to one beyond the required length (defined in the Length field in the *FHFT* or *FHFT_EXT* registers).

Note: The flex filters are temporarily disabled when read from or written to by the host. Any packet received during a read or write operation is dropped. Filter operation resumes once the read or write access completes.

The flex fitters are configured in D0 state via the *WUFC*, *FHFT* and *FHFT_EXT* registers as follows (per filter):

- Byte Sequence to be compared - Program 128 Byte sequence, mask bits and *Length* field in FHFT and *FHFT_EXT* registers.
- Filter Priority - Program filter priority in queueing field in *FHFT* and *FHFT_EXT* registers.
- Receive queue - Program receive queue to forward packet in queueing field in *FHFT* and *FHFT_EXT* registers.
- Filter actions - Program immediate interrupt requirement in queueing field in *FHFT* and *FHFT_EXT* registers.
- Filter enable - Set *WUFC.FLEX_HQ* bit to 1 to enable flex filter operation in D0 state. Set appropriate *WUFC.FLX[n]* bit to 1 to enable specific flex filter.

Before entering D3 state software device driver programs the *FHFT* and *FHFT_EXT* filters for appropriate wake events and enables relevant filters by setting the *WUFC.FLX[n]* bit to 1. Following move to D0 state the software device driver programs the FHFT and FHFT_EXT filters for appropriate queueing decisions enables relevant filters by setting the *WUFC.FLX[n]* bit to 1 and the *WUFC.FLEX_HQ* bit to 1.

Notes: If more than one flex filter with the same priority is matched by the packet, the first filter (lowest address) is used in order to define the queue destination of this packet.

The immediate interrupt action is defined by the OR of all the matching filters.

7.1.1.7 SYN Packet Filters

The 82580 might forward TCP packets whose *SYN* flag is set into a separate queue. SYN packets are often used in SYN attacks to load the system with numerous requests for new connections. By filtering such packets to a separate queue, security software can monitor and act on SYN attacks.

SYN filters are configured via the SYNQF registers as follows:

- Queue En — Enables forwarding of SYN packets to a specific queue.
- Rx Queue field — Contains the destination queue for the packet.

This filter is not to be used in a virtualized environment.

7.1.1.8 Receive-Side Scaling (RSS)

RSS is a mechanism to distribute received packets into several descriptor queues. Software then assigns each queue to a different processor, sharing the load of packet processing among several processors.

The 82580 uses RSS as one ingredient in its packet assignment policy (the others are the various filters and virtualization). The RSS output is a RSS index. The 82580’s global assignment uses these bits (or only some of the LSB bits) as part of the queue number.

RSS is enabled in the *MRQC* register. The 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 82580 provides software with the following information as required by Microsoft* RSS specification or for device driver assistance:

- A Dword result of the Microsoft* RSS hash function, to be used by the stack for flow classification, is written into the receive packet descriptor (required by Microsoft* RSS).
- A 4-bit RSS *Type* field conveys the hash function used for the specific packet (required by Microsoft* RSS).

Figure 7-4 shows the process of computing an RSS output:

1. The receive packet is parsed into the header fields used by the hash operation (such as IP addresses, TCP port, etc.).
2. A hash calculation is performed. The 82580 supports a single hash function, as defined by Microsoft* RSS. The 82580 does not indicate to the software device driver which hash function is used. The 32-bit result is fed into the packet receive descriptor.
3. The seven LSB bits of the hash result are used as an index into a 128-entry indirection table. Each entry provides a 3-bit RSS output index.

When RSS is disabled, packets are assigned an RSS output index = zero. System software might enable or disable RSS at any time. While disabled, system software might update the contents of any of the RSS-related registers.

When multiple requests queues are enabled in RSS mode, un-decodable packets are assigned an RSS output index = zero. The 32-bit tag (normally a result of the hash function) equals zero.

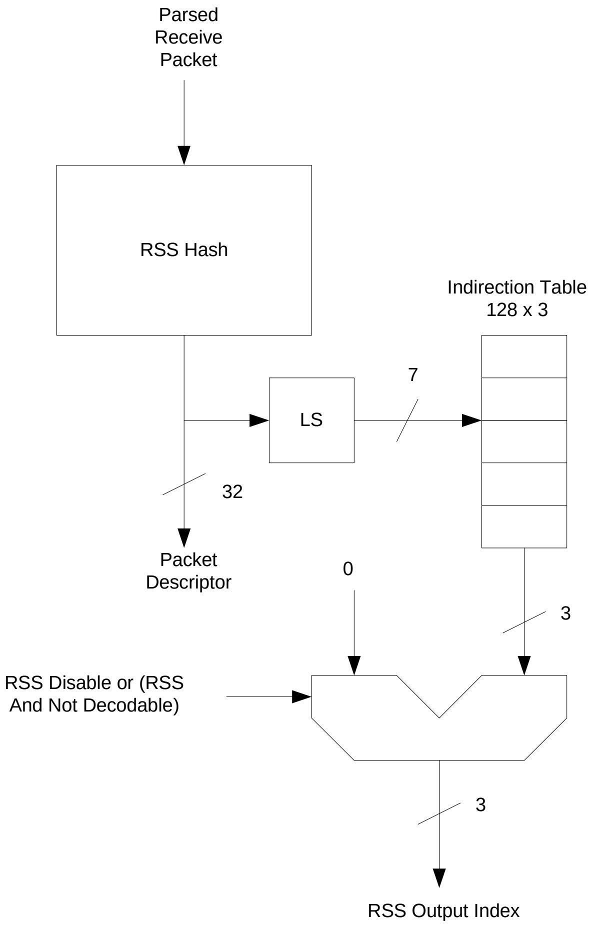


Figure 7-4. RSS Block Diagram

7.1.1.8.1 RSS Hash Function

Section 7.1.1.8.1 provides a verification suite used to validate that the hash function is computed according to Microsoft* nomenclature.

The 82580 hash function follows Microsoft* definition. A single hash function is defined with several variations for the following cases:

- TcpIPv4 — The 82580 parses the packet to identify an IPv4 packet containing a TCP segment per the criteria described later in this section. If the packet is not an IPv4 packet containing a TCP segment, RSS is not done for the packet.
- IPv4 — The 82580 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 82580 parses the packet to identify an IPv6 packet containing a TCP segment per the criteria described later in this section. If the packet is not an IPv6 packet containing a TCP segment, RSS is not done for the packet.

- TcpIPv6Ex — The 82580 parses the packet to identify an IPv6 packet containing a TCP segment with extensions per the criteria described later in this section. If the packet is not an IPv6 packet containing a TCP segment, RSS is not done for the packet. Extension headers should be parsed for a *Home-Address-Option* field (for source address) or the *Routing-Header-Type-2* field (for destination address).
- IPv6Ex — The 82580 parses the packet to identify an IPv6 packet. Extension headers should be parsed for a *Home-Address-Option* field (for source address) or the *Routing-Header-Type-2* field (for destination address). Note that the packet is not required to contain any of these extension headers to be hashed by this function. In this case, the IPv6 hash is used. If the packet is not an IPv6 packet, RSS is not done for the packet.
- IPv6 — The 82580 parses the packet to identify an IPv6 packet. If the packet is not an IPv6 packet, receive-side-scaling is not done for the packet.

The following additional cases are not part of the Microsoft* RSS specification:

- UdpIPV4 — The 82580 parses the packet to identify a packet with UDP over IPv4.
- UdpIPV6 — The 82580 parses the packet to identify a packet with UDP over IPv6.
- UdpIPV6Ex — The 82580 parses the packet to identify a packet with UDP over IPv6 with extensions.

A packet is identified as containing a TCP segment if all of the following conditions are met:

- The transport layer protocol is TCP (not UDP, ICMP, IGMP, etc.).
- The TCP segment can be parsed (such as IP options can be parsed, packet not encrypted).
- The packet is not fragmented (even if the fragment contains a complete TCP header).

Bits[31:16] of the Multiple Receive Queues Command (*MRQC)* register enable each of the above hash function variations (several can be set at a given time). If several functions are enabled at the same time, priority is defined as follows (skip functions that are not enabled):

IPv4 packet:

1. Try using the TcpIPv4 function.
2. Try using IPV4_UDP function.
3. Try using the IPv4 function.

IPv6 packet:

1. If TcpIPv6Ex is enabled, try using the TcpIPv6Ex function; else if TcpIPv6 is enabled try using the TcpIPv6 function.
2. If UdpIPv6Ex is enabled, try using UdpIPv6Ex function; else if UpdIPv6 is enabled try using UdpIPv6 function.
3. If IPv6Ex is enabled, try using the IPv6Ex function, else if IPv6 is enabled, try using the IPv6 function.

The following combinations are currently supported:

- Any combination of IPv4, TcpIPv4, and UdpIPv4.
- And/or.
- Any combination of either IPv6, TcpIPv6, and UdpIPv6 or IPv6Ex, TcpIPv6Ex, and UdpIPv6Ex.

When a packet cannot be parsed by the previously mentioned rules, it is assigned an RSS output index = zero. The 32-bit tag (normally a result of the hash function) equals zero.

The 32-bit result of the hash computation is written into the packet descriptor and also provides an index into the indirection table.

The following notation is used to describe the hash functions:

- Ordering is little endian in both bytes and bits. For example, the IP address 161.142.100.80 translates into 0xa18e6450 in the signature.
- A "^ "denotes bit-wise XOR operation of same-width vectors.
- @x-y denotes bytes x through y (including both of them) of the incoming packet, where byte 0 is the first byte of the IP header. In other words, it is considered that all byte-offsets as offsets into a packet where the framing layer header has been stripped out. Therefore, the source IPv4 address is referred to as @12-15, while the destination v4 address is referred to as @16-19.
- @x-y, @v-w denotes concatenation of bytes x-y, followed by bytes v-w, preserving the order in which they occurred in the packet.

All hash function variations (IPv4 and IPv6) follow the same general structure. Specific details for each variation are described in the following section. The hash uses a random secret key length of 320 bits (40 bytes); the key is typically supplied through the RSS Random Key Register (RSSRK).

The algorithm works by examining each bit of the hash input from left to right. Intel's nomenclature defines left and right for a byte-array as follows: Given an array K with k bytes, Intel's nomenclature assumes that the array is laid out as shown:

```
K[0] K[1] K[2] … K[k-1]
```

K[0] is the left-most byte, and the MSB of K[0] is the left-most bit. K[k-1] is the right-most byte, and the LSB of K[k-1] is the right-most bit.

```
ComputeHash(input[], N)

For hash-input input[] of length N bytes (8N bits) and a random secret key K of 320 bits
Result = 0;
For each bit b in input[] {
if (b == 1) then Result ^= (left-most 32 bits of K);
shift K left 1 bit position;
}
return Result;
```

The following four pseudo-code examples are intended to help clarify exactly how the hash is to be performed in four cases, IPv4 with and without ability to parse the TCP header and IPv6 with an without a TCP header.

7.1.1.8.1.1 Hash for IPv4 with TCP

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

```
Input[12] = @12-15, @16-19, @20-21, @22-23.

Result = ComputeHash(Input, 12);
```

7.1.1.8.1.2 Hash for IPv4 with UDP

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

```
Input[12] = @12-15, @16-19, @20-21, @22-23.
```

```
Result = ComputeHash(Input, 12);
```

7.1.1.8.1.3 Hash for IPv4 without TCP

Concatenate SourceAddress and DestinationAddress into one single byte-array

```
Input[8] = @12-15, @16-19
```

```
Result = ComputeHash(Input, 8)
```

7.1.1.8.1.4 Hash for IPv6 with TCP

Similar to above:

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

```
Result = ComputeHash(Input, 36)
```

7.1.1.8.1.5 Hash for IPv6 with UDP

Similar to above:

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

```
Result = ComputeHash(Input, 36)
```

7.1.1.8.1.6 Hash for IPv6 without TCP

```
Input[32] = @8-23, @24-39
```

```
Result = ComputeHash(Input, 32)
```

7.1.1.8.2 Indirection Table

The *RETA* indirection table is a 128-entry structure, indexed by the seven LSB bits of the hash function output. Each entry of the table contains the following:

- Bits [2:0] - RSS index

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

System software might update the indirection table during run time. Such updates of the table are not synchronized with the arrival time of received packets. Therefore, it is not guaranteed that a table update takes effect on a specific packet boundary.

7.1.1.8.3 RSS Verification Suite

Assume that the random key byte-stream is:

```
0x6d, 0x5a, 0x56, 0xda, 0x25, 0x5b, 0x0e, 0xc2,
```

```
0x41, 0x67, 0x25, 0x3d, 0x43, 0xa3, 0x8f, 0xb0,
0xd0, 0xca, 0x2b, 0xcb, 0xae, 0x7b, 0x30, 0xb4,
0x77, 0xcb, 0x2d, 0xa3, 0x80, 0x30, 0xf2, 0x0c,
0x6a, 0x42, 0xb7, 0x3b, 0xbe, 0xac, 0x01, 0xfa
```

7.1.1.8.3.1 IPv4

Table 7-2. IPv4

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

7.1.1.8.3.2 IPv6

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

Table 7-3. IPv6

Destination Address/Port	Source Address/Port	IPv6 Only	IPv6 With TCP
3ffe:2501:200:1fff::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

7.1.1.8.4 Association Through MAC Address

Each of the 24 MAC address filters can be associated with a VM. The *POOLSEL* field in the Receive Address High (RAH) register determines the target VM. Packets that do not match any of the MAC filters (such as promiscuous) are assigned with the default VM as defined in the VT_CTL.DEF_PL field.

Software can program different values to the MAC filters (any bits in RAH or RAL) at any time. The 82580 would respond to the change on a packet boundary but does not guarantee the change to take place at some precise time.

7.1.2 L2 Packet Filtering

The receive packet filtering role is to determine which of the incoming packets are allowed to pass to the local system and which of the incoming packets should be dropped since they are not targeted to the local system. Received packets can be destined to the host, to a manageability controller (BMC), or to both. This section describes how host filtering is done, and the interaction with management filtering.

As shown in Figure 7-5, host filtering has three stages:

1. Packets are filtered by L2 filters (MAC address, unicast/multicast/broadcast). See Section 7.1.2.1 for details.
2. Packets are then filtered by VLAN if a VLAN tag is present. See Section 7.1.2.2 for details.
3. Packets are filtered by the manageability filters (port, IP, flex, other). See Section 10.4 for details.

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

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

- The packet does not pass VLAN filtering as described later in this section.

- The packet passes manageability filtering and then the manageability filters determine if the packet should be sent only to the BMC (see Section 10.4 and the *MNGONLY* register).

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 are detected. However, if the *store bad packet* bit is set (*RCTL.SBP*), then bad packets that pass the filter function are stored in host memory. Packet errors are indicated by error bits in the receive descriptor (*RDESC.ERRORS*). It is possible to receive all packets, regardless of whether they are bad, by setting the promiscuous enables and the *store bad packet* bit.

If there is insufficient space in the receive FIFO, hardware drops the packet and indicates the missed packet in the appropriate statistics registers.

When the packet is routed to a queue with the *SRRCTL.Drop_En* bit set to 1, receive packets are dropped when insufficient receive descriptors exist to write the packet into system memory.

Note: CRC errors before the SFD are ignored. Any packet must have a valid SFD in order to be recognized by the 82580 (even bad packets).

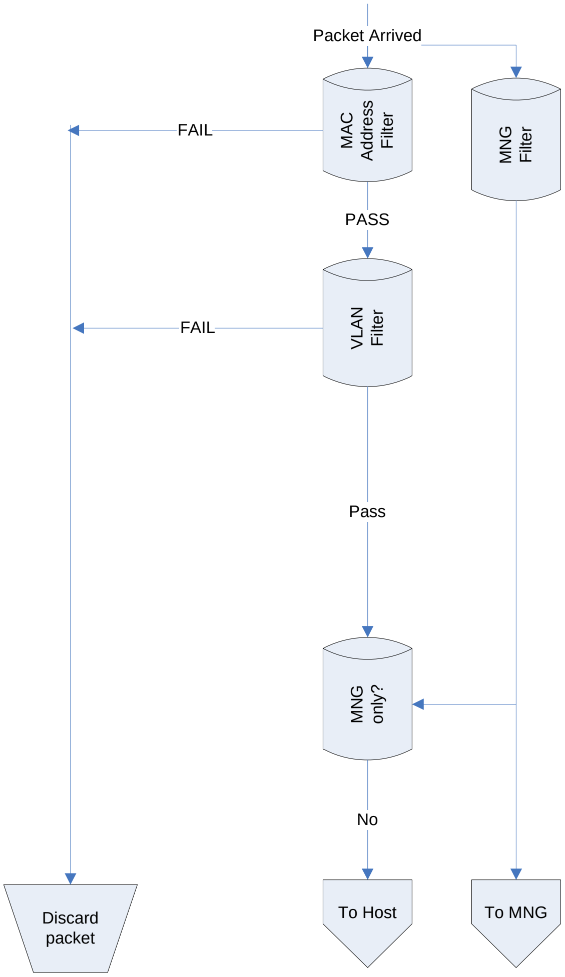


Figure 7-5. Receive Filtering Flow Chart

7.1.2.1 MAC Address Filtering

Figure 7-6 shows the MAC address filtering. A packet passes successfully through the MAC address filtering if any of the following conditions are met:

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

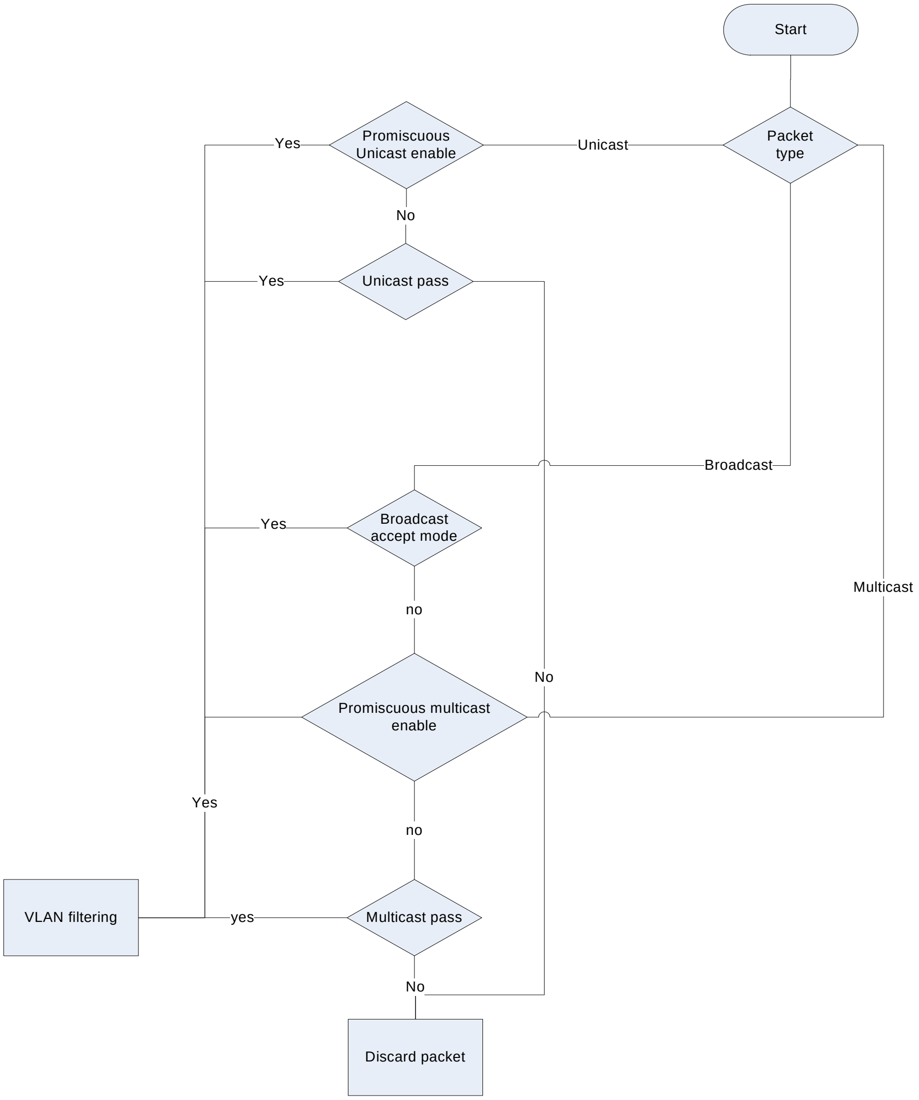


Figure 7-6. Host MAC Address Receive Filtering Flow Chart

7.1.2.1.1 Unicast Filter

The entire MAC address is checked against the 24 host unicast addresses. The 24 host unicast addresses are controlled by the host interface (the BMC must not change them). The other 2 addresses are dedicated to management functions and are only accessed by the BMC. The destination address of incoming packet must exactly match one of the pre-configured host address filters. These addresses can be unicast or multicast. Those filters are configured through *RAL*, and *RAH* registers.

Promiscuous Unicast — Receive all unicasts. Promiscuous unicast mode in the *RCTL* register can be set/cleared only through the host interface (not by the BMC). This mode is usually used when the 82580 is used as a sniffer.

Unicast Hash Table — Destination address matching the Unicast Hash Table (*UTA*).

7.1.2.1.2 Multicast Filter (Partial)

The 12-bit portion of the incoming packet multicast address must exactly match Multicast Filter Address (*MFA*) in order to pass multicast filtering. Those 12 bits out of 48 bits of the destination address can be selected by the *MO* field of *RCTL* (Section 8.10.1). The 12 bits extracted from the Multicast Destination address are used as an address for a bit in the *Multicast Table Array* (*MTA*), if the value of the bit selected in the *MTA* table is 1b, the packet is sent to the Host (See Section 8.10.15). These entries can be configured only by the host interface and cannot be controlled by the BMC. Packets received according to this mode have the *PIF* bit in the descriptor set to indicate imperfect filtering that should be validated by the software device driver.

Promiscuous Multicast — Receive all multicast. Promiscuous multicast mode can be set/cleared in the *RCTL* register only through the host interface (not by the BMC) and it is usually used when the 82580 is used as a sniffer.

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

7.1.2.2 VLAN Filtering

A receive packet that successfully passed MAC address filtering is then subjected to VLAN header filtering.

1. If the packet does not have a VLAN header, it passes to the next filtering stage.

Note: If external VLAN is enabled (*CTRL_EXT.EXT_VLAN* is set), it is assumed that the first VLAN tag is an external VLAN and it is skipped. All next stages refer to the second VLAN.

2. If VLAN filtering is disabled (*RCTL.VFE* bit is cleared), the packet is forwarded to the next filtering stage.
3. If the packet has a VLAN header, and it matches an enabled host VLAN filter (relevant bit in *VFTA* table is set), the packet is forwarded to the next filtering stage.
4. Otherwise, the packet is dropped.

Figure 7-7 shows the VLAN filtering flow.

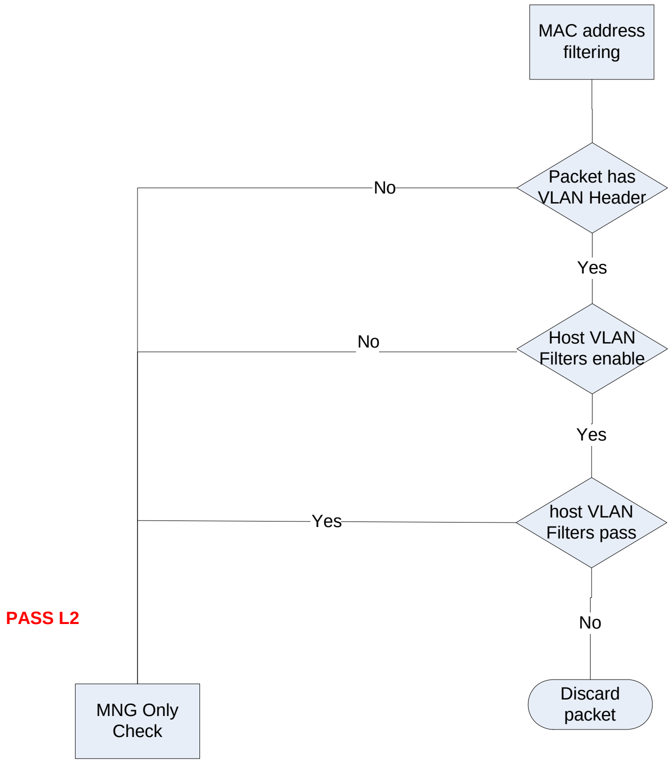


Figure 7-7. VLAN Filtering

7.1.2.3 Manageability Filtering

Manageability filtering is described in Section 10.4.

Figure 7-8 shows the manageability portion of the packet filtering and it is brought here to make the receive packet filtering functionality description complete.

Note: The manageability engine might decide to block part of the received packets from also being sent to the Host, according to the external BMC instructions and the EEPROM settings.

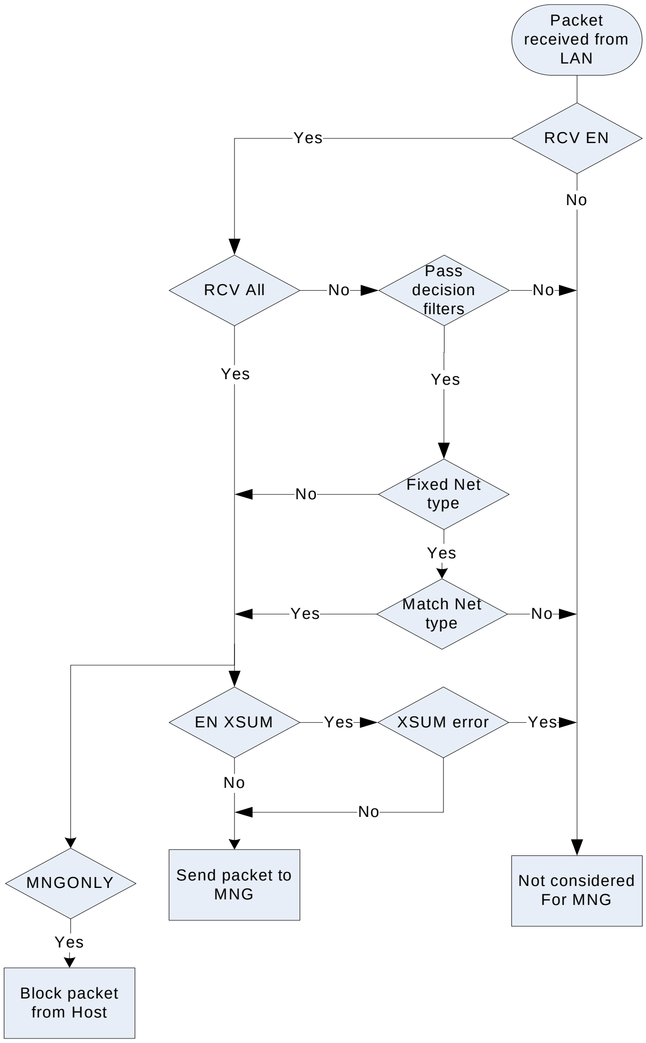


Figure 7-8. Manageability Filtering

7.1.3 Receive Data Storage

7.1.3.1 Host Buffers

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

The size of the buffer can be set using either the generic *RCTL.BSIZE* field, or the per queue *SRRCTL[n].BSIZEPACKET* field. If *SRRCTL[n].BSIZEPACKET* is programmed to zero for any queue, the buffer size defined by *RCTL.BSIZE* is used. Otherwise, the buffer size defined by *SRRCTL[n].BSIZEPACKET* is used.

When the receive buffer size is selected by settings bits in the Receive Control (*RCTL.BSIZE*) field. The register supports buffer sizes of 256, 512, 1024, and 2048 bytes.

If the receive buffer size is selected by *SRRCTL[n].BSIZEPACKET*, buffer sizes of 1 Kbytes to 127 Kbytes are supported, with a resolution of 1 Kbyte.

In addition, for advanced descriptor usage the *SRRCTL.BSIZEHEADER* field is used to define the size of the buffers allocated to headers. Header Buffer sizes of 64 bytes to 960 bytes with a resolution of 64 bytes are supported.

The 82580 places no alignment restrictions on receive memory buffer addresses. This is desirable in situations where the receive buffer was allocated by higher layers in the networking software stack, as these higher layers might have no knowledge of a specific device's buffer alignment requirements.

Note: When the *No-Snoop Enable* bit is used in advanced descriptors, the buffer address is 16-bit (2-byte) aligned.

7.1.3.2 On-Chip Receive Buffers

The 82580 allocates by default a 32 KB on-chip packet buffer per port. The buffer can be used to store packets until they are forwarded to the host. The 82580 utilizes a single common ram structure for the on-chip receive buffers allocated to the various ports. If a port is disabled, so that it can't be accessed by host and management, by either:

1. Pin assertion (LAN0_DIS_N, LAN1_DIS_N, LAN2_DIS_N, LAN3_DIS_N for port 0 to 3 respectively) and setting the EEPROM bit *PHY_in_LAN_disable* in the "*Software Defined Pins Control*" word to 1 for the relevant port.
2. Setting EEPROM bit *LAN_DIS* or the LAN_PCI_DIS in the "*Software Defined Pins Control*" word for the relevant port to 1.

The freed buffer space can be allocated to the active ports via the "*Initialization Control 4*" EEPROM word. Actual on-chip receive buffer allocated to the port can be read in the *IRPBS register*.

7.1.3.3 On-Chip descriptor Buffers

The 82580 contains a 16 descriptor cache for each receive queue used to reduce the latency of packet processing and to optimize the usage of PCIe bandwidth by fetching and writing back descriptors in bursts. The fetch and writeback algorithm are described in Section 7.1.6 and Section 7.1.7.

7.1.4 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 82580 uses the legacy Receive descriptor as shown in Table 7-4. The shaded areas indicate fields that are modified by hardware upon packet reception (so-called descriptor write-back).

Note: Legacy descriptors should not be used when advanced features such as Virtualization are activated.

Table 7-4. Legacy Receive Descriptor (RDESC) Layout

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

After receiving a packet for the 82580, hardware stores the packet data into the indicated buffer and writes the length, packet checksum, status, errors, and status fields.

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

Length Field (16)

Length covers the data written to a receive buffer including CRC bytes (if any). Software must read multiple descriptors to determine the complete length for a packet that spans multiple receive buffers.

Fragment Checksum (16)

This field is used to provide the fragment checksum value. This field equals to the unadjusted 16-bit ones complement of the packet. Checksum calculation starts at the L4 layer (after the IP header) until the end of the packet excluding the CRC bytes. In order to use the fragment checksum assist to offload L4 checksum verification, software might need to back out some of the bytes in the packet. For more details see Section 7.1.11.2

Status Field (8)

Status information indicates whether the descriptor has been used and whether the referenced buffer is the last one for the packet. See Table 7-5 for the layout of the *Status* field. Error status information is shown in Figure 7-9.

Table 7-5. Receive Status (RDESC.STATUS) Layout

7	6	5	4	3	2	1	0
PIF	IPCS	L4CS	UDPCS	VP	Rsv	EOP	DD

- PIF (bit 7) - Passed in-exact filter
- IPCS (bit 6) - Ipv4 checksum calculated on packet
- L4CS (bit 5) - L4 (UDP or 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 in 802.1q packet
- RSV (bit 2) - Reserved
- EOP (bit 1) - End of packet
- DD (bit 0) - Descriptor done

EOP and DD

The following table lists the meaning of these bits:

Table 7-6. Receive Status Bits

DD	EOP	Description
0b	0b	Software setting of the descriptor when it hands it off to the hardware.
0b	1b	Reserved (invalid option).
1b	0b	A completion status indication for a non-last descriptor of a packet that spans across multiple descriptors. In a single packet case, DD indicates that the hardware is done with the descriptor and its buffers. Only the *Length* fields are valid on this descriptor.
1b	1b	A completion status indication of the entire packet. Note that software Might take ownership of its descriptors. All fields in the descriptor are valid (reported by the hardware).

VP Field

The *VP* field indicates whether the incoming packet's type matches the VLAN Ethernet Type programmed in the *VET* Register. For example, if the packet is a VLAN (802.1q) type, it is set if the packet type matches *VET* and *CTRL.VME* is set (Vlan mode enabled). It also indicates that VLAN has been stripped from the 802.1q packet. For more details, see Section 7.4.

IPCS (Ipv4 Checksum), L4CS (L4 Checksum), and UDPCS (UDP Checksum)

The meaning of these bits is shown in the table below:

Table 7-7. IPCS, L4CS, and UDPCS

L4CS	UDPCS	IPCS	Functionality
0b	0b	0b	Hardware does not provide checksum offload. Special case: Hardware does not provide UDP checksum offload for IPV4 packet with UDP checksum = 0b
1b	0b	1b / 0b	Hardware provides IPv4 checksum offload if IPCS is active and TCP checksum is offload. A pass/fail indication is provided in the *Error* field – IPE and L4E.
0b	1b	1b / 0b	Hardware provides IPv4 checksum offload if IPCS is active and UDP checksum is offload. A pass/fail indication is provided in the *Error* field – IPE and L4E.

Refer to Table 7-19 for a description of supported packet types for receive checksum offloading. Unsupported packet types do not have the *IPCS* or *L4CS* bits set. IPv6 packets do not have the *IPCS* bit set, but might have the *L4CS* bit set if the 82580 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. Multicast packets passing only the Multicast Vector (MTA) or unicast packets passing only the Unicast Hash Table (UTA) but not any of the MAC address exact filters (RAH, RAL) have *PIF* set. In addition, the following condition causes *PIF* to be cleared:

- The DA of the packet is a multicast address and promiscuous multicast is set (*RCTL.MPE* = 1b).
- The DA of the packet is a broadcast address and accept broadcast mode is set (*RCTL.BAM* = 1b)

A MAC control frame forwarded to the host (*RCTL.PMCF* = 0b) that does not match any of the exact filters, has the *PIF* bit set.

Error Field (8)

Most error information appears only when the *store-bad-packet* bit (*RCTL.SBP*) is set and a bad packet is received. See Table 7-8 for a definition of the possible errors and their bit positions.

Table 7-8. RXE, LPE and L4E

7	6	5	4	3	2	1	0
RXE	IPE	L4E	Reserved				

- RXE (bit 7) - RX Data Error
- IPE (bit 6) - Ipv4 Checksum Error
- L4E (bit 5) - TCP/UDP Checksum Error
- Reserved (bit 4:0)

IPE/L4E

The IP and TCP/UDP checksum error bits from Table 7-8 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 affect on packet filtering.

If receive checksum offloading is disabled (RXCSUM.IPOFL and RXCSUM.TUOFL), the *IPE* and *L4E* bits are 0b.

RXE

The RXE error bit is asserted in the following case:

1. CRC error is detected. CRC can be a result of reception of /V/ symbol on the TBI interface (see section 3.5.3.3.2) or assertion of RxERR on the MII/GMII interface or bad EOP or lose of sync during packet reception. Packets with a CRC error are posted to host memory only when *store-bad-packet* bit (*RCTL.SBP*) is set.

VLAN Tag Field (16)

Hardware stores additional information in the receive descriptor for 802.1q packets. If the packet type is 802.1q (determined when a packet matches *VET* and *CTRL.VME* = 1b), then the *VLAN Tag* field records the VLAN information and the four-byte VLAN information is stripped from the packet data storage. Otherwise, the *VLAN Tag* field contains 0x0000. The rule for *VLAN tag* is to use network ordering (also called big endian). It appears in the following manner in the descriptor:

Table 7-9. VLAN Tag Field Layout (for 802.1q Packet)

15 13	12	11 0
PRI	CFI	VLAN

7.1.5 Advanced Receive Descriptors

7.1.5.1 Advanced Receive Descriptors (RDESC) - Read Format

Table 7-10 shows the receive descriptor. This is the format that software writes to the descriptor queue and hardware reads from the descriptor queue in host memory. Hardware writes back the descriptor in a different format, shown in Table 7-11.

Table 7-10. RDESC Descriptor Read Format

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

Packet Buffer Address (64) - Physical address of the packet buffer. The lowest bit is either A0 (LSB of address) or NSE (No-Snoop Enable), depending on bit *RXCTL.RXdataWriteNSEn* of the relevant queue. See Section 8.13.1.

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

Note: The 82580 does not support null descriptors (a descriptor with a packet or header address that is always equal to zero).

When software sets the *NSE* bit in the receive descriptor, the 82580 places the received packet associated with this descriptor in memory at the packet buffer address with *NSE* set in the PCIe attribute fields. NSE does not affect the data written to the header buffer address.

When a packet spans more than one descriptor, the header buffer address is not used for the second, third, etc. descriptors; only the packet buffer address is used in this case.

NSE is enabled for packet buffers that the software device driver knows have not been touched by the processor since the last time they were used, so the data cannot be in the processor cache and snoop is always a miss. Avoiding these snoop misses improves system performance. No-snoop is particularly useful when the DMA engine is moving the data from the packet buffer into application buffers, and the software device driver is using the information in the header buffer for its work with the packet.

Note: When No-Snoop Enable is used, relaxed ordering should also be enabled with *CTRL_EXT.RO_DIS*.

7.1.5.2 Advanced Receive Descriptors (RDESC) - Writeback Format

When the 82580 writes back the descriptors, it uses the descriptor format shown in Table 7-11.

Note: *SRRCTL[n]. DESCTYPE* must be set to a value other than 000b for the 82580 to write back the special descriptors.

Table 7-11. RDESC Descriptor Write-Back Format

	63 48	47 35	34 32	31	30 21	20	19 17	16 4	3 0
0	RSS Hash Value/Fragment Checksum and IP identification			SPH	HDR_LEN	RSV		Packet Type	RSS Type
8	VLAN Tag	PKT_LEN		Extended Error			Extended Status		

RSS Type (4)

Table 7-12. RSS Type

Packet Type	Description
0x0	No hash computation done for this packet.
0x1	HASH_TCP_IPV4
0x2	HASH_IPV4
0x3	HASH_TCP_IPV6
0x4	HASH_IPV6_EX
0x5	HASH_IPV6
0x6	HASH_TCP_IPV6_EX
0x7	HASH_UDP_IPV4
0x8	HASH_UDP_IPV6
0x9	HASH_UDP_IPV6_EX
0xA:0xF	Reserved

The 82580 must identify the packet type and then choose the appropriate RSS hash function to be used on the packet. The RSS type reports the packet type that was used for the RSS hash function.

Packet Type (13)

- VPKT (bit 12) - VLAN Packet indication

The 12 LSB bits of the packet type reports the packet type identified by the hardware as follows:

Table 7-13. Packet Type LSB Bits (11:0)

Bit Index	Bit 11 = 0b	Bit 11 = 1b (L2 packet)
0	IPV4 - IPv4 header present	EtherType - ETQF register index that matches the packet. Special types might be defined for 1588, 802.1X, LLDP or any other requested type.
1	IPV4E - IPv4 Header includes extensions	
2	IPV6 - IPv6 header present	
3	IPV6E- IPv6 Header includes extensions	Reserved
4	TCP - TCP header present	
5	UDP - UDP header present	Reserved
6	SCTP - SCTP header present	Reserved
7	NFS - NFS header present	Reserved
10:8	Reserved	Reserved

RSV(4): Reserved.

HDR_LEN (10) - The length (bytes) of the header as parsed by the 82580. In split mode when HBO (Header Buffer Overflow) is set in the Extended error field, the *HDR_LEN* can be greater then zero though nothing is written to the header buffer. In header replication mode, the *HDR_LEN* field does not reflect the size of the data actually stored in the header buffer because the 82580 fills the buffer up to the size configured by *SRRCTL[n].BSIZEHEADER*, which might be larger than the header size reported here. This field is only valid in the first descriptor of a packet and should be ignored in all subsequent descriptors.

Note: When the packet is time stamped and the time stamp is placed at the beginning of the buffer the *RDESC.HDR_LEN* field is updated with the additional time stamp bytes (16 bytes). For further information see Section 7.1.10.

Packet types supported by the header split and header replication are listed in Appendix B.1. 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. If the bit is not set, a packet matching the header layout is not split.

Header split and replication is described in Section 7.1.9 while the packet types for this functionality are enabled by the *PSRTYPE[n]* registers (Section 8.10.3).

Note: The header of a fragmented IPv6 packet is defined before the fragmented extension header.

SPH (1) - Split Header - When set, indicates that the *HDR_LEN* field reflects the length of the header found by hardware. If cleared, the *HDR_LEN* field should be ignored, unless *SRRCTL[n].DESCTYPE* is set to *Split – always use header buffer* mode and *PKT_LEN* = 0. In this case, the *HDR_LEN* reflects the size of the packet, even if *SPH* bit is cleared.

In the case were *SRRCTL[n].DESCTYPE* is set to *Header replication mode*, *SPH* bit is set but the *HDR_LEN* field does not reflect the size of the data actually stored in the header buffer, because the 82580 fills the buffer up to the size configured by *SRRCTL[n].BSIZEHEADER*.

RSS Hash / Fragment Checksum (32)

This field has multiplexed functionality according to the received packet type (reported on the *Packet Type* field in this descriptor) and device setting.

Fragment Checksum (16-Bit; 63:48)

The fragment checksum word contains the unadjusted one's complement checksum of the IP payload and is used to offload checksum verification for fragmented UDP packets as described in Section 7.1.11.2. This field is mutually exclusive with the RSS hash. It is enabled when the *RXCSUM.PCSD* bit is cleared and the *RXCSUM.IPPCSE* bit is set.

IP identification (16-Bit; 47:32)

The IP identification word identifies the IP packet to whom this fragment belongs and is used to offload checksum verification for fragmented UDP packets as described in Section 7.1.11.2. This field is mutually exclusive with the RSS hash. It is enabled when the *RXCSUM.PCSD* bit is cleared and the *RXCSUM.IPPCSE* bit is set.

RSS Hash Value (32)

The RSS hash value is required for RSS functionality as described in Section 7.1.1.8. This bit is mutually exclusive with the fragment checksum. It is enabled when the *RXCSUM.PCSD* bit is set.

Extended Status (20)

Status information indicates whether the descriptor has been used and whether the referenced buffer is the last one for the packet. Table 7-14 lists the extended status word in the last descriptor of a packet (*EOP* is set). Table 7-15 lists the extended status word in any descriptor but the last one of a packet (*EOP* is cleared).

Table 7-14. Receive Status (RDESC.STATUS) Layout of the Last Descriptor

19	18	17	16	15	14	13	12	11	10
Rsv	Rsv	Rsv	TS	TSIP	Reserved		Strip CRC	LLINT	UDPV

VEXT	Rsv	PIF	IPCS	L4I	UDPCS	VP	Rsv	EOP	DD
9	8	7	6	5	4	3	2	1	0

Table 7-15. Receive Status (RDESC.STATUS) Layout of Non-Last Descriptor

19 2	1	0
Reserved	EOP = 0b	DD

TS (16) - Time Stamped Packet (Time Sync). The Time Stamp bit is set to indicate that the device recognized a Time Sync packet and time stamped it in the *RXSTMPL/H* time stamp registers (See Section 7.9.3.2 and Section 7.9.2.1).

TSIP (15) - Timestamp in packet. The Timestamp In Packet bit is set to indicate that the received packet arrival time was captured by the hardware and the timestamp was placed in the receive buffer. For further details see Section 7.1.10.

Reserved (2, 8, 14:13, 17, 18, 19) - Reserved at zero.

PIF (7), IPCS(6), UDPCS(4), VP(3), EOP (1), DD (0) - These bits are described in the legacy descriptor format in Section 7.1.4.

L4I (5) - This bit indicates that an L4 integrity check was done on the packet, either TCP checksum, UDP checksum or SCTP CRC checksum. This bit is valid only for the last descriptor of the packet. An error in the integrity check is indicated by the *L4E* bit in the error field. The type of check done can be induced from the packet type bits 4, 5 and 6. If bit 4 is set, a TCP checksum was done. If bit 5 is set a UDP checksum was done, and if bit 6 is set, a SCTP CRC checksum was done.

VEXT (9) - First VLAN is found on a double VLAN packet. This bit is valid only when *CTRL_EXT.EXT_VLAN* is set. For more details see Section 7.4.5.

UDPV (10) - This bit indicates that the incoming packet contains a valid (non-zero value) checksum field in an incoming fragmented UDP Ipv4 packet. This means that the *Fragment Checksum* field in the receive descriptor contains the UDP checksum as described in Section 7.1.11.2. When this field is cleared in the first fragment that contains the UDP header, means that the packet does not contain a valid UDP checksum and the checksum field in the Rx descriptor should be ignored. This field is always cleared in incoming fragments that do not contain the UDP header.

LLINT (11) - This bit indicates that the packet caused an immediate interrupt via the low latency interrupt mechanism.

Strip CRC (12) - This bit indicates that Ethernet CRC has been stripped from incoming packet. Strip CRC operation is defined by the *RCTL.SECRC* bit.

Extended Error (12)

Table 7-16 and the text that follows describes the possible errors reported by hardware.

Table 7-16. Receive Errors (RDESC.ERRORS) Layout

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

RXE (bit 11) - RXE is described in the legacy descriptor format in Section 7.1.4.

IPE (bit 10) - The IPE error indication is described in the legacy descriptor format in Section 7.1.4.

L4E (bit 9) - L4 error indication - When set, indicates that hardware attempted to do an L4 integrity check as described in the *L4I* bit, but the check failed.

Reserved (bits 8:7)

Reserved (bits 6:4)

HBO (bit 3) - Header Buffer Overflow

Note: The HBO bit is relevant only if *SPH* is set.

1. In both header replication modes, *HBO* is set if the header size (as calculated by hardware) is bigger than the allocated buffer size (*SRRCTL.BSIZEHEADER*) but the replication still takes place up to the header buffer size. Hardware sets this bit in order to indicate to software that it needs to allocate bigger buffers for the headers.
2. In header split mode, when *SRRCTL[n] BSIZEHEADER* is smaller than *HDR_LEN*, then *HBO* is set to 1b, In this case, the header is not split. Instead, the header resides within the host packet buffer. The *HDR_LEN* field is still valid and equal to the calculated size of the header. However, the header is not copied into the header buffer.
3. In header split mode, always use header buffer mode, when *SRRCTL[n] BSIZEHEADER* is smaller than *HDR_LEN*, then *HBO* is set to 1b. In this case, the header buffer is used as part of the data buffers and contains the first BSIZEHEADER bytes of the packet. The *HDR_LEN* field is still valid and equal to the calculated size of the header.

Note: Most error information appears only when the *store–bad–packet* bit (*RCTL.SBP*) is set and a bad packet is received.

Reserved (bits 2:0) - Reserved

PKT_LEN (16) – Number of bytes existing in the host packet buffer

The length covers the data written to a receive buffer including CRC bytes (if any). Software must read multiple descriptors to determine the complete length for packets that span multiple receive buffers. If *SRRCTL.DESC_TYPE* = 4 (advanced descriptor header replication large packet only) and the total packet length is smaller than the size of the header buffer (no replication is done), this field continues to reflect the size of the packet, although no data is written to the packet buffer. Otherwise, if the buffer is not split because the header is bigger than the allocated header buffer, this field reflects the size of the data written to the first packet buffer (header and data).

Note: When the packet is time stamped and the time stamp is placed at the beginning of the buffer, the *RDESC.PKT_LEN* field is updated with the additional time stamp bytes (16 bytes). For further information see Section 7.1.10.

VLAN Tag (16)

These bits are described in the legacy descriptor format in Section 7.1.4.

7.1.6 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 paragraphs briefly describe the descriptor fetch algorithm and the software control provided.

When the *RXDCTL[n].ENABLE* bit is set and the on-chip descriptor cache is empty, a fetch happens as soon as any descriptors are made available (Host increments the *RDT[n]* tail pointer). When the on-chip buffer is nearly empty (defined by *RXDCTL.PTHRESH*), a prefetch is performed each time enough valid descriptors (defined by *RXDCTL.HTHRESH*) are available in host memory.

When the number of descriptors in host memory is greater than the available on-chip descriptor cache, the 82580 might elect to perform a fetch that is not a multiple of cache-line size. Hardware performs this non-aligned fetch if doing so results in the next descriptor fetch being aligned on a cache-line boundary. This enables the descriptor fetch mechanism to be most efficient in the cases where it has fallen behind software.

All fetch decisions are based on the number of descriptors available and do not take into account any split of the transaction due to bus access limitations.

Note: The 82580 NEVER fetches descriptors beyond the descriptor tail pointer.

7.1.7 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. Descriptors write-backs accumulate and are opportunistically written out in cache line-oriented chunks, under the following scenarios:

- *RXDCTL[n].WTHRESH* descriptors have been used (the specified maximum threshold of unwritten used descriptors has been reached).
- The receive timer expires (*EITR*) - in this case all descriptors are flushed ignoring any cache-line boundaries.
- Explicit software flush (*RXDCTL.SWFLS*).
- Dynamic packets - if at least one of the descriptors that are waiting for write-back are classified as packets requiring immediate notification the entire queue is flushed out.

When the number of descriptors specified by *RXDCTL[n].WTHRESH* have been used, they are written back regardless of cache-line alignment. It is therefore recommended that *RXDCTL[n].WTHRESH* be a multiple of cache-line size. When the receive timer (*EITR*) expires, all used descriptors are forced to be written back prior to initiating the interrupt, for consistency. Software might explicitly flush accumulated descriptors by writing the *RXDCTL[n]* register with the *SWFLS* bit set.

When the 82580 does a partial cache-line write-back, it attempts to recover to cache-line alignment on the next write-back.

For applications where the latency of received packets is more important that the bus efficiency and the CPU utilization, an *EITR* value of zero may be used. In this case, each receive descriptor will be written to the host immediately. If *RXDCTL[n].WTHRESH* equals zero, then each descriptor will be written back separately, otherwise, write back of descriptors may be coalesced if descriptor accumulates in the internal descriptor ring due to bandwidth constrains.

All write-back decisions are based on the number of descriptors available and do not take into account any split of the transaction due to bus access limitations.

7.1.8 Receive Descriptor Ring Structure

Figure 7-9 shows the structure of each of the 8 receive descriptor rings. Hardware maintains 8 circular queues of descriptors and writes back used descriptors just prior to advancing the head pointer(s). Head and tail pointers wrap back to base when size descriptors have been processed.

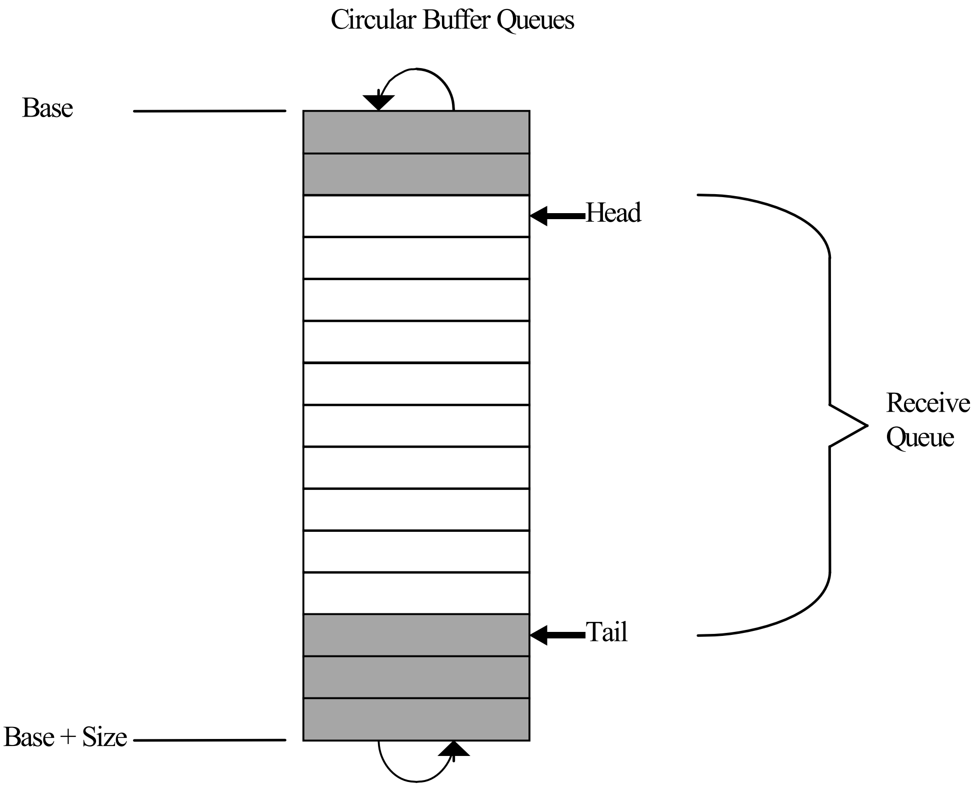


Figure 7-9. Receive Descriptor Ring Structure

Software inserts receive descriptors by advancing the tail pointer(s) to refer to the address of the entry just beyond the last valid descriptor. This is accomplished by writing the descriptor tail register(s) with the offset of the entry beyond the last valid descriptor. The hardware adjusts its internal tail pointer(s) accordingly. As packets arrive, they are stored in memory and the head pointer(s) is incremented by hardware. When the head pointer(s) is equal to the tail pointer(s), the queue(s) is empty. Hardware stops storing packets in system memory until software advances the tail pointer(s), making more receive buffers available.

The receive descriptor head and tail pointers reference to 16-byte blocks of memory. Shaded boxes in Figure 7-9 represent descriptors that have stored incoming packets but have not yet been recognized by software. Software can determine if a receive buffer is valid by reading the descriptors in memory. Any descriptor with a non-zero DD value has been processed by the hardware and is ready to be handled by the software.

Note: The head pointer points to the next descriptor that is written back. After the descriptor write-back operation completes, this pointer is incremented by the number of descriptors written back. Hardware owns all descriptors between [head... tail]. Any descriptor not in this range is owned by software.

The receive descriptor rings are described by the following registers:

- Receive Descriptor Base Address (*RDBA7* to *RDBA0*) register:

  This register indicates the start of the descriptor ring buffer. This 64-bit address is aligned on a 16-byte boundary and is stored in two consecutive 32-bit registers. Note that hardware ignores the lower 4 bits.

- Receive Descriptor Length (*RDLEN7* to *RDLEN0*) registers:

  This register determines the number of bytes allocated to the circular buffer. This value must be a multiple of 128 (the maximum cache-line size). Since each descriptor is 16 bytes in length, the total number of receive descriptors is always a multiple of eight.

- Receive Descriptor Head (*RDH7* to *RDH0*) registers:

  This register holds a value that is an offset from the base and indicates the in-progress descriptor. There can be up to 64 KB, 8 KB descriptors in the circular buffer. Hardware maintains a shadow copy that includes those descriptors completed but not yet stored in memory.

- Receive Descriptor Tail (*RDT7* to *RDT0*) registers:

  This register holds a value that is an offset from the base and identifies the location beyond the last descriptor hardware can process. This is the location where software writes the first new descriptor.

If software statically allocates buffers, uses legacy receive descriptors, and uses memory read to check for completed descriptors, it has to zero the status byte in the descriptor before bumping the tail pointer to make it ready for reuse by hardware. Zeroing the status byte is not a hardware requirement but is necessary for performing an in-memory scan.

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

7.1.8.1 Low Receive Descriptors Threshold

As described above, the size of the receive queues is measured by the number of receive descriptors. During run time the software processes completed descriptors and then increments the Receive Descriptor Tail registers (*RDT*). At the same time, the hardware may post new packets received from the LAN incrementing the Receive Descriptor Head registers (*RDH*) for each used descriptor.

The number of usable (free) descriptors for the hardware is the distance between Tail and Head registers. When the Tail reaches the Head, there are no free descriptors and further packets may be either dropped or block the receive FIFO. In order to avoid this behavior, the 82580 may generate a low latency interrupt (associated with the relevant receive queue) once the amount of free descriptors is less or equal than the threshold. The threshold is defined in 16 descriptors granularity per queue in the *SRRCTL[n].RDMTS* field.

7.1.9 Header Splitting and Replication

7.1.9.1 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 8.13) or in conjunction with a software-prefetch.

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

The 82580 supports header splitting in several modes:

- Legacy mode: legacy descriptors are used; headers and payloads are not split.
- Advanced mode, no split: advanced descriptors are in use; header and payload are not split.
- Advanced mode, split: advanced descriptors are in use; header and payload are split to different buffers. If the packet cannot be split, only the packet buffer is used.
- Advanced mode, replication: advanced descriptors are in use; header is replicated in a separate buffer and also in a payload buffer.
- Advanced mode, replication, conditioned by packet size: advanced descriptors are in use; replication is performed only if the packet is larger than the header buffer size.
- Advanced mode, split, always use header buffer: advanced descriptors are in use; header and payload are split to different buffers. If no split is done, the first part of the packet is stored in the header buffer.

7.1.9.2 Description

In Figure 7-10 and Figure 7-11, the header splitting and header replication modes are shown.

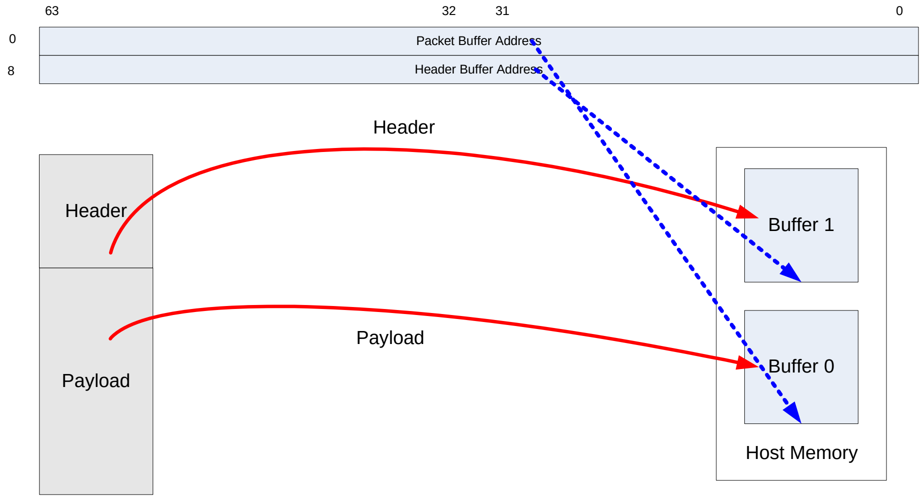


Figure 7-10. Header Splitting

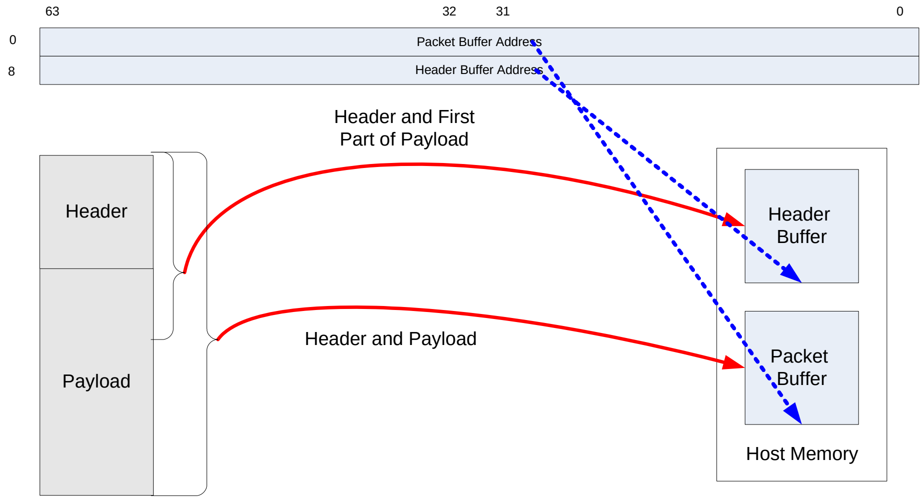


Figure 7-11. Header Replication

The physical address of each buffer is written in the *Buffer Addresses* fields. The sizes of these buffers are statically defined by *BSIZEPACKET* and *BSIZEHEADER* fields in the *SRRCTL[n]* registers.

The packet buffer address includes the address of the buffer assigned to the replicated packet, including header and data payload portions of the received packet. In the case of a split header, only the payload is included.

The header buffer address includes the address of the buffer that contains the header information. The receive DMA module stores the header portion of the received packets into this buffer.

The 82580 uses the packet replication or splitting feature when the SRRCTL[n].DESCTYPE is larger than one. The software device driver must also program the buffer sizes in the SRRCTL[n] registers.

When header split is selected, the packet is split only on selected types of packets. A bit exists for each option in PSRTYPE[n] registers so several options can be used in conjunction with them. If one or more bits are set, the splitting is performed for the corresponding packet type.

The following table lists the behavior of the 82580 in the different modes

Table 7-17. 82580 Split/Replicated Header Behavior

DESCTYPE	Condition	SPH	HBO	PKT_LEN	HDR_LEN	Header and Payload DMA
Split	1. Header can't be decoded	0b	0b	Min(Packet length, Buffer size)	N/A	Header + Payload ⇨ Packet buffer
	2. Header <= *BSIZEHEADER*	1b	0b	Min(Payload length, Buffer size)[1]	Header size	Header ⇨ Header buffer Payload ⇨ Packet buffer
	3. Header > *BSIZEHEADER*	1b	1b	Min(Packet length, Buffer size)	Header size[2]	Header + Payload ⇨ Packet buffer

Table 7-17. 82580 Split/Replicated Header Behavior

DESCTYPE	Condition	SPH	HBO	PKT_LEN	HDR_LEN	Header and Payload DMA
Split – always use header buffer	1. Packet length <= *BSIZEHEADER*	0b	0b	Zero	Packet length	Header + Payload ⇨ Header buffer
	2. Header can't be Decoded and Packet length > *BSIZEHEADER*	0b	0b	Min(Packet length – *BSIZEHEADER*, Data Buffer size)	BSIZEHEADER	Header + Payload ⇨ Header + Packet buffers[3]
	3. Header <= *BSIZEHEADER* and Packet length >= *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[2]	Header + Payload ⇨ Header + Packet buffer[3]
Replicate Large Packet only	1. Header + Payload <= *BSIZEHEADER*	0b/1b[4]	0b	Packet length	Header size, N/A[4]	Header + Payload ⇨ Header buffer
	2. Header + Payload > *BSIZEHEADER*	0b/1b[4]	0b/1b[5]	Min(Packet length, Buffer size)	Header size, N/A[4]	(Header + Payload) (partial[6]) ⇨ Header buffer Header + Payload ⇨ Packet buffer

[1] In a header only packet (such as TCP ACK packet), the PKT_LEN is zero.

[2] The HDR_LEN doesn't reflect the actual data size stored in the Header buffer. It reflects the header size determined by the parser. When timestamp in packet is enabled header size reflects the additional 16 bytes of the timestamp.

[3] If the packet spans more than one descriptor, only the header buffer of the first descriptor is used. The header buffer is used for the first part of the packet until it is filled up, and then the first packet buffer is used for the continuation of the packet.

Software Notes:

- If *SRRCTL[n].NSE* is set, all buffers' addresses in a packet descriptor must be word aligned.
- Packet header can't span across buffers, therefore, the size of the header buffer must be larger than any expected header size. Otherwise, only the part of the header fitting the header buffer is replicated. In the case of header split mode (*SRRCTL[n].DESCTYPE* = 010b), a packet with a header larger than the header buffer is not split.
- Section B.1 describes the details of the split/replicate conditions for different types of headers according to the settings of the *PSRTYPE* register values.

7.1.10 Receive Packet Timestamp in Buffer

The 82580 supports adding an optional tailored header before the MAC header of the packet in the receive buffer. The 64 MSB bits of the 128 bit tailored header include a timestamp composed of the packet reception time measured in the *SYSTIML* (Low DW) and *SYSTIMH* (High DW) registers (See Section 7.9.3.1 for further information on SYSTIML/H operation). The 64 LSB bits of the tailored header are reserved.

The timestamp information is placed in Networking order (Big Endian) format as depicted in Table 7-18.

Table 7-18. Timestamp Layout in Buffer

0 3	4 7	8 11	12 15	16...
Reserved (0x0)	Reserved (0x0)	SYSTIMH	SYSTIML	Received Packet

When the *TSAUXC.Disable systime* bit is cleared and the *SRRCTL[n].Timestamp* bit is set to 1, packets received to the queue will be time stamped if they meet one of the following conditions:

- — Meet the criteria defined in the *TSYNCRXCTL.Type* field (See Section 8.17.1 and Section 8.17.26).
- — Match the value defined in one of the *ETQF* registers with the *1588 time stamp* bit set (See Section 7.1.1.4).
- — Match a 2-tuple filter with the *TTQF.1588 time stamp* set (See Section 7.1.1.5).

When detecting a receive packet that should be time stamped, the 82580 will:

- Place a 64 bit timestamp, indicating the time a packet was received by the MAC, at the beginning of the receive buffer before the received packet.
- Set the *TSIP* bit in the *RDESC.STATUS* field of the last receive descriptor.
- Update the *RDESC.Packet Type* field in the last receive descriptor. Value in this field enables identifying that this is a PTP (Precision Time Protocol) packet (this indication is only relevant for L2 packets).
- Update the *RDESC.HDR_LEN* and *RDESC.PKT_LEN* values to include size of timestamp.

Software driver should take into account the additional size of the timestamp when preparing the receive descriptors for the relevant queue.

7.1.11 Receive Packet Checksum Off Loading

The 82580 supports the off loading of three receive checksum calculations: the packet checksum, the IPv4 header checksum, and the TCP/UDP checksum.

The packet checksum is the one's complement over the receive packet, starting from the byte indicated by *RXCSUM.PCSS* (zero corresponds to the first byte of the packet), after stripping. For packets with a VLAN header, the packet checksum includes the header if VLAN striping is not enabled by the *CTRL.VME*. If a VLAN header strip is enabled, the packet checksum and the starting offset of the packet checksum exclude the VLAN header due to masking of VLAN header. For example, for an Ethernet II frame encapsulated as an 802.3ac VLAN packet and *CTRL.VME* is set and with *RXCSUM.PCSS* set to 14, the packet checksum would include the entire encapsulated frame, excluding the 14-byte Ethernet header (DA, SA, type/length) and the 4-byte q-tag. The packet checksum does not include the Ethernet CRC if the *RCTL.SECRC* bit is set.

Software must make the required offsetting computation (to remove the bytes that should not have been included and to include the pseudo-header) prior to comparing the packet checksum against the TCP checksum stored in the packet.

For supported packet/frame types, the entire checksum calculation can be off loaded to the 82580. If *RXCSUM.IPOFL* is set to 1b, the 82580 calculates the IPv4 checksum and indicates a pass/fail indication to software via the IPv4 *Checksum Error* bit (*RDESC.IPE*) in the *Error* field of the receive descriptor. Similarly, if *RXCSUM.TUOFL* is set to 1b, the 82580 calculates the TCP or UDP checksum and indicates a pass/fail condition to software via the TCP/UDP *Checksum Error* bit (*RDESC.L4E*). These error bits are valid when the respective status bits indicate the checksum was calculated for the packet (*RDESC.IPCS* and *RDESC.L4CS*, respectively). Similarly, if *RFCTL.Ipv6_DIS* and *RFCTL.IP6Xsum_DIS* are cleared to 0b and *RXCSUM.TUOFL* is set to 1b, the 82580 calculates the TCP or UDP checksum for IPv6 packets. It then indicates a pass/fail condition in the TCP/UDP *Checksum Error* bit (*RDESC.L4E*).

If neither *RXCSUM.IPOFL* nor *RXCSUM.TUOFL* are set, the *Checksum Error* bits (*IPE* and **L4E**) are 0b for all packets.

Supported frame types:

- Ethernet II

- Ethernet SNAP

Table 7-19. Supported Receive Checksum Capabilities

Packet Type	Hardware IP Checksum Calculation	Hardware TCP/UDP Checksum Calculation
IPv4 packets.	Yes	Yes
IPv6 packets.	No (n/a)	Yes
IPv6 packet with next header options:		
• Hop-by-hop options	No (n/a)	Yes
• Destinations options (without Home option)	No (n/a)	Yes
• Destinations options (with Home option)	No (n/a)	No
• Routing (with Segments Left zero)	No (n/a)	Yes
• Routing (with Segments Left > zero)	No (n/a)	No
• Fragment	No (n/a)	No
IPv4 tunnels:		
• IPv4 packet in an IPv4 tunnel.	No	No
• IPv6 packet in an IPv4 tunnel.	Yes (IPv4)	Yes[1]
IPv6 tunnels:		
• IPv4 packet in an IPv6 tunnel.	No	No
• IPv6 packet in an IPv6 tunnel.	No	No
Packet is an IPv4 fragment.	Yes	No
Packet is greater than 1552 bytes; (LPE=1b).	Yes	Yes
Packet has 802.3ac tag.	Yes	Yes
IPv4 packet has IP options (IP header is longer than 20 bytes).	Yes	Yes
Packet has TCP or UDP options.	Yes	Yes
IP header’s protocol field contains a protocol number other than TCP or UDP.	Yes	No

[1] The IPv6 header portion can include supported extension headers as described in the IPv6 filter section.

7.1.11.1 Filters details

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:

7.1.11.1.1 MAC Address Filter

This filter checks the MAC destination address to be sure it is valid (such as IA match, broadcast, multicast, etc.). The receive configuration settings determine which MAC addresses are accepted. See the various receive control configuration registers such as *RCTL* (*RCTL.UPE*, *RCTL.MPE*, *RCTL.BAM*), MTA, RAL, and RAH.

7.1.11.1.2 SNAP/VLAN Filter

This filter checks the next headers looking for an IP header. It is capable of decoding Ethernet II, Ethernet SNAP, and IEEE 802.3ac headers. It skips past any of these intermediate headers and looks for the IP header. The receive configuration settings determine which next headers are accepted. See the various receive control configuration registers such as *RCTL* (*RCTL.VFE*), *VET*, and *VFTA*.

7.1.11.1.3 IPv4 Filter

This filter checks for valid IPv4 headers. The version field is checked for a correct value (4).

IPv4 headers are accepted if they are any size greater than or equal to five (Dwords). If the IPv4 header is properly decoded, the IP checksum is checked for validity. The *RXCSUM.IPOFL* bit must be set for this filter to pass.

7.1.11.1.4 IPv6 Filter

This filter checks for valid IPpv6 headers, which are a fixed size and have no checksum. The IPv6 extension headers accepted are: hop-by-hop, destination options, and routing. The maximum size next header accepted is 16 Dwords (64 bytes).

7.1.11.1.5 IPv6 Extension Headers

IPv4 and TCP provide header lengths, which enable hardware to easily navigate through these headers on packet reception for calculating checksum and CRCs, etc. For receiving IPv6 packets; however, there is no IP header length to help hardware find the packet's ULP (such as TCP or UDP) header. One or more IPv6 extension headers might exist in a packet between the basic IPv6 header and the ULP header. The hardware must skip over these extension headers to calculate the TCP or UDP checksum for received packets.

The IPv6 header length without extensions is 40 bytes. The IPv6 field *Next Header Type* indicates what type of header follows the IPv6 header at offset 40. It might be an upper layer protocol header such as TCP or UDP (*Next Header Type* of 6 or 17, respectively), or it might indicate that an extension header follows. The final extension header indicates with its *Next Header Type* field the type of ULP header for the packet.

IPv6 extension headers have a specified order. However, destinations must be able to process these headers in any order. Also, IPv6 (or IPv4) might be tunneled using IPv6, and thus another IPv6 (or IPv4) header and potentially its extension headers might be found after the extension headers.

The IPv4 *Next Header Type* is at byte offset nine. In IPv6, the first *Next Header Type* is at byte offset six.

All IPv6 extension headers have the *Next Header Type* in their first eight bits. Most have the length in the second eight bits (Offset Byte[1]) as shown:

Table 7-20. Typical IPv6 Extended Header Format (Traditional Representation)

	1	2 3
0 1 2 3 4 5 6 7	8 9 0 1 2 3 4 5	6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Next Header Type	Length	

The following table lists the encoding of the *Next Header Type* field and information on determining each header type's length. The IPv6 extension headers are not otherwise processed by the 82580 so their details are not covered here.

Table 7-21. Header Type Encoding and Lengths

Header	Next Header Type	Header Length (Units are Bytes Unless Otherwise Specified)
IPv6	6	Always 40 bytes
IPv4	4	Offset Bits[7:4] Unit = 4 bytes
TCP	6	Offset Byte[12].Bits[7:4] Unit = 4 bytes
UDP	17	Always 8 bytes
Hop by Hop Options	0 (Note 1)	8+Offset Byte[1]
Destination Options	60	8+Offset Byte[1]
Routing	43	8+Offset Byte[1]
Fragment	44	Always 8 bytes
Authentication	51	8+4*(Offset Byte[1])
Encapsulating Security Payload	50	Note 3
No Next Header	59	Note 2

Notes:

1. Hop-by-hop options header is only found in the first *Next Header Type* of an IPv6 header.
2. When a *No Next Header* type is encountered, the rest of the packet should not be processed.
3. Encapsulated security payload - the 82580 cannot offload packets with this header type.

Note that the 82580 hardware acceleration does not support all IPv6 extension header types (refer to Table 7-19).

Also, the *RFCTL.Ipv6_DIS* bit must be cleared for this filter to pass.

7.1.11.1.6 UDP/TCP Filter

This filter checks for a valid UDP or TCP header. The prototype next header values are 0x11 and 0x06, respectively. The *RXCSUM.TUOFL* bit must be set for this filter to pass.

7.1.11.2 Receive UDP Fragmentation Checksum

The 82580 might provide receive fragmented UDP checksum offload. The 82580 should be configured in the following manner to enable this mode:

The *RXCSUM.PCSD* bit should be cleared. The *Packet Checksu*m and *IP Identification* fields are mutually exclusive with the RSS hash. When the RXCSUM.*PCSD* bit is cleared, *Packet Checksum* and *IP Identification* are active instead of RSS hash.

The *RXCSUM.IPPCSE* bit should be set. This field enables the IP payload checksum enable that is designed for the fragmented UDP checksum.

The *RXCSUM.PCSS* field must be zero. The packet checksum start should be zero to enable auto-start of the checksum calculation. The following table lists the exact description of the checksum calculation.

The following table also lists the outcome descriptor fields for the following incoming packets types:

Table 7-22. Descriptor Fields

Incoming Packet Type	Fragment Checksum	UDPV	UDPCS / L4CS
Non IP Packet	0b	0b	0b / 0b
Ipv6 Packet	0b	0b	Depends on transport header.
Non fragmented Ipv4 packet	0b	0b	Depends on transport header.
Fragmented Ipv4, when not first fragment	The unadjusted one's complement checksum of the IP payload.	0b	1b / 0b
Fragmented Ipv4, for the first fragment	Same as above	1 if the UDP header checksum is valid (not zero)	1b / 0b

Note: When the software device driver computes the 16-bit ones complement, the sum on the incoming packets of the UDP fragments, it should expect a value of 0xFFFF. Refer to Section 7.1.11 for supported packet formats.

7.1.12 SCTP Offload

If a receive packet is identified as SCTP, the 82580 checks the CRC32 checksum of this packet and identifies this packet as SCTP. Software is notified of the CRC check via the *L4I* bit in the *Extended Status* field of the Rx descriptor. The detection of a SCTP packet is indicated via the *SCTP* bit in the *packet Type* field of the Rx descriptor. The checker assumes the following SCTP packet format:

Table 7-23. SCTP Header

	1	2	3
0 1 2 3 4 5 6 7	8 9 0 1 2 3 4 5	6 7 8 9 0 1 2 3	4 5 6 7 8 9 0 1
Source Port		Destination Port	
Verification Tag			
Checksum			
Chunks 1...n			

7.2 Transmit Functionality

7.2.1 Packet Transmission

Output packets to be transmitted are created using pointer-length pairs constituting a descriptor chain (descriptor based transmission). Software forms transmit packets by assembling the list of pointer-length pairs, storing this information in one of the transmit descriptor rings, and then updating the adequate on-chip transmit tail pointer. The transmit descriptors and buffers are stored in host memory. Hardware typically transmits the packet only after it has completely fetched all the packet data from host memory and stored it into the on-chip transmit FIFO (store and forward architecture). This permits TCP or UDP checksum computation and avoids problems with PCIe under-runs. Another transmit feature of the 82580 is TCP/UDP segmentation. The hardware has the capability to perform packet segmentation on large data buffers offloaded from the Network Stack. This feature is discussed in detail in Section 7.2.4.

In addition, the 82580 supports SCTP offloading for transmit requests. See section Section 7.2.5.3 for details about SCTP.

Table 1-9 provides a high level description of all data/control transformation steps needed for sending Ethernet packets to the line.

7.2.1.1 Transmit Data Storage

Data is stored in buffers pointed to by the descriptors. The data can be aligned to arbitrary byte boundary with the maximum size per descriptor limited only to the maximum allowed packet size (9728 bytes). A packet typically consists of two (or more) buffers, one (or more) for the header and one for the actual data. Each buffer is referenced by a different descriptor. Some software implementations may copy the header(s) and packet data into one buffer and use only one descriptor per transmitted packet.

7.2.1.2 On-Chip Transmit Buffers

The 82580 allocates by default a 20KB on-chip packet buffer per port. The buffers are used to store packets until they are transmitted on the line. The 82580 utilizes a common memory structure for the on-chip transmit buffers allocated to the various ports. If a port is disabled, so that it can't be accessed by host and management, the freed buffer space can be allocated to the active ports via the "*Initialization Control 4*" EEPROM word. A port can be disabled by either:

1. Pin assertion (LAN0_DIS_N, LAN1_DIS_N, LAN2_DIS_N, LAN3_DIS_N for port 0 to 3 respectively) and setting the EEPROM bit *PHY_in_LAN_disable* in the "*Software Defined Pins Control*" word to 1 for the relevant port.
2. Setting EEPROM bit *LAN_DIS* in the "*Software Defined Pins Control*" word for the relevant port to 1.

Actual on-chip transmit buffer allocated to the port can be read in the *ITPBS register.*

7.2.1.3 On-Chip descriptor Buffers

The 82580 contains a 24 descriptor cache for each transmit queue used to reduce the latency of packet processing and to optimize the usage of the PCIe bandwidth by fetching and writing back descriptors in bursts. The fetch and writeback algorithm are described in Section 7.2.2.5 and Section 7.2.2.6.

7.2.1.4 Transmit Contexts

The 82580 provides hardware checksum offload and TCP/UDP segmentation facilities. These features enable TCP and UDP packet types to be handled more efficiently by performing additional work in hardware, thus reducing the software overhead associated with preparing these packets for transmission. Part of the parameters used by these features is handled though context descriptors.

A context descriptor refers to a set of device registers loaded or accessed as a group to provide a particular function. The 82580 supports 2x8 context descriptor sets (two per queue) per port on-chip. The transmit queues can contain transmit data descriptors (similar to the receive queue) as well as transmit context descriptors.

The contexts are queue specific and one context cannot be reused from one queue to another. This differs from the method used in previous devices that supported a pool of contexts to be shared between queues.

A transmit context descriptor differs from a data descriptor as it does not point to packet data. Instead, this descriptor provides the ability to write to the on-chip context register sets that support the transmit checksum offloading and the segmentation features of the 82580.

The 82580 supports one type of transmit context. This on-chip context is written with a transmit context descriptor DTYP=2 and is always used as context for transmit data descriptor DTYP=3.

The *IDX* field contains an index to one of the two queue contexts. Software must track what context is stored in each *IDX* location.

Each advanced data descriptor that uses any of the advanced offloading features must refer to a context.

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

Software is responsible for ensuring that a context is only overwritten when it is no longer needed. Hardware does not include any logic to manage the on-chip contexts; it is completely up to software to populate and then use the on-chip context table.

Note: Software should not queue more than 2 context descriptors in sequence without an intervening data descriptor, to achieve adequate performance.

Each context defines information about the packet sent including the total size of the MAC header (*TDESC.MACHDR*), the maximum amount of payload data that should be included in each packet (*TDESC.MSS*), TCP header length (*TDES.TCPHDR*), IP header length (*TDESC.IPHDR*), and information about what type of protocol (TCP, IP, etc.) is used. Other than TCP, IP (*TDESC.TUCMD*), most information is specific to the segmentation capability.

Because there are dedicated on-chip resources for contexts, they remain constant until they are modified by another context descriptor. This means that a context can be used for multiple packets (or multiple segmentation blocks) unless a new context is loaded prior to each new packet. Depending on the environment, it might be unnecessary to load a new context for each packet. For example, if most traffic generated from a given node is standard TCP frames, this context could be setup once and used for many frames. Only when some other frame type is required would a new context need to be loaded by software. This new context could use a different index or the same index.

This same logic can also be applied to the TCP/UDP segmentation scenario, though the environment is a more restrictive one. In this scenario, the host is commonly asked to send messages of the same type, TCP/IP for instance, and these messages also have the same Maximum Segment Size (MSS). In this instance, the same context could be used for multiple TCP messages that require hardware segmentation.

7.2.2 Transmit Descriptors

The 82580 supports legacy descriptors and the 82580 advanced descriptors.

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

These descriptors should not be used when advanced features such as virtualization are used. The Legacy descriptors are recognized as such based on the *DEXT* bit as discussed later in this section.

In addition, the 82580 supports two types of advanced transmit descriptors:

1. Advanced Transmit Context Descriptor, DTYP = 0010b.
2. Advanced Transmit Data Descriptor, DTYP = 0011b.

Note: DTYP values 0000b and 0001b are reserved.

The transmit data descriptor (both legacy and advanced) points to a block of packet data to be transmitted. The advanced transmit context descriptor does not point to packet data. It contains control/context information that is loaded into on-chip registers that affect the processing of packets for transmission. The following sections describe the descriptor formats.

7.2.2.1 Legacy Transmit Descriptor Format

Legacy descriptors are identified by having bit 29 of the descriptor (*TDESC.DEXT*) set to 0b. In this case, the descriptor format is defined as shown in Table 7-24. Note that the address and length must be supplied by software. Also note that bits in the command byte are optional, as are the CSO, and CSS fields.

Table 7-24. Transmit Descriptor (TDESC) Fetch Layout - Legacy Mode

	63 48	47 40	39 36	35 32	31 24	23 16	15 0
0	Buffer Address [63:0]						
8	VLAN	CSS	Reserved	STA	CMD	CSO	Length

Table 7-25. Transmit Descriptor (TDESC) Write-Back Layout - Legacy Mode

	63 48	47 40	39 36	35 32	31 24	23 16	15 0
0	Reserved				Reserved		
8	VLAN	CSS	Reserved	STA	CMD	CSO	Length

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

7.2.2.1.1 Buffer Address (64)

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

7.2.2.1.2 Length

Length (*TDESC.LENGTH*) specifies the length in bytes to be fetched from the buffer address provided.

The maximum length associated with any single legacy descriptor is 9728 bytes.

Descriptor length(s) might be limited by the size of the transmit FIFO. All buffers comprising a single packet must be able to be stored simultaneously in the transmit FIFO. For any individual packet, the sum of the individual descriptors' lengths must be below 9728 bytes.

Note: The maximum allowable packet size for transmits can change, based on the value written to the *DMA TX Max Allowable packet size* (*DTXMXPKTSZ*) register.

Note: Descriptors with zero length (null descriptors) transfer no data. Null descriptors can only appear between packets and must have their *EOP* bits set.

Note: If the *TCTL.PSP* bit is set, the total length of the packet transmitted, not including FCS should be at least 17 bytes.

7.2.2.1.3 Checksum Offset and Start - CSO and CSS

A *Checksum Offset* (*TDESC.CSO*) field indicates where, relative to the start of the packet, to insert a TCP checksum if this mode is enabled. A *Checksum Start* (*TDESC.CSS*) field indicates where to begin computing the checksum.

Both CSO and CSS are in units of bytes and must be in the range of data provided to the 82580 in the descriptors. For short packets that are not padded by software, CSS and CSO must be in the range of the unpadded data length, not the eventual padded length (64 bytes). CSO must be larger than CSS, CSS must be equal or greater than 14 bytes, and CSO must be smaller than the packet length minus four bytes. Checksum calculation is not done if CSO or CSS are out of range. This occurs if (CSS > length) OR (CSO > length - 1).

In the case of an 802.1Q header, the offset values depend on the VLAN insertion enable (*VLE*) bit. If it is not set (VLAN tagging included in the packet buffers), the offset values should include the VLAN tagging. If this bit is set (VLAN tagging is taken from the packet descriptor), the offset values should exclude the VLAN tagging.

Note: Assuming CSS points to the beginning of the IP header, software must compute an offsetting entry to back out the bytes of the header that are not part of the IP pseudo header and should not be included in the TCP checksum and store it in the position where the hardware computed checksum is to be inserted. Hardware does not add the 802.1Q Ethertype or the VLAN field following the 802.1Q Ethertype to the checksum. So for VLAN packets, software can compute the values to back out only the encapsulated IP header packet and not the added fields.

Note: UDP checksum calculation is not supported by the legacy descriptors. When using legacy descriptors the 82580 is not aware of the L4 type of the packet and thus, does not support the translation of a checksum result of 0x0000 to 0xFFFF needed to differentiate between an UDP packet with a checksum of zero and an UDP packet without checksum.

Because the *CSO* field is eight bits wide, it puts a limit on the location of the checksum to 255 bytes from the beginning of the packet.

Hardware adds the checksum to the field at the offset indicated by the *CSO* field. Checksum calculations are for the entire packet starting at the byte indicated by the *CSS* field. A value of zero corresponds to the first byte in the packet.

CSS must be set in the first descriptor of the packet.

Table 7-26. Transmit Command (TDESC.CMD) Layout

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

7.2.2.1.4 Command Byte - CMD

The CMD byte stores the applicable command and has the fields shown in Figure 7-26.

- 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

VLE: Indicates that the packet is a VLAN packet. For example, hardware should add the VLAN Ethertype and an 802.1q VLAN tag to the packet.

Table 7-27. VLAN Tag Insertion Decision Table

VLE	Action
0b	Send generic Ethernet packet.
1b	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.

RS: Signals the hardware to report the status information. This is used by software that does in-memory checks of the transmit descriptors to determine which ones are done. For example, if software queues up 10 packets to transmit, it can set the *RS* bit in the last descriptor of the last packet. If software maintains a list of descriptors with the *RS* bit set, it can look at them to determine if all packets up to (and including) the one with the *RS* bit set have been buffered in the output FIFO. Looking at the status byte and checking the *Descriptor Done* (*DD*) bit enables this operation. If *DD* is set, the descriptor has been processed. Refer to Table 7-28 for the layout of the status field.

IC: If set, requests hardware to add the checksum of the data from *CSS* to the end of the packet at the offset indicated by the *CSO* field.

IFCS: When set, hardware appends the MAC FCS at the end of the packet. When cleared, software should calculate the FCS for proper CRC check. There are several cases in which software must set IFCS:

- Transmitting a short packet while padding is enabled by the *TCTL.PSP* bit.
- Checksum offload is enabled by the *IC* bit in the TDESC.CMD.
- VLAN header insertion enabled by the *VLE* bit in the TDESC.CMD.

EOP: When set, indicates this is the last descriptor making up the packet. Note that more than one descriptor can be used to form a packet.

Note: the 8257, 1VLE, IFCS, CSO, and IC must be set correctly only in the first descriptor of each packet. In previous silicon generations, some of these bits were required to be set in the last descriptor of a packet.

7.2.2.1.5 Status – STA

Table 7-28. Transmit Status (TDESC.STA) Layout

3	2	1	0
Reserved			DD

7.2.2.1.6 DD (Bit 0) - Descriptor Done Status

The DD bit provides the transmit status, when *RS* is set in the command: *DD* indicates that the descriptor is done and is written back after the descriptor has been processed.

Note: When head write back is enabled (TDWBAL[n].Head_WB_En = 1), there is no write-back of the DD bit to the descriptor.When using legacy Tx descriptors, Head writeback should not be enabled (*TDWBAL[n].Head_WB_En* = 0).

7.2.2.1.7 VLAN

The *VLAN* field is used to provide the 802.1q/802.1ac tagging information. The *VLAN* field is valid only on the first descriptor of each packet when the *VLE* bit is set. The rule for VLAN tag is to use network ordering. The VLAN field is placed in the transmit descriptor in the following manner:

Table 7-29. VLAN Field (TDESC.VLAN) Layout

15 13	12	11 0
PRI	CFI	VLAN ID

- VLAN ID - the 12-bit tag indicating the VLAN group of the packet.
- Canonical Form Indication (CFI) - Set to zero for Ethernet packets.
- PRI - indicates the priority of the packet.

Note: The VLAN tag is sent in network order (also called big endian).

7.2.2.2 Advanced Transmit Context Descriptor

Table 7-30. Transmit Context Descriptor (TDESC) Layout - (Type = 0010b)

	63 40	39 32	31 16	15 9	8 0
0	Reserved	Reserved	VLAN	MACLEN	IPLEN

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

[1] RSV - Reserved

7.2.2.2.1 IPLEN (9)

IP header length. If an offload is requested, IPLEN must be greater than or equal to 20 and less than or equal to 511.

7.2.2.2.2 MACLEN (7)

This field indicates the length of the MAC header. When an offload is requested (either TSE or IXSM or TXSM is set), MACHDR must be larger than or equal to 14 and less than or equal to 127. This field should include only the part of the L2 header supplied by the software device driver and not the parts added by hardware. The following table lists the value of MACLEN in the different cases.

Table 7-31. MACLEN Values

SNAP	Regular VLAN	External VLAN	MACLEN
No	By hardware or no VLAN	No	14
No	By hardware or no VLAN	Yes	18
No	By software	No	18
No	By software	Yes	22
Yes	By hardware or no VLAN	No	22
Yes	By hardware or no VLAN	Yes	26
Yes	By software	No	26
Yes	By software	Yes	30

VLAN (16) - 802.1Q VLAN tag to be inserted in the packet during transmission. This VLAN tag is inserted and needed only when a packet using this context has its *DCMD.VLE* bit set. This field should include the entire 16-bit *VLAN* field including the *CFI* and *Priority* fields as shown in Table 7-29.

Note: The VLAN tag is sent in network order.

7.2.2.2.3 TUCMD (11)

Table 7-32. Transmit Command (TDESC.TUCMD) Layout

10 6	5	4	3 2	1	0
Reserved	Reserved	Reserved	L4T	IPV4	SNAP

- RSV (bit 10:6) - Reserved
- RSV (bit 5:4) - Reserved
- L4T (bit 3:2) - L4 Packet TYPE (00b: UDP; 01b: TCP; 10b: SCTP; 11b: Reserved)
- IPV4 (bit 1) - IP Packet Type: When 1b, Ipv4; when 0b, Ipv6
- SNAP (bit 0) - SNAP indication

7.2.2.2.4 DTYP(4)

Always 0010b for this type of descriptor.

7.2.2.2.5 DEXT(1)

Descriptor Extension (1b for advanced mode).

7.2.2.2.6 IDX (3)

Index into the hardware context table where this context is stored. In the 82580 the 2 available register context sets per queue are accessed using the LSB bit and the two MSB bits are reserved and should always be 0.

7.2.2.2.7 L4LEN (8)

Layer 4 header length. If *TSE* is set in the data descriptor pointing to this context, this field must be greater than or equal to 12 and less than or equal to 255. Otherwise, this field is ignored.

7.2.2.2.8 MSS (16)

Controls the Maximum Segment Size (MSS). This specifies the maximum TCP payload segment sent per frame, not including any header or trailer. The total length of each frame (or section) sent by the TCP/UDP segmentation mechanism (excluding Ethernet CRC) as follows:

Total length is equal to:

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

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

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

Note: The headers lengths must meet the following:

MACLEN + IPLEN + L4LEN <= 512

Note: The maximum MSS value should not exceed 9216 bytes (9KB) length.

The context descriptor requires valid data only in the fields used by the specific offload options. The following table lists the required valid fields according to the different offload options.

Table 7-33. Valid Field in Context vs. Required Offload

Required Offload			Valid Fields in Context						
TSE	**TXSM**	**IXSM**	**VLAN**	**L4LEN**	**IPLEN**	**MACLEN**	**MSS**	**L4T**	**IPV4**
1b[1]	1b	X[2]	VLE	Yes	Yes	Yes	Yes	Yes	Yes
0b	1b	X[2]	VLE	No	Yes	Yes	No	Yes	Yes
0b	0b	1b	VLE	No	Yes	Yes	No	No	Yes
0b	0b	0b	No context required unless VLE is set.						

1. If TSE is set, *TXSM* must be set to 1.
2. X - don't care

7.2.2.3 Advanced Transmit Data Descriptor

Table 7-34. Advanced Transmit Data Descriptor (TDESD) Fetch Layout - (Type = 0011b)

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

1. RSV - Reserved

Table 7-35. Advanced Tx descriptor write-back format

0	RSV[1]		
8	Reserved	STA	Reserved
	63 36	**35 32**	**31 0**

1. RSV - Reserved

Note: For frames that span multiple descriptors, all fields **apart** from DCMD.EOP, DCMD.RS, DCMD.DEXT, DTALEN, Address and DTYP are valid only in the first descriptor and are ignored in the subsequent ones.

7.2.2.3.1 Address (64)

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

7.2.2.3.2 DTALEN (16)

Length in bytes of data buffer at the address pointed to by this specific descriptor.

Note: If the *TCTL.PSP* bit is set, the total length of the packet transmitted, not including FCS, should be at least 17 bytes.

The maximum allowable packet size for transmits is based on the value written to the *DMA TX Max Allowable packet size* (*DTXMXPKTSZ*) register. Default value is 9728 bytes.

7.2.2.3.3 MAC (2)

Table 7-36. Transmit Data (TDESD.MAC) Layout

1	0
1588	Reserved

- 1588 (bit 1) - IEEE1588 Timestamp packet.

7.2.2.3.4 DTYP (4)

0011b is the value for this descriptor type.

7.2.2.3.5 DCMD (8)

Table 7-37. Transmit Data (TDESD.DCMD) Layout

7	6	5	4	3	2	1	0
TSE	VLE	DEXT	Reserved	RS	Reserved	IFCS	EOP

- TSE (bit 7) - TCP/UDP Segmentation Enable
- VLE (bit 6) - VLAN Packet Enable
- DEXT (bit 5) - Descriptor Extension (1b for advanced mode)
- Reserved (bit 4)
- RS (bit 3) - Report Status
- Reserved (bit 2)
- IFCS (bit 1) - Insert FCS
- EOP (bit 0) - End Of Packet

TSE indicates a TCP/UDP segmentation request. When *TSE* is set in the first descriptor of a TCP packet, hardware must use the corresponding context descriptor in order to perform TCP segmentation. The type of segmentation applied is defined according to the *TUCMD.L4T* field in the context descriptor.

Note: It is recommended that *TCTL.PSP* be enabled when *TSE* is used since the last frame can be shorter than 60 bytes - resulting in a bad frame if TCTL.*PSP* is disabled.

VLE indicates that the packet is a VLAN packet and hardware must add the VLAN Ethertype and an 802.1q VLAN tag to the packet.

DEXT must be 1b to indicate advanced descriptor format (as opposed to legacy).

RS signals hardware to report the status information. This is used by software that does in-memory checks of the transmit descriptors to determine which ones are done. For example, if software queues up 10 packets to transmit, it can set the *RS* bit in the last descriptor of the last packet. If software maintains a list of descriptors with the *RS* bit set, it can look at them to determine if all packets up to (and including) the one with the *RS* bit set have been buffered in the output FIFO. Looking at the status byte and checking the *DD* bit do this. If *DD* is set, the descriptor has been processed. Refer to the next section for the layout of the status field.

Note: Descriptors with zero length transfer no data.

IFCS, when set, hardware appends the MAC FCS at the end of the packet. When cleared, software should calculate the FCS for proper CRC check. There are several cases in which the hardware changes the packet, and thus the software must set *IFCS*:

- Transmitting a short packet while padding is enabled by the *TCTL.PSP* bit.
- Checksum offload is enabled by the either the *TXSM* or *IXSM* bits in the *TDESD.POPTS* field.
- VLAN header insertion enabled by the *VLE* bit in the *TDESD.DCMD*.
- TCP/UDP segmentation offload enabled by *TSE* bit in the *TDESD.DCMD*.

EOP indicates whether this is the last buffer for an incoming packet.

7.2.2.3.6 STA (4)

- Rsv (bits 1-3) - Reserved
- DD (bit 0) - Descriptor Done

7.2.2.3.7 IDX (3)

Index into the hardware context table to indicate which context should be used for this request. If no offload is required, this field is not relevant and no context needs to be initiated before the packet is sent. See Table 7-33 for details on type of transmit packet offloads that require a context reference.

7.2.2.3.8 POPTS (6)

Table 7-38. Transmit Data (TDESD.POPTS) Layout

5 3	2	1	0
Reserved	Reserved	TXSM	IXSM

- Reserved (bits 5:3)
- Reserved (bit 2)
- TXSM (bit 1) - Insert L4 Checksum
- IXSM (bit 0) - Insert IP Checksum

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

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

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

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

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

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

7.2.2.3.9 PAYLEN (18)

PAYLEN indicates the size (in byte units) of the data buffer(s) in host memory for transmission. In a single send packet, *PAYLEN* defines the entire packet size fetched from host memory. It does not include the fields that hardware adds such as: optional VLAN tagging, Ethernet CRC or Ethernet padding. When TCP or UDP segmentation offload is enabled (*DCMD.TSE* is set), *PAYLEN* defines the TCP/UDP payload size fetched from host memory.

Note: When a packet spreads over multiple descriptors, all the descriptor fields are only valid in the first descriptor of the packet, except for *RS*, which is always checked, and *EOP*, which is always set at last descriptor of the series.

7.2.2.4 Transmit Descriptor Ring Structure

The transmit descriptor ring structure is shown in Figure 7-12. A pair of hardware registers maintains each transmit descriptor ring in the host memory. New descriptors are added to the queue by software by writing descriptors into the circular buffer memory region and moving the tail pointer associated with that queue. The tail pointer points to one entry beyond the last hardware owned descriptor. Transmission continues up to the descriptor where head equals tail at which point the queue is empty.

Descriptors passed to hardware should not be manipulated by software until the head pointer has advanced past them.

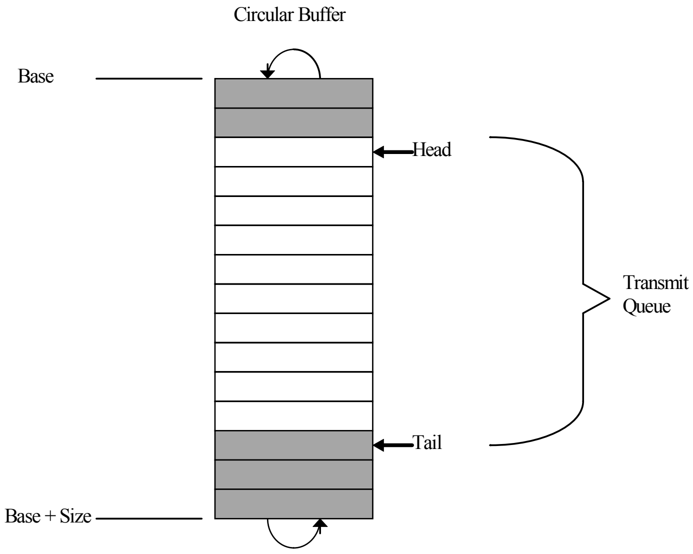


Figure 7-12. Transmit Descriptor Ring Structure

The shaded boxes in the figure represent descriptors that are not currently owned by hardware that software can modify.

The transmit descriptor ring is described by the following registers:

- Transmit Descriptor Base Address register (*TDBA 0-7*):

  This register indicates the start address of the descriptor ring buffer in the host memory; this 64-bit address is aligned on a 16-byte boundary and is stored in two consecutive 32-bit registers. Hardware ignores the lower four bits.

- Transmit Descriptor Length register (*TDLEN 0-7*):

  This register determines the number of bytes allocated to the circular buffer. This value must be zero modulo 128.

- Transmit Descriptor Head register (*TDH 0-7*):

  This register holds a value that is an offset from the base and indicates the in-progress descriptor. There can be up to 64 KB descriptors in the circular buffer. Reading this register returns the value of head corresponding to descriptors already loaded in the output FIFO. This register reflects the internal head of the hardware write-back process including the descriptor in the posted write pipe and might point further ahead than the last descriptor actually written back to the memory.

- Transmit Descriptor Tail register (*TDT 0-7*):

  This register holds a value, which is an offset from the base, and indicates the location beyond the last descriptor hardware can process. This is the location where software writes the first new descriptor.

  The driver should not handle to the 82580 descriptors that describe a partial packet. Consequently, the number of descriptors used to describe a packet can not be larger than the ring size.

- Tx Descriptor Completion Write–Back Address High/Low Registers (*TDWBAH/TDWBAL* 0-7):

  These registers hold a value that can be used to enable operation of head writeback operation. When *TDWBAL.Head_WB_En* is set and the RS bit is set in the Tx descriptor, following corresponding data upload into packet buffer, the 82580 writes the Transmit Descriptor Head value for this queue to the 64 bit address specified by the *TDWBAH* and *TDWBAL* registers. The Descriptor Head value is an offset from the base, and indicates the descriptor location hardware processed and software can utilize for new Transmit packets. See Section 7.2.3 for additional information.

The base register indicates the start of the circular descriptor queue and the length register indicates the maximum size of the descriptor ring. The lower seven bits of length are hard wired to 0b. Byte addresses within the descriptor buffer are computed as follows: address = base + (ptr * 16), where ptr is the value in the hardware head or tail register.

The size chosen for the head and tail registers permit a maximum of 65536 (64 KB) descriptors, or approximately 16 KB packets for the transmit queue given an average of four descriptors per packet.

Once activated, hardware fetches the descriptor indicated by the hardware head register. The hardware tail register points one descriptor beyond the last valid descriptor. Software can read and detect which packets have already been processed by hardware as follows:

- Read the head register to determine which packets (those logically before the head) have been transferred to the on-chip FIFO or transmitted. Note that this method is not recommended as races between the internal update of the head register and the actual write-back of descriptors might occur.
- Read the value of the head as stored at the address pointed by the *TDBAH/TDBAL* pair.
- Track the *DD* bits in the descriptor ring.

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

Note: Software can determine if a packet has been sent by either of three methods: setting the *RS* bit in the transmit descriptor command field or by performing a PIO read of the transmit head register, or by reading the head value written by the 82580 to the address pointed by the *TDWBAL* and *TDWBAH* registers (see Section 7.2.3 for details).

Checking the transmit descriptor *DD* bit or head value in memory eliminates a potential race condition. All descriptor data is written to the I/O bus prior to incrementing the head register, but a read of the head register could pass the data write in systems performing I/O write buffering. Updates to transmit descriptors use the same I/O write path and follow all data writes. Consequently, they are not subject to the race.

In general, hardware prefetches packet data prior to transmission. Hardware typically updates the value of the head pointer after storing data in the transmit FIFO.

7.2.2.5 Transmit Descriptor Fetching

When the *TXDCTL[n].ENABLE* bit is set and the on-chip descriptor cache is empty, a fetch happens as soon as any descriptors are made available (Host increments the *TDT[n]* tail pointer). The descriptor processing strategy for transmit descriptors is essentially the same as for receive descriptors except that a different set of thresholds are used. In addition the 82580 enables assigning 2 priority levels to each transmit descriptor queue using the *TXDCTL[n].Priority* bit. A transmit descriptor belonging to a queue with the *TXDCTL[n].Priority* set to 1 will always be fetched prior to a transmit descriptor belonging to a queue with the *TXDCTL[n].Priority* bit reset to 0. The number of on-chip transmit descriptors per queue is 24.

When there is an on-chip descriptor buffer empty, a descriptor fetch happens as soon as any descriptors are made available (host writes to the tail pointer). If several on-chip descriptor queues are in this situation at the same time, the queue with the highest priority as defined by the *TXDCTL[n].Priority* bit, is fetched. If several on-chip transmit descriptor queues with the same priority need to fetch descriptors, descriptors from queues that are more starved are fetched. If a number of queues have a similar priority and starvation level, highest indexed queue is served first and so forth, down to the lowest indexed queue.

Note: The starvation level of a queue corresponds to the number of descriptors above the prefetch threshold (*TXDCTL[n].PTHRESH)* that are already in the internal queue. The queue is more starved if there are less descriptors in the internal transmit descriptor cache. Comparing starvation level might be done roughly, not at the single descriptor level of resolution.

A queue is considered empty for the transmit descriptor fetch algorithm as long as:

- There is still no complete packet (single or large send) in its corresponding internal queue.
- There is no descriptor already in its way from system memory to the internal cache.
- The internal corresponding internal descriptor cache is not full.

Each time a descriptor fetch request is sent for an empty queue, the maximum available number of descriptor is requested, regardless of cache alignment issues.

When the on-chip buffer is nearly empty (below *TXDCTL[n].PTHRESH*), a prefetch is performed each time enough valid descriptors (*TXDCTL[n].HTHRESH*) are available in host memory and no other DMA activity of greater priority is pending (descriptor fetches and write-backs or packet data transfers).

When the number of descriptors in host memory is greater than the available on-chip descriptor storage, the 82580 might elect to perform a fetch that is not a multiple of cache-line size. Hardware performs this non-aligned fetch if doing so results in the next descriptor fetch being aligned on a cache-line boundary. This enables the descriptor fetch mechanism to be more efficient in the cases where it has fallen behind software.

Note: The 82580 NEVER fetches descriptors beyond the descriptor tail pointer.

7.2.2.6 Transmit Descriptor Write-Back

The descriptor write-back policy for transmit descriptors is similar to that of the receive descriptors when the *TXDCTL[n].WTHRESH* value is not 0x0. In this case, all descriptors are written back regardless of the value of their *RS* bit.

When the *TXDCTL[n].WTHRESH* value is 0x0, since transmit descriptor write-backs do not happen for every descriptor, only transmit descriptors that have the *RS* bit set are written back.

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

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

Descriptors are written back in one of three cases:

- *TXDCTL[n].WTHRESH* = 0x0 and a descriptor which has *RS* set is ready to be written back.
- The corresponding *EITR* counter has reached zero.
- *TXDCTL[n].WTHRESH* > 0x0 and *TXDCTL[n].WTHRESH* descriptors have accumulated.

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

The other two conditions are only valid if descriptor bursting is enabled (Section 8.12.15). In the second condition, the EITR counter is used to force timely write-back of descriptors. The first packet after timer initialization starts the timer. Timer expiration flushes any accumulated descriptors and sets an interrupt event (TXDW).

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

An additional mode in which transmit descriptors are not written back at all and the head pointer of the descriptor ring is written instead as described in Section 7.2.3.

Note: When transmit ring is smaller than internal cache size (24 descriptors) then at least one full packet should be placed in the ring and *TXDCTL[n].WTHRESH* value should be less than ring size. If *TXDCTL[n].WTHRESH* is 0x0 (transmit RS mode) then at least one descriptor should have the *RS* bit set inside the ring.

7.2.3 Transmit Completions Head Write Back

In legacy hardware, transmit requests are completed by writing the *DD* bit to the transmit descriptor ring. This causes cache thrash since both the software device driver and hardware are writing to the descriptor ring in host memory. Instead of writing the *DD* bits to signal that a transmit request completed, hardware can write the contents of the descriptor queue head to host memory. The software device driver reads that memory location to determine which transmit requests are complete.

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

7.2.3.1 Description

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

Head write back occurs if *TDWBAL[n].Head_WB_En* is set for this queue and the *RS* bit is set in the Tx descriptor, following corresponding data upload into packet buffer. If the head write-back feature is enabled, the 82580 ignores the *TXDCTL[n].WTHRESH* value and takes in account only descriptors with the *RS* bit set (as if the *TXDCTL[n].WTHRESH* was set to 0). In addition, the head write-back occurs upon EITR expiration for queues where the *WB_on_EITR* bit in *TDWBAL* is set.

Software can also enable coalescing of the head write-back operations to reduce traffic on the PCIe bus, by programming the *TXDCTL.HWBTHRESH* field to a value greater than 0. In this case head write-back operation will occur only after the internal pending write-back count is greater than the *TXDCTL.HWBTHRESH* value.

The software device driver has control on this feature through Tx queue 0-7 head write-back address, low (*TDWBAL[n]*) and high *(TDWBAH[n]*) registers thus supporting 64-bit address access. See registers description in Section 8.12.16 and Section 8.12.17.

The 2 low register's LSB bits of the *TDWBAL[n]* register hold the control bits.

1. The *Head_WB_En* bit enables activation of the head write back feature. When *TDWBAL[n].Head_WB_En is set to 1* no TX descriptor write-back is executed for this queue.
2. *The WB_on_EITR* bit enables head write upon EITR expiration. When Head write back operation is enabled (*TDWBAL[n].Head_WB_En* = 1) setting the *TDWBAL[n].WB_on_EITR* bit to 1 enables placing an upper limit on delay of head write-back operation.

The 30 upper bits of the *TDWBAL[n]* register hold the lowest 32 bits of the head write-back address, assuming that the two last bits are zero. The *TDWBAH[n]* register holds the high part of the 64-bit address.

Note: Hardware writes a full Dword when writing this value, so software should reserve enough space for each head value.

If software enables Head Write-Back, it must also disable PCI Express Relaxed Ordering on the write-back transactions. This is done by disabling bit 11 in the *TXCTL* register for each active transmit queue. See Section 8.13.2.

The 82580 might update the Head with values that are larger then the last Head pointer which holds a descriptor with RS bit set, but still the value will always point to a free descriptor (descriptor that is not owned by the 82580 anymore).

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

7.2.4 TCP/UDP Segmentation

Hardware TCP segmentation is one of the offloading options supported by the Windows* and Linux* TCP/IP stack. This is often referred to as TCP Segmentation Offloading or TSO. This feature enables the TCP/IP stack to pass to the network device driver a message to be transmitted that is bigger than the Maximum Transmission Unit (MTU) of medium. It is then the responsibility of the software device driver and hardware to divide the TCP message into MTU size frames that have appropriate layer 2 (Ethernet), 3 (IP), and 4 (TCP) headers. These headers must include sequence number, checksum fields, options and flag values as required. Note that some of these values (such as the checksum values) are unique for each packet of the TCP message and other fields such as the source IP address are constant for all packets associated with the TCP message.

The 82580 supports also UDP segmentation for embedded applications, although this offload is not supported by the regular Windows* and Linux* stacks. Any reference in this section to TCP segmentation, should be considered as referring to both TCP and UDP segmentation.

Padding (TCTL.PSP) must be enabled in TCP segmentation mode, since the last frame might be shorter than 60 bytes, resulting in a bad frame if *PSP* is disabled.

The offloading of these mechanisms from the software device driver to the 82580 saves significant CPU cycles. Note that the software device driver shares the additional tasks to support these options.

7.2.4.1 Assumptions

The following assumptions apply to the TCP segmentation implementation in the 82580:

- The *RS* bit operation is not changed.
- Interrupts are set after data in buffers pointed to by individual descriptors is transferred (DMA'd) to hardware.

7.2.4.2 Transmission Process

The transmission process for regular (non-TCP segmentation packets) involves:

- The protocol stack receives from an application a block of data that is to be transmitted.
- The protocol stack calculates the number of packets required to transmit this block based on the MTU size of the media and required packet headers.

For each packet of the data block:

- Ethernet, IP and TCP/UDP headers are prepared by the stack.
- The stack interfaces with the software device driver and commands it to send the individual packet.
- The software device driver gets the frame and interfaces with the hardware.
- The hardware reads the packet from host memory (via DMA transfers).
- The software device driver returns ownership of the packet to the Network Operating System (NOS) when hardware has completed the DMA transfer of the frame (indicated by an interrupt).

The transmission process for the 82580 TCP segmentation offload implementation involves:

- The protocol stack receives from an application a block of data that is to be transmitted.
- The stack interfaces to the software device driver and passes the block down with the appropriate header information.
- The software device driver sets up the interface to the hardware (via descriptors) for the TCP segmentation context.

Hardware DMA's (transfers) the packet data and performs the Ethernet packet segmentation and transmission based on offset and payload length parameters in the TCP/IP context descriptor including:

- Packet encapsulation
- Header generation and field updates including IPv4, IPV6, and TCP/UDP checksum generation
- The software device driver returns ownership of the block of data to the NOS when hardware has completed the DMA transfer of the entire data block (indicated by an interrupt).

7.2.4.2.1 TCP Segmentation Data Fetch Control

To perform TCP Segmentation in the 82580, the DMA must be able to fit at least one packet of the segmented payload into available space in the on-chip Packet Buffer. The DMA does various comparisons between the remaining payload and the Packet Buffer available space, fetching additional payload and sending additional packets as space permits.

To support interleaving between descriptor queues at Ethernet frame resolution inside TSO requests, the frame header pointed to by the so called header descriptors are reread from system memory by hardware for every LSO segment. The 82580 stores in an internal cache only the header's descriptors instead of the header's content.

To limit the internal cache size software should not spread the L3/L4 header (TCP, UDP, IPV4 or IPV6) on more than 4 descriptors. In the last header buffer it's allowed to mix header and data. This limitation stands for up to Layer4 header included, and for IPv4 or IPv6 indifferently.

7.2.4.2.2 TCP Segmentation Write-Back Modes

Since the TCP segmentation mode uses the buffers that contains the L3/L4 header multiple times, there are some limitations on the usage of different combinations of writeback and buffer release methods in order to guarantee the header buffer's availability until the entire packet is processed. These limitations are described in Table 7-39 below.

Table 7-39. Write Back Options For Large Send

WTHRESH	RS	HEAD Write Back Enable	Hardware Behavior	Software Expected Behavior for TSO packets.
0	Set in EOP descriptors only	Disable	Hardware writes back descriptors with RS bit set one at a time.	Software can retake ownership of all descriptors up to last descriptor with DD bit set.
0	Set in any descriptors	Disable	Hardware writes back descriptors with RS bit set one at a time.	Software can retake ownership of entire packets (EOP bit set) up to last descriptor with DD bit set.
0	Not set at all	Disable	Hardware does not write back any descriptor (since *RS* bit is not set)	Software should poll the TDH register. The TDH register reflects the last descriptor that software can take ownership of.[1]
0	Not set at all	Enable	Hardware writes back the head pointer only at EITR expire event reflecting the last descriptor that software can take ownership of.	Software may poll the TDH register or use the head value written back at EITR expire event. The TDH register reflects the last descriptor that software can take ownership of.
>0	Don't care	Disable	Hardware writes back all the descriptors in bursts and set all the *DD* bits.	Software can retake ownership of entire packets up to last descriptor with both DD and EOP bits set. Note: The TDH register reflects the last descriptor that software can take ownership of[1].
Don't care	Set in EOP descriptors only	Enable	Hardware writes back the Head pointer per each descriptor with *RS* bit set.[2]	Software can retake ownership of all descriptors up to the descriptor pointed by the head pointer read from system memory (by interrupt or polling).
Don't care	Set in any descriptors	Enable	Hardware writes back the Head pointer per each descriptor with *RS* bit set.	This mode is illegal since software won't access the descriptor, it cannot tell when the pointer passed the EOP descriptor.

[1] Note that polling of the TDH register is a valid method only when the RS bit is never set, otherwise race conditions between software and hardware accesses to the descriptor ring can occur.

2. At EITR expire event, the Hardware writes back the head pointer reflecting the last descriptor that software can take ownership of.

7.2.4.3 TCP Segmentation Performance

Performance improvements for a hardware implementation of TCP Segmentation off-load include:

- The stack does not need to partition the block to fit the MTU size, saving CPU cycles.
- The stack only computes one Ethernet, IP, and TCP header per segment, saving CPU cycles.
- The Stack interfaces with the device driver only once per block transfer, instead of once per frame.
- Larger PCIe bursts are used which improves bus efficiency (such as lowering transaction overhead).
- Interrupts are easily reduced to one per TCP message instead of one per packet.
- Fewer I/O accesses are required to command the hardware.

7.2.4.4 Packet Format

Typical TCP/IP transmit window size is 8760 bytes (about 6 full size frames). Today the average size on corporate Intranets is 12-14KB, and normally the maximum window size allowed is 64KB (unless Windows Scaling - RFC 1323 is used). A TCP message can be as large as 256 KB and is generally fragmented across multiple pages in host memory. The 82580 partitions the data packet into standard Ethernet frames prior to transmission according to the requested MSS. The 82580 supports calculating the Ethernet, IP, TCP, and UDP headers, including checksum, on a frame-by-frame basis.

Table 7-40. TCP/IP Packet Format Sent by Host

Pseudo Header			Data
Ethernet	IPv4/IPv6	TCP/UDP	DATA (full TCP message)

Table 7-41. TCP/IP Packet Format Sent by the 82580

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

Frame formats supported by the 82580 include:

- Ethernet 802.3
- IEEE 802.1Q VLAN (Ethernet 802.3ac)
- Ethernet Type 2
- Ethernet SNAP
- IPv4 headers with options
- IPv6 headers with extensions
- TCP with options
- UDP with options.

VLAN tag insertion might be handled by hardware

Note: UDP (unlike TCP) is not a “reliable protocol”, and fragmentation is not supported at the UDP level. UDP messages that are larger than the MTU size of the given network medium are normally fragmented at the IP layer. This is different from TCP, where large TCP messages can be fragmented at either the IP or TCP layers depending on the software implementation. The 82580 has the ability to segment UDP traffic (in addition to TCP traffic), however, because UDP packets are generally fragmented at the IP layer, the 82580's “TCP Segmentation” feature is not normally useful to handle UDP traffic.

7.2.4.5 TCP/UDP Segmentation Indication

Software indicates a TCP/UDP Segmentation transmission context to the hardware by setting up a TCP/IP Context Transmit Descriptor (see Section 7.2.2). The purpose of this descriptor is to provide information to the hardware to be used during the TCP segmentation off-load process.

Setting the *TSE* bit in the *TDESD.DCMD* field to 1b indicates that this descriptor refers to the TCP Segmentation context (as opposed to the normal checksum off loading context). This causes the checksum off loading, packet length, header length, and maximum segment size parameters to be loaded from the Context descriptor into the device.

The TCP Segmentation prototype header is taken from the packet data itself. Software must identity the type of packet that is being sent (IPv4/IPv6, TCP/UDP, other), calculate appropriate checksum off loading values for the desired checksum, and calculate the length of the header which is pre-appended. The header might be up to 240 bytes in length.

Once the TCP Segmentation context has been set, the next descriptor provides the initial data to transfer. This first descriptor(s) must point to a packet of the type indicated. Furthermore, the data it points to might need to be modified by software as it serves as the prototype header for all packets within the TCP Segmentation context. The following sections describe the supported packet types and the various updates which are performed by hardware. This should be used as a guide to determine what must be modified in the original packet header to make it a suitable prototype header.

The following summarizes the fields considered by the driver for modification in constructing the prototype header.

IP Header

For IPv4 headers:

- *Identification* Field should be set as appropriate for first packet 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 sum as follows (there is a more detailed discussion of this is Section 7.2.4.6):

Table 7-42. TCP Partial Pseudo-Header Sum for IPv4

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

Table 7-43. TCP Partial Pseudo-Header Sum for IPv6

<table>
<tr><td colspan="2">IPv6 Source Address</td></tr>
<tr><td colspan="2">IPv6 Final Destination Address</td></tr>
<tr><td colspan="2">Zero</td></tr>
<tr><td>Zero</td><td>Next Header</td></tr>
</table>

UDP Header

- Checksum should be set as in TCP header, above

The following sections describe the updating process performed by the hardware for each frame sent using the TCP Segmentation capability.

7.2.4.6 Transmit Checksum Offloading with TCP/UDP Segmentation

The 82580 supports checksum off-loading as a component of the TCP Segmentation off-load feature and as a standalone capability. Section 7.2.5 describes the interface for controlling the checksum off-loading feature. This section describes the feature as it relates to TCP Segmentation.

The 82580 supports IP and TCP header options in the checksum computation for packets that are derived from the TCP Segmentation feature.

Note: The 82580 is capable of computing one level of IP header checksum and one TCP/UDP header and payload checksum. In case of multiple IP headers, the driver needs to compute all but one IP header checksum. The 82580 calculates check sums on the fly on a frame-by-frame basis and inserts the result in the IP/TCP/UDP headers of each frame. TCP and UDP checksum are a result of performing the checksum on all bytes of the payload and the pseudo header.

Three specific types of checksum are supported by the hardware in the context of the TCP Segmentation off-load feature:

- IPv4 checksum
- TCP checksum

Each packet that is sent via the TCP segmentation off-load feature optionally includes the IPv4 checksum and either the TCP checksum.

All checksum calculations use a 16-bit wide one's complement checksum. The checksum word is calculated on the outgoing data.

Table 7-44. Supported Transmit Checksum Capabilities

Packet Type	Hardware IP Checksum Calculation	Hardware TCP/UDP Checksum Calculation
IP v4 packets	Yes	Yes
IP v6 packets (no IP checksum in Ipv6)	NA	Yes
Packet is greater than 1552 bytes; (LPE=1b)	Yes	Yes
Packet has 802.3ac tag	Yes	Yes

Table 7-44. Supported Transmit Checksum Capabilities

Packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet has TCP or UDP options	Yes	Yes
IP header's protocol field contains a protocol # other than TCP or UDP.	Yes	No

The table below summarizes the conditions of when checksum off loading can/should be calculated.

Table 7-45. Conditions for Checksum Off Loading

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 off-load
	No	No	Non-IP packet or checksum not offloaded
TSO	Yes	Yes	For TSO, checksum off-load must be done

7.2.4.7 IP/TCP/UDP Header Update

IP/TCP or IP/UDP header is updated for each outgoing frame based on the IP/TCP header prototype which hardware DMA's from the first descriptor(s). The checksum fields and other header information are later updated on a frame-by-frame basis. The updating process is performed concurrently with the packet data fetch.

The following sections define what fields are modified by hardware during the TCP Segmentation process by the 82580.

Note: Software must make PAYLEN and HDRLEN value of Context descriptors correct. Otherwise, the failure of Large Send due to either under-run or over-run might cause hardware to send bad packets or even cause TX hardware to hang. The indication of Large Send failure can be checked in the TSCTFC statistic register.

7.2.4.7.1 TCP/IP/UDP Header for the First Frames

The hardware makes the following changes to the headers of the first packet that is derived from each TCP segmentation context.

MAC Header (for SNAP)

- Type/Len field = MSS + MACLEN + IPLEN + L4LEN - 14

Ipv4 Header

- IP Total Length = MSS + L4LEN + IPLEN
- IP Checksum

Ipv6 Header

- Payload Length = MSS + L4LEN + IPV6_HDR_extension[1]

TCP Header

- Sequence Number: The value is the Sequence Number of the first TCP byte in this frame.

[1] IPV6_HDR_extension is calculated as IPLEN - 40 bytes.

- The flag values of the first frame are set by ANDing the flag word in the pseudo header with the DTXTCPFLGL.TCP_flg_first_seg register field. The default value of the DTXTCPFLGL.TCP_flg_first_seg are set so that the FIN flag and the PSH flag are cleared in the first frame.
- TCP Checksum

7.2.4.7.2 TCP/IP Header for the Subsequent Frames

The hardware makes the following changes to the headers for subsequent packets that are derived as part of a TCP segmentation context:

Number of bytes left for transmission = PAYLEN - (N * MSS). Where N is the number of frames that have been transmitted.

MAC Header (for SNAP Packets)

Type/Len field = MSS + MACLEN + IPLEN + L4LEN - 14

Ipv4 Header

- IP Identification: incremented from last value (wrap around)
- IP Total Length = MSS + L4LEN + IPLEN
- IP Checksum

Ipv6 Header

- Payload Length = MSS + L4LEN + IPV6_HDR_extension[1]

TCP Header

- Sequence Number update: Add previous TCP payload size to the previous sequence number value. This is equivalent to adding the MSS to the previous sequence number.
- The flag values of the subsequent frames are set by ANDing the flag word in the pseudo header with the DTXTCPFLGL.TCP_Flg_mid_seg register field. The default value of the DTXTCPFLGL.TCP_Flg_mid_seg are set so that if the FIN flag and the PSH flag are cleared in these frames.
- TCP Checksum

7.2.4.7.3 TCP/IP 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 based on 16 bit-width)
- IP Checksum

[1] IPV6_HDR_extension is calculated as IPLEN - 40 bytes.

Ipv6 Header

- Payload Length = last frame payload bytes + L4LEN + IPV6_HDR_extension[1]

TCP Header

- Sequence Number update: Add previous TCP payload size to the previous sequence number value. This is equivalent to adding the MSS to the previous sequence number.
- The flag values of the last frames are set by ANDing the flag word in the pseudo header with the DTXTCPFLGH.TCP_Flg_lst_seg register field. The default value of the DTXTCPFLGH.TCP_Flg_lst_seg are set so that if the FIN flag and the PSH flag are set in the last frame.
- TCP Checksum

7.2.4.8 IP/TCP/UDP Checksum Offloading

The 82580 performs checksum off loading as part of the TCP segmentation off-load feature.

These specific checksum are supported under TCP segmentation:

- IPv4 checksum
- TCP checksum

See Section 7.2.5 for description of checksum off loading of a single-send packet.

7.2.4.9 Data Flow

The flow used by the 82580 to do TCP segmentation is as follows:

1. Get a descriptor with a request for a TSO off-load of a TCP packet.
2. First Segment processing:
   a. Fetch all the buffers containing the header as calculated by the *MACLEN*, *IPLEN* and *L4LEN* fields. Save the addresses and lengths of the buffers containing the header (up to 4 buffers). The header content is not saved.
   b. Fetch data up to the MSS from subsequent buffers & calculate the adequate checksum(s).
   c. Update the Header accordingly and update internal state of the packet (next data to fetch and TCP SN).
   d. Send the packet to the network.
   e. If total packet was sent, go to step 4. else continue.
3. Next segments
   a. Wait for next arbitration of this queue.
   b. Fetch all the buffers containing the header from the saved addresses. Subsequent reads of the header might be done with a no snoop attribute.
   c. Fetch data up to the MSS or end of packet from subsequent buffers & calculate the adequate checksum(s).
   d. Update the Header accordingly and update internal state of the packet (next data to fetch and TCP SN).
   e. If total packet was sent, request is done, else restart from step 3.
4. Release all buffers (update head pointer).

Note: Descriptors are fetched in a parallel process according to the consumption of the buffers.

7.2.5 Checksum Offloading in Non-Segmentation Mode

The previous section on TCP Segmentation off-load describes the IP/TCP/UDP checksum off loading mechanism used in conjunction with TCP Segmentation. The same underlying mechanism can also be applied as a standalone feature. The main difference in normal packet mode (non-TCP Segmentation) is that only the checksum fields in the IP/TCP/UDP headers need to be updated.

Before taking advantage of the 82580's enhanced checksum off-load capability, a checksum context must be initialized. For the normal transmit checksum off-load feature this is performed by providing the device with a Descriptor with *TSE*=0b in the *TDESD.DCMD* field and setting either the *TXSM* or *IXSM* bits in the *TDESD.POPTS* field. Setting TSE=0b indicates that the normal checksum context is being set, as opposed to the segmentation context. For additional details on contexts, refer to Section 7.2.2.4.

Note: Enabling the checksum off loading capability without first initializing the appropriate checksum context leads to unpredictable results. CRC appending (TDESC.CMD.IFCS) must be enabled in TCP/IP checksum mode, since CRC must be inserted by hardware after the checksum has been calculated.

As mentioned in Section 7.2.2, it is not necessary to set a new context for each new packet. In many cases, the same checksum context can be used for a majority of the packet stream. In this case, some performance can be gained by only changing the context on an as needed basis or electing to use the off-load feature only for a particular traffic type, thereby avoiding the need to read all context descriptors except for the initial one.

Each checksum operates independently. Insertion of the IP and TCP checksum for each packet are enabled through the Transmit Data Descriptor POPTS.TSXM and POPTS.IXSM fields, respectively.

7.2.5.1 IP Checksum

Three fields in the Transmit Context Descriptor set the context of the IP checksum off loading feature:

- TUCMD.IPv4
- IPLEN
- MACLEN

TUCMD.IPv4=1b specifies that the packet type for this context is IPv4, and that the IP header checksum should be inserted. TUCMD.IPv4=0b indicates that the packet type is IPv6 (or some other protocol) and that the IP header checksum should not be inserted.

MACLEN specifies the byte offset from the start of the DMA'd data to the first byte to be included in the checksum, the start of the IP header. The minimal allowed value for this field is 12. Note that the maximum value for this field is 127. This is adequate for typical applications.

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

IPLEN specifies the IP header length. Maximum allowed value for this field is 511 Bytes.

MACLEN+IPLEN specify where the IP checksum should stop. This is limited to the first 127+511 bytes of the packet and must be less than or equal to the total length of a given packet. If this is not the case, the result is unpredictable.

The 16-bit IPv4 Header Checksum is placed at the two bytes starting at MACLEN+10.

As mentioned in Section 7.2.2.2, Transmit Contexts, it is not necessary to set a new context for each new packet. In many cases, the same checksum context can be used for a majority of the packet stream. In this case, some performance can be gained by only changing the context on an as needed basis or electing to use the off-load feature only for a particular traffic type, thereby avoiding all context descriptor reads except for the initial one.

7.2.5.2 TCP/UDP Checksum

Three fields in the Transmit Context Descriptor set the context of the TCP/UDP checksum off loading feature:

- MACLEN
- IPLEN
- TUCMD.L4T

TUCMD.L4T=01b specifies that the packet type is TCP, and that the 16-bit TCP header checksum should be inserted at byte offset MACLEN+IPLEN+16. TUCMD.L4T=00b indicates that the packet is UDP and that the 16-bit checksum should be inserted starting at byte offset MACLEN+IPLEN+6.

IPLEN+MACLEN specifies the byte offset from the start of the DMA'd data to the first byte to be included in the checksum, the start of the TCP header. The minimal allowed value for this sum is 32/42 for UDP or TCP respectively.

Note: The IPLEN+MACLEN+L4LEN value needs to be less than the total DMA length for a packet. If this is not the case, the results are unpredictable.

The TCP/UDP checksum always continues to the last byte of the DMA data.

Note: For non-TSO, software still needs to calculate a full checksum for the TCP/UDP pseudo-header. This checksum of the pseudo-header should be placed in the packet data buffer at the appropriate offset for the checksum calculation.

7.2.5.3 SCTP CRC Offloading

For SCTP packets, a CRC32 checksum offload is provided.

Three fields in the Transmit Context Descriptor set the context of the STCP checksum off loading feature:

- MACLEN
- IPLEN
- TUCMD.L4T

TUCMD.L4T=10b specifies that the packet type is SCTP, and that the 32-bit STCP CRC should be inserted at byte offset MACLEN+IPLEN+8.

IPLEN+MACLEN specifies the byte offset from the start of the DMA'd data to the first byte to be included in the checksum, the start of the STCP header. The minimal allowed value for this sum is 26.

The SCTP CRC calculation always continues to the last byte of the DMA data.

The SCTP total L3 payload size (PAYLEN - IPLEN - MACLEN) should be a multiple of 4 bytes (SCTP padding not supported).

Note: TSO is not available for SCTP packets.

7.2.5.4 Checksum Supported Per Packet Types

The following table summarizes which checksum is supported per packet type.

Note: TSO is not supported for packet types for which IP checksum & TCP checksum can not be calculated.

Table 7-46. Checksum Per Packet Type

Packet Type	Hardware IP Checksum Calculation	Hardware TCP/UDP/SCTP Checksum Calculation
Ipv4 packets	Yes	Yes
Ipv6 packets	No (n/a)	Yes
Ipv6 packet with next header options: • Hop-by-Hop options • Destinations options • Routing (w len 0b) • Routing (w len >0b) • Fragment • Home option	 No (n/a) No (n/a) No (n/a) No (n/a) No (n/a) No (n/a)	 Yes Yes Yes No No No
Ipv4 tunnels: • Ipv4 packet in an Ipv4 tunnel • Ipv6 packet in an Ipv4 tunnel	 Either IP or TCP/SCTP[1] Either IP or TCP/SCTP[1]	 Either IP or TCP/SCTP [1] Either IP or TCP/SCTP[1]
Ipv6 tunnels: • Ipv4 packet in an Ipv6 tunnel • Ipv6 packet in an Ipv6 tunnel	 No No	 Yes Yes
Packet is an Ipv4 fragment	Yes	No
Packet is greater than 1552 bytes; (LPE=1b)	Yes	Yes
Packet has 802.3ac tag	Yes	Yes
Packet has TCP or UDP options	Yes	Yes
IP header’s protocol field contains protocol # other than TCP or UDP.	Yes	No

[1] For the tunneled case, the driver might do only the TCP checksum or Ipv4 checksum. If TCP checksum is desired, the driver should define the IP header length as the combined length of both IP headers in the packet. If an IPv4 checksum is required, the IP header length should be set to the Ipv4 header length.

7.2.6 Multiple Transmit Queues

The number of transmit queues is 8, to match the expected number of CPU cores on server processors and to support virtualization mode.

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

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

The 82580 supports assigning either high or low priority to each transmit queue. High priority is assigned to by setting the *TXDCTL[n].priority* bit to 1. When high priority is assigned to a specific transmit queue, The 82580 will always prioritize DMA accesses such as transmit descriptor read, transmit descriptor writeback and transmit data fetch, before servicing lower priority transmit queues on a specific Physical Function.

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

7.3 Interrupts

7.3.1 Mapping of Interrupt Causes

The 82580 supports the following interrupt modes:

- PCI legacy interrupts or MSI - selected when GPIE.Multiple_MSIX is 0b
- MSI-X - selected when GPIE.Multiple_MSIX is 1b.

Note: If only one MSI-X vector is allocated by the operating system, then the driver might use the non MSI-X mapping method even in MSI-X mode.

Mapping of interrupts causes is different in each of the above modes and is described in the following sections of this chapter.

7.3.1.1 Legacy and MSI Interrupt Modes

In legacy and MSI modes, an interrupt cause is reflected by setting a bit in the *EICR* register. This section describes the mapping of interrupt causes, like a specific Rx queue event or a Link Status Change event, to bits in the *EICR* register.

Mapping of queue-related causes is accomplished through the *IVAR* register. Each possible queue interrupt cause (each Rx or Tx queue) is allocated an entry in the *IVAR*, and each entry in the *IVAR* identifies one bit in the *EICR* register among the bits allocated to queue interrupt causes. It is possible to map multiple interrupt causes into the same *EICR* bit.

In this mode, different queue related interrupt causes can be mapped to the first 8 bits of the *EICR* register.

Interrupt causes related to non-queue causes are mapped into the *ICR* legacy register; each cause is allocated a separate bit. The sum of all causes is reflected in the *Other Cause* bit in *EICR*. Figure 7-13 below describes the allocation process.

The following configuration and parameters are involved:

- The *IVAR[3:0]* entries map 8 Tx queues and 8 Rx queues into the EICR[7:0] bits.
- The *IVAR_MISC* that maps non-queue causes is not used.
- The *EICR[30]* bit is allocated to the TCP timer interrupt cause.
- The *EICR[31]* bit is allocated to the other interrupt causes summarized in the ICR register.
- A single interrupt vector is provided.

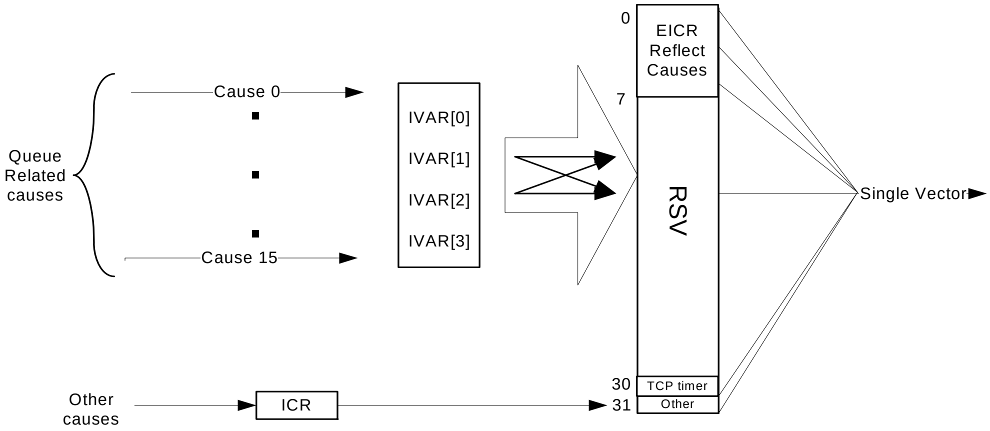


Figure 7-13. Cause Mapping in Legacy Mode

The Table below maps the different interrupt causes into the IVAR registers.

Table 7-47. Cause Allocation in the IVAR Registers - MSI and Legacy Mode

Interrupt	Entry	Description
Rx_i	i*2 (i= 0...7)	Receive queues i - Associates an interrupt occurring in the Rx queues i with a corresponding bit in the EICR register.
Tx_i	i*2+1 (i= 0...7)	Transmit queues i- Associates an interrupt occurring in the Tx queues I with a corresponding bit in the EICR register.

7.3.1.2 MSI-X Mode

In a MSI-X mode, the 82580 can request up to 10 Vectors.

In MSI-X mode, an interrupt cause is mapped into an MSI-X vector. This section describes the mapping of interrupt causes, like a specific Rx queue event or a Link Status Change event, to MSI-X vectors.

Mapping is accomplished through the *IVAR* register. Each possible cause for an interrupt is allocated an entry in the *IVAR*, and each entry in the *IVAR* identifies one MSI-X vector. It is possible to map multiple interrupt causes into the same MSI-X vector.

The *EICR* also reflects interrupt vectors. The *EICR* bits allocated for queue causes reflect the MSI-X vector (bit 2 is set when MSI-X vector 2 is used). Interrupt causes related to non-queue causes are mapped into the *ICR* (as in the legacy case). The MSI-X vector for all such causes is reflected in the *EICR*.

The following configuration and parameters are involved:

- The *IVAR[3:0]* registers map 8 Tx queues and 8 Rx queues events to up to 8 interrupt vectors
- The IVAR_MISC register maps a TCP timer and other events to 2 MSI-X vectors

Figure 7-14 describes the allocation process.

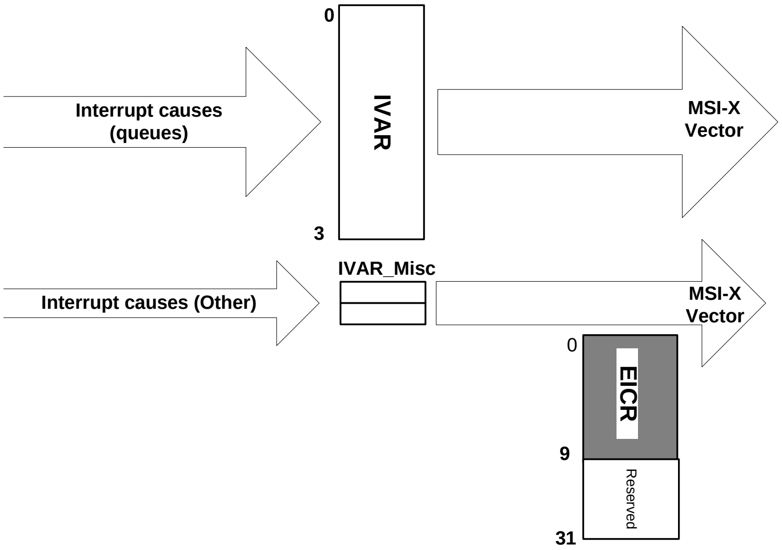


Figure 7-14. Cause Mapping in MSI-X Mode

Table 7-48 below defines which interrupt cause is represented by each entry in the MSI-X Allocation registers. The software has access to 18 mapping entries to map each cause to one of the 10 MSI-x vectors.

Table 7-48. Cause Allocation in the IVAR Registers

Interrupt	Entry	Description
Rx_i	i*2 (i= 0...7)	Receive queues i - Associates an interrupt occurring in the Rx queues i with a corresponding entry in the MSI-X Allocation registers.
Tx_i	i*2+1 (i= 0...7)	Transmit queues i- Associates an interrupt occurring in the Tx queues I with a corresponding entry in the MSI-X Allocation registers.
TCP timer	16	TCP Timer - Associates an interrupt issued by the TCP timer with a corresponding entry in the MSI-X Allocation registers
Other cause	17	Other causes - Associates an interrupt issued by the “other causes” with a corresponding entry in the MSI-X Allocation registers

7.3.2 Legacy Interrupt Registers

The interrupt logic consists of the registers listed in the tables below, plus the registers associated with MSI/MSI-X signaling. The first table describes the use of the registers in legacy mode and the second one the use of the register when using the extended interrupts functionality

Table 7-49. Interrupt Registers - Legacy Mode

Register	Acronym	Function
Interrupt Cause	ICR	Records interrupt conditions.
Interrupt Cause Set	ICS	Allows software to set bits in the *ICR*.

Table 7-49. Interrupt Registers - Legacy Mode

Interrupt Mask Set/Read	IMS	Sets or reads bits in the interrupt mask.
Interrupt Mask Clear	IMC	Clears bits in the interrupt mask.
Interrupt Acknowledge auto-mask	IAM	Under some conditions, the content of this register is copied to the mask register following read or write of *ICR*.

Table 7-50. Interrupt Registers - Extended Mode

Register	Acronym	Function
Extended Interrupt Cause	EICR	Records interrupt causes from receive and transmit queues. An interrupt is signaled when unmasked bits in this register are set.
Extended Interrupt Cause Set	EICS	Allows software to set bits in the Interrupt Cause register.
Extended Interrupt Mask Set/Read	EIMS	Sets or read bits in the interrupt mask.
Extended Interrupt Mask Clear	EIMC	Clears bits in the interrupt mask.
Extended Interrupt Auto Clear	EIAC	Allows bits in the *EICR* to be cleared automatically following an MSI-X interrupt without a read or write of the *EICR*.
Extended Interrupt Acknowledge auto-mask	EIAM	This register is used to decide which masks are cleared in the extended mask register following read or write of *EICR* or which masks are set following a write to *EICS*. In MSI-X mode, this register also controls which bits in *EIMC* are cleared automatically following an MSI-X interrupt.
Interrupt Cause	ICR	Records interrupt conditions for special conditions - a single interrupt from all the conditions of *ICR* is reflected in the “other” field of the *EICR*.
Interrupt Cause Set	ICS	Allows software to set bits in the *ICR*.
Interrupt Mask Set/Read	IMS	Sets or reads bits in the *Other* interrupt mask.
Interrupt Mask Clear	IMC	Clears bits in the *Other* interrupt mask.
Interrupt Acknowledge auto-mask	IAM	Under some conditions, the content of this register is copied to the mask register following read or write of *ICR*.
General Purpose Interrupt Enable	GPIE	Controls different behaviors of the interrupt mechanism.

7.3.2.1 Interrupt Cause Register (ICR)

7.3.2.1.1 Legacy Mode

In Legacy mode, *ICR* is used as the sole interrupt cause register. Upon reception of an interrupt, the interrupt handling routine can read this register in order to find out what are the causes of this interrupt.

7.3.2.1.2 Advanced Mode

In advanced mode, this register captures the interrupt causes not directly captured by the *EICR*. These are infrequent management interrupts and error conditions.

Note that when *EICR* is used in advanced mode, the RX /TX related bits in *ICR* should be masked.

ICR bits are cleared on register read. If *GPIE.NSICR* = 0b, then the clear on read occurs only if no bit is set in the *IMS* register or at least one bit is set in the *IMS* register and there is a true interrupt as reflected in the *ICR.INTA* bit.

7.3.2.2 Interrupt Cause Set Register (ICS)

This register allows software to set bits in the *ICR* register. Writing a 1b in an *ICS* bit causes the corresponding bit in the *ICS* register to be set. Used usually to re-arm interrupts the software device driver didn't have time to handle in the current interrupt routine.

7.3.2.3 Interrupt Mask Set/Read Register (IMS)

An interrupt is enabled if its corresponding mask bit in this register is set to 1b, and disabled if its corresponding mask bit is set to 0b. A PCIe interrupt is generated whenever one of the bits in this register is set, and the corresponding interrupt condition occurs. The occurrence of an interrupt condition is reflected by having a bit set in the Interrupt Cause Register.

Reading this register returns which bits have an interrupt mask set.

A particular interrupt might be enabled by writing a 1b to the corresponding mask bit in this register. Any bits written with a 0b are unchanged. Thus, if software desires to disable a particular interrupt condition that had been previously enabled, it must write to the *Interrupt Mask Clear* Register (see below), rather than writing a 0b to a bit in this register.

7.3.2.4 Interrupt Mask Clear Register (IMC)

Software blocks interrupts by clearing the corresponding mask bit. This is accomplished by writing a 1b to the corresponding bit in this register. Bits written with 0b are unchanged (their mask status does not change).

7.3.2.5 Interrupt Acknowledge Auto-mask register (IAM)

An *ICR* read or write has the side effect of writing the contents of this register to the *IMC* register. If *GPIE.NSICR* = 0b, then the copy of this register to the *IMC* register occurs only if at least one bit is set in the *IMS* register and there is a true interrupt as reflected in the *ICR.INTA* bit.

7.3.2.6 Extended Interrupt Cause Registers (EICR)

7.3.2.6.1 MSI/INT-A Mode (GPIE.Multiple_MSIX = 0)

This register records the interrupts causes, to provide Software with information on the interrupt source.

The interrupt causes include:

1. The Receive and Transmit queues — Each queue (either Tx or Rx) can be mapped to one of the 8 interrupt causes bits (RxTxQ) available in this register according to the mapping in the IVAR registers
2. Indication for the TCP timer interrupt.
3. Legacy and other indications — When any interrupt in the Interrupt Cause register is active.

Writing a 1b clears the corresponding bit in this register. Reading this register auto-clears all bits.

7.3.2.6.2 MSI-X Mode (GPIE.Multiple_MSIX = 1)

This register records the interrupt vectors currently emitted. In this mode only the first 10 bits are valid.

For all the subsequent registers, in MSI-X mode, each bit controls the behavior of one vector.

Bits in this register can be configured to auto-clear when the MSI-X interrupt message is sent, in order to minimize driver overhead when using MSI-X interrupt signaling.

7.3.2.7 Extended Interrupt Cause Set Register (EICS)

This register enables the software device driver to set *EICR* bits. Writing a 1b in a *EICS* bit causes the corresponding bit in the *EICR* register to be set. Used usually to re-arm interrupts that the software didn't have time to handle in the current interrupt routine.

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

Interrupts appear on PCIe only if the interrupt cause bit is a one and the corresponding interrupt mask bit is a one. Software blocks assertion of an interrupt by clearing the corresponding bit in the mask register. The cause bit stores the interrupt event regardless of the state of the mask bit. Different Clear (*EIMC*) and set (*EIMS*) registers make this register more "thread safe" by avoiding a read-modify-write operation on the mask register. The mask bit is set for each bit written to a one in the set register (*EIMS*) and cleared for each bit written in the clear register (*EIMC*). Reading the set register (*EIMS)* returns the current mask register value.

7.3.2.9 Extended Interrupt Auto Clear Enable Register (EIAC)

Each bit in this register enables clearing of the corresponding bit in *EICR* following interrupt generation. When a bit is set, the corresponding bit in the *EICR* register is automatically cleared following an interrupt. This feature should only be used in MSI-X mode.

When used in conjunction with MSI-X interrupt vector, this feature allows interrupt cause recognition, and selective interrupt cause, without requiring software to read or write the *EICR* register; therefore, the penalty related to a PCIe read or write transaction is avoided.

See section 7.3.5 for additional information on the interrupt cause reset process.

7.3.2.10 Extended Interrupt Auto Mask Enable Register (EIAM)

Each bit set in this register enables clearing of the corresponding bit in the extended mask register following read or write-to-clear to *EICR*. It also enables setting of the corresponding bit in the extended mask register following a write-to-set to *EICS*.

This mode is provided in case MSI-X is not used, and therefore auto-clear through *EIAC* register is not available.

In MSI-X mode, the driver software might set the bits of this register to select mask bits that must be reset during interrupt processing. In this mode, each bit in this register enables clearing of the corresponding bit in *EIMC* following interrupt generation.

7.3.2.11 GPIE

There are a few bits in the *GPIE* register that define the behavior of the interrupt mechanism. The setting of these bits is different in each mode of operation. The following table describes the recommended setting of these bits in the different modes:

Table 7-51. Settings for Different Interrupt Modes

Field	Bit(s)	Initial Value	Description	INT-x/ MSI + Legacy	INT-x/ MSI + Extend	MSI-X Multi vector	MSI-X Single vector
NSICR	0	0b	**Non Selective Interrupt clear on read**: When set, every read of *ICR* clears the *ICR* register. When this bit is cleared, an *ICR* read causes the *ICR* register to be cleared only if an actual interrupt was asserted or if *IMS* = 0x0.	0b[1]	1b	1b	1b
Multiple_ MSIX	4	0b	**Multiple_MSIX - multiple vectors:** 0b = non-MSIX or MSI-X with 1 vector *IVAR* maps Rx/Tx causes to 8 *EICR* bits, but MSIX[0] is asserted for all. 1b = MSIX mode, *IVAR* maps Rx/Tx causes to 10 EICR bits.	0b	0b	1b	0b
EIAME	30	0b	**EIAME:** When set, upon firing of an MSI-X message, mask bits set in *EIAM* associated with this message are cleared. Otherwise, *EIAM* is used only upon read or write of *EICR/EICS* registers.	0b	0b	1b	1b
PBA_ support	31	0b	**PBA support:** When set, setting one of the extended interrupts masks via *EIMS* causes the PBA bit of the associated MSI-X vector to be cleared. Otherwise, the 82580 behaves in a way that supports legacy INT-x interrupts. Should be cleared when working in INT-x or MSI mode and set in MSI-X mode.	0b	0b	1b	1b

[1] In systems where interrupt sharing is not expected, the *NSICR* bit can be set by legacy drivers also.

Note: As this register affects the way the hardware interprets write operations to other interrupt control registers, it should be set to the correct mode before accessing other interrupt control registers.

7.3.3 MSI-X and Vectors

MSI-X defines a separate optional extension to basic MSI functionality. Compared to MSI, MSI-X supports a larger maximum number of vectors 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 82580 interrupt causes into an interrupt vector that is conveyed by the 82580 as a posted-write PCIe transaction. Mapping of an interrupt cause into an MSI-X vector is determined by system software (a device driver) through a translation table stored in the MSI-X Allocation registers. Each entry of the allocation registers defines the vector for a single interrupt cause.

There are 18 extended interrupt causes that exist in the 82580:

1. 16 traffic causes — 8 Tx, 8 Rx.
2. TCP timer
3. Other causes — Summarizes legacy interrupts into one extended cause.

7.3.4 Interrupt Moderation

An interrupt is generated upon receiving of incoming packets, as throttled by the EITR registers (see Section 8.8.14). There is an EITR register per MSI-X vector.

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

When multi vector MSI-X is not activated, the interrupt moderation is controlled by register EITR[0].

Software can use EITR to limit the rate of delivery of interrupts to the host CPU. This register provides a guaranteed inter-interrupt delay between interrupts asserted by the network controller, regardless of network traffic conditions.

The following formula converts the inter-interrupt interval value to the common 'interrupts/sec.' performance metric:

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

Note: In the 82580 the interval granularity is 1 μsec so some of the LSB bits of the interval are used for the low latency interrupt moderation.

For example, if the interval is programmed to 125d, the network controller guarantees the CPU is not interrupted by the network controller for at least 125 μs from the last interrupt. In this case, the maximum observable interrupt rate from the adapter should not exceed 8000 interrupts/sec.

Inversely, inter-interrupt interval value can be calculated as:

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

The optimal performance setting for this register is system and configuration specific.

The Extended Interrupt Throttle Register should default to zero upon initialization and reset. It loads in the value programmed by the software after software initializes the device.

When software wants to force an immediate interrupt, for example after setting a bit in the EICR with the Extended Interrupt Cause Set register, a value of 0 can be written to the Counter to generate an interrupt immediately.   This write should include re-writing the *Interval* field with the desired constant, as it is used to reload the Counter immediately for the next throttling interval.

The 82580 implements interrupt moderation to reduce the number of interrupts software processes. The moderation scheme is based on the EITR (Interrupt Throttle Register). Whenever an interrupt event happens, the corresponding bit in the EICR is activated. However, an interrupt message is not sent out on the PCIe interface until the EITR counter assigned to that *EICR* bit has counted down to zero. As soon as the interrupt is issued, the EITR counter is reloaded with its initial value and the process repeats again. The interrupt flow should follow the diagram below:

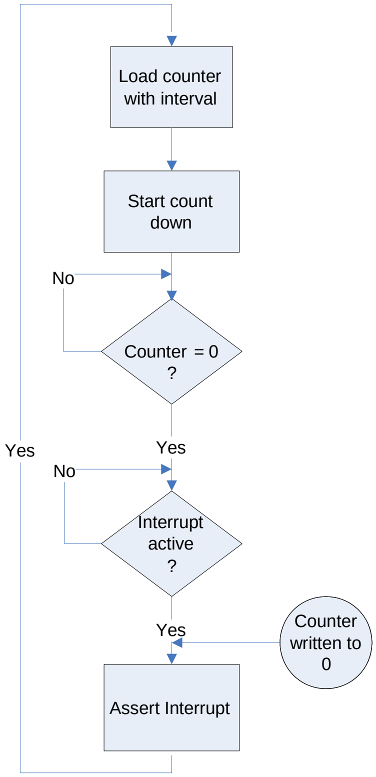


Figure 7-15. Interrupt Throttle Flow Diagram

EITR is designed to guarantee the total number of interrupts per second so for cases where the 82580 is connected to a network with low traffic load, if the EITR counter counted down to zero and no interrupt event has happened, then the EITR counter is not re-armed but stays at zero. Thus, the next interrupt event triggers an interrupt immediately. That scenario is illustrated as “Case B” below.

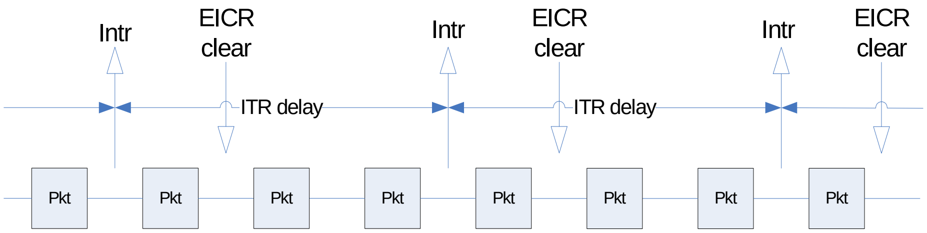


Figure 7-16. Case A: Heavy Load, Interrupts Moderated

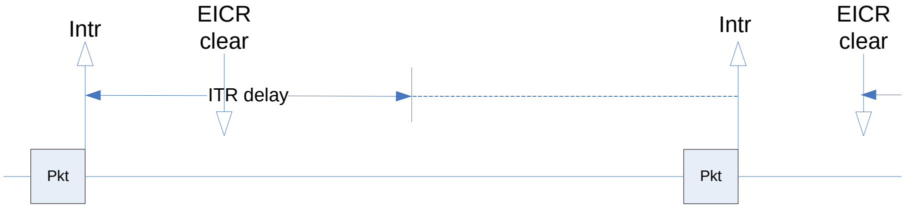


Figure 7-17. Light load, Interrupts Immediately on Packet Receive

7.3.5 Clearing Interrupt Causes

The 82580 has three methods available to clear EICR bits: Autoclear, clear-on-write, and clear-on-read. ICR bits might only be cleared with clear-on-write or clear-on-read.

7.3.5.1 Auto-Clear

In systems that support MSI-X, the interrupt vector allows the interrupt service routine to know the interrupt cause without reading the EICR. With interrupt moderation active, software load from spurious interrupts is minimized. In this case, the software overhead of a I/O read or write can be avoided by setting appropriate EICR bits to autoclear mode by setting the corresponding bits in the Extended Interrupt Auto-clear Enable Register (EIAC).

When auto-clear is enabled for an interrupt cause, the *EICR* bit is set when a cause event mapped to this vector occurs. When the EITR Counter reaches zero, the MSI-X message is sent on PCIe. Then the *EICR* bit is cleared and enabled to be set by a new cause event. The vector in the MSI-X message signals software the cause of the interrupt to be serviced.

It is possible that in the time after the *EICR* bit is cleared and the interrupt service routine services the cause, for example checking the transmit and receive queues, that another cause event occurs that is then serviced by this ISR call, yet the *EICR* bit remains set. This results in a "spurious interrupt". Software can detect this case, for example if there are no entries that require service in the transmit and receive queues, and exit knowing that the interrupt has been automatically cleared. The use of interrupt moderations through the EITR register limits the extra software overhead that can be caused by these spurious interrupts.

7.3.5.2 Write to Clear

In the case where the driver wishes to configure itself in MSI-X mode to not use the "auto-clear" feature, it might clear the *EICR* bits by writing to the *EICR* register. Any bits written with a 1b is cleared. Any bits written with a 0b remain unchanged.

7.3.5.3 Read to Clear

The EICR and ICR registers are cleared on a read.

Note that the driver should never do a read-to-clear of the *EICR* when in MSI-X mode, since this might clear interrupt cause events which are processed by a different interrupt handler (assuming multiple vectors).

7.3.6 Rate Controlled Low Latency Interrupts (LLI)

There are some types of network traffic for which latency is a critical issue. For these types of traffic, interrupt moderation hurts performance by increasing latency between the time a packet is received by hardware and the time it is handled to the host operating system. This traffic can be identified by the 2-tuple value, in conjunction with Control Bits and specific size. In addition packets with specific Ethernet types, TCP flag or specific VLAN priority might generate an immediate interrupt.

Low latency interrupts shares the filters used by the queueing mechanism described in Section 7.1.1. Each of these filters, in addition to the queueing action might also indicate matching packets might generate immediate interrupt.

If a received packet matches one of these filters, hardware should interrupt immediately, overriding the interrupt moderation by the EITR counter.

Each time a Low Latency Interrupt is fired, the EITR interval is loaded and down-counting starts again.

The logic of the low latency interrupt mechanism is as follows:

- There are 8 2-tuple filters. The content of each filter is described in Section 7.1.1.5. The immediate interrupt action of each filter can be enabled or disabled. If one of the filters detects an adequate packet, an immediate interrupt is issued.
- There are 8 flex filters. The content of each filter is described in Section 7.1.1.6. The immediate interrupt action of each filter can be enabled or disabled. If one of the filters detects an adequate packet, an immediate interrupt is issued.
- When VLAN priority filtering is enabled, VLAN packets must trigger an immediate interrupt when the VLAN Priority is equal to or above the VLAN priority threshold. This is regardless of the status of the 2-tuple or Flex filters.
- The SYN packets filter defined in Section 7.1.1.7 and the ethernet type filters defined in section Section 7.1.1.4 might also be used to indicate low latency interrupt conditions.

Note: Immediate interrupts are available only when using advanced receive descriptors and not for legacy descriptors.

Note: Packets that are dropped or have errors do not cause a Low Latency Interrupt.

7.3.6.1 Rate Control Mechanism

In a network with lots of latency sensitive traffics the Low Latency Interrupt can eliminate the Interrupt throttling capability by flooding the Host with too many interrupts (more than the Host can handle).

In order to mitigate the above, the 82580 supports a credit base mechanism to control the rate of the Low Latency Interrupts.

Rules:

- The default value of each counter is 0b (no moderation). This also preserves backward compatibility.
- The counter increments at a configurable rate, and saturates at the maximum value (31d).
  - — The configurable rate granularity is 4 µs (250K interrupt/sec. down to 250K/32 ~ 8K interrupts per sec.).
- A LLI might be issued as long as the counter value is strictly positive (> zero).
  - — The credit counter allows bursts of low latency interrupts but the interrupt average are not more than the configured rate.
- Each time a Low Latency Interrupt is fired the credit counter decrements by one.

- Once the counter reaches zero, a low latency interrupt cannot be fired
  - Must wait for the next ITR expired or for the next incrementing of this counter (if the EITR expired happened first the counter does not decrement).

The *EITR* and *GPIE* registers manage rate control of *LLI*:

- The *LL Interval* field in the GPIE register controls the rate of credits
- The 5-bit *LL Counter* field in the *EITR* register contains the credits

7.3.7 TCP Timer Interrupt

7.3.7.1 Introduction

The TCP Timer interrupt provides an accurate and efficient way for a periodic timer to be implemented using hardware. The driver would program a timeout value (usual value of 10 ms), and each time the timer expires, hardware sets a specific bit in the *EICR*. When an interrupt occurs (due to normal interrupt moderation schemes), software reads the *EICR* and discovers that it needs to process timer events during that DPC.

The timeout should be programmable by the driver, and the driver should be able to disable the timer interrupt if it is not needed.

7.3.7.2 Description

A stand-alone down-counter is implemented. An interrupt is issued each time the value of the counter is zero.

The software is responsible for setting initial value for the timer in the *TCPTIMER.Duration* field. Kick-starting is done by writing a 1b to the *TCPTIMER.KickStart* bit.

Following the kick-start, an internal counter is set to the value defined by the *TCPTIMER.Duration* field. Then during the count operation, the counter is decreased by one each millisecond. When the counter reaches zero, an interrupt is issued (see EICR register Section 8.8.3). The counter re-starts counting from its initial value if the *TCPTIMER.Loop* field is set.

7.4 802.1q VLAN Support

The 82580 provides several specific mechanisms to support 802.1q VLANs:

- Optional adding (for transmits) and stripping (for receives) of IEEE 802.1q VLAN tags.
- Optional ability to filter packets belonging to certain 802.1q VLANs.
- Double VLAN Support.

7.4.1 802.1q VLAN Packet Format

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

Table 7-52. Comparing Packets

802.3 Packet	#Octets		802.1q VLAN Packet	#Octets
DA	6		DA	6
SA	6		SA	6
Type/Length	2		802.1q Tag	4
Data	46-1500		Type/Length	2
CRC	4		Data	46-1500
			CRC*	4

Note: The CRC for the 802.1q tagged frame is re-computed, so that it covers the entire tagged frame including the 802.1q tag header. Also, max frame size for an 802.1q VLAN packet is 1522 octets as opposed to 1518 octets for a normal 802.3z Ethernet packet.

7.4.2 802.1q Tagged Frames

For 802.1q, the *Tag Header* field consists of four octets comprised of the Tag Protocol Identifier (TPID) and Tag Control Information (TCI); each taking 2 octets. The first 16 bits of the tag header makes up the TPID. It contains the "protocol type" which identifies the packet as a valid 802.1q tagged packet.

The two octets making up the TCI contain three fields:

- User Priority (UP)
- Canonical Form Indicator (CFI). Should be 0b for transmits. For receives, the device has the capability to filter out packets that have this bit set. See the *CFIEN* and *CFI* bits in the *RCTL* described in Section 8.10.1.
- VLAN Identifier (VID)

The bit ordering is shown below:

Table 7-53. TCI Bit Ordering

Octet 1								Octet 2							
UP			CFI	VID											

7.4.3 Transmitting and Receiving 802.1q Packets

7.4.3.1 Adding 802.1q Tags on Transmits

Software might command the 82580 to insert an 802.1q VLAN tag on a per packet or per flow basis. If the *VLE* bit in the transmit descriptor is set to 1b, then the 82580 inserts a VLAN tag into the packet that it transmits over the wire. The *Tag Protocol Identifier (TPID)* field of the 802.1q tag comes from the *VET* register. 8021.Q tag insertion is done in different ways for legacy and advanced Tx descriptors:

- Legacy Transmit Descriptors:, The Tag Control Information (TCI) of the 802.1q tag comes from the *VLAN* field (see Figure 7-9) of the descriptor. Refer to Table 7-27, for more information regarding hardware insertion of tags for transmits.
- Advanced Transmit Descriptor: The Tag Control Information (TCI) of the 802.1q tag comes from the VLAN *Tag* field (see Table 7.2.2.2.1) of the advanced context descriptor. The *IDX* field of the advanced Tx descriptor should be set to the adequate context.

7.4.3.2 Stripping 802.1q Tags on Receives

Software might instruct the 82580 to strip 802.1q VLAN tags from received packets. If the *CTRL.VME* bit is set to 1b, and the incoming packet is an 802.1q VLAN packet (its *Ethernet Type* field matched the *VET*), then the 82580 strips the 4 byte VLAN tag from the packet, and stores the TCI in the VLAN *Tag* field (see Figure 7-5 and See "Receive UDP Fragmentation Checksum) of the receive descriptor.

The 82580 also sets the *VP* bit in the receive descriptor to indicate that the packet had a VLAN tag that was stripped. If the *CTRL.VME* bit is not set, the 802.1Q packets can still be received if they pass the receive filter, but the VLAN tag is not stripped and the *VP* bit is not set. Refer Figure 7-18 for more information regarding receive packet filtering.

7.4.4 802.1q VLAN Packet Filtering

VLAN filtering is enabled by setting the *RCTL.VFE* bit to 1b. If enabled, hardware compares the type field of the incoming packet to a 16-bit field in the *VLAN Ether Type* (*VET*) register. If the VLAN type field in the incoming packet matches the *VET* register, the packet is then compared against the VLAN Filter Table Array (*VFTA[127:0]*) for acceptance.

The 82580 provides exact VLAN filtering for VLAN tags for host traffic and VLAN tags for manageability traffic.

Host VLAN filtering:

The *Virtual LAN ID* field indexes a 4096 bit vector. If the indexed bit in the vector is one; there is a Virtual LAN match. Software might set the entire bit vector to ones if the node does not implement 802.1q filtering. The register description of the VLAN Filter Table Array is described in detail in Section 8.10.18.

In summary, the 4096-bit vector is comprised of 128, 32-bit registers. The *VLAN Identifier (VID)* field consists of 12 bits. The upper 7 bits of this field are decoded to determine the 32-bit register in the VLAN Filter Table Array to address and the lower 5 bits determine which of the 32 bits in the register to evaluate for matching.

Manageability VLAN filtering:

The BMC configures the 82580 with eight different manageability VIDs via the Management VLAN TAG Value [7:0] - MAVTV[7:0] registers and enables each filter in the MDEF register.

Two other bits in the Receive Control register (see Section 8.10.1), *CTRL.CFIEN* and *CTRL.CFI*, are also used in conjunction with 802.1q VLAN filtering operations. *CTRL.CFIEN* enables the comparison of the value of the *CFI* bit in the 802.1q packet to the Receive Control register *CTRL.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.

The following table lists reception actions per control bit settings.

Figure 7-18. Packet Reception Decision Table

Is packet 802.1q?	CTRL. VME	RCTL. VFE	Action
No	X[1]	X[1]	Normal packet reception
Yes	0b	0b	Receive a VLAN packet if it passes the standard MAC address filters (only). Leave the packet as received in the data buffer. *VP* bit in receive descriptor is cleared.

Figure 7-18. Packet Reception Decision Table

Yes	0b	1b	Receive a VLAN packet if it passes the standard filters and the VLAN filter table. Leave the packet as received in the data buffer (the VLAN tag would not be stripped). *VP* bit in receive descriptor is cleared.
Yes	1b	0b	Receive a VLAN packet if it passes the standard filters (only). Strip off the VLAN information (four bytes) from the incoming packet and store in the descriptor. Sets *VP* bit in receive descriptor.
Yes	1b	1b	Receive a VLAN packet if it passes the standard filters and the VLAN filter table. Strip off the VLAN information (four bytes) from the incoming packet and store in the descriptor. Sets *VP* bit in receive descriptor.

1. X - Don't care

Note: A packet is defined as a VLAN/802.1q packet if its type field matches the VET.

7.4.5 Double VLAN Support

The 82580 supports a mode where all received and sent packet have at least one VLAN tag in addition to the regular tagging which might optionally be added. This mode is used for systems where the switches add an additional tag containing switching information.

This mode is activated by setting *CTRL_EXT.EXT_VLAN* bit. The default value of this bit is set according to the *EXT_VLAN* bit in the *Initialization Control 3* EEPROM word for ports 0 to 3. See Section 6.2.21 for more information.

The type of the VLAN tag used for the additional VLAN is defined in the *VET.VET_EXT* field.

7.4.5.1 Transmit Behavior With External VLAN

It is expected that the driver include the external VLAN header as part of the transmit data structure. The software may post the internal VLAN header as part of the transmit data structure or embedded in the transmit descriptor (see Section 7.2.2 for details). The 82580 does not relate to the external VLAN header other than the capability of "skipping" it for parsing of inner fields.

Notes:

- When the *CTRL_EXT.EXT_VLAN* bit is set the VLAN header in a packet that carries a single VLAN header is treated as the external VLAN.
- When the *CTRL_EXT.EXT_VLAN* bit is set The 82580 expects that any transmitted packet has at least the external VLAN added by the software. For those packets where an external VLAN is not present, any offload that relates to inner fields to the EtherType may not be provided.

7.4.5.2 Receive Behavior With External VLAN

When a port of the 82580 is working in this mode, the 82580 assumes that all packets received by this port have at least one VLAN, including packet received or sent on the manageability interface.

One exception to this rule are flow control PAUSE packets which are not expected to have any VLAN. Other packets may contain no VLAN, however a received packet that does not contain the first VLAN is forwarded to the host but filtering and offloads are not applied to this packet.

See the Table 7-54 for the supported receive processing functionality when the device is set to "Double VLAN" mode (*CTRL_EXT.EXT_VLAN* bit is set).

Stripping of VLAN is done on the second internal VLAN if it exists. All the filtering functions of the 82580 ignore the first VLAN in this mode.

The presence of a first VLAN tag is indicated it in the *RDESC.STATUS.VEXT* bit.

Queue assignment of the Rx packets is not affected by the external VLAN header. It may depend on the internal VLAN, MAC address or any upper layer content as described in Section 7.1.1.

Table 7-54. Receive Processing in Double VLAN Mode

VLAN Headers	Status.VEXT	Status.VP	Packet Parsing	Rx offload functions
External and internal	1	1	+	+
Internal Only	Not supported			
V-Ext	1	0	+	+
None[1]	0	0	+ (flow control only)	-

[1] A few examples for packets that may not carry any VLAN header may be: Flow control and Priority Flow Control; LACP; LLDP; GMRP; 802.1x packets

7.5 Configurable LED Outputs

The 82580 implements 4 output drivers intended for driving external LED circuits per port. 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 is indicated on that output. In addition, each LED can be individually configured for output polarity as well as for blinking versus non-blinking (steady-state) indication.

The configuration for LED outputs is specified via the *LEDCTL* register. Furthermore, the hardware-default configuration for all the LED outputs, can be specified via EEPROM fields, thereby supporting LED displays configurable to a particular OEM preference.

Each of the 4 LED's might be configured to use one of a variety of sources for output indication. The MODE bits control the LED source as described in Table 7-55.

The IVRT bits allow the LED source to be inverted before being output or observed by the blink-control logic. LED outputs are assumed to normally be connected to the negative side (cathode) of an external LED.

The BLINK bits control whether the LED should be blinked (on for 200ms, then off for 200ms) while the LED source is asserted. The blink control might be especially useful for ensuring that certain events, such as ACTIVITY indication, cause LED transitions, which are sufficiently visible by a human eye.

Note: When LED Blink mode is enabled the appropriate LED Invert bit should be set to 0b.

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.

The dynamic LED modes (FILTER_ACTIVITY, LINK/ACTIVITY, COLLISION, ACTIVITY, PAUSED) should be used with LED Blink mode enabled.

7.5.1 MODE Encoding for LED Outputs

Table 7-55 lists the MODE encoding for LED outputs used to select the desired LED signal source for each LED output.

Table 7-55. Mode Encoding for LED Outputs

Mode	Selected Mode	Source Indication
0000b	LINK_10/1000	Asserted when either 10 or 1000 Mb/s link is established and maintained.
0001b	LINK_100/1000	Asserted when either 100 or 1000 Mb/s link is established and maintained.
0010b	LINK_UP	Asserted when any speed link is established and maintained.
0011b	FILTER_ACTIVITY	Asserted when link is established and packets are being transmitted or received that passed MAC filtering.
0100b	LINK/ACTIVITY	Asserted when link is established and when there is no transmit or receive activity.
0101b	LINK_10	Asserted when a 10 Mb/s link is established and maintained.
0110b	LINK_100	Asserted when a 100 Mb/s link is established and maintained.
0111b	LINK_1000	Asserted when a 1000 Mb/s link is established and maintained.
1000b	SDP_MODE	LED activation is a reflection of the SDP signal. SDP0, SDP1, SDP2, SDP3 are reflected to LED0, LED1, LED2, LED3 respectively.
1001b	FULL_DUPLEX	Asserted when the link is configured for full duplex operation (de-asserted in half-duplex).
1010b	COLLISION	Asserted when a collision is observed.
1011b	ACTIVITY	Asserted when link is established and packets are being transmitted or received.
1100b	BUS_SIZE	Asserted when the 82580 detects a 4-lane PCIe connection.
1101b	PAUSED	Asserted when the 82580’s transmitter is flow controlled.
1110b	LED_ON	Always high (Asserted)
1111b	LED_OFF	Always low (De-asserted)

7.6 Memory Error Correction and Detection

The 82580 main internal memories are protected by error correcting code or parity bits. The larger memories or critical memories are protected by an error correcting code (ECC). The smaller memories are protected either with an error correcting code (ECC for critical memories) or by parity.

Software should set the *ECC Enable* field in the *RPBECCSTS* and *TPBECCSTS* registers, to enable ECC error correction for the internal packet buffer memories. To enable ECC error correction in the internal PCIe memories, Software should set the ECC enable fields in the PCIEECCCTL register. To enable parity checks of other internal memories software should set the enable bits in the *DTPARC*, *DRPARC*, *DDPARC*, *LANPERRCTL* and *PCIEERRCTL* registers.

Correctable errors, detected by the internal ECC circuitry, are silently corrected. Correctable errors in Receive and Transmit buffers are counted in the *RPBECCSTS.Corr_err_cnt* and *TPBECCSTS.Corr_err_cnt* fields respectively. Uncorrectable ECC errors in the PCIe are logged in the *PCIEECCSTS* register.

The 82580 detects and logs uncorrectable parity errors on memories that only have parity protection. Detection of uncorrectable parity errors is indicated by setting the *ICR.FER* bit, that reports occurrence of fatal parity errors. More in-depth diagnostics can be done by reading the *PEIND* register to detect the section in which the parity error occurred, and by reading the *DTPARS*, *DRPARS*, *DDPARS*, *LANPERRSTS* and *PCIEERRSTS* registers to pinpoint the specific memory with the parity error.

Note: An interrupt is generated on occurrence of a fatal memory error if the appropriate mask bits in the *PEINDM* register are set and the *IMS.FER* Mask bit is set.

If a parity error was detected in one of the internal control memories of the DMA, PCIe or LAN port clusters, the consistency of the receive/transmit flow can not be guaranteed any more. In this case the traffic on the PCIe interface is stopped, since this is considered a fatal error. Software should initiate a Device Reset (*CTRL.DEV_RST* - See Section 4.3.1) and re-initialize the port, to enable transactions on the PCIe interface to continue. If a parity error is detected on the Manageability cluster, PCIe traffic is not effected but an internal reset to the Manageability cluster is generated.

Note: Following assertion of *CTRL.DEV_RST* software should wait for the *GCR.DEV_RST in progress* bit to be cleared before re-initializing the port.

7.6.1 Management Parity Errors

Parity errors in the Management cluster are reported by setting the *ICR.FER* bit, when management parity error reporting is enabled by setting the *PEINDM.mng_parity_fatal_ind bit*. Software can program interrupt generation by setting the *IMS.FER* interrupt mask bit. To enable parity check of memories in the Management logic, the Parity check enable bits in the *SMBus Flags* EEPROM word should be set.

Note: An internal Firmware reset is issued, if enabled by the *PARITY_ERR_RST_EN* bit in the *Management HW Config Control* EEPROM word, following detection of a parity error in the Management cluster.

7.7 CPU affinity Features

7.7.1 Direct Cache Access (DCA)

7.7.1.1 DCA Description

Direct Cache Access (DCA) is a method to improve network I/O performance by placing some posted inbound writes indirectly within CPU cache. DCA requires that memory writes go to host memory and then the processor prefetch the cache lines specified by the memory write. Through research and experiments, DCA has been shown to reduce CPU Cache miss rates significantly.

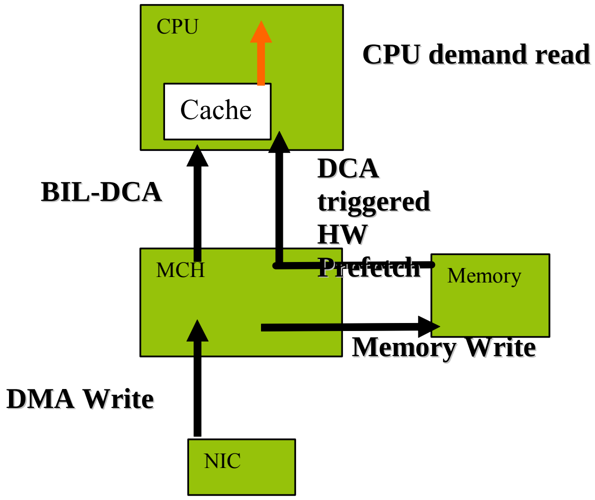


Figure 7-19. Diagram of DCA Implementation on FSB System

As shown in Figure 7-19, DCA provides a mechanism where the posted write data from an I/O device, such as an Ethernet NIC, can be placed into CPU cache with a hardware pre-fetch. This mechanism is initialized upon a power good reset. A software device driver for the I/O device configures the I/O device for DCA and sets up the appropriate DCA target ID for the device to send data. The device will then encapsulate that information in PCIe TLP headers, in the *TAG* field, to trigger a hardware pre-fetch by the MCH /IOH to the CPU cache.

DCA implementation is controlled by separated registers (RXCTL and TXCTL) for each receive and transmit queue. In addition, a *DCA Enable* bit can be found in the DCA_CTRL register, and a DCA_ID register can be found for each port, in order to make visible the function, device, and bus numbers to the driver.

The RXCTL and TXCTL registers can be written by software on the fly and can be changed at any time. When software changes the register contents, hardware applies changes only after all the previous packets in progress for DCA have been completed.

However, in order to implement DCA, the 82580 has to be aware of the Crystal Beach version used. Software driver must initialize the 82580 to be aware of the Crystal Beach version. A register named DCA_CTRL is used in order to properly define the system configuration.

There are 2 modes for DCA implementation:

1. Legacy DCA: The DCA target ID is derived from CPU ID.
2. DCA: The DCA target ID is derived from APIC ID.

The software driver selects one of these modes through the DCA_mode register.

The details of both modes are described below.

7.7.1.2 Details of Implementation

7.7.1.2.1 PCIe Message Format for DCA

Figure 7-20 shows the format of the PCIe message for DCA.

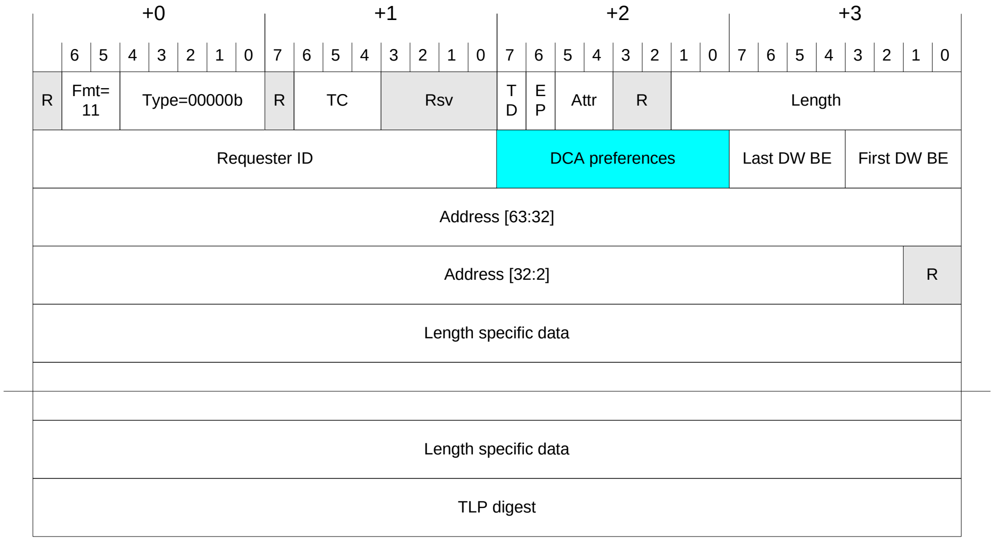


Figure 7-20. PCIe Message Format for DCA

The DCA preferences field has the following formats.

Table 7-56. Legacy DCA Systems

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.
7:5	Reserved	Reserved

Table 7-57. DCA Systems

Bits	Name	Description
7:0	DCA target ID	0000.0000b: DCA is disabled Other: Target Core Id derived from APIC Id. The method for this is described in DCA Platform Architecture Specification, section 7.3.1 (Anacapa reference number 16802)

Note: All functions within a the 82580 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 “00000000”.

7.7.2 TLP Process Hints (TPH)

The 82580 supports the TPH capability defined in the PCI Express specification (See Section 9.6.3). It does not support Extended TPH requests.

On the PCIe link existence of a TLP Process Hint (TPH) is indicated by setting the TH bit in the TLP header. Using the PCIe TLP Steering Tag (ST) and Processing Hints (PH) fields, The 82580 can provide hints to the root complex about the destination (socket ID) and about data access patterns (locality in Cache), when executing DMA memory writes or read operations. Supply of TLP Processing Hints facilitates optimized processing of transactions that target Memory Space.

The 82580 supports a steering table with 8 entries in the PCIe TPH capability structure (See Section 9.6.3.4). The PCIe Steering table can be used by Software to provide Steering Tag information to the Device via the *TXCTL.CPUID* and *RXCTL.CPUID* fields.

To enable TPH usage:

1. For a given function, the *TPH Requester Enable* bit in the PCIe configuration *TPH Requester Control Register* should be set.
2. Appropriate TPH Enable bits in *RXCTL* or *TXCTL* registers should be set.
3. Processing hints should be programmed in the *DCA_CTRL.Desc_PH* and *DCA_CTRL.Data_PH* Processing hints (PH) fields.
4. Steering information should be programed in the CPUID fields in the *RXCTL* and *TXCTL* registers.

The Processing hints (PH) and Steering Tags (ST) are set according to the characteristics of the traffic as described in Table 7-58.

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

Note: Per queue, the DCA and TPH features are exclusive. Software can enable either the DCA feature or the TPH feature for a given queue.

7.7.2.1 Steering tag and Processing hint Programming

The following table describes how the Steering tag (socket ID) and Processing hints are generated and how TPH operation is enabled for different types of DMA traffic.

Table 7-58. Steering tag and Processing hint programming

Traffic type	ST Programming	PH value	Enable
Transmit descriptor write back or head write back	TXCTL.CPUID[1]	*DCA_CTRL.Desc_PH*[2]	*Tx Descriptor Writeback TPH EN* field in *TXCTL*.
Receive data buffers write	RXCTL.CPUID[1]	*DCA_CTRL.Data_PH*[3]	*RX Header TPH EN or* *Rx Payload TPH EN* fields in *RXCTL*.
Receive descriptor writeback	RXCTL.CPUID[1]	*DCA_CTRL.Desc_PH*[2]	*RX Descriptor Writeback TPH EN* field in *RXCTL*.
Transmit descriptor fetch	TXCTL.CPUID[4]	*DCA_CTRL.Desc_PH*[2]	*Tx Descriptor Fetch TPH EN* field in *TXCTL*.
Receive descriptor fetch	RXCTL.CPUID[2]	*DCA_CTRL.Desc_PH*[2]	*Rx Descriptor fetch TPH EN* field in *RXCTL*.
Transmit packet read	TXCTL.CPUID[2]	*DCA_CTRL.Data_PH*[3]	*Tx Packet TPH EN field in TXCTL*.

[1] the driver should always set bits [7:3] to zero and place Socket ID in bits [2:0].

[2] Default is 00b (Bidirectional data structure).

[3] Default is 10b (Target).

[4] the hints are always zero.

7.8 Virtualization

7.8.1 Overview

I/O virtualization is a mechanism to share I/O resources among several consumers. For example, in a virtual system, multiple operating systems are loaded and each executes as though the whole system's resources were at its disposal. However, for the limited number of I/O devices, this presents a problem because each operating system might be in a separate memory domain and all the data movement and device management has to be done by a VMM (Virtual Machine Monitor). VMM access adds latency and delay to I/O accesses and degrades I/O performance. Virtualized devices are designed to reduce the burden of VMM by making certain functions of an I/O device shared and thus can be accessed directly from each guest operating system or Virtual Machine (VM).

Two modes to support operation in a Virtualized environment were implemented in previous products:

1. Direct assignment of part of the port resources to different guest OSes using the PCI sig SR-IOV standard. Also known as “Native mode” or pass through mode. This mode is referenced as IOV mode through this chapter
2. Central management of the networking resources by an IOVM or by the VMM. This mode is referenced as VMDq mode. Two VMDq modes exist, VMDq1 and a more advanced version named VMDq2.

The 82580 supports fully VMDq1 mode and partially VMDq2 mode and does not support SR-IOV. In a virtualized environment, the 82580 serves up to 8 virtual machines (VMs) per port. The 82580's 8 queues can be accessed by 8 different VMs if configured properly. When the 82580 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.

The following section describes the support the 82580 provides for VMDq.

7.8.1.1 Virtualized System Overview

The following drawing describes the various elements involved in the I/O process in a virtualized system. Figure 7-21 describes the flow in software VMDq operation mode.

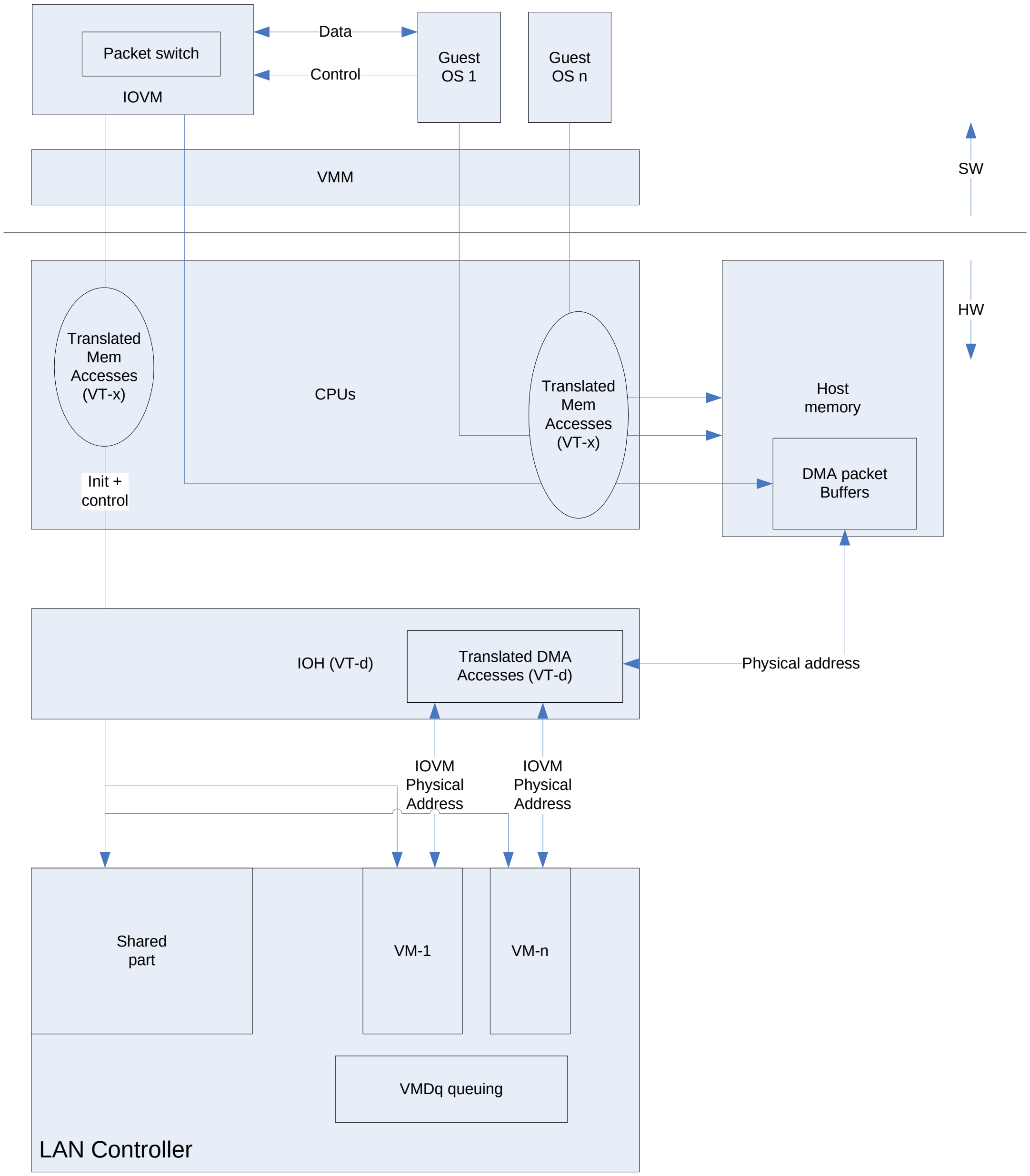


Figure 7-21. IOVM (VMDq) System

7.8.1.2 VMDq Supported Features

The 82580 supports a subset of the VMDq virtualization features supported in the 82576. The following table compares the 82580 virtualization features with the 82580's virtualization features

Table 7-59. 82580 Versus the 82576 VMDq Support

Feature	82580 VMDq Support	82576 VMDq2 Support
Queues	8	16
Pools	8 (single queue)	8
MAC addresses	24	24
Queuing to pool method	SA or VLAN or (SA and VLAN)	SA or VLAN or (SA and VLAN)
RSS in pool	RSS not supported in VMDq	Common redirection table - enable per pool.
VM to VM Switching	No	Yes
Broadcast and multicast Replication	Yes (Receive only)	Yes
MAC and VLAN Anti spoof protection	No	Yes
VLAN filtering	Global and per pool	Global and per pool
Drop if no pool	Yes	Yes
per pool statistics	Yes	Yes
Per pool offloads	Yes	Yes
Mirroring	Yes (Receive only)	Yes
Long packet filtering	Global and per pool	Global and per pool
Storm Control	Yes	Yes

7.8.1.2.1 Assignment of MSI-X Vectors to VM.

MSI-X vectors are used for three purposes:

1. Differentiation of interrupt causes that avoids the need to read an interrupt cause register.
2. Assignment of different interrupt handling to different CPUs.
3. The implementation of interrupts in the 82580 adds another use of allowing different interrupt moderation rates.

MSI-X vectors are allocated for cause differentiation and interrupt rate differentiation. The interrupt causes the VM driver needs to handle are Tx packet sent per queue, Rx packet received per queue, TCP timer and some special events.

7.8.1.2.2 VM Resource Summary

The 82580 supports 8 VMs per port, each VM can utilize 1 queue pair (Tx/Rx) per VM and 1 MSI-X vectors per VM. If the amount of VMs supported is less than 8, the available resources can be distributed between the active VMs.

7.8.2 Packet Switching (VMDq) Model

7.8.2.1 VMDq Assumptions

To support receive packet replication it is assumed that Broadcast and Multicast traffic volume is relatively low. In the case of large Broadcast and Multicast traffic volume that causes congestion on the PCIe interface, the Storm Control circuitry can be programmed to drop Broadcast and Multicast packets or flow control can be activated to avoid internal receive buffer overflow.

7.8.2.2 VM Selection

The VM selection is done by MAC address and VLAN tag. Broadcast and Multicast packets are forwarded according to the individual setting of each VM and might be replicated to multiple VMs.

7.8.2.2.1 Filtering Capabilities

The following capabilities exists in to decide what is the final destination of each packet in addition to the regular L2 filtering capabilities:

- 24 MAC addresses filters (*RAH/RAL* registers) for both unicast and multicast filtering. These are shared with L2 filtering. For example, the same MAC addresses are used to determine if a packet is received by the switch and to determine the forwarding destination.
- 32 Shared VLAN filters (*VLVF* registers) - each VM can be made member of each VLAN.
- Multicast exact filtering using the existing remaining *RAH/RAL* registers otherwise an imperfect multicast table is shared between VMs.
- 256 hash filtering of multicast addresses shared between the VMs (*MTA* table).
- Promiscuous multicast and enable broadcast per VM.

Note: Packets for which no queueing decision was done and were still accepted by the L2 filtering, are directed to the queue pool of the default VM or dropped.

7.8.2.3 L2 Filtering

L2 filtering is the 1st stage of 3 stages that determine the destination of a received packet. The 3 stages are defined in Section 7.1.1.

All received packets pass the same filtering as in the non virtualized case; regular VLAN filtering using the global VLAN table (*VFTA*) and filtering according to the *RAH/RAL* registers and according to the various promiscuous bits.

Note: Every VLAN tag set in the *VLVF* registers should be asserted also in the *VFTA* table.

The *RCTL.UPE* bit (Promiscuous unicast) is not available per VM and might be modified by the IOVM or VMM.

7.8.2.4 Size Filtering

A packet is defined as undersize if it is smaller than 64 bytes.

A packet is defined as oversize in the following conditions:

- The *RCTL.LPE* bit cleared and one of the following conditions is met:
  - — The packet is bigger than 1518 bytes and there are no VLAN tags in the packet.
  - — The packet is bigger than 1522 bytes and there is one VLAN tag in the packet.
  - — The packet is bigger than 1526 bytes and there are two VLAN tags in the packet.
- The *RCTL.LPE* bit cleared and the packet is bigger than *RLPML.RLPML* bytes.

7.8.2.5 VMDq Receive Packets Switching

Receive packet switching is the 2nd stage of 3 stages that determine the destination of a received packet. The 3 stages are defined in Section 7.1.1.

In this stage the VM is identified by the “pool list” as described in Section 7.8.2.5.1 and Section 7.8.2.5.2.

When working in a virtualized environment, a single receive queue can still be determined by the Ethertype filters. If these filters don't match, then a pool list should be found.

When working in replication mode, broadcast and multicast packets can be forwarded to more than one VM, and can be replicated to more than one receive queue. Replication is enabled by the *Rpl_En* bit in the VT_CTL register.

In virtualization mode, the pool list is a list of one or more VMs to which the packet should be forwarded. The pool list is used in choosing the target queue list except for cases in which high priority filters take precedence. There is a difference in the way the pool list is generated when replication mode is enabled or disabled.

7.8.2.5.1 VMDq Replication Mode Enabled

When replication mode is enabled (VT_CTL.*Rpl_En* = 1), each broadcast/multicast packet can go to more than one pool. Finding the pool list should be done according to the following steps:

1. Exact unicast or multicast match - If there is a match in one of the exact filters (*RAL*/*RAH*), for unicast or multicast packets, take the *RAH.POOLSEL[7:0]* field as a candidate for the pool list.
2. Broadcast - If the packet is a broadcast packet, add pools for which their *VMOLR.BAM* bit (Broadcast Accept Mode) is set.
3. Unicast hash - If the packet is a unicast packet, and the prior steps yielded no pools, check it against the Unicast hash table (*UTA*). If there is a match, add pools for which their *VMOLR.ROPE* bit (Receive Overflow packet enable) is set.
4. Multicast hash - If the packet is a multicast packet and the prior steps yielded no pools, check it against the multicast hash table (*MTA*). If there is a match, add pools for which their *VMOLR.ROMPE* bit (Receive Multicast packet enable) is set.
5. Multicast Promiscuous - If the packet is a multicast packet, take the candidate list from prior steps and add pools for which their *VMOLR.MPE* bit (Multicast Promiscuous Enable) is set.
6. Ignore MAC (VLAN only filtering) - If *VT_CTL.IGMAC* bit is set, then the previous steps are ignored and a full pool list is assumed for the next step.
7. VLAN groups - This step is relevant only if the *RCTL.VFE* bit is set, otherwise it is skipped. Packets should be sent only to VMs that belong to the packet's VLAN group.
   a. Tagged packets: enable only pools in the packet's VLAN group as defined by the VLAN filters - *VLVF[n].VLAN_id* and their pool list - *VLVF[n].POOLSEL[7:0]*.
   b. Untagged packets: enable only pools with their *VMOLR.AUPE* bit set
   c. If there is no match, the pool list should be empty.

Note: In a VLAN network, untagged packets are not expected. Such packets received by the switch should be dropped, unless their destination is a virtual port set to receive these packets. The setting is done through the *VMOLR.AUPE* bit. It is assumed that VMs for which this bit is set are members of a default VLAN and thus only MAC queuing is done on these packets.

8. Default Pool - If the pool list is empty at this stage and the *VT_CTL.Dis_Def_Pool* bit is not set, then set the default pool bit in the target pool list (from VT_CTL.Def_PL).
9. Ethertype filters - If the one of the Ethertype filters (ETQF) matches the packet and queuing action is requested, the VM list is set to the pool pointed by the filter.
10. Mirroring - For each of the 4 mirroring rules add the destination (mirroring) pool (*VMRCTL.MP*) to the pool list according to the following rules:
    a. Pool mirroring - if *VMRCTL.VPME* is set and one of the bits in the pool list matches one of the bits in the VMRVM register.

b. VLAN port mirroring - if *VMRCTL.VLME* is set and the index of the VLAN of the packet in the VLVF table matches one of the bits in the *VMRVLAN* register.

c. Uplink port mirroring - if *VMRCTL.UPME* is set and the pool list is not empty and the packet came from the LAN.

11. Length Limit: If the packet is longer than a legal Ethernet packet, remove from the pool list all the pools for which the *VMOLR.LPE* bit is not set or for which the packet length is larger than the value in the *VMOLR.RLPML* field.

The above process, up to stage 9. can be logically described by the following scheme:

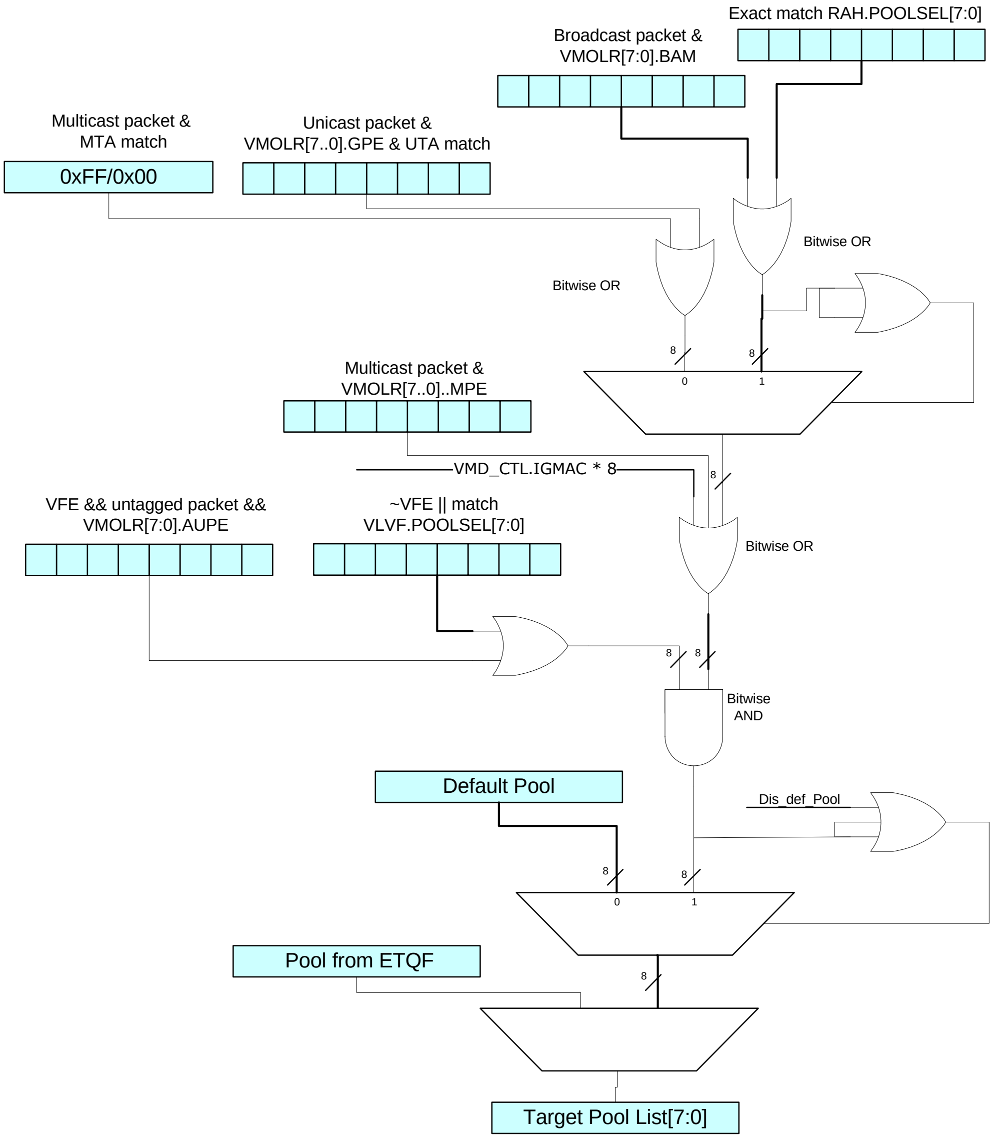


Figure 7-22. Pool List Selection - Replication Enabled

7.8.2.5.2 VMDq Replication Mode Disabled

When replication mode is disabled (VT_CTL.*Rpl_En* = 0), the software should take care of multicast and broadcast packets and check which of the VMs should get them. In this mode the pool list always contains one pool only according to the following steps:

1. Exact unicast or multicast match - If the packet DA matches one of the exact filters (RAL/RAH), take the *RAH.POOLSEL[7:0]* field as a candidate for the pool list.
2. Ignore MAC (VLAN only filtering) - If *VT_CTL.IGMAC* bit is set, then the previous step is ignored and a full pool list is assumed for the next step.
3. Unicast hash - If the packet is a unicast packet, and the prior steps yielded no pools, check it against the Unicast hash table (UTA). If there is a match, add the pool for which the *VMOLR.ROPE* bit (Receive Overflow packet enable) is set. (See software limitation no 3. below).
4. VLAN groups - This step is relevant only if the *RCTL.VFE* bit is set, otherwise it is skipped. Packets should be sent only to VMs that belong to the packet's VLAN group.
   a. Tagged packets: enable only pools in the packet's VLAN group as defined by the VLAN filters - *VLVF[n].VLAN_id* and their pool list - *VLVF[n].POOLSEL[7:0]*.
   b. Untagged packets: enable only pools with their VMOLR.AUPE bit set
   c. If there is no match, the pool list should be empty.
5. Default pool- If the packet is a unicast packet and no pool was chosen and the *VT_CTL.Dis_Def_Pool* bit is not set, then set the default pool bit in the pool list (from VT_CTL.Def_PL).
6. Broadcast or Multicast - If the packet is a Multicast or Broadcast packet and was not forwarded in step 1 and 2, set the default pool bit in the pool list (from VT_CTL.Def_PL).
7. Length Limit: If the packet is longer than a legal Ethernet packet, remove from the pool list all the pools for which the *VMOLR.LPE* bit is not set or for which the packet length is larger than the value in the *VMOLR.RLPML* field.

The above process can be logically described by the following scheme:

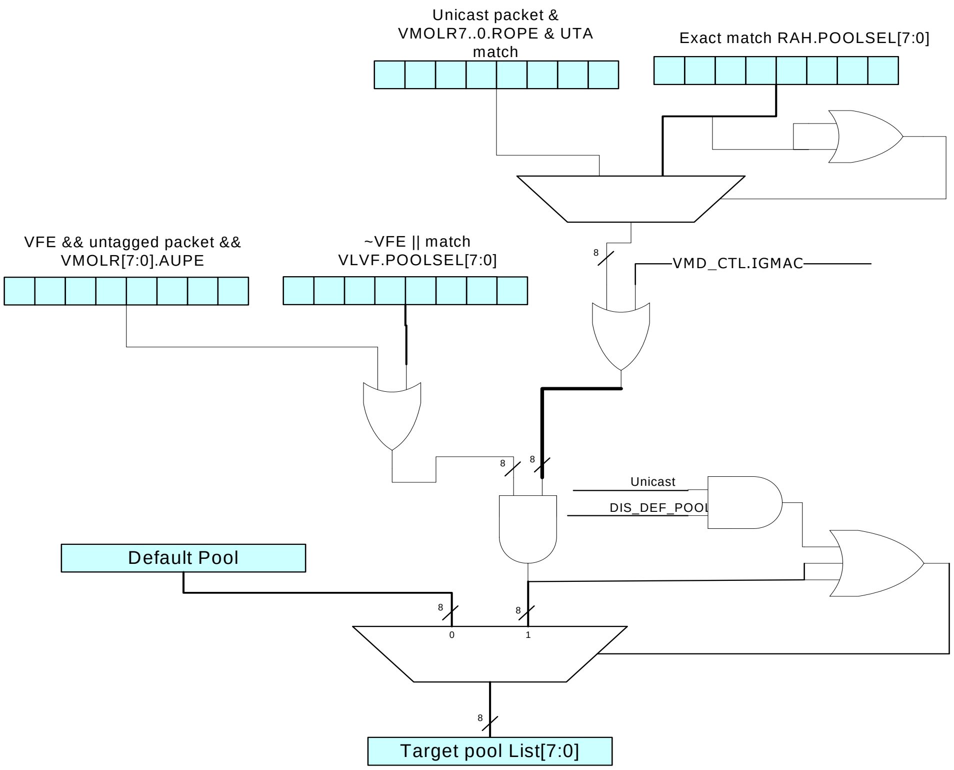


Figure 7-23. Pool List Selection - Replication Disabled

The following limitations applies when replication is disabled:

1. It is the software responsibility to not set more than one bit in the bitmaps of the exact filters. Note that multiple bits might be set in an RAH register as long as it is guaranteed that the packet is sent to only one queue by other means (VLAN)
2. The software must not set per-VM promiscuous bits (multicast or broadcast) in the *VMOLR* register.
3. The software must not set the *ROPE* bit in more than one *VMOLR* register.
4. If *VT_CTL.IGMAC* bit is set, the software must not set the *VMOLR.AUPE* in more than one VMOLR register and must not set more than one bit in each of the *VLVF.POOLSEL* bitmaps.
5. The software must not activate mirroring.

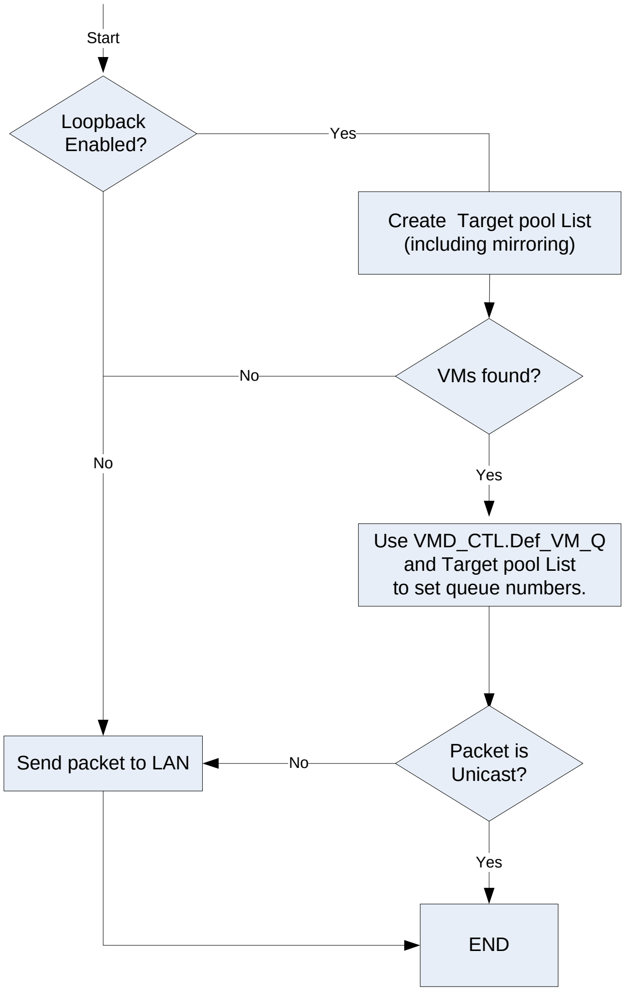

Start
Loopback Enabled?
Yes
Create Target pool List (including mirroring)
VMs found?
No
No
Yes
Use VMD_CTL.Def_VM_Q and Target pool List to set queue numbers.
Send packet to LAN
No
Packet is Unicast?
Yes
END

Figure 7-24. Tx Filtering

7.8.2.6 Mirroring Support

The 82580 supports 4 mirroring rules. Each rule can be of one of 3 types. Only Egress mirroring is supported and not ingress. For example, the mirroring is done on the receive path and mirrored packets reflect all the changes that occur to the received packet (e.g. Vlan stripping). Mirroring is supported only to virtual ports and not to the uplink (i.e. a mirrored packet can not be sent back to the Network).

Mirroring should be activated only when one of the VMDq queueing modes is used.

The following types of rules are supported:

1. Virtual port mirroring - reflects all the packets sent to a set of given VMs.
2. Uplink port mirroring - reflects all the traffic received from the network.

3. VLAN mirroring - reflects all the traffic received in a set of given VLANs.

All the modes can be accumulated into a single rule.

This new mirroring mode is controlled by a set of rule control registers:

- *VMRCTL* - controls the rules to be applied and the destination port.
- *VMRVLAN* - controls the VLAN ports as listed in the *VLVF* table taking part in the VLAN mirror rule.
- *VMRVM* - controls the VMs ports taking part of the Virtual port mirror rule.

Mirroring is supported only when replication is enabled. The exact flow of mirroring is described in step 10. in Section 7.8.2.5.1.

7.8.2.7 VMDq Offload support

In case of packets directed to one VM only, the offloads are determined by this specific VM setting. However, the 82580 can not apply different offloads (VLAN & CRC strip + decision of size of header for split/replication offload) to different replication of the same packet. The following sections describes the rules used to decide which offloads to apply in case of replicated packets.

If replication is disabled, the offloads are determined by the unique destination of the packet.

Note: In a virtualization environment (*MRQC.Multiple Receive Queues Enable* = 011b), the global VLAN strip and CRC strip bits (*CTRL.VME* & *RCTL.SECRC*) are ignored and the VM specific bits in *VMOLR* & *RPLOLR* are used instead.

7.8.2.7.1 Replication by Exact MAC Address

As mentioned above, the same MAC address can be assigned to more than one VM. This is used for the following cases:

- Multicast address - In this case, the different VMs might be part of the same VLAN. The offloads applied to packets matching this address are defined in the replicated packets offloads register (RPLOLR register).
- Unicast - Same MAC different VLAN - In this case, each VM should belong to different VLAN(s). The applied offloads is according to the pool selected by the MAC/VLAN pair. One exception is the VLAN strip decision which is done according to the first pool parameters. This means that all the pools sharing a MAC address should use a common VLAN strip policy.

7.8.2.7.2 Replication by Promiscuous Modes

A packet might be replicated to multiple VMs because part of the VMs are set to receive all multicast or broadcast packets or because of a packet matching one of the hash tables (UTA or MTA).

The offloads applied to packet are defined in the replicated packets offloads registers (RPLOLR register).

In case of unicast packet, the offloads are applied according to the first pool selected to receive the packet.

7.8.2.7.3 Replication by Mirroring

Offloads of mirrored packets are determined according to the original pool.

7.8.2.7.4 VLAN Only Filtering

If *VT_CTL.IGMAC* bit is set, the pool is defined according to the VLAN only. In this mode, only a uniform VLAN strip policy is supported. This means that the *VMOLR.STRVLAN* bit should be set to the same value for all VMs.

7.8.2.7.5 Small Packets Padding

In Virtualized systems, the driver receiving the packet in the VM might not be aware of all the hardware offloads applied to the packet. Thus, in case of stripping actions by the hardware (VLAN strip), it might receive packets which are smaller than a legal packet. The 82580 provides an option to pad small packets in such cases so that all packets have a legal size. This option can be enabled only if the CRC is stripped. In these cases, all packets are padded to 60 bytes (legal packet - 4 bytes CRC). The padding is done with zero data. This function is enabled via the *RCTL.PSP* bit.

7.8.2.8 Security Features

The 82580 allows some security checks on the inbound and outbound traffic of the switch.

7.8.2.8.1 Inbound Security

Each incoming packet from the LAN is filtered according to the VLAN tag so that packets from one VLAN can not be received by VMs that are not members of that VLAN.

7.8.2.8.2 Interrupt on Misbehavior of VM (Malicious Driver Detection).

The hardware can be programmed to take some action as a result of some misbehavior of a VM. These actions might hint to the fact that some VM is malicious and the VMM should remedy the situation. In order to inform the VMM of this fact, an interrupt bit exists in the ICR register (*ICR.DOUTSYNC* bit) to indicate the occurrence of such behavior. The LVMMC register contains information on which queue (*LVMMC.Last_Q*) and port (*LVMMC.Mal_PF*) the malicious behavior was detected. The *LVMMC* register is clear by read.

Malicious driver behavior detection is enabled by setting the *DTXCTL.MDP_EN* bit to 1. On detection of a malicious driver event the 82580 stops activity of the offending queue, asserts relevant bit in the *MDFB.Block Queue* field and generates an interrupt by asserting the *ICR.DOUTSYNC* bit. Cause of Malicious driver activation is reported in the *LVMMC* register. To re-activate offending queue driver should initiate a software reset (*CTRL.RST*) and re-initialize all queues on the port.

7.8.2.8.3 Storm Control

As there is no separate path for multicast & broadcast packets, too much replicated packets might cause congestions in the data path. In order to avoid such scenarios, broadcast and multicast storm control rate limiters are added. The rate controllers define windows and the maximal allowed number of multicast or broadcast bytes/packets per window. Once the threshold is crossed different types of policies can be applied.

7.8.2.8.3.1 Assumptions

- Only one interval size and interval counter is used for both broadcast & multicast storm control mechanisms.
- The threshold and actions for each mechanism are separate.
- The traffic used to calculate the broadcast & multicast rate is all the traffic with a local destination.
- The storm control does not block traffic to the network.
- The basic unit of traffic counted is 64 bytes of data.

7.8.2.8.3.2 Storm Control Functionality

The time interval over which Broadcast Storm control is performed is controlled by three factors.

- SCBI register
- Port speed.
- The value in SCCRL.INTERVAL

The first two factors determine the Unit time interval as described in Table 7-60. The interval is automatically chosen internal to hardware based on port speed. The third factor (*Interval* field) determines how many of such unit intervals are considered for one Storm Control Interval.

Table 7-60. Storm Control Interval by Speed

Port Speed	MIN Time Interval	MAX Time Interval
1 Gb/s	100 µs	100 ms
100 Mb/s	1 ms	1 s
10 Mb/s	10 ms	10 s

The number of 64 bytes chunk of Broadcast or Multicast packets that are allowed in a given interval is determined by setting the BSCTRH or MSCTRH register respectively.

The 82580 supports two modes of reactions to storm event:

1. Block all Multicast or Broadcast packets from the moment the threshold is crossed until the end of the interval. The block is removed at the end of the interval until the threshold is crossed again. This mode is set by asserting SCCRL.MDICW (for multicast) or SCCRL.BDICW (for broadcasts). This mode is used as a rate limiter.
2. Block all Multicast or Broadcast packets from the moment the threshold is crossed until a full interval without threshold crossing is registered. The block is removed at the end of the interval until the threshold is crossed again. This mode is set by asserting SCCRL.MDICW and SCCRL.MDIPW (for multicast) or SCCRL.BDICW and SCCRL.BDIPW (for broadcasts). This mode is used for storm blocking.

The 82580 can be programmed to add all packets for which a queue was not found for storm control calculation. For example, packets that passed the 1st stage of L2 filtering but didn't pass the 2nd stage of pooling, or where sent to the default pool, as broadcast packets. This mode is activated by setting the *SCCRL.BIDU* field.

Any change in the storm control state (block or pass of multicast or broadcast packets) is indicated to the software via the ICR.SCE interrupt cause. The current state is reflected in the SCSTS register.

For diagnostic purpose only, the storm control timer and counters can be read via the SCTC, MSCCNT & BSCCNT registers.

7.8.2.9 External Switch Loopback Support

One of the long term solutions for the switching issue is a mode where an external switch would do the loopback of VM to VM traffic and the NIC is responsible for the replication of multicast packets only. In order to support this mode the received packets SA should be compared to the exact MAC addresses to check if the packet originated from a local source, so that the packet is not forwarded to the VM originator. This mode is enabled by the *VT_CTL.FLP* bit.

7.8.2.10 Switch Control

The VMM/IOVM driver has some control of the switch logic. The following registers are available to the VMM/IOVM for this purpose:

VLVF: VLAN queuing table: A set of 32 VLAN entries with an associated per VM bit map allowing allocation of each VM to each of the 32 VLAN tags.

VT_CTL: VT Control register - contains the following fields:

- Replication enable - allows replication of multicast & broadcast packets. If this bit is cleared, Rx multicast & broadcast packets are sent to the default VM.
- Default pool - defines where to send packets that passed L2 filtering but didn’t pass any of the queueing mechanisms.
- Default pool disable- defines whether to drop packets that passed L2 filtering but didn’t pass any of the queueing mechanisms.

VMOLR/RPMOLR: Defines the offloads and pool selection options for each VM and for replicated packets.

In addition the storm control mechanism is programmed as described in Section 7.8.2.8.3.2.

7.8.3 Virtualization of the Hardware

This section describes additional features used in VMDq mode.

7.8.3.1 Per Pool Statistics

Part of the statistics are by definition shared and can not be allocated to a specific VM. For example, CRC error count can not be allocated to a specific VM, as the destination of such a packet is not known if the CRC is wrong.

All the non specific statistics is handled by the VMM in the same way as it is done in non virtualized systems. A VM might require a statistic from the VMM but might not access it directly.

The conceptual model used to gather statistics in a virtualization context is that each queue pool is considered as a virtual link and the Ethernet link is considered as the uplink of the switch. Thus any packet sent by a VM is counted in the Tx statistics, even if it was dropped by the MAC from some reason. In the same way, a replicated packet is counted in each of the VMs receiving it.

The following statistics are be provided per VM:

1. Good Packet received count (*VFGPRC*).
2. Good Packet transmitted count (*VFGPTC*).
3. Good octets received count (*VFGORC*).
4. Good octets transmitted count (*VFGOTC*).
5. Multicast Packets Received Count (*VFMPRC*).

Note: All the per VM statistics are read only (RO) and wrap around after reaching their maximal value.

7.9 Time SYNC (IEEE1588 and IEEE 802.1AS)

7.9.1 Overview

IEEE 1588 addresses the clock synchronization requirements of measurement and control systems. The protocol supports system-wide synchronization accuracy in the sub-microsecond range with minimal network and local clock computing resources. The protocol is spatially localized and allows simple systems to be installed and operated without requiring the administrative attention of users.

The IEEE802.1AS standard specifies the protocol used to ensure that synchronization requirements are met for time sensitive applications, such as audio and video, across Bridged and Virtual Bridged Local Area Networks consisting of LAN media where the transmission delays are fixed and symmetrical; for example, IEEE 802.3 full duplex links. This includes the maintenance of synchronized time during normal operation and following addition, removal, or failure of network components and network reconfiguration. It specifies the use of IEEE 1588 specifications where applicable.

Activation of the 82580 Time Sync mechanism is possible in full duplex mode only. No limitations on wire speed exist, although wire speed might affect the accuracy. Time Sync protocol is tolerant of dropping packets as well as missing timestamps.

7.9.2 Flow and Hardware/Software Responsibilities

The operation of a PTP (Precision Time Protocol) enabled network is divided into two stages, initialization and time synchronization.

At the initialization stage every master enabled node starts by sending Sync packets that include the clock parameters of its clock. Upon reception of a Sync packet a node compares the received clock parameters to its own. If the received clock parameters of a peer are better, the node moves to Slave state and stops sending Sync packets. When in slave state the node continuously compares the incoming packet clock parameters to its currently chosen master. If the new clock parameters are better then the current master selection, it changes master clock source. Eventually the best master clock source is chosen. Every node has a defined Sync packet time-out interval. If no Sync packet is received from its chosen master clock source during the interval it moves back to master state and starts sending Sync packets until a new Best Master Clock (BMC) is chosen.

The time synchronization stage is different for master and slave nodes. If a node is in master state it should periodically send a Sync packet which is time stamped by hardware on the transmit path (as close as possible to the PHY). After the Sync packet a Follow_up packet is sent which includes the value of the timestamp kept from the Sync packet. In addition the master should timestamp Delay_Req packets on its RX path and return to the slave that sent it the timestamp value using a Delay_Response packet. A node in Slave state should timestamp every incoming Sync packet that is received from its selected master, software uses this value for time offset calculation. In addition it should periodically send Delay_Req packets in order to calculate the path delay from its master. Every sent Delay_Req packet sent by the slave is time stamped and kept. Using the value received from the master Delay_Response packet the slave can now calculate the path delay from the master to the slave. The synchronization protocol flow and the offset calculation are described in the following figure.

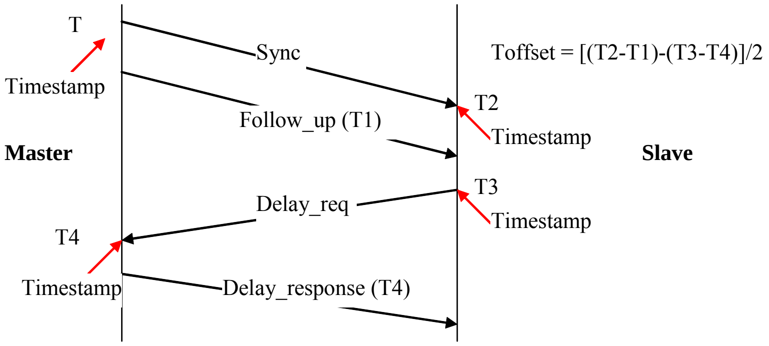


Figure 7-25. Sync Flow and Offset Calculation

The hardware’s responsibilities are:

1. Identify the packets that require time stamping.
2. Time stamp the packets on both receive and transmit paths.
3. Store the time stamp value for software.
4. Keep the system time in hardware and give a time adjustment service to the software.
5. Maintain auxiliary features related to the system time.

The software’s responsibilities are:

1. Best Master Clock protocol execution, which determines which clock is the highest quality clock within the network. Result of protocol sets the node state (master or slave). If node is slave, software also selects the master clock.
2. Generate PTP (Precision Time Protocol) packets, consume PTP packets.
3. Calculate the time offset and adjust the system time using the hardware mechanism.
4. Enable configuration and usage of the auxiliary features.

Table 7-61. Chronological Order of Events for Sync and Path Delay

Action	Responsibility	Node Role
Generate a Sync packet with timestamp notification in descriptor	Software	Master
Timestamp the packet and store the value in registers (T1)	Hardware	Master
Timestamp incoming Sync packet, store the value in register and store the sourceID and sequenceID in registers (T2)	Hardware	Slave
Read the timestamp from register, prepare a Follow_Up packet and send	Software	Master
Once Follow_Up packet is received, load T2 from registers and T1 from Follow_up packet	Software	Slave
Generate a Delay_Req packet with timestamp notification in descriptor	Software	Slave
Timestamp the packet and store the value in registers (T3)	Hardware	Slave
Timestamp incoming Delay_Req packet, store the value in register and store the sourceID and sequenceID in registers (T4)	Hardware	Master
Read the timestamp from register and send back to Slave using a Delay_Response packet	Software	Master
Once Delay_Response packet is received, calculate offset using T1, T2, T3 and T4 values	Software	Slave

7.9.2.1 TimeSync Indications in Receive and Transmit Packet Descriptors

Certain indications are transferred between software and hardware regarding PTP packets.

On the transmit path the software should set the *1588* bit in the transmit packet descriptor (MAC field bit 1). To indicate that the transmit packet time stamp should be taken and placed in the *TXSTMPH* and *TXSTMPL* time stamp registers.

On the receive path the hardware transfers three indications to software in the receive descriptor:

1. An indication in *RDESC.Packet Type* that this packet is a PTP packet (no matter if timestamp is sampled or not). This indication is used also by PTP packets required for protocol management.

Note: This indication is only relevant for L2 type packets (the PTP packet is identified according to its Ethertype). PTP packets have the *L2Type* bit in the *Packet Type* field set (bit 11) and the Etype matches the filter number set by the software to filter PTP packets. UDP type PTP packets don't require such an indication since the port number (319 for event and 320 for all other PTP packets) directs the packets toward the time sync application.

2. A second indication in the *RDESC.STATUS.TS* bit to indicate to the software that time stamp was taken for this packet and placed in the *RXSTMPH* and *RXSTMPL* time stamp registers. Software needs to access the time stamp registers to get the time stamp values.
3. A third indication in the *RDESC.STATUS.TSIP* bit to indicate that a time stamp was taken for this packet and placed at the start of the receive buffer (For further information see Section 7.1.10).

7.9.3 Hardware Time Sync Elements

All time sync hardware elements are reset to their initial values as defined in the registers section upon MAC reset.

7.9.3.1 System Time Structure and Mode of Operation

The time sync logic contains the SYSTIM counter to maintain the system time value. This is a 72 bit counter that is built of the *SYSTIMR*, ***SYSTIML*** and ***SYSTIMH*** registers. Operation of the counter is enabled by clearing the *TSAUXC.Disable systime* bit. When in Master state the ***SYSTIMH***, ***SYSTIML* and *SYSTIMR*** registers should be set once by the software according to general system requirements in the following manner:

1. Disable *SYSTIM* timer operation by setting the *TSAUXC.Disable systime* bit.
2. Program the ***SYSTIMH***, ***SYSTIML* and *SYSTIMR* registers.**
3. Enable *SYSTIM* timer operation by clearing the *TSAUXC.Disable systime* bit.

When in slave state software should update the system time on every sync event as described in Section 7.9.3.3. Setting the system time is done by direct write to the ***SYSTIMH* register as described above, enabling SYSTIM operation by clearing the** *TSAUXC.Disable systime* bit and fine tuning the setting of the **SYSTIM register,** using the adjustment mechanism described in Section 7.9.3.3.

Read access to the ***SYSTIMH***, ***SYSTIML* and *SYSTIMR*** registers should be executed in the following order:

1. Software reads register ***SYSTIMR***.

Note: At this stage the hardware latches the value of ***SYSTIMH* and *SYSTIML* registers**.

2. Software reads register ***SYSTIML*.**

3. Software reads register ***SYSTIMH.***

Note: The latched ***SYSTIMH* and *SYSTIML*** values (from last read of ***SYSTIMR* register**) should be returned by hardware when reading the ***SYSTIML*** and ***SYSTIMH* registers.**

The SYSTIM timer value in the ***SYSTIMH***, ***SYSTIML*** and ***SYSTIMR*** registers, is updated periodically each 8 nS clock cycle according to the following formula:

New SYSTIM = Old SySTIM + 8 nS +/- ***TIMINCA****.Incvalue* * 2^{-32} nS

Where subtraction or addition of the *TIMINCA.Incvalue* value is defined according to the *TIMINCA.ISGN* value (0 - Add, 1 - Subtract). For the ***TIMINCA*** register description refer to section 8.17.12.

7.9.3.2 Time Stamp Mechanism

The time stamp logic is located on transmit and receive paths at a location as close as possible to the PHY, to reduce delay uncertainties originating from implementation differences. The time stamp logic operation is slightly different on transmit and on receive paths.

When the *TSAUXC.Disable systime* bit is cleared the transmit logic decides to timestamp a packet if the transmit timestamp is enabled (*TSYNCTXCTL.EN* = 1) and the time stamp bit in the packet descriptor (*TDESD.MAC.1588* = 1) is set. On the transmit side only the time is captured in the *TXSTMPL* and *TXSTMPH* registers.

The receive logic parses the received frame and timestamps the receive packet according to the conditions defined in the following fields:

1. *TSYNCRXCTL.Type* field that defines type of packets to be sampled.
2. *TSYNCRXCFG.CTRLT* field that defines message type criteria for timestamping V1 type packets when *TSYNCRXCTL.Type* register field equals 001b.
3. *TSYNCRXCFG.MSGT* field that defines message type criteria for timestamping V2 type packets when *TSYNCRXCTL.Type* register field equals 000b or 010b.

When the *TSAUXC.Disable systime* bit is cleared and above conditions to timestamp a receive packet are met:

1. The timestamp is latched in the *RXSTMPL and RXSTMPH* registers.
2. The packet's sourceId and sequenceId fields are latched in the *RXSATRL* and *RXSATRH* timestamp registers.
3. When the *SRRCTL[n].Timestamp* bit is set to 1, packets received to the queue will have a timestamp value added to the beginning of the receive buffer (See Section 7.1.10 for additional information).

Three indications are placed in the receive descriptor to support TimeSync operation:

1. *RDESC.Packet Type* - Value in this field identifies that this is a PTP packet (this indication is only for L2 packets since on the UDP packets the port number directs the packet to the application).
2. *RDESC.STATUS.TS* - Bit identifies that a time stamp was taken for this packet and latched in *RXSTMPL* and *RXSTMPH* registers and the packet's sourceId and sequenceId are latched in the *RXSATRL* and *RXSATRH* timestamp registers.
3. *RDESC.STATUS.TSIP* - Bit identifies that a time stamp was taken for this packet and placed at the start of the receive buffer (For further information see Section 7.1.10).

For more details please refer to the timestamp registers in Section 8.17. Figure 7-26 defines the exact point where the time value is captured.

On both transmit and receive sides the timestamp values are locked in registers until software reads the TXSTMPH register to unlock Transmit timestamp registers or reads the RXSTMPH register to unlock the receive timestamp registers. As a result, if a new PTP packet that needs to be time stamped arrives before software accesses the timestamp registers, it is not time stamped. In some cases on the receive path a packet that was timestamped might be lost and not reach the host. To avoid a deadlock condition on the time stamp registers the software should keep a watch dog timer to clear locking of the time stamp register. The interval counted by such a timer should be higher then the expected interval between two Sync or Delay_Req packets depends on the node state (Master or Slave).

Note: When *TSYNCRXCTL.Type* value is 100b, the Receive timestamp registers are not locked after a timestamp event.

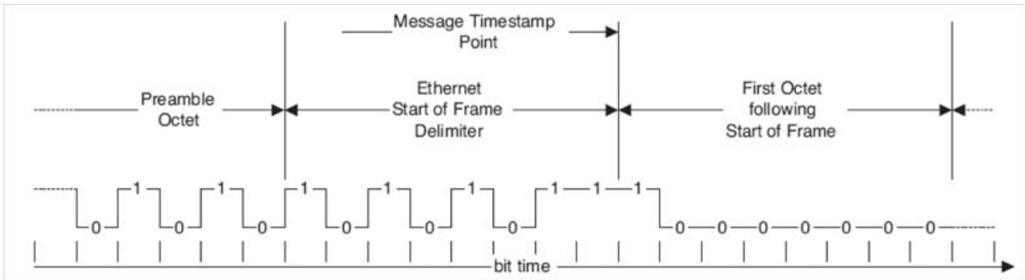

Message Timestamp Point
Preamble Octet
Ethernet Start of Frame Delimiter
First Octet following Start of Frame
1
0
bit time

Figure 7-26. Time Stamp Point

7.9.3.3 Time Adjustment Mode of Operation

A Node in a Time Sync network can be in one of two states Master or Slave. When a Time Sync entity is in the Master state it should synchronize other entities to its System Clock. In this case no time adjustments are needed. When the entity is in slave state it should adjust its system clock by using the data arriving in the Follow_Up and Delay_Response packets and the time stamp values of the Sync and Delay_Req packets. When all the values are available the software in the slave entity can calculate its offset in the following manner:

$$Toffset = [(T2-T1) - (T3-T4)]/2$$

T1 - Timing data in Follow_Up packet
T2 - Sync Time Stamp
T3 - Delay_Req Time Stamp
T4 - Timing data in Delay_Response packet

To increase the *system time* value located in the *SYSTIMH* and *SYSTIML* registers by the calculated *Toffset* value, software should write the calculated *Toffset* value to the ***TIMADJH*** and ***TIMADJL*** registers in the following order:

1. Write the low portion of *Toffset* value to the ***TIMADJL***.*TADJL* field.
2. Write the high portion of the *Toffset* value to the ***TIMADJH.TADJH* field** with the correct sign in the ***TIMADJH.****Sign* bit, to indicate if the *Toffset* value should be added or subtracted.

A write to the ***TIMADJH*** register causes the *Toffset* value in the ***TIMADJH*** and ***TIMADJL*** registers to be added or subtracted (depending on the sign bit) to the *system time* registers (*SYSTIMH* and *SYSTIML*), resulting in a new *system time* that equals *previous system time* + *Toffset*.

7.9.4 Time Sync Related Auxiliary Elements

The time sync logic implements three types of auxiliary elements using the precise system timer (*SYSTIML* and *SYSTIMH*).

7.9.4.1 Target Time

The two target time registers *TRGTTIML/H0* and *TRGTTIML/H1* enable generating a time triggered event to external hardware using one of the SDP pins according to the setup defined in the *TSSDP* and *TSAUXC* registers (See Section 8.17.15 and Section 8.17.27). Each target time register is structured the same as the system time register. If the value of the system time is equal or has passed the value written to one of the target time registers, a change in level or a pulse is generated on the programmed SDP outputs.

7.9.4.1.1 SYSTIM Synchronized Level Change Generation on SDP Pins

To generate a level change on one of the SDP pins when System Time (*SYSTIM*) reaches a pre-defined value, driver should:

1. Select SDPx pin functionality using the appropriate *TSSDP.TS_SDPx_SEL* field (where x is 0, 1, 2 or 3).
   - — Program 00b to the *TSSDP.TS_SDPx_SEL* field if level change should occur when *SYSTIM* equals *TRGTTIML/H0.*
   - — Program 01b to the *TSSDP.TS_SDPx_SEL* field if level change should occur when *SYSTIM* equals *TRGTTIML/H1.*
2. Define selected SDPx pin as output, by setting the appropriate *SDPx_IODIR* bit (where x is 0,1,2 or 3) in the *CTRL* or *CTRL_EXT* registers.
3. To define that level change is generated on selected SDP pin when *SYSTIM* is equal or greater than Target Time, program *TSAUXC.PLSGx* bit (where x is 0 or 1) to 0.
4. Program *TRGTTIML/Hx* (where x is 0 or 1) to define *SYSTIM* time where level change should occur.
5. Enable level change or pulse generation by setting the *TSAUXC.EN_TTx* bit (where x is 0 or 1).

Each target time register has an enable bit located in the auxiliary control register (*TSAUXC.EN_TTx*). When the SDP level has changed (if *TSAUXC.PLSGx* = 0) on the selected SDP pin, the enable bit is cleared and needs to be set again by software to get another target time event.

7.9.4.1.2 SYSTIM Synchronized Pulse Generation on SDP Pins

To generate a pulse on one of the SDP pins when System Time (*SYSTIM*) reaches a pre-defined value, driver should:

1. Select SDPx pin functionality using the appropriate *TSSDP.TS_SDPx_SEL* field (where x is 0, 1, 2 or 3).
   - — Program 00b to the *TSSDP.TS_SDPx_SEL* field if pulse start should occur when *SYSTIM* equals *TRGTTIML/H0.*
   - — Program 01b to the *TSSDP.TS_SDPx_SEL* field if pulse start should occur when *SYSTIM* equals *TRGTTIML/H1.*
2. Define selected SDPx pin as output, by setting the appropriate *SDPx_IODIR* bit (where x is 0,1,2 or 3) in the *CTRL* or *CTRL_EXT* registers.
3. Program *TSAUXC.PLSGx* bit (where x is 0 or 1) to 1 to define that pulse is generated on the selected SDP pin, when Target Time is equal or greater than System Time (*SYSTIM*).
4. Program the *TSAUXC.PLSNegx* bit to define if generated pulse is positive or negative.

5. Program *TRGTTIML/H0* and *TRGTTIML/H1* registers to define pulse start time and pulse duration if pulse generation.
   - — Note that for pulse generation both the *TRGTTIML/H0* and *TRGTTIML/H1* Target Time registers are used. Depending on the *TSSDP.TS_SDPx_SEL* field value, one Target Time register is used to define start of pulse and the other is used to define end of pulse.
6. Enable pulse generation by setting both the *TSAUXC.EN_TT0 and TSAUXC.EN_TT1* bits to 1.

Each target time register has an enable bit located in the auxiliary control register (*TSAUXC.EN_TTx*). When a pulse is generated on the selected SDP pin, the enable bits are cleared and need to be set again by software to get another target time event.

7.9.4.1.3 Start of Clock Generation on SDP Pins Synchronized to SYSTIM

the 82580 supports driving a configurable Clock on the SDP pins. The output clocks generated are synchronized to the global System (*SYSTIM*) clock. The Target Time registers (*TRGTTIML/H0* or *TRGTTIML/H1*) can be used to trigger toggle of the configurable clock output on a certain system time.

Setting the appropriate *TSAUXC.STx* (where x is 0 or 1) bit to 1 enables start of clock generation only after Target Time defined in the *TRGTTIML/Hx* registers is reached (further information can be found in Section 7.9.4.2).

7.9.4.2 Configurable Frequency Clock

This feature enables to generate up to 2 programmable clocks on the appropriate SDP pins by configuring the SDP pins using the TSSDP register and by programming appropriate values to the Frequency out Control registers (*FREQOUT0* and *FREQOUT1*). The output clocks are synchronized to the global System (*SYSTIM*) clock and are affected by System time corrections programmed in the *TIMINCA* register and the *TIMADJL/H* registers.

When clock generation is enabled, the error correction programmed in the *TIMADJL/H* registers is compensated from the clock output gradually, at a rate of 1 ns per 8 nS internal clock cycle. The gradual compensation is done to avoid large duty cycle variations in the output clock.

To generate either Clock 0 or Clock 1 on one of the SDP pins, the following steps should be taken:

1. Program the *CHCT* field in the relevant *FREQOUT0/1* register to define clock half cycle time (See Section 8.17.20 and Section 8.17.21 for additional information).
2. Define SDPx pin functionality to drive clock by programming the appropriate *TSSDP.TS_SDPx_SEL* field and setting the *TSSDP.TS_SDPx_EN* bit to 1 (where x is 0,1,2 or 3).
   - — Program 10b to the *TSSDP.TS_SDPx_SEL* field if the *FREQOUT0* register is used to define clock frequency.
   - — Program 11b to the *TSSDP.TS_SDPx_SEL* field if the *FREQOUT1* register is used to define clock frequency.
3. Define selected SDPx pin as output, by setting the appropriate *SDPx_IODIR* bit (where x is 0,1,2 or 3) in the *CTRL* or *CTRL_EXT* registers.
4. If clock start needs to be aligned to the system time (*SYSTIM*), program start of clock toggle in the appropriate Target Time (*TRGTTIML0/1* and *TRGTTIMH0/1*) registers and set the relevant *TSAUXC.ST0/1* field to 1.
5. To start clock operation, set the relevant *TSAUXC.EN_CLK0/1* bit to 1.

Clock out drives initially a logical 0. Clock value toggles each time a System Time duration of *FREQOUT0*/1 is reached or passed.

Note: Clock output mechanism should be activated only after SYSTIMH/L timer is aligned to global system clock and SYSTIM timer error correction entered using the *TIMADJL/H* registers is below 64 µs.

7.9.4.3 Time Stamp Events

Upon a change in the input level of one of the SDP pins that was configured to detect Time stamp events using the TSSDP register, a time stamp of the system time is captured into one of the two auxiliary time stamp registers (*AUXSTMPL/H0* or *AUXSTMPL/H1*).

For example to define timestamping of events in the *AUXSTMPL*0 and *AUXSTMPH0* registers, Software should:

1. Set the *TSSDP.AUX0_SDP_SEL* field to select the SDP pin that detects the level change and set the *TSSDP.AUX0_TS_SDP_EN* bit to 1.
2. Set the *TSAUXC.EN_TS0* bit to 1 to enable timestamping.

7.9.5 PTP Packet Structure

The time sync implementation supports both the 1588 V1 and V2 PTP frame formats. The V1 structure can come only as UDP payload over IPv4 while the V2 can come over L2 with its Ethertype or as a UDP payload over IPv4 or IPv6. The 802.1AS uses only the layer 2 V2 format.

Table 7-62. V1 and V2 PTP Message Structure

Offset in Bytes	V1 Fields	V2 Fields	
Bits	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	
0	***versionPTP***	transport Specific[1]	***messageType***
1		Reserved	***versionPTP***
2	version Network	message Length	
3			
4	Subdomain	domain Number	
5		Reserved	
6		flags	
7			
8		correctionField	
9			
10			
11			
12			
13			
14			
15			
16		reserved	
17			
18			
19			

Table 7-62. V1 and V2 PTP Message Structure (Continued)

Offset in Bytes	V1 Fields	V2 Fields
Bits	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
20	message Type	Source Port Identity
21	Source communication technology	
22	Sourceuuid	
23		
24		
25		
26		
27		
28	source port id	
29		
30	***sequenceId***	***sequenceId***
31		
32	***control***	control
33	reserved	Log Message Interval
34	flags	N/A
35		

[1] Should be all zero.

Note: Only the fields with the bold italic format colored red are of interest to the hardware.

Table 7-63. PTP Message Over Layer 2

Ethernet (L2)	VLAN (Optional)	PTP Ethertype	PTP message

Table 7-64. PTP Message Over Layer 4

Ethernet (L2)	IP (L3)	UDP	PTP message

When a PTP packet is recognized (by Ethertype or UDP port address) on the receive side then if the version is V1 then the *Control* field at offset 32 should be compared to the *TSYNCRXCFG.CTRLT* message field (see Section 8.17.26) otherwise the byte at offset zero should be used for comparison to the *TSYNCRXCFG*.MSGT field. The rest of the required fields are at the same location and size for both V1 and V2.

Table 7-65. Message Decoding for V1 (Control Field at Offset 32)

Enumeration	Value
PTP_SYNC_MESSAGE	0
PTP_DELAY_REQ_MESSAGE	1
PTP_FOLLOWUP_MESSAGE	2

Table 7-65. Message Decoding for V1 (Control Field at Offset 32)

Enumeration	Value
PTP_DELAY_RESP_MESSAGE	3
PTP_MANAGEMENT_MESSAGE	4
reserved	5–255

Table 7-66. Message Decoding for V2 (MessageId Field at Offset 0)

MessageId	Message Type	Value (hex)
PTP_SYNC_MESSAGE	Event	0
PTP_DELAY_REQ_MESSAGE	Event	1
PTP_PATH_DELAY_REQ_MESSAGE	Event	2
PTP_PATH_DELAY_RESP_MESSAGE	Event	3
Unused	Event	4-7
PTP_FOLLOWUP_MESSAGE	General	8
PTP_DELAY_RESP_MESSAGE	General	9
PTP_PATH_DELAY_FOLLOWUP_MESSAGE	General	A
PTP_ANNOUNCE_MESSAGE	General	B
PTP_SIGNALLING_MESSAGE	General	C
PTP_MANAGEMENT_MESSAGE	General	D
Unused	General	E-F

If V2 mode is configured in the TSYNCRXCTL.Type field (see Section 8.17.1) then the time stamp should be taken on PTP_PATH_DELAY_REQ_MESSAGE and PTP_PATH_DELAY_RESP_MESSAGE according to the value in the *TSYNCRXCFG*.MSGT message field described in Section 8.17.26.

7.10 Statistic Counters

The 82580 supports different statistic counters as described in Section 8.19. The statistic counters can be used to create statistic reports as required by different standards. The 82580 statistic counters allow support for the following standards:

- IEEE 802.3 clause 30 management – DTE section.
- NDIS 6.0 OID_GEN_STATISTICS.
- RFC 2819 – RMON Ethernet statistics group.
- Linux Kernel (version 2.6) net_device_stats

The following section describes the match between the internal the 82580 statistic counters and the counters requested by the different standards.

7.10.1 IEEE 802.3 clause 30 management

The 82580 supports the Basic and Mandatory Packages defined in clause 30 of the IEEE 802.3 spec. The following table describes the matching between the internal statistics and the counters requested by these packages.

Table 7-67. IEEE 802.3 Mandatory Package Statistics

Mandatory package capability	82580 counter	Notes and limitations
FramesTransmittedOK	GPTC	The 82580 doesn't include flow control packets.
SingleCollisionFrames	SCC	
MultipleCollisionFrames	MCC	
FramesReceivedOK	GPRC	The 82580 doesn't include flow control packets.
FrameCheckSequenceErrors	CRCERRS	
AlignmentErrors	ALGNERRC	

In addition, part of the recommended package is also implemented as described in the following table

Table 7-68. IEEE 802.3 Recommended Package Statistics

Recommended package capability	82580 counter	Notes and limitations
OctetsTransmittedOK	GOTCH/GOTCL	The 82580 counts also the DA/SA/LT/CRC as part of the octets. The 82580 doesn't count Flow control packets.
FramesWithDeferredXmissions	DC	
LateCollisions	LATECOL	
FramesAbortedDueToXSColls	ECOL	
FramesLostDueToIntMACXmitError	HTDMPC	The 82580 counts the excessive collisions in this counter, while 802.3 increments no other counters, while this counter is incremented
CarrierSenseErrors	TNCRS	The 82580 doesn't count cases of CRS de-assertion in the middle of the packet. However, such cases are not expected when the internal PHY is used.
OctetsReceivedOK	TORL+TORH	The 82580 counts also the DA/SA/LT/CRC as part of the octets. Doesn't count Flow control packets.
FramesLostDueToIntMACRcvError	RNBC	
SQETestErrors	N/A	
MACControlFramesTransmitted	N/A	
MACControlFramesReceived	N/A	
UnsupportedOpcodesReceived	FCURC	
PAUSEMACCtrlFramesTransmitted	XONTXC + XOFFTXC	
PAUSEMACCtrlFramesReceived	XONRXC + XOFFRXC	

Part of the optional package is also implemented as described in the following table

Table 7-69. IEEE 802.3 Optional Package Statistics

Optional package capability	82580 counter	Notes
MulticastFramesXmittedOK	MPTC	The 82580 doesn't count FC packets
BroadcastFramesXmittedOK	BPTC	
MulticastFramesReceivedOK	MPRC	The 82580 doesn't count FC packets
BroadcastFramesReceivedOK	BPRC	
InRangeLengthErrors	LENERRS	
OutOfRangeLengthField	N/A	Packets parsed as Ethernet II packets
FrameTooLongErrors	ROC + RJC	

7.10.2 OID_GEN_STATISTICS

The 82580 supports the part of the OID_GEN_STATISTICS as defined by Microsoft* NDIS 6.0 spec. The following table describes the matching between the internal statistics and the counters requested by this structure.

Table 7-70. Microsoft* OID_GEN_STATISTICS

OID entry	82580 counters	Notes
ifInDiscards;	CRCERRS + RLEC + RXERRC + MPC + RNBC + ALGNERRC	
ifInErrors;	CRCERRS + RLEC + RXERRC + ALGNERRC	
ifHCInOctets;	GORCL/GOTCL	
ifHCInUcastPkts;	GPRC - MPRC - BPRC	
ifHCInMulticastPkts;	MPRC	
ifHCInBroadcastPkts;	BPRC	
ifHCOutOctets;	GOTCL/GOTCH	
ifHCOutUcastPkts;	GPTC - MPTC - BPTC	
ifHCOutMulticastPkts;	MPTC	
ifHCOutBroadcastPkts;	BPTC	
ifOutErrors;	ECOL + LATECOL	
ifOutDiscards;	ECOL	
ifHCInUcastOctets;	N/A	
ifHCInMulticastOctets;	N/A	
ifHCInBroadcastOctets;	N/A	
ifHCOutUcastOctets;	N/A	
ifHCOutMulticastOctets;	N/A	
ifHCOutBroadcastOctets;	N/A	

7.10.3 RMON

The 82580 supports the part of the RMON Ethernet statistics group as defined by IETF RFC 2819. The following table describes the matching between the internal statistics and the counters requested by this group.

Table 7-71. RMON Statistics

RMON statistic	82580 counters	Notes
etherStatsDropEvents	MPC + RNBC	
etherStatsOctets	TOTL + TOTH	
etherStatsPkts	TPR	
etherStatsBroadcastPkts	BPRC	
etherStatsMulticastPkts	MPRC	The 82580 don’t count FC packets
etherStatsCRCAlignErrors	CRCERRS + ALGNERRC	
etherStatsUndersizePkts	RUC	
etherStatsOversizePkts	ROC	
etherStatsFragments	RFC	Should count bad aligned fragments as well
etherStatsJabbers	RJC	Should count bad aligned jabbers as well
etherStatsCollisions	COLC	

Table 7-71. RMON Statistics (Continued)

RMON statistic	82580 counters	Notes
etherStatsPkts64Octets	PRC64	RMON counts bad packets as well
etherStatsPkts65to127Octets	PRC127	RMON counts bad packets as well
etherStatsPkts128to255Octets	PRC255	RMON counts bad packets as well
etherStatsPkts256to511Octets	PRC511	RMON counts bad packets as well
etherStatsPkts512to1023Octets	PRC1023	RMON counts bad packets as well
etherStatsPkts1024to1518Octets	PRC1522	RMON counts bad packets as well

7.10.4 Linux net_device_stats

The 82580 supports part of the net_device_stats as defined by Linux Kernel version 2.6 (defined in <linux/netdevice.h>). The following table describes the matching between the internal statistics and the counters requested by this structure.

Table 7-72. Linux net_device_stats

net_device_stats field	82580 counters	Notes
rx_packets	GPRC	The 82580 doesn’t count flow controls - can be accounted for by using the XONRXC and XOFFRXC counters
tx_packets	GPTC	The 82580 doesn’t count flow controls - can be accounted for by using the XONTXC and XOFFTXC counters
rx_bytes	GORCL + GORCH	
tx_bytes	GOTCL + GOTCH	
rx_errors	CRCERRS + RLEC + RXERRC + ALGNERRC	
tx_errors	ECOL + LATECOL	
rx_dropped	N/A	
tx_dropped	N/A	
multicast	MPTC	
collisions	COLC	
rx_length_errors	RLEC	
rx_over_errors	N/A	
rx_crc_errors	CRCERRS	
rx_frame_errors	ALGNERRC	
rx_fifo_errors	HRMPC	
rx_missed_errors	MPC	
tx_aborted_errors	ECOL	
tx_carrier_errors	N/A	
tx_fifo_errors	N/A	
tx_heartbeat_errors	N/A	
tx_window_errors	LATECOL	
rx_compressed	N/A	
tx_compressed	N/A	

7.10.5 Statistics hierarchy.

The following diagrams describes the relations between the packet flow and the different statistic counters.

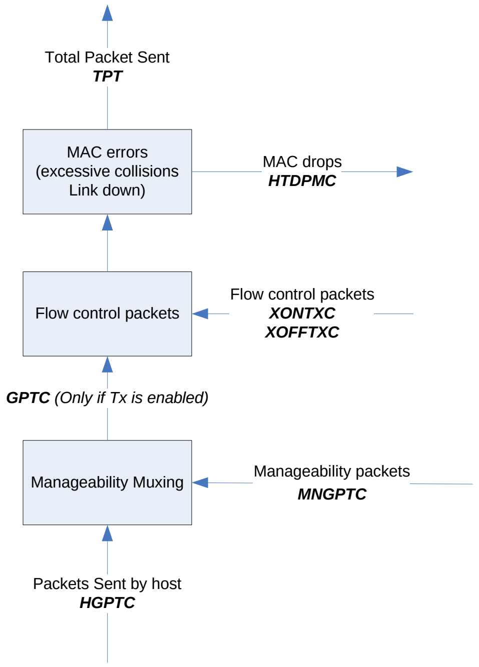


Figure 7-27. Transmit Flow Statistics

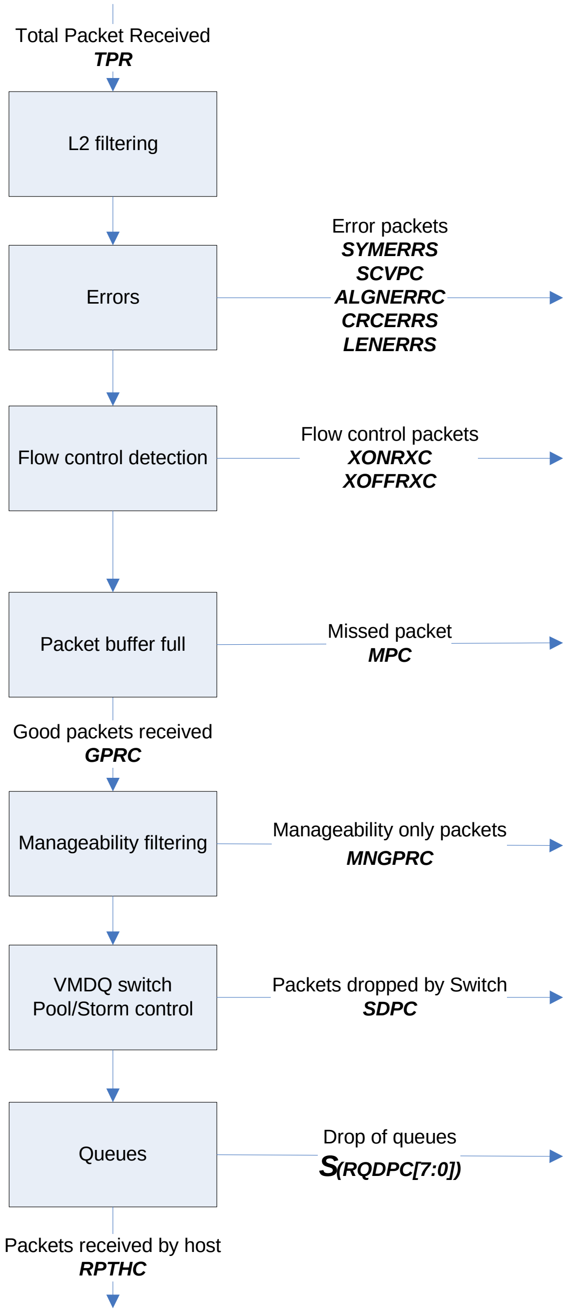


Figure 7-28. Receive Flow Statistics

§ §

8.0 Programming Interface

8.1 Introduction

This chapter details the programmer visible state inside the 82580. In some cases, it describes hardware structures invisible to software in order to clarify a concept. The 82580's address space is mapped into four regions with PCI Base Address Registers described in Section 9.4.11. These regions are listed in the table below.

Table 8-1. Address Space Regions

Addressable Content	How Mapped	Size of Region
Internal registers, memories and FLASH (“Memory BAR”)	Direct memory-mapped	128K + FLASH Size (1,2)
Flash (optional)	Direct memory-mapped	64K-8M
Expansion ROM (optional)	Direct memory-mapped	64K-8M (2)
Internal registers and memories, Flash (optional)	I/O Window mapped	32 bytes (3)
MSI-X (optional)	Direct memory-mapped	16K

(1) The FLASH size is defined by the BARCTRL register

(2) The FLASH space in the “Memory CSR” and Expansion ROM Base Address map the same FLASH memory. Accessing the “memory BAR” at offset 128K and Expansion ROM at offset 0x0 are mapped to the FLASH device at offset 0x0.

(3) The internal registers and memories can be accessed though I/O space indirectly as explained below.

The internal register/memory space is described in the following sections. The PHY registers are accessed through the MDIO interface.

8.1.1 Memory, I/O Address and Configuration Decoding

8.1.1.1 Memory-Mapped Access to Internal Registers and Memories

The internal registers and memories might be accessed as direct memory-mapped offsets from the base address register (BAR0 or BAR 0/1 see Section 9.4.11). See Section 8.1.3 for the appropriate offset for each specific internal register.

8.1.1.2 Memory-Mapped Access to Flash

The external Flash may be accessed using direct memory-mapped offsets from the Memory base address register (BAR0 in 32bit addressing or BAR0/BAR1 in 64 bit addressing see Section 9.4.11). For accesses, the offset from the Memory BAR minus 128KB corresponds to the physical address within the external Flash device. Memory mapped accesses to the external Flash are enabled when the value of the *Flash Size* field in the *Initialization Control Word 2* EEPROM word is not 000b.

8.1.1.3 Memory-Mapped Access to MSI-X Tables

The MSI-X tables may be accessed as direct memory-mapped offsets from the base address register (BAR3 see Section 9.4.11). See Section 8.1.3 for the appropriate offset for each specific internal MSIX register.

8.1.1.4 Memory-Mapped Access to Expansion ROM

The external Flash might also be accessed as a memory-mapped expansion ROM. Accesses to offsets starting from the Expansion ROM Base address (see Section 9.4.11) reference the Flash provided that access is enabled thorough the *LAN Boot Disable* bit in the *Initialization Control 3* EEPROM word, and if the Expansion ROM Base Address register contains a valid (non-zero) base memory address.

8.1.1.5 I/O-Mapped Access to Internal Registers and Memories

To support pre-boot operation (prior to the allocation of physical memory base addresses), all internal registers and memories can be accessed using I/O operations. I/O accesses are supported only if an I/O Base Address is allocated and mapped (BAR2 see Section 9.4.11), the BAR contains a valid (non-zero value), and I/O address decoding is enabled in the PCIe configuration.

When an I/O BAR is mapped, the I/O address range allocated opens a 32-byte “window” in the system I/O address map. Within this window, two I/O addressable registers are implemented: IOADDR and IODATA. The IOADDR register is used to specify a reference to an internal register or memory, and then the IODATA register is used as a “window” to the register or memory address specified by IOADDR:

Table 8-2. IOADDR and IODATA in I/O Address Space

Offset	Abbreviation	Name	RW	Size
0x00	IOADDR	Internal Register or Internal Memory location address. 0x00000-0x1FFFF – Internal Registers and Memories 0x20000-0xFFFFFFFF – Undefined	RW	4 bytes
0x04	IODATA	Data field for reads or writes to the Internal Register or Internal Memory Location as identified by the current value in IOADDR. All 32 bits of this register are read/write-able.	RW	4 bytes
0x08 – 0x1F	Reserved	Reserved	RO	4 bytes

8.1.1.5.1 IOADDR (I/O offset 0x00)

The IOADDR register must always be written as a DWORD access. Writes that are less than 32 bits are ignored. Reads of any size return a DWORD of data; however, the chipset or CPU might only return a subset of that DWORD.

For software programmers, the IN and OUT instructions must be used to cause I/O cycles to be used on the PCIe bus. Because writes must be to a 32-bit quantity, the source register of the OUT instruction must be EAX (the only 32-bit register supported by the OUT command). For reads, the IN instruction can have any size target register, but it is recommended that the 32-bit EAX register be used.

Because only a particular range is addressable, the upper bits of this register are hard coded to zero. Bits 31 through 20 are not write-able and always read back as 0b.

At hardware reset (LAN_PWR_GOOD) or PCI Reset, this register value resets to 0x00000000. Once written, the value is retained until the next write or reset.

8.1.1.5.2 IODATA (I/O offset 0x04)

The IODATA register must always be written as a DWORD access when the IOADDR register contains a value for the Internal Register and Memories (for example, 0x00000-0x1FFFC). In this case, writes that are less than 32 bits are ignored.

Reads to IODATA of any size returns a DWORD of data. However, the chipset or CPU might only return a subset of that DWORD.

For software programmers, the IN and OUT instructions must be used to cause I/O cycles to be used on the PCIe bus. Where 32-bit quantities are required on writes, the source register of the OUT instruction must be EAX (the only 32-bit register supported by the OUT command).

Writes and reads to IODATA when the IOADDR register value is in an undefined range (0x20000-0xFFFFFFFC) should not be performed. Results cannot be determined.

Notes: There are no special software timing requirements on accesses to IOADDR or IODATA. All accesses are immediate, except when data is not readily available or acceptable. In this case, 82580 delays the results through normal bus methods (for example, split transaction or transaction retry).

Because a register/memory read or write takes two IO cycles to complete, software must provide a guarantee that the two IO cycles occur as an atomic operation. Otherwise, results can be non-deterministic from the software viewpoint.

8.1.1.5.3 Undefined I/O offsets

I/O offsets 0x08 through 0x1F are considered to be reserved offsets with the I/O window. Dword reads from these addresses returns 0xFFFF; writes to these addresses are discarded.

8.1.1.6 Configuration Access to Internal Registers and Memories

To support 'legacy' pre-boot 16-bit operating environments without requiring IO address space, the 82580 enables accessing CSRs via configuration address space by mapping the *IOADDR* and *IODATA* registers into configuration address space. The registers mapping in this case is shown in Table 8-3.

Table 8-3. IOADDR and IODATA in Configuration Address space

Configuration Address	Abbreviation	Name	RW	Size
0x98	IOADDR	Internal register or internal memory location address. 0x00000-0x1FFFF – Internal Registers and Memories 0x20000-0x7FFFFF – Undefined	RW	4 bytes
0x9C	IODATA	Data field for reads or writes to the internal register or internal memory location as identified by the current value in IOADDR. All 32 bits of this register are read/write-able.	RW	4 bytes

Software writes data to an internal CSR via Configuration space in the following manner:

1. CSR address is written to the IOADDR register where:

   a. Bit 31 (*IOADDR.Configuration IO Access Enable*) of the The *IOADDR* register should be set to 1.
   b. Bits 30:0 of *IOADDR* should hold the actual address of the internal register or memory being written to.
2. Data to be written is written into the IODATA register.
   — The *IODATA* register is used as a "window" to the register or memory address specified by *IOADDR* register. As a result the data written to the *IODATA* register is written into the CSR pointed to by bits 30:0 of the *IOADDR* register.
3. *IOADDR.Configuration IO Access Enable* is cleared, to avoid un-intentional CSR read operations (that may cause clear by read) by other applications scanning the configuration space.

Software reads data from an internal CSR via Configuration space in the following manner:

1. CSR address is written to the IOADDR register where:
   a. Bit 31 (*IOADDR.Configuration IO Access Enable*) of the The *IOADDR* register should be set to 1.
   b. Bits 30:0 of *IOADDR* should hold the actual address of the internal register or memory being read.
2. CSR value is read from the *IODATA* register.
   a. The *IODATA* register is used as a "window" to the register or memory address specified by *IOADDR* register. As a result the data read from the *IODATA* register is the data of the CSR pointed to by bits 30:0 of the *IOADDR* register
3. *IOADDR.Configuration IO Access Enable* is cleared, to avoid un-intentional CSR read operations (that may cause clear by read) by other applications scanning the configuration space.

Notes:

- When Function is in D3 state Software should not attempt to access CSRs via the *IOADDR* and *IODATA* Configuration registers.
- To enable CSR access via configuration space, Software should set to 1 bit 31 (*IOADDR.Configuration IO Access Enable*) of the IOADDR register. Software should clear bit 31 of the IOADDR register after completing CSR access to avoid an unintentional "clear by read" operation, by another application scanning the configuration address space.
- Bit 31 of the *IOADDR* register (*IOADDR.Configuration IO Access Enable*) has no effect when initiating access via IO Address space.

8.1.2 Register Conventions

All registers in the 82580 are defined to be 32 bits, should be accessed as 32 bit double-words, There are some exceptions to this rule:

- Register pairs where two 32 bit registers make up a larger logical size
- Accesses to Flash memory (via Expansion ROM space, secondary BAR space, or the I/O space) might be byte, word or double word accesses.

Reserved bit positions: Some registers contain certain bits that are marked as "reserved". These bits should never be set to a value of "one" by software. Reads from registers containing reserved bits might return indeterminate values in the reserved bit-positions unless read values are explicitly stated. When read, these reserved bits should be ignored by software.

Reserved and/or undefined addresses: any register address not explicitly declared in this specification should be considered to be reserved, and should not be written to. Writing to reserved or undefined register addresses might cause indeterminate behavior. Reads from reserved or undefined configuration register addresses might return indeterminate values unless read values are explicitly stated for specific addresses.

Initial values: most registers define the initial hardware values prior to being programmed. In some cases, hardware initial values are undefined and is listed as such via the text "undefined", "unknown", or "X". Such configuration values might need to be set via EEPROM configuration or via software in order for proper operation to occur; this need is dependent on the function of the bit. Other registers might cite a hardware default which is overridden by a higher-precedence operation. Operations which might supersede hardware defaults might include a valid EEPROM load, completion of a hardware operation (such as hardware auto-negotiation), or writing of a different register whose value is then reflected in another bit.

For registers that should be accessed as 32 bit double words, partial writes (less than a 32 bit double word) does not take effect (the write is ignored).   Partial reads returns all 32 bits of data regardless of the byte enables.

Note: Partial reads to read-on-clear registers (*ICR*) can have unexpected results since all 32 bits are actually read regardless of the byte enables. Partial reads should not be done.

All statistics registers are implemented as 32 bit registers.   Though some logical statistics registers represent counters in excess of 32-bits in width, registers must be accessed using 32-bit operations (for example, independent access to each 32-bit field). When reading 64 bits statistics registers the least significant 32 bit register should be read first.

See special notes for VLAN Filter Table, Multicast Table Arrays and Packet Buffer Memory which appear in the specific register definitions.

The 82580 register fields are assigned one of the attributes described in Table 8-4.

Table 8-4. 82580 Register Field Attributes

Attribute	Description
RW	Read-Write field: Register bits are read-write and can be either set or cleared by software to the desired state.
RWS	Read-Write Status field: Register bits are read-write and can be either set or cleared by software to the desired state. However, the value of this field might be changed by the hardware to reflect a status change.
RO	Read-only register: Register bits are read-only and should not be altered by software. Register bits might be initialized by hardware mechanisms such as pin strapping, serial EEPROM or reflect a status of the hardware state.
R/W1C	Read-only status, Write-1-to-clear status register: Register bits indicate status when read, a set bit indicating a status event can be cleared by writing a 1b. Writing a 0b to R/W1C bit has no effect.
Rsv	Reserved. Write 0 to these fields and ignore read.
RC	Read-only status, Read-to-clear status register: Register bits indicate status when read, a set bit indicating a status event is cleared by reading it.
SC	Self Clear field: a command field that is self clearing. These field are always read as zero.
WO	Write only field: a command field that can not be read, These field read values are undefined.
RC/W	Read-Write status, Read-to-clear status register: Read-to-clear status register. Register bits indicate status when read. Register bits are read-write and can be either set or cleared by software to the desired state.
RC/W1C	Read-only status, Write-1-to-clear status register: Read-to-clear status register. Register bits indicate status when read, a set bit indicating a status event can be cleared by writing a 1b or by reading the register. Writing a 0b to RC/W1C bit has no effect.
RS	Read Set – This is the attribute used for Semaphore bits. These bits are set by read in case the previous values were zero. In this case the read value is zero; otherwise the read value is one. Cleared by write zero.

PHY registers described in Section 8.2.5 use a special nomenclature to define the read/write mode of individual bits in each register. See table bellow for details.

Table 8-5. PHY Register Nomenclature

Register Mode	Description
LH	Latched High. Event is latched and erased when read.
LL	Latched Low. Event is latched and erased when read. For example, Link Loss is latched when the PHY Control Register bit 2 = 0b. After read, if the link is good, the PHY Control Register bit 2 is set to 1b.
RO	Read Only.
R/W	Read and Write.
SC	Self-Clear. The bit is set, automatically executed, and then reset to normal operation.
CR	Clear after Read. For example, 1000BASE-T Status Register bits 7:0 (Idle Error Counter).
Update	Value written to the register bit does not take effect until software PHY reset is executed.

Note: For all binary equations appearing in the register map, the symbol "|" is equivalent to a binary OR operation.

8.1.2.1 Registers Byte Ordering

This section defines the structure of registers that contain fields carried over the network. Some examples are L2, L3 and L4 fields.

The following example is used to describe byte ordering over the wire (hex notation):

```
Last                         First
...,06, 05, 04, 03, 02, 01, 00
```

Each byte is sent with the LSbit first. That is, the bit order over the wire for this example is

```
Last                                       First
..., 0000 0011, 0000 0010, 0000 0001, 0000 0000
```

The general rule for register ordering is to use Host Ordering (also called little endian). Using the above example, a 6-byte fields (MAC address) is stored in a CSR in the following manner:

```
                  Byte 3  Byte 2  Byte 1  Byte0

DW address (N)    0x03    0x02    0x01    0x00

DW address (N+4)  ...     ...     0x05    0x04
```

The exceptions listed below use network ordering (also called big endian). Using the above example, a 16-bit field (EtherType) is stored in a CSR in the following manner:

```
                  Byte 3  Byte 2  Byte 1  Byte0
(DW aligned)      ...     ...     0x00    0x01
or
(Word aligned)    0x00    0x01    ...     ...
```

The following exceptions use network ordering:

All ETherType fields. For example, the VET EXT field in the VET register, the EType field in the ETQF register, the EType field in the METF register.

Note: The "normal" notation as it appears in text books, etc. is to use network ordering. Example: Suppose a MAC address of 00-A0-C9-00-00-00. The order on the network is 00, then A0, then C9, etc. However, the host ordering presentation would be:

```
                    Byte 3  Byte 2  Byte 1  Byte0
DW address (N)      00      C9      A0      00
DW address (N+4)    ...     ...     00      00
```

8.1.3 Register Summary

All the 82580's non-PCIe configuration registers, except for the MSI-X register, are listed in the table below. These registers are ordered by grouping and are not necessarily listed in order that they appear in the address space.

Table 8-6. Register Summary

Offset	Alias Offset	Abbreviation	Name	RW
		General		
0x0000	0x0004	CTRL	Device Control Register	RW
0x0008	N/A	STATUS	Device Status Register	RO
0x0018	N/A	CTRL_EXT	Extended Device Control Register	RW
0x0020	N/A	MDIC	MDI Control Register	RW
0x0E08	N/A	P1GCTRL0	Serdes Control 0	RW
0x0028	N/A	FCAL	Flow Control Address Low	RO
0x002C	N/A	FCAH	Flow Control Address High	RO
0x0030	N/A	FCT	Flow Control Type	RW
0x0034	N/A	CONNSW	Copper/Fiber switch control	RW
0x0038	N/A	VET	VLAN Ether Type	RW
0x0E04	N/A	MDICNFG	MDC/MDIO Configuration Register	RW
0x0170	N/A	FCTTV	Flow Control Transmit Timer Value	RW
0x0E00	N/A	LEDCTL	LED Control Register	RW
0x1028	N/A	I2CCMD	SFP I2C command	RW
0x102C	N/A	I2CPARAMS	SFP I2C Parameter	RW
0x1040	N/A	WDSTP	Watchdog setup register	RW
0x1044	N/A	WDSWSTS	Watchdog Software	RW
0x1048	N/A	FRTIMER	Free Running Timer	RWS
0x104C	N/A	TCPTimer	TCP timer	RW
0x2508	N/A	DMACR	DMA Coalescing Control Register	RW
0x2514	N/A	DMCTLX	DMA Coalescing Time to LX Request	RW
0x3550	N/A	DMCTXTH	DMA Coalescing Transmit Threshold	RW
0x5B70	N/A	DCA_ID	DCA Requester ID Information Register	RO
0x5B50	N/A	SWSM	Software Semaphore Register	RW

Table 8-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0x5B54	N/A	FWSM	Firmware Semaphore Register	RWS
0x5B5C	N/A	SW_FW_SYNC	Software-Firmware Synchronization	RWS
0x5B04	N/A	SWMBWR	Software Mailbox Write	RW
0x5B08	N/A	SWMB0	Software Mailbox Port 0	RO
0x5B0C	N/A	SWMB1	Software Mailbox Port 1	RO
0x5B18	N/A	SWMB2	Software Mailbox Port 2	RO
0x5B1C	N/A	SWMB3	Software Mailbox Port 3	RO
0x0E14	N/A	PHPM	PHY Power Management	RW
Flash/EEPROM registers				
0x0010	N/A	EEC	EEPROM/Flash Control Register	RW
0x0014	N/A	EERD	EEPROM Read Register	RW
0x001C	N/A	FLA	Flash Access Register	RW
0x1010	N/A	EEMNGCTL	MNG EEPROM Control Register	RO
0x1014	N/A	EEMNGDATA	MNG EEPROM Read/Write data	RO
0x1024	N/A	EEARBC	EEPROM Auto Read Bus Control	RW
0x103C	N/A	FLASHOP	Flash Opcode Register	RW
0x1038	N/A	EEDIAG	EEPROM Diagnostic	RO
0x1060	N/A	VPDDIAG	VPD Diagnostic	RO
Interrupts				
0x1500	0x00C0	ICR	Interrupt Cause Read	RC/W1C
0x1504	0x00C8	ICS	Interrupt Cause Set	WO
0x1508	0x00D0	IMS	Interrupt Mask Set/Read	RW
0x150C	0x00D8	IMC	Interrupt Mask Clear	WO
0x1510	0x00E0	IAM	Interrupt Acknowledge Auto Mask	RW
0x1520	N/A	EICS	Extended Interrupt Cause Set	WO
0x1524	N/A	EIMS	Extended Interrupt Mask Set/Read	RWS
0x1528	N/A	EIMC	Extended Interrupt Mask Clear	WO
0x152C	N/A	EIAC	Extended Interrupt Auto Clear	RW
0x1530	N/A	EIAM	Extended Interrupt Auto Mask	RW
0x1580	N/A	EICR	Extended Interrupt Cause Read	RC/W1C
0x1700 - 0x170C	N/A	IVAR	Interrupt Vector Allocation Registers	RW
0x1740	N/A	IVAR_MISC	Interrupt Vector Allocation Registers - MISC	RW
0x1680 - 0x16A0	N/A	EITR	Extended Interrupt Throttling Rate 0 - 9	RW
0x1514	N/A	GPIE	General Purpose Interrupt Enable	RW
0x5B68	N/A	PBACL	MSI-X PBA Clear	R/W1C
Receive				
0x0100	N/A	RCTL	RX Control	RW
0x01A0	N/A	LTRC	Latency Tolerance Reporting (LTR) Control	RW
0x2160	0x0168	FCRTL0	Flow Control Receive Threshold Low	RW
0x2168	0x0160	FCRTH0	Flow Control Receive Threshold High	RW
0x2170	N/A	FCRTC	Flow Control Receive Threshold Coalescing	RW
0x2404	N/A	IRPBS	Internal Receive Packet Buffer Size	RO

Table 8-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0x2460	N/A	FCRTV	Flow control Refresh timer value	RW
0xC000	0x0110, 0x2800	RDBAL[0]	RX Descriptor Base Low queue 0	RW
0xC004	0x0114, 0x2804	RDBAH[0]	RX Descriptor Base High queue 0	RW
0xC008	0x0118, 0x2808	RDLEN[0]	RX Descriptor Ring Length queue 0	RW
0xC00C	0x280C	SRRCTL[0]	Split and Replication Receive Control Register queue 0	RW
0xC010	0x0120, 0x2810	RDH[0]	RX Descriptor Head queue 0	RO
0xC018	0x0128, 0x2818	RDT[0]	RX Descriptor Tail queue 0	RW
0xC028	0x02828	RXDCTL[0]	Receive Descriptor Control queue 0	RW
0xC014	0x2814	RXCTL[0]	Receive Queue 0 DCA CTRL Register	RW
0xC040 + 0x40 * (n-1)	0x2900+ 0x100 * (n-1)	RDBAL[1 - 3]	RX Descriptor Base Low queue 1 - 3	RW
0xC044 + 0x40 * (n-1)	0x2904 + 0x100 * (n-1)	RDBAH[1 - 3]	RX Descriptor Base High queue 1 - 3	RW
0xC048 + 0x40 * (n-1)	0x2908 + 0x100 * (n-1)	RDLEN[1 - 3]	RX Descriptor Ring Length queue 1 - 3	RW
0xC04C + 0x40 * (n-1)	0x290C + 0x100 * (n-1)	SRRCTL[1 - 3]	Split and Replication Receive Control Register queue 1 - 3	RW
0xC050 + 0x40 * (n-1)	0x2910 + 0x100 * (n-1)	RDH[1 - 3]	RX Descriptor Head queue 1 - 3	RO
0xC058 + 0x40 * (n-1)	0x2918 + 0x100 * (n-1)	RDT[1 - 3]	RX Descriptor Tail queue 1 - 3	RW
0xC068 + 0x40 * (n-1)	0x2928 + 0x100 * (n-1)	RXDCTL[1 - 3]	Receive Descriptor Control queue 1 - 3	RW
0xC054 + 0x40 * (n-1)	0x2914 + 0x100 * (n-1)	RXCTL[1 - 3]	Receive Queue 1 - 3 DCA CTRL Register	RW
0xC100 + 0x40 * (n- 4)	N/A	RDBAL[4-7]	RX Descriptor Base Low queue 4 - 7	RW
0xC104 + 0x40 * (n- 4)	N/A	RDBAH[4-7]	RX Descriptor Base High queue 4 - 7	RW
0xC108 + 0x40 * (n- 4)	N/A	RDLEN[4-7]	RX Descriptor Ring Length queue 4 - 7	RW
0xC10C + 0x40 * (n- 4)	N/A	SRRCTL[4 -7]	Split and Replication Receive Control Register queue 4 - 7	RW
0xC110 + 0x40 * (n- 4)	N/A	RDH[4 - 7]	RX Descriptor Head queue 4 - 7	RO
0xC118 + 0x40 * (n- 4)	N/A	RDT[4 - 7]	RX Descriptor Tail queue 4 - 7	RW
0xC128 + 0x40 * (n- 4)	N/A	RXDCTL[4 - 7]	Receive Descriptor Control queue 4 - 7	RW
0xC114 + 0x40 * (n- 4)	N/A	RXCTL[4 - 7]	Receive Queue 4 - 7 DCA CTRL Register	RW
0x5000	N/A	RXCSUM	Receive Checksum Control	RW

Table 8-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0x5004	N/A	RLPML	Receive Long packet maximal length	RW
0x5008	N/A	RFCTL	Receive Filter Control Register	RW
0x5200- 0x53FC	0x0200-0x03FC	MTA[127:0]	Multicast Table Array (n)	RW
0x5400 + 8*n	0x0040 + 8*n	RAL[0-15]	Receive Address Low (15:0)	RW
0x5404 + 8 *n	0x0044 + 8 *n	RAH[0-15]	Receive Address High (15:0)	RW
0x54E0 + 8*n	N/A	RAL[16-23]	Receive Address Low (23:16)	RW
0x54E4 + 8 *n	N/A	RAH[16-23]	Receive Address High (23:16)	RW
0x5480 – 0x549C	N/A	PSRTYPE[7:0]	Packet Split Receive type (n)	RW
0x54C0	N/A	RPLPSRTYPE	Replicated Packet Split Receive type	RW
0x581C	N/A	VT_CTL	VMDq Control register	RW
0x5600-0x57FC	0x0600-0x07FC	VFTA[127:0]	VLAN Filter Table Array (n)	RW
0x5818	N/A	MRQC	Multiple Receive Queues Command	RW
0x5C00-0x5C7C	N/A	RETA	Redirection Table	RW
0x5C80-0x5CA4	N/A	RSSRK	RSS Random Key Register	RW
0x5DD0	N/A	DMCRTRH	DMA Coalescing Receive Packet Rate Threshold	RW
0x5DD4	N/A	DMCCNT	DMA Coalescing Current RX Count	RO
Transmit				
0x0400	N/A	TCTL	TX Control	RW
0x0404	N/A	TCTL_EXT	TX Control extended	RW
0x0410	N/A	TIPG	TX IPG	RW
0x041C	N/A	RETX_CTL	Retry Buffer Control	RW
0x3404	N/A	ITPBS	Internal Transmit Packet Buffer Size	RO
0x359C	N/A	DTXTCPFLGL	DMA TX TCP Flags Control Low	RW
0x35A0	N/A	DTXTCPFLGH	DMA TX TCP Flags Control High	RW
0x3540	N/A	DTXMXSZRQ	DMA TX Max Total Allow Size Requests	RW
0x355C	N/A	DTXMXPKTSZ	DMA TX Max Allowable packet size	RW
0x3590	N/A	DTXCTL	DMA TX Control	RW
0xE000	0x0420, 0x3800	TDBAL[0]	TX Descriptor Base Low 0	RW
0xE004	0x0424, 0x3804	TDBAH[0]	TX Descriptor Base High 0	RW
0xE008	0x0428, 0x3808	TDLEN[0]	TX Descriptor Ring Length 0	RW
0xE010	0x0430, 0x3810	TDH[0]	TX Descriptor Head 0	RO
0xE018	0x0438, 0x3818	TDT[0]	TX Descriptor Tail 0	RW
0xE028	0x3828	TXDCTL[0]	Transmit Descriptor Control queue 0	RW
0xE014	0x3814	TXCTL[0]	TX DCA CTRL Register Queue 0	RW
0xE038	0x3838	TDWBAL[0]	Transmit Descriptor WB Address Low queue 0	RW
0xE03C	0x383C	TDWBAH[0]	Transmit Descriptor WB Address High queue 0	RW

Table 8-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0xE040 + 0x40 * (n-1)	0x3900 + 0x100 * (n-1)	TDBAL[1-3]	TX Descriptor Base Low queue 1 - 3	RW
0xE044 + 0x40 * (n-1)	0x3904 + 0x100 * (n-1)	TDBAH[1-3]	TX Descriptor Base High queue 1 - 3	RW
0xE048 + 0x40 * (n-1)	0x3908 + 0x100 * (n-1)	TDLEN[1-3]	TX Descriptor Ring Length queue 1 - 3	RW
0xE050 + 0x40 * (n-1)	0x3910 + 0x100 * (n-1)	TDH[1-3]	TX Descriptor Head queue 1 - 3	RO
0xE058 + 0x40 * (n-1)	0x3918 + 0x100 * (n-1)	TDT[1-3]	TX Descriptor Tail queue 1 - 3	RW
0xE068 + 0x40 * (n-1)	0x3928 + 0x100 * (n-1)	TXDCTL[1-3]	Transmit Descriptor Control 1 - 3	RW
0xE054 + 0x40 * (n-1)	0x3914 + 0x100 * (n-1)	TXCTL[1-3]	TX DCA CTRL Register Queue 1 - 3	RW
0xE078 + 0x40 * (n-1)	0x3938 + 0x100 * (n-1)	TDWBAL[1-3]	Transmit Descriptor WB Address Low queue 1 - 3	RW
0xE07C + 0x40 * (n-1)	0x393C + 0x100 * (n-1)	TDWBAH[1-3]	Transmit Descriptor WB Address High queue 1 - 3	RW
0xE180 + 0x40 *n	N/A	TDBAL[4 - 7]	TX Descriptor Base Low queue 4 - 7	RW
0xE184 + 0x40 * n	N/A	TDBAH[4 - 7]	TX Descriptor Base High queue 4 - 7	RW
0xE188 + 0x40 * n	N/A	TDLEN[4 - 7]	TX Descriptor Ring Length queue 4 - 7	RW
0xE190 + 0x40 * n	N/A	TDH[4 - 7]	TX Descriptor Head queue 4 - 7	RO
0xE198 + 0x40 * n	N/A	TDT[4 - 7]	TX Descriptor Tail queue 4 - 7	RW
0xE1A8 + 0x40 * n	N/A	TXDCTL[4 - 7]	Transmit Descriptor Control 4 - 7	RW
0xE194 + 0x40 * n	N/A	TXCTL[4 - 7]	TX Queue 4 - 7DCA CTRL Register	RW
0xE1B8 + 0x40 * n	N/A	TDWBAL[4 - 7]	Transmit Descriptor WB Address Low queue 4 - 7	RW
0xE1BC + 0x40 * n	N/A	TDWBAH[4 - 7]	Transmit Descriptor WB Address High queue 4 - 7	RW
Filters				
0x5CB0 + 4*n	N/A	ETQF[0 - 7]	EType Queue Filter 0 - 7	RW
0x5A80 + 4*n	N/A	IMIR[0 - 7]	Immediate Interrupt Rx 0 - 7	RW
0x5AA0 + 4*n	N/A	IMIREXT[0 - 7]	Immediate Interrupt Rx Extended 0 - 7	RW
0x5AC0	N/A	IMIRVP	Immediate Interrupt Rx VLAN Priority	RW
0x59E0 + 4*n	N/A	TTQF[0 - 7]	Two-Tuple Queue Filter 0 - 7	RW
0x55FC	N/A	SYNQF	SYN Packet Queue Filter	RW
Virtualization				
0x3548	N/A	LVMMC	Last VM Misbehavior Cause	RC

Table 8-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0x3558	N/A	MDFB	Malicious Driver Free Block	RWS
0x5D00 - 0x5D7C	N/A	VLVF	VLAN VM Filter	RW
0x5AD0 - 0x5ADC	N/A	VMOLR[0 - 7]	VM Offload register[0-7]	RW
0x5AF0	N/A	RPLOLR	Replication Offload register	RW
0xA000 - 0xA1FC	N/A	UTA	Unicast Table Array	RW
0x5D80 - 0x5D8C	N/A	VMRCTL	Virtual Mirror rule control	RW
0x5D90 - 0x5D9C	N/A	VMRVLAN	Virtual Mirror rule VLAN	RW
0x5DA0 - 0x5DAC	N/A	VMRVM	Virtual Mirror rule VM	RW
0x5DB0	N/A	SCCRL	Storm Control control register	RW
0x5DB4	N/A	SCSTS	Storm Control status	RO
0x5DB8	N/A	BSCTRH	Broadcast Storm control Threshold	RW
0x5DBC	N/A	MSCTRH	Multicast Storm control Threshold	RW
0x5DC0	N/A	BSCCNT	Broadcast Storm Control Current Count	RO
0x5DC4	N/A	MSCCNT	Multicast Storm control Current Count	RO
0x5DC8	N/A	SCTC	Storm Control Time Counter	RO
VMDq Statistics				
0xC030	0x2830	RQDPC[0]	Receive Queue drop packet count Register 0	RC/W
0xC070 + 0x40 * (n-1)	0x2930 + 0x100 * (n-1)	RQDPC[1 - 3]	Receive Queue drop packet count Register 1 - 3	RC/W
0xC130 + 0x40 * (n- 4)	N/A	RQDPC[4 - 7]	Receive Queue drop packet count Register 4 - 7	RC/W
0x10010 + 0x100*n	N/A	VFGPRC[0 - 7]	Per queue Good Packets Received Count	RO
0x10014 + 0x100*n	N/A	VFGPTC[0 - 7]	Per queue Good Packets Transmitted Count	RO
0x10018 + 0x100*n	N/A	VFGORC[0 - 7]	Per queue Good Octets Received Count	RO
0x10034 + 0x100*n	N/A	VFGOTC[0 - 7]	Per queue Octets Transmitted Count	RO
0x1003C + 0x100*n	N/A	VFMPRC[0 - 7]	Per queue Multicast Packets Received Count	RO
Statistics				
0x4000	N/A	CRCERRS	CRC Error Count	RC
0x4004	N/A	ALGNERRC	Alignment Error Count	RC
0x4008	N/A	SYMERRS	Symbol Error Count	RC
0x400C	N/A	RXERRC	RX Error Count	RC
0x4010	N/A	MPC	Missed Packets Count	RC
0x4014	N/A	SCC	Single Collision Count	RC
0x4018	N/A	ECOL	Excessive Collisions Count	RC
0x401C	N/A	MCC	Multiple Collision Count	RC
0x4020	N/A	LATECOL	Late Collisions Count	RC
0x4028	N/A	COLC	Collision Count	RC

Table 8-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0x4030	N/A	DC	Defer Count	RC
0x4034	N/A	TNCRS	Transmit - No CRS	RC
0x403C	N/A	HTDPMC	Host Transmit Discarded Packets by MAC Count	RC
0x4040	N/A	RLEC	Receive Length Error Count	RC
0x4048	N/A	XONRXC	XON Received Count	RC
0x404C	N/A	XONTXC	XON Transmitted Count	RC
0x4050	N/A	XOFFRXC	XOFF Received Count	RC
0x4054	N/A	XOFFTXC	XOFF Transmitted Count	RC
0x4058	N/A	FCRUC	FC Received Unsupported Count	RC
0x405C	N/A	PRC64	Packets Received (64 Bytes) Count	RC
0x4060	N/A	PRC127	Packets Received (65-127 Bytes) Count	RC
0x4064	N/A	PRC255	Packets Received (128-255 Bytes) Count	RC
0x4068	N/A	PRC511	Packets Received (256-511 Bytes) Count	RC
0x406C	N/A	PRC1023	Packets Received (512-1023 Bytes) Count	RC
0x4070	N/A	PRC1522	Packets Received (1024-1522 Bytes)	RC
0x4074	N/A	GPRC	Good Packets Received Count	RC
0x4078	N/A	BPRC	Broadcast Packets Received Count	RC
0x407C	N/A	MPRC	Multicast Packets Received Count	RC
0x4080	N/A	GPTC	Good Packets Transmitted Count	RC
0x4088	N/A	GORCL	Good Octets Received Count (Lo)	RC
0x408C	N/A	GORCH	Good Octets Received Count (Hi)	RC
0x4090	N/A	GOTCL	Good Octets Transmitted Count (Lo)	RC
0x4094	N/A	GOTCH	Good Octets Transmitted Count (Hi)	RC
0x40A0	N/A	RNBC	Receive No Buffers Count	RC
0x40A4	N/A	RUC	Receive Under size Count	RC
0x40A8	N/A	RFC	Receive Fragment Count	RC
0x40AC	N/A	ROC	Receive Oversize Count	RC
0x40B0	N/A	RJC	Receive Jabber Count	RC
0x40B4	N/A	MNGPRC	Management Packets Receive Count	RC
0x40B8	N/A	MPDC	Management Packets Dropped Count	RC
0x40BC	N/A	MNGPTC	Management Packets Transmitted Count	RC
0x40C0	N/A	TORL	Total Octets Received (Lo)	RC
0x40C4	N/A	TORH	Total Octets Received (Hi)	RC
0x40C8	N/A	TOTL	Total Octets Transmitted (Lo)	RC
0x40CC	N/A	TOTH	Total Octets Transmitted (Hi)	RC
0x40D0	N/A	TPR	Total Packets Received	RC
0x40D4	N/A	TPT	Total Packets transmitted	RC
0x40D8	N/A	PTC64	Packets Transmitted (64 Bytes) Count	RC
0x40DC	N/A	PTC127	Packets Transmitted (65-127 Bytes) Count	RC
0x40E0	N/A	PTC255	Packets Transmitted (128-256 Bytes) Count	RC
0x40E4	N/A	PTC511	Packets Transmitted (256-511 Bytes) Count	RC
0x40E8	N/A	PTC1023	Packets Transmitted (512-1023 Bytes) Count	RC
0x40EC	N/A	PTC1522	Packets Transmitted (1024-1522 Bytes) Count	RC
0x40F0	N/A	MPTC	Multicast Packets Transmitted Count	RC

Table 8-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0x40F4	N/A	BPTC	Broadcast Packets Transmitted Count	RC
0x40F8	N/A	TSCTC	TCP Segmentation Context Transmitted Count	RC
0x4100	N/A	IAC	Interrupt Assertion Count	RC
0x4104	N/A	RPTHC	Rx Packets to host count	RC
0x4118	N/A	HGPTC	Host Good Packets Transmitted Count	RC
0x4120	N/A	RXDMTC	Rx Descriptor Minimum Threshold Count	RC
0x4128	N/A	HGORCL	Host Good Octets Received Count (Lo)	RC
0x412C	N/A	HGORCH	Host Good Octets Received Count (Hi)	RC
0x4130	N/A	HGOTCL	Host Good Octets Transmitted Count (Lo)	RC
0x4134	N/A	HGOTCH	Host Good Octets Transmitted Count (Hi)	RC
0x4138	N/A	LENERRS	Length Errors count register	RC
0x4228	N/A	SCVPC	SerDes/SGMII/1000BASE-KX Code Violation Packet Count Register	RW
0x41A4	N/A	SDPC	Switch Drop Packet Count	RC
Manageability Statistics				
0x413C	N/A	BMNGPRC	BMC Management Packets Receive Count	RC
0x4140	N/A	BMPDC	BMC Management Packets Dropped Count	RC
0x4144	N/A	BMNGPTC	BMC Management Packets Transmitted Count	RC
0x4400	N/A	BUPRC	BMC Total Unicast Packets Received	RC
0x4404	N/A	BMPRC	BMC Total Multicast Packets Received	RC
0x4408	N/A	BBPRC	BMC Total Broadcast Packets Received	RC
0x440C	N/A	BUPTC	BMC Total Unicast Packets Transmitted	RC
0x4410	N/A	BMPTC	BMC Total Multicast Packets Transmitted	RC
0x4414	N/A	BBPTC	BMC Total Broadcast Packets Transmitted	RC
0x4418	N/A	BCRCERRS	BMC FCS Receive Errors	RC
0x441C	N/A	BALGNERRC	BMC Alignment Errors	RC
0x4420	N/A	BXONRXC	BMC Pause XON Frames Received	RC
0x4424	N/A	BXOFFRXC	BMC Pause XOFF Frames Received	RC
0x4428	N/A	BXONTXC	BMC Pause XON Frames Transmitted	RC
0x442C	N/A	BXOFFTXC	BMC Pause XOFF Frames Transmitted	RC
0x4430	N/A	BSCC	BMC Single Collision Transmit Frames	RC
0x4434	N/A	BMCC	BMC Multiple Collision Transmit Frames	RC
Wake up				
0x5800	N/A	WUC	Wake Up Control	RW
0x5808	N/A	WUFC	Wake Up Filter Control	RW
0x5810	N/A	WUS	Wake Up Status	R/W1C
0x5900	N/A	WUPL	Wake Up Packet Length	RO
0x5A00- 0x5A7C	N/A	WUPM	Wake Up Packet Memory	RO
0x9000-0x93FC	N/A	FHFT	Flexible Host Filter Table registers	RW
0x9A00-0x9DFC	N/A	FHFT_EXT	Flexible Host Filter Table registers extended	RW
Manageability				
0x5010 - 0x502C	N/A	MAVTV[7:0]	VLAN TAG Value 7 - 0	RW

Table 8-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0x5030 - 0x504C	N/A	MFUTP[7:0]	Management Flex UDP/TCP Ports	RW
0x5060 - 0x506C	N/A	METF[3:0]	Management Ethernet Type Filters	RW
0x5820	N/A	MANC	Management Control	RW
0x5838	N/A	IPAV	IP Address Valid	RW
0x5840- 0x5858	N/A	IP4AT	IPv4 Address Table	RW
0x5864	N/A	MNGONLY	Management Only Traffic Register	RW
0x5880- 0x588F	N/A	IP6AT	IPv6 Address Table	RW
0x5890 - 0x58AC	N/A	MDEF[7:0]	Manageability Decision Filters	RW
0x5930 – 0x594C	N/A	MDEF_EXT[7:0]	Manageability Decision Filters	RW
0x58B0 - 0x58EC	N/A	MIPAF[15:0]	Manageability IP Address Filter	RW
0x5910 + 8*n	N/A	MMAL[1:0]	Manageability MAC Address Low 1:0	RW
0x5914 + 8*n	N/A	MMAH[1:0]	Manageability MAC Address High 1:0	RW
0x9400-0x94FC	N/A	FTFT	Flexible TCO Filter Table	RW
0x8800-0x8EFC	N/A	Flex MNG	Flex manageability memory address space	RW
PCIe				
0x5BB0	N/A	LTRMINV	Latency Tolerance Reporting (LTR) Minimum Values	RW
0x5BB4	N/A	LTRMAXV	Latency Tolerance Reporting (LTR) Maximum Values	RW
0x5B00	N/A	GCR	PCIe Control Register	RW
0x5B10	N/A	GSCL_1	PCIe statistics control #1	RW
0x5B14	N/A	GSCL_2	PCIe statistics control #2	RW
0x5B90 - 0x5B9C	N/A	GSCL_5_8	PCIe statistics control Leaky Bucket Timer	RW
0x5B20	N/A	GSCN_0	PCIe counter register #0	RW
0x5B24	N/A	GSCN_1	PCIe counter register #1	RW
0x5B28	N/A	GSCN_2	PCIe counter register #2	RW
0x5B2C	N/A	GSCN_3	PCIe counter register #3	RW
0x5B30	N/A	FACTPS	Function Active and Power State	RW
0x5B64	N/A	MREVID	Mirrored Revision ID	RO
0x5B6C	N/A	GCR_EXT	PCIe Control Extended Register	RW
0x5B74	N/A	DCA_CTRL	DCA Control Register	RW
0x5B88	N/A	PICAUSE	PCIe Interrupt Cause	R/W1C
0x5B8C	N/A	PIENA	PCIe Interrupt Enable	RW
0x5BBC	N/A	BARCTRL	PCIe BAR Control	RW
Memory Error Detection				
0x1084	N/A	PEIND	Parity and ECC Indication	RC
0x1088	N/A	PEINDM	Parity and ECC Indication Mask	RW
0x245C	N/A	RPBECCSTS	Receive Packet buffer ECC control	RW
0x345C	N/A	TPBECCSTS	Transmit Packet buffer ECC control	RW
0x5BA0	N/A	PCIEERRCTL	PCIe Parity Control Register	RW
0x5BA4	N/A	PCIEECCCTL	PCIe ECC Control Register	RW

Table 8-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0x5BA8	N/A	PCIEERRSTS	PCIe Parity status Register	R/W1C
0x5BAC	N/A	PCIEECCSTS	PCIe ECC Status Register	R/W1C
0x5F54	N/A	LANPERRCTL	LAN Port Parity Error Control register	RW
0x5F58	N/A	LANPERRSTS	LAN Port Parity Error Status register	RO
0x3500	N/A	DTPARC	DMA Transmit Descriptor Parity Control	RW
0x3510	N/A	DTPARS	DMA Transmit Descriptor Parity Status	RW
0x3504	N/A	DRPARC	DMA Receive Descriptor Parity Control	RW
0x3514	N/A	DRPARS	DMA Receive Descriptor Parity Status	R/W1C
0x3508	N/A	DDPARC	Dhost Parity Control	RW
0x3518	N/A	DDPARS	Dhost Parity Status	R/W1C
PCS				
0x4200	N/A	PCS_CFG	PCS Configuration 0 Register	RW
0x4208	N/A	PCS_LCTL	PCS Link Control Register	RW
0x420C	N/A	PCS_LSTS	PCS Link Status Register	RO
0x4210	N/A	PCS_DBG0	PCS Debug 0 register	RO
0x4214	N/A	PCS_DBG1	PCS Debug 1 register	RO
0x4218	N/A	PCS_ANADV	AN advertisement Register	RW
0x421C	N/A	PCS_LPAB	Link Partner Ability Register	RO
0x4220	N/A	PCS_NPTX	AN Next Page Transmit Register	RW
0x4224	N/A	PCS_LPABNP	Link Partner Ability Next Page Register	RO
Time Sync				
0xB620	N/A	TSYNCRXCTL	RX Time Sync Control register	RW
0xB624	N/A	RXSTMPL	RX timestamp Low	RO
0xB628	N/A	RXSTMPH	RX timestamp High	RO
0xB62C	N/A	RXSATRL	RX timestamp attributes low	RO
0xB630	N/A	RXSATRH	RX timestamp attributes low	RO
0xB614	N/A	TSYNCTXCTL	TX Time Sync Control register	RW
0xB618	N/A	TXSTMPL	TX timestamp value Low	RO
0xB61C	N/A	TXSTMPH	TX timestamp value High	RO
0xB6F8	N/A	SYSTIMR	System time residue register	RW
0xB600	N/A	SYSTIML	System time register Low	RW
0xB604	N/A	SYSTIMH	System time register High	RW
0xB608	N/A	TIMINCA	Increment attributes register	RW
0xB60C	N/A	TIMADJL	Time adjustment offset register low	RW
0xB610	N/A	TIMADJH	Time adjustment offset register high	RW
0xB640	N/A	TSAUXC	Auxiliary Control Register	RW
0xB644	N/A	TRGTTIML0	Target Time register 0 Low	RW
0xB648	N/A	TRGTTIMH0	Target Time register 0 High	RW
0xB64C	N/A	TRGTTIML1	Target Time register 1 Low	RW
0xB650	N/A	TRGTTIMH1	Target Time register 1 High	RW
0xB654	N/A	FREQOUT0	Frequency out 0 Control register	RW
0xB658	N/A	FREQOUT1	Frequency out 1 Control register	RW
0xB65C	N/A	AUXSTMPL0	Auxiliary Time Stamp 0 register Low	RO

Table 8-6. Register Summary (Continued)

Offset	Alias Offset	Abbreviation	Name	RW
0xB660	N/A	AUXSTMPH0	Auxiliary Time Stamp 0 register High	RO
0xB664	N/A	AUXSTMPL1	Auxiliary Time Stamp 1 register Low	RO
0xB668	N/A	AUXSTMPH1	Auxiliary Time Stamp 1 register High	RO
0x5F50	N/A	TSYNCRXCFG	Time Sync RX Configuration	RW
0x003C	N/A	TSSDP	Time Sync SDP Configuration Reg	RW

8.1.4 MSI-X BAR Register Summary

Certain registers maintain an alias address designed for backward compatibility with software written for previous GbE controllers. For these registers, the alias address is shown in the table above. Those registers can be accessed by software at either the new offset or the alias offset. It is recommended that software that is written solely for the 82580, use the new address offset.

Table 8-7. MSI-X Register Summary

Category	Offset	Abbreviation	Name	RW	Page
MSI-X Table	0x0000 + n*0x10 [n=0...9]	MSIXTADD	MSI–X Table Entry Lower Address	RW	page 425
MSI-X Table	0x0004 + n*0x10 [n=0...9]	MSIXTUADD	MSI–X Table Entry Upper Address	RW	page 425
MSI-X Table	0x0008 + n*0x10 [n=0...9]	MSIXTMSG	MSI–X Table Entry Message	R/W	page 425
MSI-X Table	0x000C + n*0x10 [n=0...9]	MSIXTVCTRL	MSI–X Table Entry Vector Control	R/W	page 425
MSI-X Table	0x02000	MSIXPBA	MSIXPBA Bit Description	RO	page 426

8.2 General Register Descriptions

8.2.1 Device Control Register - CTRL (0x00000; R/W)

This register, as well as the Extended Device Control register (CTRL_EXT), controls the major operational modes for the device. While software write to this register to control device settings, several bits (such as FD and SPEED) can be overridden depending on other bit settings and the resultant link configuration determined by the PHY's Auto- Negotiation resolution. See Section 4.6.7 for details on the setup of these registers in the different link modes.

Note: This register is also aliased at address 0x0004.

Field	Bit(s)	Initial Value	Description
FD	0	1b[1]	Full-Duplex Controls the MAC duplex setting when explicitly set by software. 0b = half duplex. 1b = full duplex.
Reserved	1	0b	This bit is reserved and should be set to 0b for future compatibility.
GIO Master Disable	2	0b	When set to 1b, the function of this bit blocks new master requests including manageability requests. If no master requests are pending by this function, the *STATUS.GIO Master Enable Status* bit is set. See Section 5.2.3.3 for further information.

Field	Bit(s)	Initial Value	Description
Reserved	5:3	0x0	Reserved Write 0 ignore on read.
SLU	6	0b[1]	Set Link Up. When the MAC link mode is set for GMII/MII mode (internal PHY), Set Link Up must be set to 1 to permit the MAC to recognize the LINK signal from the PHY, which indicates the PHY has gotten the link up, and is ready to receive and transmit data. See Section 3.5.4 for more information about Auto-Negotiation and link configuration in the various modes. The "Set Link Up" is normally initialized to 0. However, if the *APM Enable* bit is set in the EEPROM then it is initialized to 1b. In SerDes and 1000Base-KX modes Link up can be forced by setting this bit as described in Section 3.5.4.1.4.
ILOS	7	0b[1]	Invert Loss-of-Signal (LOS/LINK) Signal Bit controls the polarity of the SRDS_[n]_SIG_DET signal or internal Link up signal depending on the *CONNSW.ENRGSRC* bit. 0b = Do not invert (active high input signal). 1b = Invert signal (active low input signal). Should be set to zero when using internal PHY or when working in 1000BASE-KX mode. Note: *ILOS* bit value is updated to Initial Value only after LAN_PWR_GOOD or PCIe reset.
SPEED	9:8	10b	Speed selection. These bits determine the speed configuration and are written by software after reading the PHY configuration through the MDIO interface. These signals are ignored when Auto-Speed Detection is enabled. 00b = 10 Mb/s. 01b = 100 Mb/s. 10b = 1000 Mb/s. 11b = not used.
Reserved	10	0b	Reserved Write as 0b to ensure future compatibility.
FRCSPD	11	0b[1]	Force Speed This bit is set when software needs to manually configure the MAC speed settings according to the SPEED bits. Note that MAC and PHY must resolve to the same speed configuration or software must manually set the PHY to the same speed as the MAC. Software must clear this bit to enable the PHY or ASD function to control the MAC speed setting. Note that this bit is superseded by the *CTRL_EXT.SPD_BYPS* bit which has a similar function.
FRCDPLX	12	0b	Force Duplex When set to 1b, software can override the duplex indication from the PHY that is indicated in the FDX to the MAC. Otherwise, in 10/100/1000Base-T link mode, the duplex setting is sampled from the PHY FDX indication into the MAC on the asserting edge of the PHY LINK signal. When asserted, the *CTRL.FD* bit sets duplex.
Reserved	15:13	0x0	Reserved Write 0, ignore on read.
SDP0_GPIEN	16	0b	General Purpose Interrupt Detection Enable for SDP0 If software-controlled IO pin SDP0 is configured as an input, this bit (when 1b) enables the use for GPI interrupt detection.
SDP1_GPIEN	17	0b	General Purpose Interrupt Detection Enable for SDP1 If software-controlled IO pin SDP1 is configured as an input, this bit (when 1b) enables the use for GPI interrupt detection.

Field	Bit(s)	Initial Value	Description
SDP0 DATA (RWS)	18	0b[1]	SDP0 Data Value Used to read or write the value of software-controlled IO pin SDP0. If SDP0 is configured as an output (*SDP0_IODIR* = 1b), this bit controls the value driven on the pin (initial value EEPROM-configurable). If SDP0 is configured as an input, reads return the current value of the pin. When the SDP0_WDE bit is set, this field indicates the polarity of the watchdog indication.
SDP1 DATA (RWS)	19	0b[1]	SDP1 Data Value Used to read or write the value of software-controlled IO pin SDP1. If SDP1 is configured as an output (SDP1_IODIR = 1b), this bit controls the value driven on the pin (initial value EEPROM-configurable). If SDP1 is configured as an input, reads return the current value of the pin.
ADVD3WUC	20	1b[1]	D3Cold Wake up Capability Enable When bit is 0b PME (WAKE#) is not generated in D3Cold. Bit loaded from EEPROM (see Section 6.2.18).
SDP0_WDE	21	0b[1]	SDP0 used for Watchdog indication When set, SDP0 is used as a watchdog indication. When set, the SDP0_DATA bit indicates the polarity of the watchdog indication. In this mode, *SDP0_IODIR* must be set to an output.
SDP0_IODIR	22	0b[1]	SDP0 Pin Direction Controls whether software-controllable pin SDP0 is configured as an input or output (0b = input, 1b = output). Initial value is EEPROM-configurable. This bit is not affected by software or system reset, only by initial power-on or direct software writes.
SDP1_IODIR	23	0b[1]	SDP1 Pin Direction Controls whether software-controllable pin SDP1 is configured as an input or output (0b = input, 1b = output). Initial value is EEPROM-configurable. This bit is not affected by software or system reset, only by initial power-on or direct software writes.
Reserved	25:24	0b	Reserved.
RST (SC)	26	0b	Port Software Reset This bit performs reset to the respective port, resulting in a state nearly approximating the state following a power-up reset or internal PCIe reset, except for system PCI configuration and logic used by all ports. 0b = Normal. 1b = Reset. This bit is self clearing and is referred to as software reset or global reset.
RFCE	27	1b	Receive Flow Control Enable When set, indicates that the 82580 responds to the reception of flow control packets. If Auto-Negotiation is enabled, this bit is set to the negotiated flow control value. In SerDes mode the resolution is done by the hardware. In internal PHY, SGMII or 1000BASE-KX modes it should be done by the software.
TFCE	28	0b	Transmit Flow Control Enable When set, indicates that the 82580 transmits flow control packets (XON and XOFF frames) based on the receiver fullness. If Auto-Negotiation is enabled, this bit is set to the negotiated duplex value. In SerDes mode the resolution is done by the hardware. In internal PHY, SGMII or 1000BASE-KX modes it should be done by the software.

Field	Bit(s)	Initial Value	Description
DEV_RST (SC)	29	0b	Device Reset This bit performs a reset of the entire controller device, resulting in a state nearly approximating the state following a power-up reset or internal PCIe reset, except for system PCI configuration. 0b = Normal. 1b = Reset. This bit is self clearing. Notes: 1. Assertion of DEV_RST generates an interrupt on all ports via the ICR.DRSTA interrupt bit. 2. Device Reset (CTRL.DEV_RST) can be used to globally reset the entire component if the DEV_RST_EN bit in Initialization Control 4 EEPROM word is set. 3. Assertion of DEV_RST sets on all ports the *STATUS.DEV_RST_SET* bit. For additional information see Section 4.3.2.
VME	30	0b	VLAN Mode Enable When set to 1b, VLAN information is stripped from all received 802.1Q packets.
PHY_RST	31	0b	PHY Reset Generates a hardware-level reset to the internal 1000BASE-T PHY. 0b = Normal operation. 1b = Internal PHY reset asserted.

1. These bits are loaded from EEPROM.

8.2.2 Device Status Register - STATUS (0x0008; R)

Field	Bit(s)	Initial Value	Description
FD	0	X	Full Duplex. 0 = Half duplex (HD). 1= Full duplex (FD). Reflects duplex setting of the MAC and/or link. FD reflects the actual MAC duplex configuration. This normally reflects the duplex setting for the entire link, as it normally reflects the duplex configuration negotiated between the PHY and link partner (copper link) or MAC and link partner (fiber link).
LU	1	X	Link up. 0 = no link established; 1 = link established. For this to be valid, the Set Link Up bit of the Device Control Register (CTRL.SLU) must be set. Link up provides a useful indication of whether something is attached to the port. Successful negotiation of features/link parameters results in link activity. The link startup process (and consequently the duration for this activity after reset) can be several 100's of ms. When the internal PHY is used, this reflects whether the PHY's LINK indication is present. When the SerDes, SGMII or 1000BASE-KX interface is used, this indicates loss-of-signal; if Auto-Negotiation is also enabled, this can also indicate successful Auto-Negotiation. Refer to Section 3.5.4 for more details. Note: Bit is valid only when working in Internal PHY mode. In SerDes mode bit is always 0.
LAN ID	3:2	Port 0 = 00b Port 1 = 01b Port 2 = 10b Port 3 = 11b	LAN ID Provides software a mechanism to determine the LAN identifier for the MAC. 00b = LAN 0. 01b = LAN 1. 10b = LAN 2. 11b = LAN 3.
TXOFF	4	X	Transmission Paused This bit indicates the state of the transmit function when symmetrical flow control has been enabled and negotiated with the link partner. This bit is set to 1b when transmission is paused due to the reception of an XOFF frame. It is cleared (0b) upon expiration of the pause timer or the receipt of an XON frame.
Reserved	5	X	Reserved
SPEED	7:6	X	Link Speed Setting Reflects the speed setting of the MAC and/or link when it is operating in 10/100/1000BASE-T mode (internal PHY). When the MAC is operating in 10/100/1000BASE-T mode with the internal PHY, these bits normally reflect the speed of the actual link, negotiated by the PHY and link partner and reflected internally from the PHY to the MAC (*SPD_IND*). These bits also might represent the speed configuration of the MAC only, if the MAC speed setting has been forced via software (*CTRL.SPEED*) or if MAC auto-speed detection is used. If Auto-Speed Detection is enabled, the 82580's speed is configured only once after the LINK signal is asserted by the PHY. 00b = 10 Mb/s. 01b = 100 Mb/s. 10b = 1000 Mb/s. 11b = 1000 Mb/s.

Field	Bit(s)	Initial Value	Description
ASDV	9:8	X	Auto-Speed Detection Value Speed result sensed by the 82580's MAC auto-detection function. These bits are provided for diagnostics purposes only. The ASD calculation can be initiated by software writing a logic 1b to the *CTRL_EXT.ASDCHK* bit. The resultant speed detection is reflected in these bits. See Section 8.2.3 for details.
PHYRA	10	1b	PHY Reset Asserted This read/write bit is set by hardware following the assertion of an internal PHY reset; it is cleared by writing a 0b to it. This bit is also used by firmware indicating a required initialization of the 82580's PHY.
Reserved	18:11	0x0	Reserved
GIO Master Enable Status	19	1b	Cleared by the 82580 when the *CTRL.GIO Master Disable* bit is set and no master requests are pending by this function and is set otherwise. Indicates that no master requests are issued by this function as long as the *CTRL.GIO Master Disable* bit is set.
DEV_RST_SET (R/W1C)	20	0b	Device Reset Set When set indicates that a device reset (*CTRL.DEV_RST*) was initiated by one of the software drivers. Note: Bit cleared by write 1.
Reserved	30:21	0x0	Reserved
MAC clock gating Enable	31	1b[1]	MAC clock gating Enable bit loaded from the EEPROM- indicates the device support gating of the MAC clock.

[1] If the signature bits of the EEPROM's *Initialization Control Word 1* match (01b), this bit is read from the EEPROM.

8.2.3 Extended Device Control Register - CTRL_EXT (0x0018; R/W)

This register provides extended control of the 82580's functionality beyond that provided by the Device Control register (*CTRL*).

Field	Bit(s)	Initial Value	Description
Reserved	1:0	0b	Reserved. Should be written as 0b to ensure future compatibility.
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 set to 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 set to 1b) enables use for GPI interrupt detection.
Reserved	5:4	00b	Reserved. Reads as 00b.
SDP2_DATA	6	0b[1]	SDP2 Data Value. Used to read (write) the value of software-controllable IO pin SDP2. If SDP2 is configured as an output (SDP2_IODIR = 1b), this bit controls the value driven on the pin (initial value EEPROM-configurable). If SDP2 is configured as an input, reads return the current value of the pin.
SDP3_DATA	7	0b[1]	SDP3 Data Value. Used to read (write) the value of software-controllable IO pin SDP3. If SDP3 is configured as an output (SDP3_IODIR = 1b), this bit controls the value driven on the pin (initial value EEPROM-configurable). If SDP3 is configured as an input, reads return the current value of the pin.

Field	Bit(s)	Initial Value	Description
Reserved	9:8	0b[1]	Reserved Formally used as SDP5 and SDP4 pin input/output direction control, respectively.
SDP2_IODIR	10	0b[1]	SDP2 Pin Direction. Controls whether software-controllable pin SDP2 is configured as an input or output (0b = input, 1b = output). Initial value is EEPROM-configurable. This bit is not affected by software or system reset, only by initial power-on or direct software writes.
SDP3_IODIR	11	0b[1]	SDP3 Pin Direction. Controls whether software-controllable pin SDP3 is configured as an input or output (0b = input, 1b = output). Initial value is EEPROM-configurable. This bit is not affected by software or system reset, only by initial power-on or direct software writes.
ASDCHK	12	0b	ASD Check Initiates an Auto-Speed-Detection (ASD) sequence to sense the frequency of the PHY receive clock (RX_CLK). The results are reflected in STATUS.ASDV. This bit is self-clearing.
EE_RST	13	0b	EEPROM Reset When set, initiates a reset-like event to the EEPROM function. This causes the EEPROM to be read as if a RST# assertion had occurred. All the 82580 functions should be disabled prior to setting this bit. This bit is self-clearing.
Reserved	14	0b	Reserved
SPD_BYPS	15	0b	Speed Select Bypass When set to 1b, all speed detection mechanisms are bypassed, and the 82580 is immediately set to the speed indicated by *CTRL.SPEED*. This provides a method for software to have full control of the speed settings of the 82580 and when the change takes place, by overriding the hardware clock switching circuitry.
NS_DIS	16	0	No Snoop Disable When set to 1b, the 82580 does not set the no snoop attribute in any PCIe packet, independent of PCIe configuration and the setting of individual no snoop enable bits. When set to 0b, behavior of no snoop is determined by PCIe configuration and the setting of individual no snoop enable bits.
RO_DIS	17	0b	Relaxed Ordering Disabled When set to 1b, the 82580 does not request any relaxed ordering transactions on the PCIe interface regardless of the state of bit 4 in the *PCIe Device Control* register. When this bit is cleared and bit 4 of the *PCIe Device Control* register is set, the 82580 requests relaxed ordering transactions as specified by registers *RXCTL* and *TXCTL* (per queue and per flow).
SerDes Low Power Enable	18	0b[1]	When set, allows the SerDes to enter a low power state when the function is in Dr state.
Dynamic MAC Clock Gating	19	0b[1]	When set, enables Dynamic MAC Clock Gating.
PHY Power Down Enable	20	1b[1]	When set, enables the PHY to enter a low-power state as described in Section 5.4.3.
Reserved	21	0b	Reserved. Write 0, ignore on read.
LINK_MODE	23:22	0x0[1]	Link Mode Controls interface on the link. 00b = Direct copper (1000Base-T) interface (10/100/1000 BASE-T internal PHY mode). 01b = 1000BASE-KX. 10b = SGMII. 11b = SerDes interface.

Field	Bit(s)	Initial Value	Description
Reserved	24	0b	Reserved. Write 0, ignore on read.
I2C Enabled	25	0b[1]	Enable I2C This bit enables the SFPx_I2C pins that can be used to access external SFP modules or an external 1000BASE-T PHY via the MDIO interface. If cleared, the SFPx_I2C pads are isolated and accesses to the SFPx_I2C pins through the *I2CCMD* register or the *MDIC* register are ignored.
EXT_VLAN	26	0b[1]	External VLAN Enable When set, all incoming Rx packets are expected to have at least one VLAN with the Ether type as defined in *VET.EXT_VET* that should be ignored. The packets can have a second VLAN that should be used for all filtering purposes. All Tx packets are expected to have at least one VLAN added to them by the host. In the case of an additional VLAN request (*VLE* - VLAN Enable is set in transmit descriptor) the second VLAN is added after the first external VLAN is added by the host. This bit is reset only by a power up reset or by an EEPROM full auto load and should only be changed while Tx and Rx processes are stopped.
Reserved	27	0b	Reserved
DRV_LOAD	28	0b	Driver Loaded This bit should be set by the driver after it is loaded. This bit should be cleared when the driver unloads or after a PCIe soft reset. The MNG controller reads this bit to indicate to the manageability controller (BMC) that the driver has loaded.
Reserved	31:29	0b	Reserved Write 0, Ignore on read.

[1] These bits are read from the EEPROM.

The 82580 allows up to four externally controlled interrupts. All software-definable pins, these can be mapped for use as GPI interrupt bits. Mappings are enabled by the SDPx_GPIEN bits only when these signals are also configured as inputs via SDPx_IODIR. When configured to function as external interrupt pins, a GPI interrupt is generated when the corresponding pin is sampled in an active-high state.

The bit mappings are shown in the table bellow for clarity.

Table 8-8. Mappings for SDI Pins Used as GPI

SDP Pin Used as GPI	CTRL_EXT Field Settings		Resulting ICR Bit (GPI)
	Direction	Enable as GPI interrupt	
3	SDP3_IODIR	SDP3_GPIEN	14
2	SDP2_IODIR	SDP2_GPIEN	13
1	SDP1_IODIR	SDP1_GPIEN	12
0	SDP0_IODIR	SDP0_GPIEN	11

Note: If software uses the EE_RST function and desires to retain current configuration information, the contents of the control registers should be read and stored by software. Control register values are changed by a read of the EEPROM which occurs upon assertion of the EE_RST bit.

Note: The EEPROM reset function can read configuration information out of the EEPROM which affects the configuration of PCIe space BAR settings. The changes to the BARs are not visible unless the system reboots and the BIOS is allowed to re-map them.

The SPD_BYPS bit performs a similar function to the CTRL.FRCSPD bit in that the 82580's speed settings are determined by the value software writes to the CRTL.SPEED bits. However, with the SPD_BYPS bit asserted, the settings in CTRL.SPEED take effect immediately rather than waiting until after the 82580's clock switching circuitry performs the change.

8.2.4 MDI Control Register - MDIC (0x0020; R/W)

Software uses this register to read or write Management Data Interface (MDI) registers in the internal PHY or an external SGMII PHY.

See Section 3.5.2.2.2 for details on usage of this register. The PHY registers described in Section 8.24 are accessible using the MDIC register.

Field	Bit(s)	Initial Value	Description
DATA	15:0	X	Data In a Write command, software places the data bits and the MAC shifts them out to the PHY. In a Read command, the MAC reads these bits serially from the PHY and software can read them from this location.
REGADD	20:16	0x0	PHY Register Address: Reg. 0, 1, 2,...31
Reserved	25:21	0x0	Reserved.
OP	27:26	0x0	Opcode 01b = MDI Write 10b = MDI Read All other values are reserved.
R (RWS)	28	1b	Ready Bit Set to 1b by the 82580 at the end of the MDI transaction (for example, indication of a Read or Write completion). It should be reset to 0b by software at the same time the command is written.
MDI_IE	29	0b	Interrupt Enable When set to 1 an Interrupt is generated at the end of an MDI cycle to indicate an end of a read or write operation to the PHY.
MDI_ERR (RWS)	30	0b	Error This bit is set to 1b by hardware when it fails to complete an MDI read. Software should make sure this bit is clear (0b) before issuing an MDI read or write command. Note: bit is valid only when the Ready bit is set.
Reserved	31	0b	Reserved.

8.2.5 MDC/MDIO Configuration Register – MDICNFG (0x0E04; R/W)

This register is used to configure the MDIO connection that is accessed via the *MDIC* register. See Section 3.5.2.2.2 for details on usage of this register.

Field	Bit(s)	Initial Value	Description
Reserved	20:0	0x0	Reserved. Write 0, ignore on read.
PHYADD[1]	25:21	0x00 - LAN 0 0x01 - LAN 1 0x02 - LAN 2 0x03 - LAN 3	External PHY Address When *MDICNFG.Destination* bit is 0b, default PHYADD accesses internal PHY.
Reserved	29:26	0x0	Reserved. Write 0, ignore on read.
Com_MDIO[2]	30	0b	Common MDIO When this bit is set, access to the MDIO interface on this port is routed to the LAN 0 MDIO interface. In this case the LAN 0 MDIO interface is used as a common interface for MDIO accesses to a Multi PHY device that has a single MDIO interface for all PHYs. 0b - MDIO access routed to the dedicated LAN port MDIO interface. 1b - MDIO accesses on this LAN port are routed to the LAN 0 MDIO interface
Destination[3]	31	0b	Destination 0b = The MDIO transaction is to the internal PHY. 1b = The MDIO transaction is directed to the external MDIO pins (I^2C Interface). Note: • When PHY registers access is initiated via the I2CCMD interface, access is always via the external I^2C Interface. In this case the destination field should always be 0.

1. PHYADD Loaded from *Initialization Control 4* EEPROM word to allocate per port address when using external MDIO port.
2. Common MDIO usage configuration bit is loaded from *Initialization Control 3* EEPROM word.
3. Destination Loaded from EEPROM Initialization Control 3 word. When external PHY supports a MDIO interface bit is 1, otherwise bit is 0.

8.2.6 SERDES Control 0 - P1GCTRL0 (0x0E08; RW)

Field	Bit(s)	Initial Value	Description
Reserved	0	0b	Reserved.

Field	Bit(s)	Initial Value	Description
LPBKBUF	1	0b	External SerDes LoopBack 0b - Normal operation. 1b - Enable External SerDes loopback (RX to TX).
Reserved	3:2	11b	Reserved. Write 11b.
Reserved	31:4	0x0	Reserved.

8.2.7 Copper/Fiber Switch Control - CONNSW (0x0034; R/W)

Field	Bit(s)	Initial Value	Description
AUTOSENSE_EN	0	0b	Auto Sense Enable When set, the auto sense mode is active. In this mode the non-active link is sensed by hardware as follows PHY Sensing: The electrical idle detector of the receiver of the PHY is activated while in SerDes, SGMII or 1000BASE-KX mode. SerDes sensing: The electrical idle detector of the receiver of the SerDes is activated while in internal PHY mode, assuming the ENRGSRC bit is cleared If energy is detected in the non active media, the OMED bit in the ICR register is set and this bit is cleared. This includes the case where energy was present at the non-active media when this bit is being set.
AUTOSENSE_CONF	1	0b	Auto Sense Configuration Mode This bit should be set during the configuration of the PHY/SerDes towards the activation of the auto-sense mode to avoid spurious interrupts. While this bit is set, the PHY/SerDes is active even though the active link is set to SerDes, 1000BASE-KX or SGMII/PHY. Energy detection while this bit is set is not reflected to the *ICR.OMED* interrupt.
ENRGSRC	2	0b[1]	SerDes Energy Detect Source If set, the OMED interrupt cause is set after asserting the external signal detect pin. If cleared, the OMED interrupt cause is set after exiting from electrical idle of the SerDes receiver. This bit also defines the source of the signal detect indication used to set link up while is SerDes mode.
ASCLR_DIS	3	0b	Reserved
Reserved	8:4	0x0	Reserved
SerDesD (RO)	9	X	SerDes Signal Detect Indication Indicates the SerDes signal detect value according to the selected source (either external or internal). Valid only if LINK_MODE is SerDes, 1000BASE-KX or SGMII.
PHYSD (RO)	10	X	PHY Signal Detect Indication Valid only if LINK_MODE is the PHY and the receiver is not in electrical idle.
PHY_PDN (RO)	11	X	This bit indicates that the internal GbE PHY is in power down state. 0 = Internal GbE PHY not in power down. 1 = Internal GbE PHY in power down.
Reserved	31:12	0x0	Reserved

[1] The default value of the ENRGSRC bit in this register is defined in the *Initialization Control 3* (Offset 0x24) EEPROM word (bit 15).

8.2.8 VLAN Ether Type - VET (0x0038; R/W)

This register contains the type field 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.

Field	Bit(s)	Initial Value	Description
VET (RO)	15:0	0x8100	VLAN EtherType Should be programmed with 0x8100.
VET EXT	31:16	0x8100	External VLAN Ether Type.

8.2.9 LED Control - LEDCTL (0x0E00; RW)

This register controls the setup of the LEDs. See Section 7.5.1 for details of the MODE fields encoding.

Field	Bit(s)	Initial Value	Description
LED0_MODE	3:0	0010b[1]	LED0/LINK# Mode This field specifies the control source for the LED0 output. An initial value of 0010b selects LINK_UP# indication.
LED_PCI_MODE	4	0b	0b = Use LEDs as defined in the other fields of this register. 1b = Use LEDs to indicate PCI-E Lanes Idle status in SDP mode (only when the led_mode is set to 0x8 – SDP mode) For Port 0 LED0 3-0 indicates RX lanes 3- 0 Electrical Idle status For Port 1 LED1 3-0 indicates TX lanes 3- 0 Electrical Idle status
GLOBAL_BLINK_MODE	5	0b[1]	Global Blink Mode This field specifies the blink mode of all the LEDs. 0b = Blink at 200 ms on and 200 ms off. 1b = Blink at 83 ms on and 83 ms off.
LED0_IVRT	6	0b[1]	LED0/LINK# Invert This field specifies the polarity/ inversion of the LED source prior to output or blink control. 0b = Do not invert LED source (LED active low). 1b = Invert LED source (LED active High).
LED0_BLINK	7	0b[1]	LED0/LINK# Blink This field specifies whether to apply blink logic to the (possibly inverted) LED control source prior to the LED output. 0b = Do not blink asserted LED output. 1b = Blink asserted LED output.
LED1_MODE	11:8	0011b[1]	LED1/ACTIVITY# Mode This field specifies the control source for the LED1 output. An initial value of 0011b selects FILTER ACTIVITY# indication.
Reserved	12	0b	Reserved Write as 0 ignore on read.
Reserved	13	0b	Reserved Write 0 ignore on read.

Field	Bit(s)	Initial Value	Description
LED1_IVRT	14	0b[1]	LED1/ACTIVITY# Invert This field specifies the polarity/ inversion of the LED source prior to output or blink control. 0b = Do not invert LED source (LED active low). 1b = Invert LED source (LED active High).
LED1_BLINK	15	1b[1]	LED1/ACTIVITY# Blink
LED2_MODE	19:16	0110b[1]	LED2/LINK100# Mode This field specifies the control source for the LED2 output. An initial value of 0011b selects LINK100# indication.
Reserved	20	0b	Reserved Read-only as 0b. Write as 0b for future compatibility.
Reserved	21	0b	Reserved
LED2_IVRT	22	0b[1]	LED2/LINK100# Invert This field specifies the polarity/ inversion of the LED source prior to output or blink control. 0b = Do not invert LED source (LED active low). 1b = Invert LED source (LED active High).
LED2_BLINK	23	0b[1]	LED2/LINK100# Blink
LED3_MODE	27:24	0111b[1]	LED3/LINK1000# Mode This field specifies the control source for the LED3 output. An initial value of 0111b selects LINK1000# indication.
Reserved	28	0b	Reserved Read-only as 0b. Write as 0b for future compatibility.
Reserved	29	0b	Reserved
LED3_IVRT	30	0b[1]	LED3/LINK1000# Invert This field specifies the polarity/ inversion of the LED source prior to output or blink control. 0b = Do not invert LED source (LED active low). 1b = Invert LED source (LED active High).
LED3_BLINK	31	0b[1]	LED3/LINK1000# Blink

[1] These bits are read from the EEPROM.

8.3 Internal Packet Buffer Size Registers

The following registers define the size of the on-chip receive and transmit buffers used to receive and transmit packets. The overall available internal buffer size in the 82580 for all ports is 144 KB for receive buffers and 80 KB for transmit Buffers. Disabled ports memory can be shared between active ports and sharing can be asymmetric. The default buffer size for each port is loaded from the EEPROM on initialization.

8.3.1 Internal Receive Packet Buffer Size - IRPBS (0x2404; RO)

Field	Bit(s)	Initial Value	Description
RXPbsize[1]	3:0	0x0	Receive internal buffer size: 0x0 - 36 KB 0x1 - 72 KB 0x2 - 144 KB 0x3 - 1 KB 0x4 - 2 KB 0x5 - 4 KB 0x6 - 8 KB 0x7 - 16 KB 0x8 - 35 KB 0x9 - 70 KB 0xA - 140 KB 0xB:0xF - reserved Notes: 1. When 4 ports are active maximum buffer size can be 36 KB. When 2 ports are active maximum buffer size can be 72 KB. When only a single port is active maximum buffer size can be 144 KB. For further information see Section 7.1.3.2. 2. Values bellow 35 KB should be used for diagnostic purposes only. 3. When port is disabled for both PCIe and Management access, the buffer size allocated to the port is 0 Bytes. Available internal memory can be used by other ports.
Reserved	31:4	0x0	Reserved

[1] Value loaded from *Initialization Control 4* EEPROM word.

8.3.2 Internal Transmit Packet Buffer Size - ITPBS (0x3404; RO)

Field	Bit(s)	Initial Value	Description
TXPbsize[1]	3:0	0x0	Transmit internal buffer size: 0x0 - 20 KB 0x1 - 40 KB 0x2 - 80 KB 0x3 - 1 KB 0x4 - 2 KB 0x5 - 4 KB 0x6 - 8 KB 0x7 - 16 KB 0x8 - 19 KB 0x9 - 38 KB 0xA - 76 KB 0xB:0xF - reserved Notes: 1. When 4 ports are active maximum buffer size can be 20KB. When only 2 ports are active maximum buffer size is 40KB. When only a single port is active maximum buffer size is 80KB. For further information see Section 7.2.1.2. 2. Values bellow 20 KB should be used for diagnostic purposes only. 3. When port is disabled for both PCIe and management access, the buffer size allocated to the port is 0 Bytes. Available internal memory can be used by other ports.
Reserved	31:4	0x0	Reserved

[1] Value loaded from Initialization Control 4 EEPROM word.

8.4 EEPROM/Flash Register Descriptions

8.4.1 EEPROM/Flash Control Register - EEC (0x0010; R/W)

This register provides software direct access to the EEPROM. Software can control the EEPROM by successive writes to this register. Data and address information is clocked into the EEPROM by software toggling the EE_SK and EE_DI bits (0 and 2) of this register with EE_CS set to 0b. Data output from the EEPROM is latched into the EE_DO bit (bit 3) via the internal 62.5 MHz clock and can be accessed by software via reads of this register.

Note: Attempts to write to the Flash device via PCIe BAR or via I/O access when writes are disabled (FWE is not equal to 10b) should not be attempted. Behavior after such an operation is undefined and can result in component and/or system hangs. Bit banging access to the flash via FLA register is not protected by this field.

Field	Bit(s)	Initial Value	Description
EE_SK	0	0b	Clock input to the EEPROM When EE_GNT = 1b, the EE_SK output signal is mapped to this bit and provides the serial clock input to the EEPROM. Software clocks the EEPROM via toggling this bit with successive writes.
EE_CS	1	0b	Chip select input to the EEPROM When EE_GNT = 1b, the EE_CS output signal is mapped to the chip select of the EEPROM device. Software enables the EEPROM by writing a 1b to this bit.
EE_DI	2	0b	Data input to the EEPROM When EE_GNT = 1b, the EE_DI output signal is mapped directly to this bit. Software provides data input to the EEPROM via writes to this bit.
EE_DO (RO)	3	X[1]	Data output bit from the EEPROM The EE_DO input signal is mapped directly to this bit in the register and contains the EEPROM data output. This bit is RO from a software perspective; writes to this bit have no effect.
FWE	5:4	01b	Flash Write Enable Control These two bits, control whether writes to Flash memory are allowed. 00b = Flash erase (along with bit 31 in the FLA register). 01b = Flash writes disabled. 10b = Flash writes enabled. 11b = Reserved.
EE_REQ	6	0b	Request EEPROM Access The software must write a 1b to this bit to get direct EEPROM access. It has access when EE_GNT is 1b. When the software completes the access it must write a 0b.
EE_GNT	7	0b	Grant EEPROM Access When this bit is 1b the software can access the EEPROM using the SK, CS, DI, and DO bits.
EE_PRES (RO)	8	X	EEPROM Present This bit indicates that an EEPROM is present by monitoring the EE_DO input for an active-low acknowledge by the serial EEPROM during initial EEPROM scan. 1b = EEPROM present.
Auto_RD (RO)	9	0b	EEPROM Auto Read Done When set to 1b, this bit indicates that the auto read by hardware from the EEPROM is done. This bit is also set when the EEPROM is not present or when its signature is not valid.
EE_ADDR_SIZE	10	0b	EEPROM Address Size This field defines the address size of the EEPROM. This bit is set by the EEPROM size auto-detect mechanism. If no EEPROM is present or the signature is not valid, a 16-bit address is assumed. 0b = 8- and 9-bit. 1b = 16-bit.

Field	Bit(s)	Initial Value	Description
EE_SIZE (RO)	14:11[2]	0010b	EEPROM Size This field defines the size of the EEPROM: Field Value EEPROM Size EEPROM Address Size 0000b - 0110b Reserved 0111b 16 Kbytes 2 bytes 1000b 32 Kbytes 2 bytes 1001b - 1111b Reserved
EE_BLOCKED (R/W1C)	15	0b	EEPROM access blocked EEPROM access blocked - Bit is set by HW when bit banging transactions are blocked due to write to read-only sections or any access to hidden area. Note: Bit is cleared by write one.
Reserved	31:16	0x0	Reserved Write 0 ignore on read.

1. Value depends on voltage level on EE_DO pin following initialization
2. These bits are read from the EEPROM.

8.4.2 EEPROM Read Register - EERD (0x0014; RW)

This register is used by software to cause the 82580 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 82580 reads the word from the EEPROM and places it in the *Read Data* field, setting the *Read Done* field to 1b. Software can poll this register, looking for a 1b in the *Read Done* field, and then using the value in the *Read Data* field.

When this register is used to read a word from the EEPROM, that word does not influence any of the 82580's internal registers even if it is normally part of the auto-read sequence.

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 storing the result in the EE_DATA field. This bit is self-clearing.
DONE (RO)	1	0b	Read Done Set to 1b when the EEPROM read completes. Set to 0b when the EEPROM read is not completed. Writes by software are ignored. Reset by setting the START bit.
ADDR	15:2	0x0	Read Address This field is written by software along with *Start Read* to indicate the word to read.
DATA (RO)	31:16	X	Read Data. Data returned from the EEPROM read.

8.4.3 Flash Access - FLA (0x001C; R/W)

This register provides software direct access to the Flash. Software can control the Flash by successive writes to this register. Data and address information is clocked into the Flash by software toggling the FL_SCK bit (bit 0) of this register with FL_CE set to 1b. Data output from the Flash is latched into the FL_SO bit (bit 3) of this register via the internal 125 MHz clock and can be accessed by software via reads of this register.

Field	Bit(s)	Initial Value	Description
FL_SCK	0	0b	Clock Input to the Flash When FL_GNT is 1b, the FL_SCK out signal is mapped to this bit and provides the serial clock input to the Flash device. Software clocks the Flash memory via toggling this bit with successive writes.
FL_CE	1	0b	Chip Select Input to the Flash When FL_GNT is 1b, the FL_CE output signal is mapped to the chip select of the Flash device. Software enables the Flash by writing a 0b to this bit.
FL_SI	2	0b	Data Input to the Flash When FL_GNT is 1b, the FL_SI output signal is mapped directly to this bit. Software provides data input to the Flash via writes to this bit.
FL_SO	3	X	Data Output Bit from the Flash The FL_SO input signal is mapped directly to this bit in the register and contains the Flash memory serial data output. This bit is read only from the software perspective — writes to this bit have no effect.
FL_REQ	4	0b	Request Flash Access The software must write a 1b to this bit to get direct Flash memory access. It has access when FL_GNT is 1b. When the software completes the access it must write a 0b.
FL_GNT	5	0b	Grant Flash Access When this bit is 1b, the software can access the Flash memory using the FL_SCK, FL_CE, FL_SI, and FL_SO bits.
FLA_add_size	6	0b	Flash Address Size When Flash_add_size is set, all flashes (including 64 KB) are accessed using 3 bytes of the address. If this bit is set by one of the functions, it is also reflected in the other one.
Reserved	29:7	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 memory is in progress. While this bit is clear (read as 0b) software can access to write a new byte to the Flash device.
FL_ER (SC)	31	0b	Flash Erase Command When bit is set to 1b an erase command is sent to the Flash component only if the *EEC.FWE* field is 00b (Flash Erase). This bit is automatically cleared if set to 1b and the value of the *EEC.FWE* field is 00b.

8.4.4 Flash Opcode - FLASHOP (0x103C; R/W)

This register enables the host or the firmware to define the op-code used in order to erase a sector of the flash or the complete flash. This register is reset only at power on or assertion.

This register is common to all ports and manageability. Register should be programmed according to the parameters of the flash used.

Note: The default values fit to Atmel* Serial Flash Memory devices.

Field	Bit(s)	Initial Value	Description
DERASE	7:0	0x0062	Flash Device Erase Instruction The op-code for the Flash erase instruction.
SERASE	15:8	0x0052	Flash Block Erase Instruction The op-code for the Flash block erase instruction. Relevant only to Flash access by manageability.
Reserved	31:16	0x0	Reserved

8.4.5 EEPROM Auto Read Bus Control - EEARBC (0x1024; R/W)

In EEPROM-less implementations, this register is used to program the 82580 the same way it should be programmed if an EEPROM was present. See Section 3.3.1.7.1 for details of this register usage.

This register is common to all functions and should be accessed only following access coordination with the other ports.

Field	Bit(s)	Initial Value	Description
VALID_CORE0	0	0b	Valid Write Active to Core 0 Write strobe to Core 0. Firmware/software sets this bit for write access to registers loaded from EEPROM words in LAN0 section. Software should clear this bit to terminate the write transaction.
VALID_CORE1	1	0b	Valid Write Active to Core 1 Write strobe to Core 1. Firmware/software sets this bit for write access to registers loaded from EEPROM words in LAN1 section. Software should clear this bit to terminate the write transaction.
VALID_COMMON	2	0b	Valid Write Active to Common Write strobe to Common. Firmware/software sets this bit for write access to registers loaded from EEPROM words that are common to all sections. Software should clear this bit to terminate the write transaction.
Reserved	3	0b	Reserved Reads as 0b.
ADDR	12:4	0x0	Write Address This field specifies the address offset of the EEPROM word from the start of the EEPROM Section. Sections supported are: • Common and LAN0 • LAN1 • LAN2 • LAN3
VALID_CORE2	13	0b	Valid Write Active to Core 2 Write strobe to Core 2. Firmware/software sets this bit for write access to registers loaded from EEPROM words in LAN2 section. Software should clear this bit to terminate the write transaction.
VALID_CORE3	14	0b	Valid Write Active to Core 3 Write strobe to Core 3. Firmware/software sets this bit for write access to registers loaded from EEPROM words in LAN3 section. Software should clear this bit to terminate the write transaction.
Reserved	15	0b	Reserved Write 0 ignore on read.
DATA	31:16	0x0	Data written into the EEPROM auto read bus.

1. Not all EEPROM addresses are part of the auto read. By using this register software can write to the hardware registers that are configured during auto read only.
2. Write access to this register is enabled only if EEPROM presence is not detected.

8.4.6 VPD diagnostic register -VPDDIAG (0x1060; RO)

This register stores the VPD parameters as parsed by the auto-load process. This register is used for debug only.

Field	Bit(s)	Initial Value	Description
Valid	0	X	VPD structure valid
Reserved	4:1	X	Reserved
RD Tag	13:5	X	Offset of the Read tag in VPD relative to the start of VPD (in bytes).
WR Tag	22:14	X	Offset of the Write tag in VPD relative to the start of VPD (in bytes).
End Tag	31:23	X	Offset of the End tag in VPD relative to the start of VPD (in bytes).

8.4.7 Management-EEPROM CSR I/F

The following registers are reserved for Firmware access to the EEPROM and are not writable by the host.

8.4.7.1 Management EEPROM Control Register - EEMNGCTL (0x1010; RO)

Field	Bit(s)	Initial Value	Description
ADDR	14:0	0x0	Address - This field is written by MNG along with Start Read or Start write to indicate the 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 tells the EEPROM if the current operation is read or write: 0b = read 1b = write
EEBUSY	17	0b	EEPROM Busy - This bit indicates that the EEPROM is busy processing an EEPROM transaction and shouldn’t be accessed.
CFG_DONE 0[1]	18	0b	Configuration cycle is done for port 0 – This bit indicates that configuration cycle (configuration of SerDes, PHY, PCIe and PLLs) is done for port 0. This bit is set to 1b to indicate configuration done, and cleared by hardware on any of the reset sources that causes initialization of the PHY. Note: Port 0 driver should not try to access the PHY for configuration before this bit is set.
CFG_DONE 1[1]	19	0b	Configuration cycle is done for port 1 – This bit indicates that configuration cycle (configuration of SerDes, PHY, PCIe and PLLs) is done for port 1. This bit is set to 1b to indicate configuration done, and cleared by hardware on any of the reset sources that cause initialization of the PHY. Note: Port 1 driver should not try to access the PHY for configuration before this bit is set. .
CFG_DONE 2[1]	20	0b	Configuration cycle is done for port 2 – This bit indicates that the configuration cycle (configuration of SerDes, PCIe and PLLs) is done for port 2. This bit is set to 1b to indicate configuration done, and cleared by hardware on any of the reset sources that cause initialization of the PHY. Note: Port 2 driver should not try to access the PHY for configuration before this bit is set.

CFG_DONE 3[1]	21	0b	Configuration cycle is done for port 3 – This bit indicates that the configuration cycle (configuration of SerDesPHY, PCIe and PLLs) is done for port 3. This bit is set to 1b to indicate configuration done, and cleared by hardware on any of the reset sources that cause initialization of the PHY. Note: Port 3 driver should not try to access the PHY for configuration before this bit is set.
Reserved	30:22	0x0	Reserved
DONE	31	1b	Transaction Done - This bit is cleared after Start Write or Start Read bit is set by the MNG and is set back again when the EEPROM write or read transaction is done.

[1] Bit relates to physical port. If LAN Function Swap (*FACTPS.LAN Function Sel* = 1) is done, Software should poll CFG_DONE bit of original port to detect end of PHY configuration operation.

8.4.7.2 Management EEPROM Read/Write data - EEMNGDATA (0x1014; RO)

Field	Bit(s)	Initial Value	Description
WRDATA	15:0	0x0	Write Data Data to be written to the EEPROM.
RDDATA	31:16	–	Read Data Data returned from the EEPROM read.

8.5 Flow Control Register Descriptions

8.5.1 Flow Control Address Low - FCAL (0x0028; RO)

Flow control packets are defined by 802.3X to be either a unique multicast address or the station address with the Ether Type field indicating PAUSE. The FCA registers provide the value hardware uses to compare incoming packets against, to determine that it should PAUSE its output.

The FCAL register contains the lower bits of the internal 48-bit Flow Control Ethernet address. All 32 bits are valid. Software can access the High and Low registers as a register pair if it can perform a 64-bit access to the PCIe bus. The complete flow control multicast address is: 0x01_80_C2_00_00_01; where 0x01 is the first byte on the wire, 0x80 is the second, etc.

Note: Any packet matching the contents of {*FCAH*, *FCAL*, *FCT*} when *CTRL.RFCE* is set is acted on by the 82580. Whether flow control packets are passed to the host (software) depends on the state of the *RCTL.DPF* bit and whether the packet matches any of the normal filters.

Field	Bit(s)	Initial Value	Description
FCAL	31:0	0x00C28001	Flow Control Address Low

8.5.2 Flow Control Address High - FCAH (0x002C; RO)

This register contains the upper bits of the 48-bit Flow Control Ethernet address. Only the lower 16 bits of this register have meaning. The complete Flow Control address is {*FCAH*, *FCAL*}.

The complete flow control multicast address is: 0x01_80_C2_00_00_01; where 0x01 is the first byte on the wire, 0x80 is the second, etc.

Field	Bit(s)	Initial Value	Description
FCAH	15:0	0x0100	Flow Control Address High Should be programmed with 0x01_00.
Reserved	31:16	0x0	Reserved Write 0 ignore on read.

8.5.3 Flow Control Type - FCT (0x0030; R/W)

This register contains the type field that hardware matches to recognize a flow control packet. Only the lower 16 bits of this register have meaning. This register should be programmed with 0x88_08. The upper byte is first on the wire *FCT*[15:8].

Field	Bit(s)	Initial Value	Description
FCT	15:0	0x8808	Flow Control Type
Reserved	31:16	0x0	Reserved Write 0 ignore on read.

8.5.4 Flow Control Transmit Timer Value - FCTTV (0x0170; R/W)

The 16-bit value in the *TTV* field is inserted into a transmitted frame (either XOFF frames or any PAUSE frame value in any software transmitted packets). It counts in units of slot time of 64 bytes. If software needs to send an XON frame, it must set *TTV* to 0 prior to initiating the PAUSE frame.

Field	Bit(s)	Initial Value	Description
TTV	15:0	X	Transmit Timer Value
Reserved	31:16	0b	Reserved Write 0 ignore on read.

8.5.5 Flow Control Receive Threshold Low - FCRTL0 (0x2160; R/W)

This register contains the receive threshold used to determine when to send an XON packet The complete register reflects the threshold in units of bytes. The lower 4 bits must be programmed to 0b (16 byte granularity). Software must set *XONE* to enable the transmission of XON frames. Each time hardware crosses the receive-high threshold (becoming more full), and then crosses the receive-low threshold and *XONE* is enabled (1b), hardware transmits an XON frame. When XONE is set, the *RTL* field should be programmed to at least 1b (at least 16 bytes).

Flow control reception/transmission are negotiated capabilities by the Auto-Negotiation process. When the 82580 is manually configured, flow control operation is determined by the *CTRL.RFCE* and *CTRL.TFCE* bits.

Field	Bit(s)	Initial Value	Description
Reserved	3:0	0000b	Reserved Write 0 ignore on read.
RTL	16:4	0x0	Receive Threshold Low. FIFO low water mark for flow control transmission. An XON packet is sent if the occupied space in the packet buffer is smaller or equal than this watermark. This field is in 16 bytes granularity.
Reserved	30:17	0x0	Reserved Write 0 ignore on read.
XONE	31	0b	XON Enable 0b = Disabled. 1b = Enabled.

8.5.6 Flow Control Receive Threshold High - FCRTH0 (0x2168; R/W)

This register contains the receive threshold used to determine when to send an XOFF packet. The complete register reflects the threshold in units of bytes. This value must be at maximum 48 bytes less than the maximum number of bytes allocated to the Receive Packet Buffer (*IRPBS.RXPbsize*), and the lower 4 bits must be programmed to 0b (16 byte granularity). The value of *RTH* should also be bigger than *FCRTL.RTL*. Each time the receive FIFO reaches the fullness indicated by *RTH*, hardware transmits a PAUSE frame if the transmission of flow control frames is enabled.

Flow control reception/transmission are negotiated capabilities by the Auto-Negotiation process. When the 82580 is manually configured, flow control operation is determined by the *CTRL.RFCE* and *CTRL.TFCE* bits.

Field	Bit(s)	Initial Value	Description
Reserved	3:0	000b	Reserved Write 0 ignore on read.
RTH	17:4	0x0	Receive Threshold High FIFO high water mark for flow control transmission. An XOFF packet is sent if the occupied space in the packet buffer is bigger or equal than this watermark. This field is in 16 bytes granularity. See Section 3.5.5.3.1 for calculation of *FCRTH0.RTH* value. Note: When in DMA coalescing operation and internal Transmit buffer is empty threshold high value defined in *FCRTC.RTH_Coal* is used instead of the *FCRTH0.RTH* value to allow increase of Receive Threshold High value by maximum supported Jumbo frame size.
Reserved	31:18	0x0	Reserved Write 0 ignore on read.

8.5.7 Flow Control Refresh Threshold Value - FCRTV (0x2460; R/W)

Field	Bit(s)	Initial Value	Description
FC_refresh_th	15:0	0x0	Flow Control Refresh Threshold This value indicates the threshold value of the flow control shadow counter; when the counter reaches this value, and the conditions for PAUSE state are still valid (buffer fullness above low threshold value), a PAUSE (XOFF) frame is sent to link partner. If this field contains zero value, the Flow Control Refresh is disabled.
Reserved	31:16	-	Reserved

8.5.8 Flow Control Status - FCSTS0 (0x2464; RO)

This register describes the status of the flow control machine.

Field	Bit(s)	Initial Value	Description
Flow_control state	0	0b	Flow control state machine signal 0b = XON 1b = XOFF
Above high	1		The size of data in the memory is above the high threshold
Below low	2		The size of data in the memory is below the low threshold
Reserved	15:3	0x0	Reserved
Refresh counter	31:16	0x0	Flow control refresh counter

8.6 PCIe Register Descriptions

8.6.1 PCIe Control - GCR (0x5B00; RW)

Register common to all functions.

Field	Bit(s)	Initial Value	Description
Reserved	1:0	0x0	Reserved. Write 0, ignore on read.
Discard on BME de-assert	2	1b	When set, on BME fall, the PCIE discards all requests of this function
Reserved	4:3	0x0	Reserved. Write 0, ignore on read.
Normal Completion on Reset	5	0b	When set on port reset (*CTRL.RST*), the PCIE will not discard pending requests of the function, but will wait for normal completion.
Reserved	8:6	0x0	Reserved. Write 0, ignore on read.

Field	Bit(s)	Initial Value	Description
Completion Timeout resend enable	9	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. Note: This field is loaded from the "Completion Timeout Resend" bit in the EEPROM.
Reserved	10	0b	Reserved
Number of resends	12:11	11b	The number of resends in case of Timeout or Poisoned.
Reserved	17:13	0x0	Reserved
PCIe Capability Version	18	1b[2]	Reports the PCIe capability version supported. 0b = Capability version: 0x1. 1b = Capability version: 0x2.
Reserved	30:19	0x0	Reserved.
DEV_RST in progress	31	0b	Device reset in progress Bit is set following Device Reset assertion (*CTRL.DEV_RST* = 1) until no pending requests exist in PCI-E. Software driver should wait for bit to be cleared before re-initializing the port (See Section 4.3.1).

1. Loaded from *PCIe Completion Timeout Configuration* EEPROM word (word 0x15).
2. The default value for this field is read from the *PCIe Init Configuration 3* EEPROM word (address 0x1A) bits 11:10. If these bits are set to 10b, then this field is set to 1, otherwise field is reset to zero.

8.6.2 PCIe Statistics Control #1 - GSCL_1 (0x5B10; RW)

Field	Bit(s)	Initial Value	Description
GIO_COUNT_EN_0	0	0b	Enable PCIe Statistic Counter Number 0.
GIO_COUNT_EN_1	1	0b	Enable PCIe Statistic Counter Number 1.
GIO_COUNT_EN_2	2	0b	Enable PCIe Statistic Counter Number 2.
GIO_COUNT_EN_3	3	0b	Enable PCIe Statistic Counter Number 3.
LBC Enable 0	4	0b	When set, statistics counter 0 operates in Leaky Bucket mode.
LBC Enable 1	5	0b	When set, statistics counter 1 operates in Leaky Bucket mode.
LBC Enable 2	6	0b	When set, statistics counter 2 operates in Leaky Bucket mode.
LBC Enable 3	7	0b	When set, statistics counter 3 operates in Leaky Bucket mode.
Reserved	26:8	0b	Reserved.
GIO_COUNT_TEST	27	0b	Test Bit Forward counters for testability.
GIO_64_BIT_EN	28	0b	Enable two 64-bit counters instead of four 32-bit counters.
GIO_COUNT_RESET	29	0b	Reset indication of PCIe statistical counters.
GIO_COUNT_STOP	30	0b	Stop indication of PCIe statistical counters.
GIO_COUNT_START	31	0b	Start indication of PCIe statistical counters.

8.6.3 PCIe Statistics Control #2 - GSCL_2 (0x5B14; RW)

This register configures the events counted by the GSCN_0, GSCN_1, GSCN_2 and GSCN_3 counters.

Field	Bit(s)	Initial Value	Description
GIO_EVENT_NUM_0	7:0	0x0	Event type that counter 0 (GSCN_0) counts.
GIO_EVENT_NUM_1	15:8	0x0	Event type that counter 1 (GSCN_1) counts.
GIO_EVENT_NUM_2	23:16	0x0	Event type that counter 2 (GSCN_2) counts.
GIO_EVENT_NUM_3	31:24	0x0	Event type that counter 3 (GSCN_3) counts.

Table 8-9 lists the encoding of possible event types counted by GSCN_0, GSCN_1, GSCN_2 and GSCN_3.

Table 8-9. PCIe Statistic Events Encoding

Transaction layer Events	Event Mapping (Hex)	Description
Bad TLP from LL	0x0	Each cycle, the counter increase in 1, if bad TLP is received (bad crc, error reported by AL, misplaced special char, reset in thI of received tlp).
Requests that reached timeout	0x10	Number of requests that reached Time Out.
NACK DLLP received	0x20	For each cycle, the counter increase by one, if a message was transmitted.
Replay happened in Retry-Buffer	0x21	Occurs when a replay happened due to timeout (not asserted when replay initiated due to NACK
Receive Error	0x22	Set when one of the following occurs: 1. Decoder error occurred during training in the PHY. It is reported only when training ends. 2. Decoder error occurred during link-up or till the end of the current packet (in case the link failed). This error is masked when entering/exiting EI.
Replay Roll-Over	0x23	Occurs when replay was initiated for more than 3 times [threshold is configurable by the PHY CSRs]
Re-Sending Packets	0x24	Occurs when TLP is resend in case of completion timeout
Surprise Link Down	0x25	Occurs when link is unpredictably down (Not because of reset or DFT)
LTSSM in L0s in both Rx & Tx	0x30	Occurs when LTSSM enters L0s state in both Tx & Rx
LTSSM in L0s in Rx	0x31	Occurs when LTSSM enters L0s state in Rx
LTSSM in L0s in Tx	0x32	Occurs when LTSSM enters L0s state in Tx
LTSSM in L1 active	0x33	Occurs when LTSSM enters L1-Active state (Requested from Host side)
LTSSM in L1 SW	0x34	Occurs when LTSSM enters L1-Switch (Requested from Switch side)
LTSSM in recovery	0x35	Occurs when LTSSM enters Recovery state

8.6.4 PCIe Statistic Control Register #5...#8 - GSCL_5_8 (0x5B90 + 4*n[n=0...3]; RW)

Note: This register is shared for all LAN ports.

These registers control the operation of the statistical counters GSCN_0, GSCN_1, GSCN_2 and GSCN_3 when operating Leaky Bucket mode:

- GSCL_5 controls operation of GSCN_0.
- GSCL_6 controls operation of GSCN_1.
- GSCL_7 controls operation of GSCN_2.
- GSCL_8 controls operation of GSCN_3.

Note: There are no GSCL_3 and GSCL_4 registers.

Field	Bit(s)	Initial Value	Description
LBC threshold n	15:0	0x0	Threshold for the Leaky Bucket Counter n
LBC timer n	31:16	0x0	Time period between decrementing the value in Leaky Bucket Counter n.

8.6.5 PCIe Counter #0 - GSCN_0 (0x5B20; RC)

Counter is common for all functions.

Field	Bit(s)	Initial Value	Description
EVC	31:0	0x0	Event Counter. Type of event counted is defined by the *GSCL_2.GIO_EVENT_NUM_0* field. Count value does not wrap around and remains stuck at the maximum value of 0xFF...F. Value is cleared by read.

8.6.6 PCIe Counter #1 - GSCN_1 (0x5B24; RC)

Counter is common for all functions.

Field	Bit(s)	Initial Value	Description
EVC	31:0	0x0	Event Counter. Type of event counted is defined by the *GSCL_2.GIO_EVENT_NUM_1* field. Count value does not wrap around and remains stuck at the maximum value of 0xFF...F. Value is cleared by read.

8.6.7 PCIe Counter #2 - GSCN_2 (0x5B28; RC)

Counter is common for all functions.

Field	Bit(s)	Initial Value	Description
EVC	31:0	0x0	Event Counter. Type of event counted is defined by the *GSCL_2.GIO_EVENT_NUM_2* field. Count value does not wrap around and remains stuck at the maximum value of 0xFF...F. Value is cleared by read.

8.6.8 PCIe Counter #3 - GSCN_3 (0x5B2C; RC)

Counter is common for all functions.

Field	Bit(s)	Initial Value	Description
EVC	31:0	0x0	Event Counter. Type of event counted is defined *by the* GSCL_2.GIO_EVENT_NUM_3 field. Count value does not wrap around and remains stuck at the maximum value of 0xFF...F. Value is cleared by read.

8.6.9 Function Active and Power State to MNG - FACTPS (0x5B30; RO)

Firmware uses this register for configuration

Field	Bit(s)	Initial Value	Description
Func0 Power State	1:0	00b	Power state indication of Function 0 00b → DR 01b → D0u 10b →D0a 11b → D3
LAN0 Valid	2	0b	LAN 0 Enable When set to 0b, it indicates that the LAN 0 function is disabled. When the function is enabled, the bit is set to 1b. The LAN 0 enable bit is set by the LAN0_DIS_N strapping pin.
Func0 Aux_En	3	0b	Function 0 Auxiliary (AUX) Power PM Enable bit shadow from the configuration space.
Reserved	5:4	0b	Reserved.
Func1 Power State	7:6	00b	Power state indication of Function 1 00b → DR 01b → D0u 10b → D0a 11b → D3
LAN1 Valid	8	0b	LAN 1 Enable When set to 0b, it indicates that the LAN 1 function is disabled. When the function is enabled, the bit is set to 1b. The LAN 1 enable bit is set by the LAN1_DIS_N strapping pin.
Func1 Aux_En	9	0b	Function 1 Auxiliary (AUX) Power PM Enable bit shadow from the configuration space.
Func2 Power State	11:10	00b	Power state indication of Function 2 00b → DR 01b → D0u 10b → D0a 11b → D3
LAN2 Valid	12	0b	LAN 2 Enable When set to 0b, it indicates that the LAN 2 function is disabled. When the function is enabled, the bit is set to 1b. The LAN 2 enable bit is set by the LAN2_DIS_N strapping pin.
Func2 Aux_En	13	0b	Function 2 Auxiliary (AUX) Power PM Enable bit shadow from the configuration space.
Func3 Power State	15:14	00b	Power state indication of Function 3 00b → DR 01b → D0u 10b → D0a 11b → D3
LAN3 Valid	16	0b	LAN 3 Enable When set to 0b, it indicates that the LAN 3 function is disabled. When the function is enabled, the bit is set to 1b. The LAN 3 enable bit is set by the LAN3_DIS_N strapping pin.

Field	Bit(s)	Initial Value	Description
Func3 Aux_En	17	0b	Function 3 Auxiliary (AUX) Power PM Enable bit shadow from the configuration space.
Reserved	28:18	0x0	Reserved
MNGCG	29	0b	MNG Clock Gated When set, indicates that the manageability clock is gated.
LAN Function Sel	30[1]	0b	When all LAN ports are enabled and LAN Function Sel = 0b, LAN 0 is routed to PCIe Function 0, LAN 1 is routed to PCIe Function 1, etc. If LAN Function Sel = 1b, LAN 0 is routed to PCIe Function 3, LAN 1 is routed to PCIe Function 2, LAN 2 is routed to PCIe Function 1 and LAN 3 is routed to PCIe Function 0. If a port is disabled a description of the mapping between LAN port and PCIe function can be found in Section 4.4.2. ***Note:*** PCIe Functions Mapping of dual port SKU is like 4 port SKU with LAN 2 and LAN 3 disabled.
PM State Changed (RC)	31	0b	Indication that one or more of the functions power states had changed. This bit is also a signal to the MNG unit to create an interrupt. This bit is cleared on read.

[1] This bit is initiated from EEPROM word "Functions Control" (0x21). .

8.6.10 Mirrored Revision ID - MREVID (0x5B64; R/W)

Field	Bit(s)	Initial Value	Description
EEPROM RevID	7:0	0x0[1]	Mirroring of Revision ID loaded from the EEPROM in PCIe configuration space (from word *Device Rev ID* word, address 0x1E).
Step REV ID	15:8	0x1	Revision ID from FUNC configuration space.
Reserved	31:16	0x0	Reserved

[1] Loaded from EEPROM.

8.6.11 PCIe Control Extended Register - GCR_EXT (0x5B6C; RW)

Field	Bit(s)	Initial Value	Description
Reserved	3:0	0x0	Reserved
APBACD	4	0b	Auto PBA Clear Disable. When set to 1, Software can clear the PBA only by direct write to clear access to the PBA bit. When set to 0, any active PBA entry is cleared on the falling edge of the appropriate interrupt request to the PCIe block. The appropriate interrupt request is cleared when software sets the associated interrupt mask bit in the EIMS (re-enabling the interrupt) or by direct write to clear to the PBA.
Reserved	31:5	0x00	Reserved

8.6.12 PCIe BAR Control - BARCTRL (0x5BBC; R/W) Target

Note: This register is shared by all LAN functions.

Field	Bit(s)	Initial Value	Description
Reserved	7:0	0x0	Reserved
FLSize	10:8	000b[1]	This field indicates the size of the external Flash device equals to 64KB x 2^{FLSize}. See table below for the usable FLASH size. Note: Value is loaded from EEPROM word *Initialization Control Word 2*.
Reserved	12:11	0x0	Reserved
CSRSize	13	0b[1]	The *CSRSize* and *FLSize* fields define the usable FLASH size and CSR mapping window size as shown in the table below. Note: Value is loaded from EEPROM word *Initialization Control Word 2*.
PREFBAR	14	0b[1]	Prefetchable bit indication in the memory BARs 0 BARs are marked as non prefetchable 1 BARs are marked as prefetchable Note: Value is loaded from EEPROM word *Functions Control*.
BAR32	15	1b[1]	BAR 32bit Enable. When set 32bit BARs are enabled. At 0b 64 bit BAR addressing mode is selected. When *PREFBAR* bit is set *BAR32* bit value should always be 0. Note: Value is loaded from EEPROM word *Functions Control*.
Reserved	31:16	0x0	Reserved

[1] These bits are loaded from EEPROM.

8.7 Semaphore Registers

This section contains registers common to all ports used to coordinate between all functions. The usage of these registers is described in Section 4.7

8.7.1 Software Semaphore - SWSM (0x5B50; R/W)

Field	Bit(s)	Initial Value	Description
SMBI (RS)	0	0x0	Software/Software Semaphore Bit This bit is set by hardware when this register is read by the device driver and cleared when the HOST driver writes a 0b to it. The first time this register is read, the value is 0b. In the next read the value is 1b (hardware mechanism). The value remains 1b until the software device driver clears it. This bit can be used as a semaphore between all the device's drivers in the 82580. This bit is cleared on PCIe reset.
SWESMBI	1	0x0	Software/Firmware Semaphore bit This bit should be set only by the device driver (read only to firmware). The bit is not set if bit 0 in the FWSM register is set. The device driver should set this bit and than read it to verify that it was set. If it was set, it means that the device driver can access the SW_FW_SYNC register. The device driver should clear this bit after modifying the SW_FW_SYNC register. Notes: • If Software takes ownership of the *SWSM.SWESMBI* bit for a duration longer than 100 mS, Firmware may take ownership of the bit. • Hardware clears this bit on PCIe reset.
Reserved	31:2	0x0	Reserved

8.7.2 Firmware Semaphore - FWSM (0x5B54; R/WS)

Field[1]	Bit(s)	Initial Value	Description
EEP_FW_Semaphore	0	0b	Software/Firmware Semaphore Firmware should set this bit to 1b before accessing the *SW_FW_SYNC* register. If the software is using the SWSM register and does not lock the SW_FW_SYNC, firmware is able to set this bit to 1b. Firmware should set this bit back to 0b after modifying the *SW_FW_SYNC* register. Note: If Software takes ownership of the *SWSM.SWESMBI* bit for a duration longer than 100 mS, Firmware may take ownership of the bit.
FW_Mode	3:1	0x0	Firmware Mode Indicates the firmware mode as follows: 000b = No MNG. 001b = Reserved. 010b = PT mode. 011b = Reserved. 100b = Host Interface enable only.
Reserved	5:4	00b	Reserved
EEP_Reload_Ind	6	0b	EEPROM reloaded indication Set to 1b after firmware reloads the EEPROM. Cleared by firmware once the “Clear Bit” host command is received from host software.
Reserved	14:7	0x0	Reserved

Field[1]	Bit(s)	Initial Value	Description
FW_Val_Bit	15	0b	Firmware Valid Bit Hardware clears this bit in reset de-assertion so software can know firmware mode (bits 1-3) bits are invalid. Firmware should set this bit to 1b when it is ready (end of boot sequence).
Reset_Cnt	18:16	0b	Reset Counter Firmware increments the count on every Firmware reset. After 7 Firmware reset events counter stays stuck at 7 and does not wrap around.
Ext_Err_Ind	24:19	0x0	External error indication Firmware writes here the reason that the firmware operation has stopped. For example, EEPROM CRC error, etc. Possible values: 0x00: No Error 0x01 to 0x04: Reserved. 0x05: EEPROM CRC error in SB section. 0x06: EEPROM CRC error in PT-LAN/CSRs sections. 0x07: EEPROM CRC error in FW Code section. 0x08 – CRC error in PHY section. 0x09 - Reserved. 0x0A: No Manageability (No EEPROM) 0x0B to 0x0D: Reserved. 0x0E – TCO isolate mode active. 0x0F: Management memory Parity error. 0x10 to 0x03F: Reserved Note: When management error is detected, *ICR.MGMT* is set and an interrupts is sent to the Host. Ext_Err_ind values of 0x00 or 0x0A do not cause interrupt generation.
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 LAN0 PHY/SerDes. Cleared by firmware upon successful configuration of LAN0 PHY/SerDes.
PHY_SERDES1_Config_ Err_Ind	27	0b	PHY/SerDes1 configuration error indication Set to 1b by firmware when it fails to configure LAN1 PHY/SerDes. Cleared by firmware upon successful configuration of LAN1 PHY/SerDes.
Reserved	28	0b	Reserved.
SERDES2_Config_ Err_Ind	29	0b	SerDes2 configuration error indication Set to 1b by firmware when it fails to configure LAN2 SerDes. Cleared by firmware upon successful configuration of LAN2 SerDes.
SERDES3_Config_ Err_Ind	30	0b	SerDes3 configuration error indication Set to 1b by firmware when it fails to configure LAN3 SerDes. Cleared by firmware upon successful configuration of LAN3 SerDes.
Reserved	31	0b	Reserved.

Notes:

1. This register should be written only by the manageability firmware. The device driver should only read this register.
2. Firmware ignores the EEPROM semaphore in operating system hung states.

3. Bits 15:0 are cleared on firmware reset.

8.7.3 Software–Firmware Synchronization - SW_FW_SYNC (0x5B5C; RWS)

This register is intended to synchronize between software and firmware. This register is common to all ports.

Note: If Software takes ownership of bits in the *SW_FW_SYNC* register for a duration longer than 1 Second, Firmware may take ownership of the bit.

Field	Bit(s)	Initial Value	Description
SW_EEP_SM	0	0b	When set to 1b, EEPROM access is owned by software
SW_PHY_SM0	1	0b	When set to 1b, SerDes/PHY 0 access is owned by software
SW_PHY_SM1	2	0b	When set to 1b, SerDes/PHY 1 access is owned by software
SW_MAC_CSR_SM	3	0b	When set to 1b, software owns access to shared CSRs
SW_FLASH_SM	4	0	When set to 1b, software owns access to the flash.
SW_PHY_SM2	5	0b	When set to 1b, SerDes/PHY 2 access is owned by software
SW_PHY_SM3	6	0b	When set to 1b, SerDes/PHY 3 access is owned by software
Reserved	7	0b	Reserved. Write 0, ignore on read.
SW_MB_SM	8	0b	When Set to 1b, *SWMBWR* mailbox write register, is owned by software driver.
Reserved	15:9	0x0	Reserved for future use
FW_EEP_SM	16	0b	When set to 1b, EEPROM access is owned by firmware
FW_PHY_SM0	17	0b	When set to 1b, PHY 0 access is owned by firmware
FW_PHY_SM1	18	0b	When set to 1b, PHY 1 access is owned by firmware
FW_MAC_CSR_SM	19	0b	When set to 1b, firmware owns access to shared CSRs
FW_FLASH_SM	20	0	When set to 1b, firmware owns access to the flash.
FW_PHY_SM2	21	0b	When set to 1b, PHY 2 access is owned by firmware
FW_PHY_SM3	22	0b	When set to 1b, PHY 3 access is owned by firmware
Reserved	31:23	0x0	Reserved for future use

Reset conditions:

- The software-controlled bits 15:0 are reset as any other CSR on global resets, D3hot exit and Forced TCO. Software is expected to clear the bits on entry to D3 state.
- The Firmware controlled bits (bits 31:16) are reset on LAN_PWR_GOOD (power-up) and firmware reset.

8.7.4 Software Mailbox Write - SWMBWR (0x5B04; R/W)

Field	Bit(s)	Initial Value	Description
Mailbox	31:0	0x0	Message sent from driver to the other drivers. The interpretation of this field is defined by the drivers. This register is reset only by power on reset.

8.7.5 Software Mailbox 0 - SWMB0 (0x5B08; RO)

Field	Bit(s)	Initial Value	Description
Mailbox	31:0	0x0	Message sent from the driver of port 0. The interpretation of this field is defined by the drivers. This register is reset only by power on reset.

8.7.6 Software Mailbox 1 - SWMB1 (0x5B0C; RO)

Field	Bit(s)	Initial Value	Description
Mailbox	31:0	0x0	Message sent from the driver of port 1. The interpretation of this field is defined by the drivers. This register is reset only by power on reset.

8.7.7 Software Mailbox 2 - SWMB2 (0x5B18; RO)

Field	Bit(s)	Initial Value	Description
Mailbox	31:0	0x0	Message sent from the driver of port 2. The interpretation of this field is defined by the drivers. This register is reset only by power on reset.

8.7.8 Software Mailbox 3 - SWMB3 (0x5B1C; RO)

Field	Bit(s)	Initial Value	Description
Mailbox	31:0	0x0	Message sent from the driver of port 3.The interpretation of this field is defined by the drivers. This register is reset only by power on reset.

8.8 Interrupt Register Descriptions

8.8.1 PCIe Interrupt Cause - PICAUSE (0x5B88; RW1/C)

Field	Bit(s)	Init.	Description
CA	0	0b	PCI Completion Abort exception issued.
UA	1	0b	Unsupported IO address exception. Bit is set when: IO access to address outside of the allocated address space is detected. IO access to non-CSR address space (Flash access). Write access to IOADDR with partial Byte-Enable. **Note:** When IO access via configuration space is enabled bit should be masked.
BE	2	0b	Wrong Byte-Enable exception in the FUNC unit.
TO	3	0b	PCI Timeout exception in the FUNC unit.
BMEF	4	0b	Asserted when bus-master-enable (BME) of the PF is de-asserted.
ABR	5	0b	PCI Completer Abort Received. PCI Completer Abort (CA) or Unsupported Request (UR) received (Set on reception of CA or UR). **Note:** When bit is set all PCIe master activity is stopped. Software should issue a software (*CTRL.RST*) reset to enable PCIe activity on all ports.
Reserved	31:6	0x0	Reserved

8.8.2 PCIe Interrupt Enable - PIENA (0x5B8C; R/W)

Field	Bit(s)	Init.	Description
CA	0	0b	When set to 1 the PCI Completion Abort interrupt is enabled.
UA	1	0b	When set to 1 the Unsupported IO address interrupt is enabled. **Note:** When IO access via configuration space is enabled bit should be 0b.
BE	2	0b	When set to 1 the Wrong Byte-Enable interrupt is enabled.
TO	3	0b	When set to 1 the PCI Timeout interrupt is enabled.
BMEF	4	0b	When set to 1 the Bus Master Enable interrupt is enabled.
ABR	5	0b	When set to 1 the PCI completion abort received interrupt is enabled.
Reserved	31:6	0x0	Reserved

8.8.3 Extended Interrupt Cause - EICR (0x1580; RC/W1C)

This register contains the frequent interrupt conditions for the 82580. Each time an interrupt event occurs, the corresponding interrupt bit is set in this register. An interrupt is generated each time one of the bits in this register is set and the corresponding interrupt is enabled via the Interrupt Mask Set/ Read (*EIMS*) register. The interrupt might be delayed by the selected Interrupt Throttling register.

Note that the software device driver cannot determine from the *RxTxQ* bits what was the cause of the interrupt. The possible causes for asserting these bits are:

- Receive Descriptor Write Back, Receive Descriptor Minimum Threshold hit, low latency interrupt for Rx, Transmit Descriptor Write Back.

Writing a 1b to any bit in the register clears that bit. Writing a 0b to any bit has no effect on that bit.

Register bits are cleared on register read.

Auto clear can be enabled for any or all of the bits in this register, via the *EIAC* register.

Table 8-10. EICR Register Bit Description - non MSI-X mode (GPIE.Multiple_MSIX = 0)

Field	Bit(s)	Initial Value	Description
RxTxQ	7:0	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 to the appropriate RxTxQ bit is according to the *IVAR* registers.
Reserved	29:8	0x0	Reserved
TCP Timer	30	0b	TCP Timer Expired Activated when the TCP timer reaches its terminal count.
Other Cause	31	0b	Interrupt Cause Active Activated when any bit in the ICR register is set.

Note: Bits are not reset by Device Reset (*CTRL.DEV_RST*).

Table 8-11. EICR Register Bit Description - MSI-X mode (GPIE.Multiple_MSIX = 1)

Field	Bit(s)	Initial Value	Description
MSIX	9:0	0x0	Indicates an interrupt cause mapped to MSI-X vectors 9:0 Note: Bits are not reset by Device Reset (*CTRL.DEV_RST*).
Reserved	31:10	0x0	Reserved

8.8.4 Extended Interrupt Cause Set - EICS (0x1520; WO)

Software uses this register to set an interrupt condition. Any bit written with a 1b sets the corresponding bit in the Extended Interrupt Cause Read register. An interrupt is then generated if one of the bits in this register is set and the corresponding interrupt is enabled via the Extended Interrupt Mask Set/Read register. Bits written with 0b are unchanged.

Table 8-12. EICS Register Bit Description - non MSI-X mode (GPIE.Multiple_MSIX = 0)

Field	Bit(s)	Initial Value	Description
RxTxQ	7:0	0x0	Sets to corresponding *EICR RxTXQ* interrupt condition.
Reserved	29:8	0x0	Reserved
TCP Timer	30	0b	Sets the corresponding *EICR TCP Timer* interrupt condition.
Reserved	31	0b	Reserved

Note: In order to set bit 31 of the *EICR* (*Other Causes*), the ICS and IMS registers should be used in order to enable one of the legacy causes.

Table 8-13. EICS Register Bit Description - MSI-X mode (GPIE.Multiple_MSIX = 1)

Field	Bit(s)	Initial Value	Description
MSIX	9:0	0x0	Sets the corresponding EICR bit of MSI-X vectors 9:0
Reserved	31:10	0x0	Reserved

8.8.5 Extended Interrupt Mask Set/Read - EIMS (0x1524; RWS)

Reading of this register returns which bits have an interrupt mask set. An interrupt in *EICR* is enabled if its corresponding mask bit is set to 1b and disabled if its corresponding mask bit is set to 0b. A PCI interrupt is generated each time one of the bits in this register is set and the corresponding interrupt condition occurs (subject to throttling). The occurrence of an interrupt condition is reflected by having a bit set in the Extended Interrupt Cause Read register.

An interrupt might be enabled by writing a 1b to the corresponding mask bit location (as defined in the *EICR* register) in this register. Any bits written with a 0b are unchanged. As a result, if software needs to disable an interrupt condition that had been previously enabled, it must write to the Extended Interrupt Mask Clear (*EIMC*) register rather than writing a 0b to a bit in this register.

Table 8-14. EIMS Register Bit Description - non MSI-X mode (GPIE.Multiple_MSIX = 0)

Field	Bit(s)	Initial Value	Description
RxTxQ	7:0	0x0	Set Mask bit for the corresponding *EICR RxTXQ* interrupt.
Reserved	29:8	0x0	Reserved
TCP Timer	30	0b	Set Mask bit for the corresponding *EICR TCP* timer interrupt condition.
Other Cause	31	1b	Set Mask bit for the corresponding *EICR Other Cause* interrupt condition.

Note: Bits are not reset by Device Reset (*CTRL.DEV_RST*).

Table 8-15. EIMS Register Bit Description - MSI-X mode (GPIE.Multiple_MSIX = 1)

Field	Bit(s)	Initial Value	Description
MSIX	9:0	0x0	Set Mask bit for the corresponding *EICR* bit of MSI-X vectors 9:0. Note: Bits are not reset by Device Reset (*CTRL.DEV_RST*).
Reserved	31:10	0x0	Reserved

8.8.6 Extended Interrupt Mask Clear - EIMC (0x1528; WO)

This register provides software a way to disable certain or all interrupts. Software disables a given interrupt by writing a 1b to the corresponding bit in this register.

On interrupt handling, the software device driver should set all the bits in this register related to the current interrupt request even though the interrupt was triggered by part of the causes that were allocated to this vector.

Interrupts are presented to the bus interface only when the mask bit is set to 1b and the cause bit is set to 1b. The status of the mask bit is reflected in the Extended Interrupt Mask Set/Read register and the status of the cause bit is reflected in the Interrupt Cause Read register.

Software blocks interrupts by clearing the corresponding mask bit. This is accomplished by writing a 1b to the corresponding bit location (as defined in the *EICR* register) of that interrupt in this register. Bits written with 0b are unchanged (their mask status does not change).

Table 8-16. EIMC Register Bit Description - non MSI-X mode (GPIE.Multiple_MSIX = 0)

Field	Bit(s)	Initial Value	Description
RxTxQ	7:0	0x0	Clear Mask bit for the corresponding *EICR* RxTXQ interrupt.
Reserved	29:8	0x0	Reserved
TCP Timer	30	0b	Clear Mask bit for the corresponding *EICR* TCP timer interrupt.
Other Cause	31	1b	Clear Mask bit for the corresponding *EICR* other cause interrupt.

Table 8-17. EIMC Register Bit Description - MSI-X mode (GPIE.Multiple_MSIX = 1)

Field	Bit(s)	Initial Value	Description
MSIX	9:0	0x0	Clear Mask bit for the corresponding *EICR* bit of MSI-X vectors 9:0
Reserved	31:10	0x0	Reserved

8.8.7 Extended Interrupt Auto Clear - EIAC (0x152C; R/W)

This register is mapped like the *EICS*, *EIMS*, and *EIMC* registers, with each bit mapped to the corresponding MSI-X vector.

This register is relevant to MSI-X mode only, where read-to-clear can not be used, as it might erase causes tied to other vectors. If any bits are set in *EIAC*, the *EICR* register should not be read. Bits without auto clear set, need to be cleared with write-to-clear.

Note: *EICR* bits that have auto clear set are cleared by the internal emission of the corresponding MSI-X message even if this vector is disabled by the operating system.

The MSI-X message can be delayed by *EITR* moderation from the time the *EICR* bit is activated.

Table 8-18. EIAC Register Bit Description - MSI-X mode (GPIE.Multiple_MSIX = 1)

Field	Bit(s)	Initial Value	Description
MSIX	9:0	0x0	Auto clear bit for the corresponding *EICR* bit of MSI-X vectors 9:0. Notes: • Bits are not reset by Device Reset (*CTRL.DEV_RST*). • When *GPIE.Multiple_MSIX* = 0 (Non MSI-X mode) bits 8 and 9 are read only and should be ignored.
Reserved	31:10	0x0	Reserved

8.8.8 Extended Interrupt Auto Mask Enable - EIAM (0x1530; R/W)

Each bit in this register enables clearing of the corresponding bit in *EIMS* following read- or write-to-clear to *EICR* or setting of the corresponding bit in *EIMS* following a write-to-set to *EICS*.

In MSI-X mode, this register controls which of the bits in *EIMC* to clear upon interrupt generation.

Table 8-19. EIAM Register Bit Description - non MSI-X mode (GPIE.Multiple_MSIX = 0)

Field	Bit(s)	Initial Value	Description
RxTxQ	7:0	0x0	Auto Mask bit for the corresponding *EICR RxTxQ* interrupt.
Reserved	29:8	0x0	Reserved
TCP Timer	30	0b	Auto mask bit for the corresponding *EICR TCP Timer* interrupt condition.
Other Cause	31	0b	Auto mask bit for the corresponding *EICR Other Cause* interrupt condition.

Note: Bits are not reset by Device Reset (*CTRL.DEV_RST*).

Table 8-20. EIAM Register Bit Description - MSI-X mode (GPIE.Multiple_MSIX = 1)

Field	Bit(s)	Initial Value	Description
MSIX	9:0	0x0	Auto Mask bit for the corresponding *EICR* bit of MSI-X vectors 9:0. Note: Bits are not reset by Device Reset (*CTRL.DEV_RST*).
Reserved	31:10	0x0	Reserved

8.8.9 Interrupt Cause Read Register - ICR (0x1500; RC/W1C)

This register contains the interrupt conditions for the 82580 that are not present directly in the *EICR*. Each time an *ICR* interrupt causing event occurs, the corresponding interrupt bit is set in this register. The *EICR.Other* bit reflects the setting of interrupt causes from *ICR* as masked by the Interrupt Mask Set/Read register. Each time all un-masked causes in *ICR* are cleared, the *EICR.Other Cause* bit is also cleared.

ICR bits are cleared on register read. Clear-on-read can be enabled/disabled through a general configuration register bit. See Section 7.3.2 for additional information.

Auto clear is not available for the bits in this register.

In order to prevent unwanted *LSC* (Link Status Change) interrupts during initialization, software should disable this interrupt until the end of initialization.

Field	Bit(s)	Initial Value	Description
TXDW	0	0b	Transmit Descriptor Written Back Set when the 82580 writes back a Tx descriptor to memory.
Reserved	1	0b	Reserved Write 0, ignore on read.
LSC	2	0b	Link Status Change This bit is set each time the link status changes (either from up to down, or from down to up). This bit is affected by the LINK indication from the PHY (internal PHY mode).
Reserved	3	0b	Reserved Write 0, ignore on read.
RXDMT0	4	0b	Receive Descriptor Minimum Threshold Reached Indicates that the minimum number of receive descriptors are available and software should load more receive descriptors.
Reserved	5	0b	Reserved Write 0, ignore on read.

Field	Bit(s)	Initial Value	Description
Rx Miss	6	0b	Missed packet interrupt is activated for each received packet that overflows the Rx packet buffer (overrun). Note that the packet is dropped and also increments the associated MPC counter. Note: Could be caused by no available receive buffers or because PCIe receive bandwidth is inadequate.
RXDW	7	0b	Receiver Descriptor Write Back Set when the 82580 writes back an Rx descriptor to memory.
SWMB	8	0b	Set when one of the drivers wrote a message using the SWMBWR mailbox register.
Reserved	9	0b	Reserved
GPHY	10	0b	Internal 1000/100/10BASE-T PHY interrupt. See Section 8.24.2.20 for further information.
GPI_SDP0	11	0b	General Purpose Interrupt on SDP0 If GPI interrupt detection is enabled on this pin (via *CTRL.SDP0_GPIEN*), this interrupt cause is set when the SDP0 is sampled high.
GPI_SDP1	12	0b	General Purpose Interrupt on SDP1 If GPI interrupt detection is enabled on this pin (via *CTRL.SDP1_GPIEN*), this interrupt cause is set when the SDP1 is sampled high.
GPI_SDP2	13	0b	General Purpose Interrupt on SDP2 If GPI interrupt detection is enabled on this pin (via *CTRL_EXT.SDP2_GPIEN*), this interrupt cause is set when the SDP2 is sampled high.
GPI_SDP3	14	0b	General Purpose Interrupt on SDP3 If GPI interrupt detection is enabled on this pin (via *CTRL_EXT.SDP3_GPIEN*), this interrupt cause is set when the SDP3 is sampled high.
Reserved	17:15	000b	Reserved
MNG	18	0b	Manageability Event Detected Indicates that a manageability event happened. When bit is set due to detection of error by Management, *FWSM.Ext_Err_Ind* field is updated with the error cause.
Reserved	19	0b	Reserved
OMED	20	0b	Other Media Energy Detect When in SerDes/SGMII mode, indicates that link status has changed on the external media or when in internal 1000BASE-T PHY mode, there is a change in the 10/100/1000BASE-T link status.
Reserved	21	0b	Reserved
FER	22	0b	Fatal Error This bit is set when a fatal error is detected in one of the memories
Reserved	23	0b	Reserved Write 0, ignore on read.
PCI Exception	24	0b	The PCI timeout Exception is activated by one of the following events when the specific PCI event is reported in the *PICAUSE* register and the appropriate bit in the *PIENA* register is set: (1) IO completion abort. (2) Unsupported IO request (Wrong address). (3) Byte-Enable error - Access to client that does not support Partial BE access (All but Flash, MSIX & PCIE-target). (4) Timeout occurred in the FUNC block. (5) Bus-master-enable (*BME*) of the PF is cleared.
SCE	25	0b	Storm Control Event This bit is set when multicast or broadcast storm control mechanism is activated or de-activated.

Field	Bit(s)	Initial Value	Description
Software WD	26	0b	Software Watchdog This bit is set after a software watchdog timer times out.
Reserved	27	0b	Reserved
DOUTSYNC	28	0b	DMA Detected out of Sync Situation Occurs when one of the queues used malformed descriptors. In virtualized systems, might indicate a malicious or buggy driver. Note: This bit should never rise during normal operation.
TCP timer	29	0b	TCP timer interrupt Activated when the TCP timer reaches its terminal count.
DRSTA	30	0b	Device Reset Asserted Indicates the *CTRL.DEV_RST* was asserted on another port or on this port. When device reset occurs all ports should re-initialize registers and descriptor rings. Note: Bit is not reset by Device Reset (*CTRL.DEV_RST*).
INTA	31	0	Interrupt Asserted: Indicates that the INT line is asserted. Can be used by driver in shared interrupt scenario to decide if the received interrupt was emitted by the 82580. This bit is not valid in MSI/MSI-X environments

8.8.10 Interrupt Cause Set Register - ICS (0x1504; WO)

Software uses this register to set an interrupt condition. Any bit written with a 1b sets the corresponding interrupt. This results in the corresponding bit being set in the Interrupt Cause Read Register (see Section 8.8.9). A PCIe interrupt is generated if one of the bits in this register is set and the corresponding interrupt is enabled through the Interrupt Mask Set/Read Register (see Section 8.8.11). Bits written with 0b are unchanged. See Section 7.3.2 for additional information.

Field	Bit(s)	Initial Value	Description
TXDW	0	0b	Sets the Transmit Descriptor Written Back Interrupt.
Reserved	1	0b	Reserved Write 0, ignore on read.
LSC	2	0b	Sets the Link Status Change Interrupt.
Reserved	3	0b	Reserved Write 0, ignore on read.
RXDMT0	4	0b	Sets the Receive Descriptor Minimum Threshold Hit Interrupt.
Reserved	5	0b	Reserved Write 0, ignore on read.
Rx Miss	6	0b	Sets the Rx Miss Interrupt.
RXDW	7	0b	Sets the Receiver Descriptor Write Back Interrupt.
SWMB	8	0b	Sets the SWMB mailbox interrupt.
Reserved	9	0b	Reserved
GPHY	10	0b	Sets the Internal 1000/100/10BASE-T PHY interrupt.
GPI_SDP0	11	0b	Sets the General Purpose Interrupt, related to SDP0 pin.
GPI_SDP1	12	0b	Sets the General Purpose Interrupt, related to SDP1 pin.
GPI_SDP2	13	0b	Sets the General Purpose Interrupt, related to SDP2 pin.
GPI_SDP3	14	0b	Sets the General Purpose Interrupt, related to SDP3 pin.
Reserved	17:15	000b	Reserved.
MNG	18	0b	Sets the Management Event Interrupt.
Reserved	19	0b	Reserved

Field	Bit(s)	Initial Value	Description
OMED	20	0b	Sets the Other Media Energy Detected Interrupt.
Reserved	21	0b	Reserved
FER	22	0b	Sets the Fatal Error Interrupt.
Reserved	23	0b	Reserved Write 0, ignore on read.
PCI Exception	24	0b	Sets the PCI Exception Interrupt.
SCE	25	0b	Set the Storm Control Event Interrupt
Software WD	26	0b	Sets the Software Watchdog Interrupt.
Reserved	27	0b	Reserved.
DOUTSYNC	28	0b	Sets the DMA Detected out of Sync Interrupt.
TCP timer	29	0b	Sets the TCP timer interrupt.
DRSTA	30	0b	Sets the Device Reset Asserted Interrupt. Note that when setting this bit a DRSTA interrupt is generated on this port only.
Reserved	31	0b	Reserved.

8.8.11 Interrupt Mask Set/Read Register - IMS (0x1508; R/W)

Reading this register returns bits that have an interrupt mask set. An interrupt is enabled if its corresponding mask bit is set to 1b and disabled if its corresponding mask bit is set to 0b. A PCIe interrupt is generated each time one of the bits in this register is set and the corresponding interrupt condition occurs. The occurrence of an interrupt condition is reflected by having a bit set in the Interrupt Cause Read Register (see Section 8.8.9).

A particular interrupt can be enabled by writing a 1b to the corresponding mask bit in this register. Any bits written with a 0b are unchanged. As a result, if software desires to disable a particular interrupt condition that had been previously enabled, it must write to the Interrupt Mask Clear Register (see Section 8.8.12) rather than writing a 0b to a bit in this register. See Section 7.3.2 for additional information.

Field	Bit(s)	Initial Value	Description
TXDW	0	0b	Sets/Reads the mask for Transmit Descriptor Written Back Interrupt.
Reserved	1	0b	Reserved Write 0, ignore on read.
LSC	2	0b	Sets/Reads the mask for Link Status Change Interrupt.
Reserved	3	0b	Reserved Write 0, ignore on read.
RXDMT0	4	0b	Sets/Reads the mask for Receive Descriptor Minimum Threshold Hit Interrupt.
Reserved	5	0b	Reserved Write 0, ignore on read.
Rx Miss	6	0b	Sets/Reads the mask for the Rx Miss Interrupt.
RXDW	7	0b	Sets/Reads the mask for Receiver Descriptor Write Back Interrupt.
SWMB	8	0b	Sets/Reads the mask for Software Mailbox Interrupt.
Reserved	9	0b	Reserved
GPHY	10	0b	Sets/Reads the mask for Internal 1000/100/10BASE-T PHY interrupt.
GPI_SDP0	11	0b	Sets/Reads the mask for General Purpose Interrupt, related to SDP0 pin.
GPI_SDP1	12	0b	Sets/Reads the mask for General Purpose Interrupt, related to SDP1 pin.
GPI_SDP2	13	0b	Sets/Reads the mask for General Purpose Interrupt, related to SDP2 pin.
GPI_SDP3	14	0b	Sets/Reads the mask for General Purpose Interrupt, related to SDP3 pin.
Reserved	17:15	000b	Reserved.
MNG	18	0b	Sets/Reads the mask for Management Event Interrupt.
Reserved	19	0b	Reserved
OMED	20	0b	Sets/Reads the mask for Other Media Energy Detected Interrupt.
Reserved	21	0b	Reserved
FER	22	0b	Sets/Reads the mask for the Fatal Error Interrupt.
Reserved	23	0b	Reserved Write 0, ignore on read.
PCI Exception	24	0b	Sets/Reads the mask for the PCI Exception Interrupt.
SCE	25	0b	Sets/Reads the mask for the Storm Control Event Interrupt.
Software WD	26	0b	Sets/Reads the mask for the Software Watchdog Interrupt.
Reserved	27	0b	Reserved.
DOUTSYNC	28	0b	Sets/Reads the mask for DMA Detected out of Sync Interrupt.
TCP timer	29	0b	Sets/Reads the mask for TCP timer interrupt.
DRSTA	30	0b	Sets/Reads the mask for Device Reset Asserted Interrupt. Note: Bit is not reset by Device Reset (CTRL.DEV_RST).
Reserved	31	0b	Reserved.

8.8.12 Interrupt Mask Clear Register - IMC (0x150C; WO)

Software uses this register to disable an interrupt. Interrupts are presented to the bus interface only when the mask bit is set to 1b and the cause bit set to 1b. The status of the mask bit is reflected in the Interrupt Mask Set/Read Register (see Section 8.8.11), and the status of the cause bit is reflected in the Interrupt Cause Read Register (see Section 8.8.9). Reading this register returns the value of the IMS register.

Software blocks interrupts by clearing the corresponding mask bit. This is accomplished by writing a 1b to the corresponding bit in this register. Bits written with 0b are unchanged (their mask status does not change).

The software device driver should set all the bits in this register related to the current interrupt request when handling interrupts, even though the interrupt was triggered by part of the causes that were allocated to this vector. See Section 7.3.2 for additional information.

Field	Bit(s)	Initial Value	Description
TXDW	0	0b	Clears the mask for Transmit Descriptor Written Back Interrupt.
Reserved	1	0b	Reserved Write 0, ignore on read.
LSC	2	0b	Clears the mask for Link Status Change Interrupt.
Reserved	3	0b	Reserved Write 0, ignore on read.
RXDMT0	4	0b	Clears the mask for Receive Descriptor Minimum Threshold Hit Interrupt.
Reserved	5	0b	Reserved Write 0, ignore on read.
Rx Miss	6	0b	Clears the mask for the Rx Miss Interrupt.
RXDW	7	0b	Clears the mask for the Receiver Descriptor Write Back Interrupt.
SWMB	8	0b	Clears the mask for the Software Mailbox interrupt.
Reserved	9	0b	Reserved
GPHY	10	0b	Clears the mask for the Internal 1000/100 10BASE-T PHY interrupt
GPI_SDP0	11	0b	Clears the mask for the General Purpose Interrupt, related to SDP0 pin.
GPI_SDP1	12	0b	Clears the mask for the General Purpose Interrupt, related to SDP1 pin.
GPI_SDP2	13	0b	Clears the mask for the General Purpose Interrupt, related to SDP2 pin.
GPI_SDP3	14	0b	Clears the mask for the General Purpose Interrupt, related to SDP3 pin.
Reserved	17:15	000b	Reserved.
MNG	18	0b	Clears the mask for the Management Event Interrupt.
Reserved	19	0b	Reserved
OMED	20	0b	Clears the mask for the Other Media Energy Detected Interrupt.
Reserved	21	0b	Reserved
FER	22	0b	Clears the mask for the Fatal Error Interrupt.
Reserved	23	0b	Reserved Write 0, ignore on read.
PCI Exception	24	0b	Clears the mask for the PCI Exception Interrupt.
SCE	25	0b	Clears the mask for the Storm Control Event Interrupt.
Software WD	26	0b	Clears the mask for Software Watchdog Interrupt.
Reserved	27	0b	Reserved.
DOUTSYNC	28	0b	Clears the mask for DMA Detected out of Sync Interrupt.
TCP timer	29	0b	Clears the mask for TCP timer interrupt.
DRSTA	30	0b	Clears the mask for Device Reset Asserted Interrupt.
Reserved	31	0b	Reserved. Write 0, ignore on read.

8.8.13 Interrupt Acknowledge Auto Mask Register - IAM (0x1510; R/W)

Field	Bit(s)	Initial Value	Description
IAM_VALUE	31:0	0b	An *ICR* read or write will have the side effect of writing the contents of this register to the *IMC* (*Interrupt Mask Clear*) register. If *GPIE.NSICR* = 0, then the copy of this register to the *IMC* register will occur only if at least one bit is set in the *IMS* register and there is a true interrupt as reflected in the *ICR.INTA* bit. See Section 7.3.2 for additional information. Note: Bit 30 of this register is not reset by Device Reset (*CTRL.DEV_RST*).

8.8.14 Interrupt Throttle - EITR (0x1680 + 4*n [n = 0...9]; R/W)

Each EITR is responsible for an interrupt cause (RxTxQ, TCP timer and Other Cause). The allocation of EITR-to-interrupt cause is through the IVAR registers.

Software uses this register to pace (or even out) the delivery of interrupts to the host processor. This register provides a guaranteed inter-interrupt delay between interrupts asserted by the 82580, regardless of network traffic conditions. To independently validate configuration settings, software can use the following algorithm to convert the inter-interrupt interval value to the common interrupts/sec. performance metric:

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

A counter counts in units of $1*10^{-6}$ sec. After counting "interval "number of units, an interrupt is sent to the software. The above equation gives the number of interrupts per second. The equation below time in seconds between consecutive interrupts.

For example, if the interval is programmed to 125 (decimal), the 82580 guarantees the processor does not receive an interrupt for 125 µs from the last interrupt. The maximum observable interrupt rate from the 82580 should never exceed 8000 interrupts/sec.

Inversely, inter-interrupt interval value can be calculated as:

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

The optimal performance setting for this register is very system and configuration specific. An initial suggested range is 2 to 175 (0x02 to 0xAF).

Note: Setting EITR to a non zero value can cause an interrupt cause Rx/Tx statistics miscount.

Field	Bit(s)	Initial Value	Description
Reserved	1:0	0x0	Reserved
Interval	14:2	0x0	Minimum inter-interrupt interval. The interval is specified in 1 µs increments. A zero disables interrupt throttling logic.
LLI_EN	15	0b	LLI moderation enable.

Field	Bit(s)	Initial Value	Description
Reserved	1:0	0x0	Reserved
LL Counter (RWS)	20:16	0x0	Reflects the current credits for that EITR for LL interrupts. If the CNT_INGR is not set this counter can be directly written by software at any time to alter the throttles performance
Moderation Counter (RWS)	30:21	0x0	Down counter, exposes only the 10 most significant bits of the real 12-bit counter. Loaded with Interval value whenever the associated interrupt is signaled. Counts down to 0 and stops. The associated interrupt is signaled whenever this counter is zero and an associated (via the Interrupt Select register) EICR bit is set. If the CNT_INGR is not set this counter can be directly written by software at any time to alter the throttles performance.
CNT_INGR (WO)	31	0b	When set the hardware does not override the counters fields (ITR counter and LLI credit counter), so they keep their previous value. Relevant for the current write only and is always read as zero

Note: EITR register and interrupt mechanism is not reset by Device Reset (*CTRL.DEV_RST*). Occurrence of Device Reset interrupt causes immediate generation of all pending interrupts.

8.8.15 Interrupt Vector Allocation Registers - IVAR (0x1700 + 4*n [n=0...3]; RW)

These registers have two modes of operation:

1. In MSI-X mode these registers define the allocation of the different interrupt causes as defined in Table 7-48 to one of the MSI-X vectors. Each INT_Alloc[i] (i=0…15) field is a byte indexing an entry in the MSI-X Table Structure and MSI-X PBA Structure.
2. In non MSI-X mode these registers define the allocation of the Rx and Tx queues interrupt causes to one of the RxTxQ bits in the EICR. Each INT_Alloc[i] (i=0…15) field is a byte indexing the appropriate RxTxQ bit as defined in Table 7-47.

Entries are mapped as follows:

a. Queues RX0, TX0, RX1, TX1 are mapped in IVAR[0] Register.

b. Queues RX2, TX2, RX3, TX3 are mapped in IVAR[1] Register.

c. Queues RX4, TX4, RX5, TX5 are mapped in IVAR[2] Register.

d. Queues RX6, TX6, RX7, TX7 in are mapped IVAR[3] Register.

Field	Bit(s)	Initial Value	Description
INT_Alloc[0]	3:0	0x0	Defines the MSI-X vector assigned to the interrupt cause associated with this entry, as defined in Table 7-48. Valid values are 0 to 9 for MSI-X mode and 0 to 7 in non MSI-X mode.
Reserved	6:4	0x0	Reserved
INT_Alloc_val[0]	7	0b	Valid bit for INT_Alloc[0]
INT_Alloc[1]	11:8	0x0	Defines the MSI-X vector assigned to the interrupt cause associated with this entry, as defined in Table 7-48. Valid values are 0 to 9 for MSI-X mode and 0 to 7 in non MSI-X mode.
Reserved	14:12	0x0	Reserved
INT_Alloc_val[1]	15	0b	Valid bit for INT_Alloc[1]
INT_Alloc[2]	19:16	0x0	Defines the MSI-X vector assigned to the interrupt cause associated with this entry, as defined in Table 7-48. Valid values are 0 to 9 for MSI-X mode and 0 to 7 in non MSI-X mode.

Field	Bit(s)	Initial Value	Description
Reserved	22:20	00b	Reserved
INT_Alloc_val[2]	23	0b	Valid bit for INT_Alloc[2]
INT_Alloc[3]	27:24	0x0	Defines the MSI-X vector assigned to the interrupt cause associated with this entry, as defined in Table 7-48. Valid values are 0 to 9 for MSI-X mode and 0 to 7 in non MSI-X mode.
Reserved	30:28	00b	Reserved
INT_Alloc_val[3]	31	0b	Valid bit for INT_Alloc[3]

Note: If invalid values are written to the INT_Alloc fields the result is unexpected.

DW	31 24	23 16	15 8	7 0
0	INT_ALLOC[3]	INT_ALLOC[2]	INT_ALLOC[1]	INT_ALLOC[0]
1			...	...
2		...		
3	INT_ALLOC[15]	INT_ALLOC[14]	INT_ALLOC[13]	INT_ALLOC[12]

8.8.16 Interrupt Vector Allocation Registers - MISC IVAR_MISC (0x1740; RW)

This register is used only in MSI-X mode. This register defines the allocation of the *Other Cause* and *TCP Timer* interrupts to one of the MSI-X vectors.

Field	Bit(s)	Initial Value	Description
INT_Alloc[16]	3:0	0x0	Defines the MSI-X vector assigned to the TCP timer interrupt cause. Valid values are 0 to 9.
Reserved	6:4	00b	Reserved
INT_Alloc_val[16]	7	0b	Valid bit for INT_Alloc[16]
INT_Alloc[17]	11:8	0x0	Defines the MSI-X vector assigned to the “Other Cause” interrupt. Valid values are 0 to 9.
Reserved	14:12	00b	Reserved
INT_Alloc_val[17]	15	0b	Valid bit for INT_Alloc[17]
Reserved	31:16	0x0	Reserved

8.8.17 General Purpose Interrupt Enable - GPIE (0x1514; RW)

Field	Bit(s)	Initial Value	Description
NSICR	0	0b	Non Selective Interrupt clear on read: When set, every read of *ICR* clears it. When this bit is cleared, an *ICR* read causes it to be cleared only if an actual interrupt was asserted or if no bit is set in the *IMS* register.
Reserved	3:1	0x0	Reserved.
Multiple MSIX	4	0b	0 = on-MSI mode, or MSI-X with single vector, IVAR maps Rx/Tx causes, to 8 EICR bits, but MSIX[0] is asserted for all. 1 = MSIX mode, IVAR maps Rx/Tx causes, TCP Timer and “Other Cause” interrupts to 10 MSI-x vectors reflected in 10 EICR bits.
Reserved	6:5	0x0	Reserved
LL Interval	11:7	0x0	Low latency credits increment rate. The interval is specified in 4 µs increments. A value of 0x0 disables moderation of LLI for all interrupt vectors.

Reserved	29:12	0x0	Reserved
EIAME	30	0b	Extended Interrupt Auto Mask enable: When set (usually in MSI-X mode); upon firing of an MSI-X message, bits set in EIAM associated with this message are cleared. Otherwise, EIAM is used only upon read or write of 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 82580 behaves in a way that supports legacy INT-x interrupts. Note: Should be cleared when working in INT-x or MSI mode and set in MSI-X mode.

8.9 MSI-X Table Register Descriptions

These registers are used to configure the MSI-X mechanism. The *message address* and *message upper address* registers sets the address for each of the vectors. The message register sets the data sent to the relevant address. The vector control registers are used to enable specific vectors.

The pending bit array register indicates which vectors have pending interrupts. The structure is listed in Table 8-21.

Table 8-21. MSI-X Table Structure

DWORD3 MSIXTVCTRL	DWORD2 MSIXTMSG	DWORD1 MSIXTUADD	DWORD0 MSIXTADD	Entry Number	BAR 3 - Offset
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry 0	Base (0x0000)
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry 1	Base + 1*16
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry 2	Base + 2*16
…	…	…	…	…	
Vector Control	Msg Data	Msg Upper Addr	Msg Addr	Entry (N-1)	Base + (N-1) *16

Note: N = 10.

Table 8-22. MSI-X PBA Structure

MSIXPBA[63:0]	QWORD Number	BAR 3 - Offset
Pending Bits 0 through 63	QWORD0	Base (0x2000)
Pending Bits 64 through 127	QWORD1	Base+1*8
…	…	…
Pending Bits ((N-1) div 64)*64 through N-1	QWORD((N-1) div 64)	BASE + ((N-1) div 64)*8

Note: N = 10. As a result, only QWORD0 is implemented.

8.9.1 MSI–X Table Entry Lower Address - MSIXTADD (BAR3: 0x0000 + 0x10*n [n=0...9]; R/W)

Field	Bit(s)	Initial Value	Description
Message Address LSB (RO)	1:0	0x0	For proper DWORD alignment, software must always write 0b's to these two bits. Otherwise, the result is undefined.
Message Address	31:2	0x0	System-Specific Message Lower Address For MSI-X messages, the contents of this field from an MSI-X table entry specifies the lower portion of the DWORD-aligned address for the memory write transaction.

8.9.2 MSI–X Table Entry Upper Address - MSIXTUADD (BAR3: 0x0004 + 0x10*n [n=0...9]; R/W)

Field	Bit(s)	Initial Value	Description
Message Address	31:0	0x0	System-Specific Message Upper Address

8.9.3 MSI–X Table Entry Message - MSIXTMSG (BAR3: 0x0008 + 0x10*n [n=0...9]; R/W)

Field	Bit(s)	Initial Value	Description
Message Data	31:0	0x0	System-Specific Message Data For MSI-X messages, the contents of this field from an MSI-X table entry specifies the data written during the memory write transaction. In contrast to message data used for MSI messages, the low-order message data bits in MSI-X messages are not modified by the function.

8.9.4 MSI–X Table Entry Vector Control - MSIXTVCTRL (BAR3: 0x000C + 0x10*n [n=0...9]; R/W)

Field	Bit(s)	Initial Value	Description
Mask	0	1b	When this bit is set, the function is prohibited from sending a message using this MSI-X table entry. However, any other MSI-X table entries programmed with the same vector are still capable of sending an equivalent message unless they are also masked.
Reserved	31:1	0x0	Reserved

8.9.5 MSIXPBA Bit Description – MSIXPBA (BAR3: 0x2000; RO)

Field	Bit(s)	Initial Value	Description
Pending Bits	9:0	0x0	For each pending bit that is set, the function has a pending message for the associated MSI-X Table entry. Pending bits that have no associated MSI-X table entry are reserved.
Reserved	31:10	0x0	Reserved

8.9.6 MSI-X PBA Clear – PBACL (0x5B68; R/W1C)

Field	Bit(s)	Initial Value	Description
PENBITCLR	9:0	0x0	MSI-X Pending bits Clear Writing a 1b to any bit clears the corresponding MSIXPBA bit; writing a 0b has no effect. Note: Bits are set for a single PCIe clock cycle and than cleared.
Reserved	31:10	0x0	Reserved

8.10 Receive Register Descriptions

8.10.1 Receive Control Register - RCTL (0x0100; R/W)

This register controls all the 82580 receiver functions.

Field	Bit(s)	Initial Value	Description
Reserved	0	0b	Reserved Write to 0b for future compatibility.
RXEN	1	0b	Receiver Enable The receiver is enabled when this bit is set to 1b. Writing this bit to 0b stops reception after receipt of any in progress packet. All subsequent packets are then immediately dropped until this bit is set to 1b.
SBP	2	0b	Store Bad Packets 0b = do not store. 1b = store bad packets. This bit controls the MAC receive behavior. A packet is required to pass the address (or normal) filtering before the SBP bit becomes effective. If SBP = 0b, then all packets with layer 1 or 2 errors are rejected. The appropriate statistic would be incremented. If SBP = 1b, then these packets are received (and transferred to host memory). The receive descriptor error field (RDESC.ERRORS) should have the corresponding bit(s) set to signal the software device driver that the packet is erred. In some operating systems the software device driver passes this information to the protocol stack. In either case, if a packet only has layer 3+ errors, such as IP or TCP checksum errors, and passes other filters, the packet is always received (layer 3+ errors are not used as a packet filter). Note: symbol errors before the SFD are ignored. Any packet must have a valid SFD (RX_DV with no RX_ER in 10/100/1000BASE-T mode) in order to be recognized by the 82580 (even bad packets). Also, erred packets are not routed to the MNG even if this bit is set.
UPE	3	0b	Unicast Promiscuous Enabled 0b = Disabled. 1b = Enabled.

Field	Bit(s)	Initial Value	Description
MPE	4	0b	Multicast Promiscuous Enabled 0b = Disabled. 1b = Enabled.
LPE	5	0b	Long Packet Reception Enable 0b = Disabled. 1b = Enabled. LPE controls whether long packet reception is permitted. Hardware discards long packets if LPE is 0b. A long packet is one longer than 1522 bytes. If LPE is 1b, the maximum packet size that the 82580 can receive is 9.5Kbytes.
LBM	7:6	00b	Loopback mode. Controls the loopback mode of the 82580. 00b = Normal operation (or PHY loopback in 10/100/1000BASE-T mode). 01b = MAC loopback (test mode). 10b = Undefined. 11b = Loopback via internal SerDes (SerDes/SGMII/KX mode only). When using the internal PHY, LBM should remain set to 00b and the PHY instead configured for loopback through the MDIO interface. Note: PHY devices require programming for loopback operation using MDIO accesses.
Reserved	9:8	00b	Reserved
Reserved	11:10	00b	Reserved Set to 0b for compatibility.
MO	13:12	00b	Multicast Offset Determines which bits of the incoming multicast address are used in looking up the bit vector. 00b = bits [47:36] of received destination multicast address. 01b = bits [46:35] of received destination multicast address. 10b = bits [45:34] of received destination multicast address. 11b = bits [43:32] of received destination multicast address.
Reserved	14	0b	Reserved
BAM	15	0b	Broadcast Accept Mode. 0b = Ignore broadcast (unless it matches through exact or imperfect filters). 1b = Accept broadcast packets.
BSIZE	17:16	00b	Receive Buffer Size BSIZE controls the size of the receive buffers and permits software to trade-off descriptor performance versus required storage space. Buffers that are 2048 bytes require only one descriptor per receive packet maximizing descriptor efficiency. 00b = 2048 Bytes. 01b = 1024 Bytes. 10b = 512 Bytes. 11b = 256 Bytes. **Notes:** 1. BSIZE should not be modified when RXEN is set to 1b. Set RXEN =0 when modifying the buffer size by changing this field. 2. BSIZE value only defines receive buffer size of queues with a SRRCTL.BSIZEPACKET value of 0.

Field	Bit(s)	Initial Value	Description
VFE	18	0b	VLAN Filter Enable 0b = Disabled (filter table does not decide packet acceptance). 1b = Enabled (filter table decides packet acceptance for 802.1Q packets). Three bits [20:18] control the VLAN filter table. The first determines whether the table participates in the packet acceptance criteria. The next two are used to decide whether the CFI bit found in the 802.1Q packet should be used as part of the acceptance criteria.
CFIEN	19	0b	Canonical Form Indicator Enable 0b = Disabled (CFI bit found in received 802.1Q packet's tag is not compared to decide packet acceptance). 1b = Enabled (CFI bit found in received 802.1Q packet's tag must match RCTL.CFI to accept 802.1Q type packet.
CFI	20	0b	Canonical Form Indicator bit value 0b = 802.1Q packets with CFI equal to this field are accepted. 1b = 802.1Q packet is discarded.
PSP	21	0b	Pad Small Receive packets. If this field is set, *RCTL.SECRC* should be set also.
DPF	22	0b	Discard Pause Frames with Station MAC Address Controls whether pause frames directly addressed to this station are forwarded to the host. 0b = incoming pause frames with station MAC address are forwarded to the host. 1b = incoming pause frames with station MAC address are discarded. Note: Pause frames with other MAC addresses (multicast address) are always discarded unless the specific address is added to the accepted MAC addresses (either multicast or unicast).
PMCF	23	0b	Pass MAC Control Frames Filters out unrecognized pause and other control frames. 0b = Pass/forward pause frames. 1b = Filter pause frames (default). PMCF controls the DMA function of MAC control frames (other than flow control). A MAC control frame in this context must be addressed to either the MAC control frame multicast address or the station address, match the type field, and NOT match the PAUSE opcode of 0x0001. If PMCF = 1b then frames meeting this criteria are transferred to host memory.
Reserved	25:24	0b	Reserved Should be written with 0b to ensure future compatibility.
SECRC	26	0b	Strip Ethernet CRC from incoming packet Causes the CRC to be stripped from all packets. 0b = Does not strip CRC 1b = Strips CRC. This bit controls whether the hardware strips the Ethernet CRC from the received packet. This stripping occurs prior to any checksum calculations. The stripped CRC is not transferred to host memory and is not included in the length reported in the descriptor.
Reserved	31:27	0x0	Reserved Should be written with 0b to ensure future compatibility.

8.10.2 Split and Replication Receive Control - SRRCTL (0xC00C + 0x40*n [n=0...7]; R/W)

Field	Bit(s)	Initial Value	Description
BSIZEPACKET	6:0	0x0	Receive Buffer Size for Packet Buffer The value is in 1 KB resolution. Valid values can be from 1 KB to 127 KB. Default buffer size is 0 KB. If this field is equal 0x0, then RCTL.BSIZE determines the packet buffer size.
DMACQ_Dis	7	0b	DMA Coalescing disable 0 - Enable DMA Coalescing on this queue if *DMACR.DMAC_EN* is set to 1. 1 - Disable DMA Coalescing on this queue. When packet is destined to this queue and device is in coalescing mode, Coalescing mode is exited immediately and PCIe moves to L0 link power management state.
BSIZEHEADER	11:8	0x4	Receive Buffer Size for Header Buffer The value is in 64 bytes resolution. Valid value can be from 64 bytes to 960 bytes. Default buffer size is 256 bytes. This field must be greater than 0 if the value of DESCTYPE is greater or equal to 2. Note: When SRRCTL.Timestamp is set to 1 and the value of DESCTYPE is greater or equal to 2, BSIZEHEADER size should be equal or greater than 2 (128 bytes).
Reserved	13:12	00b	Reserved Must be set to 00b.
ReservedReserved	19:14	0x0	Reserved. Write 0 ignore on read.
RDMTS	24:20	0x0	Receive Descriptor Minimum Threshold Size A low latency interrupt (LLI) associated with this queue is asserted whenever the number of free descriptors becomes equal to RDMTS multiplied by 16.
DESCTYPE	27:25	000b	Defines the descriptor in Rx 000b = Legacy. 001b = Advanced descriptor one buffer. 010b = Advanced descriptor header splitting. 011b = Advanced descriptor header replication - replicate always. 100b = Advanced descriptor header replication large packet only (larger than header buffer size). 101b = Reserved. 111b = Reserved.

Field	Bit(s)	Initial Value	Description
ReservedReserved	29:28	0x0	Reserved. Write 0 ignore on read.
TimestampReserved	30	0b	Timestamp Received packet 0 - Do not place timestamp at beginning of receive buffer. 1- Place timestamp at beginning of receive buffer. Timestamp is placed only in buffers of received packets that meet the criteria defined in the *TSYNCRXCTL.Type* field, 2-tuple filters or *ETQF* registers. When set a 40 bit time stamp generated from the value in SYSTIMH and SYSTIML registers is placed in the receive buffer before the MAC header of the packets defined in the *TSYNCRXCTL.Type* field.
Drop_En	31	0b/1b	Drop Enabled If set, packets received to the queue when no descriptors are available to store them are dropped. The packet is dropped only if there are not enough free descriptors in the host descriptor ring to store the packet. If there are enough descriptors in the host, but they are not yet fetched by the 82580, then the packet is not dropped and there are no release of packets until the descriptors are fetched. Default is 0b for queue 0 and 1b for the other queues.

8.10.3 Packet Split Receive Type - PSRTYPE (0x5480 + 4*n [n=0...7]; R/W)

This register enables or disables each type of header that needs to be split. Each register controls the behavior of 1 queue.

- Packet Split Receive Type Register (queue 0) - PSRTYPE0 (0x5480)
- Packet Split Receive Type Register (queue 1) - PSRTYPE1 (0x5484)
- Packet Split Receive Type Register (queue 2) - PSRTYPE2 (0x5488)
- Packet Split Receive Type Register (queue 3) - PSRTYPE3 (0x548C)
- Packet Split Receive Type Register (queue 4) - PSRTYPE3 (0x5490)
- Packet Split Receive Type Register (queue 5) - PSRTYPE3 (0x5494)
- Packet Split Receive Type Register (queue 6) - PSRTYPE3 (0x5498)
- Packet Split Receive Type Register (queue 7) - PSRTYPE3 (0x549C)

Field	Bit(s)	Initial Value	Description
Reserved	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
Reserved	10	1b	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
Reserved	13	1b	Reserved

Field	Bit(s)	Initial Value	Description
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
Reserved	16	1b	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	0x0	Reserved

8.10.4 Replicated Packet Split Receive Type - RPLPSRTYPE (0x54C0; R/W)

This register enables or disables each type of header that needs to be split. This register controls the behavior of replicated packets.

Field	Bit(s)	Initial Value	Description
Reserved	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
Reserved	10	1b	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
Reserved	13	1b	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
Reserved	16	1b	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	0x0	Reserved

8.10.5 Receive Descriptor Base Address Low - RDBAL (0xC000 + 0x40*n [n=0...7]; R/W)

This register contains the lower bits of the 64-bit descriptor base address. The lower four bits are always ignored. The Receive Descriptor Base Address must point to a 128 byte-aligned block of data.

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x2800, 0x2900, 0x2A00 & 0x2B00 respectively.

Field[1]	Bit(s)	Initial Value	Description
Lower_0	6:0	0x0	Ignored on writes. Returns 0x0 on reads.
RDBAL	31:7	X	Receive Descriptor Base Address Low

1. Software should program *RDBAL[n]* register only when queue is disabled (*RXDCTL[n].Enable* = 0).

8.10.6 Receive Descriptor Base Address High - RDBAH (0xC004 + 0x40*n [n=0...7]; R/W)

This register contains the upper 32 bits of the 64-bit descriptor base address.

Field[1]	Bit(s)	Initial Value	Description
RDBAH	31:0	X	Receive Descriptor Base Address [63:32]

1. Software should program *RDBAH[n]* register only when queue is disabled (*RXDCTL[n].Enable* = 0).

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x2804, 0x2904, 0x2A04 & 0x2B04 respectively.

8.10.7 Receive Descriptor Ring Length - RDLEN (0xC008 + 0x40*n [n=0...7]; R/W)

This register sets the number of bytes allocated for descriptors in the circular descriptor buffer. It must be 128-byte aligned.

Field[1]	Bit(s)	Initial Value	Description
0	6:0	0x0	Ignore on writes. Bits 6:0 must be set to zero. Bits 4:0 always read as zero.
LEN	19:7	0x0	Descriptor Ring Length (number of 8 descriptor sets). Note: maximum allowed value in RDLEN field 19:0 is 0x80000 (32K descriptors).
Reserved	31:20	0x0	Reserved Reads as 0b. Should be written to 0b for future compatibility.

1. Software should program *RDLEN[n]* register only when queue is disabled (*RXDCTL[n].Enable* = 0).

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x2808, 0x2908, 0x2A08 & 0x2B08 respectively.

8.10.8 Receive Descriptor Head - RDH (0xC010 + 0x40*n [n=0...7]; RO)

The value in this register might point to descriptors that are still not in host memory. As a result, the host cannot rely on this value in order to determine which descriptor to process.

Field	Bit(s)	Initial Value	Description
RDH	15:0	0x0	Receive Descriptor Head
Reserved	31:16	0x0	Reserved Should be written to 0b.

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x2810, 0x2910, 0x2A10 & 0x2B10 respectively.

8.10.9 Receive Descriptor Tail - RDT (0xC018 + 0x40*n [n=0...7]; R/W)

This register contains the tail pointers for the receive descriptor buffer. The register points to a 16-byte datum. Software writes the tail register to add receive descriptors to the hardware free list for the ring.

Note: Writing the RDT register while the corresponding queue is disabled is ignored by the 82580.

In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x2818, 0x2918, 0x2A18& 0x2B18 respectively.

Field	Bit(s)	Initial Value	Description
RDT	15:0	0x0	Receive Descriptor Tail
Reserved	31:16	0x0	Reserved. Ignore on read, write 0 for future compatibility.

8.10.10 Receive Descriptor Control - RXDCTL (0xC028 + 0x40*n [n=0...7]; R/W)

This register controls the fetching and write-back of receive descriptors. The three threshold values are used to determine when descriptors are read from and written to host memory. The values are in units of descriptors (each descriptor is 16 bytes).

Field	Bit(s)	Initial Value	Description
PTHRESH	4:0	0xC	Prefetch Threshold *PTHRESH* is used to control when a prefetch of descriptors is considered. This threshold refers to the number of valid, unprocessed receive descriptors the 82580 has in its on-chip buffer. If this number drops below *PTHRESH*, the algorithm considers pre-fetching descriptors from host memory. This fetch does not happen unless there are at least *HTHRESH* valid descriptors in host memory to fetch. Note: *HTHRESH* should be given a non zero value each time *PTHRESH* is used. Possible values for this field are 0 to 16.
Reserved	7:5	0x0	Reserved
HTHRESH	12:8	0xA	Host Threshold Field defines when receive descriptor prefetch is performed. Each time enough valid descriptors, as defined in the *HTHRESH* field, are available in host memory a prefetch is performed. Possible values for this field are 0 to 16.
Reserved	15:13	0x0	Reserved
WTHRESH	20:16	0x1	Write-Back Threshold *WTHRESH* controls the write-back of processed receive descriptors. This threshold refers to the number of receive descriptors in the on-chip buffer that are ready to be written back to host memory. In the absence of external events (explicit flushes), the write-back occurs only after at least *WTHRESH* descriptors are available for write-back. Possible values for this field are 0 to 15. Note: Since the default value for write-back threshold is 1b, the descriptors are normally written back as soon as one cache line is available. *WTHRESH* must contain a non-zero value to take advantage of the write-back bursting capabilities of the 82580. Note: It's recommended not to place a value above 0xC in the *WTHRESH* field.
Reserved	24:21	0x0	Reserved
ENABLE	25	0b	Receive Queue Enable When set, the *Enable* bit enables the operation of the specific receive queue. 1b =Enables queue. 0b =Disables queue. Setting this bit initializes Head and Tail registers (*RDH*[n] and *RDT*[n]) of the specific queue. Until then, the state of the queue is kept and can be used for debug purposes. When disabling a queue, this bit is cleared only after all activity in the queue has stopped. Note: When receive queue is enabled and descriptors exist, descriptors and are fetched immediately. Actual receive activity on port starts only if the *RCTL.RXEN* bit is set.
SWFLUSH (WC)	26	0b	Receive Software Flush Enables software to trigger receive descriptor write-back flushing, independently of other conditions. This bit is cleared by hardware after write-back flush is triggered (may take a number of cycles).
Reserved	31:27	0x0	Reserved

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x2828, 0x2928, 0x2A28 & 0x2B28 respectively.

8.10.11 Receive Queue drop packet count - RQDPC (0xC030 + 0x40*n [n=0...7]; RC/W)

Field	Bit(s)	Initial Value	Description
RQDPC	31:0	0x0	Receive Queue drop packet count - counts the number of packets dropped by a queue due to lack of descriptors available. Note: Counter does not wrap around when reaching a value of 0xFFFFFFFF.

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x2830, 0x2930, 0x2A30 & 0x2B30 respectively.

Packets dropped due to the queue being disabled may not be counted by this register.

8.10.12 Receive Checksum Control - RXCSUM (0x5000; R/W)

The Receive Checksum Control register controls the receive checksum off loading features of the 82580. The 82580 supports the off loading of three receive checksum calculations: the Packet Checksum, the IP Header Checksum, and the TCP/UDP Checksum.

Note: This register should only be initialized (written) when the receiver is not enabled (only write this register when RCTL.RXEN = 0b)

Field	Bit(s)	Initial Value	Description
PCSS	7:0	0x0	Packet Checksum Start Controls the packet checksum calculation. The packet checksum shares the same location as the RSS field and is reported in the receive descriptor when the RXCSUM.PCSD bit is cleared. If the RXCSUM.IPPCSE is set, the Packet checksum is aimed to accelerate checksum calculation of fragmented UDP packets. Please refer to section Section 7.1.11.2 for detailed explanation. If RXCSUM.IPPCSE is cleared (the default value), the checksum calculation that is reported in the Rx Packet checksum field is the unadjusted 16-bit ones complement of the packet. The packet checksum starts from the byte indicated by RXCSUM.PCSS (0b corresponds to the first byte of the packet), after VLAN stripping if enabled by the CTRL.VME. For example, for an Ethernet II frame encapsulated as an 802.3ac VLAN packet and with RXCSUM.PCSS set to 14, the packet checksum would include the entire encapsulated frame, excluding the 14-byte Ethernet header (DA, SA, Type/Length) and the 4-byte VLAN tag. The packet checksum does not include the Ethernet CRC if the RCTL.SECRC bit is set. Software must make the required offsetting computation (to back out the bytes that should not have been included and to include the pseudo-header) prior to comparing the packet checksum against the L4 checksum stored in the packet checksum. The partial checksum in the descriptor is aimed to accelerate checksum calculation of fragmented UDP packets. Note: The PCSS value should not exceed a pointer to the IP header start. If exceeded, the IP header checksum or TCP/UDP checksum is not calculated correctly.
IPOFLD	8	1b	IP Checksum Off-load Enable RXCSUM.IPOFLD is used to enable the IP Checksum off-loading feature. If *RXCSUM.IPOFLD* is set to 1b, the 82580 calculates the IP checksum and indicates a pass/fail indication to software via the IP Checksum Error bit (IPE) in the *Error* field of the receive descriptor. Similarly, if *RXCSUM.TUOFLD* is set to 1b, the 82580 calculates the TCP or UDP checksum and indicates a pass/fail indication to software via the TCP/UDP Checksum Error bit (L4E). Similarly, if *RFCTL.IPv6_DIS* and *RFCTL.IP6Xsum_DIS* are cleared to 0b and *RXCSUM.TUOFLD* is set to 1b, the 82580 calculates the TCP or UDP checksum for IPv6 packets. It then indicates a pass/fail condition in the TCP/UDP *Checksum Error* bit (*RDESC.L4E*). This applies to checksum off loading only. Supported frame types: Ethernet II Ethernet SNAP
TUOFLD	9	1b	TCP/UDP Checksum Off-load Enable
Reserved	10	0b	Reserved
CRCOFL	11	0b	CRC32 Offload Enable Enables the SCTP CRC32 checksum off-loading feature. If *RXCSUM.CRCOFL* is set to 1b, the 82580 calculates the CRC32 checksum and indicates a pass/fail indication to software via the CRC32 *Checksum Valid* bit (RDESC.L4I) in the *Extended Status* field of the receive descriptor. In non I/OAT, this bit is read only as 0b.

Field	Bit(s)	Initial Value	Description
IPPCSE	12	0b	IP Payload Checksum Enable See PCSS description.
PCSD	13	0b	Packet Checksum Disable The packet checksum and IP identification fields are mutually exclusive with the RSS hash. Only one of the two options is reported in the Rx descriptor. RXCSUM.PCSD Legacy Rx Descriptor (SRRCTL.DESCTYPE = 000b): 0b (checksum enable) - Packet checksum is reported in the Rx descriptor. 1b (checksum disable) - Not supported. RXCSUM.PCSD Extended or Header Split Rx Descriptor (SRRCTL.DESCTYPE not equal 000b): 0b (checksum enable) - checksum and IP identification are reported in the Rx descriptor. 1b (checksum disable) - RSS Hash value is reported in the Rx descriptor.
Reserved	31:14	0x0	Reserved

8.10.13 Receive Long Packet Maximum Length - RLPML (0x5004; R/W)

Field	Bit(s)	Initial Value	Description
RLPML	13:0	0x2600	Maximum allowed long packet length. This length is the global length of the packet including all the potential headers of suffixes in the packet.
Reserved	31:14	0x0	Reserved

8.10.14 Receive Filter Control Register - RFCTL (0x5008; R/W)

Field	Bit(s)	Initial Value	Description
Reserved	5:0	1b	Reserved
NFSW_DIS	6	0b	NFS Write Disable Disables filtering of NFS write request headers.
NFSR_DIS	7	0b	NFS Read Disable Disables filtering of NFS read reply headers.
NFS_VER	9:8	00b	NFS Version 00b = NFS version 2. 01b = NFS version 3. 10b = NFS version 4. 11b = Reserved for future use.
IPv6_DIS	10	0b	IPv6 Disable Disables IPv6 packet filtering. Any received IPv6 packet is parsed only as an L2 packet.
IPv6XSUM_DIS	11	0b	IPv6 XSUM Disable Disables XSUM on IPv6 packets.
Reserved	12	0b	Reserved
Reserved	13	0b	Reserved.
IPFRSP_DIS	14	0b	IP Fragment Split Disable When this bit is set, the header of IP fragmented packets are not set.

Field	Bit(s)	Initial Value	Description
Reserved	15	0b	Reserved
Reserved	17:16	00b	Reserved Must be set to 00b.
LEF	18	0b	Forward Length Error Packet 0b = packet with length error are dropped. 1b = packets with length error are forwarded to the host.
SYNQFP	19	0b	Defines the priority between SYNQF & 2 tuple filter 0b = 2-tuple filter priority 1b = SYN filter priority.
Reserved	31:20	0x0	Reserved Write 0 ignore on read.
Reserved	31:20	0x08	Reserved Should be written with 0b to ensure future capability.

8.10.15 Multicast Table Array - MTA (0x5200 + 4*n [n=0...127]; R/W)

There is one register per 32 bits of the Multicast Address Table for a total of 128 registers. Software must mask to the desired bit on reads and supply a 32-bit word on writes. The first bit of the address used to access the table is set according to the *RCTL.MO* field.

Note: All accesses to this table must be 32 bit.

Field	Bit(s)	Initial Value	Description
Bit Vector	31:0	X	Word wide bit vector specifying 32 bits in the multicast address filter table.

Figure 8-1 shows the multicast lookup algorithm. The destination address shown represents the internally stored ordering of the received DA. Note that bit 0 indicated in this diagram is the first on the wire.

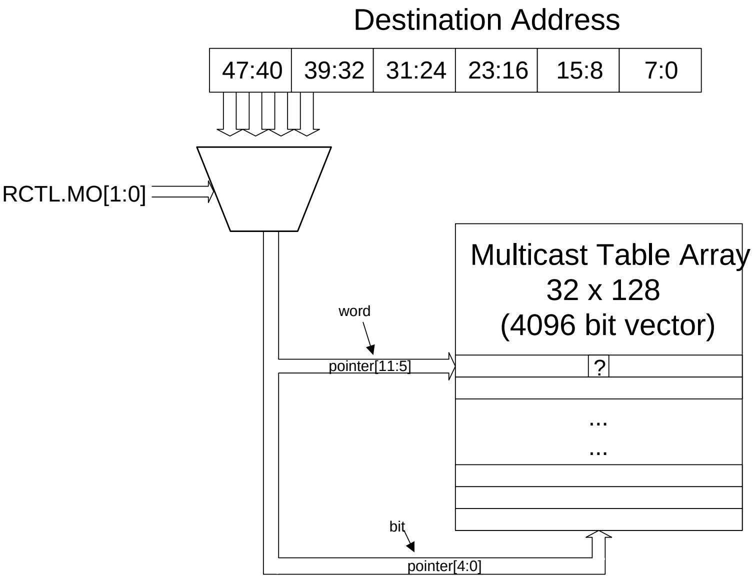


Figure 8-1. Multicast Table Array

8.10.16 Receive Address Low - RAL (0x5400 + 8*n [n=0...15]; 0x54E0 + 8*n [n=0...7]; R/W)

While “n” is the exact unicast/multicast address entry and it is equal to 0,1,...15.

These registers contain the lower bits of the 48 bit Ethernet address. All 32 bits are valid.

These registers are reset by a software reset or platform reset. If an EEPROM is present, the first register (RAL0) is loaded from the EEPROM after a software or platform reset.

Note: The *RAL* field should be written in network order.

Field	Bit(s)	Initial Value	Description
RAL	31:0	X	Receive address low Contains the lower 32-bit of the 48-bit Ethernet address.

8.10.17 Receive Address High - RAH (0x5404 + 8*n [n=0...15]; 0x54E4 + 8*n [n=0...7]; R/W)

These registers contain the upper bits of the 48 bit Ethernet address. The complete address is [RAH, RAL]. AV determines whether this address is compared against the incoming packet and is cleared by a master reset.

ASEL enables the 82580 to perform special filtering on receive packets.

After reset, if an EEPROM is present, the first register (Receive Address Register 0) is loaded from the *IA* field in the EEPROM with its *Address Select* field set to 00b and its *Address Valid* field set to 1b. If no EEPROM is present, the *Address Valid* field is set to 0b and the *Address Valid* field for all of the other registers is set to 0b.

Note: The *RAH* field should be written in network order.

The first receive address register (RAH0) is also used for exact match pause frame checking (DA matches the first register). As a result, RAH0 should always be used to store the individual Ethernet MAC address of the 82580.

Field	Bit(s)	Initial Value	Description
RAH	15:0	X	Receive address High Contains the upper 16 bits of the 48-bit Ethernet address.
ASEL	17:16	X	Address Select Selects how the address is to be used in the address filtering. 00b = Destination address (required for normal mode) 01b = Source address. This mode should not be used in virtualization mode. 10b = Reserved 11b = Reserved
POOLSEL	25:18	0x0	Pool Select In virtualization modes (*MRQC.Multiple Receive Queues Enable* = 011b) indicates which Pool should get the packets matching this MAC address. This field is a bit map (bit per VM) where more than one bit can be set according to the limitations defined in Section 7.8.2.5. If all the bits are zero, this address is used only for L2 filtering and is not used as part of the queueing decision.
Reserved	30:26	0x0	Reserved Write 0 Ignore on reads.
AV	31		Address Valid Cleared after master reset. If an EEPROM is present, the *Address Valid* field of the 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.

8.10.18 VLAN Filter Table Array - VFTA (0x5600 + 4*n [n=0...127]; R/W)

There is one register per 32 bits of the VLAN Filter Table. The size of the word array depends on the number of bits implemented in the VLAN Filter Table. Software must mask to the desired bit on reads and supply a 32-bit word on writes.

Note: All accesses to this table must be 32 bit.

The algorithm for VLAN filtering using the VFTA is identical to that used for the Multicast Table Array. Refer to Section 8.10.15 for a block diagram of the algorithm. If VLANs are not used, there is no need to initialize the VFTA.

Field	Bit(s)	Initial Value	Description
Bit Vector	31:0	X	Double-word wide bit vector specifying 32 bits in the VLAN Filter table.

8.10.19 Multiple Receive Queues Command Register - MRQC (0x5818; R/W)

Field	Bit(s)	Initial Value	Description
Multiple Receive Queues Enable	2:0	0x0	Multiple Receive Queues Enable Enables support for Multiple Receive Queues and defines the mechanism that controls queue allocation. 000b = Multiple receive queues as defined by filters (2-tuple filters, L2 Ether-type filters, SYN filter and Flex Filters). 001b = Reserved. 010b = Multiple receive queues as defined by filters and RSS for 8 queues[1]. 011b = Multiple receive queues as defined by VMDq based on packet destination MAC address (*RAH.POOLSEL*) and Ether-type queuing decision filters. 100b = Reserved 101b = Reserved 110b = Reserved 111b = Reserved. If VT is not supported, the only allowed values for this field are 000b and 010b. Writing any other value is ignored. TheAllowed values for this field are 000b, 010b and 011b. Any other value is ignored.
Def_Q	5:3	0x0	Defines the default queue in non VMDq mode according to value of the *Multiple Receive Queues Enable* field. If Multiple Receive Queues Enable = 000b: Def_Q defines the destination of all packets not forwarded by filters. 001b: Def_Q field is ignored. 010b: Def_Q defines the destination of all packets not forwarded by RSS or filters. 011b - Def_Q field is ignored. Queueing decision of all packets not forwarded by MAC address and Ether-type filters is according to VT_CTL.DEF_PL field. 100-101b: Def_Q field is ignored. 110b: Def_Q field is ignored. Note: In VMDq mode (*Multiple Receive Queues Enable* = 011b) the default queue is set according to the *VT_CTL.DEF_PL* if packet passes MAC Address filtering of a filter with *RAH.POOLSEL* = 0x0 or is a broadcast or multicast packet and does not match Ether-type queuing decision filters.
Reserved	15:6	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] = Enable IPv6Ex hash function
			Bit[20] = Enable IPv6 hash function
			Bit[21] = Enable TCPIPv6 hash function
			Bit[22] = Enable UDPIPv4
			Bit[23] = Enable UDPIPv6
			Bit[24] = Enable UDPIPv6Ext
			ReservedBits[31:26] - Reserved Zero

[1] Note that the *RXCSUM.PCSD* bit should be set to enable reception of the RSS hash value in the receive descriptor.

Note: The *MRQC.Multiple Receive Queues Enable* field is used to enable/disable RSS hashing and also to enable multiple receive queues. Disabling this feature is not recommended. Model usage is to reset the 82580 after disabling the RSS.

8.10.20 RSS Random Key Register - RSSRK (0x5C80 + 4*n [n=0...9]; R/W)

Field	Bit(s)	Initial Value	Description
K0	7:0	0x0	Byte n*4 of the RSS random key (n=0,1,...9).
K1	15:8	0x0	Byte n*4+1 of the RSS random key (n=0,1,...9).
K2	23:16	0x0	Byte n*4+2 of the RSS random key (n=0,1,...9).
K3	31:24	0x0	Byte n*4+3 of the RSS random key (n=0,1,...9).

The RSS Random Key Register stores a 40 byte key used by the RSS hash function.

31 24	23 16	15 8	7 0
K[3]	K[2]	K[1]	K[0]
...	...	...	...
K[39]	...	...	K[36]

8.10.21 Redirection Table - RETA (0x5C00 + 4*n [n=0...31]; R/W)

The redirection table is a 128-entry table with each entry being eight bits wide. Only 1 to 3 bits of each entry are used to store the queue index. The table is configured through the following R/W registers.

Field	Bit(s)	Initial Value	Description
Entry 0	7:0	0x0	Determines the tag value and physical queue for index 4*n+0 (n=0...31).
Entry 1	15:8	0x0	Determines the tag value and physical queue for index 4*n+1 (n=0...31).
Entry 2	23:16	0x0	Determines the tag value and physical queue for index 4*n+2 (n=0...31).
Entry 3	31:24	0x0	Determines the tag value and physical queue for index 4*n+3 (n=0...31).

31 24	23 16	15 8	7 0
Tag 3	Tag 2	Tag 1	Tag 0
...	...	...	...
Tag 127	...	...	...

Each entry (byte) of the redirection table contains the following:

7:3	2:0
Reserved	Queue index

- Bits [7:3] - Reserved
- Bits [2:0] - Queue index for all pools or in regular RSS. In RSS only mode, all bits are used.

The contents of the redirection table are not defined following reset of the Memory Configuration Registers. System software must initialize the Table prior to enabling multiple receive queues. It can also update the redirection table during run time. Such updates of the table are not synchronized with the arrival time of received packets. Therefore, it is not guaranteed that a table update takes effect on a specific packet boundary.

Note: In case the operating system provides a redirection Table whose size is smaller than 128 bytes, the software usually replicates the operating system-provided redirection table to span the whole 128 bytes of the hardware's redirection table.

8.11 Filtering Register Descriptions

8.11.1 Immediate Interrupt RX - IMIR (0x5A80 + 4*n [n=0...7]; R/W)

This register defines the filtering that corrects which packet triggers low latency interrupt. The following *IMIREXT* register includes a size threshold and a control bits bitmap to trigger an immediate interrupt.

Note: The *Port* field should be written in network order.

If one of the actions for this filter is set, then at least one of the *PORT_BP*, *Size_BP*, one of the Mask bits or *CtrlBit_BP* bits should be cleared.

Field	Bit(s)	Initial Value	Description
Destination Port	15:0	0x0	Destination TCP port This field is compared with the Destination TCP port in incoming packets.
Immediate Interrupt	16	0b	Enables issuing an immediate interrupt when the following conditions are met: • The 2-tuple filter associated with this register matches • The Length filter associated with this filter matches • The TCP flags filter associated with this filter matches
PORT_BP	17	X	Port Bypass When set to 1b, the TCP port check is bypassed and only other conditions are checked. When set to 0b, the TCP port is checked to fit the port field.
Reserved	28:18	0x0	Reserved
Filter Priority	31:29	000b	Defines the priority of the filter assuming two filters with same priority don’t match. If two filters with the same priority match the incoming packet, the first filter (lowest ordinal number) is used in order to define the queue destination of this packet.

8.11.2 Immediate Interrupt Rx Ext. - IMIREXT (0x5AA0 + 4*n [n=0...7]; R/W)

Field	Bit(s)	Initial Value	Description
Size_Thresh	11:0	X	Size Threshold These 12 bits define a size threshold; a packet with a length below this threshold triggers an interrupt. Enabled by Size_Thresh_en.
Size_BP	12	X	Size Bypass When 1b, the size check is bypassed. When 0b, the size check is performed.
CtrlBit	18:13	X	Control Bit When a bit in this field equals 1b, an interrupt is immediately issued after receiving a packet with the corresponding TCP control bits turned on. Bit 13 (URG): Urgent pointer field significant Bit 14 (ACK): Acknowledgment field Bit 15 (PSH): Push function Bit 16 (RST): Reset the connection Bit 17 (SYN): Synchronize sequence numbers Bit 18 (FIN): No more data from sender
CtrlBit_BP	19	X	Control Bits Bypass When set to 1b, the control bits check is bypassed. When set to 0b, the control bits check is performed.
Reserved	31:20	0x0	Reserved

Note: The size used for this comparison is the size of the packet as forwarded to the host and does not include any of the fields stripped by the MAC (VLAN or CRC). As a result, setting the *RCTL.SECRC* & *CTRL.VME* bits should be taken into account while calculating the size threshold.

The value of the IMIR and *IMIREXT* registers after reset is unknown (apart from the *IMIR.PORT_IM_EN* bit which is guaranteed to be cleared). Therefore, both registers should be programmed before I*MIR.PORT_IM_EN* is set for a given flow.

8.11.3 2-tuples Queue Filter - TTQF (0x59E0 + 4*n[n=0...7]; RW)

Field	Bit(s)	Initial Value	Description
Protocol	7:0	0x0	IP L4 protocol, part of the 2-tuple queue filters.
Queue Enable	8	0b	When set, enables filtering of Rx packets by the 2-tuples defined in this filter to the queue indicated in this register.
Reserved	11:9	0x0	Reserved. Write 0 ignore on read.
Reserved	14:12	0	Reserved.
Reserved_1	15	1b (For legacy reasons)	Reserved. Write 1 ignore on read.
Rx Queue	18:16	0x0	Identifies the Rx queue associated with this 2-tuple filter.

Field	Bit(s)	Initial Value	Description
Reserved	26:19	0b	Reserved
1588 time stamp	27	0b	When set, packets that match this filter are time stamped according to the IEEE 1588 specification.
Mask	31:28	0xF (for legacy reasons)	Mask bits for the 2-tuple fields . The corresponding field participates in the match if the bit below is cleared: Bit 28 - Mask protocol comparison Bits 31 - 29 Reserved

8.11.4 Immediate Interrupt Rx VLAN Priority - IMIRVP (0x5AC0; R/W)

Field	Bit(s)	Initial Value	Description
Vlan_Pri	2:0	000b	VLAN Priority This field includes the VLAN priority threshold. When Vlan_pri_en is set to 1b, then an incoming packet with a VLAN tag with a priority field equal or higher to VlanPri triggers an immediate interrupt, regardless of the EITR moderation.
Vlan_pri_en	3	0b	VLAN Priority Enable When set to 1b, an incoming packet with VLAN tag with a priority equal or higher to Vlan_Pri triggers an immediate interrupt, regardless of the EITR moderation. When set to 0b, the interrupt is moderated by EITR.
Reserved	31:4	0x0	Reserved

8.11.5 SYN Packet Queue Filter - SYNQF (0x55FC; RW)

Field	Bit(s)	Initial Value	Description
Queue Enable	0	0b	When set, enables forwarding of Rx packets to the queue indicated in this register.
Rx Queue	3:1	0x0	Identifies an Rx queue associated with SYN packets.
Reserved	31:4	0x0	Reserved

8.11.6 EType Queue Filter - ETQF (0x5CB0 + 4*n[n=0...7]; RW)

Field	Bit(s)	Initial Value	Description
EType	15:0	0x0	Identifies the protocol running on top of IEEE 802. Used to forward Rx packets containing this EType to a specific Rx queue.
Rx Queue	18:16	0x0	Identifies the receive queue associated with this EType.
Reserved	25:19	0x0	Reserved
Filter enable	26	0b	When set, this filter is valid. Any of the actions controlled by the following fields are gated by this field.
Reserved	27	0b	Reserved
Reserved	28	0b	Reserved

Field	Bit(s)	Initial Value	Description
Immediate Interrupt	29	0x0	When set, packets that match this filter generate an immediate interrupt.
1588 time stamp	30	0b	When set, packets with this EType are time stamped according to the IEEE 1588 specification.
Queue Enable	31	0b	When set, enables filtering of Rx packets by the EType defined in this register to the queue indicated in this register.

8.12 Transmit Register Descriptions

8.12.1 Transmit Control Register - TCTL (0x0400; R/W)

This register controls all transmit functions for the 82580.

Field	Bit(s)	Initial Value	Description
Reserved	0	0b	Reserved Write 0, ignore on read.
EN	1	0b	Transmit Enable The transmitter is enabled when this bit is set to 1b. Writing 0b to this bit stops transmission after any in progress packets are sent. Data remains in the transmit FIFO until the device is re-enabled. Software should combine this operation with reset if the packets in the TX FIFO should be flushed.
Reserved	2	0b	Reserved Write 0, ignore on read.
PSP	3	1b	Pad Short Packets 0b = Do not pad. 1b = Pad. Padding makes the packet 64 bytes long. This is not the same as the minimum collision distance. If padding of short packets is allowed, the total length of a packet not including FCS should be not less than 17 bytes.
CT	11:4	0xF	Collision Threshold This determines the number of attempts at retransmission prior to giving up on the packet (not including the first transmission attempt). While this can be varied, it should be set to a value of 15 in order to comply with the IEEE specification requiring a total of 16 attempts. The Ethernet back-off algorithm is implemented and clamps to the maximum number of slot-times after 10 retries. This field only has meaning when in half-duplex operation. Note: Software can choose to abort packet transmission in less than the Ethernet mandated 16 collisions. For this reason, hardware provides CT support.
BST	21:12	0x40	Back-Off Slot Time This value determines the back-off slot time value in byte time.
SWXOFF	22	0b	Software XOFF Transmission When set to 1b, the 82580 schedules the transmission of an XOFF (PAUSE) frame using the current value of the PAUSE timer (*FCTTV.TTV*). This bit self-clears upon transmission of the XOFF frame. Note: While 802.3x flow control is only defined during full duplex operation, the sending of PAUSE frames via the SWXOFF bit is not gated by the duplex settings within the 82580. Software should not write a 1b to this bit while the 82580 is configured for half-duplex operation.
Reserved	23	0b	Reserved

Field	Bit(s)	Initial Value	Description
RTLC	24	0b	Re-transmit on Late Collision When set, enables the 82580 to re-transmit on a late collision event. Note: RTLC configures the 82580 to perform retransmission of packets when a late collision is detected. Note that the collision window is speed dependent: 64 bytes for 10/100 Mb/s and 512 bytes for 1000 Mb/s operation. If a late collision is detected when this bit is disabled, the transmit function assumes the packet has successfully transmitted. This bit is ignored in full-duplex mode.
Reserved	25	0b	Reserved
Reserved	27:26	0x1	Reserved
Reserved	31:28	0xA	Reserved

8.12.2 Transmit Control Extended - TCTL_EXT (0x0404; R/W)

This register controls late collision detection.

The COLD field is used to determine the latest time in which a collision indication is considered as a valid collision and not a late collision. When using the internal PHY, the default value of 0x40 provides a behavior consistent with the 802.3 spec requested behavior. However, when using an SGMII connected external PHY, the SGMII interface adds some delay on top of the time budget allowed by the specification (collisions in valid network topographies even after 512 bit time can be expected). In order to accommodate this condition, COLD should be updated to take the SGMII inbound and outbound delays.

Field	Bit(s)	Initial Value	Description
Reserved	9:0	0x40	Reserved
COLD	19:10	0x42	Collision Distance Used to determine the latest time in which a collision indication is considered as a valid collision and not a late collision.
Reserved	31:20	0x0	Reserved.

8.12.3 Transmit IPG Register - TIPG (0x0410; R/W)

This register controls the Inter Packet Gap (IPG) timer.

Field	Bit(s)	Initial Value	Description
IPGT	9:0	0x08	IPG Back to Back Specifies the IPG length for back to back transmissions in both full and half duplex. Measured in increments of the MAC clock: 8 ns MAC clock when operating @ 1 Gb/s. 80 ns MAC clock when operating @ 100 Mb/s. 800 ns MAC clock when operating @ 10 Mb/s. IPGT specifies the IPG length for back-to-back transmissions in both full duplex and half duplex. Note that an offset of 4 byte times is added to the programmed value to determine the total IPG. As a result, a value of 8 is recommended to achieve a 12 byte time IPG.
IPGR1	19:10	0x04	IPG Part 1 Specifies the portion of the IPG in which the transmitter defers to receive events. IPGR1 should be set to 2/3 of the total effective IPG (8). Measured in increments of the MAC clock: 8 ns MAC clock when operating @ 1 Gb/s. 80 ns MAC clock when operating @ 100 Mb/s 800 ns MAC clock when operating @ 10 Mb/s.
IPGR	29:20	0x06	IPG After Deferral Specifies the total IPG time for non back-to-back transmissions (transmission following deferral) in half duplex. Measured in increments of the MAC clock: 8 ns MAC clock when operating @ 1 Gb/s. 80 ns MAC clock when operating @ 100 Mb/s 800 ns MAC clock when operating @ 10 Mb/s. An offset of 5-byte times must be added to the programmed value to determine the total IPG after a defer event. A value of 7 is recommended to achieve a 12-byte effective IPG. Note that the IPGR must never be set to a value greater than IPGT. If IPGR is set to a value equal to or larger that IPGT, it overrides the IPGT IPG setting in half duplex resulting in inter-packet gaps that are larger then intended by IPGT. In this case, full duplex is unaffected and always relies on IPGT.
Reserved	31:30	00b	Reserved Read as 0b. Should be written with 0b for future compatibility.

8.12.4 Retry Buffer Control – RETX_CTL (0x041C; RW)

This register controls the collision retry buffer.

Field	Bit(s)	Initial Value	Description
Water Mark	3:0	0x3	Retry buffer water mark. This parameters defines the minimal number of QWords that should be present in the retry buffer before transmission is started.
Reserved	31:4	0x0	Reserved.

8.12.5 DMA Tx Control - DTXCTL (0x3590; R/W)

This register is used to set some parameters controlling the DMA Tx behavior.

Field	Bit(s)	Initial Value	Description
Reserved	1:0	0b	Reserved.
8023LL	2	1b	802.3 length location 1b = The location of the 802.3 length field in 802.3+SNAP packets, is assumed to be 8 bytes before the end of the MAC header. 0b = The location of the 802.3 length field in 802.3+SNAP packets, is calculated from the beginning of the MAC header assuming no VLAN present in the packet sent by the software. This bit is used only in case of large send (TSO) with SNAP mode.
Reserved	3	0b	Reserved
OutOfSyncEnable	4	0b	0b = Out Of sync mechanism is disabled. 1b = Out Of sync mechanism is enabled.
MDP_EN	5	0b	Malicious driver protection enable 0b = mechanism is disabled. 1b = mechanism is enabled.
Reserved	19:6	0x0	Reserved
Cswthresh	25:20	0x4	Context switch threshold. Defines the amount of back to back transmit context descriptors above which the driver will be considered as malicious. Maximum value should be less than 19.
Reserved	31:26	0x0	Reserved

8.12.6 DMA TX TCP Flags Control Low - DTXTCPFLGL (0x359C; RW)

This register holds the buses that “AND” the control flags in TCP header for the first and middle segments of a TSO packet. See Section 7.2.4.7.1 and Section 7.2.4.7.2 for details on the use of this register.

Field	Bit(s)	Initial Value	Description
TCP_flg_first_seg	11:0	0xFF6	TCP Flags first segment. Bits that are used to execute an AND operation with the TCP flags in the TCP header in the first segment
Reserved	15:12	0x00	Reserved
TCP_Flg_mid_seg	27:16	0xF76	TCP Flags middle segments. Bits that are used to execute an AND operation with the TCP flags in the TCP header in the middle segments
Reserved	31:28	0x00	Reserved

8.12.7 DMA TX TCP Flags Control High - DTXTCPFLGH (0x35A0; RW)

This register holds the buses that “AND” the control flags in TCP header for the last segment of a TSO packet. See Section 7.2.4.7.3 for details of use of this register

Field	Bit(s)	Initial Value	Description
TCP_Flg_lst_seg	11:0	0xF7F	TCP Flags last segment. Bits that are used to execute an AND operation with the TCP flags at TCP header in the last segment
Reserved	31:12	0x00	Reserved.

8.12.8 DMA TX Max Total Allow Size Requests - DTXMXSZRQ (0x3540; RW)

This register limits the allowable size of concurrent outstanding TX read requests from the host memory on the PCIe. Limiting the size of concurrent outstanding PCIe requests allows low latency packet read requests to be serviced in a timely manner, as the low latency request is serviced right after current outstanding requests are completed.

Note: Register is shared by all functions.

Field	Bit(s)	Initial Value	Description
Max_bytes_num_req	11:0	0x10	Maximum allowable size of concurrent TX outstanding requests on PCIe. Field defines maximum size in 256 byte resolution of outstanding TX requests to be sent on PCIe. If total amount of outstanding TX requests is higher than defined in this field, no further TX outstanding requests are sent.
Reserved	31:12	0x0	Reserved

8.12.9 DMA TX Maximum Packet Size - DTXMXPKTSZ (0x355C; RW)

This register limits the total number of data bytes that might be transmitted in a single frame. Reducing packet size enables better utilization of transmit buffer.

Field	Bit(s)	Initial Value	Description
MAX_TPKT_SIZE	8:0	0x98	Maximum transmit packet size that is allowed to be transmitted by the driver. Value entered is in 64 Bytes resolution. Default value enables transmission of maximum sized 9728 Byte Jumbo frames.
Reserved	31:9	0x0	Reserved

8.12.10 Transmit Descriptor Base Address Low - TDBAL (0xE000 + 0x40*n [n=0...7]; R/W)

These registers contain the lower 32 bits of the 64-bit descriptor base address. The lower 7 bits are ignored. The Transmit Descriptor Base Address must point to a 128-byte aligned block of data.

Field[1]	Bit(s)	Initial Value	Description
Lower_0	6:0	0x0	Ignored on writes. Returns 0x0 on reads.
TDBAL	31:7	X	Transmit Descriptor Base Address Low

[1] Software should program *TDBAL[n]* register only when queue is disabled (*TXDCTL[n].Enable* = 0).

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x3800, 0x3900, 0x3A00 & 0x3B00 respectively.

8.12.11 Transmit Descriptor Base Address High - TDBAH (0xE004 + 0x40*n [n=0...7]; R/W)

These registers contain the upper 32 bits of the 64-bit descriptor base address.

Field[1]	Bit(s)	Initial Value	Description
TDBAH	31:0	X	Transmit Descriptor Base Address [63:32]

[1] Software should program *TDBAH[n]* register only when queue is disabled (*TXDCTL[n].Enable* = 0).

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x3804, 0x3904, 0x3A04 & 0x3B04 respectively.

8.12.12 Transmit Descriptor Ring Length - TDLEN (0xE008 + 0x40*n [n=0...7]; R/W)

These registers contain the descriptor ring length. The registers indicates the length in bytes and must be 128-byte aligned.

Field[1]	Bit(s)	Initial Value	Description
0	6:0	0x0	Ignore on writes. Reads back as 0b.
LEN	19:7	0x0	Descriptor Ring Length (number of 8 descriptor sets). Note: maximum allowed value in TDLEN field 19:0 is 0x80000 (32K descriptors).
Reserved	31:20	0x0	Reserved Reads as 0b. Should be written to 0b.

[1] Software should program *TDLEN[n]* register only when queue is disabled (*TXDCTL[n].Enable* = 0).

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x3808, 0x3908, 0x3A08 & 0x3B08 respectively.

8.12.13 Transmit Descriptor Head - TDH (0xE010 + 0x40*n [n=0...7]; RO)

These registers contain the head pointer for the transmit descriptor ring. It points to a 16-byte datum. Hardware controls this pointer.

Note: The values in these registers might point to descriptors that are still not in host memory. As a result, the host cannot rely on these values in order to determine which descriptor to release.

Field	Bit(s)	Initial Value	Description
TDH	15:0	0x0	Transmit Descriptor Head
Reserved	31:16	0x0	Reserved Should be written to 0b.

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x3810, 0x3910, 0x3A10 & 0x3B10 respectively.

8.12.14 Transmit Descriptor Tail - TDT (0xE018 + 0x40*n [n=0...7]; R/W)

These registers contain the tail pointer for the transmit descriptor ring and points to a 16-byte datum. Software writes the tail pointer to add more descriptors to the transmit ready queue. Hardware attempts to transmit all packets referenced by descriptors between head and tail.

Field	Bit(s)	Initial Value	Description
TDT	15:0	0x0	Transmit Descriptor Tail
Reserved	31:16	0x0	Reserved Reads as 0b. Should be written to 0b for future compatibility.

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x3818, 0x3918, 0x3A18 & 0x3B18 respectively.

8.12.15 Transmit Descriptor Control - TXDCTL (0xE028 + 0x40*n [n=0...7]; R/W)

These registers control the fetching and write-back operations of transmit descriptors. The three threshold values are used to determine when descriptors are read from and written to host memory. The values are in units of descriptors (each descriptor is 16 bytes).

Since write-back of transmit descriptors is optional (under the control of *RS* bit in the descriptor), not all processed descriptors are counted with respect to *WTHRESH*. Descriptors start accumulating after a descriptor with *RS* set is processed. In addition, with transmit descriptor bursting enabled, some descriptors are written back that did not have *RS* set in their respective descriptors.

Note: When *WTHRESH* = 0, only descriptors with the *RS* bit set are written back

Field	Bit(s)	Initial Value	Description
PTHRESH	4:0	0x0	Prefetch Threshold Controls when a prefetch of descriptors is considered. This threshold refers to the number of valid, unprocessed transmit descriptors the 82580 has in its on-chip buffer. If this number drops below *PTHRESH*, the algorithm considers pre-fetching descriptors from host memory. However, this fetch does not happen unless there are at least *HTHRESH* valid descriptors in host memory to fetch. Note: When *PTHRESH* is 0x0 a Transmit descriptor fetch operation is done when any valid descriptors are available in Host memory and space is available in internal buffer.
Reserved	7:5	0x0	Reserved
HTHRESH	12:8	0x0	Host Threshold Prefetch of transmit descriptors is considered when number of valid transmit descriptors in host memory is at least *HTHRESH*. Note: *HTHRESH* should be given a non zero value each time *PTHRESH* is used.
Reserved	15:13	0x0	Reserved

Field	Bit(s)	Initial Value	Description
WTHRESH	20:16	0x0	Write-Back Threshold Controls the write-back of processed transmit descriptors. This threshold refers to the number of transmit descriptors in the on-chip buffer that are ready to be written back to host memory. In the absence of external events (explicit flushes), the write-back occurs only after at least *WTHRESH* descriptors are available for write-back. Possible values for this field are 0 to 23. Note: Since the default value for write-back threshold is 0b, descriptors are normally written back as soon as they are processed. *WTHRESH* must be written to a non-zero value to take advantage of the write-back bursting capabilities of the 82580.
Reserved	23:21	0x0	Reserved
Reserved	24	0b	Reserved
ENABLE	25	0b	Transmit Queue Enable When set, this bit enables the operation of a specific transmit queue. Setting this bit initializes the Tail and Head registers (*TDT*[n] and *TDH*[n]) of a specific queue. Until then, the state of the queue is kept and can be used for debug purposes. When disabling a queue, this bit is cleared only after all transmit activity on this queue is stopped. Note: When transmit queue is enabled and descriptors exist, descriptors and data are fetched immediately. Actual transmit activity on port starts only if the *TCTL.EN* bit is set.
SWFLSH (WC)	26	0b	Transmit Software Flush This bit enables software to trigger descriptor write-back flushing, independently of other conditions. This bit is self cleared by hardware. Bit will clear after write-back flush is triggered (may take a number of cycles). Note: When working in head write-back mode (*TDWBAL.Head_WB_En* = 1) *TDWBAL.WB_on_EITR* bit should be set for transmit descriptor flush to occur.
Priority	27	0b	Transmit queue priority 0 - Low priority 1 - High priority When set, Transmit DMA resources are always allocated to the queue before low priority queues. Arbitration between transmit queues with same priority is done in a Round Robin fashion.
HWBTHRESH	31:28	0x0	Transmit Head writeback threshold If value of field is greater than 0x0, head writeback to host will occur only when the amount of internal pending write backs exceeds this threshold. See Section 7.2.3 for additional information. Note: When activating this mode the *WB_on_EITR* bit in the *TDWBAL* register should be set to guarantee a write back after a timeout even if the threshold has not been reached.

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x3828, 0x3928, 0x3A28 and 0x3B28 respectively.

8.12.16 Tx Descriptor Completion Write–Back Address Low - TDWBAL (0xE038 + 0x40*n [n=0...7]; R/W)

Field[1]	Bit(s)	Initial Value	Description
Head_WB_En	0	0b	Head Write-Back Enable 1b = Head write-back is enabled. 0b = Head write-back is disabled. When *head_WB_en* is set, *TXDCTL.SWFLSH* is ignored and no descriptor write-back is executed.
WB_on_EITR	1	0b	When set, a head write back is done upon EITR expiration.
HeadWB_Low	31:2	0x0	Lowest 32 bits of the head write-back memory location (DWORD aligned). Last 2 bits of this field are ignored and are always read as 0.0, meaning that the actual address is QWORD aligned.

[1] Software should program *TDWBAL[n]* register only when queue is disabled (*TXDCTL[n].Enable* = 0).

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x3838, 0x3938, 0x3A38 & 0x3B38 respectively.

8.12.17 Tx Descriptor Completion Write–Back Address High - TDWBAH (0xE03C + 0x40*n [n=0...7];R/W)

Field[1]	Bit(s)	Initial Value	Description
HeadWB_High	31:0	0x0	Highest 32 bits of the head write-back memory location.

[1] Software should program *TDWBAH[n]* register only when queue is disabled (*TXDCTL[n].Enable* = 0).

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x383C, 0x393C, 0x3A3C & 0x3B3C respectively.

8.13 DCA and TPH Register Descriptions

8.13.1 Rx DCA Control Registers - RXCTL (0xC014 + 0x40*n [n=0...7]; R/W)

Note: RX data write no-snoop is activated when the NSE bit is set in the receive descriptor.

Field	Bit(s)	Initial Value	Description
RX Descriptor Fetch TPH EN	0	0b	Receive Descriptor Fetch TPH Enable When set, hardware enables TPH for all Rx descriptors fetch from memory. When cleared, hardware does not enable TPH for descriptor fetches. This bit is cleared as a default.
RX Descriptor Writeback TPH EN[1]	1	0b	Receive Descriptor Writeback TPH Enable When set, hardware enables TPH for all Rx descriptors written back into memory. When cleared, hardware does not enable TPH for descriptor write-backs. This bit is cleared as a default. The hint used is the hint set in the Socket ID field.
Rx Header TPH EN[1]	2	0b	Receive Header TPH Enable When set, hardware enables TPH for all received header buffers. When cleared, hardware does not enable TPH for Rx headers. This bit is cleared as a default. The hint used is the hint set in the Socket ID field.
Rx Payload TPH EN[1]	3	0b	Receive Payload TPH Enable When set, hardware enables TPH for all Ethernet payloads written into memory. When cleared, hardware does not enable TPH for Ethernet payloads. This bit is cleared as a default. The hint used is the hint set in the Socket ID field.
Reserved	4	0b	Reserved
RX Descriptor DCA EN[1]	5	0b	Descriptor DCA Enable When set, hardware enables DCA for all Rx descriptors written back into memory. When cleared, hardware does not enable DCA for descriptor write-backs. This bit is cleared as a default.
Rx Header DCA EN[1]	6	0b	Receive Header DCA Enable When set, hardware enables DCA for all received header buffers. When cleared, hardware does not enable DCA for Rx headers. This bit is cleared as a default.
Rx Payload DCA EN[1]	7	0b	Receive Payload DCA Enable When set, hardware enables DCA for all Ethernet payloads written into memory. When cleared, hardware does not enable DCA for Ethernet payloads. This bit is cleared as a default.
RXdescRead NSEn	8	0b	Receive Descriptor Read No Snoop Enable This bit must be reset to 0b to ensure correct functionality (Except if the software driver can guarantee the data is present in the main memory before the DMA process occurs). Note: When TPH is enabled No Snoop bit should be 0.
RXdescRead ROEn	9	1b	Receive Descriptor Read Relax Order Enable
RXdescWBNSen	10	0b	Receive Descriptor Write-Back No Snoop Enable This bit must be reset to 0b to ensure correct functionality of descriptor write-back. Note: When TPH is enabled No Snoop bit should be 0.
RXdescWBROen (RO)	11	0b	Receive Descriptor Write-Back Relax Order Enable This bit must be reset to 0b to ensure correct functionality of descriptor write-back.
RXdataWrite NSEn	12	0b	Receive Data Write No Snoop Enable (header replication: header and data) When set to 0b, the last bit of the *Packet Buffer Address* field in the advanced receive descriptor is used as the LSB of the packet buffer address (A0), thus enabling Byte alignment of the buffer. When set to 1b, the last bit of the *Packet Buffer Address* field in advanced receive descriptor is used as the No-Snoop Enabling (NSE) bit (buffer is Word aligned). If also set to 1b, the NSE bit determines whether the data buffer is snooped or not. Note: When TPH is enabled No Snoop bit should be 0.
RXdataWrite ROEn	13	1b	Receive Data Write Relax Order Enable (header replication: header and data)

Field	Bit(s)	Initial Value	Description
RxRepHeader NSEn	14	0b	Receive Replicated/Split Header No Snoop Enable This bit must be reset to 0b to ensure correct functionality of header write to host memory. Note: When TPH is enabled No Snoop bit should be 0.
RxRepHeader ROEn	15	1b	Receive Replicated/Split Header Relax Order Enable
Reserved	23:16	0b	Reserved
CPUID	31:24	0x0	Physical ID Legacy DCA capable platforms - the device driver, upon discovery of the physical CPU ID and CPU Bus ID, programs the CPUID field with the Physical CPU and Bus ID associated with this Rx queue. DCA 1.0 capable platforms - the device driver programs a value, based on the relevant APIC ID, associated with this Tx queue. See Table 3.1.3.1.2.3 for details. TPH capable platforms - the device driver programs a value, based on the relevant Socket ID, associated with this receive queue. Note that for TPH platforms, bits 31:27 of this field should always be set to zero. See Section 7.7.2 for details.

[1] Both DCA Enable bit and TPH Enable bit should not be set for the same type of traffic.

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x2814, 0x2914, 0x2A14 & 0x2B14 respectively.

8.13.2 Tx DCA Control Registers - TXCTL (0xE014 + 0x40*n [n=0...7]; R/W)

Field	Bit(s)	Initial Value	Description
Tx Descriptor Fetch TPH EN[1]	0	0b	Transmit Descriptor Fetch TPH Enable When set, hardware enables TPH for all Tx descriptors fetch from memory. When cleared, hardware does not enable TPH for descriptor fetches. This bit is cleared as a default.
Tx Descriptor Writeback TPH EN	1	0b	Transmit Descriptor Writeback TPH Enable When set, hardware enables TPH for all Tx descriptors written back into memory. When cleared, hardware does not enable TPH for descriptor write-backs. This bit is cleared as a default. The hint used is the hint set in the Socket ID field.
Reserved	2	0b	Reserved
Tx Packet TPH EN	3	0b	Transmit Packet TPH Enable When set, hardware enables TPH for all Ethernet payloads read from memory. When cleared, hardware does not enable TPH for Ethernet payloads. This bit is cleared as a default.
Reserved	4	0b	Reserved
TX Descriptor DCA EN[1]	5	0b	Descriptor DCA Enable When set, hardware enables DCA for all Tx descriptors written back into memory. When cleared, hardware does not enable DCA for descriptor write-backs. This bit is cleared as a default and also applies to head write-back when enabled.
Reserved	7:6	00b	Reserved

Field	Bit(s)	Initial Value	Description
TXdescRDNSen	8	0b	Tx Descriptor Read No Snoop Enable This bit must be reset to 0b to ensure correct functionality (unless the software device driver has written this bit with a write-through instruction). Note: When TPH is enabled No Snoop bit should be 0.
TXdescRDROEn	9	1b	Tx Descriptor Read Relax Order Enable
TXdescWBNSen	10	0b	Tx Descriptor Write-Back No Snoop Enable This bit must be reset to 0b to ensure correct functionality of descriptor write-back. Also applies to head write-back, when enabled. Note: When TPH is enabled No Snoop bit should be 0.
TXdescWBROen	11	1b	Tx Descriptor Write-Back Relax Order Enable Applies to head write-back, when enabled.
TXDataReadNSEn	12	0b	Tx Data Read No Snoop Enable Note: When TPH is enabled No Snoop bit should be 0.
TXDataReadROEn	13	1b	Tx Data Read Relax Order Enable
Reserved	23:14	0b	Reserved Write 0 ignore on read.
CPUID	31:24	0x0	Physical ID Legacy DCA capable platforms - the device driver, upon discovery of the physical CPU ID and CPU Bus ID, programs the CPUID field with the Physical CPU and Bus ID associated with this Tx queue. DCA 1.0 capable platforms - the device driver programs a value, based on the relevant APIC ID, associated with this Tx queue. See Table 3.1.3.1.2.3 for details TPH capable platforms - the device driver programs a value, based on the relevant Socket ID, associated with this transmit queue. Note that for TPH platforms, bits 31:27 of this field should always be set to zero. See Section 7.7.2 for details.

1. Both DCA Enable bit and TPH Enable bit should not be set for the same type of traffic.

Note: In order to keep compatibility with the 82575, for queues 0-3, these registers are aliased to addresses 0x3814, 0x3914, 0x3A14 & 0x3B14 respectively.

8.13.3 DCA Requester ID Information - DCA_ID (0x5B70; RO)

The DCA requester ID field, composed of Device ID, Bus #, and Function # is set up in MMIO space for software to program the DCA Requester ID Authentication register.

Field	Bit(s)	Initial Value	Description
Function Number	2:0	000b	Function Number Function number assigned to the function based on BIOS/operating system enumeration.
Device Number	7:3	0x0	Device Number Device number assigned to the function based on BIOS/operating system enumeration.
Bus Number	15:8	0x0	Bus Number Bus number assigned to the function based on BIOS/operating system enumeration.
Reserved	31:16	0x0	Reserved

8.13.4 DCA Control - DCA_CTRL (0x5B74; R/W)

This CSR is common to all functions.

Field	Bit(s)	Initial Value	Description
DCA_DIS	0	1b	DCA Disable 0b = DCA tagging is enabled. 1b = DCA tagging is disabled.
DCA_MODE	4:1	0x0	DCA Mode 000b = Legacy DCA is supported. The TAG field in the TLP header is based on the following coding: bit 0 is DCA enable; bits 3:1 are CPU ID). 001b = DCA 1.0 is supported. When DCA is disabled for a given message, the TAG field is 0000,0000b. If DCA is enabled, the TAG is set per queue as programmed in the relevant DCA Control register. All other values are undefined.
Reserved	8:5	0x0	Reserved
Desc_PH	10:9	00b	Descriptor PH - defines the PH field used when a TPH hint is given for descriptor associated traffic (descriptor fetch, descriptor write back or head write back).
Data_PH	12:11	10b	Data PH - defines the PH field used when a TPH hint is given for data associated traffic (Tx data read, Rx data write).
Reserved	31:13	0x0	Reserved

8.14 Virtualization Register Descriptions

8.14.1 VMDq Control register – VT_CTL (0x581C; R/W)

Field	Bit(s)	Initial Value	Description
Reserved	6:0	0x0	Reserved
DEF_PL	9:7	000b	Default pool - used to queue packets that did not pass any VM queuing decision.
Reserved	26:10	0x0	Reserved
FLP	27	0b	Filter local packets - filter incoming packets whose MAC source address matches one of the LAN port DA MAC addresses. If the SA of the received packet matches one of the DA in the RAH/RAL registers, then the VM tied to this DA does not receive the packet. Other VMs can still receive it.
IGMAC	28	0b	If set, MAC address is ignored during pool decision. Pooling is based on VLAN only. If this bit is set, then the *VMOLR.strvlan* should be set to the same value for all pools,
Dis_Def_Pool	29	0b	Drop if no pool is found. If this bit is asserted, then in a RX switching virtualized environment, if there is no destination pool, the packet is discarded and not sent to the default pool. Otherwise, it is sent to the pool defined by the DEF_PL field.
Rpl_En	30	0b	Replication Enable
Reserved	31	0b	Reserved

8.14.2 Malicious Driver Free Block - MDFB (0x3558; RWS)

Field	Bit(s)	Initial Value	Description
Block Queue	7:0	0x0	Indicates queue that was blocked due to malicious behavior. When bit is set, to commence activity on offending queue, Software should initiate a software reset and re-initialize all queues on the port.
Reserved	31:8	0x0	Reserved

8.14.3 Last VM Misbehavior Cause – LVMMC (0x3548; RC)

Bits in LVMMC register define the cause for blocking the malicious queue that was reported in the *MDFB.Block Queue* field.

Field	Bit(s)	Initial Value	Description
Mac Header	0	0b	Illegal MAC header size.
IPV4 Header	1	0b	Illegal IPV4 header size.
IPV6 Header	2	0b	Illegal IPV6 header size.
Wrong MAC_IP	3	0b	Wrong MAC+IP header size
TCP LSO	4	0b	Illegal TCP header was detected in a large send operation.
Reserved	5	0b	Reserved
UDP LSO	6	0b	Illegal UDP header was detected in a large send operation.
SCTP SSO	7	0b	Illegal SCTP header was detected in a single send operation.
Leg_Size	8	0b	Illegal legacy descriptor size.
Adv_Size	9	0b	Illegal advanced descriptor size.
Off_Ill	10	0b	Illegal offload request.
SCTP_aligned	11	0b	CRC request of non 4 byte aligned data.
Reserved	15:5	0b	Reserved
SSO_UDP	17	0b	Wrong parameter of headers for UDP SSO
SSO_TCP	18	0b	Wrong parameter of headers for TCP SSO
Reserved	19	0b	Reserved
DESC_TYPE	20	0b	Wrong descriptor type (other than 2,3)
Wrong_null	21	0b	Null without EOP
No EOP	22	0b	Packet without EOP (i.e. bigger than the ring size)
ILL_DBU	24	0b	Illegal DBU configuration.
Reserved	27:25	0b	Reserved Ignore on read write 0.
Mal_PF	28	0b	Malicious Driver behavior detected on current PF
Last_Q	31:29	0x0	Last queue that detected malicious behavior.

8.14.4 VM Offload register - VMOLR (0x5AD0 + 4*n [n=0...7]; RW)

This register controls the offload and queueing options applied to each pool (VM).

Field	Bit(s)	Initial Value	Description
rlpml	13:0	0x2600	Long packet size (9k default)
Reserved	15:14	0x0	Reserved
lpe	16	0b	Long packet enable

Field	Bit(s)	Initial Value	Description
Reserved	17	0b	Reserved
Reserved	23:18	0x0	Reserved
aupe	24	0b	Accept Untagged packets enable. When set, packets without VLAN tag can be forwarded to this queue, assuming they pass the MAC address queueing mechanism.
rompe	25	0b	receive overflow multicast packets - accept packets that match the MTA table.
rope	26	0b	receive overflow packets - accept packets that match the UTA table.
bam	27	0b	Broadcast accept
mpe	28	0b	multicast promiscuous
Reserved	29	0b	Reserved
strvlan	30	0b	VLAN strip
Reserved	31	1b	Reserved - must be set to one.

8.14.5 Replication Offload register - RPLOLR (0x5AF0; RW)

This register describes the off loads applied to multicast packets.

Field	Bit(s)	Initial Value	Description
Reserved	29:0	0x0	Reserved
strvlan	30	0b	VLAN strip
Reserved	31	1b	Reserved - must be set to one.

8.14.6 VLAN VM Filter - VLVF (0x5D00 + 4*n [n=0...31]; RW)

This register set describes which VLANs the local VMs are part of. Each of the 32 registers contains a VLAN tag and a list of the VMs which are part of it. Only packets with a VLAN matching one of the VLAN tags of which the VM is member of are forwarded to this VM.

Field	Bit(s)	Initial Value	Description
VLAN_Id	11:0	0x0	Defines a VLAN tag, to which each VM whose bit is set in the POOLSEL field, belongs to.
POOLSEL	19:12	0x0	Pool Select (bitmap) Field defines to which VMs a packet with the VLAN_Id should be forwarded to. A bit is allocated to each of the 8 VMs, enabling forwarding the packet with the VLAN_Id to multiple VMs.
Reserved	30:20	0x0	Reserved Write 0, ignore on read.
VI_En	31	0b	VLAN Id Enable - this filter is valid. Note: If *RCTL.VFE* is 0 all *VLVF* filters are disabled.

8.14.7 Unicast Table Array - UTA (0xA000 + 4*n [n=0...127]; R/W)

There is one register per 32 bits of the Unicast Address Table for a total of 128 registers (the UTA[127:0] designation). Software must mask to the desired bit on reads and supply a 32-bit word on writes. The first bit of the address used to access the table is set according to the *RCTL.MO* field.

Note: All accesses to this table must be 32 bit.

The lookup algorithm is the same one used for the MTA table.

This table should be zeroed by software before start of work.

Field	Bit(s)	Initial Value	Description
Bit Vector	31:0	X	Word wide bit vector specifying 32 bits in the unicast destination address filter table.

8.14.8 Storm Control control register- SCCRL (0x5DB0;RW)

Field	Bit(s)	Initial Value	Description
MDIPW	0	0b	Drop multicast packets (excluding flow control and manageability packets) if multicast threshold is exceeded in previous window
MDICW	1	0b	Drop multicast packets (excluding flow control and manageability packets) if multicast threshold is exceeded in current window
BDIPW	2	0b	Drop broadcast packets (excluding flow control and manageability packets) if broadcast threshold is exceeded in previous window
BDICW	3	0b	Drop broadcast packets (excluding flow control and manageability packets) if broadcast threshold is exceeded in current window
BIDU	4	0b	BSC Includes Destination Unresolved packets: If bit is set, unicast received packets with no destination pool and sent to the default pool are included in IBSC
RSVD	7:5	0x0	Reserved
INTERVAL	17:8	0x8	BSC/MSC Time-interval-specification: The interval size for applying Ingress Broadcast or Multicast Storm Control. Interrupt decisions are made at the end of each interval (and most flags are also set at interval end). Setting this field resets the counter.
RSVD	31:18	0x0	Reserved

8.14.9 Storm Control status - SCSTS (0x5DB4;RO)

Field	Bit(s)	Initial Value	Description
BSCA	0	0b	Broadcast storm control active
BSCAP	1	0b	Broadcast storm control active in previous window
MSCA	2	0b	Multicast storm control active
MSCAP	3	0b	Multicast storm control active in previous window
RSVD	31:4	0x0	Reserved

8.14.10 Broadcast Storm control Threshold - BSCTRH (0x5DB8;RW)

Field	Bit(s)	Initial Value	Description
UTRESH	18:0	0x0	Traffic Upper Threshold-size: Represents the upper threshold for broadcast storm control.
RSVD	31:19	0x0	Reserved

8.14.11 Multicast Storm control Threshold - MSCTRH (0x5DBC; RW)

Field	Bit(s)	Initial Value	Description
UTRESH	18:0	0x0	Traffic Upper Threshold-size: Represents the upper threshold for multicast storm control.
RSVD	31:19	0x0	Reserved

8.14.12 Broadcast Storm Control Current Count - BSCCNT (0x5DC0;RO)

Field	Bit(s)	Initial Value	Description
CCOUNT	24:0	0x0	IBSC Traffic Current Count: Represents the count of broadcast traffic received in the current time interval in units of 64-byte segments.
RSVD	31:25	0x0	Reserved.

8.14.13 Multicast Storm control Current Count - MSCCNT (0x5DC4;RO)

Field	Bit(s)	Initial Value	Description
CCOUNT	24:0	0x0	IMSC Traffic Current Count: Represents the count of multicast traffic received in the current time interval in units of f 64-byte segments.
RSVD	31:25	0x0	Reserved.

8.14.14 Storm Control Time Counter - SCTC (0x5DC8; RO)

This register keeps track of the number of time units elapsed since the end of last time interval.

Field	Bit(s)	Initial Value	Description
COUNT	9:0	0x0	SC Time Counter: The counter for number of time units elapsed since the end of the last time interval.
RSVD	31:10	0x0	Reserved.

8.14.15 Storm Control Basic interval- SCBI (0x5DCC; RW)

This register defines the basic interval used as the base for the SCCRL.Interval counting in 10 Mb/s speed. This register is defined in 16 ns clock cycles. The interval in 1000/100 is 100 or 10 time smaller respectively.

Field	Bit(s)	Initial Value	Description
BI	24:0	0x5F5E10	Basic interval
RSVD	31:25	0x0	Reserved.

8.14.16 Virtual Mirror rule control - VMRCTL (0x5D80 + 0x4*n [n= 0...3]; RW)

This register controls the rules to be applied and the destination port.

Field	Bit(s)	Initial Value	Description
VPME	0	0b	Virtual pool mirroring enable- reflects all the packets sent to a set of given VMs.
UPME	1	0b	Uplink port mirroring enable - reflects all the traffic received from the network.
Reserved	2	0b	Reserved
VLME	3	0b	VLAN mirroring enable - reflects all the traffic received in a set of given VLANs. bit enables mirroring operation based
Reserved	7:4	0x0	Reserved
MP	10:8	0x0	VM Mirror port destination. Packets destined to certain VLAN groups, are mirrored to the queue defined by the MP field, according to the *VMRVLAN* register. Packets destined to certain VMs, are mirrored to the queue defined by the MP field, according to the *VMRVM* register. Note: If the VMRCTL.UPME bit is set to 1 all packets received on the port will be forwarded to the queue defined in the MP field.
Reserved	31:11	0x0	Reserved

8.14.17 Virtual Mirror rule VLAN - VMRVLAN (0x5D90 + 0x4*n [n= 0...3]; RW)

This register controls the VLAN ports as listed in the *VLVF* table taking part in the VLAN mirror rule.

Field	Bit(s)	Initial Value	Description
VLAN	31:0	0x0	Bitmap listing that defines which of the 32 VLANs defined in the *VLVF* registers participate in the mirror rule. Packets that have a matching Vlan_ID as defined by the *VLVF* registers will also be forwarded (mirrored) to the queue defined in the *VMRCTL.MP* field, if the *VMRCTL.VLME bit* is set to 1.

8.14.18 Virtual Mirror rule VM - VMRVM (0x5DA0 + 0x4*n [n= 0...3]; RW)

This register controls the VLAN ports as listed in the *VLVF* table taking part in the VLAN mirror rule.

Field	Bit(s)	Initial Value	Description
VM	7:0	0x0	Bitmap listing of VMs participating in the mirror rule. Packets that are forwarded to the queues defined in the *VMRVM.VM* field, will also be forwarded (mirrored) to the queue defined in the *VMRCTL.MP* field, if the *VMRCTL.VPME bit* is set to 1.
Reserved	31:8	0x0	Reserved

8.15 Power Management Registers Description

Following registers enable control of various DMA power saving features.

8.15.1 DMA Coalescing Control Register - DMACR (0x2508; R/W)

Field	Bit(s)	Initial Value	Description
DMACWT	13:0	0x0	DMA Coalescing Watchdog Timer This value sets the upper limit in 32 µsec units between arrival of receive packet or interrupt cause and initiation of PCIe access to service the received packet or interrupt cause.
Reserved	15:14	0x0	Reserved Write 0 ignore on read.
DMACTHR	23:16	0x0	DMA Coalescing Receive Threshold This value defines the DMA coalescing Receive threshold in 1 Kilobyte units. When amount of data in internal receive buffer exceeds *DMACTHR* value, DMA coalescing is stopped and PCIe moves to L0 state. **Notes:** 1. Value should be lower than *FCRTC.RTH_Coal* threshold value, to avoid needless generation of flow control packets when in DMA coalescing operating mode. 2. Receive threshold size should be smaller than internal receive buffer area reported in *IRPBS.RXPbsize* field.
Reserved	26:24	0x0	Reserved Write 0 ignore on read.
Reserved	27:24	0x0	Reserved Write 0 ignore on read.

Field	Bit(s)	Initial Value	Description
DMAC_Lx	29:28	10b	Move to Lx low power link state when no PCIe transactions 00b – Stay in L0. 01b – Move to L0s when no PCIe transactions 10b - Move to L1 when no PCIe transactions 11b - Reserved When DMA coalescing is enabled (*DMACR.DMAC_EN* = 1) value of field should be 10b.
Reserved	30	0b	Reserved Write 0 ignore on read.
DMAC_EN	31	0b	DMA Coalescing Enable 0 - Disable DMA Coalescing 1 - Enable DMA Coalescing

8.15.2 DMA Coalescing Transmit Threshold - DMCTXTH (0x3550;RW)

Field	Bit(s)	Initial Value	Description
DMCTTHR	11:0	0xE4	DMA Coalescing Transmit Threshold This value defines the DMA coalescing transmit threshold in 64 byte units. When amount of empty space in internal transmit buffer exceeds *DMCTTHR* value and additional transmit data is available in main memory, DMA coalescing is stopped and PCIe moves to L0 state. **Notes**: 1. If value is 0x0 DMA Coalescing transmit threshold mechanism is disabled. 2. Transmit threshold size should be smaller than Internal Transmit Buffer area reported in *ITPBS.TXPbsize* field.
RSVD	31:12	0b	Reserved

8.15.3 DMA Coalescing Time to Lx Request - DMCTLX (0x2514;RW)

This CSR is common to all functions.

Field	Bit(s)	Initial Value	Description
TTLX	11:0	0x0	Time to LX request: Represents the time between detection of low power Link condition to actual request. Timer counts is in 1µsec intervals.
RSVD	31:12	0x0	Reserved

8.15.4 DMA Coalescing Receive Packet Rate Threshold - DMCRTRH (0x5DD0;RW)

Field	Bit(s)	Initial Value	Description
UTRESH	18:0	0x0	Receive Traffic Threshold-size: Represents the upper threshold for RX packet rate. Packet rate below this value will not allow entering DMA coalescing. Threshold is measured in 64 Byte chunks of data received during the interval defined in the *SCCRL.INTERVAL* field using the time units defined in the *SCBI* register.
RSVD	30:19	0b	Reserved
LRPRCW (RO)	31	0b	Low Receive packet rate in current window 0b - Packet rate above *DMCRTRH.UTRESH* threshold detected. 1b - Packet rate below *DMCRTRH.UTRESH* threshold detected.

8.15.5 DMA Coalescing Current RX Count - DMCCNT (0x5DD4;RO)

Field	Bit(s)	Initial Value	Description
CCOUNT	24:0	0x0	DMA Coalescing Receive Traffic Current Count: Represents the count of receive traffic in the current time interval in units of 64-byte segments. ***Notes:*** 1. Counter does not wrap around. 2. Count stops when value defined in *DMCRTRH.UTRESH* field is reached.
Reserved	31:25	0x0	Reserved. Write 0, ignore on read.

8.15.6 Flow Control Receive Threshold Coalescing - FCRTC (0x2170; R/W)

Field	Bit(s)	Initial Value	Description
Reserved	3:0	0x0	Reserved Write 0 ignore on read.
RTH_Coal	17:4	0x0	Flow control Receive Threshold High watermark value used to generate XOFF flow control packet when executing DMA coalescing, internal transmit fifo is empty and Transmit Flow control is enabled (*CTRL.TFCE* = 1b). When previous conditions exist a XOFF packet is sent if the occupied space in the RX packet buffer is more or equal to this watermark. This field is in 16 bytes granularity. See Section 3.5.5.3.1 for calculation of *FCRTH0.RTH_Coal* value. ***Notes:*** 1. To avoid sending XOFF flow control packets needlessly when executing DMA Coalescing and internal transmit buffer is empty, value should be higher than threshold defined in *DMACR.DMACTHR* field. Maximum threshold value can be up to *FCRTH0.RTH* + maximum allowable packet size * 1.25. 2. *RTH_Coal* threshold value is used as watermark for sending flow control packets when DMA Coalescing is enabled and internal transmit buffer is empty.
Reserved	31:18	0x0	Reserved Write 0 ignore on read.

8.15.7 Latency Tolerance Reporting (LTR) Minimum Values - LTRMINV (0x5BB0; R/W)

Note:

Field	Bit(s)	Initial Value	Description
LTRV	9:0	0x5	Latency Tolerance Value Field indicates latency tolerance supported when conditions for minimum latency tolerance exist (See Section 5.8). *LTRV* values are multiplied by 32,768ns or 1,024ns depending on the Scale field, to indicate latency tolerance supported in nanoseconds. A value of 0 indicates that the device will be impacted by any delay and that best possible service is requested. the 82580 reports the same value for both Snoop and No Snoop requirements. If no memory latency requirement exists for either Snoop or No Snoop accesses the appropriate Requirement bit is cleared. Note: Software should subtract time required to move from L1 to L0 from LTR value.
Scale	12:10	011b	Latency Scale This field provides a scale for the value contained within the *LTRV* field. Encoding: 010 – *LTRV* value times 1,024ns 011 – *LTRV* value times 32,768ns Others - Reserved
Reserved	14:13	0x0	Reserved Write 0 ignore on read.

Field	Bit(s)	Initial Value	Description
LSNP Requirement	15	0b	LTR Snoop requirement 0 - No Latency requirements in Snoop memory access. 1 - Maximum latency tolerance in Snoop memory access specified in LTRV field.
Reserved	30:16	0x0	Reserved Write 0 ignore on read.
LNSNP Requirement	31	0b	LTR Non-Snoop Requirement 0 - No Latency requirements in Non-Snoop memory access. 1 - Maximum latency tolerance in Non-Snoop memory access specified in LTRV field.

8.15.8 Latency Tolerance Reporting (LTR) Maximum Values - LTRMAXV (0x5BB4; R/W)

Field	Bit(s)	Initial Value	Description
LTRV	9:0	0x5	Latency Tolerance Value Field indicates latency tolerance supported when conditions for maximum latency tolerance exist (See Section 5.8). *LTRV* values are multiplied by 32,768ns or 1,024ns depending on the Scale field, to indicate latency tolerance supported in nanoseconds. A value of 0 indicates that the device will be impacted by any delay and that best possible service is requested. the 82580 reports the same value for both Snoop and No Snoop requirements. If no memory latency requirement exists for either Snoop or No Snoop accesses the appropriate Requirement bit is cleared. Note: Software should subtract time required to move from L1 to L0 from LTR value.
Scale	12:10	011b	Latency Scale This field provides a scale for the value contained within the *LTRV* field. Encoding: 010 – *LTRV* value times 1,024ns 011 – *LTRV* value times 32,768ns Others - Reserved
Reserved	14:13	0x0	Reserved Write 0 ignore on read.
LSNP Requirement	15	0b	LTR Snoop requirement 0 - No Latency requirements in Snoop memory access. 1 - Maximum latency tolerance in Snoop memory access specified in LTRV field.
Reserved	30:16	0x0	Reserved Write 0 ignore on read.
LNSNP Requirement	31	0b	LTR Non-Snoop Requirement 0 - No Latency requirements in Non-Snoop memory access. 1 - Maximum latency tolerance in Non-Snoop memory access specified in LTRV field.

8.15.9 Latency Tolerance Reporting (LTR) Control - LTRC (0x01A0; R/W)

Field	Bit(s)	Initial Value	Description
Reserved	0	0b	Reserved
LTR_MIN	1	0b	LTR Send Minimum Values When set to 1 the 82580 sends a PCIe LTR message with the LTR Snoop value, LTR No-snoop value and LTR requirement bits as defined in the *LTRMINV* register. Note: To resend LTR message with minimum value, bit should be cleared and set again. LTR_MIN and LTR_MAX bits are exclusive.
LTR_MAX	2	0b	LTR Send Maximum Values When set to 1 the 82580 sends a PCIe LTR message with the LTR Snoop value, LTR No-snoop value and LTR requirement bits as defined in the *LTRMAXV* register. Note: To resend LTR message with maximum value, bit should be cleared and set again. LTR_MIN and LTR_MAX bits are exclusive.
PDLS_EN	3	1b	Port Disable LTR send enable 0 - Do not issue PCIe LTR message with requirement bits cleared on port disable. 1 - Issue PCIe LTR message with requirement bits cleared on port disable.
LNKDLS_EN	4	1b	Link Disconnect LTR send enable 0 - Do not issue PCIe LTR message with requirement bits cleared on link disconnect. 1 - Issue PCIe LTR message with requirement bits cleared on link disconnect.
Reserved	31:5	0x0	Reserved Write 0 ignore on read.

8.16 Timer Registers Description

8.16.1 Watchdog Setup - WDSTP (0x1040; R/W)

Field	Bit(s)	Initial Value	Description
WD_Enable	0	0b[1]	Enable Watchdog Timer
WD_Timer_ Load_enable (SC)	1	0b	Enables the load of the watchdog timer by writing to WD_Timer field. If this bit is not set, the WD_Timer field is loaded by the value of WD_Timeout. Note: Writing to this field is only for DFX purposes.
Reserved	15:2	0x0	Reserved
WD_Timer (RWS)	23:16	WD_Timeout	Indicates the current value of the timer. Resets to the timeout value each time the 82580 functional bit in Software Device Status register is set. If this timer expires, the WD interrupt to the firmware and the WD SDP is asserted. As a result, this timer is stuck at zero until it is re-armed. Note: Writing to this field is only for DFX purposes.
WD_Timeout	31:24	0x0[1]	Defines the number of seconds until the watchdog expires. The granularity of this timer is 1 sec. The minimal value allowed for this register when the watchdog mechanism is enabled is two. Setting this field to 1b might cause the watchdog to expire immediately. Note: Only 4 LSB bits loaded from EEPROM. Initial value of 4 MSB bits is 0000b.

[1] Value read from the EEPROM.

8.16.2 Watchdog Software Device Status - WDSWSTS (0x1044; R/W)

Field	Bit(s)	Initial Value	Description
Dev_Functional (SC)	0	0b	Each time this bit is set, the watchdog timer is re-armed. This bit is self clearing
Force_WD (SC)	1	0b	Setting this bit causes the WD timer to expire immediately. The WD_timer field is set to 0b. It can be used by software in order to indicate some fatal error detected in the software or in the hardware. This bit is self clearing.
Reserved	23:2	0x0	Reserved
Stuck Reason	31:24	0x0	This field can be used by software to indicate to the firmware the reason the 82580 is malfunctioning. The encoding of this field is software/firmware dependent. A value of 0 indicates a functional the 82580.

8.16.3 Free Running Timer - FRTIMER (0x1048; RWS)

This register reflects the value of a free running timer that can be used for various timeout indications. The register is reset by a PCI reset and/or software reset.

Note: Writing to this register is for DFX purposes only.

Field	Bit(s)	Initial Value	Description
Microsecond	9:0	X	Number of microseconds in the current millisecond.
Millisecond	19:10	X	Number of milliseconds in the current second.
Seconds	31:20	X	Number of seconds from the timer start (up to 4095 seconds).

8.16.4 TCP Timer - TCPTIMER (0x104C; R/W)

Field	Bit(s)	Initial Value	Description
Duration	7:0	0x0	Duration Duration of the TCP interrupt interval in msec.
KickStart (WS)	8	0b	Counter Kick-Start Writing a 1b to this bit kick-starts the counter down-count from the initial value defined in the *Duration* field. Writing a 0b has no effect.
TCPCountEn	9	0b	TCP Count Enable 1b = TCP timer counting enabled. 0b = TCP timer counting disabled. Once enabled, the TCP counter counts from its internal state. If the internal state is equal to 0b, the down-count does not restart until KickStart is activated. If the internal state is not 0b, the down-count continues from internal state. This enables a pause in the counting for debug purpose.

Field	Bit(s)	Initial Value	Description
TCPCountFinish (WS)	10	0b	TCP Count Finish This bit enables software to trigger a TCP timer interrupt, regardless of the internal state. Writing a 1b to this bit triggers an interrupt and resets the internal counter to its initial value. Down-count does not restart until either KickStart is activated or Loop is set. Writing a 0b has no effect.
Loop	11	0b	TCP Loop When set to 1b, the TCP counter reloads duration each time it reaches zero, and continues down-counting from this point without kick-starting. When set to 0b, the TCP counter stops at a zero value and does not re-start until KickStart is activated. Note: Setting this bit alone is not enough to start the timer activity. The KickStart bit should also be set.
Reserved	31:12	-	Reserved

8.17 Time Sync Register Descriptions

8.17.1 RX Time Sync Control register - TSYNCRXCTL (0xB620;RW)

Field	Bit(s)	Initial Value	Description
RXTT(RO/V)	0	0x0	Rx timestamp valid Bit is set when a valid value for Rx timestamp is captured in the Rx timestamp registers. Bit is cleared by read of Rx timestamp high register (*RXSTMPH*)).
Type	3:1	0x0	Type of packets to timestamp - 000b – time stamp L2 (V2) packets only (Sync or Delay_req depends on message type in Section 8.17.26 and packets with Pdelay_Req and Pdelay_Resp message ID values) 001b – time stamp L4 (V1) packets only (Sync or Delay_req depends on message type in Section 8.17.26) 010b – time stamp V2 (L2 and L4) packets (Sync or Delay_req depends on message type in Section 8.17.26 and packets with Pdelay_Req and Pdelay_Resp message ID values) 100b – time stamp all packets. In this mode no locking is done to the timestamp value in the *RXSTMPL/H* timestamp registers, the RDESC.STATUS.TS bit in the receive descriptor stays 0, while the RDESC.STATUS.TSIP bit in the receive descriptor is always 1 if placing timestamp in receive buffer is enabled (See Section 7.1.10). 101b - Time stamp all packets which have a Message Type bit 3 zero, which means timestamp all event packets. This is applicable for V2 packets only. 011b, 110b and 111b – reserved Note: Field is also used for defining packets that have timestamp captured in receive buffer (See Section 7.1.10).
En	4	0b	Enable RX timestamp 0 = time stamping disabled. 1 = time stamping enabled.
RSV	31:6	0x0	Reserved

8.17.2 RX timestamp Low - RXSTMPL (0xB624; RO)

Field	Bit(s)	Initial Value	Description
RTSL	31:0	0x0	Rx timestamp LSB value Value in 1 nS resolution.

8.17.3 RX timestamp High - RXSTMPH (0xB628; RO)

Field	Bit(s)	Initial Value	Description
RTSH	7:0	0x0	Rx timestamp MSB value Value in 2^{32} nS resolution.
Reserved	31:8	0x0	Reserved.

8.17.4 RX timestamp attributes low - RXSATRL(0xB62C; RO)

Field	Bit(s)	Initial Value	Description
SourceIDL	31:0	0x0	Sourceuuid low

8.17.5 RX timestamp attributes high- RXSATRH (0xB630; RO)

Field	Bit(s)	Initial Value	Description
SourceIDH	15:0	0x0	Sourceuuid high
SequenceID	31:16	0x0	SequenceId

8.17.6 TX Time Sync Control register - TSYNCTXCTL (0xB614; RW)

Field	Bit(s)	Initial Value	Description
TXTT(RO/V)	0	0b	Transmit timestamp valid (equals 1b when a valid value for Tx timestamp is captured in the Tx timestamp register, clear by read of Tx timestamp register TXSTMPH)
RSV	3:1	0x0	Reserved
EN	4	0b	Enable Transmit timestamp 0b = time stamping disabled. 1b = time stamping enabled.
RSV	31:5	0x0	Reserved

8.17.7 TX timestamp value Low - TXSTMPL (0xB618;RO)

Field	Bit(s)	Initial Value	Description
TTSL	31:0	0x0	Transmit timestamp LSB value Value in 1 nS resolution.

8.17.8 TX Timestamp Value High - TXSTMPH(0xB61C; RO)

Field	Bit(s)	Initial Value	Description
TTSH	7:0	0x0	Transmit timestamp MSB value Value in 2^{32} nS resolution.
Reserved	31:8	0x0	Reserved Write 0 ignore on read.

8.17.9 System Time Register Residue - SYSTIMR (0xB6F8; RW)

Field	Bit(s)	Initial Value	Description
STR	31:0	0x0	System time Residue value. Value in 2^{-32} nS resolution.

8.17.10 System Time Register Low - SYSTIML (0xB600; RW)

Field	Bit(s)	Initial Value	Description
STL	31:0	0x0	System time LSB value Value in 1 nS resolution.

8.17.11 System Time Register High - SYSTIMH (0xB604; RW)

Field	Bit(s)	Initial Value	Description
STH	7:0	0x0	System time MSB value Value in 2^{32} nS resolution.
Reserved	31:8	0x0	Reserved Write 0 ignore on read.

8.17.12 Increment Attributes Register - TIMINCA (0xB608; RW)

Field	Bit(s)	Initial Value	Description
Incvalue	30:0	0x0	Increment value. Value to be added or subtracted (depending on ISGN value) from 8 nS clock cycle in resolution of 2^{-32} nS.
ISGN	31	0b	Increment sign. 0 - Each 8 nS cycle add to SYSTIM a value of 8 nS + Incvalue * 2^{-32} nS. 1 - Each 8 nS cycle add to SYSTIM a value of 8 nS -Incvalue * 2^{-32} nS.

8.17.13 Time Adjustment Offset Register Low - TIMADJL (0xB60C; RW)

Field	Bit(s)	Initial Value	Description
TADJL	31:0	0x0	Time adjustment value Low Value in 1 nS resolution.

8.17.14 Time Adjustment Offset Register High - TIMADJH (0xB610;RW)

Field	Bit(s)	Initial Value	Description
TADJH	7:0	0x0	Time adjustment value High Value in 2^{32} resolution. Software write to *TIMADJH* register increases or reduces system time value (*SYSTIMH*, *SYSTIML*), depending on *TIMADJH.Sign* bit, by the time adjustment value (*TIMADJH*, *TIMADJL*).
Reserved	30:8	0x0	Reserved. Write 0 ignore on read.
Sign	31	0b	Sign (0b=”+”, 1b =”-“)

8.17.15 TimeSync Auxiliary control register - TSAUXC (0xB640; RW)

Field	Bit(s)	Initial Value	Description
EN_TT0	0	0b	Enable target time 0. Enable bit is set by software to 1b, to enable pulse or level change generation as a function of the *TSAUXC.PLSG0* and *TSAUXC.PLSNeg0* bits. The bit is cleared by hardware when the target time is hit and Pulse or level change occurs.
EN_TT1	1	0b	Enable target time 1. Enable bit is set by software to 1b, to enable pulse or level change generation as a function of the *TSAUXC.PLSG1* and *TSAUXC.PLSNeg1* bits. The bit is cleared by hardware when the target time is hit and Pulse or level change occurs.
EN_CLK0	2	0b	Enable Configurable Frequency Clock 0 Clock is generated according to frequency defined in the *FREQOUT0* register on the SDP pin (0 to 3) that has both: 1. *TSSDP.TS_SDPx_SEL* field with a value of 10b. 2. *TSSDP.TS_SDPx_EN* value of 1b.
Reserved	3	0b	Reserved
ST0	4	0b	Start Clock 0 Toggle on Target Time 0 Enable Clock 0 toggle only after target time 0, that’s defined in the TRGTTIML0 and TRGTTIMH0 registers, has passed. The clock output is initially 0 and toggles with a frequency defined in the FREQOUT0 register.
EN_CLK1	5	0b	Enable Configurable Frequency Clock 1 Clock is generated according to frequency defined in the *FREQOUT1* register on the SDP pin (0 to 3) that has both: 1. *TSSDP.TS_SDPx_SEL* field with a value of 11b. 2. *TSSDP.TS_SDPx_EN* value of 1b.
Reserved	6	0b	Reserved

Field	Bit(s)	Initial Value	Description
ST1	7	0b	Start Clock 1 Toggle on Target Time 1 Enable Clock 1 toggle only after Target Time 1, that's defined in the *TRGTTIML1* and *TRGTTIMH1* registers, has passed. The clock output is initially 1 and toggles with a frequency defined in the *FREQOUT1* register
EN_TS0	8	0b	Enable hardware time stamp 0 Enable Time stamping occurrence of change in SDP pin into the *AUXSTMPL0* and *AUXSTMPH0* registers. SDP pin (0 to 3) is selected for time stamping, if the SDP pin is selected via the *TSSDP.AUX0_SDP_SEL* field and the *TSSDP.AUX0_TS_SDP_EN* bit is set to 1b.
AUTT0	9	0b	Auxiliary timestamp taken - cleared when read from auxiliary timestamp 0 occurred
EN_TS1	10	0b	Enable hardware time stamp 1 Enable Time stamping occurrence of change in SDP pin into the *AUXSTMPL*1 and *AUXSTMPH1* registers. SDP pin (0 to 3) is selected for time stamping, if the SDP pin is selected via the *TSSDP.AUX1_SDP_SEL* field and the *TSSDP.AUX1_TS_SDP_EN* bit is set to 1b.
AUTT1	11	0b	Auxiliary timestamp taken - cleared when read from auxiliary timestamp 1 occurred
Reserved	16:12	0x0	Reserved
PLSG0	17	0b	Use Target Time 0 to generate start of pulse and Target Time 1 to generate end of pulse. SDP pin selected to drive pulse or level change is set according to the TSSDP.TS_SDPx_SEL field with a value of 00b and TSSDP.TS_SDPx_EN bit with a value of 1b. 0 – Target Time 0 generates change in SDP level. 1 – Target time 0 generates start of pulse on SDP pin. Note: Pulse or level change is generated when *TSAUXC.EN_TT0* is set to 1.
— PLSNeg0	— 18	— 0b	Generate Negative pulse on Target Time 0 when PLSG0 is 1. 0 – Generate positive pulse on Target Time 0 when PLSG0 is 1. 1 – Generate Negative pulse on target time 0 when PLSG0 is 1. Note: If PLSNeg0 = 1, at start the selected SDP pin is set to 1.
— PLSG1	— 19	— 0b	Use Target Time 1 to generate start of pulse and Target Time 0 to generate end of pulse. SDP pin selected to drive pulse or level change is set according to the *TSSDP.TS_SDPx_SEL* field with a value of 01b and *TSSDP.TS_SDPx_EN* bit with a value of 1b. 0 – Target Time 1 generates change in SDP level. 1 – Target time 1 generates start of pulse on SDP pin. Note: Pulse or level change is generated when TSAUXC.EN_TT1 is set to 1.
— PLSNeg1	— 20	— 0b	Generate Negative pulse on Target Time 1 when PLSG1 is 1. 0 – Generate positive pulse on Target Time 1when PLSG1 is 1. 1 – Generate Negative pulse on target time 1 when PLSG1 is 1. Note: If PLSNeg1 = 1, at start the selected SDP pin is set to 1.
— Reserved	— 30:21	— 0x0	Reserved Write 0, ignore on read
— Disable systime	— 31	— 1b	Disable SYSTIM count operation 0b - SYSTIM timer activated 1b - SYSTIM timer disabled. Value of SYSTIMH, SYSTIML and SYSTIMR remains constant.

8.17.16 Target Time Register 0 Low - TRGTTIML0 (0xB644; RW)

Field	Bit(s)	Initial Value	Description
TTL	31:0	0x0	Target Time 0 LSB register Value in 1 nS resolution.

8.17.17 Target Time Register 0 High - TRGTTIMH0 (0xB648; RW)

Field	Bit(s)	Initial Value	Description
TTH	7:0	0x0	Target Time 0 MSB register Value in 2^{32} nS resolution.
Reserved	31:8	0x0	Reserved Write 0 ignore on read.

8.17.18 Target Time Register 1 Low - TRGTTIML1 (0xB64C; RW)

Field	Bit(s)	Initial Value	Description
TTL	31:0	0x0	Target Time 1 LSB register Value in 1 nS resolution.

8.17.19 Target Time Register 1 High - TRGTTIMH1 (0xB650; RW)

Field	Bit(s)	Initial Value	Description
TTH	7:0	0x0	Target Time 1 MSB register Value in 2^{32} nS resolution.
Reserved	31:8	0x0	Reserved Write 0 ignore on read.

8.17.20 Frequency Out 0 Control Register FREQOUT0 (0xB654; RW)

Field	Bit(s)	Initial Value	Description
CHCT	7:0	0x0	Clock Out Half Cycle Time Half Cycle time of Clock 0 in 8 nS resolution. Clock is generated on SDP pin when T*SAUXC.EN_CLK0* is set to 1. SDP pin (0 to 3) that drives Clock 0 is selected according to the *TSSDP.TS_SDPx_SEL* field that has a value of 10b and a *TSSDP.TS_SDPx_EN* value of 1b. If T*SAUXC*.ST0 is set to 1, start of clock toggle is defined by Target Time 0 (*TRGTTIML0* and *TRGTTIMH0*) registers. Notes: 1. Setting this register to zero while using the frequency out feature, is illegal. 2. Clock 0 generation should not be enabled so long as absolute value of SYSTIM error correction using the *TRGTTIMH0* and *TRGTTIML0* registers is greater then 2 msec.
Reserved	31:8	0x0	Reserved Write 0 ignore on read.

8.17.21 Frequency Out 1 Control Register - FREQOUT1 (0xB658; RW)

Field	Bit(s)	Initial Value	Description
CHCT	7:0	0x0	Clock Out Half Cycle Time Half Cycle time of Clock 1 in 8 nS resolution. Clock is generated on SDP pin when T*SAUXC.EN_CLK1* is set to 1. SDP pin (0 to 3) that drives Clock 1 is selected according to the *TSSDP.TS_SDPx_SEL* field that has a value of 11b and a *TSSDP.TS_SDPx_EN* value of 1b. If T*SAUXC*.ST1 is set to 1, start of clock toggle is defined by Target Time 1 (*TRGTTIML1* and *TRGTTIMH1*) registers. Notes: 1. Setting this register to zero while using the frequency out feature, is illegal. 2. Clock 1 generation should not be enabled so long as absolute value of SYSTIM error correction using the *TRGTTIMH1* and *TRGTTIML1* registers is greater then 2 msec.
Reserved	31:8	0x0	Reserved Write 0 ignore on read.

8.17.22 Auxiliary Time Stamp 0 Register Low - AUXSTMPL0 (0xB65C; RO)

Field	Bit(s)	Initial Value	Description
TSTL	31:0	0x0	Auxiliary Time Stamp 0 LSB value Value in 1 nS resolution.

8.17.23 Auxiliary Time Stamp 0 Register High -AUXSTMPH0 (0xB660; RO)

Reading this register will release the value stored in AUXSTMPH/L0 and will allow time stamping of the next value.

Field	Bit(s)	Initial Value	Description
TSTH	7:0	0x0	Auxiliary Time Stamp 0 MSB value Value in 2^{32} nS resolution.
Reserved	31:8	0x0	Reserved Write 0 ignore on read.

8.17.24 Auxiliary Time Stamp 1 Register Low AUXSTMPL1 (0xB664; RO)

Field	Bit(s)	Initial Value	Description
TSTL	31:0	0x0	Auxiliary Time Stamp 1 LSB value Value in 1 nS resolution.

8.17.25 Auxiliary Time Stamp 1 Register High - AUXSTMPH1 (0xB668; RO)

Reading this register will release the value stored in AUXSTMPH/L1 and will allow stamping of the next value.

Field	Bit(s)	Initial Value	Description
TSTH	7:0	0x0	Auxiliary Time Stamp 1 MSB value Value in 2^{32} nS resolution.
Reserved	31:8	0x0	Reserved Write 0 ignore on read.

8.17.26 Time Sync RX Configuration - TSYNCRXCFG (0x5F50; R/W)

Field	Bit(s)	Initial Value	Description
CTRLT	7:0	0x0	V1 control to timestamp
MSGT	15:8	0x0	V2 Message Type to timestamp
Reserved	31:16	0x0	Reserved

8.17.27 Time Sync SDP Configuration Register - TSSDP (0x003C; R/W)

This register defines the assignment of SDP pins to the Time sync auxiliary capabilities.

Field	Bit(s)	Initial Value	Description
AUX0_SDP_SEL	1:0	00b	Select one of the SDPs to serve as the trigger for auxiliary time stamp 0 (AUXSTMPL0 and AUXSTMPH0 registers) 00b = SDP0 is assigned 01b = SDP1 is assigned 10b = SDP2 is assigned 11b = SDP3 is assigned
AUX0_TS_SDP_EN	2	0b	When set indicates that one of the SDPs can be used as an external trigger to Aux timestamp 0 (note that if this bit is set to one of the SDP pins, the corresponding pin should be configured to input mode using SPD_DIR)
AUX1_SDP_SEL	4:3	00b	Select one of the SDPs to serve as the trigger for auxiliary time stamp 1 (in AUXSTMPL1 and AUXSTMPH1 registers) 00b = SDP0 is assigned 01b = SDP1 is assigned 10b = SDP2 is assigned 11b = SDP3 is assigned
AUX1_TS_SDP_EN	5	0b	When set indicates that one of the SDPs can be used as an external trigger to Aux timestamp 1 (note that if this bit is set to one of the SDP pins, the corresponding pin should be configured to input mode using SPD_DIR)
TS_SDP0_SEL	7:6	00b	SDP0 allocation to Tsync event – when TS_SDP0_EN is set, these bits select the Tsync event that is routed to SDP0. 00b = Target Time 0 is output on SDP0 01b = Target Time 1 is output on SDP0 10b = Freq Clock 0 is output on SDP0 11b = Freq Clock 1 is output on SDP0
TS_SDP0_EN	8	0b	When set indicates that SDP0 is assigned to Tsync.
TS_SDP1_SEL	10:9	00b	SDP1 allocation to Tsync event – when TS_SDP1_EN is set, these bits select the Tsync event that is routed to SDP1. 00b = Target Time 0 is output on SDP1 01b = Target Time 1 is output on SDP1 10b = Freq Clock 0 is output on SDP1 11b = Freq Clock 1 is output on SDP1
TS_SDP1_EN	11	0b	When set indicates that SDP1 is assigned to Tsync.
TS_SDP2_SEL	13:12	00b	SDP2 allocation to Tsync event – when TS_SDP2_EN is set, these bits select the Tsync event that is routed to SDP2. 00b = Target Time 0 is output on SDP2 01b = Target Time 1 is output on SDP2 10b = Freq Clock 0 is output on SDP2 11b = Freq Clock 1 is output on SDP2
TS_SDP2_EN	14	0b	When set indicates that SDP2 is assigned to Tsync.
TS_SDP3_SEL	16:15	00b	SDP3 allocation to Tsync event – when TS_SDP3_EN is set, these bits select the Tsync event that is routed to SDP3. 00b = Target Time 0 is output on SDP3 01b = Target Time 1 is output on SDP3 10b = Freq Clock 0 is output on SDP3 11b = Freq Clock 1 is output on SDP3
TS_SDP3_EN	17	0b	When set indicates that SDP3 is assigned to Tsync.
Reserved	31:18	0x0	Reserved

8.18 PCS Register Descriptions

These registers are used to configure the SerDes, SGMII and 1000BASE-KX PCS logic. Usage of these registers is described in Section 3.5.4.1 & Section 3.5.4.3.

8.18.1 PCS Configuration - PCS_CFG (0x4200; R/W)

Field	Bit(s)	Initial Value	Description
Reserved	2:0	000b	Reserved
PCS Enable	3	1b	PCS Enable Enables the PCS logic of the MAC. Should be set in SGMII, 1000BASE-KX and SerDes mode for normal operation. Clearing this bit disables RX/TX of both data and control codes. Use this to force link down at the far end.
Reserved	29:4	0x0	Reserved
PCS Isolate	30	0b	PCS Isolate Setting this bit isolates the PCS logic from the MAC's data path. PCS control codes are still sent and received.
SRESET	31	0b	Soft Reset Setting this bit puts all modules within the MAC in reset except the Host Interface. The Host Interface is reset via HRST. This bit is NOT self clearing; GMAC is in a reset state until this bit is cleared.

8.18.2 PCS Link Control - PCS_LCTL (0x4208; RW)

Field	Bit(s)	Initial Value	Description
FLV	0	0b	Forced Link Value This bit denotes the link condition when force link is set. 0b = Forced link down. 1b = Forced link up.
FSV	2:1	10b	Forced Speed Value These bits denote the speed when force speed and duplex is set. This value is also used when AN is disabled or when in SerDes mode. 00b = 10 Mb/s (SGMII). 01b = 100 Mb/s (SGMII). 10b = 1000 Mb/s (SerDes/SGMII/1000BASE-KX). 11b = Reserved.
FDV	3	1b	Forced Duplex Value This bit denotes the duplex mode when force speed and duplex is set. This value is also used when AN is disabled or when in SerDes mode. 1b = Full duplex (SerDes/SGMII/1000BASE-KX). 0b = Half duplex (SGMII).
FSD	4	0b	Force Speed and Duplex If this bit is set, then speed and duplex mode is forced to forced speed value and forced duplex value, respectively. Otherwise, speed and duplex mode are decided by internal AN/SYNC state machines.

Field	Bit(s)	Initial Value	Description
Force Link	5	0b	Force Link If this bit is set, then the internal LINK_OK variable is forced to forced link value (bit 0 of this register). Otherwise, LINK_OK is decided by internal AN/SYNC state machines.
LINK LATCH LOW (LL)	6	0b	Link Latch Low Enable If this bit is set, then link OK going LOW (negative edge) is latched until a processor read. Afterwards, link OK is continuously updated until link OK again goes LOW (negative edge is seen).
Force Flow Control	7	0b	0 = Flow control mode is set according to the AN process by following Table 37-4 in the IEEE 802.3 spec. 1 = Flow control is set according to FC_TX_EN / FC_RX_EN bits in CTRL register.
Reserved	15:8	-	Reserved
AN_ENABLE	16	0b[1]	AN Enable Setting this bit enables the AN process in SerDes operating mode. Note: When link-up is forced (*CTRL.SLU*=1) the AN_ENABLE bit should be 0.
AN RESTART (SC)	17	0b	AN Restart Used to reset/restart the link auto-negotiation process when using SerDes mode. Setting this bit restarts the Auto-negotiation process. This bit is self clearing.
AN TIMEOUT EN	18	1b	AN Timeout Enable This bit enables the AN Timeout feature. During AN, if the link partner does not respond with AN pages, but continues to send good IDLE symbols, then LINK UP is assumed. (This enables LINK UP condition when link partner is not AN-capable and does not affect otherwise). This bit should not be set in SGMII mode.
AN SGMII BYPASS	19	0b	AN SGMII Bypass If this bit is set, then IDLE detect state is bypassed during AN in SGMII mode. This reduces the acknowledge time in SGMII mode.
AN SGMII TRIGGER	20	0B	AN SGMII Trigger If this bit is cleared, then AN is not automatically triggered in SGMII mode even if SYNC fails. AN is triggered only in response to PHY messages or by a manual setting like changing the AN Enable/Restart bits.
Reserved	23:21	000b	Reserved
FAST LINK TIMER	24	0b	Fast Link Timer AN timer is reduced if this bit is set.
LINK OK FIX EN	25	1b	Link OK Fix Enable Control for enabling/disabling LinkOK/SyncOK fix. Should be set for normal operation.
Reserved	26	0b	Reserved
Reserved	31:27	0x0	Reserved

[1] Read from EEPROM word 0x0F, bit 11.

8.18.3 PCS Link Status - PCS_LSTS (0x420C; RO)

Field	Bit(s)	Initial Value	Description
LINK OK	0	0b	Link OK This bit denotes the current link ok status. 0b = Link down. 1b = Link up/OK.
SPEED	2:1	10b	Speed This bit denotes the current operating Speed. 00b = 10 Mb/s. 01b = 100 Mb/s. 10b = 1000 Mb/s. 11b = Reserved.
DUPLEX	3	1b	Duplex This bit denotes the current duplex mode. 1b = Full duplex. 0b = Half duplex.
SYNC OK	4	0b	Sync OK This bit indicates the current value of Sync OK from the PCS Sync state machine.
Reserved	15:5	-	Reserved
AN COMPLETE	16	0b	AN Complete This bit indicates that the AN process has completed.This bit is set when the AN process reached the Link OK state. It is reset upon AN restart or reset. It is set even if the AN negotiation failed and no common capabilities where found.
AN PAGE RECEIVED	17	0b	AN Page Received This bit indicates that a link partner's page was received during an AN process. This bit is cleared on reads.
AN TIMEDOUT	18	0b	AN Timed Out This bit indicates an AN process was timed out. Valid after the *AN Complete* bit is set.
AN REMOTE FAULT	19	0b	AN Remote Fault This bit indicates that an AN page was received with a remote fault indication during an AN process. This bit cleared on reads.
AN ERROR (RWS)	20	0B	AN Error This bit indicates that a AN error condition was detected in SerDes/SGMII mode. Valid after the *AN Complete* bit is set. AN error conditions: SerDes mode: Both nodes not Full Duplex SGMII mode: PHY is set to 1000 Mb/s Half Duplex mode. Software can also force a AN error condition by writing to this bit (or can clear a existing AN error condition). This bit is cleared at the start of AN.
Reserved	31:21	0x0	Reserved

8.18.4 AN Advertisement - PCS_ANADV (0x4218; R/W)

Field	Bit(s)	Initial Value	Description
Reserved	4:0	-	Reserved
FDCAP	5	1b	Full Duplex Setting this bit indicates that the 82580 is capable of full duplex operation. This bit should be set to 1b for normal operation.
HDCAP (RO)	6	0b	Half Duplex This bit indicates that the 82580 is capable of half duplex operation. This bit is tied to 0b because the 82580 does not support half duplex in SerDes mode.
ASM	8:7	00b[1]	Local PAUSE Capabilities The 82580's PAUSE capability is encoded in this field. 00b = No PAUSE. 01b = Symmetric PAUSE. 10b = Asymmetric PAUSE to link partner. 11b = Both symmetric and asymmetric PAUSE to the 82580.
Reserved	11:9	-	Reserved
RFLT	13:12	00b	Remote Fault The 82580's remote fault condition is encoded in this field. The 82580 might indicate a fault by setting a non-zero remote fault encoding and re-negotiating. 00b = No error, link OK. 01b = Link failure. 10b = Offline. 11b = Auto-negotiation error.
Reserved	14	-	Reserved
NEXTP	15	0b	Next Page Capable The 82580 asserts this bit to request a next page transmission. The 82580 clears this bit when no subsequent next pages are requested.
Reserved	31:16	0x0	Reserved

[1] Loaded from EEPROM word 0x0F, bits 13:12.

8.18.5 Link Partner Ability - PCS_LPAB (0x421C; RO)

Field	Bit(s)	Initial Value	Description
Reserved	4:0	0x0	Reserved
LPFD	5	0b	LP Full Duplex (SerDes) When set to 1b, the link partner is capable of full duplex operation. When set to 0b, the link partner is not capable of full duplex mode. This bit is reserved while in SGMII mode.
LPHD	6	0b	LP Half Duplex (SerDes) When set to 1b, the link partner is capable of half duplex operation. When set to 0b, the link partner is not capable of half duplex mode. This bit is reserved while in SGMII mode.

Field	Bit(s)	Initial Value	Description
LPASM	8:7	00b	LP ASMDR/LP PAUSE (SerDes) The link partner's PAUSE capability is encoded in this field. 00b = No PAUSE. 01b = Symmetric PAUSE. 10b = Asymmetric PAUSE to link partner. 11b = Both symmetric and asymmetric PAUSE to the 82580. These bits are reserved while in SGMII mode.
Reserved	9	0b	Reserved
SGMII SPEED	11:10	00b	SerDes: reserved. Speed (SGMII): Speed indication from the PHY.
PRF	13:12	00b	LP Remote Fault (SerDes) The link partner's remote fault condition is encoded in this field. 00b = No error, link ok. 10b = Link failure. 01b = Offline. 11b = Auto-negotiation error. SGMII [13]: Reserved SGMII [12]: Duplex mode indication from the PHY.
ACK	14	0b	Acknowledge (SerDes) The link partner has acknowledge page reception. SGMII: Reserved.
LPNEXTP	15	0b	LP Next Page Capable (SerDes) The link partner asserts this bit to indicate its ability to accept next pages. SGMII: Link-OK indication from the PHY.
Reserved	31:16	0x0	Reserved

8.18.6 Next Page Transmit - PCS_NPTX (0x4220; RW)

Field	Bit(s)	Initial Value	Description
CODE	10:0	0x0	Message/Unformatted Code Field The Message Field is an 11-bit wide field that encodes 2048 possible messages. Unformatted Code Field is an 11-bit wide field that might contain an arbitrary value.
TOGGLE	11	0b	Toggle This bit is used to ensure synchronization with the Link Partner during Next Page exchange. This bit always takes the opposite value of the *Toggle* bit in the previously exchanged Link Code Word. The initial value of the *Toggle* bit in the first Next Page transmitted is the inverse of bit 11 in the base Link Code Word and, therefore, can assume a value of 0b or 1b. The *Toggle* bit is set as follows: 0b = Previous value of the transmitted Link Code Word when 1b 1b = Previous value of the transmitted Link Code Word when 0b.
ACK2	12	0b	Acknowledge 2 Used to indicate that a device has successfully received its Link Partners' Link Code Word.

Field	Bit(s)	Initial Value	Description
PGTYPE	13	0b	Message/Unformatted Page This bit is used to differentiate a Message Page from an Unformatted Page. The encoding is: 0b = Unformatted page. 1b = Message page.
Reserved	14	-	Reserved
NXTPG	15	0b	Next Page Used to indicate whether or not this is the last Next Page to be transmitted. The encoding is: 0b = Last page. 1b = Additional Next Pages follow.
Reserved	31:16	-	Reserved

8.18.7 Link Partner Ability Next Page - PCS_LPABNP (0x4224; RO)

Field	Bit(s)	Initial Value	Description
CODE	10:0	-	Message/Unformatted Code Field The Message Field is an 11-bit wide field that encodes 2048 possible messages. Unformatted Code Field is an 11-bit wide field that might contain an arbitrary value.
TOGGLE	11	-	Toggle This bit is used to ensure synchronization with the Link Partner during Next Page exchange. This bit always takes the opposite value of the *Toggle* bit in the previously exchanged Link Code Word. The initial value of the *Toggle* bit in the first Next Page transmitted is the inverse of bit 11 in the base Link Code Word and, therefore, can assume a value of 0b or 1b. The *Toggle* bit is set as follows: 0b = Previous value of the transmitted Link Code Word when 1b 1b = Previous value of the transmitted Link Code Word when 0b.
ACK2	12	-	Acknowledge 2 Used to indicate that a device has successfully received its Link Partners' Link Code Word.
MSGPG	13	-	Message Page This bit is used to differentiate a Message Page from an Unformatted Page. The encoding is: 0b = Unformatted page. 1b = Message page.
ACK	14	-	Acknowledge The Link Partner has acknowledged Next Page reception.
NXTPG	15	-	Next Page Used to indicate whether or not this is the last Next Page to be transmitted. The encoding is: 0b = Last page. 1b = Additional Next Pages follow.
Reserved	31:16	-	Reserved

8.18.8 SFP I2C Command- I2CCMD (0x1028; R/W)

This register is used by software to read or write to the configuration registers in an SFP module when the *CTRL_EXT.I2C Enabled* bit is set to 1.

Note: According to the SFP specification, only reads are allowed from this interface; however, SFP vendors also provide a writable register through this interface (for example, PHY registers). As a result, write capability is also supported.

Field	Bit(s)	Initial Value	Description
DATA	15:0	X	Data In a write command, software places the data bits and then the MAC shifts them out to the I^2C bus. In a read command, the MAC reads these bits serially from the I^2C bus and then software reads them from this location. Note: **This field is read in byte order and not in word order.**
REGADD	23:16	0x0	I^2C Register Address For example, register 0, 1, 2... 255.
PHYADD	26:24	0x0	Device Address bits 3 -1 The actual address used is b{1010, PHYADD[2:0], 0}.
OP	27	0b	Op Code 0b = I^2C write. 1b = I^2C read.
Reset	28	0b	Reset Sequence If set, sends a reset sequence before the actual read or write. This bit is self clearing. A reset sequence is defined as nine consecutive stop conditions.
R	29	0b	Ready Bit Set to 1b by the 82580 at the end of the I^2C transaction. For example, indicates a read or write has completed. Reset by a software write of a command.
IReserved	30	0b	Interrupt Enable When set to 1b by software, it causes an Interrupt to be asserted to indicate the end of an I^2C cycle (*ICR.MDAC*).Reserved
E	31	0b	Error This bit set is to 1b by hardware when it fails to complete an I^2C read. Reset by a software write of a command. Note: Bit is valid only when Ready bit is set.

8.18.9 SFP I2C Parameters - I2CPARAMS (0x102C; R/W)

This register is used to set the parameters for the I^2C access to the SFP module and to allow bit banging access to the I^2C interface.

Field	Bit(s)	Initial Value	Description
Write Time	4:0	110b	Write Time Defines the delay between a write access and the next access. The value is in milliseconds. A value of zero is not valid.
Read Time	7:5	010b	Read Time Defines the delay between a read access and the next access. The value is in microseconds. A value of Zero is not valid
I2CBB_EN	8	0b	I^2C Bit Bang Enable If set, the I^2C_CLK and I^2C_DATA lines are controlled via the CLK, DATA and DATA_OE_N fields of this register. Otherwise, they are controlled by the hardware machine activated via the I2CCMD or MDIC registers.

Field	Bit(s)	Initial Value	Description
CLK	9	0b	I^{2}C Clock While in bit bang mode, controls the value driven on the I2C_CLK pad of this port.
DATA_OUT	10	0b	I^{2}C_DATA While in bit bang mode and when the DATA_OE_N field is zero, controls the value driven on the I2C_DATA pad of this port.
DATA_OE_N	11	0b	I^{2}C_DATA_OE_N While in bit bang mode, controls the direction of the I2C_DATA pad of this port. 0b = Pad is output. 1b = Pad is input.
DATA_IN (RO)	12	X	I^{2}C_DATA_IN Reflects the value of the I2C_DATA pad. While in bit bang mode when the DATA_OE_N field is zero, this field reflects the value set in the DATA_OUT field.
CLK_OE_N	13	0b	I^{2}C Clock Output Enable While in bit bang mode, controls the direction of the I2C_CLK pad of this port. 0b = Pad is output. 1b = Pad is input.
CLK_IN (RO)	14	X	I^{2}C Clock In Value Reflects the value of the I2C_CLK pad. While in bit bang mode when the CLK_OE_N field is zero, this field reflects the value set in the CLK_OUT field.
clk_stretch_dis	15	0b	0b - Enable slave clock stretching support in I^{2}C access. 1b - Disable clock stretching support in I^{2}C access.
Reserved	31:16	0x0	Reserved

8.19 Statistics Register Descriptions

All Statistics registers reset when read. In addition, they stick at 0xFFFF_FFFF when the maximum value is reached.

For the receive statistics it should be noted that a packet is indicated as received if it passes the 82580's filters and is placed into the packet buffer memory. A packet does not have to be transferred to host memory in order to be counted as received.

Due to divergent paths between interrupt-generation and logging of relevant statistics counts, it might be possible to generate an interrupt to the system for a noteworthy event prior to the associated statistics count actually being incremented. This is extremely unlikely due to expected delays associated with the system interrupt-collection and ISR delay, but might be observed as an interrupt for which statistics values do not quite make sense. Hardware guarantees that any event noteworthy of inclusion in a statistics count is reflected in the appropriate count within 1 µs; a small time-delay prior to a read of statistics might be necessary to avoid the potential for receiving an interrupt and observing an inconsistent statistics count as part of the ISR.

8.19.1 CRC Error Count - CRCERRS (0x4000; RC)

Counts the number of receive packets with CRC errors. In order for a packet to be counted in this register, it must pass address filtering and must be 64 bytes or greater (from <Destination Address> through <CRC>, inclusively) in length. If receives are not enabled, then this register does not increment.

Field	Bit(s)	Initial Value	Description
CEC	31:0	0x0	CRC error count

8.19.2 Alignment Error Count - ALGNERRC (0x4004; RC)

Counts the number of receive packets with alignment errors (the packet is not an integer number of bytes in length). In order for a packet to be counted in this register, it must pass address filtering and must be 64 bytes or greater (from <Destination Address> through <CRC>, inclusive) in length. If receives are not enabled, then this register does not increment. This register is valid only in MII mode during 10/100 Mb/s operation.

Field	Bit(s)	Initial Value	Description
AEC	31:0	0x0	Alignment error count

8.19.3 Symbol Error Count - SYMERRS (0x4008; RC)

Counts the number of symbol errors between reads. The count increases for every bad symbol received, whether or not a packet is currently being received and whether or not the link is up. When working in SerDes/SGMII/1000BASE-KX mode these statistics can be read from the SCVPC register.

Field	Bit(s)	Initial Value	Description
SYMERRS	31:0	0x0	Symbol Error Count

8.19.4 RX Error Count - RXERRC (0x400C; RC)

Counts the number of packets received in which RX_ER was asserted by the PHY. In order for a packet to be counted in this register, it must pass address filtering and must be 64 bytes or greater (from <Destination Address> through <CRC>, inclusive) in length. If receives are not enabled, then this register does not increment.

This register is not available in SerDes/SGMII/1000BASE-KX modes.

Field	Bit(s)	Initial Value	Description
RXEC	31:0	0x0	RX error count

8.19.5 Missed Packets Count - MPC (0x4010; RC)

Counts the number of missed packets. Packets are missed when the receive FIFO has insufficient space to store the incoming packet. This can be caused because of too few buffers allocated, or because there is insufficient bandwidth on the PCI bus. Events setting this counter causes *ICR.Rx Miss*, the Receiver Overrun Interrupt, to be set. This register does not increment if receives are not enabled.

These packets are also counted in the Total Packets Received register as well as in Total Octets Received.

Field	Bit(s)	Initial Value	Description
MPC	31:0	0x0	Missed Packets Count

8.19.6 Single Collision Count - SCC (0x4014; RC)

This register counts the number of times that a successfully transmitted packet encountered a single collision. This register only increments if transmits are enabled (*TCTL.EN* is set) and the 82580 is in half-duplex mode.

Field	Bit(s)	Initial Value	Description
SCC	31:0	0x0	Number of times a transmit encountered a single collision.

8.19.7 Excessive Collisions Count - ECOL (0x4018; RC)

When 16 or more collisions have occurred on a packet, this register increments, regardless of the value of collision threshold. If collision threshold is set below 16, this counter won't increment. This register only increments if transmits are enabled (*TCTL.EN* is set) and the 82580 is in half-duplex mode.

Field	Bit(s)	Initial Value	Description
ECC	31:0	0x0	Number of packets with more than 16 collisions.

8.19.8 Multiple Collision Count - MCC (0x401C; RC)

This register counts the number of times that a transmit encountered more than one collision but less than 16. This register only increments if transmits are enabled (*TCTL.EN* is set) and the 82580 is in half-duplex mode.

Field	Bit(s)	Initial Value	Description
MCC	31:0	0x0	Number of times a successful transmit encountered multiple collisions.

8.19.9 Late Collisions Count - LATECOL (0x4020; RC)

Late collisions are collisions that occur after one slot time. This register only increments if transmits are enabled (*TCTL.EN* is set) and the 82580 is in half-duplex mode.

Field	Bit(s)	Initial Value	Description
LCC	31:0	0x0	Number of packets with late collisions.

8.19.10 Collision Count - COLC (0x4028; RC)

This register counts the total number of collisions seen by the transmitter. This register only increments if transmits are enabled (*TCTL.EN* is set) and the 82580 is in half-duplex mode. This register applies to clear as well as secure traffic.

Field	Bit(s)	Initial Value	Description
CCC	31:0	0x0	Total number of collisions experienced by the transmitter.

8.19.11 Defer Count - DC (0x4030; RC)

This register counts defer events. A defer event occurs when the transmitter cannot immediately send a packet due to the medium being busy either because another device is transmitting, the IPG timer has not expired, half-duplex deferral events, reception of XOFF frames, or the link is not up. This register only increments if transmits are enabled (*TCTL.EN* is set). This counter does not increment for streaming transmits that are deferred due to TX IPG.

Field	Bit(s)	Initial Value	Description
CDC	31:0	0x0	Number of defer events.

8.19.12 Transmit with No CRS - TNCRS (0x4034; RC)

This register counts the number of successful packet transmissions in which the CRS input from the PHY was not asserted within one slot time of start of transmission from the MAC. Start of transmission is defined as the assertion of TX_EN to the PHY.

The PHY should assert CRS during every transmission. Failure to do so might indicate that the link has failed, or the PHY has an incorrect link configuration. This register only increments if transmits are enabled (*TCTL.EN* is set). This register is not valid in SGMII mode and is only valid when the 82580 is operating at half duplex.

Field	Bit(s)	Initial Value	Description
TNCRS	31:0	0x0	Number of transmissions without a CRS assertion from the PHY.

8.19.13 Host Transmit Discarded Packets by MAC Count - HTDPMC (0x403C; RC)

Field	Bit(s)	Initial Value	Description
HTDPMC	31:0	0x0	Number of packets sent by the host but discarded by the MAC

This register counts the number of packets sent by the host (and not the manageability engine) that are dropped by the MAC. This can include packets dropped because of excessive collisions or link fail events.

8.19.14 Receive Length Error Count - RLEC (0x4040; RC)

This register counts receive length error events. A length error occurs if an incoming packet passes the filter criteria but is undersized or oversized. Packets less than 64 bytes are undersized. Packets over 1518/1522/1526 bytes (according to the number of VLAN tags present) are oversized if Long Packet Enable (LPE) is 0b. If LPE is 1b, then an incoming, packet is considered oversized if it exceeds the size defined in RLPML.RLPML field.

If receives are not enabled, this register does not increment. These lengths are based on bytes in the received packet from <Destination Address> through <CRC>, inclusive.

Note: Runt packets smaller than 25 bytes may not be counted by this counter.

Field	Bit(s)	Initial Value	Description
RLEC	31:0	0x0	Number of packets with receive length errors.

8.19.15 XON Received Count - XONRXC (0x4048; RC)

This register counts the number of valid XON packets received. XON packets can use the global address, or the station address. This register only increments if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
XONRXC	31:0	0x0	Number of XON packets received.

8.19.16 XON Transmitted Count - XONTXC (0x404C; RC)

This register counts the number of XON packets transmitted. These can be either due to a full queue or due to software initiated action (using TCTL.SWXOFF). This register only increments if transmits are enabled (*TCTL.EN* is set).

Field	Bit(s)	Initial Value	Description
XONTXC	31:0	0x0	Number of XON packets transmitted.

8.19.17 XOFF Received Count - XOFFRXC (0x4050; RC)

This register counts the number of valid XOFF packets received. XOFF packets can use the global address or the station address. This register only increments if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
XOFFRXC	31:0	0x0	Number of XOFF packets received.

8.19.18 XOFF Transmitted Count - XOFFTXC (0x4054; RC)

This register counts the number of XOFF packets transmitted. These can be either due to a full queue or due to software initiated action (using TCTL.SWXOFF). This register only increments if transmits are enabled (*TCTL.EN* is set).

Field	Bit(s)	Initial Value	Description
XOFFTXC	31:0	0x0	Number of XOFF packets transmitted.

8.19.19 FC Received Unsupported Count - FCRUC (0x4058; RC)

This register counts the number of unsupported flow control frames that are received.

The FCRUC counter increments when a flow control packet is received that matches either the reserved flow control multicast address (in FCAH/L) or the MAC station address, and has a matching flow control type field match (to the value in FCT), but has an incorrect opcode field. This register only increments if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
FCRUC	31:0	0x0	Number of unsupported flow control frames received.

8.19.20 Packets Received [64 Bytes] Count - PRC64 (0x405C; RC)

This register counts the number of good packets received that are exactly 64 bytes (from <Destination Address> through <CRC>, inclusive) in length. Packets that are counted in the Missed Packet Count register are not counted in this register. Packets sent to the manageability engine are included in this counter. This register does not include received flow control packets and increments only if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
PRC64	31:0	0x0	Number of packets received that are 64 bytes in length.

8.19.21 Packets Received [65—127 Bytes] Count - PRC127 (0x4060; RC)

This register counts the number of good packets received that are 65-127 bytes (from <Destination Address> through <CRC>, inclusive) in length. Packets that are counted in the Missed Packet Count register are not counted in this register. Packets sent to the manageability engine are included in this counter. This register does not include received flow control packets and increments only if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
PRC127	31:0	0x0	Number of packets received that are 65-127 bytes in length.

8.19.22 Packets Received [128—255 Bytes] Count - PRC255 (0x4064; RC)

This register counts the number of good packets received that are 128-255 bytes (from <Destination Address> through <CRC>, inclusive) in length. Packets that are counted in the Missed Packet Count register are not counted in this register. Packets sent to the manageability engine are included in this counter. This register does not include received flow control packets and increments only if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
PRC255	31:0	0x0	Number of packets received that are 128-255 bytes in length.

8.19.23 Packets Received [256—511 Bytes] Count - PRC511 (0x4068; RC)

This register counts the number of good packets received that are 256-511 bytes (from <Destination Address> through <CRC>, inclusive) in length. Packets that are counted in the Missed Packet Count register are not counted in this register. Packets sent to the manageability engine are included in this counter. This register does not include received flow control packets and increments only if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
PRC511	31:0	0x0	Number of packets received that are 256-511 bytes in length.

8.19.24 Packets Received [512—1023 Bytes] Count - PRC1023 (0x406C; RC)

This register counts the number of good packets received that are 512-1023 bytes (from <Destination Address> through <CRC>, inclusive) in length. Packets that are counted in the Missed Packet Count register are not counted in this register. Packets sent to the manageability engine are included in this counter. This register does not include received flow control packets and increments only if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
PRC1023	31:0	0x0	Number of packets received that are 512-1023 bytes in length.

8.19.25 Packets Received [1024 to Max Bytes] Count - PRC1522 (0x4070; RC)

This register counts the number of good packets received that are from 1024 bytes to the maximum (from <Destination Address> through <CRC>, inclusive) in length. The maximum is dependent on the current receiver configuration (for example, LPE, etc.) and the type of packet being received. If a packet is counted in Receive Oversized Count, it is not counted in this register (see Section 8.19.37). This register does not include received flow control packets and only increments if the packet has passed address filtering and receives are enabled (*RCTL.RXEN* is set). Packets sent to the manageability engine are included in this counter.

Due to changes in the standard for maximum frame size for VLAN tagged frames in 802.3, the 82580 accepts packets that have a maximum length of 1522 bytes. The RMON statistics associated with this range has been extended to count 1522 byte long packets. If CTRL.EXT_VLAN is set, packets up to 1526 bytes are counted by this counter.

Field	Bit(s)	Initial Value	Description
PRC1522	31:0	0x0	Number of packets received that are 1024-Max bytes in length.

8.19.26 Good Packets Received Count - GPRC (0x4074; RC)

This register counts the number of good packets received of any legal length. The legal length for the received packet is defined by the value of Long Packet Enable (CTRL.LPE) (see Section 8.19.37). This register does not include received flow control packets and only counts packets that pass filtering. This register only increments if receives are enabled (*RCTL.RXEN* is set). This register does not count packets counted by the Missed Packet Count (MPC) register. Packets sent to the manageability engine are included in this counter.

Note: GPRC can count packets interrupted by a link disconnect although they have a CRC error.

Field	Bit(s)	Initial Value	Description
GPRC	31:0	0x0	Number of good packets received (of any length).

8.19.27 Broadcast Packets Received Count - BPRC (0x4078; RC)

This register counts the number of good (no errors) broadcast packets received. This register does not count broadcast packets received when the broadcast address filter is disabled. This register only increments if receives are enabled (*RCTL.RXEN* is set). This register does not count packets counted by the Missed Packet Count (MPC) register. Packets sent to the manageability engine are included in this counter.

Field	Bit(s)	Initial Value	Description
BPRC	31:0	0x0	Number of broadcast packets received.

8.19.28 Multicast Packets Received Count - MPRC (0x407C; RC)

This register counts the number of good (no errors) multicast packets received. This register does not count multicast packets received that fail to pass address filtering nor does it count received flow control packets. This register only increments if receives are enabled (*RCTL.RXEN* is set). This register does not count packets counted by the Missed Packet Count (MPC) register. Packets sent to the manageability engine are included in this counter.

Field	Bit(s)	Initial Value	Description
MPRC	31:0	0x0	Number of multicast packets received.

8.19.29 Good Packets Transmitted Count - GPTC (0x4080; RC)

This register counts the number of good (no errors) packets transmitted. A good transmit packet is considered one that is 64 or more bytes in length (from <Destination Address> through <CRC>, inclusively) in length. This does not include transmitted flow control packets. This register only increments if transmits are enabled (*TCTL.EN* is set). The register counts clear as well as secure packets.

Field	Bit(s)	Initial Value	Description
GPTC	31:0	0x0	Number of good packets transmitted.

8.19.30 Good Octets Received Count - GORCL (0x4088; RC)

These registers make up a 64-bit register that counts the number of good (no errors) octets received. This register includes bytes received in a packet from the <Destination Address> field through the <CRC> field, inclusive; GORCL must be read before GORCH.

In addition, it sticks at 0xFFFF_FFFF_FFFF_FFFF when the maximum value is reached. Only octets of packets that pass address filtering are counted in this register. This register does not count octets of packets counted by the Missed Packet Count (MPC) register. Octets of packets sent to the manageability engine are included in this counter. This register only increments if receives are enabled (*RCTL.RXEN* is set).

These octets do not include octets of received flow control packets.

Field	Bit(s)	Initial Value	Description
GORCL	31:0	0x0	Number of good octets received – lower 4 bytes.

8.19.31 Good Octets Received Count - GORCH (0x408C; RC)

Field	Bit(s)	Initial Value	Description
GORCH	31:0	0x0	Number of good octets received – upper 4 bytes.

8.19.32 Good Octets Transmitted Count - GOTCL (0x4090; RC)

These registers make up a 64-bit register that counts the number of good (no errors) packets transmitted. This register must be accessed using two independent 32-bit accesses; GOTCL must be read before GOTCH.

In addition, it sticks at 0xFFFF_FFFF_FFFF_FFFF when the maximum value is reached. This register includes bytes transmitted in a packet from the <Destination Address> field through the <CRC> field, inclusive. This register counts octets in successfully transmitted packets that are 64 or more bytes in length. This register only increments if transmits are enabled (*TCTL.EN* is set). The register counts clear as well as secure octets.

These octets do not include octets in transmitted flow control packets.

Field	Bit(s)	Initial Value	Description
GOTCL	31:0	0x0	Number of good octets transmitted – lower 4 bytes.

8.19.33 Good Octets Transmitted Count - GOTCH (4094; RC)

Field	Bit(s)	Initial Value	Description
GOTCH	31:0	0x0	Number of good octets transmitted – upper 4 bytes.

8.19.34 Receive No Buffers Count - RNBC (0x40A0; RC)

This register counts the number of times that frames were received when there were no available buffers in host memory to store those frames (receive descriptor head and tail pointers were equal). The packet is still received if there is space in the FIFO. This register only increments if receives are enabled (*RCTL.RXEN* is set).

This register does not increment when flow control packets are received.

Field	Bit(s)	Initial Value	Description
RNBC	31:0	0x0	Number of receive no buffer conditions.

8.19.35 Receive Undersize Count - RUC (0x40A4; RC)

This register counts the number of received frames that passed address filtering, and were less than minimum size (64 bytes from <Destination Address> through <CRC>, inclusive), and had a valid CRC. This register only increments if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
RUC	31:0	0x0	Number of receive undersize errors.

8.19.36 Receive Fragment Count - RFC (0x40A8; RC)

This register counts the number of received frames that passed address filtering, and were less than minimum size (64 bytes from <Destination Address> through <CRC>, inclusive), but had a bad CRC (this is slightly different from the Receive Undersize Count register). This register only increments if receives are enabled (*RCTL.RXEN* is set).

Field	Bit(s)	Initial Value	Description
RFC	31:0	0x0	Number of receive fragment errors.

Note: Runt packets smaller than 25 bytes may not be counted by this counter.

8.19.37 Receive Oversize Count - ROC (0x40AC; RC)

This register counts the number of received frames with valid CRC field that passed address filtering, and were greater than maximum size. Packets over 1522 bytes are oversized if LongPacketEnable (RCTL.LPE) is 0b. If LongPacketEnable is 1b, then an incoming packet is considered oversized if it exceeds the value set in the RLPML register.

In VMDq mode, a packet is counted only if it is bigger than the *VOMLR.RLPML* value for all the VMs that where supposed to receive the packet.

If receives are not enabled, this register does not increment. These lengths are based on bytes in the received packet from <Destination Address> through <CRC>, inclusive.

Note: The maximum size of a packet when LPE is 0b is fixed according to the *CTRL_EXT.EXT_VLAN* bit and the detection of a VLAN tag in the packet.

Field	Bit(s)	Initial Value	Description
ROC	31:0	0x0	Number of receive oversize errors.

8.19.38 Receive Jabber Count - RJC (0x40B0; RC)

This register counts the number of received frames that passed address filtering, and were greater than maximum size and had a bad CRC (this is slightly different from the Receive Oversize Count register).

Packets over 1518/1522/1526 bytes are oversized if LPE is 0b. If LPE is 1b, then an incoming packet is considered oversized if it exceeds RLPML.LPML bytes.

If receives are not enabled, this register does not increment. These lengths are based on bytes in the received packet from <Destination Address> through <CRC>, inclusive.

Note: The maximum size of a packet when LPE is 0b is fixed according to the CTRL_EXT.EXT_VLAN bit and the detection of a VLAN tag in the packet.

Field	Bit(s)	Initial Value	Description
RJC	31:0	0x0	Number of receive jabber errors.

8.19.39 Management Packets Received Count - MNGPRC (0x40B4; RC)

This register counts the total number of packets received that pass the management filters as described in Section 10.4. Any packets with errors are not counted, except packets that are dropped because the management receive FIFO is full.

Packets sent to both the host and the management interface are not counted by this counter.

Field	Bit(s)	Initial Value	Description
MNGPRC	31:0	0x0	Number of management packets received.

8.19.40 Management Packets Dropped Count - MPDC (0x40B8; RC)

This register counts the total number of packets received that pass the management filters as described in Section 10.4, that are dropped because the management receive FIFO is full. Management packets include any packet directed to the manageability console (for example, BMC and ARP packets).

Field	Bit(s)	Initial Value	Description
MPDC	31:0	0x0	Number of management packets dropped.

8.19.41 Management Packets Transmitted Count - MNGPTC (0x40BC; RC)

This register counts the total number of transmitted packets originating from the manageability path.

Field	Bit(s)	Initial Value	Description
MPTC	31:0	0x0	Number of management packets transmitted.

8.19.42 Total Octets Received - TORL (0x40C0; RC)

These registers make up a logical 64-bit register which counts the total number of octets received. This register must be accessed using two independent 32-bit accesses; TORL must be read before TORH. This register sticks at 0xFFFF_FFFF_FFFF_FFFF when the maximum value is reached.

All packets received have their octets summed into this register, regardless of their length, whether they are erred, or whether they are flow control packets. This register includes bytes received in a packet from the <Destination Address> field through the <CRC> field, inclusive. This register only increments if receives are enabled (*RCTL.RXEN* is set).

Note: Broadcast rejected packets are counted in this counter (as opposed to all other rejected packets that are not counted).

Field	Bit(s)	Initial Value	Description
TORL	31:0	0x0	Number of total octets received – lower 4 bytes.

8.19.43 Total Octets Received - TORH (0x40C4; RC)

Field	Bit(s)	Initial Value	Description
TORH	31:0	0x0	Number of total octets received – upper 4 bytes.

8.19.44 Total Octets Transmitted - TOTL (0x40C8; RC)

These registers make up a 64-bit register that counts the total number of octets transmitted. This register must be accessed using two independent 32-bit accesses; TOTL must be read before TOTH. This register sticks at 0xFFFF_FFFF_FFFF_FFFF when the maximum value is reached.

All transmitted packets have their octets summed into this register, regardless of their length or whether they are flow control packets. This register includes bytes transmitted in a packet from the <Destination Address> field through the <CRC> field, inclusive.

Octets transmitted as part of partial packet transmissions (for example, collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*TCTL.EN* is set).

Field	Bit(s)	Initial Value	Description
TOTL	31:0	0x0	Number of total octets transmitted – lower 4 bytes.

8.19.45 Total Octets Transmitted - TOTH (0x40CC; RC)

Field	Bit(s)	Initial Value	Description
TOTH	31:0	0x0	Number of total octets transmitted – upper 4 bytes.

8.19.46 Total Packets Received - TPR (0x40D0; RC)

This register counts the total number of all packets received. All packets received are counted in this register, regardless of their length, whether they have errors, or whether they are flow control packets. This register only increments if receives are enabled (*RCTL.RXEN* is set).

Note: Broadcast rejected packets are counted in this counter (as opposed to all other rejected packets that are not counted).

Note: Runt packets smaller than 25 bytes may not be counted by this counter.

TPR can count packets interrupted by a link disconnect although they have a CRC error.

Field	Bit(s)	Initial Value	Description
TPR	31:0	0x0	Number of all packets received.

8.19.47 Total Packets Transmitted - TPT (0x40D4; RC)

This register counts the total number of all packets transmitted. All packets transmitted are counted in this register, regardless of their length, or whether they are flow control packets.

Partial packet transmissions (collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*TCTL.EN* is set). This register counts all packets, including standard packets, secure packets, packets received over the SMBus, and packets generated by the PT function.

Field	Bit(s)	Initial Value	Description
TPT	31:0	0x0	Number of all packets transmitted.

8.19.48 Packets Transmitted [64 Bytes] Count - PTC64 (0x40D8; RC)

This register counts the number of packets transmitted that are exactly 64 bytes (from <Destination Address> through <CRC>, inclusive) in length. Partial packet transmissions (collisions in half-duplex mode) are not included in this register. This register does not include transmitted flow control packets (which are 64 bytes in length). This register only increments if transmits are enabled (*TCTL.EN* is set). This register counts all packets, including standard packets, secure packets, packets received over the SMBus, and packets generated by the PT function.

Field	Bit(s)	Initial Value	Description
PTC64	31:0	0x0	Number of packets transmitted that are 64 bytes in length.

8.19.49 Packets Transmitted [65—127 Bytes] Count - PTC127 (0x40DC; RC)

This register counts the number of packets transmitted that are 65-127 bytes (from <Destination Address> through <CRC>, inclusive) in length. Partial packet transmissions (for example, collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*TCTL.EN* is set). This register counts all packets, including standard packets, secure packets, packets received over the SMBus, and packets generated by the PT function.

Field	Bit(s)	Initial Value	Description
PTC127	31:0	0x0	Number of packets transmitted that are 65-127 bytes in length.

8.19.50 Packets Transmitted [128—255 Bytes] Count - PTC255 (0x40E0; RC)

This register counts the number of packets transmitted that are 128-255 bytes (from <Destination Address> through <CRC>, inclusive) in length. Partial packet transmissions (collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*TCTL.EN* is set). This register counts all packets, including standard packets, secure packets, packets received over the SMBus, and packets generated by the PT function.

Field	Bit(s)	Initial Value	Description
PTC255	31:0	0x0	Number of packets transmitted that are 128-255 bytes in length.

8.19.51 Packets Transmitted [256—511 Bytes] Count - PTC511 (0x40E4; RC)

This register counts the number of packets transmitted that are 256-511 bytes (from <Destination Address> through <CRC>, inclusive) in length. Partial packet transmissions (for example, collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*TCTL.EN* is set). This register counts all packets, including standard and secure packets. Management packets must never be more than 200 bytes.

Field	Bit(s)	Initial Value	Description
PTC511	31:0	0x0	Number of packets transmitted that are 256-511 bytes in length.

8.19.52 Packets Transmitted [512—1023 Bytes] Count - PTC1023 (0x40E8; RC)

This register counts the number of packets transmitted that are 512-1023 bytes (from <Destination Address> through <CRC>, inclusive) in length. Partial packet transmissions (for example, collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*TCTL.EN* is set). This register counts all packets, including standard and secure packets. Management packets must never be more than 200 bytes.

Field	Bit(s)	Initial Value	Description
PTC1023	31:0	0x0	Number of packets transmitted that are 512-1023 bytes in length.

8.19.53 Packets Transmitted [1024 Bytes or Greater] Count - PTC1522 (0x40EC; RC)

This register counts the number of packets transmitted that are 1024 or more bytes (from <Destination Address> through <CRC>, inclusive) in length. Partial packet transmissions (for example, collisions in half-duplex mode) are not included in this register. This register only increments if transmits are enabled (*TCTL.EN* is set).

Due to changes in the standard for maximum frame size for VLAN tagged frames in 802.3, the 82580 transmits packets that have a maximum length of 1522 bytes. The RMON statistics associated with this range has been extended to count 1522 byte long packets. This register counts all packets, including standard and secure packets (management packets must never be more than 200 bytes). If *CTRL.EXT_VLAN* is set, packets up to 1526 bytes are counted by this counter.

Field	Bit(s)	Initial Value	Description
PTC1522	31:0	0x0	Number of packets transmitted that are 1024 or more bytes in length.

8.19.54 Multicast Packets Transmitted Count - MPTC (0x40F0; RC)

This register counts the number of multicast packets transmitted. This register does not include flow control packets and increments only if transmits are enabled (*TCTL.EN* is set). Counts clear as well as secure traffic.

Field	Bit(s)	Initial Value	Description
MPTC	31:0	0x0	Number of multicast packets transmitted.

8.19.55 Broadcast Packets Transmitted Count - BPTC (0x40F4; RC)

This register counts the number of broadcast packets transmitted. This register only increments if transmits are enabled (*TCTL.EN* is set). This register counts all packets, including standard and secure packets (management packets must never be more than 200 bytes).

Field	Bit(s)	Initial Value	Description
BPTC	31:0	0x0	Number of broadcast packets transmitted count.

8.19.56 TCP Segmentation Context Transmitted Count - TSCTC (0x40F8; RC)

This register counts the number of TCP segmentation offload transmissions and increments once the last portion of the TCP segmentation context payload is segmented and loaded as a packet into the on-chip transmit buffer. Note that it is not a measurement of the number of packets sent out (covered by other registers). This register only increments if transmits and TCP segmentation offload are enabled.

This counter only counts pure TSO transmissions.

Field	Bit(s)	Initial Value	Description
TSCTC	31:0	0x0	Number of TCP Segmentation contexts transmitted count.

8.19.57 Interrupt Assertion Count - IAC (0x4100; RC)

This counter counts the total number of LAN interrupts generated in the system. In case of MSI-X systems, this counter reflects the total number of MSI-X messages that are emitted.

Field	Bit(s)	Initial Value	Description
IAC	31:0	0x0	This is a count of all the LAN interrupt assertions that have occurred.

8.19.58 Rx Packets to Host Count - RPTHC (0x4104; RC)

Field	Bit(s)	Initial Value	Description
RPTHC	31:0	0x0	This is a count of all the received packets sent to the host.

8.19.59 Host Good Packets Transmitted Count-HGPTC (0x4118; RC)

Field	Bit(s)	Initial Value	Description
HGPTC	31:0	0x0	Number of good packets transmitted by the host.

This register counts the number of good (non-erred) packets transmitted sent by the host. A good transmit packet is considered one that is 64 or more bytes in length (from <Destination Address> through <CRC>, inclusively) in length. This does not include transmitted flow control packets or packets sent by the manageability engine. This register only increments if transmits are enabled (*TCTL.EN* is set).

8.19.60 Receive Descriptor Minimum Threshold Count-RXDMTC (0x4120; RC)

Field	Bit(s)	Initial Value	Description
RXDMTC	31:0	0x0	This is a count of the receive descriptor minimum threshold events

This register counts the number of events where the number of descriptors in one of the Rx queues was lower than the threshold defined for this queue.

8.19.61 Host Good Octets Received Count - HGORCL (0x4128; RC)

Field	Bit(s)	Initial Value	Description
HGORCL	31:0	0x0	Number of good octets received by host – lower 4 bytes

8.19.62 Host Good Octets Received Count - HGORCH (0x412C; RC)

Field	Bit(s)	Initial Value	Description
HGORCH	31:0	0x0	Number of good octets received by host – upper 4 bytes

These registers make up a logical 64-bit register which counts the number of good (non-erred) octets received. This register includes bytes received in a packet from the <Destination Address> field through the <CRC> field, inclusive. This register must be accessed using two independent 32-bit accesses.; HGORCL must be read before HGORCH.

In addition, it sticks at 0xFFFF_FFFF_FFFF_FFFF when the maximum value is reached. Only packets that pass address filtering are counted in this register. This register counts only octets of packets that reached the host. The only exception is packets dropped by the DMA because of lack of descriptors in one of the queues. These packets are included in this counter.

This register only increments if receives are enabled (*RCTL.RXEN* is set).

8.19.63 Host Good Octets Transmitted Count - HGOTCL (0x4130; RC)

Field	Bit(s)	Initial Value	Description
HGOTCL	31:0	0x0	Number of good octets transmitted by host – lower 4 bytes

8.19.64 Host Good Octets Transmitted Count - HGOTCH (0x4134; RC)

Field	Bit(s)	Initial Value	Description
HGOTCH	31:0	0x0	Number of good octets transmitted by host – upper 4 bytes

These registers make up a logical 64-bit register which counts the number of good (non-erred) packets transmitted. This register must be accessed using two independent 32-bit accesses. This register resets whenever the upper 32 bits are read (HGOTCH).

In addition, it sticks at 0xFFFF_FFFF_FFFF_FFFF when the maximum value is reached. This register includes bytes transmitted in a packet from the <Destination Address> field through the <CRC> field, inclusive. This register counts octets in successfully transmitted packets which are 64 or more bytes in length. This register only increments if transmits are enabled (*TCTL.EN* is set). The register counts clear as well as secure octets.

These octets do not include octets in transmitted flow control packets or manageability packets.

8.19.65 Length Error Count - LENERRS (0x4138; RC)

Field	Bit(s)	Initial Value	Description
LENERRS	31:0	0x0	Length error count.

Counts the number of receive packets with Length errors. For example, valid packets (no CRC error) with a length/Type field with a value smaller or equal to 1500 greater than the frame size. In order for a packet to be counted in this register, it must pass address filtering and must be 64 bytes or greater (from <Destination Address> through <CRC>, inclusive) in length. If receives are not enabled, then this register does not increment.

8.19.66 SerDes/SGMII/KX Code Violation Packet Count - SCVPC (0x4228; RW)

This register contains the number of code violation packets received. Code violation is defined as an invalid received code in the middle of a packet.

Field	Bit(s)	Initial Value	Description
CODEVIO	31:0	0x0	Code Violation Packet Count: At any point of time this field specifies number of unknown protocol packets received. Valid only in SGMII/SerDes/1000BASE-KX modes.

8.19.67 Switch Drop Packet Count - SDPC (0x41A4; RC)

Field	Bit(s)	Initial Value	Description
SDPC	31:0	0x0	Switch Drop Packet Count: This register counts Rx packets dropped at the pool selection stage of the switch or by the storm control mechanism. For example, packets that where not routed to any of the pools and the VT_CTL.Dis_Def_Pool is set. Valid only in VMDq mode.

8.19.68 Virtualization Statistical Counters

The 82580 supports 5 statistical counters per queue to reduce processing overhead in virtualization operating mode.

8.19.68.1 Per Queue Good Packets Received Count - VFGPRC (0x10010 + n*0x100 [n=0...7]; RO)

This register counts the number of legal length good packets received in queue[n]. The legal length for the received packet is defined by the value of Long Packet Enable (*CTRL.LPE*) (see Section 8.19.37). This register does not include received flow control packets and only counts packets that pass filtering. This register only increments if receive is enabled.

Note: VFGPRC may count packets interrupted by a link disconnect although they have a CRC error.

Unlike some other statistics registers that are not allocated per VM, this register is not cleared on read. Furthermore, the register wraps around back to 0x0000 on the next increment when reaching a value of 0xFFFF and then continues normal count operation.

Field	Bit(s)	Initial Value	Description
GPRC	31:0	0x0	Number of good packets received (of any length).

8.19.68.2 Per Queue Good Packets Transmitted Count - VFGPTC (0x10014 + n*0x100 [n=0...7]; RO)

This register counts the number of good (no errors) packets transmitted on queue[n]. A good transmit packet is considered one that is 64 or more bytes in length (from <Destination Address> through <CRC>, inclusively) in length. This does not include transmitted flow control packets. This register only increments if transmits are enabled (*TCTL.EN* is set). The register counts clear as well as secure packets. This counter includes loopback packets or packets later dropped by the MAC.

Note: Unlike some other statistic registers that are not allocated per VM, this register is not cleared on read. Furthermore, the register wraps around back to 0x0000 on the next increment when reaching a value of 0xFFFF and then continues normal count operation.

Field	Bit(s)	Initial Value	Description
GPTC	31:0	0x0	Number of good packets transmitted.

8.19.68.3 Per Queue Good Octets Received Count - VFGORC (0x10018 + n*0x100 [n=0...7]; RO)

This register counts the number of good (no errors) octets received on queue[n]. This register includes bytes received in a packet from the <Destination Address> field through the <CRC> field, inclusive.

Only octets of packets that pass address filtering are counted in this register. This register only increments if receive is enabled.

Note: Unlike some other statistic registers that are not allocated per VM, this register is not cleared on read. Furthermore, the register wraps around back to 0x0000 on the next increment when reaching a value of 0xFFFF and then continues normal count operation.

Field	Bit(s)	Initial Value	Description
GORC	31:0	0x0	Number of good octets received.

8.19.68.4 Per Queue Good Octets Transmitted Count - VFGOTC (0x10034 + n*0x100 [n=0...7]; RO)

This register counts the number of good (no errors) packets transmitted on queue[n]. This register includes bytes transmitted in a packet from the <Destination Address> field through the <CRC> field, inclusive. Register also counts any padding and VLAN tag that were added by the hardware. This register counts octets in successfully transmitted packets that are 64 or more bytes in length. Octets counted do not include octets in transmitted flow control packets. This register only increments if transmit is enabled.

Note: Unlike some other statistic registers that are not allocated per VM, this register is not cleared on read. Furthermore, the register wraps around back to 0x0000 on the next increment when reaching a value of 0xFFFF and then continues normal count operation.

Field	Bit(s)	Initial Value	Description
GOTC	31:0	0x0	Number of good octets transmitted – lower 4 bytes.

8.19.68.5 Per Queue Multicast Packets Received Count - VFMPRC (0x10038 + n*0x100 [n=0...7]; RO)

This register counts the number of good (no errors) multicast packets received on queue[n]. This register does not count multicast packets received that fail to pass address filtering nor does it count received flow control packets. This register only increments if receive is enabled.

Note: Unlike some other statistic registers that are not allocated per VM, this register is not cleared on read. Furthermore, the register wraps around back to 0x0000 on the next increment when reaching a value of 0xFFFF and then continues normal count operation.

Field	Bit(s)	Initial Value	Description
MPRC	31:0	0x0	Number of multicast packets received.

8.20 Manageability statistics

This section describes a set of statistics counters used by the NC-SI interface and are not accessible to the host driver.

8.20.1 BMC Management Packets Dropped Count - BMPDC (0x4140; RC)

This register counts the total number of packets received that pass the management filters as described in Section 10.4, that are dropped because the management receive FIFO is full. Management packets include any packet directed to the manageability console (for example, BMC and ARP packets).

This register is available to the firmware only.

Field	Bit(s)	Initial Value	Description
MPDC	31:0	0x0	Number of management packets dropped.

8.20.2 BMC Management Packets Transmitted Count - BMNGPTC (0x4144; RC)

This register counts the total number of transmitted packets originating from the manageability path. This register is available to the firmware only.

Field	Bit(s)	Initial Value	Description
MPTC	31:0	0x0	Number of management packets transmitted.

8.20.3 BMC Management Packets Received Count - BMNGPRC (0x413C; RC)

This register counts the total number of packets received that pass the management filters as described in Section 10.4. Any packets with errors are not counted, except packets that are dropped because the management receive FIFO is full. This register is available to the firmware only.

Field	Bit(s)	Initial Value	Description
MNGPRC	31:0	0x0	Number of management packets received.

8.20.4 BMC Total Unicast Packets Received - BUPRC (0x4400; RC)

This register counts the number of good (no errors) unicast packets received. This register does not count unicast packets received that fail to pass address filtering. This register only increments if receives are enabled (*RCTL.RXEN* is set). This register does not count packets counted by the Missed Packet Count (MPC) register. Packets sent to the manageability engine are included in this counter.

This register is available to the firmware only.

Field	Bit(s)	Initial Value	Description
BUPRC	31:0	0x0	Number of Unicast packets received.

8.20.5 BMC Total Multicast Packets Received - BMPRC (0x4404; RC)

This register counts the same events as the MPRC register (Section 8.19.28) for the BMC usage. This register is available to the firmware only.

8.20.6 BMC Total Broadcast Packets Received - BBPRC (0x4408; RC)

This register counts the same events as the BPRC register (Section 8.19.27) for the BMC usage. This register is available to the firmware only.

8.20.7 BMC Total Unicast Packets Transmitted - BUPTC (0x440C; RC)

This register counts the number of unicast packets transmitted. This register increments only if transmits are enabled (*TCTL.EN* is set). This register is available to the firmware only

Field	Bit(s)	Initial Value	Description
BUPTC	31:0	0x0	Number of unicast packets transmitted.

8.20.8 BMC Total Multicast Packets Transmitted - BMPTC (0x4410; RC)

This register counts the same events as the *MPTC* register (Section 8.19.54) for the BMC usage. This register increments only if transmits are enabled (*TCTL.EN* is set). This register is available to the firmware only.

8.20.9 BMC Total Broadcast Packets Transmitted - BBPTC (0x4414; RC)

This register counts the same events as the *BPTC* register (Section 8.19.55) for the BMC usage. This register increments only if transmits are enabled (*TCTL.EN* is set). This register is available to the firmware only.

8.20.10 BMC FCS Receive Errors - BCRCERRS (0x4418; RC)

This register counts the same events as the *CRCERRS* register (Section 8.19.1) for the BMC usage. This register increments only if transmits are enabled (*TCTL.EN* is set). This register is available to the firmware only.

8.20.11 BMC Alignment Errors - BALGNERRC (0x441C; RC)

This register counts the same events as the *ALGNERRC* register (Section 8.19.2) for the BMC usage. This register increments only if transmits are enabled (*TCTL.EN* is set).This register is available to the firmware only.

8.20.12 BMC Pause XON Frames Received - BXONRXC (0x4420; RC)

This register counts the same events as the *XONRXC* register (Section 8.19.15) for the BMC usage. This register increments only if transmits are enabled (*TCTL.EN* is set). This register is available to the firmware only.

8.20.13 BMC Pause XOFF Frames Received - BXOFFRXC (0x4424; RC)

This register counts the same events as the *XOFFRXC* register (Section 8.19.17) for the BMC usage. This register increments only if transmits are enabled (*TCTL.EN* is set). This register is available to the firmware only.

8.20.14 BMC Pause XON Frames Transmitted - BXONTXC (0x4428; RC)

This register counts the same events as the *XONTXC* register (Section 8.19.16) for the BMC usage. This register increments only if transmits are enabled (*TCTL.EN* is set). This register is available to the firmware only.

8.20.15 BMC Pause XOFF Frames Transmitted - BXOFFTXC (0x442C; RC)

This register counts the same events as the *XOFFTXC* register (Section 8.19.18) for the BMC usage. This register increments only if transmits are enabled (*TCTL.EN* is set). This register is available to the firmware only.

8.20.16 BMC Single Collision Transmit Frames- BSCC (0x4430; RC)

This register counts the same events as the *SCC* register (Section 8.19.6) for the BMC usage. This register increments only if transmits are enabled (*TCTL.EN* is set). This register is available to the firmware only.

8.20.17 BMC Multiple Collision Transmit Frames - BMCC (0x4434; RC)

This register counts the same events as the *MCC* register (Section 8.19.8) for the BMC usage. This register increments only if transmits are enabled (*TCTL.EN* is set). This register is available to the firmware only.

8.21 Wake Up Control Register Descriptions

8.21.1 Wakeup Control Register - WUC (0x5800; R/W)

The *PME_En* and *PME_Status* bits of this register are reset when LAN_PWR_GOOD is 0b. When AUX_PWR = 0b, this register is also reset by de-asserting PE_RST_N and during a D3 to D0 transition. The other bits are reset using the standard internal resets.

Field	Bit(s)	Initial Value	Description
APME	0	0b[1]	Advance Power Management Enable If set to 1b, APM Wakeup is enabled. If this bit is set and the *APMPME* bit is cleared, reception of a magic packet asserts the *WUS.MAG* bit but does not assert a PME.
PME_En	1	0b	PME_En This read/write bit is used by the software device driver to enable generation of a PME event without writing to the Power Management Control / Status Register (*PMCSR*) in the PCIe configuration space. Note: Bit reflects value of *PMCSR.PME_En* bit when the bit in the *PMCSR* register is modified. However when value of *WUC.PME_En* bit is modified by software device driver, value is not reflected in the *PMCSR.PME_En* bit.

Field	Bit(s)	Initial Value	Description
PME_Status (R/W1C)	2	0b	PME_Status This bit is set when the 82580 receives a wakeup event. It is the same as the PME_Status bit in the Power Management Control / Status Register (PMCSR). Writing a 1b to this bit clears also the PME_Status bit in the PMCSR.
APMPME	3	0b[1]	Assert PME On APM Wakeup If set to 1b, the 82580 sets the PME_Status bit in the Power Management Control / Status Register (PMCSR) and asserts PE_WAKE_N and sends a PM_PME PCIe message when APM Wakeup is enabled (*WUC.APME* = 1) and the 82580 receives a matching Magic Packet. Note: When *WUC.APMPME* is set PE_WAKE_N is asserted and a PM_PME message is sent even if *PMCSR.PME_En* is cleared.
Reserved	31:4	0x0	Reserved

[1] Loaded from the EEPROM.

8.21.2 Wakeup Filter Control Register - WUFC (0x5808; R/W)

This register is used to enable each of the pre-defined and flexible filters for wakeup support. A value of 1b means the filter is turned on.; A value of 0b means the filter is turned off.

If the NoTCO bit is set, then any packet that passes the manageability packet filtering as described in Section 10.4, does not cause a Wake Up event even if it passes one of the Wake Up Filters. This bit is set at initialization and during any EEPROM read if the SMBus Enable bit of the EEPROM's Management Control word is 1b. Otherwise its initial value is 0b.

Field	Bit(s)	Initial Value	Description
LNKC	0	0b	Link Status Change Wakeup Enable.
MAG	1	0b	Magic Packet Wakeup Enable.
EX	2	0b	Directed Exact Wakeup Enable.[1]
MC	3	0b	Directed Multicast Wakeup Enable.
BC	4	0b	Broadcast Wakeup Enable.
ARP	5	0b	ARP Request Packet Wakeup Enable.
IPv4	6	0b	Directed IPv4 Packet Wakeup Enable.
IPv6	7	0b	Directed IPv6 Packet Wakeup Enable.
Reserved	13:8	0b	Reserved. Set these bits to 0b.
FLEX_HQ	14	0b	Flex filters Host Queuing 0 - Do not use Flex filters for queueing decisions in D0 state. 1 - Use flex filters in queuing decisions in D0 state. Note: Should be enabled only when multi queueing is enabled (MRQC.Multiple Receive Queues = 010b or 000b).
NoTCO	15	0b	Ignore TCO/management packets for wake up.
FLX0	16	0b	Flexible Filter 0 Enable.
FLX1	17	0b	Flexible Filter 1 Enable.
FLX2	18	0b	Flexible Filter 2 Enable.
FLX3	19	0b	Flexible Filter 3 Enable.
FLX4	20	0b	Flexible Filter 4 Enable.
FLX5	21	0b	Flexible Filter 5 Enable.

Field	Bit(s)	Initial Value	Description
FLX6	22	0b	Flexible Filter 6 Enable.
FLX7	23	0b	Flexible Filter 7 Enable.
Reserved	31:24	0x0	Reserved.

1. If the *RCTL.UPE* is set, and the EX bit is set also, any unicast packet wakes up the system.

8.21.3 Wakeup Status Register - WUS (0x5810; R/W1C)

This register is used to record statistics about all wakeup packets received. If a packet matches multiple criteria then multiple bits could be set. Writing a 1b to any bit clears that bit.

This register is not cleared when RST# is asserted. It is only cleared when LAN_PWR_GOOD is de-asserted or when cleared by the software device driver.

Note: If additional packets are received that matches one of the wakeup filters, after the original wakeup packet is received, the WUS register is updated with the matching filters accordingly.

Field	Bit(s)	Initial Value	Description
LNKC	0	0b	Link Status Change.
MAG	1	0b	Magic Packet Received.
EX	2	0b	Directed Exact Packet Received The packet's address matched one of the 16 pre-programmed exact values in the Receive Address registers.
MC	3	0b	Directed Multicast Packet Received The packet was a multicast packet hashed to a value that corresponded to a 1 bit in the Multicast Table Array.
BC	4	0b	Broadcast Packet Received.
ARP	5	0b	ARP 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 happened.
Reserved	15:9	0b	Reserved.
FLX0[1]	16	0b	Flexible Filter 0 Match.
FLX1[1]	17	0b	Flexible Filter 1 Match.
FLX2[1]	18	0b	Flexible Filter 2 Match.
FLX3[1]	19	0b	Flexible Filter 3 Match.
FLX4[1]	20	0b	Flexible Filter 4 Match.
FLX5[1]	21	0b	Flexible Filter 5 Match.
FLX6[1]	22	0b	Flexible Filter 6 Match.
FLX7[1]	23	0b	Flexible Filter 7 Match.
Reserved	31:24	0b	Reserved.

1. Bit is set only when flex filter match is detected and *WUFC.FLEX_HQ* is 0.

8.21.4 Wakeup Packet Length - WUPL (0x5900; RO)

This register indicates the length of the first wakeup packet received. It is valid if one of the bits in the Wakeup Status register (WUS) is set. It is not cleared by any reset.

Field	Bit(s)	Initial Value	Description
LEN	11:0	X	Length of wakeup packet. (If jumbo frames are enabled and the packet is longer than 2047 bytes then this field is 2047.)
Reserved	31:12	0x0	Reserved

8.21.5 Wakeup Packet Memory - WUPM (0x5A00 + 4*n [n=0...31]; RO)

This register is read-only and it is used to store the first 128 bytes of the wakeup packet for software retrieval after system wakeup. It is not cleared by any reset.

Field	Bit(s)	Initial Value	Description
WUPD	31:0	X	Wakeup Packet Data

8.21.6 IP Address Valid - IPAV (0x5838; R/W)

The IP Address Valid indicates whether the IP addresses in the IP Address Table are valid.

Field	Bit(s)	Initial Value	Description
V40	0	0b	IPv4 Address 0 Valid.
V41	1	0b	IPv4 Address 1 Valid.
V42	2	0b	IPv4 Address 2 Valid.
V43	3	0b	IPv4 Address 3 Valid.
Reserved	15:4	0x0	Reserved.
V60	16	0b	IPv6 Address 0 Valid.
Reserved	31:17	0b	Reserved.

8.21.7 IPv4 Address Table - IP4AT (0x5840 + 8*n [n=0...3]; R/W)

The IPv4 Address Table is used to store the four IPv4 addresses for the ARP/IPv4 Request packet and Directed IP packet wakeup.

Note: This table is not cleared by any reset.

Field	Bit(s)	Initial Value	Description
IP Address	31:0	X	IPv4 Address n Note: These registers are written in Big Endian order (LS byte is first on the wire and is the MS byte of the IPV4 Address).

Field	Dword #	Address	Bit(s)	Initial Value	Description
IPV4ADDR0	0	0x5840	31:0	X	IPv4 Address 0
IPV4ADDR1	2	0x5848	31:0	X	IPv4 Address 1
IPV4ADDR2	4	0x5850	31:0	X	IPv4 Address 2
IPV4ADDR3	6	0x5858	31:0	X	IPv4 Address 3

8.21.8 IPv6 Address Table - IP6AT (0x5880 + 4*n [n=0...3]; R/W)

The IPv6 Address Table is used to store the IPv6 addresses for neighbor Discovery packet filtering and Directed IP packet wakeup.

Note: This table is not cleared by any reset.

Field	Bit(s)	Initial Value	Description
IP Address	31:0	X	IPv6 Address bytes 4*n+1:4*n +4 Note: These registers appear in Big Endian order (LS byte, LS address is first on the wire and is the MS byte of the IPV6 Address).

Field	Dword #	Address	Bit(s)	Initial Value	Description
IPV6ADDR0	0	0x5880	31:0	X	IPv6 Address 0, bytes 1-4
	1	0x5884	31:0	X	IPv6 Address 0, bytes 5-8
	2	0x5888	31:0	X	IPv6 Address 0, bytes 9-12
	3	0x588C	31:0	X	IPv6 Address 0, bytes 16-13

8.21.9 Flexible Host Filter Table registers - FHFT (0x9000 - 0x93FC; RW)

Each of the 4 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 DW entries, where each 2 DWs are accompanied by an 8-bit mask, one bit per filter byte. When a bit in the 8-bit mask field is set the corresponding Byte in the filter is compared.

The 8 lsb bits of the last DW of each filter contains a length field defining the number of bytes from the beginning of the packet compared by this filter, the length field should be 8 bytes aligned value. If actual packet length is less than (length - 8) (length is the value specified by the length field), the filter fails. Otherwise, it depends on the result of actual byte comparison. The value should not be greater than 128.

Note: The length field must be 8 bytes aligned. For filtering packets shorter than 8 bytes aligned the values should be rounded up to the next 8 bytes aligned value, the hardware implementation compares 8 bytes at a time so it should get extra zero masks (if needed) until the end of the length value.

Bits 31:8 of the last DW of each filter also includes a Queueing field (See Section 8.21.9.1). When the 82580 is in D0 state, the *WUFC.FLEX_HQ* bit is set to 1, *MRQC.Multiple Receive Queues* = 010b or 000b and the packet matches the flex filter, the Queueing field defines the receive queue for the packet, priority of the filter and actions to be initiated.

31 0	31 8	7 0	31 0	31 0
Reserved	Reserved	Mask [7:0]	DW 1	DW 0
Reserved	Reserved	Mask [15:8]	DW 3	DW 2
Reserved	Reserved	Mask [23:16]	DW 5	DW 4
Reserved	Reserved	Mask [31:24]	DW 7	DW 6

....

31 8	7 0	31 8	7 0	31 0	31 0
Reserved	Reserved	Reserved	Mask [127:120]	DW 29	DW 28
Queueing	Length	Reserved	Mask [127:120]	DW 31	DW 30

Accessing the FHFT registers during filter operation can result in a packet being mis-classified if the write operation collides with packet reception. It is therefore advised that the flex filters are disabled prior to changing their setup.

8.21.9.1 Flex Filter Queueing Field

Queueing field resides in bits 31:8 of last DW (DW 63) of flex filter. The queueing field defines the receive queue to forward the packet (*RQUEUE*), the filter priority (*FLEX_PRIO*) and additional filter actions. Operations defined in queueing field are enabled when the 82580 is in D0 state, *MRQC.Multiple Receive Queues* = 010b or 000b, *WUFC.FLEX_HQ* is 1 and relevant *WUFC.FLX[n]* bit is set.

Field	Bit(s)	Initial Value	Description
Length	7:0	X	Length Filter Length in bytes. Should be 8 bytes aligned and not greater then 128 bytes.
RQUEUE	10:8	X	Receive Queue Defines receive queue associated with this Flex filter. When match occurs in D0 state, packet is forwarded to the receive queue.
Reserved	15:11	X	Reserved Write 0, ignore on read.
FLEX_PRIO	18:16	X	Flex filter Priority Defines the priority of the filter assuming two filters with same priority don’t match. If two filters with the same priority match the incoming packet, the first filter (lowest address) is used in order to define the queue destination of this packet.
Reserved	23:19	X	Reserved Write 0, ignore on read.
Immediate Interrupt	24	X	Enables issuing an immediate interrupt when the Flex filter matches incoming packet.
Reserved	31:25	X	Reserved Write 0, ignore on read.

8.21.9.2 Flex Filter 0 - Example

Field	Dword	Address	Bit(s)	Initial Value
Filter 0 DW0	0	0x9000	31:0	X
Filter 0 DW1	1	0x9004	31:0	X
Filter 0 Mask[7:0]	2	0x9008	7:0	X
Reserved	3	0x900C	31:0	X
Filter 0 DW2	4	0x9010	31:0	X
…				
Filter 0 DW30	60	0x90F0	31:0	X
Filter 0 DW31	61	0x90F4	31:0	X
Filter 0 Mask[127:120]	62	0x90F8	7:0	X
Length	63	0x90FC	7:0	X
Filter 0 Queueing	63	0x90FC	31:8	X

8.21.10 Flexible Host Filter Table Extended Registers - FHFT_EXT (0x9A00 - 0x9DFC; RW)

Each of the 4 additional Flexible Host Filters Table extended registers (FHFT_EXT) contains a 128 Byte pattern and a corresponding 128-bit mask array. If enabled, the first 128 Bytes of the received packet are compared against the non-masked bytes in the FHFT_EXT register. The layout and access rules of this table are the same as in FHFT.

8.22 Management Register Descriptions

All management registers are controlled by the remote BMC for both read and write. Host accesses to the management registers are blocked for write. The attributes for the fields in this chapter refer to the BMC access rights.

8.22.1 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.

Field	Bit(s)	Initial Value	Description
VID	11:0	0x0	Contains the VLAN ID that should be compared with the incoming packet if the corresponding bit in *MDEF* is set.
Reserved	31:12	0x0	Reserved Write 0, ignore on read

The *MAVTV* registers are written by the BMC and are not accessible to the host for writing. The registers are used to filter manageability packets as described in the Management chapter.

8.22.2 Management Flex UDP/TCP Ports - MFUTP (0x5030 + 4*n [n=0...7]; RW)

Where each 32-bit register (n=0,…,3) refers to two UDP/TCP port filters (register at address offset n=0 refers to UDP/TCP ports 0 and 1, register at address offset n=1 refers to UDP/TCP ports 2 and 3, etc.).

Field	Bit(s)	Initial Value	Description
MFUTP_even	15:0	0x0	2*n Management Flex UDP/TCP port
MFUTP_odd	31:16	0x0	2*n+1 Management Flex UDP/TCP port

The MFUTP registers are written by the BMC and are not accessible to the host for writing. The registers are used to filter manageability packets. See Section 10.4 .

Reset - The MFUTP registers are cleared on LAN_PWR_GOOD only. The initial values for this register can be loaded from the EEPROM after power-up reset.

Note: The MFUTP_even and MFUTP_odd fields should be written in network order.

8.22.3 Management Ethernet Type Filters- METF (0x5060 + 4*n [n=0...3]; RW)

Field	Bit(s)	Initial Value	Description
METF	15:0	0x0	EtherType value to be compared against the L2 EtherType field in the Rx packet.
Reserved	29:16	0x0	Reserved
Polarity	30	0b	
Reserved	31	0b	Reserved

The METF registers are written by the BMC and are not accessible to the host for writing. The registers are used to filter manageability packets. See Section 10.4 .

Reset - The METF registers are cleared on LAN_PWR_GOOD only. The initial values for this register might be loaded from the EEPROM after power-up reset.

8.22.4 Management Control Register - MANC (0x5820; RW)

The MANC register is written by the BMC and is not accessible to the host for writing.

Field	Bit(s)	Initial Value	Description
Reserved	13:0	0x0	Reserved. Write 0, ignore on read.
FW_RESET	14	0b	FW Reset occurred. Set to 1b on a TCO Firmware Reset. Cleared following a BMC write 0b operation. .
TCO_Isolate (RO)	15	0b	Set to 1 on a TCO Isolate command. When the "TCO_Isolate" bit is set. Host write cycles are completed successfully on the PCIe but silently ignored by internal logic. Note that when FW initiates the TCO Isolate command it also initiates a FW interrupt via the ICR.MNG bit to the host and writes a value of 0x0E to the FWSM.Ext_Err_Ind field. This bit is Read Only and mirrors the value of the Isolate bit in the internal Management registers.
TCO_RESET	16	0b	TCO Reset occurred. Set to 1b on a TCO Reset, to reset LAN port by BMC.
RCV_TCO_EN	17	0b	Enable BMC to network and network to BMC traffic 0 = The BMC can not communicate with the network 1 = The BMC can communicate with the network When cleared the BMC traffic is not forwarded to the network and the network traffic is not forwarded to the BMC even if the decision filters indicates it should.
KEEP_PHY_LINK_UP	18	0b	Block PHY reset and power state changes. When this bit is set the PHY reset and power state changes do not reach to the PHY (PHY is not reset), This bit can not be written to, unless the *Keep_PHY_Link_Up_En* EEPROM bit is set.
Reserved	22:2019	0b	Reserved. Write 0 ignore on read.
EN_XSUM_FILTER	23	0b	Enable checksum filtering to MNG When this bit is set, only packets that pass L3, L4 checksum are sent to the MNG block.
EN_IPv4_FILTER	24	0b	Enable IPv4 address Filters – when set, the last 128 bits of the MIPAF register are used to store 4 IPv4 addresses for IPv4 filtering. When cleared, these bits store a single IPv6 filter.
FIXED_NET_TYPE	25	0b	Fixed net type: If set, only packets matching the net type defined by the NET_TYPE field passes to manageability. Otherwise, both tagged and un-tagged packets can be forwarded to the manageability engine.
NET_TYPE	26	0b	NET TYPE: 0b = pass only un-tagged packets. 1b = pass only VLAN tagged packets. Valid only if FIXED_NET_TYPE is set.
Reserved	31:27	0x0	Reserved Write 0 ignore on read.

8.22.5 Management Only Traffic Register - MNGONLY (0x5864; RW)

The MNGONLY register allows exclusive filtering of certain type of traffic to the BMC. Exclusive filtering enables the BMC to define certain packets that are forwarded to the BMC but not to the host. The packets will not be forwarded to the host even if they pass the host L2 filtering process.

Each manageability decision filter (*MDEF* and *MDEF_EXT*) has a corresponding bit in the *MNGONLY* register. When a manageability decision filter (*MDEF* and *MDEF_EXT*) forwards a packet to manageability, it may also block the packet from being forwarded to the host if the corresponding *MNGONLY* bit is set.

Field	Bit(s)	Initial Value[1]	Description
Exclusive to MNG	7:0	0x0	Exclusive to MNG – when set, indicates that packets forwarded by the manageability filters to manageability are not sent to the host. Bits 0...7 correspond to decision rules defined in registers MDEF[0...7] and MDEF_EXT[0...7].
Reserved	31:8	0x0	Reserved

1. Reset - The MNGONLY register is cleared on LAN_PWR_GOOD and firmware reset. The initial values for this register can be loaded from the EEPROM after power-up reset or firmware reset.

8.22.6 Manageability Decision Filters- MDEF (0x5890 + 4*n [n=0...7]; RW)

Where “n” is the decision filter.

Field	Bit(s)	Initial Value[1]	Description
Unicast AND	1:0	0x0	Unicast - Controls the inclusion of unicast address 0 to 1 in the manageability filter decision (AND section). Bit 0 corresponds to unicast address 0, etc.
Reserved	3:2	0x0	Reserved
Broadcast AND	4	0b	Broadcast - Controls the inclusion of broadcast address filtering in the manageability filter decision (AND section).
VLAN AND	12:5	0x0	VLAN - Controls the inclusion of VLAN tag 0 to 7 respectively in the manageability filter decision (AND section). Bit 5 corresponds to VLAN tag 0, etc.
IPv4 Address	16:13	0x0	IPv4 Address - Controls the inclusion of IPV4 address 0 to 3 respectively in the manageability filter decision (AND section). Bit 13 corresponds to IPV4 address 0, etc. Note: This field is relevant only if MANC.EN_IPv4_FILTER is set.
IPv6 Address	20:17	0b	IPv6 Address - Controls the inclusion of IPV6 address 0 to 3 respectively in the manageability filter decision (AND section). Bit 17 corresponds to IPV6 address 0, etc Note: This field is relevant only if MANC.EN_IPv4_FILTER is cleared.
Unicast OR	22:21	0x0	Unicast - Controls the inclusion of unicast address 0 to 1 respectively in the manageability filter decision (OR section).
Reserved	24:23	0x0	Reserved
Broadcast OR	25	0b	Broadcast - Controls the inclusion of broadcast address filtering in the manageability filter decision (OR section).

Field	Bit(s)	Initial Value[1]	Description
Multicast AND	26	0b	Multicast - Controls the inclusion of Multicast address filtering in the manageability filter decision (AND section). Broadcast packets are not included by this bit.
ARP Request	27	0b	ARP Request - Controls the inclusion of ARP Request filtering in the manageability filter decision (OR section).
ARP Response	28	0b	ARP Response - Controls the inclusion of ARP Response filtering in the manageability filter decision (OR section).
Neighbor Discovery	29	0b	neighbor Discovery - Controls the inclusion of neighbor Discovery filtering in the manageability filter decision (OR section). The neighbor types accepted by this filter are types 0x86, 0x87, 0x88 and 0x89.
Port 0x298	30	0b	Port 0x298 - Controls the inclusion of Port 0x298 filtering in the manageability filter decision (OR section).
Port 0x26F	31	0b	Port 0x26F - Controls the inclusion of Port 0x26F filtering in the manageability filter decision (OR section).

[1] Default values are read from EEPROM.

8.22.7 Manageability Decision Filters- MDEF_EXT (0x5930 + 4*n[n=0...7]; RW)

Field	Bit(s)	Initial Value[1]	Description
L2 EtherType AND	3:0	0x0	L2 EtherType - Controls the inclusion of L2 EtherType filtering in the manageability filter decision (AND section).
Reserved	7:4	0x0	Reserved for additional L2 EtherType AND filters.
L2 EtherType OR	11:8	0x0	L2 EtherType - Controls the inclusion of L2 EtherType filtering in the manageability filter decision (OR section).
Reserved	15:12	0x0	Reserved for additional L2 EtherType OR filters.
Flex port	23:16	0x0	Flex port - Controls the inclusion of Flex port filtering in the manageability filter decision (OR section). Bit 16 corresponds to flex port 0, etc.
Flex TCO	24	0b	Flex TCO - Controls the inclusion of Flex TCO filtering in the manageability filter decision (OR section). Bit 24 corresponds to Flex TCO filter 0.
Reserved	27:25	0x0	Reserved
NC-SI Discard	28	1b	0 = Apply filtering rules to packets with NC-SI EtherType. 1 = Discard packets with NC-SI EtherType. Note: NC-SI EtherType is 0x88F8
Flow Control Discard	29	1b	0 = Apply filtering rules to packets with Flow Control EtherType. 1 = Discard packets with Flow Control EtherType. Note: Flow Control EtherType is 0x8808
Apply_to_network _traffic	30	0b	Do not apply this decision filter to traffic received from the network. Apply this decision filter to traffic received from the network.
Reserved	31	0b	Reserved Write 0, ignore on read.

[1] Default values are read from EEPROM.

8.22.8 Manageability IP Address Filter - MIPAF (0x58B0 + 4*n [n=0...15]; 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 = 0: the last 128 bits of the register store a single IPv6 address (IPV6ADDR3)
- EN_IPv4_FILTER = 1: the last 128 bits of the register store 4 IPv4 addresses (IPV4ADDR[3:0])

MANC.EN_IPv4_FILTER = 0:

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 definitions for 0 setting:

Field	Dword #	Address	Bit(s)	Initial Value	Description
IPV6ADDR0	0	0x58B0	31:0	X*	IPv6 Address 0, bytes 1-4 (LS byte is first on the wire)
	1	0x58B4	31:0	X*	IPv6 Address 0, bytes 5-8
	2	0x58B8	31:0	X*	IPv6 Address 0, bytes 9-12
	3	0x58BC	31:0	X*	IPv6 Address 0, bytes 13-16
IPV6ADDR1	0	0x58C0	31:0	X*	IPv6 Address 1, bytes 1-4 (LS byte is first on the wire)
	1	0x58C4	31:0	X*	IPv6 Address 1, bytes 5-8
	2	0x58C8	31:0	X*	IPv6 Address 1, bytes 9-12
	3	0x58CC	31:0	X*	IPv6 Address 1, bytes 13-16
IPV6ADDR2	0	0x58D0	31:0	X*	IPv6 Address 2, bytes 1-4 (LS byte is first on the wire)
	1	0x58D4	31:0	X*	IPv6 Address 2, bytes 5-8
	2	0x58D8	31:0	X*	IPv6 Address 2, bytes 9-12
	3	0x58DC	31:0	X*	IPv6 Address 2, bytes 13-16

Field	Dword #	Address	Bit(s)	Initial Value	Description
IPV6ADDR3	0	0x58E0	31:0	X*	IPv6 Address 3, bytes 1-4 (LS byte is first on the wire)
	1	0x58E4	31:0	X*	IPv6 Address 3, bytes 5-8
	2	0x58E8	31:0	X*	IPv6 Address 3, bytes 9-12
	3	0x58EC	31:0	X*	IPv6 Address 3, bytes 13-16

MANC.EN_IPv4_FILTER = 1:

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 definitions for 1 Setting:

Field	Dword #	Address	Bit(s)	Initial Value[1]	Description
IPV6ADDR0	0	0x58B0	31:0	X	IPv6 Address 0, bytes 1-4 (LS byte is first on the wire)
	1	0x58B4	31:0	X	IPv6 Address 0, bytes 5-8
	2	0x58B8	31:0	X	IPv6 Address 0, bytes 9-12
	3	0x58BC	31:0	X	IPv6 Address 0, bytes 16-13
IPV6ADDR1	0	0x58C0	31:0	X	IPv6 Address 1, bytes 1-4 (LS byte is first on the wire)
	1	0x58C4	31:0	X	IPv6 Address 1, bytes 5-8
	2	0x58C8	31:0	X	IPv6 Address 1, bytes 9-12
	3	0x58CC	31:0	X	IPv6 Address 1, bytes 16-13
IPV6ADDR2	0	0x58D0	31:0	X	IPv6 Address 2, bytes 1-4 (LS byte is first on the wire)
	1	0x58D4	31:0	X	IPv6 Address 2, bytes 5-8
	2	0x58D8	31:0	X	IPv6 Address 2, bytes 9-12
	3	0x58DC	31:0	X	IPv6 Address 2, bytes 16-13
IPV4ADDR0	0	0x58E0	31:0	X	IPv4 Address 0 (LS byte is first on the wire)

Field	Dword #	Address	Bit(s)	Initial Value[1]	Description
IPV4ADDR1	1	0x58E4	31:0	X	IPv4 Address 1 (LS byte is first on the wire)
IPV4ADDR2	2	0x58E8	31:0	X	IPv4 Address 2 (LS byte is first on the wire)
IPV4ADDR3	3	0x58EC	31:0	X	IPv4 Address 3 (LS 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.

Initial value:

Field	Bit(s)	Initial Value[1]	Description
IP_ADDR 4 bytes	31:0	X	4 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 4 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.

Reset - The registers are cleared on LAN_PWR_GOOD only.

Note: These registers should be written in network order.

8.22.9 Manageability MAC Address Low - MMAL (0x5910 + 8*n [n= 0...1]; RW)

Where “n” is the exact unicast/Multicast address entry, equal to 0…1.

Field	Bit(s)	Initial Value[1]	Description
MMAL	31:0	X	Manageability MAC Address Low. The lower 32 bits of the 48 bit Ethernet address.

[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.

These registers contain the lower bits of the 48 bit Ethernet address. The MMAL registers are written by the BMC and are not accessible to the host for writing. The registers are used to filter manageability packets. See Section 10.4 .

Reset - The MMAL registers are cleared on LAN_PWR_GOOD only. The initial values for this register can be loaded from the EEPROM after power-up reset.

Note: The MMAL.MMAL field should be written in network order.

8.22.10 Manageability MAC Address High - MMAH (0x5914 + 8*n [n=0...1]; RW)

Where “n” is the exact unicast/Multicast address entry, equal to 0…1.

Field	Bit(s)	Initial Value[1]	Description
MMAH	15:0	X	**Manageability MAC Address High**. The upper 16 bits of the 48 bit Ethernet address.
Reserved	31:16	0x0	**Reserved**. Reads as 0. Ignored on write.

[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.

These registers contain the upper bits of the 48 bit Ethernet address. The complete address is {MMAH, MMAL}. The MMAH registers are written by the BMC and are not accessible to the host for writing. The registers are used to filter manageability packets. See Section 10.4 .

Reset - The MMAL registers are cleared on LAN_PWR_GOOD only. The initial values for this register can be loaded from the EEPROM after power-up reset or firmware reset.

Note: The MMAH.MMAH field should be written in network order.

8.22.11 Flexible TCO Filter Table registers - FTFT (0x9400-0x94FC; RW)

The Flexible TCO Filter 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.

The 128 byte filter is composed of 32 DW entries, where each 2 DWs are accompanied by an 8-bit mask, one bit per filter byte. The bytes in each 2 DWs are written in network order i.e. 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 1 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 8 bytes 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 quad word. Any mask bit that is located after the length should be set to 0 indicating no comparison should be done.

In case the actual length which is defined by the length field register and the mask bits is not 8 bytes aligned there may be a case that a packet which is shorter then the actual required length passes the flexible filter. This may occur due to comparison of up to 7 bytes that come after the packet, but are not a real part of the packet.

The last DW of the 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 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.

31 0	31 8	7 0	31 0	31 0
Reserved	Reserved	Mask [7:0]	DW 1	DW 0
Reserved	Reserved	Mask [15:8]	DW 3	DW 2
Reserved	Reserved	Mask [23:16]	DW 5	DW 4
Reserved	Reserved	Mask [31:24]	DW 7	DW 6

....

31 8	7 0	31 8	7 0	31 0	31 0
Reserved	Reserved	Reserved	Mask [127:120]	DW 29	DW 28
Reserved	Length	Reserved	Mask [127:120]	DW 31	DW 30

Field definitions for Filter Table Registers:

Field	Dword	Address	Bit(s)	Initial Value
Filter 0 DW0	0	0x9400	31:0	X
Filter 0 DW1	1	0x9404	31:0	X
Filter 0 Mask[7:0]	2	0x9408	7:0	X
Reserved	3	0x940C		X
Filter 0 DW2	4	0x9410	31:0	X
...				
Filter 0 DW30	60	0x94F0	31:0	X
Filter 0 DW31	61	0x94F4	31:0	X
Filter 0 Mask[127:120]	62	0x94F8	7:0	X
Length	63	0x94FC	6:0	X

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 are not accessible to the host for writing. The registers are used to filter manageability packets as described in Section 10.4.3.5.

Reset - The FTFT registers are cleared on LAN_PWR_GOOD only.

8.23 Memory Error Registers Description

The 82580 main internal memories are protected by error correcting code (ECC) or parity bits.

8.23.1 Parity and ECC Error Indication- PEIND (0x1084; RC)

Field	Bit(s)	Initial Value	Description
lanport_parity_fatal_ind (RC)	0	0b	Fatal Error detected in LAN port Memory. Bit is latched high and cleared on read.
mng_parity_fatal_ind (RC)	1	0b	Fatal Error detected in Management Memory. Bit is latched high and cleared on read.
pcie_parity_fatal_ind (RC)	2	0b	Fatal Error detected in PCIe Memory. Bit is latched high and cleared on read.
dma_parity_fatal_ind (RC)	3	0b	Fatal Error detected in DMA Memory Bit is latched high and cleared on read.
Reserved	31:4	0x0	Reserved Write 0 ignore on read.

8.23.2 Parity and ECC Indication Mask – PEINDM (0x1088; RW)

Field	Bit(s)	Initial Value	Description
lanport_parity_fatal_ind	0	1b	When set and *PEIND.lanport_parity_fatal_ind* is set, enable interrupt generation by setting the *ICR.FER bit.*
mng_parity_fatal_ind	1	1b	When set and *PEIND.mng_parity_fatal_ind* is set, enable interrupt generation by setting the *ICR.FER bit.*
pcie_parity_fatal_ind	2	1b	When set and *PEIND.pcie_parity_fatal_ind* is set, enable interrupt generation by setting the *ICR.FER bit.*
dma_parity_fatal_ind	3	1b	When set and *PEIND.dma_parity_fatal_ind* is set, enable interrupt generation by setting the *ICR.FER bit.*
Reserved	31:4	0x0	Reserved Write 0 ignore on read.

8.23.3 DMA Transmit Descriptor Parity Control - DTPARC (0x3500; RW)

Field	Bit(s)	Initial Value	Description
ram_dtx_lso_par_en	0	0b	ram_dtx_lso parity check enable.
Reserved	1	0b	Reserved. Write 0, ignore on read.
ram_dtx_cntxt_par_en	2	0b	ram_dtx_cntxt parity check enable.
Reserved	3	0b	Reserved. Write 0, ignore on read.
ram_dtx_hdr_par_en	4	0b	ram_dtx_hdr parity check enable.
Reserved	5	0b	Reserved. Write 0, ignore on read.

Field	Bit(s)	Initial Value	Description
ram_dtx_temp_par_en	6	0b	ram_dtx_temp parity check enable.
Reserved	7	0b	Reserved. Write 0, ignore on read.
ram_dtx_cmd_par_en	8	0b	ram_dtx_cmd parity check enable.
Reserved	9	0b	Reserved. Write 0, ignore on read.
ram_dtx_dhrf_par_en	10	0b	ram_dtx_dhrf parity check enable.
Reserved	11	0b	Reserved. Write 0, ignore on read.
ram_dtx_icache_par_en	12	0b	ram_dtx_icache parity check enable.
Reserved	31:13	0x0	Reserved. Write 0, ignore on read.

8.23.4 DMA Transmit Descriptor Parity Status- DTPARS (0x3510; RW1C)

8.23.5 DMA Receive Descriptor Parity Control - DRPARC (0x3504; RW)

8.23.6 DMA Receive Descriptor Parity Status - DRPARS (0x3514; RW1C)

8.23.7 Dhost Parity Control - DDPARC (0x3508; RW)

8.23.8 Rx Packet Buffer ECC Status - RPBECCSTS (0x245C; RW)

Field	Bit(s)	Initial Value	Description
Corr_err_cnt (RC)	7:0	0x0	Correctable Error Count This counter is incremented every time a correctable error is detected; the counter stops after reaching 0xFF. This field is cleared by read.
Reserved	15:8	0x0	Reserved Write 0 ignore on read
ECC Enable (RW)	16	1b	ECC Enable for Packet Buffer.
Reserved	25 :17	0x0	Reserved
Pb_cor_err_sta (RC)	26	0b	Status of PB Correctable Error. This bit is cleared by a read.
Reserved	31:27	0x0	Reserved

8.23.9 Tx Packet Buffer ECC Status - TPBECCSTS (0x345C; RW)

Field	Bit(s)	Initial Value	Description
Corr_err_cnt (RC)	7:0	0x0	Correctable Error Count This counter is incremented every time a correctable error is detected; the counter stops after reaching 0xFF. This field is cleared by read.
Reserved	15:8	0x0	Reserved Write 0 ignore on read
ECC Enable	16	1b	ECC Enable for Packet Buffer.
Reserved	25:17	0x0	Reserved
Pb_cor_err_sta (RC)	26	0b	Status of PB Correctable Error. This bit is cleared by a read.
Reserved	31:27	0x0	Reserved

8.23.10 PCIe ECC Status Register - PCIEECCSTS (0x5BAC; R/W1C)

Field	Bit(s)	Initial Value	Description
ECC ERR TX WR CMD	0	0b	TX Write Request Command ECC Error
ECC ERR TX RD CMD	1	0b	TX Read Request Command ECC Error
Reserved	31:2	0x0	Reserved

8.23.11 LAN Port Parity Error Control Register - LANPERRCTL (0x5F54; RW)

Field	Bit(s)	Initial Value	Description
mrx_flx_en	8:0	0x0	Enable mrx_flx parity error indication When set to 0x1FF enables Flex filter memory parity error detection and indication. **Note:** Software should program the *FHFT*, *FHFT_EXT*, *FTFT* memories before enabling parity check, to avoid incorrect parity error detection.
retx_buf_en	9	0b	Enable retx_buf parity error indication When set to 1b enables RETX buffer (retransmit buffer) parity error detection and indication.
stat_regs_en	10	0b	Enable stat_regs parity error indication When set to 1b enables statistics memory parity error detection and indication.
f_uta_en	11	0b	Enable f_uta parity error indication When set to 1b enable unicast filter table parity error detection and indication. **Note:** Software should program the *Unicast Table Array* (*UTA*) before enabling parity check, to avoid incorrect parity error detection.
f_rss_en	12	0b	Enable f_rss parity error indication When set to 1b enables RSS memory parity error detection and indication. **Note:** Software should program the Redirection Table (*RETA*) before enabling parity check, to avoid incorrect parity error detection.

Field	Bit(s)	Initial Value	Description
f_mulc _en	13	0b	Enable f_mulc parity error indication When set to 1b enables multicast filter table parity error detection and indication. ***Note:*** Software should program the Multicast Table Array (*MTA*) before enabling parity check, to avoid incorrect parity error detection.
f_vlan_en	14	0b	Enable f_vlan parity error indication When set to 1b enables VLAN filter table parity error detection and indication. ***Note:*** Software should program the VLAN Filter Table Array (*VFTA*) before enabling parity check, to avoid incorrect parity error detection.
Reserved	31:15	0b	Reserved. Write 0, ignore on read.

8.23.12 LAN Port Parity Error Status Register - LANPERRSTS (0x5F58; RO)

Note: Parity indication bits cleared by issuing reset.

Field	Bit(s)	Initial Value	Description
mrx_flx	8:0	0x0	mrx_flx parity error indication When set to 1b indicates Flex filter memory parity error detection.
retx_buf	9	0b	retx_buf parity error indication when set to 1b indicates RETX buffer (retransmit buffer) parity error detection.
stat_regs	10	0b	stat_regs parity error indication When set to 1b indicates statistics memory parity error detection.
f_uta	11	0b	f_uta parity error indication When set to 1b indicates unicast filter table parity error detection.
f_rss	12	0b	f_rss parity error indication When set to 1b indicates RSS memory parity error detection.
f_mulc	13	0b	f_mulc parity error indication When set to 1b indicates multicast filter table parity error detection.
f_vlan	14	0b	f_vlan parity error indication When set to 1b indicates VLAN filter table parity error detection.
Reserved	31:15	0x0	Reserved

8.24 PHY Software Interface

8.24.1 PHY Power Management - PHPM (0x0E14, RW)

The *PHPM* register controls Internal PHY Power management operation.

Field	Bit(s)	Description	Mode	Default
Reserved	31:10	Always read as 0b. Write to 0b for normal operation.	R/W	0x0
Disable 100 in non-D0a[1]	9	Disables 100 Mb/s and 1000 Mb/s operation in non-D0a states.	R/W	1b
rst_compl	8	Indicates PHY internal reset cleared.	RO, LH	0b

Field	Bit(s)	Description	Mode	Default
SPD_B2B_EN	7	SPD back-to-back enable. Note: Bit 2 in PHMIC (21d) register should be 0b to allow the *PHPM.SPD_B2B_EN* bit to disable Smart Power Down Back to Back operation.	R/W	1b
Disable 1000[2]	6	When set, disables 1000 Mb/s in all power modes. Note: Bit 0 in PHMIC (21d) register should be 0b to allow the *PHPM.Disable 1000* bit to enable 1G operation.	R/W	0b
Go Link disconnect	5	Setting this bit will cause the PHY to enter link disconnect mode immediately.	R/W	0b
Link Energy Detect	4	This bit is set when the PHY detects energy on the link. Note that this bit is valid only if *PHPM.Go Link disconnect* is set to 1b.	RO, LH	0b
Disable 1000 in non-D0a[3]	3	Disables 1000 Mb/s operation in non-D0a states. Note: Bit 0 in PHMIC (21d) register should be 0b to allow the *PHPM.Disable 1000 in non-D0a* bit to enable 1G operation in non-D0a PM state.	R/W	1b
LPLU[4]	2	Low Power on Link Up When set, enables the decrease in link speed while in non-D0a states when the power policy and power management state specify it. Note: Bit 8 in *PHCTRL1* (23d) register should be 0b for the *PHPM.LPLU* bit to disable Low Power Link Up operation.	R/W	1b
D0LPLU	1	D0 Low Power Link Up When set, configures the PHY to negotiate for a low speed link in all states. Note: Bit 8 in *PHCTRL1* (23d) register should be 0b for the *PHPM.D0LPLU* bit to disable Low Power Link Up operation.	R/W	0b
SPD_EN[5]	0	Smart Power Down When set, enables PHY Smart Power Down mode. Note: Bit 3 in PHMIC (21d) register should be 0b to allow the *PHPM.SPD_EN* bit to disable Smart Power Down operation.	R/W	1b
Reserved	0	Reserved		

[1] Bit Loaded from *Disable 100 in non-D0a* bit in *Software Defined Pins Control* EEPROM word on reset

[2] Bit Loaded from *Giga Disable* bit in *Software Defined Pins Control* EEPROM word on reset

[3] Bit Loaded from *Disable 1000 in non-D0a* bit in *Software Defined Pins Control* EEPROM word on reset

[4] Bit Loaded from *LPLU* bit in *Initialization Control 4* EEPROM word on reset

[5] Bit Loaded from *SPD Enable* bit in *Initialization Control 4* EEPROM word on reset

8.24.2 Internal PHY Software Interface (PHYREG)

1. Base Registers (0 through 10 and 15) are defined in accordance with the "Reconciliation Sub layer and Media Independent Interface" and "Physical Layer Link Signaling for 10/100/ 1000 Mb/s Auto-Negotiation" sections of the IEEE 802.3.
2. Additional registers (PHYREG.16 through 31) are defined in accordance with the IEEE 802.3 specification for adding unique chip functions.
3. Registers in Table 8-23 are accessed using the internal MDIO interface via the MDIC register (See Section 3.5.2.2).

Table 8-23. Table of PHYREG Registers

Offset	Abbreviation	Name	RW	Link to Page
00d	PCTRL	PHY Control Register	R/W	page 529
01d	PSTATUS	PHY Status Register	RO	page 530
02d	PHY ID 1	PHY Identifier Register 1 (LSB)	RO	page 531
03d	PHY ID 2	PHY Identifier Register 2 (MSB)	RO	page 532
04d	ANA	Auto–Negotiation Advertisement Register	R/W	page 532
05d	ANLPA	Auto–Negotiation link partner Ability Register	RO	page 533
06d	ANE	Auto–Negotiation Expansion Register	RO	page 533
07d	NPT	Auto–Negotiation Next Page Transmit Register	R/W	page 534
08d	LPN	Auto–Negotiation Next Page Link Partner Register	RO	page 534
09d	GCON	1000BASE–T/100BASE–T2 Control Register	R/W	page 535
10d	GSTATUS	1000BASE–T/100BASE–T2 Status Register	RO	page 535
11d - 14d		Reserved		
15d	ESTATUS	Extended Status Register	RO	page 536
16d - 17d		Reserved		
18d	PHCTRL2	PHY control register 2	R/W	page 536
19d	PHLBKC	Loopback control register.	R/W	page 538
20d	PHRERRC	RX error counter register	RO, SC	page 539
21d	PHMIC	Management interface (MI) control register.	R/W	page 539
22d	PHCNFG	PHY configuration register.	R/W	page 540
23d	PHCTRL1	PHY control register 1.	R/W	page 541
24d	PHINTM	Interrupt mask register.	R/W	page 542
25d	PHINTR	Interrupt status register.	RO	page 542
26d	PHSTAT	PHY status register.	RO	page 543
29d - 27d		Reserved.		
30d	PHDIAG	Diagnostics control register (Linking Disabled).	R/W	page 545
31d	PHDSTAT	Diagnostics status register.	RO	page 546

8.24.2.1 PHY Control Register - PCTRL (00d; R/W)

Field	Bit(s)	Description	Mode	Default
Reserved	5:0	Reserved Ignore on read, Write 0x0.	RW	Always 000000b
Speed Selection 1000 Mb/s (MSB)[5]	6	Speed Selection is determined by bits 6 (MSB) and 13 (LSB) as follows. 11b = Reserved 10b = 1000 Mb/s 01b = 100 Mb/s 00b = 10 Mb/s Note: If auto-negotiation is enabled, this bit is ignored.	R/W	00b
Collision Test[1]	7	1b = Enable COL signal test. 0b = Disable COL signal test.	R/W	0b

Field	Bit(s)	Description	Mode	Default
Duplex Mode[2]	8	1b = Full Duplex. 0b = Half Duplex. Note: If auto-negotiation is enabled, this bit is ignored.	R/W	1b
Restart Auto-Negotiation	9	1b = Restart Auto-Negotiation Process. 0b = Normal operation. Auto-Negotiation automatically restarts after hardware or software reset regardless of whether or not the restart bit is set.	R/W, SC	0b
Isolate[3]	10	1b = Isolates PHY from internal MII/GMII interface. 0b = Normal operation.	R/W	0b
Power Down	11	1b = Power down. 0b = Normal operation.	R/W	0b
Auto-Negotiation Enable[4]	12	1b = Enable Auto-Negotiation Process. 0b = Disable Auto-Negotiation Process. This bit must be enabled for 1000BASE-T operation.	R/W	1b
Speed Selection (LSB)[5]	13	See Speed Selection (MSB), bit 6. Note: If auto-negotiation is enabled, this bit is ignored.	R/W	1b
Loopback[6]	14	1b = Enable loopback. 0b = Disable loopback.	R/W	0b
Reset[7]	15	1b = PHY reset. 0b = Normal operation. Note: When using PHY Reset, the PHY default configuration is not loaded from the EEPROM. The preferred way to reset the 82580 PHY is using the *CTRL.PHY_RST* field.	WO, SC	0b

1. Enables IEEE 802.3 Clause 22.2.4.1.9 collision test.
2. This bit may be used to configure the link manually. Setting this bit has no effect unless address 0d, bit 12 is clear.
3. Setting this bit isolates the PHY from the internal MII or GMII interfaces.
4. When this bit is cleared, the link configuration is determined manually.
5. The speed selection address 0d, bits 13 and 6 may be used to configure the link manually. Setting these bits has no effect unless address 0d bit 12 is clear.
6. This is the master enable for digital and analog loopback as defined by the standard. The exact type of loopback is determined by the loopback control register (address 19d).
7. The reset bit is automatically cleared upon completion of the reset sequence. This bit is set to 1 during reset.

8.24.2.2 PHY Status Register - PSTATUS (01d; RO)

Field	Bit(s)	Description	Mode	Default
Extended Capability[1]	0	1b = Extended register capabilities.	RO	1b
Jabber Detect	1	1b = Jabber condition detected. 0b = Jabber condition not detected.	RO LH	0b
Link Status[2]	2	1b = Link is up. 0b = Link is down.	RO, LL	0b
Auto-Negotiation Ability	3	1b = PHY able to perform Auto-Negotiation. 0b = PHY is not able to perform Auto-Negotiation.	RO	1b
Remote Fault[3]	4	1b = Remote fault condition detected. 0b = Remote fault condition not detected.	RO LH	0b

Field	Bit(s)	Description	Mode	Default
Auto-Negotiation [4] Complete	5	1b = Auto-Negotiation process complete. 0b = Auto-Negotiation process not complete.	RO	0b
MF Preamble Suppression	6	1b = PHY accepts management frames with preamble suppressed. 0b = PHY does not accept management frames with preamble suppressed.	RO	1b
Reserved	7	Reserved. Ignore on read, write 0.	RO	0b
Extended Status	8	1b = Extended status information in the Extended PHY Status Register (15d). 0b = No extended status information in the Extended PHY Status Register (15d).	RO	1b
100BASE-T2 Half Duplex	9	1b = PHY able to perform half duplex 100BASE-T2 (not supported). 0b = PHY not able to perform half duplex 100BASE-T2.	RO	0b
100BASE-T2 Full Duplex	10	1b = PHY able to perform full duplex 100BASE-T2 (not supported). 0b = PHY not able to perform full duplex 100BASE-T2.	RO	0b
10 Mb/s Half Duplex	11	1b = PHY able to perform half duplex 10BASE-T. 0b = PHY not able to perform half duplex 10BASE-T.	RO	1b
10 Mb/s Full Duplex	12	1b = PHY able to perform full duplex 10BASE-T. 0b = PHY not able to perform full duplex 10BASE-T.	RO	1b
100BASE-X Half Duplex	13	1b = PHY able to perform half duplex 100BASE-X. 0b = PHY not able to perform half duplex 100BASE-X.	RO	1b
100BASE-X Full Duplex	14	1b = PHY able to perform full duplex 100BASE-X. 0b = PHY not able to perform full duplex 100BASE-X.	RO	1b
100BASE-T4	15	1b = PHY able to perform 100BASE-T4. 0b = PHY not able to perform 100BASE-T4.	RO	0b

1. Indicates that the PHY provides an extended set of capabilities that may be accessed through the extended register set. For a PHY that incorporates a GMII/RGMII, the extended register set consists of all management registers except registers 0, 1, and 15.
2. This bit indicates that a valid link has been established. Once cleared due to link failure, this bit remains cleared until register 1d is read via the management interface.
3. This bit indicates that a remote fault has been detected. Once set, it remains set until it is cleared by reading register 1d via the management interface or by PHY reset.
4. Upon completion of autonegotiation, this bit becomes set.

8.24.2.3 PHY Identifier Register 1 (LSB) - PHY ID 1 (02d; RO)

Field	Bit(s)	Description	Mode	Default
PHY ID Number[1]	15:0	The PHY identifier composed of bits 3 through 18 of the Organizationally Unique Identifier (OUI)	RO	0x0154

1. PHY ID Number based on Intel assigned OUI number of 00-AA-00 following bit reversal.

8.24.2.4 PHY Identifier Register 2 (MSB) - PHY ID 2 (03d; RO)

Field	Bit(s)	Description	Mode	Default
Manufacturer's Revision Number	3:0	4 bits containing the manufacturer's revision number.	RO	0x1
Manufacturer's Model Number	9:4	6 bits containing the manufacturer's part number.	RO	0x3A
PHY ID Number[1]	15:10	The PHY identifier composed of bits 19 through 24 of the OUI	RO	0x0

[1] PHY ID Number based on Intel assigned OUI number of 00-AA-00 following bit reversal.

8.24.2.5 Auto–Negotiation Advertisement Register - ANA (04d; R/W)

Field	Bit(s)	Description	Mode	Default
Selector Field	4:0	00001b = 802.3 Other combinations are reserved. Unspecified or reserved combinations should not be transmitted. Note: Setting this field to a value other than 00001b can cause auto negotiation to fail.	R/W	00001b
10Base-T Half Duplex	5	1b = DTE is 10BASE-T Half Duplex capable. 0b = DTE is not 10BASE-T Half Duplex capable.	R/W	1b
10Base-T Full Duplex	6	1b = DTE is 10BASE-T Full duplex capable. 0b = DTE is not 10BASE-T Full duplex capable.	R/W	1b
100Base-TX Half Duplex	7	1b = DTE is 100BASE-TX Half Duplex capable. 0b = DTE is not 100BASE-TX Half Duplex capable.	R/W	1b
100BASE-TX Full Duplex	8	1b = DTE is 100BASE-TX Full duplex capable. 0b = DTE is not 100BASE-TX Full duplex capable.	R/W	1b
100BASE-T4	9	1b = Capable of 100BASE-T4 (not supported). 0b = Not capable of 100BASE-T4.	R/W	0b
PAUSE	10	Advertise to Partner that Pause operation (as defined in 802.3x) is desired.	R/W	1b
ASM_DIR	11	Advertise Asymmetric Pause direction bit. This bit is used in conjunction with PAUSE.	R/W	1b
Reserved	12	Reserved. Ignore on read, write 0.	R/W	0b
Remote Fault	13	1b = Set Remote Fault bit. 0b = Do not set Remote Fault bit.	R/W	0b
Reserved	14	Reserved. Ignore on read, write 0.	R/W	0b
Next Page	15	1 = Advertises next page ability supported. 0 = Advertises next page ability not supported.	R/W	0b

8.24.2.6 Auto–Negotiation Link Partner Ability Register - ANLPA (05d; RO)

Field	Bit(s)	Description	Mode	Default
Selector Fields[4:0]	4:0	<00001> = IEEE 802.3 Other combinations are reserved. Unspecified or reserved combinations must not be transmitted. If field does not match PHY Register 04d, bits 4:0, the AN process does not complete and no HCD is selected.	RO	N/A
10BASE-T Half Duplex	5	1b = Link Partner is 10BASE-T Half Duplex capable. 0b = Link Partner is not 10BASE-T Half Duplex capable.	RO	N/A
10BASE-T Full Duplex	6	1b = Link Partner is 10BASE-T Full duplex capable. 0b = Link Partner is not 10BASE-T Full duplex capable.	RO	N/A
100BASE-TX Half Duplex	7	1b = Link Partner is 100BASE-TX Half Duplex capable. 0b = Link Partner is not 100BASE-TX Half Duplex capable.	RO	N/A
100BASE-TX Full Duplex	8	1b = Link Partner is 100BASE-TX Full duplex capable. 0b = Link Partner is not 100BASE-TX Full duplex capable.	RO	N/A
100BASE-T4	9	1b = Link Partner is 100BASE-T4 capable. 0b = Link Partner is not 100BASE-T4 capable.	RO	N/A
LP Pause	10	Link Partner uses Pause Operation as defined in 802.3x.	RO	N/A
LP ASM_DIR	11	Asymmetric Pause Direction Bit 1b = Link Partner is capable of asymmetric pause. 0b = Link Partner is not capable of asymmetric pause.	RO	N/A
Reserved	12	Reserved. Ignore on read, write 0.		
Remote Fault	13	1b = Remote fault. 0b = No remote fault.	RO	N/A
Acknowledge	14	1b = Link Partner has received Link Code Word from the PHY. 0b = Link Partner has not received Link Code Word from the PHY.	RO	N/A
Next Page	15	1b = Link Partner has ability to send multiple pages. 0b = Link Partner has no ability to send multiple pages.	RO	N/A

8.24.2.7 Auto–Negotiation Expansion Register - ANE (06d; RO)

Field	Bit(s)	Description	Mode	Default
Link Partner Auto-Negotiation Able	0	1b = Link Partner is Auto-Negotiation able. 0b = Link Partner is not Auto-Negotiation able.	RO	0b
Page Received	1	Indicates that a new page has been received and the received code word has been loaded into PHY register 05d (base pages) or PHY register 08d (next pages) as specified in clause 28 of 802.3. 1 = New page has been received from link partner. 0 = New page has not been received.	RO/LH	0b
Next Page Able	2	1b = Local device is next page able. 0b = Local device is not next page able.	RO	1b

Field	Bit(s)	Description	Mode	Default
Link Partner Next Page Able	3	1b = Link Partner is next page able. 0b = Link Partner is not next page able.	RO	0b
Parallel Detection Fault	4	1b = Parallel detection fault has occurred. 0b = Parallel detection fault has not occurred.	RO/LH	0b
Reserved	15:5	Reserved. Ignore on read, write 0.		

8.24.2.8 Auto–Negotiation Next Page Transmit Register - NPT (07d; R/W)

Field	Bit(s)	Description	Mode	Default
Message/Unformatted Field	10:0	11-bit Next page message code or unformatted data.	R/W	0x1
Toggle	11	1b = Previous value of the transmitted Link Code Word = 0b. 0b = Previous value of the transmitted Link Code Word = 1b.	RO	0b
Acknowledge 2	12	1b = Complies with message. 0b = Cannot comply with message.	R/W	0b
Message Page	13	1b = Message page. 0b = Unformatted page.	R/W	1b
Reserved	14	Reserved. Ignore on read, write 0.		
Next Page	15	1b = Additional next pages follow. 0b = Last page.	R/W	0b

8.24.2.9 Auto–Negotiation Next Page Link Partner Register - LPN (08d; RO)

Bit(s)	Field	Description	Mode	Default
10:0	Message/Unformatted Field	11-bit Next page message code or unformatted data.	RO	0x0
11	Toggle	1b = Previous value of the transmitted Link Code Word = 0b. 0b = Previous value of the transmitted Link Code Word = 1b.	RO	0b
12	Acknowledge 2	1b = Link Partner complies with the message. 0b = Link Partner cannot comply with the message.	RO	0b
13	Message Page	1b = Page sent by the Link Partner is a Message Page. 0b = Page sent by the Link Partner is an Unformatted Page.	RO	0b
14	Acknowledge	1b = Link Partner has received Link Code Word from the PHY. 0b = Link Partner has not received Link Code Word from the PHY.	RO	0b
15	Next Page	1b = Link Partner has additional next pages to send. 0b = Link Partner has no additional next pages to send.	RO	0b

8.24.2.10 1000BASE–T/100BASE–T2 Control Register - GCON (09d; R/W)

Bit(s)	Field	Description	Mode	Default
7:0	Reserved	Reserved. Ignore on read, write 0.		
8	1000BASE-T Half Duplex	1b = DTE is 1000BASE-T Half Duplex capable. 0b = DTE is not 1000BASE-T Half Duplex capable. This bit is used by Smart Negotiation.	R/W	0b
9	1000BASE-T Full Duplex	1b = DTE is 1000BASE-T full duplex capable. 0b = DTE is not 1000BASE-T full duplex capable. This bit is used by Smart Negotiation.	R/W	1b
10	Port Type	1b = Prefer multi-port device (Master). 0b = Prefer single port device (Slave). This bit is only used when PHY register 9, bit 12 is set to 0b.	R/W	1b
11	Master/Slave Config Value[1]	1b = Configure PHY as MASTER during MASTER-SLAVE negotiation (only when PHY register 9, bit 12 is set to 1b. 0b = Configure PHY as SLAVE during MASTER-SLAVE negotiation (only when PHY register 9, bit 12 is set to 1b.	R/W	0b
12	Master/Slave Config Enable	1b = Manual Master/Slave configuration. 0b = Automatic Master/Slave configuration.	R/W	0b
15:13	Test mode	000b = Normal Mode. 001b = Pulse and Droop Template. 010b = Jitter Template. 011b = Jitter Template. 100b = Distortion Packet. 101b, 110b, 111b = Reserved.	R/W	000b

[1] Setting this bit has no effect unless address 9d, bit 12 is set.

8.24.2.11 1000BASE–T/100BASE–TX Status Register - GSTATUS (10d; RO)

Bit(s)	Field	Description	Mode	Default
7:0	Idle Error Count[1]	MSB of idle error count.	RO, SC	0x0
9:8	Reserved	Reserved. Ignore on read, write 0.		
10	LP 1000T HD	1b = Link Partner is capable of 1000BASE-T half duplex. 0b = Link Partner is not capable of 1000BASE-T half duplex. Value in bit 10 are not valid until the ANE Register Page Received bit equals 1b.	RO	0b
11	LP 1000T FD	1b = Link Partner is capable of 1000BASE-T full duplex. 0b = Link Partner is not capable of 1000BASE-T full duplex. Value in bit 11 are not valid until the ANE Register Page Received bit equals 1b.	RO	0b
12	Remote Receiver Status	1b = Remote Receiver OK. 0 b = Remote Receiver Not OK.	RO	0b

Bit(s)	Field	Description	Mode	Default
13	Local Receiver Status	1b = Local Receiver OK. 0b = Local Receiver Not OK.	RO	0b
14	Master/Slave Resolution[2]	1b = Local PHY configuration resolved to Master. 0b = Local PHY configuration resolved to Slave. Value in bit 14 is not valid until the ANE Register Page Received bit equals 1b.	RO	0b
15	Master/Slave Config Fault[3]	1b = Master/Slave configuration fault detected. 0b = No Master/Slave configuration fault detected.	RO, LH, SC	0b

[1] These bits contain a cumulative count of the errors detected when the receiver is receiving idles and both local and remote receiver status are OK. The count is held at 255 in the event of overflow and is reset to zero by reading register 10d via the management interface or by reset.

[2] This bit is not valid when bit 15 is set.

[3] Once set, this bit remains set until cleared by the following actions:

- Read of register 10d via the management interface.
- Reset.
- Completion of autonegotiation.
- Enable of autonegotiation

8.24.2.12 Extended Status Register - ESTATUS (15d; RO)

Field	Bit(s)	Description	Mode	Default
Reserved	11:0	Reserved. Ignore on read, write 0.		
1000BASE-T Half Duplex	12	1b = 1000BASE-T half duplex capable. 0b = not 1000BASE-T half duplex capable.	RO	1b
1000BASE-T Full Duplex	13	1b = 1000BASE-T full duplex capable. 0b = Not 1000BASE-T full duplex capable.	RO	1b
1000BASE-X Half Duplex	14	1b =1000BASE-X half duplex capable. 0b = Not 1000BASE-X half duplex capable.	RO	0b
1000BASE-X Full Duplex	15	1b =1000BASE-X full duplex capable. 0b = Not 1000BASE-X full duplex capable.	RO	0b

8.24.2.13 PHY Control Register 2 - PHCTRL2(18d; R/W)

Field	Bit(s)	Description	Mode	Default
Reserved	0	Reserved. Ignore on read, write 0.		0b
Reserved_1	1	Reserved. Ignore on read, write 1.		1b
Enable Diagnostics[1]	2	1b = Enables diagnostics. 0b = Disables diagnostics.	R/W	0b
Reserved_1	3	Reserved. Ignore on read, write 1.		1b
Reserved	8:4	Reserved. Ignore on read, write 0.		

Field	Bit(s)	Description	Mode	Default
MDI/MDI-X Configuration	9	1b = Manual MDI-X configuration. 0b = Manual MDI configuration.	R/W	0b
Automatic MDI/MDI-X	10	1b = Enables automatic MDI/MDI-X detection. 0b = Disables automatic MDI/MDI-X detection.	R/W	1b
Reserved	12:11	Reserved. Ignore on read, write 0.		
Count Symbol Errors	13[2]	1b = Rx error counter counts symbol errors. 0b = Rx error counter counts CRC errors.	R/W	0b
Count False Carrier Events	14[2]	1b = Rx error counter counts false carrier events. 0b = Rx error counter does not count false carrier events.	R/W	0b
Resolve MDI/MDI-X before Forced Speed	15	1b = Resolves MDI/MDI-X configuration before forcing speed. 0b = Does not resolve MDI/MDI-X configuration before forcing speed.	R/W	1b

1. This bit enables PHY diagnostics, which include IP phone detection and TDR cable diagnostics. It is not recommended to enable this bit in normal operation (when the link is active). This bit does not need to be set for link analysis cable diagnostics.
2. Count symbol errors (18.13) and count false carrier events (18.14) control the type of errors that the Rx error counter (20.15:0) counts (settings are shown below). The default is to count CRC errors (See Table 8-24).

Table 8-24. RX Error Counter Programming

Count False Carrier Events	Count Symbol Errors	Rx Error Counter
1	1	Counts symbol errors and false carrier events.
1	0	Counts CRC errors and false carrier events
0	1	Counts symbol errors.
0	0	Counts CRC errors.

Bit 9, PHY Control Register 2, manually sets the MDI/MDI-X configuration if automatic MDIX is disabled, as indicated below.

Table 8-25. MDI/MDI-X Configuration

Automatic MDI/MDI-X	MDI/MDI-X Configuration	MDI/MDI-X Mode
1	X	Automatic MDI/MDI-X detection.
0	0	MDI configuration (NIC/DTE).
0	1	MDI-X configuration (switch).

The mapping of the transmitter and receiver to pins, for MDI and MDI-X configurations for 10Base-T, 100Base-TX, and 1000Base-T is shown in Table 8-26. Note that even in manual MDI/MDI-X configuration, the PHY automatically detects and corrects for C and D pair swaps.

Table 8-26. MDI/MDI-X Pin Mapping

Pin	MDI Pin Mapping			MDI-X Pin Mapping		
	10Base-T	100Base-TX	1000Base-T	10Base-T	100Base-TX	1000Base-T
MDI_0_P/N	Transmit +/–	Transmit +/–	Transmit A+/– Receive B+/–	Receive +/–	Receive +/–	Transmit B+/– Receive A+/–
MDI_1_P/N	Receive +/–	Receive +/–	Transmit B+/– Receive A+/–	Transmit +/–	Transmit +/–	Transmit A+/– Receive B+/–
MDI_2_P/N			Transmit C+/– Receive D+/–			Transmit D+/– Receive C+/–
MDI_3_P/N			Transmit D+/– Receive C+/–			Transmit C+/– Receive D+/–

8.24.2.14 Loopback Control Register - PHLBKC (19d; R/W)

Field	Bit(s)	Description	Mode	HW Rst
Force Link Status[1]	0	1b = Forces link status okay in MII loopback. 0b = Forces link status not okay in MII loopback.	R/W	1b
Reserved	5:1	Reserved. Write 0, ignore on read.		
Tx Suppression	6	1b = Suppress Tx during all digital loopback. 0b = Do not suppress Tx during all digital loopback.	R/W	1b
External Cable	7	1b = External cable loopback enabled. 0b = External cable loopback disabled.	R/W	0b
Reserved	8	Reserved. Write 0, ignore on read.		
Remote	9	1b = Remote loopback enabled. 0b = Remote loopback disabled.	R/W	0b
Line Driver	10	1b = Line driver loopback selected. 0b = Line driver loopback not selected.	R/W	0b
Reserved	11	Reserved. Write 0, ignore on read.		
All Digital	12	1b = All digital loopback selected. 0b = All digital loopback not selected.	R/W	1b
Reserved	14:13	Reserved. Write 0, ignore on read.		
MII	15	1b = MII loopback selected. 0b = MII loopback not selected.	R/W	0b

[1] This bit can be used to force link status okay during MII loopback. In MII loopback, the link status bit is not set unless force link status is used. In all other loopback modes, the link status bit is set when the link comes up.

8.24.2.14.1 Loopback Mode Setting

Table 8-27 shows how the loopback bit (0.14) and the LNK_EN bit (23.13) should be set for each loopback mode. It also indicates whether the loopback mode sets the link status bit and when the PHY is ready to receive data.

Table 8-27. Loopback Bit (0.14) Settings for Loopback Mode

Loopback	Bit 0.14 = 1 Loopback Required	Bit 26.6 Link Status Set	PHY Ready for Data
MII	Yes	19.0	After a few ms
All Digital	Yes	Yes	Link Status
Line Driver	Yes	Yes	Link Status
Ext Cable	No	Yes	Link Status
Remote	No	Yes	Never

8.24.2.15 RX Error Counter Register - PHRERRC (20d; RO)

Field	Bit(s)	Description	Mode	Default
Rx Error Counter[1]	15:0	16-bit Rx error counter. This register is clear-on-read.	RO, SC	0x0

[1] See Register 18d (bits 13 and 14) for error type descriptions.

8.24.2.16 Management Interface (MI) Control Register - PHMIC (21d; R/W)

Field	Bit(s)	Description	Mode	Default
Auto Negotiation to 1000 disable	0	Disable auto-negotiation to 1000BASE-T. Note: Bit should be 0b to allow enabling 1G operation via the *PHPM.Disable 1000* and *PHPM.Disable 1000 in non-D0a* bits.	R/W	0b
phy_in_nrg_pd	1	Energy Detect (Cable Disconnect) Status. When set it indicates that the PHY has entered the energy detect power-down mode because energy detect power-down is enabled and no energy has been detected on the line for 4s (cable disconnected).	RO	x
nrg_pd_tx_en	2	Software enable for periodic NLP transmission in energy detect power-down. 1b = Enables NLP transmission during energy-detect powerdown. 0b = Disables NLP transmission during energy-detect powerdown. Note: Bit should be 0b to allow the *PHPM.SPD_B2B_EN* bit to disable Smart Power Down Back to Back operation.	R/W	1b
Energy Detect Powerdown Enable	3	1b = Enables energy detect powerdown. 0b = Disables energy detect powerdown. Note: Bit should be 0b to allow the *PHPM.SPD_EN* bit to disable Smart Power Down operation.	R/W	1b
Reserved	15:4	Reserved. Write 0, ignore on read.		

8.24.2.17 PHY Configuration Register - PHCNFG (22d; R/W)

Field	Bit(s)	Description	Mode	Default
Reserved	2:0	Reserved. Write 0, ignore on read.		
Reserved_1	3	Reserved. Write 1, ignore on read.		
Reserved	4	Reserved. Write 0, ignore on read.		
Transmit Clock Enable	5	1b = Enables output of mixer clock (transmit clock in 1000Base-T). 0b = Disables output.	R/W	0b
Group MDIO Mode Enable	6	When this bit is set, the PHY processes MDIO accesses to the group address 31 as if they are accesses to it's own PHY address. 1b = Enables group MDIO mode. 0b = Disables Group MDIO mode.	R/W	0b
Alternate Next-Page	7	1b = Enables manual control of 1000Base-T next pages only. 0b = Normal operation of 1000Base-T next page exchange.	R/W	0b
Reserved	9:8	Reserved. Write 0, ignore on read.		
Automatic Speed Downshift Mode[1]	11:10	00b = Automatic speed downshift disabled. 01b = 10Base-T downshift enabled. 10b = 100Base-TX downshift enabled. 11b = 100Base-TX and 10Base-T enabled.	R/W	11b
Transmit FIFO depth (1000Base-T)	13:12	00b= ±8. 01b = ±16. 10b = ±24. 11b = ±32.	R/W	11b
Ignore 10G Frames	14	1b = Management frames with ST = <00> are ignored. 0b = Management frames with ST = <00> are treated as wrong frames	R/W	1b
CRS Transmit Enable	15	1b = Enables CRS on transmit in half-duplex mode. 0b = Disables CRS on transmit.	R/W	0b

[1] If automatic speed downshift is enabled and the PHY fails to autonegotiate at 1000Base-T, the PHY falls back to attempt connection at 100Base-TX and, subsequently, 10Base-T. This cycle repeats. If the link is broken at any speed, the PHY restarts this process by re-attempting connection at the highest possible speed (e.g., 1000Base-T).

8.24.2.18 PHY Control Register 1 - PHCTRL1 (23d; R/W)

Field	Bit(s)	Description	Mode	Default
Force Interrupt	0	1b = Assert PHY interrupt. 0b = Deassert PHY interrupt.	R/W	0b
Reserved	1	Reserved. Write 0, ignore on read.		
10BASE-T Preamble Length	3:2	00b = 10BASE-T preamble length of 0 octets in the received frames sent over the MII. 01b = 10BASE-T preamble length of 1 octets. 10b = 10BASE-T preamble length of 2 octets. 11b = 10BASE-T preamble length of 7 octets.	R/W	10b
10BASE-T MAU Loopback Function Enable (10BASE-T)	4	1b = Enables MAU loopback function (half-duplex only). 0b = Disables MAU loopback function.	R/W	0b
SQE Test Enable (10Base-T)	5	1b = Enables heartbeat. 0b = Disables heartbeat.	R/W	0b
Jabber Enable (10Base-T)	6	1b = Disables jabber. 0b = Normal operation.	R/W	1b
Link Partner Detected[1]	7	1b = Link partner detected. 0b = Link partner not detected	RO, LH	0b
Reverse Auto-Negotiation	8	1b = Reverse Auto-negotiation Enabled. 0b = Reverse Auto-negotiation Disabled. When bit is set device attempts to Auto-negotiate to lowest common denominator (LCD). Note: Bit should be 0b to enable the *PHPM.LPLU* and *PHPM.D0LPLU* bits to disable Low Power Link Up operation.	R/W	0b
Reserved	9	Reserved. Write 0, ignore on read.		
Link Attempts Before Automatic Speed Downshift	12:10	000b = 1. 001b = 2. 010b = 3. 011b = 4. 100b = 5. 101b = 6. 110b = 7. 111b = 8.	R/W	100b
LNK_EN[2]	13	1b = Enables linking. 0b = Disables linking.	R/W	1b
IP Phone Detect Enable[3]	14	1b = Enables automatic IP phone detect. 0b = Disables automatic IP phone detect.	R/W, SC	0b
IP Phone Detected[4]	15	1b = IP phone detected. 0b = IP phone not detected.	RO	0b

1. When linking is disabled, the PHY automatically monitors for the appearance of a link partner and sets this bit if detected. Linking is disabled when LNK_EN is cleared (23.13 = 0).
2. If LNK_EN is set, the PHY attempts to bring up a link with a remote partner and will monitor the MDI for link pulses. If LNK_EN is cleared the PHY takes down any active link, goes into standby, and does not respond to link pulses from a remote link partner. In standby, IP phone detect and TDR functions are available.

3. When this bit is set, the PHY performs automatic IP phone detection whenever linking is disabled. Linking is disabled when LNK_EN is cleared (23.13 = 0). If an IP phone is detected it is indicated in 23.15.
4. When linking is disabled, the PHY automatically monitors for the appearance of a link partner and sets this bit if detected. Linking is disabled when LNK_EN is cleared (23.13 = 0).

8.24.2.19 Interrupt Mask Register - PHINTM (24d; R/W)

Field	Bit(s)	Description	Mode	Default
MDINT_N Enable	0	1b = PHY interrupt enabled. 0b = PHY interrupt disabled.	R/W	0b
Automatic Speed Downshift	1	1b = Interrupt enabled. 0b = Interrupt disabled.	R/W	0b
Link Status Change	2	1b = Interrupt enabled. 0b = Interrupt disabled.	R/W	0b
Receive Status Change	3	1b = Interrupt enabled. 0b = Interrupt disabled.	R/W	0b
FIFO Overflow/Underflow	4	1b = Interrupt enabled. 0b = Interrupt disabled.	R/W	0b
Error Counter Full	5	1b = Interrupt enabled. 0b = Interrupt disabled.	R/W	0b
Next Page Received	6	1b = Interrupt enabled. 0b = Interrupt disabled.	R/W	0b
CRC Errors	7	1b = Interrupt enabled. 0b = Interrupt disabled.	R/W	0b
Autonegotiation Status Change	8	1b = Interrupt enabled. 0b = Interrupt disabled.	R/W	0b
MDIO Sync Lost	9	1b = Interrupt enabled. 0b = Interrupt disabled.	R/W	0b
TDR/IP Phone	10	1b = Interrupt enabled. 0b = Interrupt disabled.	R/W	0b
Reserved	15:11	Reserved. Write 0, ignore on read.		

8.24.2.20 Interrupt Status Register - PHINT (25d; RC)

The Interrupt Status Register reports interrupt conditions detected in the internal PHY. An interrupt bit that is set and is not masked via the *PHINTM* register will assert the *ICR.GPHY* bit and generate a Host interrupt if the bit is not masked. To clear the interrupt the Host should read the *PHINT* register before clearing the *ICR.GPHY* bit.

Field	Bit(s)	Description	Mode	Default
MII Interrupt Pending[1]	0	1b = Interrupt pending. 0b = No interrupt pending.	RC, LH	0b
Automatic Speed Downshift	1	1b = Event has occurred. 0b = Event has not occurred.	RC, LH	0b
Link Status Change	2	1b = Event has occurred. 0b = Event has not occurred.	RC, LH	0b
Receive Status Change	3	1b = Event has occurred. 0b = Event has not occurred.	RC, LH	0b
FIFO Overflow/Underflow	4	1b = Event has occurred. 0b = Event has not occurred.	RC, LH	0b
Error Counter Full	5	1b = Event has occurred. 0b = Event has not occurred.	RC, LH	0b
Next Page Received	6	1b = Event has occurred. 0b = Event has not occurred.	RC, LH	0b
CRC Errors	7	1b = Event has occurred. 0b = Event has not occurred.	RC, LH	0b
Autonegotiation Status Change	8	1b = Event has occurred. 0b = Event has not occurred.	RC, LH	0b
MDIO Sync Lost[2]	9	1b = Event has occurred. 0b = Event has not occurred.	RC, LH	0b
TDR/IP Phone	10	1b = Event has occurred. 0b = Event has not occurred.	RC, LH	0b
Reserved	15:11	Reserved. Write 0, ignore on read.		

[1] An event has occurred and the corresponding interrupt mask bit is enabled (set = 1).

[2] If the management frame preamble is suppressed (MF preamble suppression, register 0, bit 6), it is possible for the PHY to lose synchronization if there is a glitch at the interface. The PHY can recover if a single frame with a preamble is sent to the PHY. The MDIO sync lost interrupt can be used to detect loss of synchronization and, thus, enable recovery.

8.24.2.21 PHY Status Register - PHSTAT (26d; RO)

Field	Bit(s)	Description	Mode	Default
Link partner advertised asymmetric PAUSE	0	1b = Link partner advertised asymmetric PAUSE. 0b = Link partner did not advertised asymmetric PAUSE.	RO	0b
Link partner advertised PAUSE	1	1b = Link partner advertised PAUSE. 0b = Link partner did not advertised PAUSE.	RO	0b
Autonegotiation Enabled	2	1b = Both partners have autonegotiation enabled. 0b = Both partners do not have autonegotiation enabled.	RO	0b
Collision Status	3	1b = Collision occurring. 0b = Collision not occurring.	RO	0b
Receive Status	4	1b = PHY receiving a packet. 0b = PHY not receiving a packet.	RO	0b
Transmit Status	5	1b = PHY transmitting a packet. 0b = PHY not transmitting a packet.	RO	0b

Field	Bit(s)	Description	Mode	Default
Link Status	6	1b = Link up. 0b = Link down.	RO	0b
Duplex Status	7	1b = Full duplex. 0b = Half duplex.	RO	0b
Speed Status	9:8	11b = Undetermined. 10b = 1000Base-T. 01b = 100Base-TX. 00b = 10Base-T.	RO	11b
Polarity Status	10	1b = Polarity inverted (10Base-T only). 0b = Polarity normal (10Base-T only).	RO	0b
Pair Swap on Pairs A and B	11	1b = Pairs A and B swapped. 0b = Pairs A and B not swapped.	RO	0b
Autonegotiation Status	12	1b = Autonegotiation complete. 0b = Autonegotiation not complete.	RO	0b
Autonegotiation Fault Status	14:13	11b = Reserved. 10b = Master/slave autonegotiation fault. 01b = Parallel detect autonegotiation fault. 00b = No autonegotiation fault.	RO	00b
PHY in Standby Mode[1]	15	1b = PHY in standby mode. 0b = PHY not in standby mode.	RO	0b

[1] This bit indicates that the PHY is in standby mode and is ready to perform IP phone detection or TDR cable diagnostics. The PHY enters standby mode when LNK_EN is cleared (23.13 = 0) and exits standby mode and attempts to autonegotiate a link when LNK_EN is set (23.13= 1).

8.24.2.22 Diagnostics Control Register (Linking Disabled) - PHDIAG (30d; R/W)

Field	Bit(s)	Description	Mode	Default
Reserved	9:0	Reserved. Write 0, ignore on read.		
TDR Rx Dim[1]	11:10	Receive dimension for single-pair TDR analysis: 00b = TDR receive on pair A. 01b = TDR receive on pair B. 10b = TDR receive on pair C. 11b = TDR receive on pair D.	R/W	00b
TDR Tx Dim[2]	13:12	Transmit dimension for single-pair TDR analysis/first dimension to be reported for automatic TDR analysis: 00b = TDR transmit on pair A. 01b = TDR transmit on pair B. 10b = TDR transmit on pair C. 11b = TDR transmit on pair D.	R/W	00b
TDR Request[3]	15:14	11b = Automatic TDR analysis in progress. 10b = Single-pair TDR analysis in progress. 01b = TDR analysis complete, results valid. 00b = TDR analysis complete, results invalid.	R/W, SC	00b

[1] The TDR receive dimension is only valid for single-pair TDR analysis. It is ignored for automatic TDR analysis when all ten pair combinations are analyzed.

[2] The TDR transmit dimension is only valid for single-pair TDR analysis. For automatic TDR analysis, these bits specify the first dimension to be reported in register 31.

[3] Automatic TDR analysis is enabled by setting TDR request to {11}. All ten combinations of pairs are analyzed in sequence, and the results are available in register 31. TDR analysis for a single pair combination can be enabled by setting TDR request to {10}. Linking must be disabled (23.13 = 0) and IP phone detect must be disabled (23.14 = 0) to do TDR operations. Bit 15 self-clears when the TDR operation is complete. When TDR is complete, bit 14 indicates if the results are valid.

8.24.2.23 Diagnostics Status Register (Linking Disabled) - PHDSTAT (31d; RO)

Field	Bit(s)	Description	Mode	Default
Pair Indication[1]	1:0	00b = Results are for pair A. 01b = Results are for pair B. 10b = Results are for pair C. 11b = Results are for pair D.	RO	00b
Distance to Fault[2,3]	9:2	Distance to first open, short, or SIM fault on pair X.	RO	0x0
Short Between Pairs X and A[4]	10	1b = Short between pairs X and A. 0b = No short between pairs X and A.	RO	0b
Short Between Pairs X and B[4]	11	1b = Short between pairs X and B. 0b = No short between pairs X and B.	RO	0b
Short Between Pairs X and C[4]	12	1b = Short between pairs X and C. 0b = No short between pairs X and C.	RO	0b
Short Between Pairs X and D[4]	13	1b = Short between pairs X and D. 0b = No short between pairs X and D.	RO	0b
TDR Fault Type Pair X[5,6,7]	15:14	11b = Result invalid. 10b = Open or short found on pair X. 01b = Strong impedance mismatch found on pair X. 00b = Good termination found on pair X.	RO	11b

[1] This indicates the pair to which the results in register bits 31.15:2 correspond.

[2] The first time this register is read after automatic TDR analysis has completed, it indicates the distance to the first fault on pair A. The second time it is read, it indicates the distance to the first fault on pair B; the third time, on pair C; and the fourth time, on pair D. It then cycles back to pair A. Pair indication bits 31.1:0 indicate to which pair the results correspond. Bits 30.13:12 can be used to specify a pair other than pair A as the first dimension to be reported.

[3] This 8-bit integer value is the distance in meters. The value 0xff indicates an unknown result.

[4] The first time these bits are read after automatic TDR analysis has completed, they indicate a short between pair A and pair A, B, C, and D, respectively. The second time they are read, they indicate a short between pair B and pair A, B, C, and D, respectively. The third time, with pair C; and the fourth time, with pair D. It then cycles back to pair A. Pair indication bits 31.1:0 indicate to which pair the results correspond. Bits 30.13:12 can be used to specify a pair other than pair A as the first dimension to be reported.

[5] The first time this register is read after automatic TDR analysis has completed, it indicates the fault type for pair A. The second time it is read, it indicates the fault type for pair B; the third, for pair C; and the fourth time, for pair D. It then cycles back to pair A. Pair indication bits 31.1:0 indicate pair to which the results correspond. Bits 30.13:12 can be used to specify a pair other than pair A as the first dimension to be reported.

[6] A value of 01b indicates either an open or a short. If 31.13:10 = 0000b, it is an open. For all other values of 31.13:10, each bit indicates a short to pair A, B, C and D.

[7] A value of 11 indicates that the results for this pair are invalid. An invalid result usually occurs when unexpected pulses are received during the TDR operation, e.g., from a remote PHY also doing TDR or trying to bring up a link. When an invalid result is indicated, the distance in bits 31.9:2 is 0xff and should be ignored.

8.24.2.24 Diagnostics Status Register (Linking Enabled) - PHDSTAT (31d; RO)

Field	Bit(s)	Description	Mode	Default
Excessive Pair Skew	0	1b = Excessive pair skew (1000BASE-T only). 0b = Not excessive pair skew (1000BASE-T only).	RO	0b
Reserved	1	Reserved. Write 0, ignore on read.		
Cable Length	9:2	Cable length when the link is active. This 8-bit integer value is the cable length in meters when the link is active. The value 0xFF indicates an unknown result.	RO	0xFF

Field	Bit(s)	Description	Mode	Default
Polarity on Pair A	10	1b = Polarity on pair A is inverted (10BASE-T or 1000BASE-T). 0b = Polarity on pair A is normal (10BASE-T or 1000BASE-T).	RO	0b
Polarity on Pair B	11	1b = Polarity on pair B is inverted (10BASE-T or 1000BASE-T). 0b = Polarity on pair B is normal (10BASE-T or 1000BASE-T).	RO	0b
Polarity on Pair C	12	1b = Polarity on pair C is inverted (1000BASE-T only). 0b = Polarity on pair C is normal (1000BASE-T only).	RO	0b
Polarity on Pair D	13	1b = Polarity on pair D is inverted (1000BASE-T only). 0b = Polarity on pair D is normal (1000BASE-T only).	RO	0b
Pair Swap on Pairs C and D[1]	14	1b = Pairs C and D are swapped (1000BASE-T only). 0b = Pairs C and D are not swapped (1000BASE-T only).	RO	0b
Reserved	15	Reserved. Write 0, ignore on read.		

[1] When bit is set, the PHY detects the crossover of the received pair 2 (RJ-45 pins 4 and 5) and pair 3 (RJ-45 pins 7 and 8).

§ §

NOTE: This page intentionally left blank.

9.0 PCIe Programming Interface

9.1 PCIe* Compatibility

PCIe is completely compatible with existing deployed PCI software. To achieve this, PCIe hardware implementations conform to the following requirements:

- All devices required to be supported by deployed PCI software must be enumerable as part of a tree through PCI device enumeration mechanisms.
- Devices in their default operating state must conform to PCI ordering and cache coherency rules from a software viewpoint.
- PCIe devices must conform to PCI power management specifications and must not require any register programming for PCI-compatible power management beyond those available through PCI power management capabilities registers. Power management is expected to conform to a standard PCI power management by existing PCI bus drivers.
- PCIe devices implement all registers required by the PCI specification as well as the power management registers and capability pointers specified by the PCI power management specification. In addition, PCIe defines a PCIe capability pointer to indicate support for PCIe extensions and associated capabilities.

The 82580 is a multi-function device with the following functions:

- LAN 0
- LAN 1
- LAN 2
- LAN 3

All functions contain the following regions of the PCI configuration space:

- Mandatory PCI configuration registers
- Power management capabilities
- MSI and MSI-X capabilities
- PCIe extended capabilities

Different parameters affect how LAN functions are exposed on the PCIe*. Table 9-1 describes the various mapping options.

Table 9-1. 82580 Function Mapping Options

Function Number	Function Description	Disable options
0 or 3	LAN 0	Strapping option
1 or 0 or 2	LAN 1	Strapping option/ *Software Defined Pins Control* (Offset 0x20) EEPROM word in LAN1 section, bit 11
2 or 1 or 0	LAN 2	Strapping option/ *Software Defined Pins Control* (Offset 0x20) EEPROM word in LAN2 section, bit 11
3 or 0	LAN 3	Strapping option/ *Software Defined Pins Control* (Offset 0x20) EEPROM word in LAN3 section, bit 11

The mapping of each port to a PCIe function is influenced by the number of functions enabled, the dummy function mode selected and the function select word in the EEPROM. See Section 4.4 for description of the function mapping when part of the functions are disabled.

9.2 Configuration Sharing Among PCI Functions

The 82580 contains a single physical PCIe core interface. The 82580 is designed so that each of the logical LAN ports appears as a distinct function. Many of the fields of the PCIe header space contain hardware default values that are either fixed or might be overridden by data from the EEPROM, but may not be independently specified for each function. The following fields are considered to be common to all LAN functions:

Table 9-2. Common Fields for LAN Devices

Vendor ID	The Vendor ID of the 82580 can be specified via EEPROM, but only a single value can be specified. The value is reflected identically for all functions. Default value is 0x8086. The value is reflected identically for all LAN devices.
Revision	The revision number of the 82580 is reflected identically for all LAN functions.
Header Type	This field indicates if a device is single function or multifunction. The value reflected in this field is reflected identically for all LAN functions, but the actual value reflected depends on LAN disable configuration. When more than one the 82580 LAN function is enabled, all PCIe headers return 0x80 in this field, acknowledging being part of a multi-function device. If only a single function is enabled, then a *single-function device* is indicated (this field returns a value of 0x00) and the LAN exists as device function 0 (when dummy mode is not enabled). See Table 9-6 for details.
Subsystem ID	The subsystem ID of the 82580 can be specified via EEPROM, but only a single value can be specified. The value is reflected identically for all LAN functions.
Subsystem Vendor ID	The 82580 subsystem vendor ID can be specified via EEPROM, but only a single value can be specified. The value is reflected identically for all LAN functions.
Cap_Ptr, Max Latency, Min Grant	These fields reflect fixed values that are constant values reflected for all LAN functions.

The following fields are implemented individually for each LAN function:

Table 9-3. Fields Implemented Differently in LAN Devices

Device ID	The device ID reflected for each LAN function can be independently specified via EEPROM.
Command, Status	Each LAN function implements its own command/status registers.
Latency Timer, Cache Line Size	Each LAN function implements these registers individually. The system should program these fields identically for each LAN to ensure consistent behavior and performance of the device.
Memory BAR, IO BAR, Expansion ROM BAR, MSI-X BAR	Each LAN function implements its own base address registers, enabling each function to claim its own address region(s). The IO BAR and Flash BAR are supported depending on BAR32 setting in the EEPROM (See Section 9.4.11).
Interrupt Pin	Each LAN function independently indicates which interrupt pin (INTA#, INTB#, INTC# or INTD#) is used by that device's MAC to signal system interrupts. The value for each LAN device can be independently specified via EEPROM, but only if more than one LAN function is enabled.
Class Code	Different class code values (iSCSI/LAN) can be set for each function.

9.3 PCIe Register Map

9.3.1 Register Attributes

Configuration registers are assigned one of the attributes described in the following table.

Table 9-4. Configuration Registers

Rd/Wr	Description
RO	Read-only register: Register bits are read-only and cannot be altered by software.
RW	Read-write register: Register bits are read-write and can be either set or reset.
R/W1C	Read-only status, write-1-to-clear status register, writing a 0b to R/W1C bits has no effect.
ROS	Read-only register with sticky bits: Register bits are read-only and cannot be altered by software. Bits are not cleared by reset and can only be reset with the PWRGOOD signal. Devices that consume AUX power are not allowed to reset sticky bits when AUX power consumption (either via AUX power or PME enable) is enabled.
RWS	Read-write register: Register bits are read-write and can be either set or reset by software to the desired state. Bits are not cleared by reset and can only be reset with the PWRGOOD signal. Devices that consume AUX power are not allowed to reset sticky bits when AUX power consumption (either via AUX power or PME enable) is enabled.
R/W1CS	Read-only status, write-1-to-clear status register: Register bits indicate status when read, a set bit indicating a status event can be cleared by writing a 1b. Writing a 0b to R/W1C bits has no effect. Bits are not cleared by reset and can only be reset with the PWRGOOD signal. Devices that consume AUX power are not allowed to reset sticky bits when AUX power consumption (either via AUX power or PME enable) is enabled.
HwInit	Hardware initialized: Register bits are initialized by firmware or hardware mechanisms such as pin strapping or serial EEPROM. Bits are read-only after initialization and can only be reset (for write-once by firmware) with PWRGOOD signal.
RsvdP	Reserved and preserved: Reserved for future R/W implementations; software must preserve value read for writes to bits.
RsvdZ	Reserved and zero: Reserved for future R/W1C implementations; software must use 0b for writes to bits.

The PCI configuration registers map is listed in Table 9-5. Refer to a detailed description for registers loaded from the EEPROM at initialization time. Note that initialization values of the configuration registers are marked in parenthesis.

9.3.2 PCIe Configuration Space Summary

Table 9-5. PCIe Configuration Registers Map

Section	Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
Mandatory PCI register	0x0	Device ID		Vendor ID	
	0x4	Status Register		Control Register	
	0x8	Class Code (0x020000/0x010000)			Revision ID
	0xC	BIST (0x00)	Header Type (0x0/0x80)	Latency Timer	Cache Line Size (0x10)
	0x10	Base Address Register 0			
	0x14	Base Address Register 1			
	0x18	Base Address Register 2			
	0x1C	Base Address Register 3			
	0x20	Base Address Register 4			
	0x24	Base Address Register 5			
	0x28	CardBus CIS pointer (0x0000)			
	0x2C	Subsystem Device ID		Subsystem Vendor ID	
	0x30	Expansion ROM Base Address			
	0x34	Reserved			Cap Ptr (0x40)
	0x38	Reserved			
	0x3C	Max Latency (0x00)	Min Grant (0x00)	Interrupt Pin (0x01…0x04)	Interrupt Line (0x00)
Power management capability	0x40	Power Management Capabilities		Next Pointer (0x50)	Capability ID (0x01)
	0x44	Data	Bridge Support Extensions	Power Management Control & Status	
MSI capability	0x50	Message Control (0x0080)		Next Pointer (0x70)	Capability ID (0x05)
	0x54	Message Address			
	0x58	Message Upper Address			
	0x5C	Reserved		Message Data	
	0x60	Mask bits			
	0x64	Pending bits			
MSI-X capability	0x70	Message Control (0x00090)		Next Pointer (0xA0)	Capability ID (0x11)
	0x74	Table Offset			
	0x78	PBA offset			
CSR Access Registers	0x98	IOADDR			
	0x9C	IODATA			

Table 9-5. PCIe Configuration Registers Map (Continued)

Section	Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
PCIe capability	0xA0	PCIe Capability Register (0x0002)		Next Pointer (0xE0)	Capability ID (0x10)
	0xA4	Device Capability			
	0xA8	Device Status		Device Control	
	0xAC	Link Capability			
	0xB0	Link Status		Link Control	
	0xB4	Reserved			
	0xB8	Reserved		Reserved	
	0xBC	Reserved			
	0xC0	Reserved		Reserved	
	0xC4	Device Capability 2			
	0xC8	Reserved		Device Control 2	
	0xCC	Reserved			
	0xD0	Link Status 2		Link Control 2	
	0xD4	Reserved			
	0xD8	Reserved		Reserved	
VPD capability	0xE0	VPD address		Next Pointer (0x00)	Capability ID (0x03)
	0xE4	VPD data			
AER capability	0x100	Next Capability Ptr. (0x140/0x1A0)	Version (0x1)	AER Capability ID (0x0001)	
	0x104	Uncorrectable Error Status			
	0x108	Uncorrectable Error Mask			
	0x10C	Uncorrectable Error Severity			
	0x110	Correctable Error Status			
	0x114	Correctable Error Mask			
	0x118	Advanced Error Capabilities and Control Register			
	0x11C: 0x128	Header Log			
Serial ID capability	0x140	Next Capability Ptr. (0x1A0)	Version (0x1)	Serial ID Capability ID (0x0003)	
	0x144	Serial Number Register (Lower Dword)			
	0x148	Serial Number Register (Upper Dword)			
TPH Requester capability	0x1A0	Next Capability Ptr. (0x1C0/0x000)	Version (0x1)	TPH Capability ID (0x17)	
	0x1A4	TPH Requester Capability Register			
	0x1A8	TPH Requester Control Register			
	0x1AC: 0x1B8	TPH Steering Table			
LTR capability	0x1C0	Next Capability Ptr. (0x000)	Version (0x1)	LTR Capability ID (0x18)	
	0x1C4	Maximum Non-Snooped Platform Latency Tolerance Register		Maximum Snooped Platform Latency Tolerance Register	

A description of the registers is provided in the following sections.

9.4 Mandatory PCI Configuration Registers

9.4.1 Vendor ID (0x0; RO)

This value can be loaded automatically from EEPROM address 0x0E 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.

Note: To avoid a system hang situation, if a value of 0xFFFF is read from the EEPROM, the value of the Vendor ID field defaults back to 0x8086.

9.4.2 Device ID (0x2; RO)

This is a read-only register. This field identifies individual 82580 functions. It has the same default value for all LAN functions but can be auto-loaded from the EEPROM during initialization with a different value for each port. The following table describes the possible values according to the SKU and functionality of each function.

PCI Function	Default Value	EEPROM Address	Meaning
LAN 0	0x1509	0x0D 0x1D (Dummy Function)	0x1509 - Quad port EEPROM-less Device ID (Default)[1] 0x150E - Quad port 10/100/1000 Mb/s Ethernet controller, x4 PCIe, copper.[2] 0x150F - Quad port 10/100/1000 Mb/s Ethernet controller, x4 PCIe, Fiber.[3] 0x1510 - Quad port 10/100/1000 Mb/s Ethernet controller, x4 PCIe, 1000BASE-KX/1000BASE-BX backplane[4]. 0x1511 - Quad port 10/100/1000 Mb/s Ethernet controller, x4 PCIe, External SGMII PHY.[5] 0x1516 - Dual port 10/100/1000 Mb/s Ethernet controller, x4 PCIe, copper.[2] 0x10A6 – Dummy function[6].
LAN 1	0x1509	0x11	Same as port zero.
LAN 2	0x1509	0x44	Same as port zero.
LAN 3	0x1509	0x4A	Same as port zero.

1. Default ID.
2. *CTRL_EXT.Link_Mode* field value 00b (10/100/1000 BASE-T internal PHY mode).
3. *CTRL_EXT.Link_Mode* field value 11b (SerDes).
4. *CTRL_EXT.Link_Mode* field value either 01b (1000BASE-KX) or 11b (SerDes - 1000BASE-BX). User option to enable Clause 37 Auto-negotiation.
5. *CTRL_EXT.Link_Mode* field value 10b (SGMII).
6. The Dummy function device ID is loaded from the *Dummy Device ID* EEPROM word and is used according to the disable status of the function. It is applicable only for function 0. See Section 6.2.7 for details.

9.4.3 Command Register (0x4; R/W)

This is a read/write register. Each function has its own command register. Unless explicitly specified, functionality is the same in all functions.

Bit(s)	R/W	Initial Value	Description
0	R/W[1]	0b	I/O Access Enable For LAN functions this field is R/W. For Dummy function this field is RO as zero.
1	R/W	0b	Memory Access Enable For LAN functions this field is R/W. For Dummy function this field is RO as zero.
2	R/W	0b	Bus Master Enable (BME) For LAN functions this field is R/W. For Dummy function this field is RO as zero.
3	RO	0b	Special Cycle Monitoring Hardwired to 0b.
4	RO	0b	MWI Enable Hardwired to 0b.
5	RO	0b	Palette Snoop Enable Hardwired to 0b.
6	RW	0b	Parity Error Response
7	RO	0b	Wait Cycle Enable Hardwired to 0b.
8	RW	0b	SERR# Enable
9	RO	0b	Fast Back-to-Back Enable Hardwired to 0b.
10	RW	0b	Interrupt Disable[2].
15:11	RO	0x0	Reserved

[1] If IO_Sup bit in PCIe Init Configuration 2 EEPROM Word (0x19) is 0, I/O Access Enable bit is RO with a value of 0.

[2] The Interrupt Disable register bit is a read-write bit that controls the ability of a PCIe device to generate a legacy interrupt message. When set, devices are prevented from generating legacy interrupt messages.

9.4.4 Status Register (0x6; RO)

Each function has its own status register. Unless explicitly specified, functionality is the same in all functions.

Bits	R/W	Initial Value	Description
2:0		000b	Reserved
3	RO	0b	Interrupt Status[1]
4	RO	1b	New Capabilities Indicates that a device implements extended capabilities. The 82580 sets this bit, and implements a capabilities list, to indicate that it supports PCI power management, Message Signaled Interrupts (MSI), Enhanced Message Signaled Interrupts (MSI-X), Vital Product Data (VPD), and the PCIe extensions.
5		0b	66 MHz Capable Hardwired to 0b.
6		0b	Reserved
7		0b	Fast Back-to-Back Capable Hardwired to 0b.
8	R/W1C	0b	Data Parity Reported

Bits	R/W	Initial Value	Description
10:9		00b	DEVSEL Timing Hardwired to 0b.
11	R/W1C	0b	Signaled Target Abort
12	R/W1C	0b	Received Target Abort
13	R/W1C	0b	Received Master Abort
14	R/W1C	0b	Signaled System Error
15	R/W1C	0b	Detected Parity Error

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

9.4.5 Revision (0x8; RO)

The default revision ID of the 82580 is 0x01. The value of the rev ID is a logic XOR between the default value and the value in EEPROM word 0x1E. Note that all LAN functions have the same revision ID.

9.4.6 Class Code (0x9; RO)

The class code is a RO hard coded value that identifies the 82580's functionality.

- LAN 0...LAN3 - 0x020000/0x010000 - Ethernet/SCSI Adapter[1]

9.4.7 Cache Line Size (0xC; R/W)

This field is implemented by PCIe devices as a read-write field for legacy compatibility purposes but has no impact on any PCIe device functionality. Field is loaded from the *PCIe Init Configuration 3* (Word 0x1A) EEPROM word and defines cache line size in Dwords. All functions are initialized to the same value.

9.4.8 Latency Timer (0xD; RO)

Not used. Hardwired to zero.

9.4.9 Header Type (0xE; RO)

This indicates if a device is single function or multifunction. If a single LAN function is the only active one then this field has a value of 0x00 to indicate a single function device. If other functions are enabled then this field has a value of 0x80 to indicate a multi-function device. The following table lists the different options to set the header type field:

Table 9-6. Header Type Settings

LAN 0	LAN 1	LAN 2	LAN 3	Cross Mode Enable	Dummy Function Enable	Header Type Expected Value
Disabled	Disabled	Disabled	Disabled	X	X	N/A (no function)
Only one function enabled				X	0	0x00

[1] Selected according to bit 11, 12, 13 or 14 in *Device Rev ID* EEPROM word for LAN0, LAN 1, LAN2 or LAN3 respectively.

Table 9-6. Header Type Settings

More than one function is enabled		X	X	0x80 (multi function)
Disabled	At least on of functions 1 - 3 is enabled	0	1	0x80 (dummy exist)
At least one of functions 0 to 3 is enabled	Disabled	1	1	0x80 (dummy exist)

9.4.10 BIST (0xF; RO)

BIST is not supported in the 82580.

9.4.11 Base Address Registers (0x10...0x27; R/W)

The Base Address registers (BARs) are used to map the 82580 register space of the various functions. The 82580 has a memory BAR, IO BAR and MSI-X BAR described in Table 9-7 below. The BARs location and sizes are described below. The fields within each BAR are then described in Table 9-10. 32-bit addresses may be used in one register for each memory mapping window or 64-bit addresses with 2 registers for each memory mapping window depending on the *BARCTRL.BAR32* bit.

Table 9-7. Base Address Registers description - LAN 0...3

Mapping Windows	Mapping Description
Memory BAR	The internal registers memories and external FLASH device are accessed as direct memory mapped offsets from the Base Address register. Software can access a Dword or 64 bits. The FLASH space in this BAR is enabled by the FLSize and CSRSize fields in the BARCTRL register. Address 0 in the FLASH device is mapped to address 128K in the Memory BAR. When the usable FLASH size + CSR space is smaller than the memory BAR, then accessing addresses above the top of the FLASH wraps back to the beginning of the FLASH.
IO BAR	All internal registers and memories can be accessed using I/O operations. There are two 4-byte registers in the IO mapping window: Addr Reg and Data Reg accessible as Dword entities. IO BAR support depends on the *IO_Sup* bit in the EEPROM "*PCIe Init Configuration 2*" word.
MSI-X BAR	The MSI-X vectors and Pending bit array (PBA) structures are accessed as direct memory mapped offsets from the MSI-X BAR. Software can access Dword entities.

9.4.11.1 32-bit BARs Mode Mapping

This mapping is selected when bit 10 in the *Functions Control* EEPROM word is equal to 1b.

Table 9-8. Base Address setting in 32bit BARs mode (*BARCTRL.BAR32* = 1b)

BAR	Addr	31 5	4	3	2	1	0
0	0x10	Memory CSR + FLASH BAR (R/W - 31:17; 0b - 16:4)		0/1	0	0	0
1	0x14	Reserved (read as all 0b's)					
2	0x18	IO BAR (R/W - 31:5)	0	0	0	0	1
3	0x1C	MSI-X BAR (R/W - 31:14; 0b - 13:4)		0/1	0	0	0
4	0x20	Reserved (read as all 0b's)					
5	0x24	Reserved (read as all 0b's)					

9.4.11.2 64-bit BARs Mode Mapping

This mapping is selected when bit 10 in the *Functions Control* EEPROM word is equal to 0b.

Table 9-9. Base Address setting in 64bit BARs mode (*BARCTRL.BAR32* = 0b)

<table>
<tr><th>BAR</th><th>Addr</th><th>31 5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>0</td><td>0x10</td><td colspan="2">Memory CSR + FLASH BAR Low (RW - 31:17;RO 0b - 16:4)</td><td>0/1</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0x14</td><td colspan="6">Memory CSR + FLASH BAR High (RW)</td></tr>
<tr><td>2</td><td>0x18</td><td>IO BAR (R/W - 31:5)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>3</td><td>0x1C</td><td colspan="6">Reserved (RO - 0)</td></tr>
<tr><td>4</td><td>0x20</td><td colspan="2">MSI-X BAR Low (RW - 31:14; RO 0b - 13:4)</td><td>0/1</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>5</td><td>0x24</td><td colspan="6">MSI-X BAR High (RW)</td></tr>
</table>

9.4.11.3 Base Address Register Fields

All base address registers have the following fields.

Table 9-10. Base Address Registers' Fields

<table>
<tr><th>Field</th><th>Bits</th><th>R/W</th><th colspan="2">Description</th></tr>
<tr><td>Mem / IO Space Indication</td><td>0</td><td>RO</td><td colspan="2">0b = Indicates memory space.
1b = Indicates I/O.</td></tr>
<tr><td>Memory Type</td><td>2:1</td><td>RO</td><td colspan="2">00b = 32-bit BAR (BAR32 in the EEPROM equals 1b)
10b = 64-bit BAR (BAR32 in the EEPROM equals 0b)</td></tr>
<tr><td>Prefetch Memory</td><td>3</td><td>R</td><td colspan="2">0b = Non-prefetchable space.
1b = Prefetchable space.
The 82580 implements Non-prefetchable space in memory BAR, since it has read side effects. This bit is loaded from the PREFBAR bit in the EEPROM.</td></tr>
<tr><td rowspan="5">Address Space (Low register for 64bit Memory BARs)</td><td rowspan="5">31:4</td><td rowspan="5">R/W</td><td colspan="2">The length of the RW bits and RO 0b bits depend on the mapping window sizes. Init value of the RW fields is 0x0.</td></tr>
<tr><td>Mapping Window</td><td>RO bits</td></tr>
<tr><td>Memory CSR + FLASH BAR size depends on <i>BARCTRL.FLSize</i> and <i>BARCTRL.CSRSize</i> fields.</td><td>16:4 for 128KB
17:4 for 256KB
and so on...</td></tr>
<tr><td>MSI-X space is 16KB</td><td>13:4</td></tr>
<tr><td>I/O spaces size is 32 bytes</td><td>4:0</td></tr>
</table>

9.4.12 CardBus CIS (0x28; RO)

Not used. Hardwired to zero.

9.4.13 Subsystem Vendor ID (0x2C; RO)

This value can be loaded automatically from EEPROM address 0x0C 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.

9.4.14 Subsystem ID (0x2E; RO)

This value can be loaded automatically from EEPROM address 0x0B at power up with a default value of 0x0000.

9.4.15 Expansion ROM Base Address (0x30; RO)

This register is used to define the address and size information for boot-time access to the optional Flash memory. Expansion ROM is enabled by placing 0b in bit 7 (*LAN Boot Disable*) of the *Initialization Control 3* EEPROM word for LAN 0, LAN1, LAN2 and LAN 3, respectively. This register returns a zero value for functions without an expansion ROM window.

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 hard wired to 0b and dependent on the memory mapping window size. The LAN Expansion ROM spaces can be either 64 KB or up to 8 MB in powers of 2. Mapping window size is set by EEPROM word 0x0F.

9.4.16 Cap_Ptr (0x34; RO)

The *Capabilities Pointer* field (Cap_Ptr) is an 8-bit field that provides an offset in the device's PCI configuration space for the location of the first item in the Capabilities Linked List (CLL). The 82580 sets this bit and implements a capabilities list to indicate that it supports PCI power management, Message Signaled Interrupts (MSIs), and PCIe extended capabilities. Its value is 0x40, which is the address of the first entry: PCI power management.

9.4.17 Interrupt Line (0x3C; RW)

Read/write register programmed by software to indicate which of the system interrupt request lines this 82580's interrupt pin is bound to. See the PCIe definition for more details. Each of the PCIe functions has its own register.

9.4.18 Interrupt Pin (0x3D; RO)

Read only register.

- LAN 0 / LAN 1/LAN2/ LAN3 [1] - A value of 0x1, 0x2, 0x3 or 0x4 indicates that this function implements legacy interrupt on INTA#, INTB#, INTC# or INTD#, respectively. Value is loaded from *Initialization Control 3* (Offset 0x24) EEPROM words from relevant LAN 0, LAN 1, LAN2 and LAN3 EEPROM sections.

Note: If only a single port is enabled while the other ports are disabled, the enabled port uses INTA, independent of the EEPROM setting.

9.4.19 Max_Lat/Min_Gnt (0x3E; RO)

Not used. Hardwired to zero.

9.5 PCI Capabilities

The first entry of the PCI capabilities link list is pointed by the *Cap_Ptr* register. The following table describes the capabilities supported by the 82580.

[1] If only a single device/function of the 82580 component is enabled, this value is ignored and the *Interrupt Pin* field of the enabled device reports INTA# usage.

Address	Item	Next Pointer
0x40-47	PCI Power Management	0x50
0x50-67	Message Signaled Interrupt	0x70
0x70-8B	Extended Message Signaled Interrupt	0xA0
0xA0-DB	PCIe Capabilities	0xE0/0x00[1]
0xE0-0xE7	Vital Product Data Capability	0x00

[1] Next pointer is 0x00 if the VPD area in the EEPROM does not exist.

9.5.1 PCI Power Management Capability

All fields are reset on full power-up. All of the fields except *PME_En* and *PME_Status* are reset on exit from D3cold state. If aux power is not supplied, the *PME_En* and *PME_Status* fields also reset on exit from D3cold state.

See the detailed description for registers loaded from the EEPROM at initialization time. Behavior of some fields in this section depend on the *Power Management* bit in EEPROM word 0x0A.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x40	Power Management Capabilities		Next Pointer (0x50)	Capability ID (0x01)
0x44	Data	Bridge Support Extensions	Power Management Control & Status	

9.5.1.1 Capability ID (0x40; RO)

This field equals 0x01 indicating the linked list item as being the PCI Power Management registers.

9.5.1.2 Next Pointer (0x41; RO)

This field provides an offset to the next capability item in the capability list. Its value of 0x50 points to the MSI capability.

9.5.1.3 Power Management Capabilities - PMC (0x42; RO)

This field describes the 82580’s functionality at the power management states as described in the following table. Note that each device function has its own register.

Bits	Default	R/W	Description
15:11	01001b See value in description column	RO	PME_Support - This 5-bit field indicates the power states in which the function may assert PME#. A value of 0b for any bit indicates that the function is not capable of asserting the PME# signal while in that power state. bit(11) X XXX1b - PME# can be asserted from D0 bit(12) X XX1Xb - PME# can be asserted from D1 bit(13) X X1XXb - PME# can be asserted from D2 bit(14) X 1XXXb - PME# can be asserted from D3hot bit(15) 1 XXXXb - PME# can be asserted from D3cold Value of bit 15 is a function of Aux Pwr availability and *Power Management* (PM Ena) bit in *Initialization Control Word 1* (word 0x0A) EEPROM word. Condition — Functionality — Value PM Dis in EEProm — No PME at all states — 00000b PM Ena & NoAux Pwr — PME at D0 and D3hot — 01001b PM Ena & Aux Pwr — PME at D0, D3hot and D3cold — 11001b Note: Aux Pwr is considered available if AUX_PWR pin is connected to 3.3V and *D3COLD_WAKEUP_ADVEN* EEPROM bit is set to 1b.
10	0b	RO	D2_Support The 82580 does not support D2 state.
9	0b	RO	D1_Support The 82580 does not support D1 state.
8:6	000b	RO	AUX Current – Required current defined in the Data Register.
5	1b	RO	DSI The 82580 requires its device driver to be executed following transition to the D0 uninitialized state.
4	0b	RO	Reserved
3	0b	RO	PME_Clock Disabled. Hardwired to 0b.
2:0	011b	RO	Version The 82580 complies with the PCI PM specification, revision 1.2.

9.5.1.4 Power Management Control / Status Register - PMCSR (0x44; R/W)

This register is used to control and monitor power management events in the 82580. Note that each device function has its own PMCSR.

Bits	Default	R/W	Description
15	0b (at power up)	R/W1CS	PME_Status This bit is set to 1b when the function detects a wake-up event independent of the state of the *PME_En* bit. Writing a 1b clears this bit.
14:13	01b	RO	Data_Scale This field indicates the scaling factor to be used when interpreting the value of the Data register. This field equals 01b (indicating 0.1 watt units) if power management is enabled in the *Power Management* (PM Ena) bit in *Initialization Control Word 1* (word 0x0A) EEPROM word and the *Data_Select* field is set to 0, 3, 4, 7, (or 8 for Function 0). Otherwise, this field equals 00b.
12:9	0000b	R/W	Data_Select This four-bit field is used to select which data is to be reported through the Data register and *Data_Scale* field. These bits are writable only when power management is enabled by setting the *Power Management* (PM Ena) bit in *Initialization Control Word 1* (word 0x0A) EEPROM word.
8	0b (at power up)	R/WS	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 affect and does not set the bit to 1b.
7:4	000000b	RO	Reserved
3	0b	RO	No_Soft_Reset This bit is always set to 0b to indicate that the 82580 performs an internal reset after a transition from D3hot to D0 via software control of the *PowerState* bits. Configuration context is lost when performing the soft reset. After transitioning from the D3hot to the D0 state, full re-initialization sequence is needed to return the 82580 to D0 Initialized.
2	0b	RO	Reserved for PCIe.
1:0	00b	R/W	Power State This field is used to set and report the power state of a function as follows: 00b = D0 01b = D1 (cycle ignored if written with this value) 10b = D2 (cycle ignored if written with this value) 11b = D3 (cycle ignored if power management is not enabled in the EEPROM)

9.5.1.5 Bridge Support Extensions - PMCSR_BSE (0x46; RO)

This register is not implemented in the 82580. Values are set to 0x00.

9.5.1.6 Data Register (0x47; RO)

This optional register is used to report power consumption and heat dissipation. Reported register is controlled by the *Data_Select* field in the PMCSR and the power scale is reported in the *Data_Scale* field in the PMCSR. The data of this field is loaded from the EEPROM if power management is enabled in the EEPROM or with a default value of 0x00. The values for the 82580 functions are read from EEPROM word 0x22.

Function	D0 (Consume/ Dissipate)	D3 (Consume/ Dissipate)	Common
PMCSR.Data Select	0x0 / 0x4	0x3 / 0x7	0x8
Function 0	EEPROM addr 0x22	EEPROM addr 0x22	EEPROM addr 0x22
Functions 1 - 3	EEPROM addr 0x22	EEPROM addr 0x22	0x00

For other *Data_Select* values, the Data register output is reserved (0x0).

9.5.2 MSI Configuration

This structure is required for PCIe devices.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x50	Message Control (0x0180)		Next Pointer (0x70)	Capability ID (0x05)
0x54	Message Address			
0x58	Message Upper Address			
0x5C	Reserved		Message Data	
0x60	Mask bits			
0x64	Pending bits			

9.5.2.1 Capability ID (0x50; RO)

This field equals 0x05 indicating the linked list item as being the MSI registers.

9.5.2.2 Next Pointer (0x51; RO)

This field provides an offset to the next capability item in the capability list. Its value of 0x70 points to the MSI-X capability structure.

9.5.2.3 Message Control (0x52; R/W)

The register fields are described in the following table. There is a dedicated register per PCI function to separately enable their MSI.

Bits	Default	R/W	Description
0	0b	R/W	MSI Enable If set to 1b, equals MSI. In this case, the 82580 generates an MSI for interrupt assertion instead of INTx signaling.
3:1	000b	RO	Multiple Message Capable The 82580 indicates a single requested message per each function.
6:4	000b	RO	Multiple Message Enable The 82580 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 82580 is capable of generating 64-bit message addresses.
8	1b[1]	RO	MSI per-vector masking. A value of 1b indicates that the 82580 is capable of per-vector masking. This field is loaded from the *MSI-X Configuration* (Offset 0x16) EEPROM word.
15:9	0b	RO	Reserved Write 0 ignore on read.

[1] Default value is read from the EEPROM

9.5.2.4 Message Address Low (0x54; R/W)

Written by the system to indicate the lower 32 bits of the address to use for the MSI memory write transaction. The lower two bits always return 0b regardless of the write operation.

9.5.2.5 Message Address High (0x58; R/W)

Written by the system to indicate the upper 32-bits of the address to use for the MSI memory write transaction.

9.5.2.6 Message Data (0x5C; R/W)

Written by the system to indicate the lower 16 bits of the data written in the MSI memory write Dword transaction. The upper 16 bits of the transaction are written as 0b.

9.5.2.7 Mask bits (0x60; R/W)

The Mask Bits and Pending Bits registers enable software to disable or defer message sending on a per-vector basis. As the 82580 supports only one message, only bit 0 of these register is implemented.

Bits	Default	R/W	Description
0	0b	R/W	MSI Vector 0 Mask If set, the 82580 is prohibited from sending MSI messages.
31:1	000b	RO	Reserved

9.5.2.8 Pending Bits (0x64; R/W)

Bits	Default	R/W	Description
0	0b	RO	If set, the 82580 has a pending MSI message.
31:1	000b	RO	Reserved

9.5.3 MSI-X Configuration

More than one MSI-X capability structure per function is prohibited, but a function is permitted to have both an MSI and an MSI-X capability structure.

In contrast to the MSI capability structure, which directly contains all of the control/status information for the function's vectors, the MSI-X capability structure instead points to an MSI-X table structure and a MSI-X Pending Bit Array (PBA) structure, each residing in memory space.

Each structure is mapped by a Base Address Register (BAR) belonging to the function, located beginning at 0x10 in configuration space. A BAR Indicator Register (BIR) indicates which BAR, and a Qword-aligned offset indicates where the structure begins relative to the base address associated with the BAR. The BAR is permitted to be either 32-bit or 64-bit, but must map to memory space. A function is permitted to map both structures with the same BAR, or to map each structure with a different BAR.

The MSI-X table structure, listed in Section 8.9, typically contains multiple entries, each consisting of several fields: message address, message upper address, message data, and vector control. Each entry is capable of specifying a unique vector.

The PBA structure, described in the same section, contains the function's pending bits, one per Table entry, organized as a packed array of bits within Qwords. Note that the last Qword might not be fully populated.

To request service using a given MSI-X table entry, a function performs a Dword memory write transaction using:

- The contents of the Message Data field entry for data.
- The contents of the Message Upper Address field for the upper 32 bits of the address.
- The contents of the Message Address field entry for the lower 32 bits of the address.

A memory read transaction from the address targeted by the MSI-X message produces undefined results.

The MSI-X table and MSI-X PBA are permitted to co-reside within a naturally aligned 4 KB address range, though they must not overlap with each other.

MSI-X table entries and Pending bits are each numbered 0 through N-1, where N-1 is indicated by the Table Size field in the MSI-X Message Control register. For a given arbitrary MSI-X table entry K, its starting address can be calculated with the formula:

Entry starting address = Table base + K*16

For the associated Pending bit K, its address for Qword access and bit number within that Qword can be calculated with the formulas:

Qword address = PBA base + (K div 64)*8

Qword bit# = K mod 64

Software that chooses to read Pending bit K with Dword accesses can use these formulas:

Dword address = PBA base + (K div 32)*4

Dword bit# = K mod 32

The 82580 also supports the table-less MSI-X mode, where a single interrupt vector is provided. The MSI-X table and MSI-X PBA are not used. Instead, the capability structure includes several additional fields (Message Address, Message Address Upper, and Message Data) for vector configuration. The 82580 embeds the number of the original MSI-X vectors (i.e. the vectors supported if the number of vectors was not limited to 1) in the LSB bits of the Message Data field.

Table 9-11. MSI-X capability Structure

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x70	Message Control (0x00090)		Next Pointer (0xA0)	Capability ID (0x11)
0x74	Table Offset			
0x78	PBA offset			

9.5.3.1 Capability ID (0x70; RO)

This field equals 0x11 indicating the linked list item as being the MSI-X registers.

9.5.3.2 Next Pointer (0x71; RO)

This field provides an offset to the next capability item in the capability list. Its value of 0xA0 points to the PCIe capability.

9.5.3.3 Message Control (0x72; R/W)

The register fields are described in the following table. There is a dedicated register per PCI function to separately configure their MSI-X functionality.

Bits	Default	R/W	Description
10:0	0x009[1]	RO	TS - Table Size System software reads this field to determine the MSI-X Table Size N, which is encoded as N-1. For example, a returned value of 0x00F indicates a table size of 16. The 82580 supports 10 MSI-X vectors. This field is loaded from the *MSI-X Configuration* (Offset 0x16) EEPROM word.
13:11	000b	RO	Reserved Always return 000b on read. Write operation has no effect.
14	0b	R/W	FM - Function Mask If set to 1b, all of the vectors associated with the function are masked, regardless of their per-vector *Mask* bit states. If set to 0b, each vector's *Mask* bit determines whether the vector is masked or not. Setting or clearing the *MSI-X Function Mask* bit has no effect on the state of the per-vector *Mask* bits.
15	0b	R/W	En - MSI-X Enable If set to 1b and the *MSI Enable* bit in the MSI Message Control (MMC) register is 0b, the function is permitted to use MSI-X to request service and is prohibited from using its INTx# pin. System configuration software sets this bit to enable MSI-X. A software device driver is prohibited from writing this bit to mask a function's service request. If set to 0b, the function is prohibited from using MSI-X to request service.

[1] Default value is read from the EEPROM

9.5.3.4 MSI-X Table Offset (0x74; R/W)

Bits	Default	Type	Description
31:3	0x000	RO	Table Offset Used as an offset from the address contained by one of the function's BARs to point to the base of the MSI-X table. The lower three table BIR bits are masked off (set to zero) by software to form a 32-bit Qword-aligned offset.
2:0	0x3	RO	Table BIR Indicates which one of a function's BARs, located beginning at 0x10 in configuration space, is used to map the function's MSI-X table into memory space. BIR values: 0...5 correspond to BARs 0x10…0x 24 respectively. A BIR value of 3 indicates that the table is mapped in BAR 3 (address 0x1C). When *BARCTRL.BAR32* equals 0b (64 bit MMIO mapping) the table BIR equals 0x4. When *BARCTRL.BAR32* equals 1b (32 bit MMIO mapping) the table BIR equals 0x3.

9.5.3.5 MSI-X Pending Bit Array - PBA Offset (0x78; R/W)

Bits	Default	Type	Description
31:3	0x400	RO	PBA Offset Used as an offset from the address contained by one of the function's BARs to point to the base of the MSI-X PBA. The lower three PBA BIR bits are masked off (set to zero) by software to form a 32-bit Qword-aligned offset.
2:0	0x3	RO	PBA BIR: Indicates which one of a function's Base Address registers, located beginning at 10h in Configuration Space, is used to map the function's MSI-X PBA into Memory Space. BIR values: 0...5 correspond to BARs 0x10…0x 24 respectively. A BIR value of 3 indicates that the table is mapped in BAR 3 (address 0x1C). When *BARCTRL.BAR32* equals 0b (64 bit MMIO mapping) the table BIR equals 0x4. When *BARCTRL.BAR32* equals 1b (32 bit MMIO mapping) the table BIR equals 0x3.

9.5.4 CSR Access Via Configuration Address Space

9.5.4.1 IOADDR Register (0x98; R/W)

This is a read/write register. Each function has its own *IOADDR* register. Functionality is the same in all functions. Register is cleared at Power-up (LAN_PWR_GOOD) or PCIe reset.

Note: When function is in D3 state Software should not attempt to access CSRs via the *IOADDR* and *IODATA* registers.

Bit(s)	R/W	Initial Value	Description
30:0	R/W[1]	0x0	Internal Register or Internal Memory location Address. 0x00000-0x1FFFF – Internal Registers and Memories 0x20000-0x7FFFFFFF – Undefined
31	R/W	0b	Configuration IO Access Enable. 0b - CSR configuration read or write disabled. 1b - CSR Configuration read or write enabled When bit is set accesses to the IODATA register actually generate transactions to the device. Otherwise, accesses to the IODATA register are don't-cares (write are discarded silently, reads return arbitrary results).

[1] In the event that the *CSR_conf_en* bit in the *PCIe Init Configuration 2* EEPROM word is cleared, accesses to the *IOADDR* register via configuration address space is ignored and has no effect on the register and the CSRs referenced by the *IOADDR* register.

9.5.4.2 IODATA Register (0x9C; R/W)

This is a read/write register. Each function has its own *IODATA* register. Functionality is the same in all functions. Register is cleared at Power-up (LAN_PWR_GOOD) or PCIe reset.

Bit(s)	R/W	Initial Value	Description
31:0	R/W[1]	0x0	Data field for reads or writes to the Internal register or internal memory location as identified by the current value in IOADDR. All 32 bits of this register are read/write-able.

[1] In the event that the *CSR_conf_en* bit in the *PCIe Init Configuration 2* EEPROM word is cleared, access to the *IODATA* register via configuration address space is ignored and has no effect on the register and the CSRs referenced by the *IOADDR* register.

9.5.5 Vital Product Data Registers

The 82580 supports access to a VPD structure stored in the EEPROM using the following set of registers.

Note: The VPD structure is available through all port functions. As the interface is common to all functions, accessing the VPD structure of one function while an access to the EEPROM is in process on another function can yield unexpected results.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0xE0	VPD address		Next Pointer (0x00)	Capability ID (0x03)
0xE4	VPD data			

9.5.5.1 Capability ID (0xE0; RO)

This field equals 0x3 indicating the linked list item as being the VPD registers.

9.5.5.2 Next Pointer (0xE1; RO)

Offset to the next capability item in the capability list. A 0x00 value indicates that it is the last item in the capability-linked list.

9.5.5.3 VPD Address (0xE2; RW)

Dword-aligned byte address of the VPD area in the EEPROM to be accessed. The register is read/write with the initial value at power-up indeterminate.

Bits	Default	R/W	Description
14:0	X	RW	Address Dword-aligned byte address of the VPD area in the EEPROM to be accessed. The register is read/write with the initial value at power-up indeterminate. The two LSBs are RO as zero. This is the address relative to the start of the VPD area. As the maximal size supported by the 82580 is 256 bytes, bits 14:8 should always be zero.
15	0b	RW	**F** A flag used to indicate when the transfer of data between the VPD Data register and the storage component completes. The Flag register is written when the VPD Address register is written. 0b = Read. Set by hardware when data is valid. 1b = Write. Cleared by hardware when data is written to the EEPROM. The VPD address and data should not be modified before the action completes.

9.5.5.4 VPD Data (0xE4; RW)

This register contains the VPD read/write data.

Bits	Default	R/W	Description
31:0	X	RW	VPD Data VPD data can be read or written through this register. The LSB of this register (at offset four in this capability structure) corresponds to the byte of VPD at the address specified by the VPD Address register. The data read from or written to this register uses the normal PCI byte transfer capabilities. Four bytes are always transferred between this register and the VPD storage component. Reading or writing data outside of the VPD space in the storage component is not allowed. In a write access, the data should be set before the address and the flag is set.

9.5.6 PCIe Configuration Registers

PCIe provides two mechanisms to support native features:

- PCIe defines a PCI capability pointer indicating support for PCIe.
- PCIe extends the configuration space beyond the 256 bytes available for PCI to 4096 bytes.

The 82580 implements the PCIe capability structure for endpoint devices as follows:

9.5.6.1 Capability ID (0xA0; RO)

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0xA0	PCI Express Capability Register (0x0002)		Next Pointer (0xE0)	Capability ID (0x10)
0xA4	Device Capability			
0xA8	Device Status		Device Control	
0xAC	Link Capability			
0xB0	Link Status		Link Control	
0xB4	Reserved			
0xB8	Reserved		Reserved	
0xBC	Reserved			
0xC0	Reserved		Reserved	
0xC4	Device Capabilities 2			
0xC8	Reserved		Device Control 2	
0xCC	Reserved			
0xD0	Link Status 2		Link Control 2	
0xD4	Reserved			
0xD8	Reserved		Reserved	

This field equals 0x10 indicating the linked list item as being the PCIe Capabilities registers.

9.5.6.2 Next Pointer (0xA1; RO)

Offset to the next capability item in the capability list. Its value of 0xE0 points to the VPD structure. If VPD is disabled, a value of 0x00 value indicates that it is the last item in the capability-linked list.

9.5.6.3 PCIe CAP (0xA2; RO)

The PCIe capabilities register identifies the 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 number. The 82580 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 a native PCI function with a value of 0000b.
8	0b	RO	Slot Implemented The 82580 does not implement slot options therefore this field is hardwired to 0b.
13:9	00000b	RO	Interrupt Message Number The 82580 does not implement multiple MSI interrupts per function, therefore this field is hardwired to 0x0.
15:14	00b	RO	Reserved

9.5.6.4 Device Capabilities (0xA4; RO)

This register identifies the PCIe device specific capabilities. It is a read only register with the same value for all functions.

Bits	R/W	Default	Description
2:0	RO	010b	Max Payload Size Supported This field indicates the maximum payload that the 82580 can support for TLPs. It is loaded from the EEPROM's *PCIe Init Configuration 3* word, 0x1A (with a default value of 512 bytes).
4:3	RO	00b	Phantom Function Supported Not supported by the 82580.
5	RO	0b	Extended Tag Field Supported Max supported size of the *Tag* field. The 82580 supported 5-bit *Tag* field for all functions.
8:6	RO	110b	Endpoint L0s Acceptable Latency This field indicates the acceptable latency that the 82580 can withstand due to the transition from the L0s state to the L0 state. All functions share the same value loaded from the EEPROM *PCIe Init Configuration 1* word, 0x18 (See Section).
11:9	RO	110b	Endpoint L1 Acceptable Latency This field indicates the acceptable latency that the 82580 can withstand due to the transition from the L1 state to the L0 state. All functions share the same value loaded from the EEPROM PCIe L1 Exit latencies word, 0x14 (See Section 6.2.11).
12	RO	0b	Attention Button Present Hardwired in the 82580 to 0b for all functions.
13	RO	0b	Attention Indicator Present Hardwired in the 82580 to 0b for all functions.
14	RO	0b	Power Indicator Present Hardwired in the 82580 to 0b for all functions.
15	RO	1b	Role-Based Error Reporting This bit, when set, indicates that the 82580 implements the functionality originally defined in the Error Reporting ECN for PCIe Base Specification 1.0a and later incorporated into PCIe Base Specification 1.1. Set to 1b in the 82580.
17:16	RO	000b	Reserved
25:18	RO	0x00	Slot Power Limit Value Hardwired in the 82580 to 0x00 for all functions, as the 82580 consumes less than the 25W allowed for it's form factor.

Bits	R/W	Default	Description
27:26	RO	00b	Slot Power Limit Scale Hardwired in the 82580 to 0 for all functions, as the 82580 consumes less than the 25W allowed for it's form factor.
28	RO	1b[1]	Function Level Reset (FLR) Capability A value of 1b indicates the function supports the optional FLR mechanism.
31:29	RO	000b	Reserved

[1] Loaded from EEPROM

9.5.6.5 Device Control (0xA8; RW)

This register controls the PCIe specific parameters. There is a dedicated register per each function.

Bits	R/W	Default	Description
0	RW	0b	Correctable Error Reporting Enable Enable report of correctable errors.
1	RW	0b	Non-Fatal Error Reporting Enable Enable report of non fatal errors.
2	RW	0b	Fatal Error Reporting Enable Enable report of fatal errors.
3	RW	0b	Unsupported Request Reporting Enable Enable report of unsupported requests error.
4	RW	1b	Enable Relaxed Ordering If this bit is set, the 82580 is permitted to set the *Relaxed Ordering* bit in the attribute field of write transactions that do not need strong ordering. For more details, refer to the description about the RO_DIS bit in the CTRL_EXT register bit in See Section 8.2.3 .
7:5	RW	000b (128 bytes)	Max Payload Size This field sets maximum TLP payload size for the 82580 functions. As a receiver, the 82580 must handle TLPs as large as the set value. As a transmitter, the 82580 must not generate TLPs exceeding the set value. The max payload size supported in the 82580 Device capabilities register indicates permissible values that can be programmed.
8	R0	0b	Extended Tag field Enable Not implemented in the 82580.
9	R0	0b	Phantom Functions Enable Not implemented in the 82580.
10	RWS	0b	Auxiliary Power PM Enable When set, enables the 82580 to draw AUX power independent of PME AUX power. The 82580 is a multi function device, therefore it is allowed to draw AUX power if at least one of the functions has this bit set.

Bits	R/W	Default	Description
11	RW	1b	Enable No Snoop Snoop is gated by *NONSNOOP* bits in the GCR register in the CSR space.
14:12	RW	010b / 000b	Max Read Request Size - this field sets maximum read request size for the Device as a requester. 000b = 128 bytes (the default value for non LAN functions). 001b = 256 bytes. 010b = 512 bytes. (the default value for the LAN devices). 011b = 1 KB. 100b = 2 KB. 101b = Reserved. 110b = Reserved. 111b = Reserved.
15	RW	0b	Initiate Function Level Reset A write of 1b initiates an FLR to the function. The value read by software from this bit is always 0b.

9.5.6.6 Device Status (0xAA; RW1C)

This register provides information about PCIe device's specific parameters. There is a dedicated register per each function.

Bits	R/W	Default	Description
0	RW1C	0b	Correctable Error 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 82580 received an unsupported request. This field is identical in all functions. The 82580 cannot distinguish which function caused an 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 identical for all functions. Reset on LAN_PWR_GOOD and GIO Power Good only.
5	RO	0b	Transactions Pending Indicates whether the 82580 has any transaction pending.
15:6	RO	0x00	Reserved

9.5.6.7 Link CAP (0xAC; RO)

This register identifies PCIe link specific capabilities. This is a read only register identical to all functions.

Bits	Rd/Wr	Default	Description
3:0	RO	0010b	Max Link Speed This field indicates the supported Link speed(s) of the associated link port. Defined encodings are: 0001b = 2.5 Gb/s Link speed supported. 0010b = 5 Gb/s and 2.5 Gb/s Link speeds supported. Value of this field is determined by the *Disable PCIe Gen 2* bit in the *PCIe PHY Auto Configuration* EEPROM section.
9:4	RO	0x4	Max Link Width Indicates the maximum link width. The 82580 can support by 1-, by 2- and by 4-link width. The field is loaded from the EEPROM (See Section 6.5.4.3), with a default value of four lanes. Relevant encoding: 000000b = Reserved. 000001b = x1. 000010b = x2. 000100b = x4.
11:10	RO	11b	Active State Power Management (ASPM) Support – This field indicates the level of ASPM supported on the 82580 PCI Express Link. Defined encodings are: 00b = Reserved. 01b = L0s Entry Supported. 10b = Reserved. 11b = L0s and L1 Supported.
14:12	RO	110b (2 µs – 4µs)	L0s Exit Latency Indicates the exit latency from L0s to L0 state. 000b = Less than 64ns. 001b = 64ns – 128ns. 010b = 128ns – 256ns. 011b = 256ns - 512ns. 100b = 512ns - 1 µs. 101b = 1 µs – 2 µs. 110b = 2 µs – 4 µs. 111b = Reserved. Depending on usage of common clock or separate clock the value of this field is loaded from PCIe Init Config 1 EEPROM word, 0x18 (See Section).

Bits	Rd/Wr	Default	Description
17:15	RO	110b (32-64 µs)	L1 Exit Latency Indicates the exit latency from L1 to L0 state. 000b = Less than 1 µs. 001b = 1 µs - 2 µs. 010b = 2 µs - 4 µs. 011b = 4 µs - 8 µs. 100b = 8 µs - 16 µs. 101b = 16 µs - 32 µs. 110b = 32 µs - 64 µs. 111b = L1 transition not supported. Depending on usage of common clock or separate clock the value of this field is loaded from *PCIe L1 Exit latencies* EEPROM word, 0x14 (See Section 6.2.11).
18	RO	0b	Clock Power Management Status Not supported in the 82580. RO as zero.
19	RO	0b	Surprise Down Error Reporting Capable Status Not supported in the 82580. RO as zero
20	RO	0b	Data Link Layer Link Active Reporting Capable Status Not supported in the 82580. RO as zero.
21	RO	0b	Link Bandwidth Notification Capability Status Not supported in the 82580. RO as zero.
23:22	RO	00b	Reserved
31:24	HwInit	0x0	Port Number The PCIe port number for the given PCIe link. Field is set in the link training phase.

9.5.6.8 Link Control (0xB0; RW)

This register controls PCIe link specific parameters. There is a dedicated register per each function.

Bits	R/W	Default	Description
1:0	RW	00b	Active State Power Management (ASPM) Control – This field controls the level of Active State Power Management (ASPM) supported on the 82580 PCI Express Link. Link PM functionality is determined by the lowest common denominator of all functions. Defined encodings are: 00b = PM disabled. 01b = L0s entry Enabled. 10b = L1 Entry Enabled. 11b = L0s and L1 Enabled. Note: "L0s Entry Enabled" indicates the Transmitter entering L0s is supported. The Receiver must be capable of entering L0s even when the field is disabled (00b).
2	RO	0b	Reserved
3	RW	0b	Read Completion Boundary Read Completion Boundary (RCB) – Optionally Set by configuration software to indicate the RCB value of the Root Port Upstream from the Endpoint or Bridge. Defined encodings are: 0b = 64 byte 1b = 128 byte Configuration software must only Set this bit if the Root Port Upstream from the Endpoint or Bridge reports an RCB value of 128 bytes (a value of 1b in the Read Completion Boundary bit).
4	RO	0b	Link Disable Not applicable for endpoint devices; hardwired to 0b.
5	RO	0b	Retrain Clock Not applicable for endpoint devices; hardwired to 0b.
6	RW	0b	Common Clock Configuration When this bit is set, it indicates that the 82580 and the component at the other end of the link are operating with a common reference clock. A value of 0b indicates that both operate with an asynchronous clock. This parameter affects the L0s exit latencies.
7	RW	0b	Extended Synch When this bit is set, it forces an extended Tx of a FTS ordered set in FTS and an extra TS1 at exit from L0s prior to enter L0.
8	RO	0b	Enable Clock Power Management Not supported in the 82580. RO as zero.
9	RO	0b	Hardware Autonomous Width Disable Not supported in the 82580. RO as zero.
10	RO	0b	Link Bandwidth Management Interrupt Enable Not supported in the 82580. RO as zero.
11	RO	0b	Link Autonomous Bandwidth Interrupt Enable Not supported in the 82580. RO as zero.
15:12	RO	0000b	Reserved

9.5.6.9 Link Status (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	Link Speed This field indicates the negotiated link speed of the given PCIe link. Defined encodings are: 0001b = 2.5 Gb/s PCIe link. 0010b = 5 Gb/s PCIe link. All other encodings are reserved.
9:4	RO	000001b	Negotiated Link Width Indicates the negotiated width of the link. Relevant encoding for the 82580 are: 000001b = x1 000010b = x2 000100b = x4
10	RO	0b	Reserved (was: Link Training Error)
11	RO	0b	Link Training Indicates that link training is in progress.
12	HwInit	1b	Slot Clock Configuration When set, indicates that the 82580 uses the physical reference clock that the platform provides on the connector. This bit must be cleared if the 82580 uses an independent clock. The Slot Clock Configuration bit is loaded from the *Slot_Clock_Cfg* bit in *PCIe Init Configuration 3* Word (Word 0x1A) EEPROM word.
13	RO	0b	Data Link Layer Link Active Not supported in the 82580. RO as zero.
14	RO	0b	Link Bandwidth Management Status Not supported in the 82580. RO as zero.
15	RO	0b	Reserved

9.5.6.10 Reserved (0xB4-0xC0; RO)

Unimplemented reserved registers not relevant to PCIe endpoint.

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

9.5.6.11 Device Capabilities 2 (0xC4; RO)

This register identifies PCIe device specific capabilities. It is a read only register with the same value for all functions.

Bit Location	R/W	Default	Description
3:0	RO	1111b	Completion Timeout Ranges Supported This field indicates 82580 support for the optional completion timeout programmability mechanism. This mechanism enables system software to modify the completion timeout value. Description of the mechanism can be found in Section 3.1.3.2. Four time value ranges are defined: • Range A = 50 µs to 10 ms • Range B = 10 ms to 250 ms • Range C = 250 ms to 4 s • Range D = 4 s to 64 s Bits are set according to the following table to show the timeout value ranges that are supported. • 0000b = Completion timeout programming not supported. the 82580 must implement a timeout value in the range 50 µs to 50 ms. • 0001b = Range A. • 0010b = Range B. • 0011b = Ranges A & B. • 0110b = Ranges B & C. • 0111b = Ranges A, B & C. • 1110b = Ranges B, C & D. • 1111b = Ranges A, B, C & D. • All other values are reserved. It is strongly recommended that the completion timeout mechanism not expire in less than 10 ms
4	RO	1b	Completion Timeout Disable Supported A value of 1b indicates support for the completion timeout disable mechanism.
5	RO	0b	ARI Forwarding Supported Applicable only to switch downstream ports and root ports; must be set to 0b for other function types.
10:6	RO	0x0	Reserved
11	RO	1b[1]	LTR Mechanism Supported – A value of 1b indicates support for the optional Latency Tolerance Requirement Reporting (LTR) mechanism capability. Note: Value loaded from LTR_EN bit in Initialization Control Word 1 EEPROM word.
13:12	RO	00b	TPH Completer supported - the 82580 does not use the hints as a completer
31:14	RO	0x0	Reserved

[1] Value loaded from EEPROM word.

9.5.6.12 Device Control 2 (0xC8; RW)

This register controls PCIe specific parameters. There is a dedicated register per each function.

Bit location	R/W	Default	Description
3:0	RW	0000b	Completion Timeout Value[1] In devices that support completion timeout programmability, this field enables system software to modify the completion timeout value. Encoding: • 0000b = Allowable default range: 50 µs to 50 ms. It is strongly recommended that the completion timeout mechanism not expire in less than 10 ms. Actual completion timeout range supported in the 82580 is 16 ms to 32 ms. Values available if Range A (50 µs to 10 ms) programmability range is supported: • 0001b = Allowable range is 50 µs to 100 µs. Actual completion timeout range supported in the 82580 is 50 µs to 100 µs. • 0010b = Allowable range is 1 ms to 10 ms. Actual completion timeout range supported in the 82580 is 1 ms to 2 ms. Values available if Range B (10 ms to 250 ms) programmability range is supported: • 0101b = Allowable range is 16 ms to 55 ms. Actual completion timeout range supported in the 82580 is 16 ms to 32 ms. • 0110b = Allowable range is 65 ms to 210 ms. Actual completion timeout range supported in the 82580 is 65 ms to 130 ms. Values available if Range C (250 ms to 4 s) programmability range is supported: • 1001b = Allowable range is 260 ms to 900 ms. Actual completion timeout range supported in the 82580 is 260 ms to 520 ms. • 1010b = Allowable range is 1 s to 3.5 s. Actual completion timeout range supported in the 82580 is 1 s to 2 s. Values available if the Range D (4 s to 64 s) programmability range is supported: • 1101b = Allowable range is 4 s to 13 s. Actual completion timeout range supported in the 82580 is 4 s to 8 s. • 1110b = Allowable range is 17 s to 64 s. Actual completion timeout range supported in the 82580 is 17 s to 34 s. Values not defined are reserved. Software is permitted to change the value in this field at any time. For requests already pending when the completion timeout value is changed, hardware is permitted to use either the new or the old value for the outstanding requests and is permitted to base the start time for each request either when this value was changed or when each request was issued. The default value for this field is 0000b.
4	RW	0b	Completion Timeout Disable When set to 1b, this bit disables the completion timeout mechanism. Software is permitted to set or clear this bit at any time. When set, the completion timeout detection mechanism is disabled. If there are outstanding requests when the bit is cleared, it is permitted but not required for hardware to apply the completion timeout mechanism to the outstanding requests. If this is done, it is permitted to base the start time for each request on either the time this bit was cleared or the time each request was issued. The default value for this bit is 0b.
5	RO	0b	Alternative RID Interpretation (ARI) Forwarding Enable Applicable only to switch devices.

Bit location	R/W	Default	Description
9:6	RO	0b	Reserved.
10	RW	0b	LTR Mechanism Enable – When Set to 1b, this bit enables the Latency Tolerance Requirement Reporting (LTR) mechanism. Notes: 1. Since the 82580 is a Multi-Function device, the bit in Function 0 is of type RW, and only Function 0 controls the component's Link behavior. In all other Functions this bit is of type RsvdP. 2. If Value of LTR_EN bit in Initialization Control Word 1 EEPROM word is 0, then bit is RO with a value of 0b.
15:11	RO	0x0	Reserved.

[1] The completion timeout value must be programmed correctly in PCIe configuration space (in Device Control 2 Register); the value must be set above the expected maximum latency for completions in the system in which the 82580 is installed. This will ensure that the 82580 receives the completions for the requests it sends out, avoiding a completion timeout scenario. It is expected that the system BIOS will set this value appropriately for the system.

9.5.6.13 Link Control 2 (0xD0; RW)

Bits	R/W	Default	Description
3:0	RWS	0010b	Target Link Speed. This field is used to set the target compliance mode speed when software is using the *Enter Compliance* bit to force a link into compliance mode. Defined encodings are: 0001b = 2.5 Gb/s Target Link Speed. 0010b = 5 Gb/s Target Link Speed. All other encodings are reserved. If a value is written to this field that does not correspond to a speed included in the *Max Link Speed* field, the result is undefined. The default value of this field is the highest link speed supported by the 82580 (as reported in the *Max Link Speed* field of the Link Capabilities register).
4	RWS	0b	Enter Compliance. Software is permitted to force a link to enter compliance mode at the speed indicated in the *Target Link Speed* field by setting this bit to 1b in both components on a link and then initiating a hot reset on the link. The default value of this field following a fundamental reset is 0b.
5	RO	0b	Hardware Autonomous Speed Disable. When set to 1b, this bit disables hardware from changing the link speed for reasons other than attempting to correct unreliable link operation by reducing link speed. Hard wired to 0b.
6	RO	0b	Selectable De-emphasis This bit is not applicable and reserved for Endpoints.
9:7	RWS	000b	Transmit Margin – This field controls the value of the non de emphasized voltage level at the Transmitter pins. Encodings: 000b - Normal operating range 001b - 800-1200 mV for full swing and 400-700 mV for half-swing 010b - (n-1) - Values must be monotonic with a non-zero slope. The value of n must be greater than 3 and less than 7. At least two of these must be below the normal operating range of n: 200-400 mV for full-swing and 100-200 mV for half-swing n - 111b reserved

Bits	R/W	Default	Description
10	RWS	0b	Enter Modified Compliance – When this bit is set to 1b, the device transmits modified compliance pattern if the LTSSM enters Polling.Compliance state.
11	RWS	0b	Compliance SOS – When set to 1b, the LTSSM is required to send SOS periodically in between the (modified) compliance patterns.
12	RWS	0b	Compliance De-emphasis This bit sets the de-emphasis level in Polling.Compliance state if the entry occurred due to the Enter Compliance bit being 1b. Encodings: 1b -3.5 dB 0b -6 dB When the Link is operating at 2.5 GT/s, the setting of this bit has no effect. Components that support only 2.5 GT/s speed are permitted to hardwire this bit to 0b. ***Note:*** For the 82580, which is a Multi-Function device, the field in Function 0 is of type RWS; only Function 0 controls the component's Link behavior. In all other Functions of the device, this field is of type RsvdP.
15:11	RO	0x0	reserved

9.5.6.14 Link Status 2 (0xD2; RW)

Bits	R/W	Default	Description
0	RO	0b	Current De-emphasis Level – When the Link is operating at 5 GT/s speed, this bit reflects the level of de-emphasis. it is undefined when the Link is operating at 2.5 GT/s speed Encodings: 1b -3.5 dB 0b -6 dB
15:1	RO	0x0	Reserved

9.6 PCIe Extended Configuration Space

PCIe extended configuration space is located in a flat memory-mapped address space. PCIe extends the configuration space beyond the 256 bytes available for PCI to 4096 bytes. The 82580 decodes an additional 4-bits (bits 27:24) to provide the additional configuration space as shown in Table 9-12. PCIe reserves the remaining 4 bits (bits 31:28) for future expansion of the configuration space beyond 4096 bytes.

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

Table 9-12. PCIe Extended Configuration Space

31 28	27 20	19 15	14 12	11 2	1 0
0000b	Bus #	Device #	Fun #	Register Address (offset)	00b

PCIe extended configuration space is allocated using a linked list of optional or required PCIe extended capabilities following a format resembling PCI capability structures. The first PCIe extended capability is located at offset 0x100 in the device configuration space. The first Dword of the capability structure identifies the capability/version and points to the next capability.

The 82580 supports the following PCIe extended capabilities.

Table 9-13. PCIe Extended Capability Structure

Capability	Offset	Next Header
Advanced Error Reporting Capability	0x100	0x140/0x1A0[1]
Serial Number[2]	0x140	0x1A0
TLP processing hints	0x1A0	0x1C0
Latency Tolerance Requirement Reporting Capability	0x1C0	0x000

1. Depends on EEPROM settings enabling the Serial Number in *PCIe Init Configuration 2* EEPROM Word.
2.

9.6.1 Advanced Error Reporting (AER) Capability

The PCIe AER capability is an optional extended capability to support advanced error reporting. The following table lists the PCIe AER extended capability structure for PCIe devices.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x100	Next Capability Ptr. (0x140/0x1A0[1])	Version (0x1)	AER Capability ID (0x0001)	
0x104	Uncorrectable Error Status			
0x108	Uncorrectable Error Mask			
0x10C	Uncorrectable Error Severity			
0x110	Correctable Error Status			
0x114	Correctable Error Mask			
0x118	Advanced Error Capabilities and Control Register			
0x11C... 0x128	Header Log			

1. Depends on EEPROM settings enabling the Serial Number structure in *PCIe Init Configuration 2* EEPROM Word.

9.6.1.1 PCIe CAP ID (0x100; RO)

Bit Location	Attribute	Default Value	Description
15:0	RO	0x0001	Extended Capability ID PCIe extended capability ID indicating AER capability.
19:16	RO	0x1	Version Number PCIe AER extended capability version number.
31:20	RO	0x1A0/ 0x140	Next Capability Pointer Next PCIe extended capability pointer. A value of 0x140 points to the serial ID capability. When serial ID is disabled in the EEPROM the next pointer is 0x1A0.

9.6.1.2 Uncorrectable Error Status (0x104; R/W1CS)

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

Bit Location	Attribute	Default Value	Description
3:0	RO	0x0	Reserved
4	R/W1CS	0b	Data Link Protocol Error Status
5	RO	0b	Surprise Down Error Status (Optional) Not supported in the 82580.
11:6	RO	0x0	Reserved
12	R/W1CS	0b	Poisoned TLP Status
13	R/W1CS	0b	Flow Control Protocol Error Status
14	R/W1CS	0b	Completion Timeout Status
15	R/W1CS	0b	Completer Abort Status
16	R/W1CS	0b	Unexpected Completion Status
17	R/W1CS	0b	Receiver Overflow Status
18	R/W1CS	0b	Malformed TLP Status
19	R/W1CS	0b	ECRC Error Status
20	R/W1CS	0b	Unsupported Request Error Status
21	RO	0b	ACS Violation Status Not supported in the 82580.
22	RO	0b	Uncorrectable Internal Error Status (Optional) Not supported in the 82580.
23	RO	0b	MC Blocked TLP Status (Optional) Not supported in the 82580.
24	RO	0b	AtomicOps Egress Blocked Status (Optional) Not supported in the 82580.
25	RO	0b	TLP Prefix Blocked Error Status (Optional) Not supported in the 82580.
31:26	RO	0x0	Reserved

9.6.1.3 Uncorrectable Error Mask (0x108; RWS)

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

Bit Location	Attribute	Default Value	Description
3:0	RO	0x0	Reserved
4	RWS	0b	Data Link Protocol Error Mask
5	RO	0b	Surprise Down Error Mask (Optional) Not supported in the 82580.
11:6	RO	0x0	Reserved

Bit Location	Attribute	Default Value	Description
12	RWS	0b	Poisoned TLP Mask
13	RWS	0b	Flow Control Protocol Error Mask
14	RWS	0b	Completion Timeout Mask
15	RWS	0b	Completer Abort Mask
16	RWS	0b	Unexpected Completion Mask
17	RWS	0b	Receiver Overflow Mask
18	RWS	0b	Malformed TLP Mask
19	RWS	0b	ECRC Error Mask
20	RWS	0b	Unsupported Request Error Mask
21	RO	0b	ACS Violation Mask Not supported in the 82580.
22	RO	0b	Uncorrectable Internal Error Mask (Optional) Not supported in the 82580.
23	RO	0b	MC Blocked TLP Mask (Optional) Not supported in the 82580.
24	RO	0b	AtomicOps Egress Blocked Mask (Optional) Not supported in the 82580.
25	RO	0b	TLP Prefix Blocked Error Mask (Optional) Not supported in the 82580.
31:26	RO	0x0	Reserved

9.6.1.4 Uncorrectable Error Severity (0x10C; RWS)

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

Bit Location	Attribute	Default Value	Description
3:0	RO	0001b	Reserved
4	RWS	1b	Data Link Protocol Error Severity
5	RO	1b	Surprise Down Error Severity (Optional) Not supported in the 82580.
11:6	RO	0x0	Reserved
12	RWS	0b	Poisoned TLP Severity
13	RWS	1b	Flow Control Protocol Error Severity
14	RWS	0b	Completion Timeout Severity
15	RWS	0b	Completer Abort Severity
16	RWS	0b	Unexpected Completion Severity
17	RWS	1b	Receiver Overflow Severity
18	RWS	1b	Malformed TLP Severity
19	RWS	0b	ECRC Error Severity
20	RWS	0b	Unsupported Request Error Severity
21	RO	0b	ACS Violation Severity

Bit Location	Attribute	Default Value	Description
22	RO	1b	Uncorrectable Internal Error Severity (Optional) Not supported in the 82580.
23	RO	0b	MC Blocked TLP Severity (Optional) Not supported in the 82580.
24	RO	0b	AtomicOps Egress Blocked Severity (Optional) Not supported in the 82580.
25	RO	0b	TLP Prefix Blocked Error Severity (Optional) Not supported in the 82580.
31:26	RO	0x0	Reserved

9.6.1.5 Correctable Error Status (0x110; R/W1CS)

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

Bit Location	Attribute	Default Value	Description
0	R/W1CS	0b	Receiver Error Status
5:1	RO	0x0	Reserved
6	R/W1CS	0b	Bad TLP Status
7	R/W1CS	0b	Bad DLLP Status
8	R/W1CS	0b	REPLAY_NUM Rollover Status
11:9	RO	000	Reserved
12	R/W1CS	0b	Replay Timer Timeout Status
13	R/W1CS	0b	Advisory Non-Fatal Error Status
14	RO	0b	Corrected Internal Error Status (Optional) Not supported in the 82580.
15	RO	0b	Header Log Overflow Status (Optional) Not supported in the 82580.
31:16	RO	0x0	Reserved

9.6.1.6 Correctable Error Mask (0x114; RWS)

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

Bit Location	Attribute	Default Value	Description
0	RWS	0b	Receiver Error Mask
5:1	RO	0x0	Reserved
6	RWS	0b	Bad TLP Mask
7	RWS	0b	Bad DLLP Mask

Bit Location	Attribute	Default Value	Description
8	RWS	0b	REPLAY_NUM Rollover Mask
11:9	RO	000b	Reserved
12	RWS	0b	Replay Timer Timeout Mask
13	RWS	1b	Advisory Non-Fatal Error Mask
14	RO	0b	Corrected Internal Error Mask (Optional) Not supported in the 82580.
15	RO	0b	Header Log Overflow Mask (Optional) Not supported in the 82580.
31:16	RO	0x0	Reserved

9.6.1.7 Advanced Error Capabilities and Control Register (0x118; RWS)

Bit Location	Attribute	Default Value	Description
4:0	ROS	0x0	First Error Pointer The First Error Pointer is a field that identifies the bit position of the first error reported in the Uncorrectable Error Status register. Field holds a vector pointing to the first recorded error in the Uncorrectable Error Status register.
5	RO	1b	ECRC Generation Capable This bit indicates that the 82580 is capable of generating ECRC. This bit is loaded from EEPROM PCIe Control 2 word (Word 0x28).
6	RWS	0b	ECRC Generation Enable When set, enables ECRC generation.
7	RO	1b	ECRC Check Capable If Set, this bit indicates that the Function is capable of checking ECRC. This bit is loaded from EEPROM PCIe Control 2 word (Word 0x28).
8	RWS	0b	ECRC Check Enable When set, enables ECRC checking.
9	RO	0b	Multiple Header Recording Capable – If Set, this bit indicates that the Function is capable of recording more than one error header.
10	RO	0b	This bit enables the Function to record more than one error header. Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b.
11	RO	0b	TLP Prefix Log Present – If Set and the First Error Pointer is valid, indicates that the TLP Prefix Log register contains valid information. If Clear or if First Error Pointer is invalid, the TLP Prefix Log register is undefined. Default value of this bit is 0b. This bit is RsvdP if the End-End TLP Prefix Supported bit is Clear.
31:12	RO	0x0	Reserved

9.6.1.8 Header Log (0x11C:0x128; ROS)

The Header Log register captures the header for the transaction that generated an error. This register is 16 bytes in length.

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

9.6.2 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.

9.6.2.1 Device Serial Number Enhanced Capability Header (0x140;

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x140	Next Capability Ptr. (0x1A0)	Version (0x1)	Serial ID Capability ID (0x0003)	
0x144	Serial Number Register (Lower Dword)			
0x148	Serial Number Register (Upper Dword)			

RO)

The following table lists the allocation of register fields in the device serial number enhanced capability header. It also lists the respective bit definitions. The extended capability ID for the device serial number capability is 0x0003.

Bit(s) Location	Default value	Attributes	Description
15:0	0x0003	RO	PCIe Extended Capability ID This field is a PCI-SIG defined ID number that indicates the nature and format of the extended capability. The extended capability ID for the device serial number capability is 0x0003.
19:16	0x1	RO	Capability Version This field is a PCI-SIG defined version number that indicates the version of the current capability structure.
31:20	0x1A0	RO	Next Capability Offset This field contains the offset to the next PCIe capability structure or 0x000 if no other items exist in the linked list of capabilities.

9.6.2.2 Serial Number Register (0x144:0x148; RO)

The Serial Number register is a 64-bit field that contains the IEEE defined 64-bit extended unique identifier (EUI-64™). Table 9-14 lists the allocation of register fields in the Serial Number register. Table 9-14 also lists the respective bit definitions.

Table 9-14. Serial Number Register

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

Serial number definition in the 82580:

Table 9-15. SN Definition

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

Serial number uses the MAC address according to the following definition:

Table 9-16. SN and MAC Address

Field	**Extension identifier**					**Company ID**		
Order	Addr+0	Addr+1	Addr+2	Addr+3	Addr+4	Addr+5	Addr+6	Addr+7

Most significant byte — Least significant byte

Most significant bit — Least significant bit

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

Table 9-17. SN Constructed from 48-bit MAC Address

Field	**Extension identifier**			**MAC Label**		**Company ID**		
Order	Addr+0	Addr+1	Addr+2	Addr+3	Addr+4	Addr+5	Addr+6	Addr+7

Most significant bytes — Least significant byte

Most significant bit — Least significant bit

The MAC label in this case is 0xFFFF.

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

Table 9-18. Example 64-bit SN

Field	**Extension identifier**	**MAC Label**	**Company ID**

Table 9-18. Example 64-bit SN (Continued)

Order	Addr+0	Addr+1	Addr+2	Addr+3	Addr+4	Addr+5	Addr+6	Addr+7
	67	45	23	FF	FF	C9	A0	00

Most significant byte Least significant byte

Most significant bit Least significant bit

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

The translation from EEPROM words 0 to 2 to the serial number is as follows:

- Serial number ADDR+0 = EEPROM byte 5
- Serial number ADDR+1 = EEPROM byte 4
- Serial number ADDR+2 = EEPROM byte 3
- Serial number ADDR+3,4 = 0xFF 0xFF
- Serial number ADDR+5 = EEPROM byte 2
- Serial number ADDR+6 = EEPROM byte 1
- Serial number ADDR +7 = EEPROM byte 0

The official document defining EUI-64 is: http://standards.ieee.org/regauth/oui/tutorials/EUI64.html

9.6.3 TLP Processing Hint Requester (TPH) Capability

The PCIe TPH Requester capability is an optional extended capability to support TLP Processing Hints. The following table lists the PCIe TPH extended capability structure for PCIe devices.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x1A0	Next Capability Ptr. (0x1C0/0x000[1])	Version (0x1)	TPH Capability ID (0x17)	
0x1A4	TPH Requester Capability Register			
0x1A8	TPH Requester Control Register			
0x1AC-0x1B8	TPH ST Table			

[1] Depends on EEPROM settings of the *LTR_EN* bit in *Initialization Control Word 1* EEPROM word, that controls enabling of the LTR structures.

9.6.3.1 TPH CAP ID (0x1A0; RO)

Bit Location	Attribute	Default Value	Description
15:0	RO	0x17	Extended Capability ID PCIe extended capability ID indicating TPH capability.
19:16	RO	0x1	Version Number PCIe TPH extended capability version number.
31:20	RO	0x1C0/ 0x000[1]	Next Capability Pointer This field contains the offset to the next PCIe capability structure or 0x000 if no other items exist in the linked list of capabilities.

1. Depends on EEPROM settings of the *LTR_EN* bit in *Initialization Control Word 1* EEPROM word, that controls enabling of the LTR structures.

9.6.3.2 TPH Requester Capabilities (0x1A4; RO)

Bit Location	Attribute	Default Value	Description
0	RO	1	No ST Mode Supported: When set indicates the Function is capable of generating Requests without using ST.
1	RO	0	Interrupt Vector Mode Supported: Cleared to indicates that the 82580 does not support Interrupt Vector Mode of operation
2	RO	1	Device Specific Mode: Set to indicate that the 82580 supports Device Specific Mode of operation
7:3	RO	0	Reserved
8	RO	0	Extended TPH Requester Supported – Cleared to indicate that the function is not capable of generating requests with Extended TPH TLP Prefix
10:9	RO	01b	ST Table Location – Value indicates if and where the ST Table is located. Defined Encodings are: 00b – ST Table is not present. 01b – ST Table is located in the TPH Requester Capability structure. 10b – ST Table is located in the MSI-X Table structure. 11b – Reserved Default value of 01b indicates that function supports ST table that’s located in the TPH Requester Capability structure.
15:11	RO	0x0	Reserved
26:16	RO	0x7	ST_Table Size – System software reads this field to determine the ST_Table_Size N, which is encoded as N-1. The 82580 supports a table with 8 entries.
31:27	RO	0x0	Reserved

9.6.3.3 TPH Requester Control (0x1A8; R/W)

Bit Location	Attribute	Default Value	Description
2:0	RW	0x0	ST Mode Select – Indicates the ST mode of operation selected. The ST mode encodings are as defined below 000b – No Table Mode 001b – Interrupt Vector Mode (not supported by the 82580) 010b – Device Specific Mode 0thers – reserved for future use The default value of 000 indicates No Table mode of operation.
7:3	RO	0x0	Reserved
9:8	RW	0x0	TPH Requester Enable: Defined Encodings are 00b – The 82580 is not permitted to issue transactions with TPH or Extended TPH as Requester 01b – The 82580 is permitted to issue transactions with TPH as Requester and is not permitted to issue transactions with Extended TPH as Requester 10b – Reserved 11b – The 82580 is permitted to issue transactions with TPH and Extended TPH as Requester (The 82580 does not issue transactions with Extended TPH).
31:10	RO	0x0	Reserved

9.6.3.4 TPH Steering table (0x1AC - 0x1B8; R/W)

Bit Location	Attribute	Default Value	Description
7:0	RW	0x0	Steering Table Lower Entry 2*n (n = 0...3). A value of zero indicates the tag is not valid
15:8	RO	0x0	Steering Table Upper Entry 2*n (n = 0...3) - RO zero in the 82580, as extended tags are not supported.
23:16	RW	0x0	Steering Table Entry 2*n + 1 (n = 0...3) - A value of zero indicates the tag is not valid
31:24	RO	0x0	Steering Table Upper Entry 2*n + 1 (n = 0...3) - RO zero in the 82580, as extended tags are not supported.

9.6.3.5 Latency Tolerance Requirement Reporting (LTR) Capability

The PCI Express Latency Tolerance Requirement Reporting Capability is an optional Extended Capability that allows software to provide platform latency information to devices with upstream ports (Endpoints and Switches). This capability structure is required if the device supports Latency Tolerance Requirement Reporting (LTR).

Note: The LTR Capability structure is implemented only in Function 0, and controls the component’s Link behavior on behalf of all the Functions of the device

The following table lists the PCIe LTR extended capability structure for PCIe devices.

Byte Offset	Byte 3	Byte 2	Byte 1	Byte 0
0x1C0	Next Capability Ptr. (0x000)	Version (0x1)	LTR Capability ID (0x18)	
0x1C4	Maximum Non-Snooped Platform Latency Tolerance Register		Maximum Snooped Platform Latency Tolerance Register	

9.6.3.6 LTR CAP ID (0x1C0; RO)

Bit Location	Attribute	Default Value	Description
15:0	RO	0x18	LTR Capability ID PCIe extended capability ID indicating LTR capability.
19:16	RO	0x1	Version Number PCIe LTR extended capability version number.
31:20	RO	0x000	Next Capability Pointer This is the last capability, so the next pointer is 0x000.

9.6.3.7 LTR Capabilities (0x1C4; RW)

Bit Location	Attribute	Default Value	Description
9:0	RW	0x0	Maximum Snoop Latency Value Along with the Max Snoop Latency Scale field, this register specifies the maximum nosnoop latency that a device is permitted to request. Software should set this to the platform's maximum supported latency or less. Field is also an indicator of the platforms maximum latency, should an endpoint send up LTR Latency Values with the Requirement bit not set.
12:10	RW	0x0	Max Snoop Latency Scale This field provides a scale for the value contained within the Maximum Snoop Latency Value field. Encoding: 000 – Value times 1ns 001 – Value times 32ns 010 – Value times 1,024ns 011 – Value times 32,768ns 100 – Value times 1,048,576ns 101 – Value times 33,554,432ns 110-111 – Not Permitted
15:13	RW	0x0	Reserved
25:16	RW	0x0	Max No-Snoop Latency Value Along with the Max No-Snoop Latency Scale field, this register specifies the maximum no-snoop latency that a device is permitted to request. Software should set this to the platform's maximum supported latency or less. Field is also an indicator of the platforms maximum latency, should an endpoint send up LTR Latency Values with the Requirement bit not set.
28:26	RW	0x0	Max No-Snoop Latency Scale — This register provides a scale for the value contained within the Maximum Non-Snoop Latency Value field. Encoding: 000 – Value times 1 ns 001 – Value times 32 ns 010 – Value times 1,024 ns 011 – Value times 32,768 ns 100 – Value times 1,048,576 ns 101 – Value times 33,554,432 ns 110-111 – Not Permitted
31:29	RW	0x0	Reserved.

§ §

10.0 Manageability

10.1 Platform Configurations

This section describes the hardware configurations for platform management. It describes the partitioning of platform manageability among system components and the functionality provided by the 82580 in each of the platform configurations.

The 82580 interfaces with an on-board BMC component for basic manageability functionality. The link between the 82580 and the BMC is either NC-SI or SMBus.

Note: Link configuration is loaded from the “Redirection LAN Interface” EEPROM bits in the Manageability Capability/Manageability Enable word (word 0x54 - see Section 6.6.6).

10.1.1 On-Board BMC Configurations

Figure 10-1 depicts a SMBus-only connection between the 82580 and the BMC. All communication between the 82580 and the BMC (pass-through traffic, configuration, and status) pass through a single interface. The protocol details for this configuration are according to the SMBus commands described in Section 10.2.2.

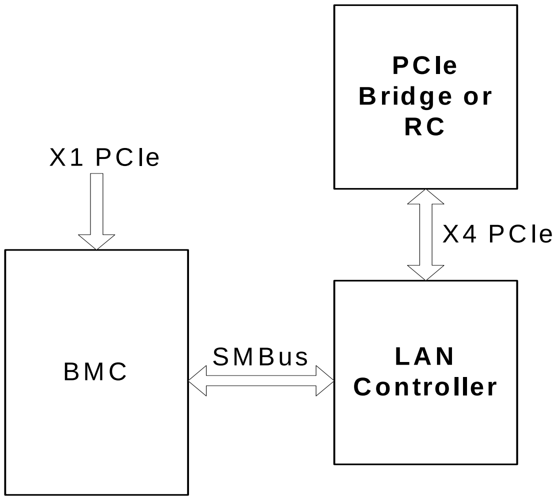


Figure 10-1. 82580 to BMC Connectivity Through a SMBus Connection

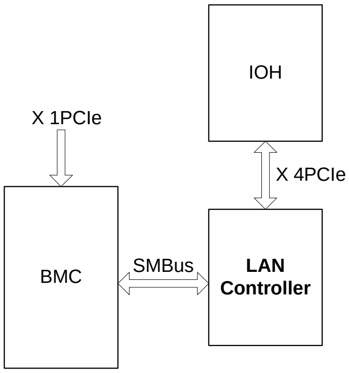


Figure 10-2. Intel® 82580 Quad/Dual Gigabit Ethernet Controller to BMC Connectivity Through SMBus Link

Figure 10-3 depicts an NC-SI connection between the 82580 and the BMC. All communication between the 82580 and the BMC (pass-through traffic, configuration, and status) pass through a single interface. The protocol details for this configuration are according to the DMTF NC-SI protocol.

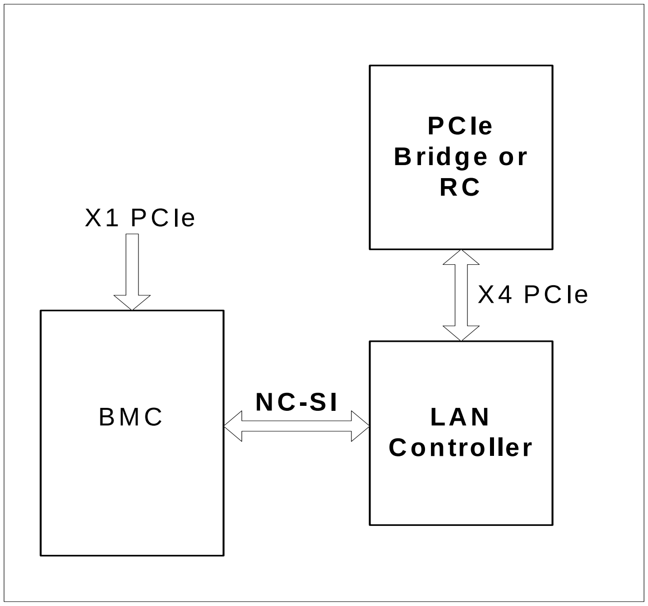


Figure 10-3. 82580 to BMC Connectivity Through a NC-SI Link

See following sections for description of traffic types that use the NC-SI or SMBus interfaces.

10.2 Pass Through Functionality

The 82580 supports traffic pass through to an external BMC. The usable bandwidth for either direction is up to 100 Mb/s in NC-SI mode & up to 400 Kbps in SMBus mode. The following list describes the different flows that can make up pass through traffic:

- BMC management traffic
- Keyboard or mouse traffic for KVM (low data rate)
- Video traffic for KVM (low average rate of 150 Kbytes/s to 200 Kbytes/s) - transmit only
- USB 2.0 redirect (up to 50 Mb/s)
- IDE redirect for remote CD/floppy (rate - priority 1 - CDx7 = 1.05 Mb/s. Priority 2 - CDx24 = 64 Mb/s
- Serial Over LAN (SoL) - 300 Kbytes/s

Pass through traffic is carried through a NC_SI interface or SMBus (legacy devices) based on the configuration loaded from the EEPROM. The management interface does not change dynamically and is loaded only during a the 82580 power up.

For a description of how traffic is carried over each of the above interfaces, see Section 3.2.1 and Section 3.2.2.

10.2.1 DMTF NC-SIMode

The 82580 supports all the mandatory features of the DMTF NC-SI spec rev 1.0.0a.

10.2.1.1 Supported Features

Table 10-1 describes the NC-SI commands supported by the 82580.

Table 10-1. Standard NC-SI Commands Support

Command	Supported
Clear initial state	Yes
Get Version ID	Yes
Get Parameters	Yes
Get Controller Packet Statistics	Yes, partially
Get Link Status	Yes[1]
Enable Channel	Yes
Disable Channel	Yes
Reset Channel	Yes
Enable VLAN	Yes[2,3]
Disable VLAN	Yes
Enable Broadcast Filtering	Yes
Disable Broadcast Filtering	Yes
Set MAC Address	Yes
Get NC-SI Statistics	Yes, partially
Set NC-SI Flow-Control	No
Set Link	Yes[1]
Enable Global Multicast Filtering	Yes
Disable Global Multicast Filtering	Yes
Get Capabilities	Yes
Set VLAN Filter	Yes
AEN Enable	Yes
Get NC-SI Pass-Through Statistics	Yes, partially
Select Package	Yes
Deselect Package	Yes
Enable Channel Network TX	Yes
Disable Channel Network TX	Yes
OEM Command	Yes

[1] When working with external PHY, this command is not supported.

[2] The 82580 does not support filtering of User priority/CFI Bits of VLAN

[3] If VLAN only mode is enabled and VLAN Filter settings are not enabled The 82580 will forward to BMC all packets with VLAN header that pass MAC address filtering.

Table 10-2 describes the optional NC-SI features supported by the 82580.

Table 10-2. Optional NC-SI Features Support

Feature	Supported	Details
AENs	Yes.	Note: The Driver state AEN may be emitted up to 15 sec. after actual driver change.
Get Controller Packet Statistics command	Yes, partially	Supports the following counters[1]: 2-9,13-18
Get NC-SI statistics command	Yes, partially	Supports the following counters[2]: 1-4, 7

Table 10-2. Optional NC-SI Features Support (Continued)

Feature	Supported	Details
Get NC-SI Pass-Through statistics command	Yes, partially	Supports counter 2 and also supports the following counters only when the OS is down: 1, 6, 7
VLAN modes	Yes, partially	Supports only modes 1,3
Buffering capabilities	Yes	Internal Buffering size according to number of ports enabled for BMC traffic[3]: • 4 LAN Ports enabled 2 KByte per port. • 2 LAN ports are enabled 4 KByte per port. • 1 LAN port enabled 8 KByte per port. Note: Internal Buffer size specified is for large packets. For small packets effective Internal buffer size can be reduced to 1.6 KByte per port.
MAC Address filters	Yes	Supports 2 MAC addresses per port
Channel count	Yes	Supports 4 channels
VLAN filters	Yes	Supports 8 VLAN filters per port
Broadcast filters	Yes	Supports the following filters: • ARP • DHCP • NetBIOS
Multicast filters	Yes	Supports the following filters[4]: • IPv6 Neighbor Advertisement • IPv6 Router Advertisement • DHCPv6 relay and server multicast
NC-SI Flow Control command	No	
Hardware Arbitration	No	

[1] *TCTL.EN* should be set to 1b to activate TX related counters and *RCTL.RXEN*, *MANC.RCV_EN* or *WUC.APME* should be set to enable RX related counters.

[2] the 82580 does not increment the NC-SI Control Packets Dropped counter when packets with Checksum errors or invalid NC-SI Channel ID are dropped.

[3] Ports enabled defined in *Manageability Capability/Manageability Enable* EEPROM word (Word 0x54) bits 7:4.

[4] Supports only when all three filters are enabled.

10.2.2 SMBus Pass Through (PT) Functionality

In addition to carrying pass through traffic, in SMBus mode, the 82580 enables loading default configuration of filters by EEPROM. When working in SMBus mode, the default values of the manageability receive filters can be set according to the PT LAN structure defined in Section 6.8.

10.2.2.1 Pass Through (PT) Modes

In pass through mode the manageability traffic is handled by the BMC. The 82580 is presented on the manageability link as four different devices (via different SMBus addresses) where each device is connected to a different LAN port. There is no logical connection between the devices. Fail-over if required between the LAN ports is done by the external BMC (by sending/receiving packets through different ports). The status reports to the BMC, ARP handling, DHCP and other pass through functionality are unique for each port.

10.3 Programming Interfaces

10.3.1 NC-SI Programming

In addition to the regular NC-SI commands described in Table 10-1 the following Intel vendor specific commands are supported. The purpose of these commands is to provide a means for the Management Controller (BMC) to access some of the Intel specific features present in the 82580.

10.3.1.1 Method

The following features are abstracted via NC-SI using the OEM specific command:

- Get System MAC Address – This command allows the BMC to retrieve the System MAC address used by the NC. This MAC address may be used for a “Shared MAC Address” mode.
- “Keep PHY Link Up” (Veto bit) Enable/Disable – This feature allows the BMC to block PHY reset, which might cause session loss.
- Force TCO – Allows the Management Controller (BMC) to:
  - — Reset the Network Adapter.
  - — Disable write operation of Host to function connected to the LAN port.
  - — Issue a Firmware reset to re-load FW related EEPROM words.
- Checksum offloading – offloads IP/UDP/TCP checksum checking from the Management Controller.

These commands are designed to be compliant with their corresponding SMBus commands (if exist).

All of the commands are based on a single DMTF defined NC-SI command, known as “OEM Command”. This command is detailed in Section 10.3.1.2.

10.3.1.2 OEM Command (0x50)

The OEM command may be used by the Management Controller to request the sideband interface to provide vendor-specific information. The Vendor Enterprise Number 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		

10.3.1.3 OEM Response (0xD0)

The sideband interface shall return Unknown Command Type reason code for any un-recognized enterprise number using the following frame format. If the command is valid, the response, if any, is allowed to be vendor-specific. It is recommended to use the 0x8000 range for vendor-specific code.

	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		

Table 10-3. OEM Specific Command Response and Reason Codes

Response Code		Reason Code	
Value	**Description**	**Value**	**Description**
0x1	Command Failed	0x5081	Invalid Intel Command Number
		0x5082	Invalid Intel Command Parameter Number
		0x5085	Internal Network Controller Error
		0x5086	Invalid Vendor Enterprise Code
		0x20003	Invalid parameter

10.3.1.4 OEM Command Summary

Table 10-4. OEM Command Summary

Intel Command	Parameter	Command name
0x06	N/A	Get System MAC Address
0x20	N/A	Set Intel Management Control
0x21	N/A	Get Intel Management Control
0x22	N/A	Force TCO (reset, Isolate and FW reset)
0x23	N/A	Enable IP/UDP/TCP Checksum offloading
0x24	N/A	Disable IP/UDP/TCP Checksum offloading

10.3.1.5 System MAC Address

10.3.1.5.1 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 (i.e. – "Shared MAC Address" mode), the following command is provided to allow the MC to query the NC for a valid MAC address.

The NC shall 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 uses the Packet Reduction and Manageability Only 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			

10.3.1.5.2 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	*MSB* MAC Address		
28…30	MAC Address *LSB*			

10.3.1.6 Set Intel Management Control Formats

10.3.1.6.1 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.

Table 10-5. Intel Management Control 1

Bit #	Default value	Description
0	0b	Enable Critical Session Mode (a.k.a. "Keep PHY Link Up", "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 - LAN_PWR_GOOD reset, Device Disable, Force TCO, and PHY reset by SW. 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

10.3.1.6.2 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		

10.3.1.7 Get Intel Management Control Formats

10.3.1.7.1 Get Intel Management Control Command (Intel Command 0x21)

	Bits			
Bytes	**31…24**	**23…16**	**15…08**	**07…00**
00…15	NC-SI Header			
16…19	Manufacturer ID (Intel 0x157)			
20…21	0x21	0x00		

Where

"Intel Management Control 1" is as described in the "Set Intel Management Control" command (See Section 10.3.1.6).

10.3.1.7.2 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	

10.3.1.8 Force TCO

Depending on the bit set in the TCO mode field this command will cause the 82580 to perform either:

1. TCO Reset, if Force TCO reset is enabled in the EEPROM (see Section 6.6.7). The Force TCO reset will clear the data-path (RX/TX) of the 82580 to enable the BMC to transmit/receive packets through the 82580.
   - If the BMC has detected that the OS is hung and has blocked the RX/TX path The Force TCO reset will clear the data-path (RX/TX) of the Network Controller to enable the BMC to transmit/receive packets through the Network Controller.
   - When this command is issued to a channel in a package, it applies only to the specific channel.
   - After successfully performing the command the Network Controller will consider Force TCO command as an indication that the OS is hung and will clear the DRV_LOAD flag (disable the driver). If TCO reset is disabled in EEPROM the 82580 clears the *CTRL_EXT.DRV_LOAD* bit but does not reset the data-path and notifies BMC on successful completion.
   - Following TCO reset management sets *MANC.TCO_RESET* to 1.
2. TCO isolate, if TCO isolate is enabled in the EEPROM (See Section 6.6.6). The TCO Isolate command will disable PCIe write operations to the LAN port.
   - If TCO Isolate is disabled in EEPROM The 82580 does not execute the command but sends a response to the BMC with successful completion.
   - Following TCO Isolate management sets *MANC.TCO_Isolate* to 1.
3. Firmware Reset. This command will cause re-initialization of all the manageability functions and re-load of manageability related EEPROM words (e.g. Firmware patch code).
   - When the BMC has loaded new management related EEPROM image (e.g. Firmware patch) the Firmware Reset command will load management related EEPROM information without need to power down the system.
   - This command is issued to the package and affects all channels. After the Firmware reset the FW Semaphore register (FWSM) is re-initialized.
   - On reception of Firmware Reset command management sets *MANC.FW_RESET* to 1 and increments the FWSM.Reset_Cnt field.

Notes: Force TCO reset and TCO Isolate will affect only the channel (port) that the command was issued to.

Following firmware reset, BMC will need to re-initialize all ports.

10.3.1.8.1 Perform Intel Force TCO Command (Intel Command 0x22)

	Bits			
Bytes	31...24	23...16	15...08	07...00
00...15	NC-SI Header			
16...19	Manufacturer ID (Intel 0x157)			
20	0x22	TCO Mode		

Where:

TCO Mode:

Field	Bit(s)	Description
DO_TCO_RST[1]	0	Do TCO reset. 0b = Do Nothing 1b = Perform TCO Reset Note: Setting this bit generates a one time LAN port reset event.
DO_TCO_ISOLATE[2]	1	Do TCO Isolate 0b = Enable PCIe write access to LAN port. 1b = Isolate Host PCIe write operation to the port Note: Should be used for debug only.
Firmware Reset	2	Reset Manageability and re-load Manageability related EEPROM words (see Section 3.3.1.3). 0b = Do Nothing 1b = Issue Firmware Reset to manageability. Note: Setting this bit generates a one time Firmware reset event. Following Firmware Reset Management related data from EEPROM is loaded.
Reserved	3-7	Reserved. (Set to 0x00)

[1] TCO Reset operation enabled in EEPROM.

[2] TCO Isolate Host Write operation enabled in EEPROM.

10.3.1.8.2 Perform Intel Force TCO 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	0x22			

Note: If Firmware Reset is asserted in Force TCO command the 82580 does not send a response, but executes Firmware Reset silently.

10.3.1.9 Checksum offloading

This command will enable the checksum offloading filters in the Network Controller.

When enabled, these filters will block any packets that did not pass IP, UDP & TCP checksum from being forwarded to the Management Controller.

10.3.1.9.1 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			

10.3.1.9.2 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	0x23			

10.3.1.9.3 Disable Checksum Offloading Command (Intel Command 0x24)

This command shall cause the Network Controller to stop verifying the IP/UDP/TCP checksum.

	Bits			
Bytes	**31…24**	**23…16**	**15…08**	**07…00**
00…15	NC-SI Header			
16…19	Manufacturer ID (Intel 0x157)			
20	0x24			

10.3.1.9.4 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	0x24			

10.3.2 SMBus Programming

This section describes the SMBus transactions supported in APT mode.

10.3.2.1 Write SMBus Transactions (BMC → 82580)

Table 10-6 below summarizes the different SMBus write transactions supported by the 82580.

Table 10-6. SMBus Write Transactions

TCO Command	Transaction	Command		Fragmentation	Section
Transmit Packet	Block Write	First: Middle: Last:	0x84 0x04 0x44	Multiple	10.3.2.1.1
Transmit Packet	Block Write	Single:	0xC4	Single	10.3.2.1.1
Receive Enable	Block Write	Single:	0xCA	Single	10.3.2.1.3
Management Control	Block Write	Single:	0xC1	Single	10.3.2.1.5
Update MNG RCV filter parameters	Block Write	Single:	0xCC	Single	10.3.2.1.6
Force TCO	Block Write	Single:	0xCF	Single	10.3.2.1.4
Request Status	Block Write	Single:	0xDD	Single	10.3.2.1.2

10.3.2.1.1 Transmit Packet Command

The Transmit Packet command behavior is detailed in section 3.2.1.1.4.

The Transmit Packet fragments will have the following format.

Function	Command	Byte Count	Data 1	...	Data N
Transmit first fragment	0x84	N	Packet data MSB	...	Packet data LSB
Transmit middle fragment	0x04				
Transmit last fragment	0x44				
Transmit single fragment	0xC4				

If the overall packet length is bigger than 1536 bytes or smaller than 32 bytes (including padding), the packet is discarded by the 82580 and Abort is asserted.

10.3.2.1.2 Request Status Command

An external MC can initiate a request to read manageability status by sending a Request Status command. When received, the 82580 initiates a notification to an external MC when status is ready. After this, the external controller will be able to read the status, by issuing a read status command (see section 10.3.2.2.3).

Request Status Command format:

Function	Command	Byte Count	Data 1
Request status	0xDD	1	0

10.3.2.1.3 Receive Enable Command

The Receive Enable command is a single fragment command that is used to configure the 82580.

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).

Note: If the receive enable command is short and thus does not include all the parameters, then the parameters will be taken from most recent previous configuration (either the most recent long Receive-Enable command in which the particular value was set, or the EEPROM if there was no such previous long Receive-Enable command).

Func.	Cmd	Byte Count	Data 1	Data 2	...	Data 7	Data 8	...	Data 11	Data 12	Data 13	Data 14
Legacy receive enable	0xCA	1	Receive control byte	-	...	-	-	...	-	-	-	-
Advanced receive enable		14 0x0E		MAC addr. MSB		MAC addr. LSB	RSV		RSV	BMC SMBus addr.	Interf. data byte	Alert value byte

Where:

- Receive control byte (data byte 1):

Field	Bit(s)	Description
RCV_EN	0	Receive TCO enable. 0b = Disable Receive and Transmit of TCO packets 1b = Enable Receive and Transmit of TCO packets. Setting this bit will enable BMC to network and network to BMC traffic and all manageability receive filtering operations (Will set bit *MANC.RCV_TCO_EN*). The enable of the specific filtering is done through a special configuration command (see Section 10.3.2.1.6). **Notes:** 1. When this bit is cleared all receive TCO functionality is disabled, not just the packets that are directed to the BMC. 2. Bit can also be set using the *Enable TCO to Network* bit in the *Update MNG RCV Filter Parameters* command (Parameter Filter Enable, See Table 10-7).
Reserved	1	Reserved
Reserved	1	Reserved. Write 0 ignore on read.
EN_STA	2	Enable Status reporting to the BMC, on status change. 0b = Disable status reporting 1b = Enable status reporting
Reserved	3	Reserved, must be 0b.
NM	4-5	Notification Method. Define the notification method the the 82580 will use. 00b = SMBus Alert 01b = Asynchronous Notify 10b = Direct Receive 11b = Not Supported. Note: In multiple SMBus address mode, all SMBus addresses must be configured to the same notification method.

Field	Bit(s)	Description
Reserved	6	Reserved, must be 1b.
CBDM	7	Configure BMC dedicated MAC Address. Note: This bit should be zero when the RCV_EN bit (bit 0) is not set. 0b = The 82580 will share MAC address for MNG traffic with host MAC address which is specified in EEPROM words located at LAN Base Address + offset 0x0-0x2 per port (See Section 6.2.1). 1b = The 82580 will use the BMC dedicated MAC address as filter for incoming receive packets. The BMC MAC address is set in bytes 2-7 in this command. If short version of the command is used, the 82580 will use 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 82580 will use MAC address 1(Mac address programmed in MMAL/H1 registers). When Dedicated MAC address feature is activated, the 82580 will use the following registers to filter in all the traffic addressed to the BMC MAC. BMC should not modify these registers: Manageability Decision Filter – MDEF7 (and corresponding bit 7 in Management Only traffic Register – MNGONLY) Manageability MAC Address Low – MMAL1 Manageability MAC Address High – MMAH1

- MNG MAC Address (data bytes 2-7)

  Ignored if CBDM bit is not set. This MAC address will be used for configuration of the dedicated MAC address. This MAC address will continue to be used when CBDM bit is set in subsequent short versions of this command.

- Reserved (data bytes 8-11).
- Asynchronous Notification SMBus address (byte 12)

  This address will be used for the Asynchronous Notification SMBus transaction and for Direct Receive.

- Interface Data (Data byte 13)

  Interface data byte to be used in Asynchronous Notification.

- Alert data (Data byte 14).

  Alert Value data byte to be used in the Asynchronous Notification.

10.3.2.1.4 Force TCO Command

Depending on the bit set in the TCO mode field this command will cause the 82580 to perform either:

1. TCO Reset, if Force TCO reset is enabled in the EEPROM (see Section 6.6.7). The Force TCO reset will clear the data-path (RX/TX) of the 82580 to enable the BMC to transmit/receive packets through the 82580.
   - The 82580 will consider Force TCO reset command as an indication that the OS is hung and will clear the DRV_LOAD flag. If TCO reset is disabled in EEPROM the 82580 clears the *CTRL_EXT.DRV_LOAD* bit but does not reset the data-path.
   - Following TCO reset management sets *MANC.TCO_RESET* to 1.
2. TCO isolate, if TCO isolate is enabled in the EEPROM (see Section 6.6.6). The TCO Isolate command will disable PCIe write operations to the LAN port.
   - If TCO Isolate is disabled in EEPROM The 82580 does not execute the command but sends a response to the BMC with successful completion.
   - Following TCO Isolate management sets *MANC.TCO_Isolate* to 1.

3. Firmware Reset. This command will cause re-initialization of all the manageability functions and re-load of manageability related EEPROM words (e.g. Firmware patch code).
   - On reception of Firmware reset command management sets *MANC.FW_RESET* to 1 and increments the FWSM.Reset_Cnt field.

Note: Force TCO reset will effect only the port related to the SMBus address the command was issued to.

Following firmware reset, BMC will need to re-initialize all ports.

Force TCO Command format:

Function	Command	Byte Count	Data 1
Force TCO	0xCF	1	TCO mode

Where TCO Mode field controls the following operations:

Field	Bit(s)	Description
DO_TCO_RST[1]	0	Do TCO reset. 0b = Do Nothing 1b = Perform TCO Reset Note: Setting this bit generates a one time LAN port reset event.
DO_TCO_ISOLATE[2]	1	Do TCO Isolate 0b = Enable PCIe write access to LAN port. 1b = Isolate Host PCIe write operation to the port Note: Should be used for debug only.
Firmware Reset	2	Reset Manageability and re-load Manageability related EEPROM words (see Section 3.3.1.3). 0b = Do Nothing 1b = Issue Firmware Reset to manageability. Note: Setting this bit generates a one time Firmware reset event. Following Firmware Reset Management related data from EEPROM is loaded.
Reserved	3-7	Reserved (Set to 0x00).

[1] TCO Reset operation enabled in EEPROM.

[2] TCO Isolate Host Write operation enabled in EEPROM.

10.3.2.1.5 Management Control

This command is used to set generic manageability parameters. The parameters list is shown in the table below. The command is C1h, 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 the table.

Note: If in the update configuration, the parameter that the BMC sets is not supported by the 82580, the 82580 will not NACK the transaction. After the transaction ends, the 82580 will discard the data and will assert a transaction abort status (see Section 3.2.1.1.5).

This is the format of the Management control command:

Function	Command	Byte Count	Data 1	Data 2	...	Data N
Management control	0xC1	N	Parameter number	Parameter dependent		

The table below shows the different parameters and their content:

Parameter	Parameter Number	Parameter Data
Keep PHY Link Up	0x00	A single byte parameter: Data 2 - Bit 0 Set to indicate that the PHY link for this port should be kept up. Sets the Keep_PHY_Link_Up bit. When cleared, clears the Keep_PHY_Link_Up bit. See Section 8.22.4. Bit [7:1] Reserved See Section 4.3.4 for details of this bit behavior.

10.3.2.1.6 Update MNG RCV Filter Parameters

This command is used to set the manageability receive filters parameters. The parameters list is shown in the table below. The command is 0xCC, which states that it is a parameter update. The first data byte is the parameter number and the data after-words (length and content) are parameter specific as shown in the table.

Note: If in the update configuration, the parameter that the BMC sets is not supported by the 82580, the 82580 will not NACK the transaction. After the transaction ends, the 82580 will discard the data and will assert a transaction abort status (see Section 3.2.1.1.5).

Detailed description of receive filtering capabilities and configuration is in Section 10.4

This is the format of the Update MNG RCV filter parameters command:

Function	Command	Byte Count	Data 1	Data 2	...	Data N
Update MNG RCV filter parameters	0xCC	N	Parameter number	Parameter dependent		

The table below shows the different parameters and their contents**.**

Parameter	#	Parameter Data
Filters Enable	0x1	Defines generic filters configuration. The structure of this parameter is 4 bytes as shown in Table 10-7. Where Data 2 is the MSB and Data 5 is the LSB. The parameter updates the MANC register. Note the general filter enable is in the receive enable command. Which enable receive filtering. This parameter specifies which filters should be enabled. ARP filtering and dedicated MAC address can also be enabled through the receive enable command (see Section 10.3.2.1.3).
MNGONLY configuration	0xF	This parameter defines which of the packets types identified as manageability packets in the receive path will never be directed to the host memory. Data 2:5: MNGONLY register bytes (see Section 8.22.5) - Data 2 is the MSB
Flex filter 0 enable MASK and length	0x10	Flex filter 0 mask. This parameter defines the Mask bits in the FTFT filter. Data 2:17 – MASK. Bit 0 in data 2 is the first bit of the MASK Data 18:19 – Reserved. Should be zero. Data 20 – Flexible Filter length. Must be greater than or equal one.
Flex filter 0 data	0x11	Data 2 – Group of Flex filter’s bytes. This parameter defines the Data pattern bytes in the FTFT filter: 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 mandatory 30 bytes; it may vary according to filter’s length and must NOT be padded by zeros.
VLAN Filters	0x62	3 bytes to load the VLAN tag filters (MAVTV) Data 2 – VLAN Filter number Data 3 – MSB of VLAN Filter Data 4 – LSB of VLAN Filter
Flex Ports Filters	0x63	3 bytes to load the manageability flex port filters (MFUTP). Data 2 – Flex Port Filter number Data 3 – MSB of flex port filter Data 4 - LSB of flex port filter
IPv4 Filters	0x64	5 bytes to load the IPv4 address filter (MIPAF, DW 15:12) Data 2 – IPv4 address filter number (0-3) Data 3 – LSB of IPv4 address filter … Data 6 – MSB of IPv4 address filter
IPv6 Filters	0x65	17 bytes to load IPv6 address filter (MIPAF) Data 2 – IPv6 address filter number (0-3) Data 3 – LSB of IPv6 address filter … Data 18 – MSB of IPv6 address filter

Parameter	#	Parameter Data
MAC Filters	0x66	7 bytes to load MAC address filters (MMAL, MMAH) Data 2 – MAC address filters pair number (0-1) Data 3 – MSB of MAC address … Data 8 – LSB of MAC address
Ethertype Filters	0x67	5 bytes to load Ethertype filters (METF) Data 2 - METF filter index (valid values are 0,1,2 and 3) Data 3 - MSB of METF ... Data 6 - LSB of METF
Extended Decision Filter	0x68	9 bytes to load the extended decision filters (MDEF_EXT & MDEF) Data 2 - MDEF filter index (valid values are 0...6) Data 3 - MSB of MDEF_EXT (DecisionFilter1) .... Data 6 - LSB of MDEF_EXT (DecisionFilter1) Data 7 - MSB of MDEF (DecisionFilter0) .... Data 10 - LSB of MDEF (DecisionFilter0) The command shall overwrite any previously stored value.

Table 10-7. Filter Enable Parameter

Bit	Name	Description
31:27	Reserved	Reserved.
26	NET_TYPE	NET TYPE: 0b = pass only un-tagged packets. 1b = pass only VLAN tagged packets. Valid only if FIXED_NET_TYPE is set.
25	FIXED_NET_TYPE	Fixed net type: If set, only packets matching the net type defined by the NET_TYPE field passes to manageability. Otherwise, both tagged and un-tagged packets can be forwarded to the manageability engine.
24	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.
23	Enable Xsum Filtering to MNG	When this bit is set, only packets that pass the L3 and L4 checksum are send to the manageability block.
22:19	Reserved	Reserved
18	KEEP_PHY_LINK_UP	Block PHY reset and power state changes. When this bit is set the PHY reset and power state changes does not get to the PHY, This bit can not be written unless Keep_PHY_Link_Up_En EEPROM bit is set.
17	Enable TCO to Network	Enable TCO receive and transmit Traffic to Network Note: Bit can also be set using the RCV_EN bit in the *receive enable* command (see Section 10.3.2.1.3).
16:0	Reserved	Reserved.

10.3.2.2 Read SMBus Transactions (82580 to BMC)

The table below summarizes the different SMBus read transactions supported by the 82580. All the read transactions are compatible with SMBus Read Block Protocol format.

TCO Command	Transaction	Command	Op-Code		Fragmentation	Section
Receive TCO Packet	Block Read	0xC0 or 0xD0	First: Middle: Last[1]	0x90 0x10 0x50	Multiple	10.3.2.2.1
Read Receive Enable configuration	Block Read	0xDA	Single:	0xDA	Single	10.3.2.2.7
Read 82580 Status	Block Read	0xC0 or 0xD0 or 0xDE	Single:	0xDD	Single	10.3.2.2.3
Read Management parameters	Block Read	0xD1	Single:	0xD1	Single	10.3.2.2.5
Read MNG RCV filter parameters	Block Read	0xCD	Single:	0xCD	Single	10.3.2.2.6
Get system MAC address	Block Read	0xD4	Single	0xD4	Single	10.3.2.2.4

[1] Last fragment of the receive TCO packet is the packet status.

Notes: 0xC0/0xD0 commands are used for more than one payload. If the BMC issues these read commands, and the 82580 has no pending data to transfer, it will always return as default opcode 0xDD with the 82580 status, and will not NACK the transaction.

10.3.2.2.1 Receive TCO LAN Packet Transaction

The BMC will use this command to read the packet received on the LAN and its status. When the 82580 has a packet to deliver to the BMC, it asserts the SMBus notification, for the BMC to read the data (for direct receive frame format see Section 3.2.1.1.2.3). Upon receiving notification of the arrival of LAN receive packet, the BMC should begin issuing a Receive TCO packet command using the block read protocol. The packet can be delivered in more than one SMBus fragment (at least two - one for the packet, and the other one for the status), and the BMC should follow the “F” and “L” bits (2 MSB bits of the op-code).

The op-code can have these values:

- 0x90 - First Fragment
- 0x10 - Middle Fragment.
- 0x50 - Packet status (last fragment) as described below in Section 10.3.2.2.2.

It should be noted that the Receive packet will be silently discarded if following packet reception and notification no valid BMC transaction on the SMBus is detected (no receive fragment is read by the BMC, no Status Read operation is executed by the BMC...) within a timeout period. The timeout period is set according to the SMBus notification timeout EEPROM parameter (See Section 6.10.4).

Function	Command
Receive TCO packet	0xC0 or 0xD0

Data returned from the 82580:

Function	Byte Count	Data 1 (Op-Code)	Data 2	...	Data N
Receive TCO First Fragment	N	90	Packet Data Byte	...	Packet Data Byte
Receive TCO Middle Fragment		10			
Receive TCO Last Fragment	17 (0x11)	50	TCO status, see Section 10.3.2.2.2		

10.3.2.2.2 Receive TCO LAN Status Payload Transaction

This transaction is the last transaction that the 82580 will issue when a packet that was received from the LAN is transferred to the BMC. The transaction will contain the status of the received packet.

This is the format of the status transaction:

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 0-7) is set in Data 2 of the status and byte 15 is set in Data 17 of the status.

The table below describes the content of the status data:

Table 10-8. TCO LAN Packet Status Data

Name	Bits	Description
Packet Length	13:0	Packet length including CRC, only 14 LSB bits.
Packet status	35:14	See Table 10-9
Reserved	42:36	Reserved
Error	47:43	See Table 10-10
VLAN	63:48	The two bytes of the VLAN header tag.
Reserved	67:64	Reserved
Packet type	80:68	See Table 10-12
Reserved	84:81	Reserved
MNG status	127:85	See Table 10-13. This field should be ignored if Receive TCO is not enabled,

The meaning of the bits inside of each field can be found in Section 7.1.5.

Table 10-9. Packet Status Info

Field	Bit(s)	Description
LAN#	21:20	Indicates the source port of the packet
VP	19	VLAN Stripped –insertion of VLAN TAG is needed.
VEXT	18	Additional VLAN present in packet
Reserved	17:15	Reserved
Reserved	14:12	Reserved
CRC stripped	11	Insertion of CRC is needed.

Table 10-9. Packet Status Info

Field	Bit(s)	Description
Reserved	10:6	Reserved
UDPV	5	UDP checksum valid
Reserved	4:3	Reserved
IPCS	2	Ipv4 Checksum Calculated on packet
L4I	1	L4 (TCP/UDP) Checksum calculated on packet
UDPCS	0	UDP checksum calculated on packet

Table 10-10. Error Status Info

Field	Bit(s)	Description
RXE	4	RX Data Error
IPE	3	Ipv4 Checksum Error
L4E	2	L4 (TCP/UDP) Checksum Error
Reserved	1:0	Reserved

Table 10-12. Packet type

Bit Index	Bit 11 = 0b	Bit 11 = 1b (L2 packet)
12	VLAN packet indication	
11	Packet matched one of the ETQF filters.	
10:8	Reserved	Reserved
7	Reserved	Reserved
6		
5		
4		
3		
2		EtherType - ETQF register index that matches the packet. Special types might be defined for 1588, 802.1X, or any other requested type.
1		
0		

Table 10-13. MNG Status

Name	Bits	Description
Decision Filter match	42:35	Set when there is a match to one of the Decision filters
IPv4/IPv6 match	34	Set when there is an IPv6 match and cleared when there’s an IPV4 match. This bit is valid only if bit 33 (IP match bit) is set.
IP address match	33	Set when there is a match to any of the IP address filters
IP address Index	32:31	Set when there is a match to the IP filter number. (IPv4 or IPv6)
Flex TCO filter match	30	Set when there is a match to the Flex port filter
Reserved	29:27	Reserved
L4 port match	26	Set when there is a match to any of the UDP / TCP port filters
L4 port Filter Index	25:19	Indicate the flex filter number
Unicast Address match	18	Set when there is a match to any of the 2 Unicast MAC addresses.
Unicast Address Index	17:15	Indicates which of the 2 Unicast MAC addresses match the packet. Valid only if the Unicast Address match is set.
MNG VLAN Address Match	14	Set when the MNG packet matches one of the MNG VLAN filters

Table 10-13. MNG Status (Continued)

Name	Bits	Description
Pass MNG VLAN Filter Index	13:11	Indicates which of the Vlan filters match the packet.
Reserved	10:8	Reserved
Pass ARP req / ARP resp	7	Set when the MNG packet is an ARP response/request packet
Pass MNG neighbor	6	Set when the MNG packet is a neighbor discovery packet.
Pass MNG broadcast	5	Set when the MNG packet is a broadcast packet
Pass RMCP 0x0298	4	Set when the UDP/TCP port of the MNG packet is 0x298
Pass RMCP 0x026F	3	Set when the UDP/TCP port of the MNG packet is 0x26F
Manageability Ethertype filter passed	2	Indicates that one of the METF filters matched
manageability Ethertype filter index	1:0	Indicates which of the METF filters matched

10.3.2.2.3 Read Status Command

The BMC can read the 82580 status. The 82580 will assert an alert prior to the BMC reading the status bytes. There can be two reasons for the 82580 to send status to the BMC:

1. The external BMC asserts a request for reading the 82580 status (Section 3.2.1.1.2).
2. The 82580 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 (Section 3.2.1.1.2).

Note that commands C0h/D0h are for backward compatibility. 0xD0/0xC0 can be used for other payloads the 82580 will define in the op-code, which payload this transaction will be. When 0xDE command is set, the 82580 will always return opcode 0xDD with the 82580 status. The BMC will read the event causing the notification, using the read status command shown below:

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	

The table below shows Status Data 1 byte:

Bit	Name	Description
7	LAN Port Lsb	LAN port Lsb together with Lan Port Msb define port that sent status. See further information in description of Lan Port Msb (bit 2).
6	TCO command aborted	0b = A TCO command abort event has not occurred since the last Read Status cycle. 1b = A TCO command abort event has occurred since the last Read Status cycle. See Section 3.2.1.1.5 for command abort flow.
5	Link Status indication	0b = LAN link down 1b = LAN link is up
4	PHY Link Forced Up	Contains the value of the MANC.Keep_PHY_Link_Up bit. When set, indicates that the PHY link is configured to keep link up. See Section 8.22.4.

Bit	Name	Description
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. See note 1.
2	LAN Port Msb	Defines together with LAN Port Lsb the port that sent the Status: Lan Port Msb Lan Port Lsb 0 0 Status came from LAN port 0. 0 1 Status came from LAN port 1. 1 0 Status came from LAN port 2. 1 1 Status came from LAN port 3.
1:0	Power state	00b = Dr state 01b = D0u state 10b = D0 state 11b = D3 state

Notes:

1. This indication will be asserted when the 82580 manageability block will reload EEPROM and its internal data-base will be updated to EEPROM default values. This is an indication that the external BMC should re-configure the 82580, if other values beside the EEPROM default should be configured.

Status data byte 2 is used by the BMC as an indication whether the LAN driver is alive and running.

The driver valid indication is a bit that is set by the driver when it is coming up, and cleared when it goes down to Dx state or cleared by the HW on PCI reset.

Bits 2 and 1 indicate that the LAN driver is not stuck. Bit 2 indicates whether the interrupt line of the LAN function is asserted, and bit 1 indicates whether driver between the last Read Status Cycle dealt the interrupt line.

The table below shows Status Data 2 byte:

Bit	Name	Description
7	Reserved	Reserved
6	Reserved	Reserved
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	0b = LAN interrupt is not asserted. 1b = LAN interrupt is asserted.
1	ICR register read/write	0b = ICR register was not read since the last Read Status Cycle. 1b = ICR register was read since the last Read Status cycle. Reading the ICR means that the driver has dealt with the interrupt that was asserted
0	Reserved	Reserved

The table below shows the possible values of bits 2, 1 and what the BMC can assume based on their value:

Previous	Current	
Don't care	00b	Interrupt is not pending – O.K
00b	01b	New interrupt is asserted – O.K
10b	01b	New interrupt is asserted – O.K
11b	01b	Interrupt is waiting for reading – O.K
01b	01b	Interrupt is waiting for reading by driver more than one read status Cycle – Not OK (possible driver hang state).
Don't Care	11b	Previous interrupt was read and current interrupt is pending – O.K
Don't Care	10b	Interrupt is not pending – O.K

Note: The BMC reads should consider the time it takes for the driver to deal with the interrupt (few micro-seconds), too frequent reads will give false indications.

10.3.2.2.4 Get System MAC Address

The Get System MAC Address will return system MAC Address over the SMBus. This command is a single fragment Read Block transaction, which will return the following database:

The Get System MAC Address returns the system MAC Address (RAL0, RAH0).

Get System MAC Address format:

Function	Command
Get system MAC address	0xD4

Data returned from the 82580:

Function	Byte Count	Data 1 (Op-Code)	Data 2	...	Data 7
Get system MAC address	7	0xD4	MAC address MSB	...	MAC address LSB

Note:

10.3.2.2.5 Read Management Parameters

In order to read the management 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 reads the parameter.

This is the block write transaction:

Function	Command	Byte Count	Data 1
Management control Request	0xC1	1	Parameter number

Following the block write the BMC should issue a block read that will read the parameter that was set in the block write command:

Function	Command
Read management parameter	0xD1

Data returned from the 82580:

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 as follows:

Parameter	#	Parameter Data
Keep PHY Link Up	0x00	A single byte parameter: Data 3 - Bit 0 Set to indicate that the PHY link for this port should be kept up. See Section 8.22.4 and Section 4.3.4 for details on the behavior of this bit. Bit [7:1] Reserved
Wrong parameter request	0xFE	Returned by the 82580 only. This parameter is returned on read transaction, if in the previous read command the BMC sets a parameter that is not supported by the 82580.
82580 not ready	0xFF	Returned by the 82580 only, on read parameters command when the data that should have been read is not ready. This parameter has no data. The BMC should retry the read transaction.

Note: 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 1h).

It is BMC’s responsibility to follow the procedure defined above. If the BMC sends a block read command (as described above) which is not preceded by a block write command with bytecount=1, the 82580 will set Parameter Number in the read block transaction to be 0xFE.

10.3.2.2.6 Read MNG RCV Filter Parameters

In order to read the MNG 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 a block read that reads the parameter.

This is the block write transaction:

Function	Command	Byte Count	Data 1	Data 2
MNG RCV filter parameters to read	0xCC	1 or 2	Parameter number	Parameter data

The table below shows the different parameters and their contents**:**

Parameter	#	Parameter Data
Filters Enable	0x1	None
MNGONLY configuration	0xF	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
VLAN Filters	0x62	1 byte to define the accessed VLAN tag filter (MAVTV) Data 2 – VLAN Filter number
Flex Ports Filters	0x63	1 byte to define the accessed manageability flex port filter (MFUTP). Data 2 – Flex Port Filter number
IPv4 Filter	0x64	1 byte to define the accessed IPv4 address filter (MIPAF) Data 2 – IPv4 address filter number
IPv6 Filters	0x65	1 byte to define the accessed IPv6 address filter (MIPAF) Data 2 – IPv6 address filter number
MAC Filters	0x66	1 byte to define the accessed MAC address filters pair (MMAL, MMAH) Data 2 – MAC address filters pair number (0-1)
Ethertype Filters	0x67	1 byte to define Ethertype filters (METF) Data 2 - METF filter index (valid values are 0,1,2 and 3)
Extended Decision Filter	0x68	1 byte to define the extended decision filters (MDEF_EXT & MDEF) Data 2 - MDEF filter index (valid values are 0...6)
Wrong parameter request	0xFE	Returned by the 82580 only. This parameter is returned on read transaction, if in the previous read command the BMC sets a parameter that is not supported by the 82580.
82580 not ready	0xFF	Returned by the 82580 only, on read parameters command when the data that should have been read is not ready. This parameter has no data.

Following the block write the BMC should issue a block read that will read the parameter that was defined in the block write command:

Function	Command
Request MNG RCV filter parameters	0xCD

Data returned from the 82580:

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 in the same format of the “update” command 1.

Note: If 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 1h).

If the parameter number is 0xFF, it means that the data that the 82580 should supply is not ready yet. The BMC should retry the read transaction.

It is BMC's responsibility to follow the procedure defined above. If the BMC sends a block read command (as described above) which is not preceded by a block write command, the 82580 will set Parameter Number in the read block transaction to be 0xFE.

10.3.2.2.7 Read Receive Enable Configuration

The BMC uses this command to read the receive configuration data. This data can be configured in receive enable command or through EPROM loading upon power-up.

Read Receive Enable Configuration command format (SMBus Read Block Protocol):

Function	Command
Read receive enable	0xDA

Data returned from the 82580.

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 address MSB	...	MAC addres s LSB	IP addres s MSB	...	IP addres s LSB	BMC SMBus address	Interfac e data byte	Alert value byte

The detailed description of each field is specified in the receive enable command description in Section 10.3.2.1.3.

10.3.2.3 SMBus ARP Transactions

Note: All SMBus-ARP transactions include PEC byte.

10.3.2.3.1 Prepare to ARP

This command will clear the Address Resolved flag (set to false). It will not affect the status or validity of the dynamic SMBus Address (will not clear the address Valid flag). It 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	

10.3.2.3.2 Reset Device (General)

This command will clear the Address Resolved flag (set to false). It will not affect the status or validity of the dynamic SMBus Address (will not clear the address Valid flag).

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	

10.3.2.3.3 Reset Device (Directed)

The Command field is NACK-ed if the bits 7 through 1 do not match the current the 82580 SMBus address.

It will clear the Address Resolved flag (set to false). It will not affect the status or validity of the dynamic SMBus Address (will not clear the address Valid flag).

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	

10.3.2.3.4 Assign Address

This command assigns the 82580 SMBus address. The address and command bytes are always acknowledged.

The transaction will be aborted immediately (NACK-ed in simple English) if any of the UDID bytes differ from the 82580 UDID bytes as defined in Section 3.2.1.1.7. If successful, the MNG will update the SMBus address internally. This command will also set the Address Resolved flag 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	

10.3.2.3.5 Get UDID (General and Directed)

The Get UDID command depends on whether this is a directed or general command.

The General Get UDID SMBus transaction will support a constant command value of 0x03.

The Directed Get UDID SMBus transaction will support a dynamic command value equal to the dynamic SMBus address with the LSB bit set.

Note: Bit 0 (LSB) of Data byte 17 will always be 1b.

If the SMBus Address has been resolved (Address Resolved flag is true); for a general command the MNG will not acknowledge (NACK) this transaction, for a directed command the MNG will always acknowledge (ACK) this transaction.

This command will not affect the status or validity of the dynamic SMBus Address (will not clear the address Valid flag) nor of the Address Resolved flag.

The command returns the UDID bytes as defined in Section 3.2.1.1.7.

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	

10.4 Manageability Receive Filtering

10.4.1 Overview and General Structure

This section describes the manageability receive packet filtering flow. The description applies to each of the 82580 LAN ports. A packet that is received by the 82580 can have one of the following results:

- Discarded
- Sent to host memory
- Sent to the external BMC
- Sent to both the BMC and host memory

The decisions regarding forwarding of packets to the host and to the BMC are separate and are configured through two sets of registers. However, the BMC may define some types of traffic as exclusive. This traffic will be forwarded only to the BMC, even if it passes the filtering process of the host. These types of traffic are defined using the MNGONLY register.

The BMC controls the types of packets that it receives by programming the receive manageability filters. The software device driver can not write to the manageability filter registers. The table below lists the registers that are only written by the BMC.

Table 10-14. Registers Written by the BMC

Register	Functionality	When Reset
MANC	General configuration of the manageability filters.	LAN_PWR_GOOD and Firmware Reset
MNGONLY	Enables routing of packets exclusively to the manageability.	LAN_PWR_GOOD and Firmware Reset
MDEF[7:0] MDEF_EXT[7:0]	Configuration of manageability decision filters	LAN_PWR_GOOD and Firmware Reset
MMAH[1:0], MMAL[1:0]	2 Unicast MAC manageability addresses	LAN_PWR_GOOD
MAVTV[7:0]	Eight VLAN tag values	LAN_PWR_GOOD
MFUTP[3:0]	8 UDP/TCP destination port values	LAN_PWR_GOOD
FTFT	Values and mask and length for the 1 flex TCO filter	LAN_PWR_GOOD
MIPAF	IP address for manageability filtering	LAN_PWR_GOOD
METF	L2 EtherType values	LAN_PWR_GOOD

These registers are reset only on LAN_PWR_GOOD. In SMBus PT mode, registers that enable filters or functionality are loaded from the EEPROM following a firmware reset. See Section 6.8 for description of their location in the EEPROM map.

The high-level structure of manageability filtering is done using two steps.

1. The packet is parsed and fields in the header are compared to programmed filters.
2. A set of decision filters are applied to the result of the first step.

Some general rules:

- Fragmented packets are passed to manageability but not parsed beyond the IP header.
- Packets with L2 errors (CRC, alignment, etc.) are never forwarded to manageability.

The following sections describe the manageability filtering, followed by the final filtering rules.

The filtering rules are created by programming the decision filters as described in Section 10.4.4.

10.4.2 L2 Filters

10.4.2.1 MAC and VLAN filters

The manageability MAC filters allow comparison of the Destination MAC address to one of 2 filters defined in the *MMAH* and *MMAL* registers.

The VLAN filters allow comparison of the 12 bit VLAN tag to one of 8 filters defined in the *MAVTV* registers.

10.4.2.2 EtherType Filters

The manageability L2 EtherType filters allow filtering of receive packets based on the Layer 2 EtherType field. The L2 type field of incoming packets is compared against the EtherType filters programmed in the *METF.EType* (up to 4 filters) and the result is incorporated to the decision filters.

Each of the manageability EtherType filters can be configured as pass (“positive”) or reject (“negative”) polarity. When there is no match on its EtherType, and the polarity bit is set for a filter, that filter is enabled so as to participate in the decision process.

Note: In order for the reverse polarity mode to be effective and block certain type of packets, the ethertype filter should be part of all the enabled decision filters.

One example for usage of the L2 EtherType filters is:

- Block routing of packets with the NC-SI EtherType from being routed to the management controller. The NC-SI EtherType is used for communications between the management controller on the NC-SI link and the 82580. Packets coming from the network are not expected to carry this EtherType and such packets are blocked to prevent attacks on the management controller.

10.4.3 L3 and L4 Filters

10.4.3.1 ARP Filtering

The 82580 supports filtering of both ARP request packets (initiated externally) and ARP responses (to requests initiated by the BMC or the 82580).

10.4.3.2 Neighbor Discovery Filtering

The 82580 supports filtering of neighbor Discovery packets. neighbor Discovery uses the IPV6 destination address filters defined in the MIPAF registers (all enabled IPv6 addresses are matched for neighbor Discovery).

10.4.3.3 RMCP Port Filtering

The 82580 supports reception of RMCP packets by enabling filtering of the fixed destination ports numbers, port 0x26F and port 0x298.

10.4.3.4 Flex Port Filtering

The 82580 implements 8 flex UDP/TCP destination port filters. The 82580 directs packets whose L4 destination port matches the value of the respective word in the *MFUTP* registers. The BMC must ensure that only valid entries are enabled in the decision filters below.

10.4.3.5 Flex TCO Filters

The 82580 provides 1 flex TCO filter. The filter looks for a pattern match within the 1st 128 bytes of the packet. The BMC configures the pattern to match into the FTFT table. The BMC must ensure that only valid entries are enabled in the decision filters).

Note: The flex filter is temporarily disabled when read or written by the host. Any packet received during a read or write operation is dropped. Filter operation resumes once the read or write access completes.

10.4.3.6 IP Address Filtering

The 82580 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 82580 provides four IPv6 address filters.
- EN_IPv4_FILTER = 1b: the 82580 provides three IPv6 address filters and four IPv4 address filters.

10.4.3.7 Checksum Filter

If bit MANC.EN_XSUM_FILTER is set, the 82580 directs packets to the BMC only if they match all other filters previously described as well as pass L3/L4 checksum (if it exists).

10.4.4 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. The filtering rule defines which of the manageability filters participate in the decision. A packet that passes at least one rule is directed to manageability.

The manageability filters are controlled by the BMC only and not by the LAN driver. To filter network traffic according to a filtering rule, the BMC should define the filtering rule by programming the *MDEF* and *MDEF_EXT* registers and set the *MDEF_EXT.apply_to_network_traffic bit to 1, to enable the decision filter.*

The inputs to each decision filter are:

- Packet passed one of the management L2 unicast address filter.
- Packet is a broadcast packet.
- Packet has a VLAN header and it passed one of the manageability VLAN filters.
- Packet matched one of the IPv4 or IPv6 manageability address filters.
- Packet is a multicast packet.
- 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 passed or failed an L2 EtherType filter.
- Packet passed or failed Flow Control or NC-SI L2 EtherType Discard filter.

The structure of each of the decision filters is shown in Figure 10-4 . A boxed “x.y” number indicates that the input is conditioned on a mask bit “y” defined in the *MDEF* and *MDEF_EXT* registers for this rule (where x=0 denotes MDEF and x=1 denotes MDEF_EXT). The decision filter rules are as follows:

- All enabled AND filters must match for the decision filter to match. An AND filter not enabled in the MDEF/MDEF_EXT registers is ignored. In a set of AND filters from the same type, it is enough that one of the enabled filter match. The AND filter types are:
  - L2 address - includes the Unicast AND, Broadcast AND and Multicast AND filters.
  - VLAN - includes all the VLAN AND filters.
  - IPv4 - includes all the IPv4 AND filters.
  - IPv6 - includes all the IPv6 AND filters.
- If at least one OR filter is enabled in the respective *MDEF* and *MDEF_EXT* registers, then at least one of the enabled OR filters must match for the decision filter to match. Otherwise, the OR filters are ignored in the decision (the filter might still match).

A decision filter (for any of the 8 filters) defines which of the above inputs is enabled as part of the rule. The BMC programs two 32-bit registers per rule (*MDEF[7:0]* and *MDEF_EXT[7:0]*) with the settings as described in Section 8.22.6 and Section 8.22.7. A set bit allows its corresponding filter to participate in the filtering decision.

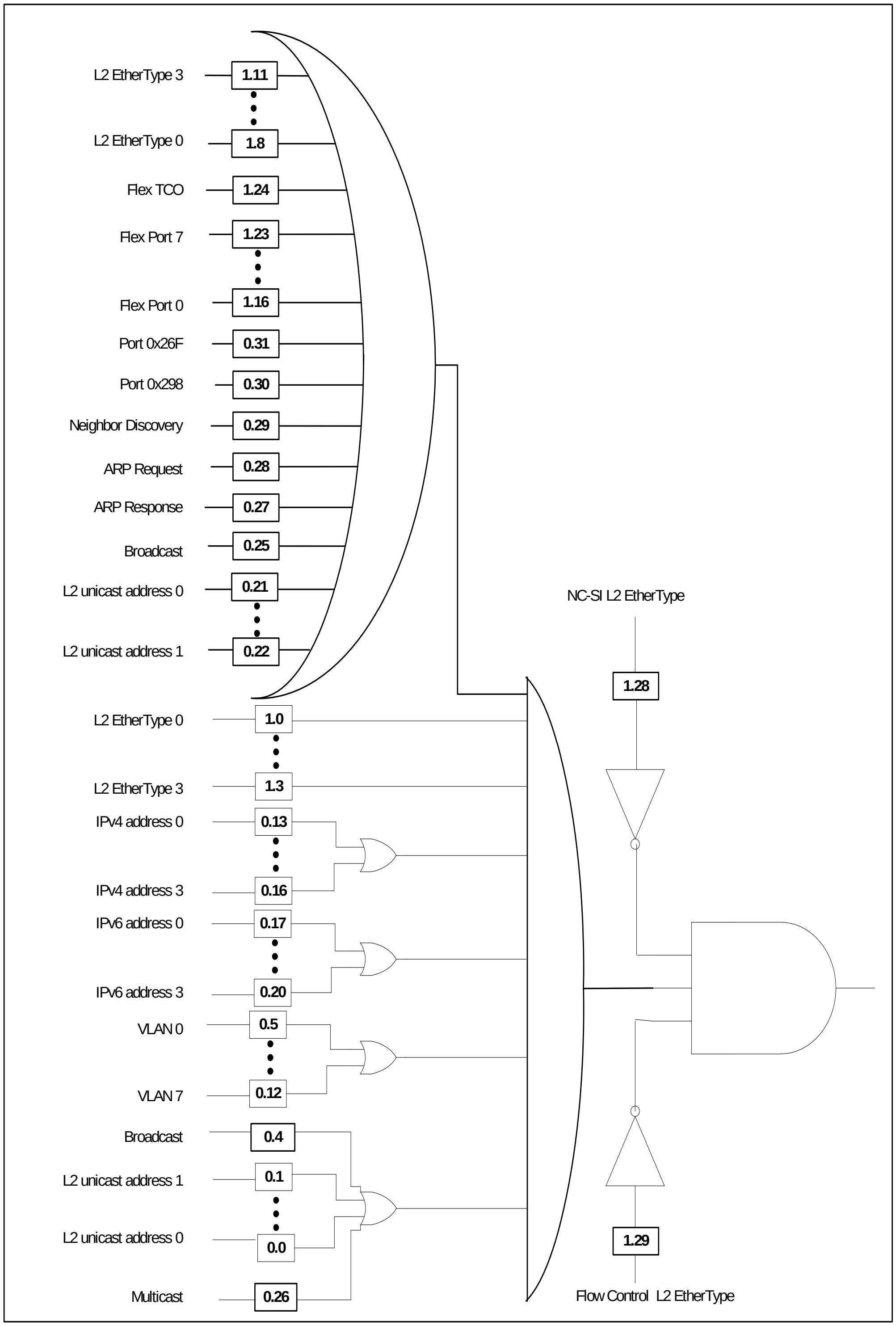


Figure 10-4. Manageability Decision Filters

10.4.4.1 Exclusive traffic

The decisions regarding forwarding of packets to the host for LAN traffic or to the LAN for host traffic are independent from the management decision filters. However, the BMC may define some types of traffic as exclusive. The behavior for such traffic is defined by the using the bits corresponding to the decision filter in the *MNGONLY* register (one bit per each of the eight decision rules) and the *MDEF_EXT.apply_to_network_traffic* bit. Table 10-15 describes the behavior in each case. If one or more filters match the traffic and at least one of the filters is set as exclusive, the traffic is treated as exclusive.

Table 10-15. Exclusive Traffic behavior

Filter match		Filter doesn't match
MNGONLY = 0	**MNGONLY = 1**	**N/A**
Traffic is forwarded to the manageability. Traffic is forwarded to the host according to host filtering	Traffic is forwarded only to manageability.	Traffic is forwarded to the host according to host filtering

10.4.5 Possible Configurations

This section describes possible ways of using the management filters. Actual usage may vary.

10.4.5.1 Dedicated MAC Packet Filtering

- Select one of the eight rules for broadcast filtering.
- Set bit 0 of the decision rule to enforce MAC address filtering.
- Set other bits to qualify which packets are allowed to pass through. For example:
  - Set bit 2 to qualify with manageability VLAN.
  - Set bit 3 to qualify with a match to an IP address.
  - Set any L3/L4 bits (30:7) to qualify with any of a set of L3/L4 filters.

10.4.5.2 Broadcast Packet Filtering

- Select one of the eight rules for broadcast filtering.
- Set bit 1 of the decision rule to enforce broadcast filtering.
- Set other bits to qualify which broadcast packets are allowed to pass through. For example:
  - Set bit 2 to qualify with manageability VLAN.
  - Set bit 3 to qualify with a match to an IP address.
  - Set any L3/L4 bits (30:7) to qualify with any of a set of L3/L4 filters.

10.4.5.3 VLAN Packet Filtering

- Select one of the eight rules for VLAN filtering.
- Set bit two of the decision rule to enforce VLAN filtering.
- Set other bits to qualify which VLAN packets are allowed to pass through. For example:
  - Set any L3/L4 bits (30:7) to qualify with any of a set of L3/L4 filters.

IPv6 filtering is done via the following IPv6-specific filters:

- IP Unicast filtering - requires filtering for Link Local address and a Global address. Filtering setup might depend on whether or not the MAC address is shared with the host or dedicated to manageability:
  - Dedicated MAC address (for example, dynamic address allocation with DHCP does not support multiple IP addresses for one MAC address). In this case, filtering can be done at L2 using two dedicated unicast MAC filters.
  - Shared MAC address (for example, static address allocation sharing addresses with host). In this case, filtering needs to be done at L3, requiring two IPv6 address filters, one per address.
- A neighbor Discovery filter - The 82580 supports IPv6 neighbor Discovery protocol. Since the protocol relies on multicast packets, the 82580 supports filtering of these packets. IPv6 multicast addresses are translated into corresponding Ethernet multicast addresses in the form of 33-33-xx-xx-xx-xx, where the last 32 bits of address are taken from the last 32 bits of the IPv6 multicast address. As a result, two direct MAC filters can be used to filter IPv6 solicited-node multicast packets as well as IPv6 all node multicast packets.

10.4.5.4 Receive Filtering with Shared IP - CPMP

When the BMC shares the MAC and IP address with the host, receive filtering is based mainly on identifying specific flows through port allocation. The following setting might be used:

- Select one of the eight rules.
- Set a manageability dedicated MAC filter to the host MAC address and set bit 0 in the MNG_ FILTER_RULE register.
- If VLAN is used for management, load one or more management VLAN filters and set bit 2 in the MNG_ FILTER_RULE register.
- ARP filter/neighbor Discovery filter is enabled when the BMC is responsible for handling the ARP protocol. Set bit 7 or bit 8 in the MNG_ FILTER_RULE register for this functionality.

§ §

NOTE: This page intentionally left blank.

11.0 Electrical/Mechanical Specification

11.1 Introduction

These specifications are subject to change without notice.

This chapter describes the 82580 DC and AC (timing) electrical characteristics. This includes absolute maximum rating, recommended operating conditions, power sequencing requirements, DC and AC timing specifications. The DC and AC characteristics include generic digital 3.3V IO specification as well as other specifications supported by the 82580.

11.2 Operating Conditions

Table 11-1. Absolute Maximum Ratings[1]

Symbol	Parameter	Min	Max	Units
T_{case}	Case Temperature Under Bias		100	°C
$T_{storage}$	Storage Temperature Range	–65	140	°C
Vi/Vo	3.3V Compatible I/Os Voltage Analog 1.0 I/O Voltage Analog 1.8 I/O Voltage	Vss – 0.5 Vss – 0.2 Vss – 0.3	4.6 1.68 2.52	V
VCC3P3	3.3V Periphery DC Supply Voltage	Vss – 0.5	4.6	V
VCC	1.0V Core DC Supply Voltage	Vss – 0.2	1.68V	V
VCC1P8	1.8V Analog DC Supply Voltage	Vss – 0.3	2.52	V
VCC1P0	1.0V Analog DC Supply Voltage	Vss – 0.2	1.68V	V

1. 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.

11.2.1 Recommended Operating Conditions

Table 11-2. Recommended Operating Conditions

Symbol	Parameter	Min	Max	Units	Notes
Ta	Operating Temperature Range Commercial (Ambient; 0 CFS airflow)	-10	55 85	°C	1,2,3

Note:

1. For normal device operation, adhere to the limits in this table. Sustained operations of a device at conditions exceeding these values, even if they are within the absolute maximum rating limits, may result in permanent device damage or impaired device reliability. Device functionality to stated DC and AC limits is not guaranteed if conditions exceed recommended operating conditions.
2. Recommended operation conditions require accuracy of power supply as defined in Section 11.3.1 .
3. With external heat sink. Airflow required for operation in 85′C ambient temperature.

11.3 Power Delivery

11.3.1 Power Supply Specification

VCC3P3 (3.3V) Parameters

Title	Description	Min	Max	Units
Rise Time	Time from 10% to 90% mark	0.1	100	mS
Monotonicity	Voltage dip allowed in ramp	N/A	0	mV
Slope	Ramp rate at any given time between 10% and 90% Min: 0.8*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
Ripple[1]	Maximum voltage ripple (peak to peak)	N/A	70	mV
Overshoot	Maximum overshoot allowed	N/A	100	mV
Overshoot Settling Time	Maximum overshoot allowed duration. (At that time delta voltage should be lower than 5mv from steady state voltage)	N/A	0.05	mS
Decoupling Capacitance	Capacitance range	15		µF
Capacitance ESR	Equivalent series resistance of output capacitance	N/A	50	mΩ

VCC1P8 (1.8V) Parameters

Title	Description	Min	Max	Units
Rise Time	Time from 10% to 90% mark	0.1	100	mS
Monotonicity	Voltage dip allowed in ramp	N/A	0	mV
Slope	Ramp rate at any given time between 10% and 90% Min: 0.8*V(min)/Rise time (max) Max: 0.8*V(max)/Rise time (min)	14	60000	V/S
Operational Range	Voltage range for normal operating conditions	1.71	1.89	V
Ripple[1]	Maximum voltage ripple (peak to peak)	N/A	40	mV
Overshoot	Maximum overshoot allowed	N/A	100	mV
Overshoot Settling Time	Maximum overshoot allowed duration. (At that time delta voltage should be lower than 5mv from steady state voltage)	N/A	0.1	mS
Decoupling Capacitance	Capacitance range	15	25	µF
Capacitance ESR	Equivalent series resistance of output capacitance	N/A	50	mΩ

VCC1P0 (1.0V) Parameters

Title	Description	Min	Max	Units
Rise Time	Time from 10% to 90% mark	0.1	80	mS
Monotonicity	Voltage dip allowed in ramp	N/A	0	mV
Slope	Ramp rate at any given time between 10% and 90% Min: 0.8*V(min)/Rise time (max) Max: 0.8*V(max)/Rise time (min)	7.6	33600	V/S
Operational Range	Voltage range for normal operating conditions	0.93	1.08	V

Ripple[1]	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
Decoupling Capacitance	Capacitance range	15	25	µF
Capacitance ESR	Equivalent series resistance of output capacitance	5	50	mΩ

[1] Power supply voltage with ripple should not be below minimum power supply operating range.

11.3.1.1 Power On/Off sequence

On power-on, after 3.3V reaches 90% of its final value, all voltage rails (1.8V and 1.0V) are allowed 30ms to reach their final operating values. However, to keep leakage current at a minimum, it is recommended to turn on power supplies almost simultaneously (with delay between supplies at most a few milliseconds).

For power-down, it is recommended to turn off all power rails at the same time and let power supply voltage decay.

Table 11-3. Power Sequencing

Symbol	Parameter	Min	Max	units
T_{3_1}	VCC3P3 (3.3V) stable to 1.0V stable		100	ms
T_{3_18}	VCC3P3 (3.3V) stable to 1.8V stable		100	ms
T_{18_1}	VCC1P8 (1.8V) stable to 1.0V stable	0		ms
Tm-per	3.3V power supply to PE_RST_N de-assertion[1]	100		ms
Tm-ppo	3.3V power supply to MAIN_PWR_OK assertion	0		ms
Tlpg	Power Supplies Stable to LAN_PWR_GOOD assertion	0		ms
Tlpg-per	LAN_PWR_GOOD assertion to PE_RST_N de-assertion[1]	100		ms
Tper-m, Tppo-m	PE_RST_N, MAIN_PWR_OK off before 3.3V power supply down	0		ms
Tlpgw	LAN_PWR_GOOD de-assertion time[2]	1		ms

[1] If external LAN_PWR_GOOD is used, this time should be kept between LAN_PWR_GOOD assertion and PERST# de-assertion.

[2] parameter relevant only if external LAN_PWR_GOOD used.

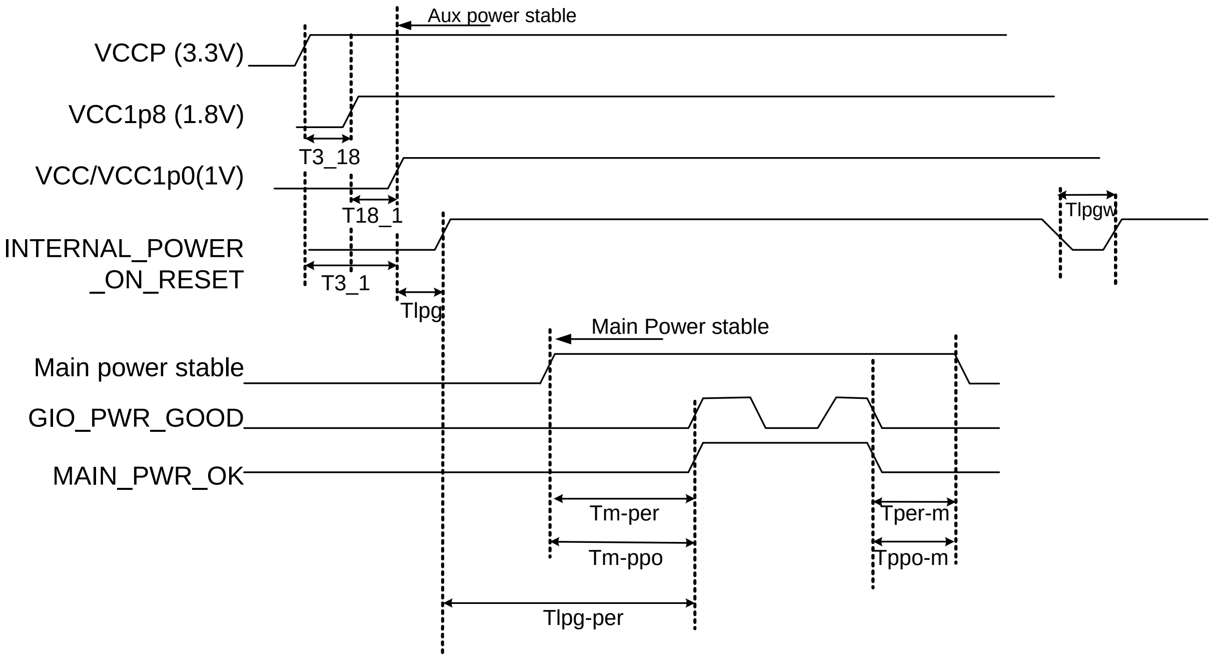


Figure 11-1. Power and Reset Sequencing

11.3.1.2 Power-On Reset Thresholds

the 82580 internal Power-on Reset circuitry initiates a full chip reset when voltage levels of VCC3P3 and VCC1P0 power supplies are below certain thresholds. To avoid false power-on reset following power-up voltage levels that trigger internal power-on reset when power supplies ramp-up and ramp-down differ.

Table 11-4. Power-on Reset Thresholds

Symbol	Parameter	Specifications			Units
		Min	Typ	Max	
V1a	Threshold for 3.3 V power supply in power-up	2.1		2.7	V
V2a	Threshold for 3.3 V power supply in power-down[1]	2.0		2.6	V
V1b	Threshold for 1.0 V power supply in power-up	0.49		0.87	V

[1] On power-down only the 3.3V threshold level (V2a) is used as a criteria for internal power-on reset assertion.

11.4 Ball Summary

See Chapter 2.0 for balls description and ball out map.

11.5 Current consumption

Table 11-5. Power Consumption 4 ports

Condition	Speed (MBps)		3.3V (mA)	1.8V (mA)	1.0V (mA)	Total power (mW)
D0a - Active Link	10	Typ	300	150	1317	2577
	100	Typ	202	150	1360	2297
	1000 Copper	Typ	335	150	1974	3350
		Max	340	152	2528	3924
	1000 Fiber	Typ	152	150	1324	2096
		Max	162	152	1810	2618
D0a - Idle Link	No link	Typ	49	147	840	1266
	10	Typ	185	147	978	1853
	100	Typ	202	147	1025	1956
	1000 Copper	Typ	339	147	1565	2948
	1000 Fiber	Typ	151	147	930	1693
D3cold - wake-up enabled on 4 ports	No link	Typ	44	2	434	583
	10	Typ	180	2	548	1146
	100	Typ	196	2	589	1239
D3cold - wake-up enabled on 1 port only	No link	Typ	45	2	365	517
	10	Typ	82	2	394	668
	100	Typ	86	2	405	692
D3cold-wake disabled (PCIe L3)	No Link	Max	45	2	365	517
D0 Uninitialized - Disabled through LAN_DIS_N	No Link	Typ	47	2	343	502
D0 Uninitialized Disabled through DEV_OFF_N	No Link	Typ	44	2	394	543

Notes: Typical conditions: room temperature (TA) = 25 C, nominal voltages and continuous network traffic at link speed at full duplex.

Maximum conditions: maximum operating temperature (TJ) values, Nominal voltage values and continuous network traffic at link speed at full duplex.

PCIe Configured to x4 Gen2 operation.

Table 11-6. Power Consumption 2 Ports

Condition	Speed (MBps)		3.3V (mA)	1.8V (mA)	1.0V (mA)	Total power (mW)
D0a - Active Link	10	Typ	177	91	931	1679
	100	Typ	128	91	952	1538
	1000 Copper	Typ	194	91	1268	2072
		Max	199	95	1860	2688
	1000 Fiber	Typ	105	91	936	1446
		Max	117	95	1450	2007

Condition	Speed (MBps)		3.3V (mA)	1.8V (mA)	1.0V (mA)	Total power (mW)
D0a - Idle Link	No link	Typ	50	91	836	1165
	10	Typ	119	91	905	1462
	100	Typ	128	91	927	1513
	1000 Copper	Typ	197	91	1199	2013
	1000 Fiber	Typ	105	91	879	1389
D3cold - wake-up enabled on 2 ports	No link	Typ	45	2	390	542
	10	Typ	114	2	447	827
	100	Typ	123	2	466	876
D3cold - wake-up enabled on 1 port only	No link	Typ	45	2	365	517
	10	Typ	82	2	394	668
	100	Typ	86	2	405	692
D3cold-wake disabled (PCIe L3)	No Link	Max	45	2	365	517
D0 Uninitialized - Disabled through LAN_DIS_N	No Link	Typ	46	2	347	502
D0 Uninitialized Disabled through DEV_OFF_N	No Link	Typ	44	2	394	543

Notes: Typical conditions: room temperature (TA) = 25 C, nominal voltages and continuous network traffic at link speed at full duplex.

Maximum conditions: maximum operating temperature (TJ) values, Nominal voltage values and continuous network traffic at link speed at full duplex.

PCIe configured to x2 Gen2 operation.

Table 11-7. Power Consumption 2 Ports GbE and 2 Ports SerDes

Condition	Speed (MBps)		3.3V (mA)	1.8V (mA)	1.0V (mA)	Total power (mW)
D0a - Active Link	10 Copper + 1000 Fiber	Typ	232	150	1319	2355
	100 Copper + 1000 Fiber	Typ	183	150	1340	2214
	1000 Copper + 1000 Fiber	Typ	249	150	1653	2745
		Max	260	152	2215	3347
D0a - Idle Link	No link	Typ	105	139	880	1477
	10 Copper + 1000 Fiber	Typ	175	140	955	1785
	100 Copper + 1000 Fiber	Typ	183	140	975	1831
	1000 Copper + 1000 Fiber	Typ	252	140	1248	2332
D3cold - wake-up enabled on 4 ports	No link	Typ	99	2	478	808
	10 Copper + 1000 Fiber	Typ	169	2	537	1098
	100 Copper + 1000 Fiber	Typ	177	2	556	1144
D3cold - wake-up enabled on 1 Cu port only	No link	Typ	45	2	365	517
	10 Copper	Typ	82	2	394	668
	100 Copper	Typ	86	2	405	692
D3cold-wake disabled	No Link	Max	45	2	365	517
D0 Uninitialized - Disabled through LAN_DIS_N	No Link	Typ	47	2	343	502
D0 Uninitialized Disabled through DEV_OFF_N	No Link	Typ	44	2	394	543

Notes: Typical conditions: room temperature (TA) = 25 C, nominal voltages and continuous network traffic at link speed at full duplex.

Maximum conditions: maximum operating temperature (TJ) values, Nominal voltage values and continuous network traffic at link speed at full duplex.

PCIe Configured to x4 Gen2 operation.

11.6 DC/AC specification

11.6.1 DC specifications

11.6.1.1 Digital I/O

Table 11-8. Digital IO DC Electrical Characteristics (Note 3)

Symbol	Parameter	Conditions	Min	Max	Units	Note
VCC3P3	Periphery supply		3.0	3.6	V	
VCC	Core supply		0.93	1.08	V	
VOH	Output High Voltage	IOH = -8mA; VCC3P3 = Min	2.4		V	
		IOH = -100µA; VCC3P3 = Min	VCC3P3–0.2			
VOL	Output Low Voltage	IOL = 8mA; VCC=Min		0.4	V	
		IOL = 100µA; VCC=Min		0.2	V	
VIH	Input High Voltage		2.0	VCC3P3 + 0.3	V	1
VIL	Input Low Voltage		-0.3	0.8	V	1
Iil	Input Current	VCC3P3 = Max; VI =3.6V/GND		+/- 10	µA	
PU	Internal pullup	VIL = 0V	40	150	kΩ	2
Hys	Built-in hysteresis		100		mV	
Cin	Input Pin Capacitance			8	pF	
Vos	Overshoot		N/A	4	V	
Vus	Undershoot		N/A	-0.4	V	

Notes:

1. The input buffer also has hysteresis > 100mV
2. Internal pullup Max characterized at slow corner (125C, VCC3P3=min, process slow); internal pullup Min characterized at fast corner (0C, VCC3P3=max, process fast).
3. Applies to PE_RSTn, SFP0_I2C_CLK, SFP1_I2C_CLK, SFP2_I2C_CLK, SFP3_I2C_CLK, SFP0_I2C_DATA, SFP1_I2C_DATA, SFP2_I2C_DATA, SFP3_I2C_DATA, SRDS_0_SIG_DET, SRDS_1_SIG_DET, SRDS_2_SIG_DET, SRDS_3_SIG_DET DEV_OFF_N, M_PWR_OK, JTCK, JTDI, JTDO, JTMS, SDP0[3:0],SDP1[3:0], SDP2[3:0], SDP3[3:0], FLSH_SI, FLSH_SO, FLSH_SCK, FLSH_CE_N, EE_DI, EE_DO, EE_SK, EE_CS_N.

11.6.1.2 LEDs I/O

Table 11-9. LED IO DC Electrical Characteristics

Symbol	Parameter	Conditions	Min	Max	Units	Note
VCC3P3	Periphery supply		3.0	3.6	V	
VCC	Core supply		0.93	1.08	V	
VOH	Output High Voltage	IOH = -16mA; VCC3P3 = Min	2.4		V	
VOL	Output Low Voltage	IOL = 16mA; VCC=Min		0.4	V	
VIH	Input High Voltage		2.0	VCC3P3 + 0.3	V	1
VIL	Input Low Voltage		-0.3	0.8	V	1
Iil	Input Current	VCC3P3 = Max; VI =3.6V/GND		+/- 20	µA	
Hys	Built-in hysteresis		100		mV	
Vos	Overshoot		N/A	4	V	
Vus	Undershoot		N/A	-0.4	V	

Notes:

1. The input buffer also has hysteresis > 100mV
2. Applies to LED0[3:0], LED1[3:0], LED2[3:0] and LED3[3:0]

11.6.1.3 Open Drain I/Os

Table 11-10. Open Drain DC Specifications (Note 1, 4)

Symbol	Parameter	Condition	Min	Max	Units	Note
VCC3P3	Periphery supply		3.0	3.6	V	
VCC	Core supply		0.93	1.08	V	
Vih	Input High Voltage		2.1		V	
Vil	Input Low Voltage			0.8	V	
Ileakage	Output Leakage Current	0 < Vin < VCC3P3		+/-10	µA	2
Vol	Output Low Voltage	@ Ipullup		0.4	V	4
Iol	Output Low Current	Vol=0.4V	6		mA	
Cin	Input Pin Capacitance			8	pF	3
Ioffsmb	Input leakage current	VCC3P3 off or floating		+/-10	µA	2

Notes:

1. Applies to SMBD, SMBCLK, SMBALRT _N, PE_WAKE_N pads.
2. Device meets this whether powered or not.
3. Characterized, not tested.
4. OD no high output drive. VOL max=0.4V at 6mA, VOL max=0.2V at 0.1mA

11.6.1.4 NC-SI Input and Output Pads

Table 11-11. NC-SI Pads DC Specifications

Symbol	Parameter	Conditions	Min	Max	Units
VCC3P3	Periphery supply		3.0	3.6	V
VCC	Core supply		0.93	1.08	V
Vabs	Signal voltage range		-0.3	3.765	V
VOH	Output High Voltage	IOH = -4mA; VCC3P3 = Min	2.4		V

Table 11-11. NC-SI Pads DC Specifications (Continued)

Symbol	Parameter	Conditions	Min	Max	Units
VOL	Output Low Voltage	IOL = 4mA; VCC3P3 = Min		0.4	V
VIH	Input High Voltage		2.0		V
VIL	Input Low Voltage			0.8	V
Vihyst	Input hysteresis		100		mV
Iil/Iih	Input Current	VCC3P3 = Max; Vin =3.6V/GND		20	µA
Cin	Input Capacitance			5	pF

Note: Applies to the NC-SI_CLK_OUT, NC-SI_CRS_DV, NC-SI_RXD[1:0] - input/output pads and NC-SI_TX_EN, NC-SI_TXD[1:0], NC-SI_CLK_IN - input pads.

11.6.2 Digital I/F AC Specifications

11.6.2.1 Reset signals

The timing between the power up sequence and the different reset signals is described in Figure 11-3 and in Table 11-227.

11.6.2.1.1 LAN_PWR_GOOD

The 82580 uses an internal power on detection circuit in order to generate the iLAN_PWR_GOOD signal. Reset can also be implemented when the external power on detection circuit determines that the device is powered up and asserts the LAN_PWR_GOOD signal to reset the device.

11.6.2.2 SMBus

The following table indicates the timing guaranteed when the driver or the agent is performing the action. Where only a typical value is specified, the actual value will be within 2% of the value indicated.

Table 11-12. SMBus Timing Parameters (Master Mode)

Symbol	Parameter	Min	Typ	Max	Units
F_{SMB}	SMBus Frequency		84	100	kHz
T_{BUF}	Time between STOP and START condition driven by the 82580	4.7	6.56		µs
$T_{HD:STA}$	Hold time after Start Condition. After this period, the first clock is generated.	4	6.72		µs
$T_{SU:STA}$	Start Condition setup time	4.7			µs
$T_{SU:STO}$	Stop Condition setup time	4	6.88		µs
$T_{HD:DAT}$	Data hold time	0.3	0.48		µs
$T_{SU:DAT}$	Data setup time	0.25			µs
$T_{TIMEOUT}$	Detect SMBClk low timeout	26.2		31.5	ms
T_{LOW}	SMBClk low time	4.7	5.76		µs
T_{HIGH}	SMBClk high time	4	6.56		µs

The following table indicates the timing requirements of the 82580 when it is the receiver of the indicated signal.

Table 11-13. SMBus Timing Parameters (Slave Mode)

Symbol	Parameter	Min 100KHz[1]	Min 400KHz[2]	Max	Units
F_{SMB}	SMBus Frequency	10	10	400	kHz
T_{BUF}	Time between STOP and START condition driven by the 82580.	4.7	1.3		µs
$T_{HD:STA}$	Hold time after Start Condition. After this period, the first clock is generated.	4	0.6		µs
$T_{SU:STA}$	Start Condition setup time	4.7	0.6		µs
$T_{SU:STO}$	Stop Condition setup time	4	0.6		µs
$T_{HD:DAT}$	Data hold time	300	100		ns
$T_{SU:DAT}$	Data setup time	250	100		ns
T_{LOW}	SMBClk low time	4.7	1.3		µs
T_{HIGH}	SMBClk high time	4	0.6		µs

1. Specifications based on SMBus specification
2. Specifications based on I^{2}C specification for Fast-mode (400 KHz)

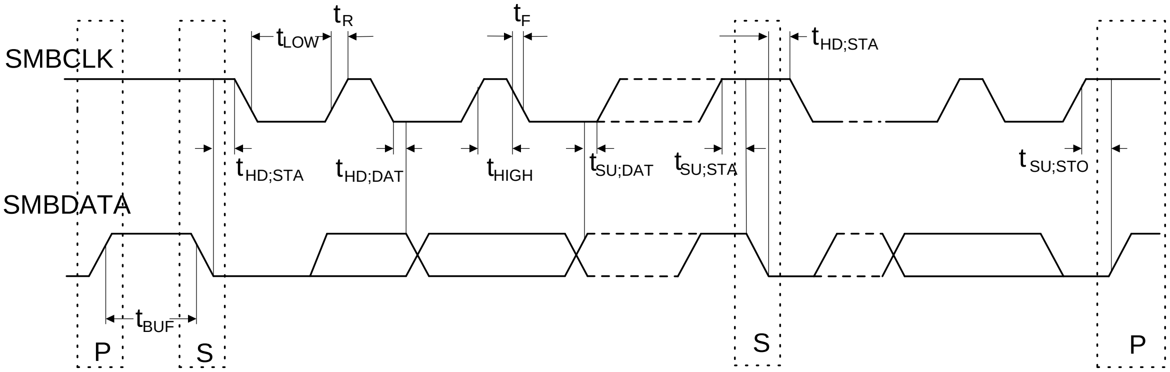


Figure 11-2. SMBus I/F Timing Diagram

11.6.2.3 I^{2}C AC Specification

The following table indicates the timing of the I2C_CLK and I2C_DATA pins when operating in I^{2}C mode.

Table 11-14. I^{2}C Timing Parameters

Symbol	Parameter	Min	Typ	Max	Units
F_{SCL}	I2C_CLK Frequency			100	kHz
T_{BUF}	Time between STOP and START condition driven by the 82580	4.7			µs
$T_{HD:STA}$	Hold time after Start Condition. After this period, the first clock is generated.	4			µs
$T_{SU:STA}$	Start Condition setup time	4.7			µs
$T_{SU:STO}$	Stop Condition setup time	4			µs
$T_{HD:DAT}$	Data hold time	0.3			µs

Table 11-14. I^{2}C Timing Parameters

Symbol	Parameter	Min	Typ	Max	Units
$T_{SU:DAT}$	Data setup time	0.25			µs
T_{LOW}	I2C_CLK low time	4.7			µs
T_{HIGH}	I2C_CLK high time	4			µs

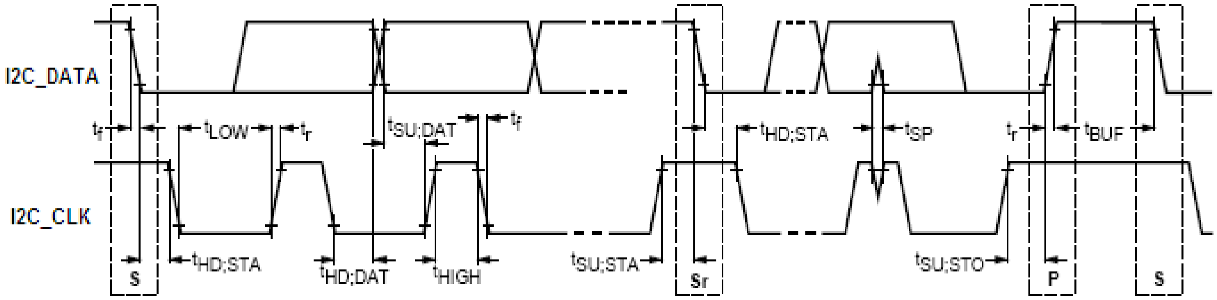


Figure 11-3. I^{2}C I/F Timing Diagram

11.6.2.4 FLASH AC specification

The 82580 is designed to support a serial flash. Applicable over the recommended operating range from Ta = -40C to +85C, VCC3P3 = 3.3V, Cload = 1 TTL Gate and 16 pF (unless otherwise noted). For FLASH I/F timing specification Table 11-239 and Figure 11-4.

Table 11-15. FLASH I/F Timing Parameters

Symbol	Parameter	Min	Typ	Max	Units	Note
t_{SCK}	SCK clock frequency	0	15.625	20	MHz	[1]
t_{RI}	Input rise time		2.5	20	ns	
t_{FI}	Input fall time		2.5	20	ns	
t_{WH}	SCK high time	20	32		ns	[2]
t_{WL}	SCK low time	20	32		ns	[2]
t_{CS}	CS high time	25			ns	
t_{CSS}	CS setup time	25			ns	
t_{CSH}	CS 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	

Notes:

1. Clock is 62.5MHz divided by 4. In Bit Banging mode maximum allowable frequency is 20MHz
2. 45% to 55% duty cycle.

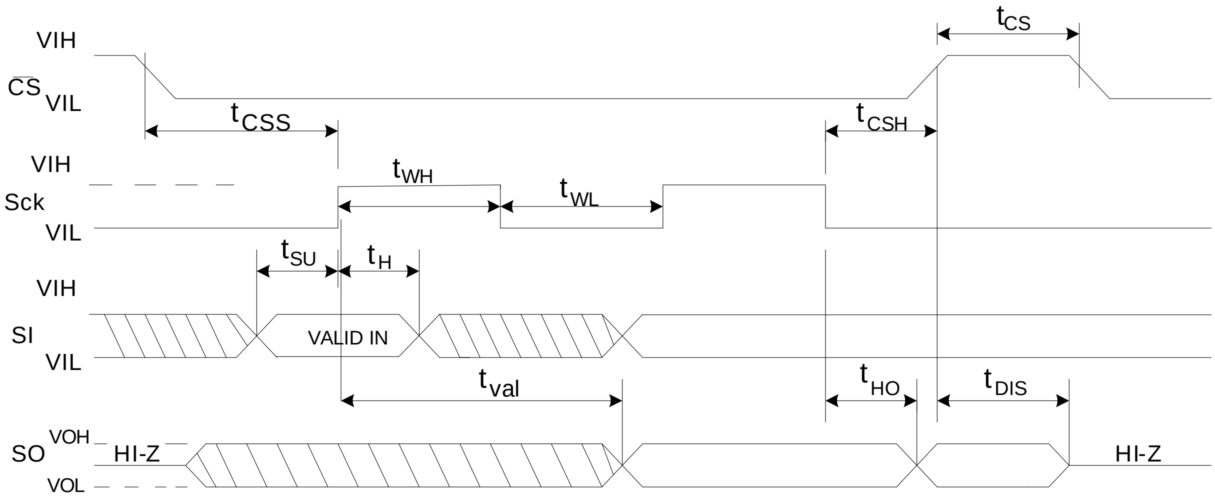


Figure 11-4. Flash Timing Diagram

11.6.2.5 EEPROM AC specification

The 82580 is designed to support a standard serial EEPROM. Applicable over recommended operating range from Ta = -40C to +85C, VCC3P3 = 3.3V, Cload = 1 TTL Gate and 16pF (unless otherwise noted). For EEPROM I/F timing specification see Table 11-240 and Figure 11-5.

Table 11-16. EEPROM I/F Timing Parameters

Symbol	Parameter	Min	Typ	Max	Units	Note
t_{SCK}	SCK clock frequency	0	2	2.1	MHz	[1]
t_{RI}	Input rise time			2	µs	
t_{FI}	Input fall time			2	µs	
t_{WH}	SCK high time	200	250		ns	[2]
t_{WL}	SCK low time	200	250		ns	
t_{CS}	CS high time	250			ns	
t_{CSS}	CS setup time	250			ns	
t_{CSH}	CS 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	

Notes:

1. Clock is 2MHz
2. 45% to 55% duty cycle.

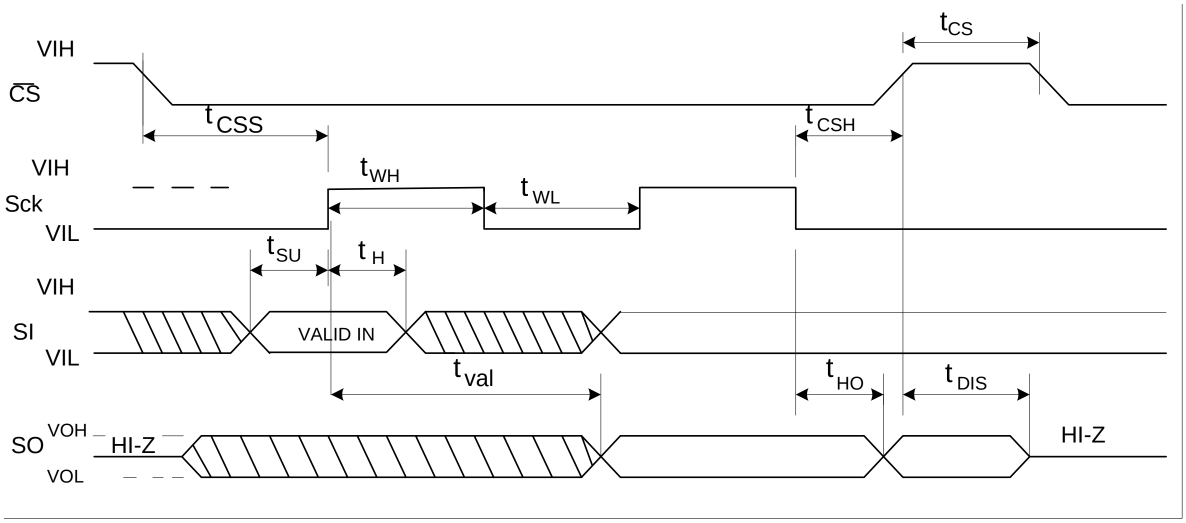


Figure 11-5. EEPROM Timing Diagram

11.6.2.6 NC-SI AC specification

The 82580 is designed to support the standard DMTF NC-SI interface. For NC-SI I/F timing specification see Table 11-241 and Figure 11-6.

Table 11-17. NC-SI AC Specifications

Symbol	Parameter	Min	Typ	Max	Units	Notes
Tckf	NCSI_CLK_IN Frequency		50		MHz	2
Rdc	NCSI_CLK_IN Duty Cycle	35		65	%	1
Racc	NCSI_CLK_IN accuracy			100	ppm	
Tco	Clock-to-out (10 pF =< cload <=50 pF) NCSI_RXD[1:0], NCSI_CRS_DV Data valid from NCSI_CLK_IN rising edge	2.5		12.5	ns	4
Tsu	NCSI_TXD[1:0], NCSI_TX_EN, Data Setup to NCSI_CLK_IN rising edge	3			ns	
Thold	NCSI_TXD[1:0], NCSI_TX_EN Data hold from NCSI_CLK_IN rising edge	1			ns	
Tor	NCSI_RXD[1:0], NCSI_CRS_DV Output Time rise	0.5		6	ns	3
Tof	NCSI_RXD[1:0], NCSI_CRS_DV Output Time fall	0.5		6	ns	3
Tckr/Tckf	NCSI_CLK_IN Rise/Fall Time	0.5		3.5	ns	
Tckor/Tckof	NCSI_CLK_OUT Rise/Fall Time	0.5		3.5	ns	5

Notes:

1. Clock Duty cycle measurement: High interval measured from Vih to Vil points, Low from Vil to next Vih.
2. Clock interval measurement from Vih to Vih.
3. Cload = 25 pF.
4. This timing relates to the output pins, while Tsu and Thd relate to timing at the input pins
5. 10 pF =< Cload <= 30 pF

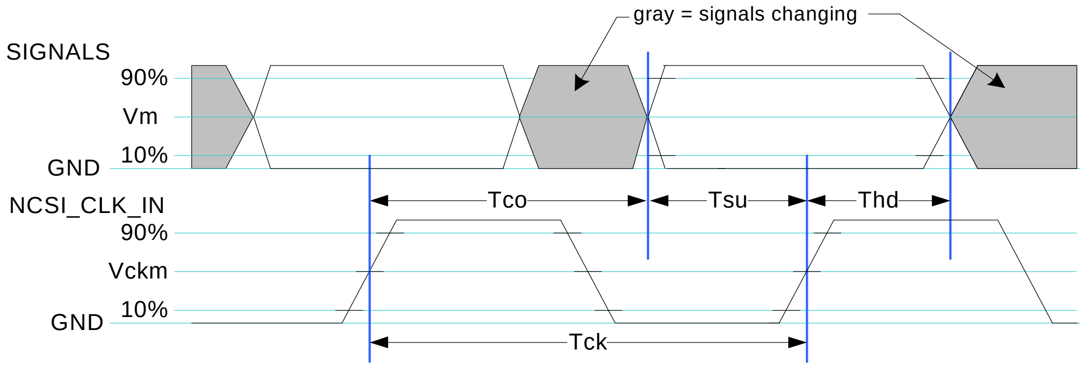


Figure 11-6. NC-SI Timing Diagram

11.6.2.7 JTAG AC specification

The 82580 is designed to support the IEEE 1149.1 standard. Following timing specifications are applicable over recommended operating range from Ta = 0°C to +70°C, VCC3P3 = 3.3V, Cload = 16pF (unless otherwise noted). For JTAG I/F timing specification see Table 11-242 and Figure 11-7.

Table 11-18. JTAG I/F Timing Parameters

Symbol	Parameter	Min	Typ	Max	Units	Note
t_{JCLK}	JTCK clock frequency			10	MHz	
t_{JH}	JTMS and JTDI hold time	10			nS	
t_{JSU}	JTMS and JTDI setup time	10			nS	
t_{JPR}	JTDO propagation Delay			15	nS	

Notes:

1. The table above applies to JTCK, JTMS, JTDI and JTDO.
2. Timing measured relative to JTCK reference voltage of VCC3P3/2.

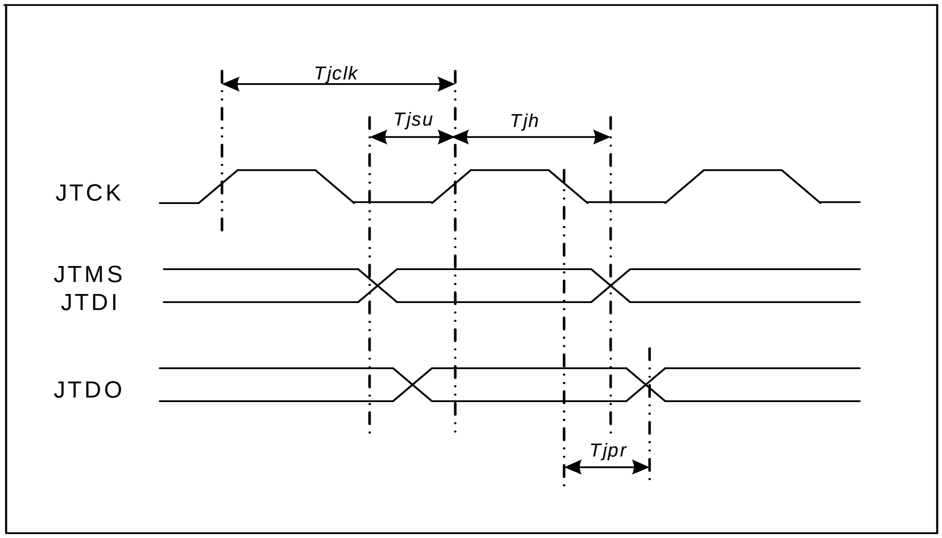


Figure 11-7. JTAG AC Timing Diagram

11.6.2.8 MDIO AC specification

The 82580 is designed to support the MDIO specifications defined in IEEE 802.3 clause 22. Following timing specifications are applicable over recommended operating range from Ta = 0°C to +70°C, VCC3P3 = 3.3V, Cload = 16pF (unless otherwise noted). For MDIO I/F timing specification see Table 11-243, Figure 11-8 and Figure 11-9.

Table 11-19. MDIO I/F Timing Parameters

Symbol	Parameter	Min	Typ	Max	Units	Note
t_{MCLK}	MDC clock frequency			2	MHz	
t_{MH}	MDIO hold time	10			nS	
t_{MSU}	MDIO setup time	10			nS	
t_{MPR}	MDIO propagation Delay	10		300	nS	

Notes:

1. The table above applies to MDIO0, MDC0, MDIO1, MDC1, MDIO2, MDC2, MDIO3, and MDC3.
2. Timing measured relative to MDC reference voltage of 2.0V (Vih).

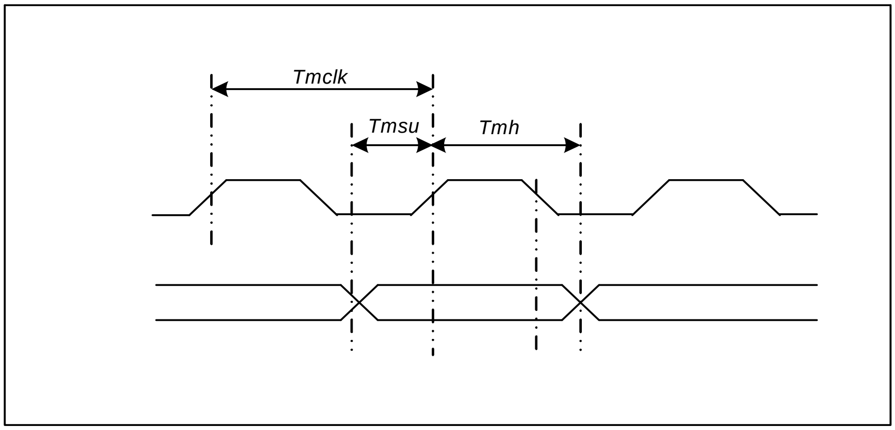


Figure 11-8. MDIO Input AC Timing Diagram

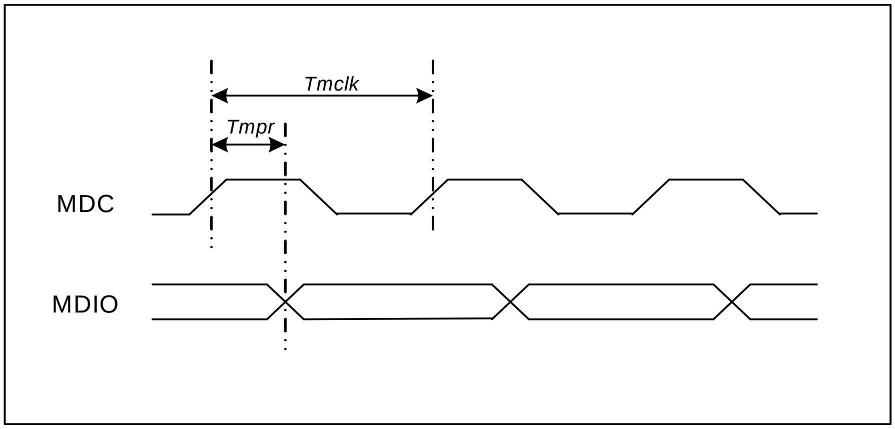


Figure 11-9. MDIO Output AC Timing Diagram

11.6.2.9 SFP 2 Wires I/F AC Specification

According to Atmel's AT24C01A/02/04 definition of the 2 wires I/F bus.

11.6.2.10 PCIe/SerDes DC/AC Specification

The transmitter and receiver specification are given per PCIe Card Electromechanical Specification rev 2.0.

11.6.2.11 PCIe Specification - Receiver

Specifications are from the PCIe v2.0 (5Gbps and 2.5Gbps) specification.

11.6.2.12 PCIe Specification - Transmitter

Specifications are from the PCI Express* 2.0 (5Gbps or 2.5Gbps) specification.

11.6.2.13 PCIe Specification - Input Clock

The input clock for PCIe must be a differential input clock in frequency of 100 MHz. For full specifications please check the PCI-Express Card Electromechanical specifications (refclk specifications).

11.6.3 Serdes DC/AC Specification

The Serdes interface supports the following standards:

1. PICMG 3.1 specification Rev 1.0 1000BASE-BX.
2. 1000BASE-KX electrical specification defined IEEE802.3ap clause 70.
3. SGMII on 1000BASE-BX or 1000BASE-KX compliant electrical interface (AC coupling with internal clock recovery).
4. SFP (Small Form factor Pluggable) Transceiver Rev 1.0

The specifications define the interface for the back-plane board connection, Interface to external 1000BASE-T PHY and the interface to fiber or SFP module.

11.6.4 PHY Specification

DC/AC specification is according to Standard 802.3 and 802.3ab.

100 Base-T parameters are also described in standard ANSI X3.263.

11.6.5 XTAL/Clock specification

The 25 MHz reference clock of the 82580 can be supplied either from a crystal or from an external oscillator. The recommended solution is to use a crystal.

11.6.5.1 Crystal Specification

Table 11-20. Specification for External Crystal

Parameter Name	Symbol	Recommended Value	Conditions
Frequency	f_o	25.000 [MHz]	@25 [°C]
Vibration mode		Fundamental	
Cut		AT	
Operating /Calibration Mode		Parallel	
Frequency Tolerance @25°C	$\Delta f/f_o$ @25°C	±30 [ppm]	@25 [°C]
Temperature Tolerance	$\Delta f/f_o$	±30 [ppm]	
Operating Temperature	T_{opr}	-20 to +70 [°C]	
Non Operating Temperature Range	T_{opr}	-40 to +90 [°C]	
Equivalent Series Resistance (ESR)	R_s	50 [Ω] maximum	@25 [MHz]
Shunt Capacitance	C_o	6 [pF] maximum	

Table 11-20. Specification for External Crystal (Continued)

Parameter Name	Symbol	Recommended Value	Conditions
Pullability from Nominal Load Capacitance	$\Delta f/C_{load}$	15 [ppm/pF] maximum	
Max Drive Level	D_L	0.5 [mW]	
Insulation Resistance	IR	500 [MΩ] minimum	@ 100V DC
Aging	$\Delta f/f_o$	±5 [ppm/year]	
Damp Capacitor	Cd	10 [pF]	
External Capacitors	C_1, C_2	36 [pF] to 40 [pF]	
Board Resistance	R_s	0.1 [Ω]	

11.6.5.2 External Clock Oscillator Specification

When using an external oscillator the following connection must be used:

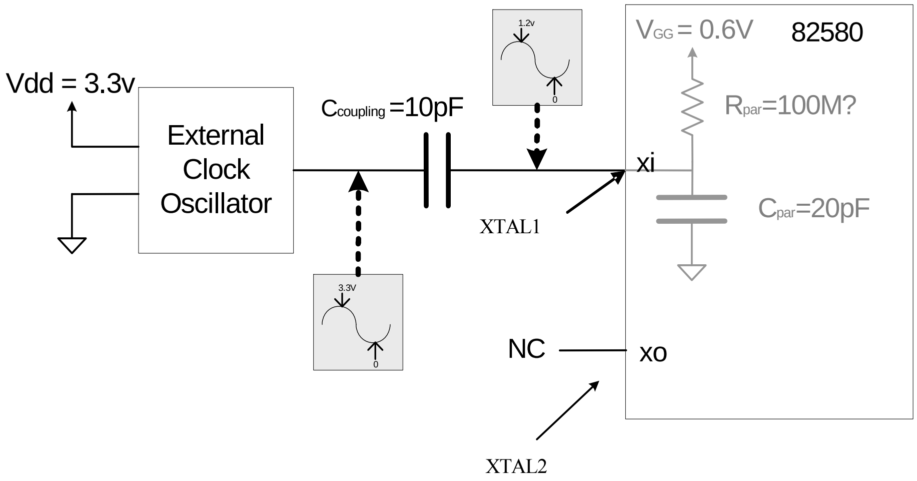


Figure 11-10. External Clock Oscillator Connectivity to the 82580

Table 11-21. Specification for External Clock Oscillator

Parameter Name	Symbol	Value	Conditions
Frequency	f_o	25.0 [MHz]	@25 [°C]
External OSC Supply Swing	V_{p-p}	3.3 ± 0.3 [V]	
Frequency Tolerance	$\Delta f/f_o$	±50 [ppm]	-20 to +70 [°C]
Operating Temperature	T_{opr}	-20 to +70 [°C]	
Aging	$\Delta f/f_o$	±5 ppm per year	

11.6.6 GbE PHY GE_REXT Bias Connection

For the PHY circuit, an external resistor of 3.01KΩ (accuracy 1%) is used as reference for the internal bias currents. This resistor is connected to the GE_REXT ball and GND as shown in Figure 11-11.

Short connections for this resistor are compulsory.

Place the resistor as close as possible to the device (less than 1").

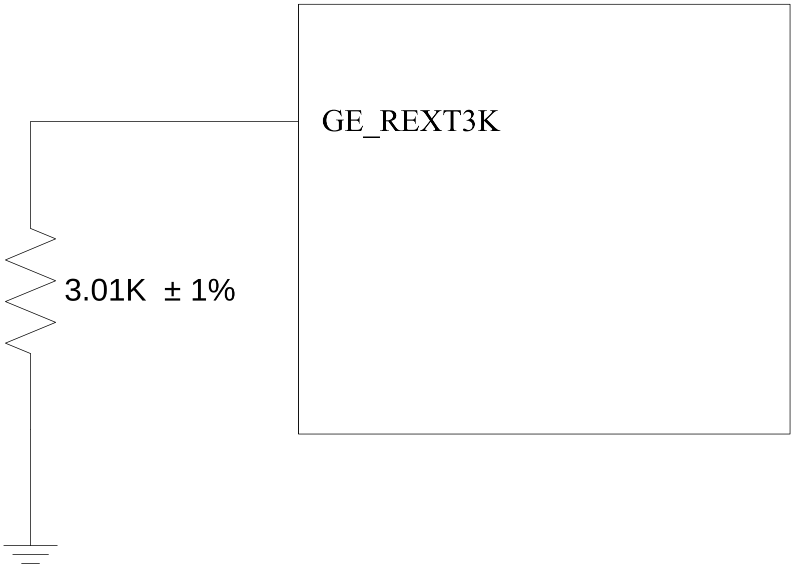


Figure 11-11. GbE PHY Bias Connection

11.6.7 SerDes SE_RSET Bias Connection

For the SerDes circuit, an external resistor of 2.37KΩ (accuracy 1%) is used as reference for the internal bias currents. This resistor is connected to the SE_RSET ball and GND as shown in Figure 11-12.

Short connections for this resistor are compulsory.

Place the resistor as close as possible to the device (less than 1").

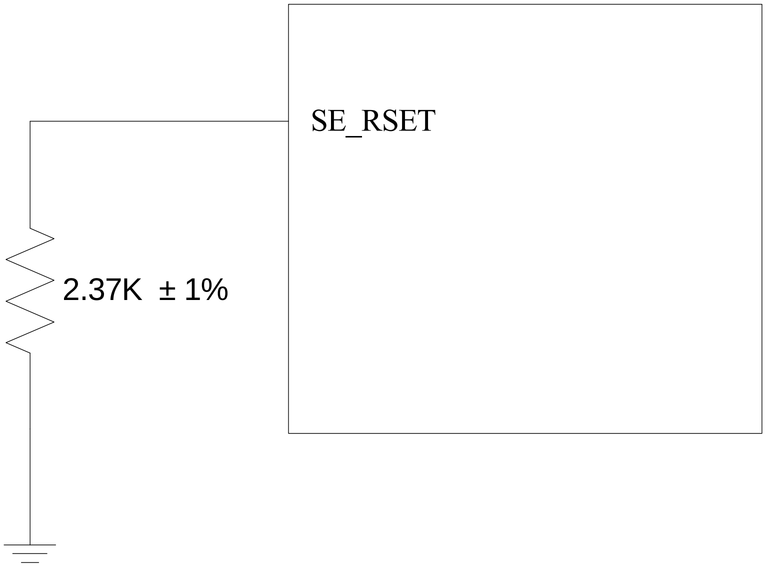


Figure 11-12. SerDes Bias Connection

11.6.8 PCIe PE_TRIM Bias Connection

For the PCIe SerDEs circuit, an external resistor of 1.5KΩ (accuracy 1%) is used as reference for the internal bias currents. This resistor is connected between the PE_TRIM1 and PE_TRIM2 balls as shown in Figure 11-13.

Short connections for this resistor are compulsory. Place the resistor as close as possible to the device (less than 1").

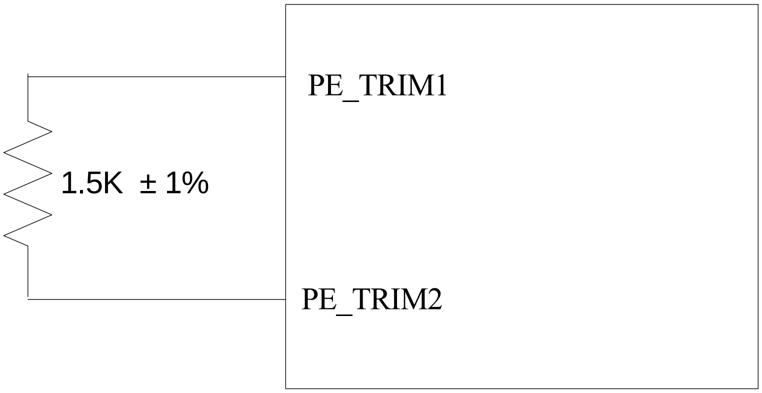


Figure 11-13. PCIe Bias Connection

11.7 Package

11.7.1 Mechanical

The 82580 is assembled into a 17x17 PBGA package.

Table 11-22. 82580 Package Mechanical Specifications

Body Size	Ball Count (Pin Count)	Ball Pitch	Ball Matrix	Substrate
17x17	256	1.0 mm	16 x 16 array, fully populated	4 layers

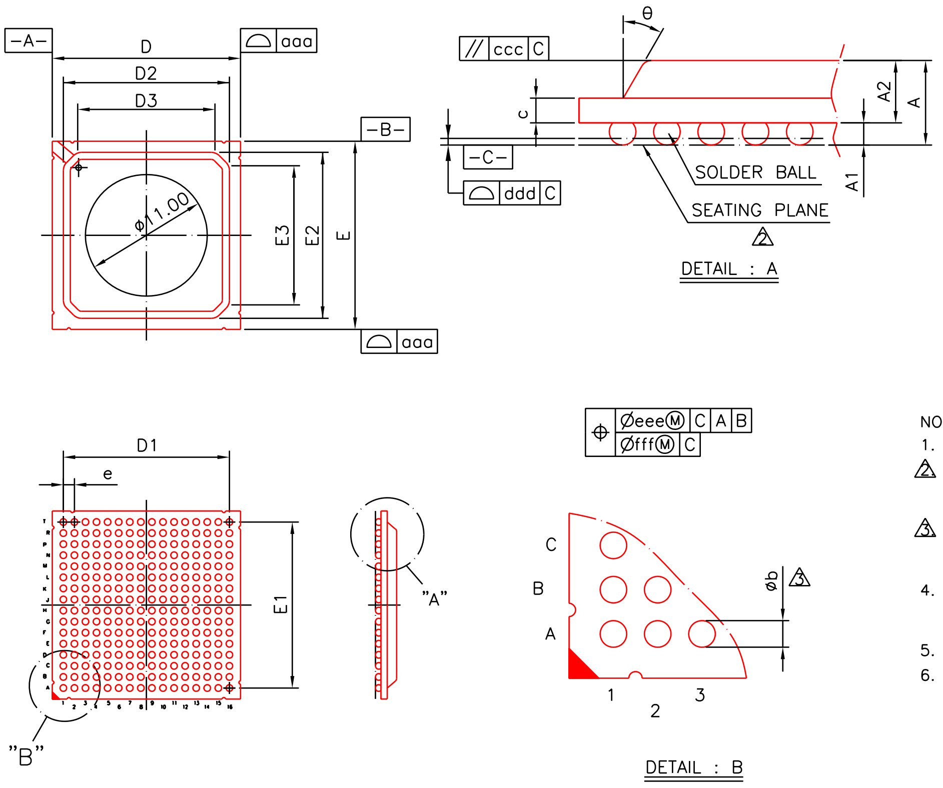
Note: The 82580 uses the P-free SAC305 solder ball (P<2 ppm) and WF6063M5 ball attach flux. The package pad copper size is 0.6 mm in diameter. The solder mask opening is 0.45 mm in diameter.

11.7.2 Thermal

See Section 13.0, Thermal Management .

11.7.3 Package Schematics

See the following diagram.



Symbol	Dimension in mm			Dimension in inch		
	MIN	NOM	MAX	MIN	NOM	MAX
A	---	1.81	2.01	---	0.071	0.079
A1	0.30	0.40	0.50	0.012	0.016	0.020
A2	1.31	1.41	1.51	0.052	0.056	0.059
b	0.40	0.50	0.60	0.016	0.020	0.024
c	0.51	0.56	0.61	0.020	0.022	0.024
D/E	16.80	17.00	17.20	0.661	0.670	0.677
D1/E1	---	15.00	---	---	0.591	---
D2/E2	14.90	15.00	15.10	0.587	0.591	0.594
D3/E3	---	12.57	---	---	0.495	---
e	---	1.00	---	---	0.039	---
aaa	0.20			0.008		
ccc	0.25			0.010		
ddd	0.15			0.006		
eee	0.25			0.010		
fff	0.10			0.004		
θ	30° TYP			30° TYP		
MD/ME	16/16			16/16		

NOTE :

1. CONTROLLING DIMENSION : MILLIMETER.
2. PRIMARY DATUM C AND SEATING PLANE ARE DEFINED BY THE SPHERICAL CROWNS OF THE SOLDER BALLS.
3. DIMENSION b IS MEASURED AT THE MAXIMUM SOLDER BALL DIAMETER, PARALLEL TO PRIMARY DATUM C.
4. THERE SHALL BE A MINIMUM CLEARANCE OF 0.25mm BETWEEN THE EDGE OF THE SOLDER BALL AND THE BODY EDGE.
5. REFERENCE DOCUMENT : JEDEC MO-192
6. SPECIAL CHARACTERISTICS C CLASS: ccc, ddd

11.8 EEPROM Flash Devices

While Intel does not make recommendations regarding these devices, the following devices have been used successfully in previous designs.

11.8.1 Flash

Type: SPI Flash

Size: 256 Kbytes (typical), depending on application.

Table 11-23. Serial Flash Table

Density	Intel PN	Atmel PN	STM PN	SST PN
512KBit		AT25F512N-10SI-2.7	M25P05-AVMN6T	SST25VF512A
1MBit		AT25F1024N-10SI-2.7	M25P10-AVMN6T	SST25VF010A
2MBit		AT25F2048N-10SI-2.7	M25P20-AVMN6T	SST25LF020A
4MBit		AT25F4096N-10SI-2.7	M25P40-AVMN6T	SST25VF040A
8MBit			M25P80-AVMN6T	SST25VF080A
16MBit	QB25F160S33T60 QB25F160S33B60 QH25F160S33T60 QH25F160S33B60 QB25F016S33T60 QB25F016S33B60 QH25F016S33T60 QH25F016S33B60		M25P16-AVMN6T	
32Mbit	QB25F320S33T60 QB25F320S33B60 QH25F320S33T60 QH25F320S33B60		M25P32-AVMN6T	

11.8.2 EEPROM

Table 11-24. EEPROM Devices

Density [Kbits]	Atmel PN	STM PN	OnSemi PN
128	AT25128AN-10SI-2.7	M95128WMN6T	CAT25CS128-TE13
256	AT25256AN-10SI-2.7	M95256WMN6T	

§ §

NOTE: This page intentionally left blank.

12.0 Design Guidelines

12.1 Ethernet Interface

12.1.1 Magnetics for 1000 BASE-T

Magnetics for the 82580 can be either integrated or discrete.

The magnetics module has a critical effect on overall IEEE and emissions conformance. The device should meet the performance required for a design with reasonable margin to allow for manufacturing variation. Occasionally, components that meet basic specifications can cause the system to fail IEEE testing because of interactions with other components or the printed circuit board itself. Carefully qualifying new magnetics modules prevents this problem.

When using discrete magnetics it is necessary to use ‘Bob Smith’ termination: Use four 75 Ω resistors for cable-side center taps and unused pins. This method terminates pair-to-pair common mode impedance of the CAT5 cable.

Use an EFT capacitor attached to the termination plane. Suggested values are 1500 pF/2 KV or 1000 pF/3 KV. A minimum of 50-mil spacing from capacitor to traces and components should be maintained.

12.1.2 Magnetics Module Qualification Steps

The steps involved in magnetics module qualification are similar to those for crystal qualification:

1. Verify that the vendor’s published specifications in the component datasheet meet or exceed specifications.
2. Independently measure the component’s electrical parameters on the test bench, checking samples from multiple lots. Check that the measured behavior is consistent from sample to sample and that measurements meet the published specifications.
3. Perform physical layer conformance testing and EMC (FCC and EN) testing in real systems. Vary temperature and voltage while performing system level tests.

12.1.3 Third-Party Magnetics Manufacturers

The following magnetics modules have been used successfully in previous designs.

Table 12-1. Magnetics Modules

Manufacturer	Part Number
Low Profile Discrete:	
Midcom Inc.	000-7412-35R-LF1
Standard Discrete:	
BelFuse	S558-5999-P3 (12-core)
Pulse Eng.	H5007NL (12-core)
Integrated:	
FOXCONN	JFM38U1C-L1U1W
Pulse Eng.	JW0-0013NL
Amphenol	RJMG2310 22830ER C03-002
BelFuse	0862-1J1T-Z4-F
Tyco	6368472-1

12.1.4 Layout Considerations for the Ethernet Interface

The following sections provide recommendations for performing printed circuit board layouts. Good layout practices are essential to meet IEEE PHY conformance specifications and EMI regulatory requirements.

Critical signal traces should be kept as short as possible to decrease the likelihood of being affected by high frequency noise from other signals, including noise carried on power and ground planes. Keeping the traces as short as possible can also reduce capacitive loading.

Since the transmission line medium extends onto the printed circuit board, special attention must be paid to layout and routing of the differential signal pairs.

Designing for 1000 BASE-T Gigabit operation is very similar to designing for 10 and 100 Mb/s. For the 82580, system level tests should be performed at all three speeds.

12.1.4.1 Guidelines for Component Placement

Component placement can affect signal quality, emissions, and component operating temperature This section provides guidelines for component placement.

Careful component placement can:

- Decrease potential problems directly related to electromagnetic interference (EMI), which could cause failure to meet applicable government test specifications.
- Simplify the task of routing traces. To some extent, component orientation will affect the complexity of trace routing. The overall objective is to minimize turns and crossovers between traces.

Minimizing the amount of space needed for the Ethernet LAN interface is important because other interfaces compete for physical space on a motherboard near the connector. The Ethernet LAN circuits need to be as close as possible to the connector.

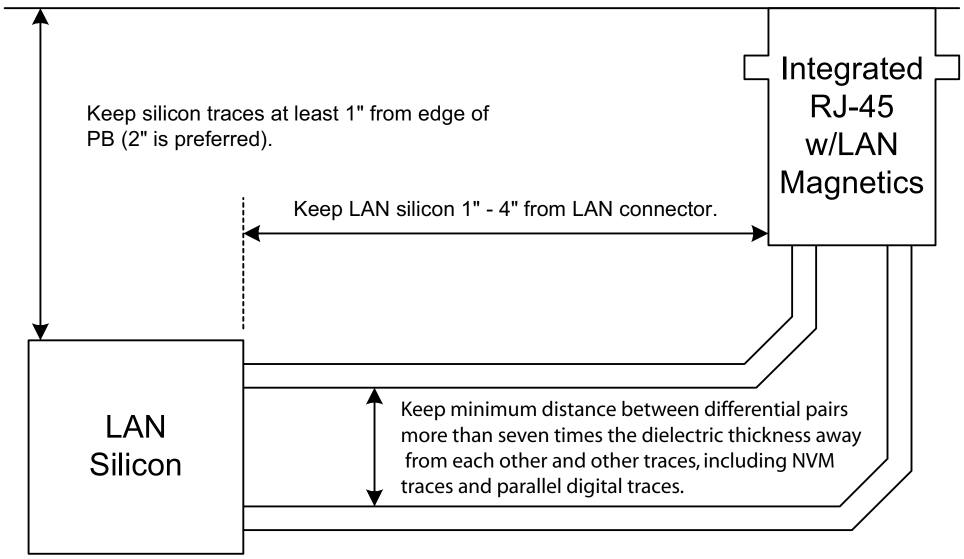

Integrated RJ-45 w/LAN Magnetics
Keep silicon traces at least 1" from edge of PB (2" is preferred).
Keep LAN silicon 1" - 4" from LAN connector.
LAN Silicon
Keep minimum distance between differential pairs more than seven times the dielectric thickness away from each other and other traces, including NVM traces and parallel digital traces.

Figure 12-1. General Placement Distances for 1000 BASE-T Designs

Figure 12-1 shows some basic placement distance guidelines. Figure 12-1 shows two differential pairs, but can be generalized for a Gigabit system with four analog pairs. The ideal placement for the Ethernet silicon would be approximately one inch behind the magnetics module.

While it is generally a good idea to minimize lengths and distances, Figure 12-1 also illustrates the need to keep the LAN silicon away from the edge of the board and the magnetics module for best EMI performance.

12.1.4.2 Layout Guidelines for Use with Integrated and Discrete Magnetics

Layout requirements are slightly different when using discrete magnetics.

These include:

- Ground cut for HV installation (not required for integrated magnetics)
- A maximum of two (2) vias
- Turns less than 45°
- Discrete terminators

Figure 12-2 shows a reference layout for discrete magnetics.

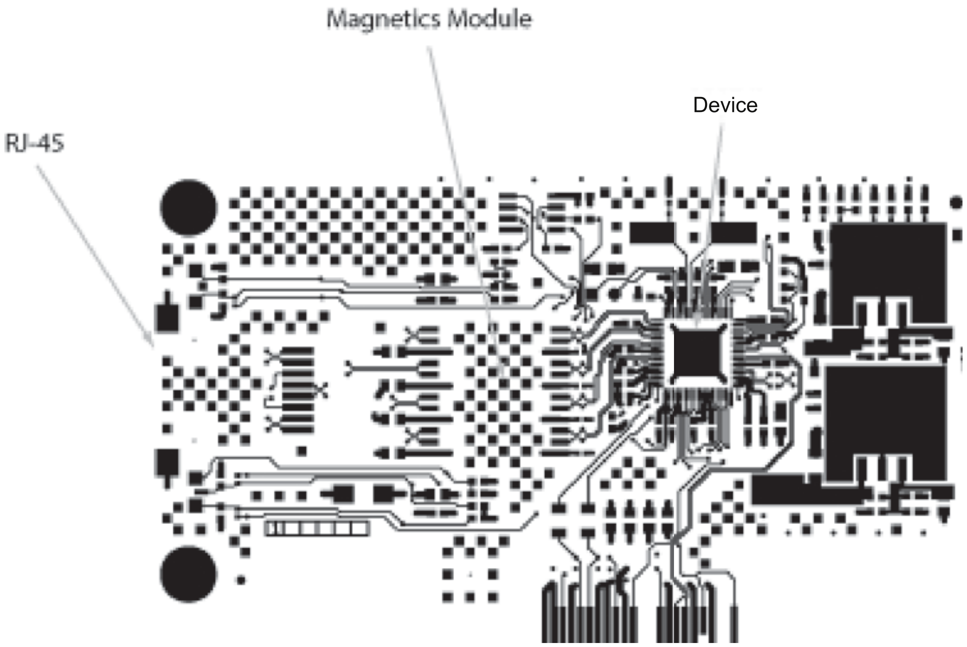


Figure 12-2. Layout for Discrete Magnetics

12.1.4.3 Board Stack-Up Recommendations

Printed circuit boards for these designs typically have four, six, eight, or more layers. Although, the 82580 does not dictate the stack up, here is an example of a typical six-layer board stack up:

- Layer 1 is a signal layer. It can contain the differential analog pairs from the Ethernet device to the magnetics module, or to an optical transceiver.
- Layer 2 is a signal ground layer. Chassis ground may also be fabricated in Layer 2 under the connector side of the magnetics module.
- Layer 3 is used for power planes.
- Layer 4 is a signal layer.
- Layer 5 is an additional ground layer.
- Layer 6 is a signal layer. For 1000 BASE-T (copper) Gigabit designs, it is common to route two of the differential pairs (per port) on this layer.

This board stack up configuration can be adjusted to conform to specific OEM design rules.

12.1.4.4 Differential Pair Trace Routing for 10/100/1000 Designs

Trace routing considerations are important to minimize the effects of crosstalk and propagation delays on sections of the board where high-speed signals exist. Signal traces should be kept as short as possible to decrease interference from other signals, including those propagated through power and ground planes. Observe the following suggestions to help optimize board performance:

- Maintain constant symmetry and spacing between the traces within a differential pair.
- Minimize the difference in signal trace lengths of a differential pair.
- Keep the total length of each differential pair under 4 inches. Although possible, designs with differential traces longer than 5 inches are much more likely to have degraded receive BER (Bit Error Rate) performance, IEEE PHY conformance failures, and/or excessive EMI (Electromagnetic Interference) radiation.
- Keep differential pairs more than seven times the dielectric thickness away from each other and other traces, including NVM traces and parallel digital traces.
- Keep maximum separation within differential pairs to 7 mils.
- For high-speed signals, the number of corners and vias should be kept to a minimum. If a 90° bend is required, it is recommended to use two 45° bends instead. Refer to Figure 12-3.

Note: In manufacturing, vias are required for testing and troubleshooting purposes. The via size should be a 17-mil (±2 mils for manufacturing variance) finished hole size (FHS).

- Traces should be routed away from board edges by a distance greater than the trace height above the reference plane. This allows the field around the trace to couple more easily to the ground plane rather than to adjacent wires or boards.
- Do not route traces and vias under crystals or oscillators. This will prevent coupling to or from the clock. And as a general rule, place traces from clocks and drives at a minimum distance from apertures by a distance that is greater than the largest aperture dimension.

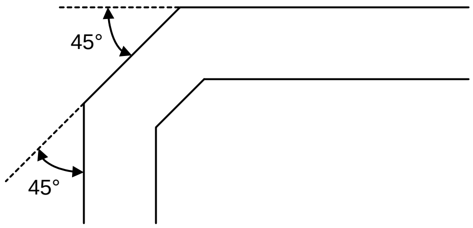


Figure 12-3. Trace Routing

- The reference plane for the differential pairs should be continuous and low impedance. It is recommended that the reference plane be either ground or 1.9 V dc (the voltage used by the PHY). This provides an adequate return path for and high frequency noise currents.
- Do not route differential pairs over splits in the associated reference plane as it may cause discontinuity in impedances.

12.1.4.5 Signal Termination and Coupling

The device has internal termination on the MDI signals. External resistors are not needed. Adding pads for external resistors can degrade signal integrity.

12.1.4.6 Signal Trace Geometry for 1000 BASE-T Designs

The key factors in controlling trace EMI radiation are the trace length and the ratio of trace-width to trace-height above the reference plane. To minimize trace inductance, high-sped signals and signal layers that are close to a reference or power plane should be as short and wide as practical. Ideally, this trace width to height above the ground plane ratio is between 1:1 and 3:1. To maintain trace impedance, the width of the trace should be modified when changing from one board layer to another if the two layers are not equidistant from the neighboring planes.

Each pair of signal should have a differential impedance of 100 Ω +/- 15%. If a particular tool cannot design differential traces, it is permissible to specify 55-65 Ω single-ended traces as long as the spacing between the two traces is minimized. As an example, consider a differential trace pair on Layer 1 that is 8 mils (0.2 mm) wide and 2 mils (0.05 mm) thick, with a spacing of 8 mils (0.2 mm). If the fiberglass layer is 8 mils (0.2 mm) thick with a dielectric constant, E_R, of 4.7, the calculated single-ended impedance would be approximately 61 Ω and the calculated differential impedance would be approximately 100 Ω

When performing a board layout, do not allow the CAD tool auto-router to route the differential pairs without intervention. In most cases, the differential pairs will have to be routed manually.

Note: Measuring trace impedance for layout designs targeting 100 Ω often results in lower actual impedance. Designers should verify actual trace impedance and adjust the layout accordingly. If the actual impedance is consistently low, a target of 105 – 110 Ω should compensate for second order effects.

It is necessary to compensate for trace-to-trace edge coupling, which can lower the differential impedance by up to 10 Ω, when the traces within a pair are closer than 30 mils (edge to edge).

12.1.4.7 Trace Length and Symmetry for 1000 BASE-T Designs

As indicated earlier, the overall length of differential pairs should be less than four inches measured from the Ethernet device to the magnetics.

The differential traces (within each pair) should be equal in total length to within 50 mils (1.25 mm) and as symmetrical as possible. Asymmetrical and unequal length traces in the differential pairs contribute to common mode noise. If a choice has to be made between matching lengths and fixing symmetry, more emphasis should be placed on fixing symmetry. Common mode noise can degrade the receive circuit's performance and contribute to radiated emissions.

12.1.4.7.1 Signal Detect

Each port of the 82580 has a signal detect pin for connection to optical transceivers. For designs without optical transceivers, these signals can be left unconnected because they have internal pull-up resistors. Signal detect is not a high-speed signal and does not require special layout.

12.1.4.8 Impedance Discontinuities

Impedance discontinuities cause unwanted signal reflections. Minimize vias (signal through holes) and other transmission line irregularities. If vias must be used, a reasonable budget is two per differential trace. Unused pads and stub traces should also be avoided.

12.1.4.9 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 be separated from analog signal grounds to reduce coupling. Noisy logic 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.

12.1.4.10 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 300 mils (7.5 mm) of the differential pairs. If digital signals on other board layers cannot be separated by a ground plane, they should be routed perpendicular to the differential pairs. If there is another LAN controller on the board, take care to keep the differential pairs from that circuit away.

Some rules to follow for signal isolation:

- Separate and group signals by function on separate layers if possible. Keep a minimum distance between differential pairs more than seven times the dielectric thickness away from each other and other traces, including NVM traces and parallel digital traces.
- Physically group together all components associated with one clock trace to reduce trace length and radiation.
- Isolate 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.

12.1.4.11 Traces for Decoupling Capacitors

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.

12.1.4.12 Light Emitting Diodes for Designs Based on the 82580

The 82580 provides three programmable high-current push-pull (active high) outputs 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 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.

Since the LEDs are likely to be integral to a magnetics module, 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.

The LED ports are fully programmable through the NVM interface.

12.1.5 Physical Layer Conformance Testing

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

12.1.5.1 Conformance Tests for 10/100/1000 Mb/s Designs

Crucial tests are as follows, listed in priority order:

- Bit Error Rate (BER). Good indicator of real world network performance. Perform bit error rate testing with long and short cables and many link partners. The test limit is 10^{-11} errors.
- Output Amplitude, Rise and Fall Time (10/100 Mb/s), Symmetry and Droop (1000Mbps). For the 82575 controller, use the appropriate PHY test waveform.
- Return Loss. Indicator of proper impedance matching, measured through the RJ-45 connector back toward the magnetics module.
- Jitter Test (10/100 Mb/s) or Unfiltered Jitter Test (1000 Mb/s). Indicator of clock recovery ability (master and slave for Gigabit controller).

12.1.6 Troubleshooting Common Physical Layout Issues

The following is a list of common physical layer design and layout mistakes in LAN On Motherboard Designs.

1. Lack of symmetry between the two traces within a differential pair. Asymmetry can create common-mode noise and distort the waveforms. For each component and/or via that one trace encounters, the other trace should encounter the same component or a via at the same distance from the Ethernet silicon.
2. Unequal length of the two traces within a differential pair. Inequalities create common-mode noise and will distort the transmit or receive waveforms.
3. Excessive distance between the Ethernet silicon and the magnetics. Long traces on FR4 fiberglass epoxy substrate will attenuate the analog signals. In addition, any impedance mismatch in the traces will be aggravated if they are longer than the four inch guideline.
4. Routing any other trace parallel to and close to one of the differential traces. Crosstalk getting onto the receive channel will cause degraded long cable BER. Crosstalk getting onto the transmit channel can cause excessive EMI emissions and can cause poor transmit BER on long cables. At a minimum, other signals should be kept 0.3 inches from the differential traces.
5. Routing one pair of differential traces too close to another pair of differential traces. After exiting the Ethernet silicon, the trace pairs should be kept 0.3 inches or more away from the other trace pairs. The only possible exceptions are in the vicinities where the traces enter or exit the magnetics, the RJ-45 connector, and the Ethernet silicon.
6. Use of a low-quality magnetics module.
7. Re-use of an out-of-date physical layer schematic in a Ethernet silicon design. The terminations and decoupling can be different from one PHY to another.
8. Incorrect differential trace impedances. It is important to have ~100 Ω impedance between the two traces within a differential pair. This becomes even more important as the differential traces become longer. To calculate differential impedance, many impedance calculators only multiply the single-ended impedance by two. This does not take into account edge-to-edge capacitive coupling between the two traces. When the two traces within a differential pair are kept close to each other, the edge coupling can lower the effective differential impedance by 5 Ω to 20 Ω Short traces have fewer problems if the differential impedance is slightly off target.

12.2 PCIe

The controller connects to the host system using a PCIe interface. The interface can be configured to operate in several link modes. These are detailed in the functional description chapter. 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 100 differential DC impedance. Board termination is not required.

Refer to the *PCI Express* Base Specification*, *Revision 2.0* and *PCI Express* Card Electromechanical Specification*, *Revision 2.0.*

12.2.1 Link Width Configuration

The device supports link widths of x4, x2, or x1 as determined by the PCIe PHY Auto Configuration Structure. The configuration is loaded into the Maximum Link Width field of the PCIe capability Register (LCAP[11:6]; with the silicon default of a x4 link).

During link configuration, the platform and the controller negotiate a common link width. In order for this to work, the selected maximum number of PCIe lanes must be connected to the host system.

12.2.2 Polarity Inversion and Lane Reversal

To ease routing, designers have the flexibility to the lane reversal modes supported by the the 82580. 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 description of available combinations, consult the functional description in the PCIe interconnects chapter of this document.

12.2.3 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.

12.3 Clock Source

All designs require a 25 MHz clock source. The 82580 uses the 25 MHz source to generate clocks up to 125 MHz and 1.25 GHz for the PHY circuits. For optimum results with lowest cost, connect a 25 MHz parallel resonant crystal and appropriate load capacitors at the XTAL1 and XTAL2 leads. The frequency tolerance of the timing device should be 30 ppm or better.

Refer to the Intel® Ethernet Controllers Timing Device Selection Guide for more information on choosing crystals. For further information regarding the clock for the 82580, refer to the sections about frequency control, crystals, and oscillators that follow.

12.3.1 Frequency Control Device Design Considerations

This section provides information regarding frequency control devices, including crystals and oscillators, for use with all Intel Ethernet controllers. Several suitable frequency control devices are available; none of which present any unusual challenges in selection. The concepts documented herein are applicable to other data communication circuits, including Platform LAN Connect devices (PHYs).

The Intel Ethernet controllers contain amplifiers, which when used with the specific external components, form the basis for feedback oscillators. These oscillator circuits, which are both economical and reliable, are described in more detail in Section 12.4.1.

The Intel Ethernet controllers also have bus clock input functionality, however a discussion of this feature is beyond the scope of this document, and will not be addressed.

The chosen frequency control device vendor should be consulted early in the design cycle. Crystal and oscillator manufacturers familiar with networking equipment clock requirements may provide assistance in selecting an optimum, low-cost solution.

12.3.2 Frequency Control Component Types

Several types of third-party frequency reference components are currently marketed. A discussion of each follows, listed in preferred order.

12.3.2.1 Quartz Crystal

Quartz crystals are generally considered to be the mainstay of frequency control components due to their low cost and ease of implementation. They are available from numerous vendors in many package types and with various specification options.

12.3.2.2 Fixed Crystal Oscillator

A packaged fixed crystal oscillator comprises an inverter, a quartz crystal, and passive components conveniently packaged together. The device renders a strong, consistent square wave output. Oscillators used with microprocessors are supplied in many configurations and tolerances.

Crystal oscillators should be restricted to use in special situations, such as shared clocking among devices or multiple controllers. As clock routing can be difficult to accomplish, it is preferable to provide a separate crystal for each device.

12.3.2.3 Programmable Crystal Oscillators

A programmable oscillator can be configured to operate at many frequencies. The device contains a crystal frequency reference and a phase lock loop (PLL) clock generator. The frequency multipliers and divisors are controlled by programmable fuses.

A programmable oscillator’s accuracy depends heavily on the Ethernet device’s differential transmit lines. The Physical Layer (PHY) uses the clock input from the device to drive a differential Manchester (for 10 Mb/s operation), an MLT-3 (for 100 Mbps operation) or a PAM-5 (for 1000 Mbps operation) encoded analog signal across the twisted pair cable. These signals are referred to as self-clocking, which means the clock must be recovered at the receiving link partner. Clock recovery is performed with another PLL that locks onto the signal at the other end.

PLLs are prone to exhibit frequency jitter. The transmitted signal can also have considerable jitter even with the programmable oscillator working within its specified frequency tolerance. PLLs must be designed carefully to lock onto signals over a reasonable frequency range. If the transmitted signal has high jitter and the receiver's PLL loses its lock, then bit errors or link loss can occur.

PHY devices are deployed for many different communication applications. Some PHYs contain PLLs with marginal lock range and cannot tolerate the jitter inherent in data transmission clocked with a programmable oscillator. The American National Standards Institute (ANSI) X3.263-1995 standard test method for transmit jitter is not stringent enough to predict PLL-to-PLL lock failures, therefore, the use of programmable oscillators is not recommended.

12.3.2.4 Ceramic Resonator

Similar to a quartz crystal, a ceramic resonator is a piezoelectric device. A ceramic resonator typically carries a frequency tolerance of ±0.5%, – inadequate for use with Intel Ethernet controllers, and therefore, should not be utilized.

12.4 Crystal Support

12.4.1 Crystal Selection Parameters

All crystals used with Intel Ethernet controllers are described as AT-cut, which refers to the angle at which the unit is sliced with respect to the long axis of the quartz stone. Table 12-250 lists crystals which have been used successfully in other designs (however, no particular product is recommended):

Table 12-2. Crystal Manufacturers and Part Numbers

Manufacturer	Part No.
KDS America	DSX321G
NDK America Inc.	41CD25.0F1303018
TXC Corporation - USA	7A25000165 9C25000008

For information about crystal selection parameters, see the electrical specification.

12.4.1.1 Vibrational Mode

Crystals in the above-referenced frequency range are available in both fundamental and third overtone. Unless there is a special need for third overtone, use fundamental mode crystals.

At any given operating frequency, third overtone crystals are thicker and more rugged than fundamental mode crystals. Third overtone crystals are more suitable for use in military or harsh industrial environments. Third overtone crystals require a trap circuit (extra capacitor and inductor) in the load circuitry to suppress fundamental mode oscillation as the circuit powers up. Selecting values for these components is beyond the scope of this document.

12.4.1.2 Nominal Frequency

Intel Ethernet controllers use a crystal frequency of 25.000 MHz. The 25 MHz input is used to generate a 125 MHz transmit clock for 100BASE-TX and 1000BASE-TX operation – 10 MHz and 20 MHz transmit clocks, for 10BASE-T operation.

12.4.1.3 Frequency Tolerance

The frequency tolerance for an Ethernet Platform LAN Connect is dictated by the IEEE 802.3 specification as ±50 parts per million (ppm). This measurement is referenced to a standard temperature of 25° C. Intel recommends a frequency tolerance of ±30 ppm.

12.4.1.4 Temperature Stability and Environmental Requirements

Temperature stability is a standard measure of how the oscillation frequency varies over the full operational temperature range (and beyond). Several optional temperature ranges are currently available, including -40° C to +85° C for industrial environments. Some vendors separate operating temperatures from temperature stability. Manufacturers may also list temperature stability as 50 ppm in their data sheets.

Note: Crystals also carry other specifications for storage temperature, shock resistance, and reflow solder conditions. Crystal vendors should be consulted early in the design cycle to discuss the application and its environmental requirements.

12.4.1.5 Calibration Mode

The terms series-resonant and parallel-resonant are often used to describe crystal oscillator circuits. Specifying parallel mode is critical to determining how the crystal frequency is calibrated at the factory.

A crystal specified and tested as series resonant oscillates without problem in a parallel-resonant circuit, but the frequency is higher than nominal by several hundred parts per million. The purpose of adding load capacitors to a crystal oscillator circuit is to establish resonance at a frequency higher than the crystal’s inherent series resonant frequency.

Figure 12-4 shows the recommended placement and layout of an internal oscillator circuit. Note that pin X1 and X2 refers to XTAL1 and XTAL2 in the Ethernet device, respectively. The crystal and the capacitors form a feedback element for the internal inverting amplifier. This combination is called parallel-resonant, because it has positive reactance at the selected frequency. In other words, the crystal behaves like an inductor in a parallel LC circuit. Oscillators with piezoelectric feedback elements are also known as “Pierce” oscillators.

12.4.1.6 Load Capacitance

The formula for crystal load capacitance is as follows:

$$C_L = \frac{(C1 \cdot \ \ C2)}{(C1 + C2)} + C_{stray}$$

where C1 = C2 = 27 pF
and C_{stray} = allowance for additional capacitance in pads, traces and the chip carrier within the Ethernet device package

An allowance of 3 pF to 7 pF accounts for lumped stray capacitance. The calculated load capacitance is 16 pF with an estimated stray capacitance of about 5 pF.

Individual stray capacitance components can be estimated and added. For example, surface mount pads for the load capacitors add approximately 2.5 pF in parallel to each capacitor. This technique is especially useful if Y1, C1 and C2 must be placed farther than approximately one-half (0.5) inch from the device. It is worth noting that thin circuit boards generally have higher stray capacitance than thick circuit boards. Consult the PCIe Design Guide for more information.

The oscillator frequency should be measured with a precision frequency counter where possible. The load specification or values of C1 and C2 should be fine tuned for the design. As the actual capacitance load increases, the oscillator frequency decreases.

Note: C1 and C2 may vary by as much as 5% (approximately 1 pF) from their nominal values.

12.4.1.7 Shunt Capacitance

The shunt capacitance parameter is relatively unimportant compared to load capacitance. Shunt capacitance represents the effect of the crystal's mechanical holder and contacts. The shunt capacitance should equal a maximum of 6 pF.

12.4.1.8 Equivalent Series Resistance

Equivalent Series Resistance (ESR) is the real component of the crystal's impedance at the calibration frequency, which the inverting amplifier's loop gain must overcome. ESR varies inversely with frequency for a given crystal family. The lower the ESR, the faster the crystal starts up. Use crystals with an ESR value of 50 Ω or better.

12.4.1.9 Drive Level

Drive level refers to power dissipation in use. The allowable drive level for a Surface Mounted Technology (SMT) crystal is less than its through-hole counterpart, because surface mount crystals are typically made from narrow, rectangular AT strips, rather than circular AT quartz blanks.

Some crystal data sheets list crystals with a maximum drive level of 1 mW. However, Intel Ethernet controllers drive crystals to a level less than the suggested 0.3 mW value. This parameter does not have much value for on-chip oscillator use.

12.4.1.10 Aging

Aging is a permanent change in frequency (and resistance) occurring over time. This parameter is most important in its first year because new crystals age faster than old crystals. Use crystals with a maximum of ±5 ppm per year aging.

12.4.1.11 Reference Crystal

The normal tolerances of the discrete crystal components can contribute to small frequency offsets with respect to the target center frequency. To minimize the risk of tolerance-caused frequency offsets causing a small percentage of production line units to be outside of the acceptable frequency range, it is important to account for those shifts while empirically determining the proper values for the discrete loading capacitors, C1 and C2.

Even with a perfect support circuit, most crystals will oscillate slightly higher or slightly lower than the exact center of the target frequency. Therefore, frequency measurements (which determine the correct value for C1 and C2) should be performed with an ideal reference crystal. When the capacitive load is exactly equal to the crystal's load rating, an ideal reference crystal will be perfectly centered at the desired target frequency.

12.4.1.11.1 Reference Crystal Selection

There are several methods available for choosing the appropriate reference crystal:

- If a Saunders and Associates (S&A) crystal network analyzer is available, then discrete crystal components can be tested until one is found with zero or nearly zero ppm deviation (with the appropriate capacitive load). A crystal with zero or near zero ppm deviation will be a good reference crystal to use in subsequent frequency tests to determine the best values for C1 and C2.

- If a crystal analyzer is not available, then the selection of a reference crystal can be done by measuring a statistically valid sample population of crystals, which has units from multiple lots and approved vendors. The crystal, which has an oscillation frequency closest to the center of the distribution, should be the reference crystal used during testing to determine the best values for C1 and C2.
- It may also be possible to ask the approved crystal vendors or manufacturers to provide a reference crystal with zero or nearly zero deviation from the specified frequency when it has the specified CLoad capacitance.

When choosing a crystal, customers must keep in mind that to comply with IEEE specifications for 10/100 and 10/100/1000Base-T Ethernet LAN, the transmitter reference frequency must be precise within ±50 ppm. Intel® recommends customers to use a transmitter reference frequency that is accurate to within ±30 ppm to account for variations in crystal accuracy due to crystal manufacturing tolerance.

12.4.1.11.2 Circuit Board

Since the dielectric layers of the circuit board are allowed some reasonable variation in thickness, the stray capacitance from the printed board (to the crystal circuit) will also vary. If the thickness tolerance for the outer layers of dielectric are controlled within ±17 percent of nominal, then the circuit board should not cause more than ±2 pF variation to the stray capacitance at the crystal. When tuning crystal frequency, it is recommended that at least three circuit boards are tested for frequency. These boards should be from different production lots of bare circuit boards.

Alternatively, a larger sample population of circuit boards can be used. A larger population will increase the probability of obtaining the full range of possible variations in dielectric thickness and the full range of variation in stray capacitance.

Next, the exact same crystal and discrete load capacitors (C1 and C2) must be soldered onto each board, and the LAN reference frequency should be measured on each circuit board.

The circuit board, which has a LAN reference frequency closest to the center of the frequency distribution, should be used while performing the frequency measurements to select the appropriate value for C1 and C2.

12.4.1.11.3 Temperature Changes

Temperature changes can cause the crystal frequency to shift. Therefore, frequency measurements should be done in the final system chassis across the system’s rated operating temperature range.

12.4.2 Crystal Placement and Layout Recommendations

Crystal clock sources should not be placed near I/O ports or board edges. Radiation from these devices can be coupled into the I/O ports and radiate beyond the system chassis. Crystals should also be kept away from the Ethernet magnetics module to prevent interference.

Note: Failure to follow these guidelines could result in the 25 MHz clock failing to start.

When designing the layout for the crystal circuit, the following rules must be used:

- Place load capacitors as close as possible (within design-for-manufacturability rules) to the crystal solder pads. They should be no more than 90 mils away from crystal pads.

- The two load capacitors, crystal component, the Ethernet controller device, and the crystal circuit traces must all be located on the same side of the circuit board (maximum of one via-to-ground load capacitor on each XTAL trace).
- Use 27 pF (5% tolerance) 0402 load capacitors.
- Place load capacitor solder pad directly in line with circuit trace (see Figure 12-4, point A).
- Use 50 Ω impedance single-ended microstrip traces for the crystal circuit.
- Route traces so that electro-magnetic fields from XTAL2 do not couple onto XTAL1. No differential traces.
- Route XTAL1 and XTAL2 traces to nearest inside corners of crystal pad (see Figure 12-4, point B).
- Ensure that the traces from XTAL1 and XTAL2 are symmetrically routed and that their lengths are matched.
- The total trace length of XTAL1 or XTAL2 should be less than 750 mils.

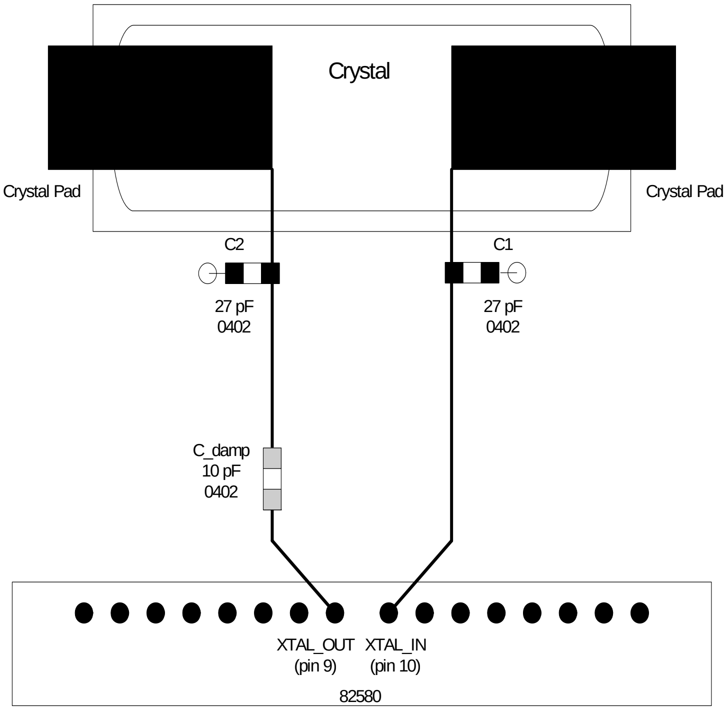


Figure 12-4. Recommended Crystal Placement and Layout

12.5 Oscillator Support

The 82580 clock input circuit is optimized for use with an external crystal. However, an oscillator can also be used in place of the crystal with the proper considerations:

- The input capacitance introduced by the 82580 (approximately 20 pF) is greater than the capacitance specified by a typical oscillator (approximately 15 pF).
- The input clock jitter from the oscillator can impact the 82580 clock and its performance.

Note: The power consumption of additional circuitry equals about 1.5 mW.

Table 12-251 lists oscillators that can be used with the controller. Please note that no particular oscillator is recommended).

Table 12-3. Oscillator Manufacturers and Part Numbers

Manufacturer	Part No.
MtronPTI*	M214TCN
Kyocera* Corporation	K-30-3C0-SE

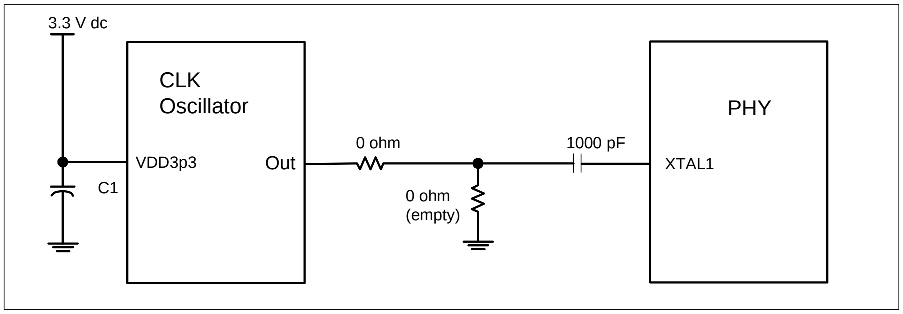


Figure 12-5. Oscillator Solution

12.5.1 Oscillator Placement and Layout Recommendations

Oscillator clock sources should not be placed near I/O ports or board edges. Radiation from these devices can be coupled into the I/O ports and radiate beyond the system chassis. Oscillators should also be kept away from the Ethernet magnetics module to prevent interference.

12.6 Device Disable

For a LOM design, it might be desirable for the system to provide BIOS-setup capability for selectively enabling or disabling LOM devices. This enables designers more control over system resource-management, avoid conflicts with add-in NIC solutions, etc. The 82580 provides support for selectively enabling or disabling it.

Device disable is initiated by asserting the asynchronous DEV_OFF_N pin. The DEV_OFF_N pin has an internal pull-up resistor, so that it can be left not connected to enable device operation.

The NVM's *Device Disable Power Down En* bit enables device disable mode (hardware default is that the mode is disabled).

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

Assertion or deassertion of PCIe PE_RST_N does not have any effect while the 82580 is in device disable mode (that is, the 82580 stays in the respective mode as long as DEV_OFF_N is asserted). However, the 82580 might momentarily exit the device disable mode from the time PCIe PE_RST_N is de-asserted again and until the NVM is read.

During power-up, the DEV_OFF_N pin is ignored until the NVM is read. From that point, the 82580 might enter device disable if DEV_OFF_N is asserted.

Note: The DEV_OFF_N pin should maintain its state during system reset and system sleep states. It should also insure the proper default value on system power up. For example, a designer could use a GPIO pin that defaults to 1b (enable) and is on system suspend power. For example, it maintains the state in S0-S5 ACPI states).

12.6.1 BIOS Handling of Device Disable

Assume that in the following power-up sequence the DEV_OFF_N signal is driven high (or it is already disabled)

1. The PCIe is established following the GIO_PWR_GOOD.
2. BIOS recognizes that the entire 82580 should be disabled.
3. The BIOS drives the DEV_OFF_N signal to the low level.
4. As a result, the 82580 samples the DEV_OFF_N signals and enters either the device disable mode.
5. The BIOS could put the link in the Electrical IDLE state (at the other end of the PCIe link) by clearing the *Link Disable* bit in the Link Control register.
6. BIOS might start with the device enumeration procedure (the entire 82580 functions are invisible).
7. Proceed with normal operation

Re-enable could be done by driving high the DEV_OFF_N signal, followed later by bus enumeration.

12.7 SMBus and NC-SI

SMBus and NC-SI are interfaces for pass-through and configuration traffic between the Management Controller (MC) and the device.

Note: Intel recommends that the SMBus be connected to the ICH or MC for the EEPROM recovery solution. If the connection is to a MC, it will be able to send the EEPROM release command.

The 82580 can be connected to an external MC. It operates in one of two modes:

- SMBus mode
- NC-SI mode

The Clock-out (if enabled) is provided in all power states (unless the device is disabled).

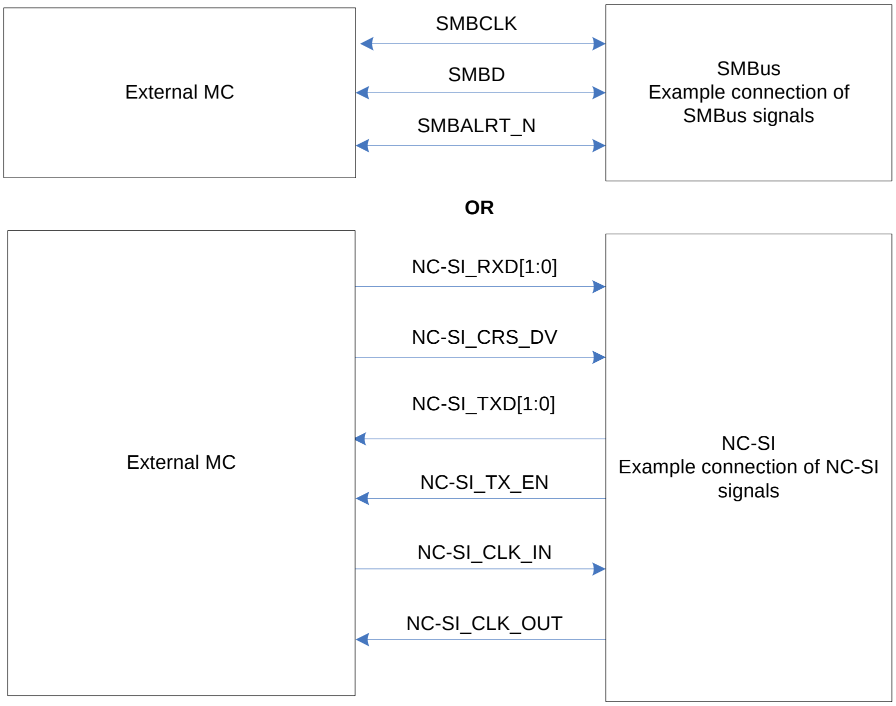


Figure 12-6. External MC Connections with NC-SI and SMBus

12.8 NC-SI

12.8.1 Design Requirements

12.8.1.1 Network Controller

The NC-SI Interface enables network manageability implementations required by information technology personnel for remote control and alerting via the LAN. Management packets can be routed to or from a management processor.

12.8.1.2 External Management Controller (MC)

An external MC is required to meet the requirements called out in the latest NC-SI specification as it relates to this interface.

12.8.1.3 Reference Schematic

The following reference schematic (provides connectivity requirements for single and mulit drop applications. This configuration only has a single connection to the MC. The network device also supports multi-drop NC-SI configuration architecture with software arbitration support from the management controller.

See the NC-SI specification for connectivity requirements for multi-drop applications.

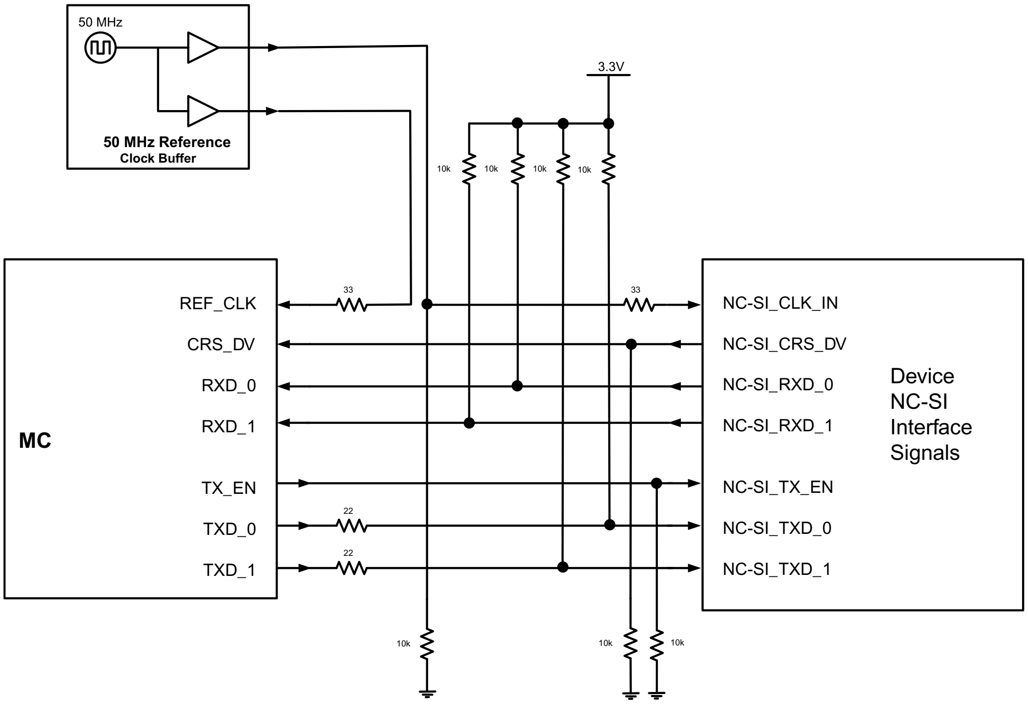


Figure 12-7. NC-SI Connection Schematic: Single-Drop Configuration

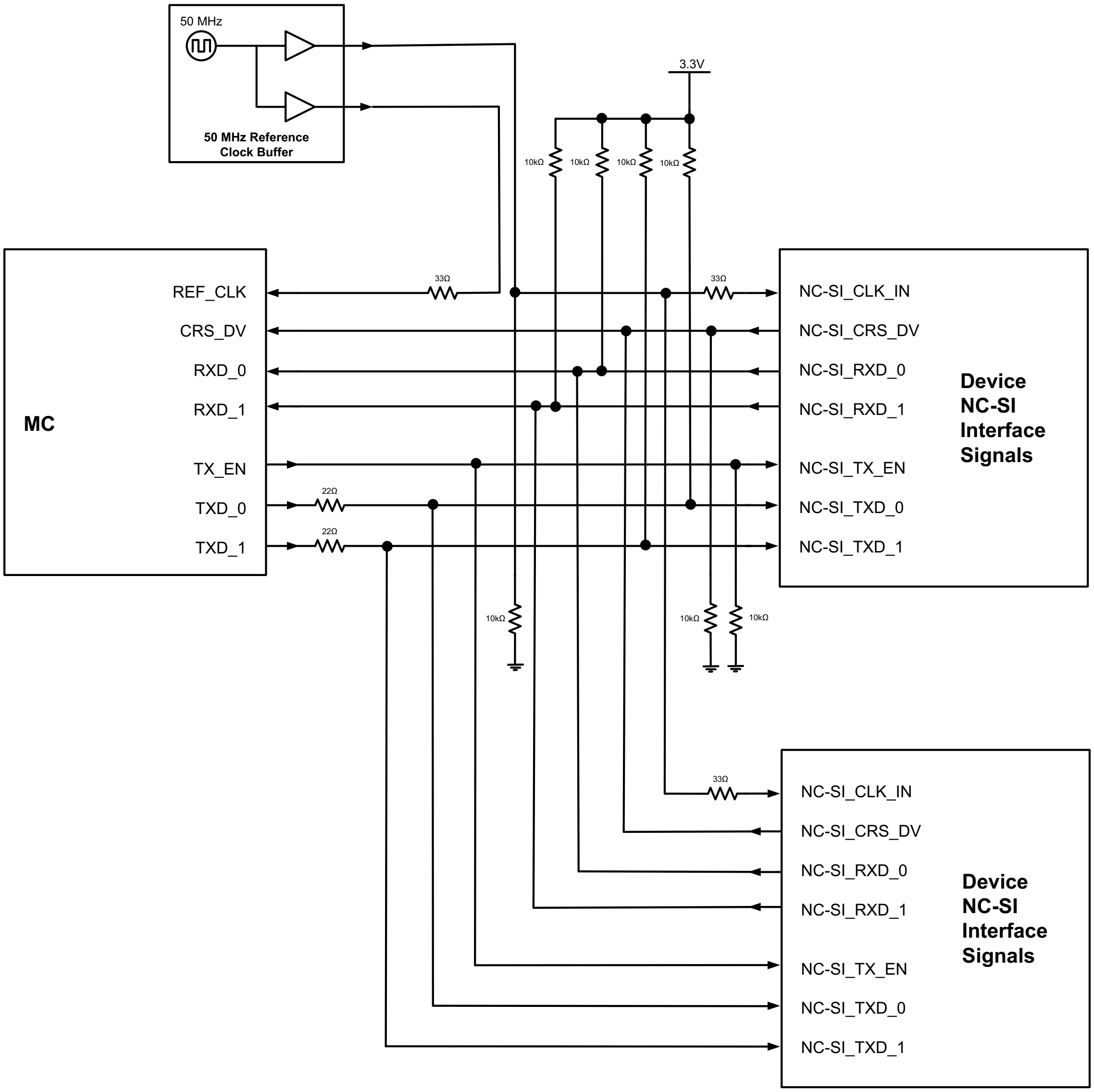


Figure 12-8. NC-SI Connection Schematic: Multi-Drop Configuration

12.8.2 Layout Requirements

12.8.2.1 Board Impedance

The NC-SI signaling interface is a single ended signaling environment and as such Intel recommends a target board and trace impedance of 50 Ohms plus 20% and minus 10%. This impedance ensures optimal signal integrity and quality.

12.8.2.2 Trace Length Restrictions

The recommended maximum trace lengths for each circuit board application is dependent on the number drops and the total capacitive loading from all the trace segments on each NC-SI signal net. The number via's must also be considered. Circuit board material variations and trace etch process variations affect the trace impedance and trace capacitance. For each fixed design, highest trace capacitance occurs when trace impedance is lowest.For the FR4 board stack-up provided in direct connect applications, the maximum length for a 50 ohm NC-SI trace would be approximately 9 inches on a minus 10% board impedance skew. This ensures that signal integrity and quality are preserved and enables the design to comply with NC-SI electrical requirements.

For special applications which require longer NC-SI traces, the total functional NC-SI trace length can be extended with non-compliant rise time by:

- providing good clock and signal alignment
- testing with the target receiver to verify it meets setup and hold requirements.

For multi-drop applications, the total capacitance and the extra resistive loading affect the rise time. A multi-drop of two devices limits the total length to 8 inches. A multi-drop of four limits the total length to 6.5 inches. Capacitive loading of extra via's have a nominal effect on the total load.

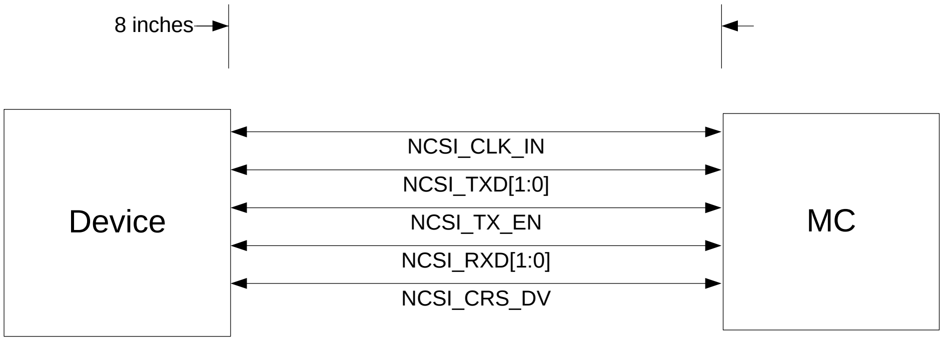


Figure 12-9. NC-SI Trace Length Requirement for Direct Connect

Table 12-252 shows how 7 more vias increase the rise time by 0.5ns. Again, longer trace lengths can be achieved.

Table 12-4. Stack Up, 7 Vias

Item	Value	Units
Trace width	4.5	mils
Trace thickness	1.9	mils

Table 12-4. Stack Up, 7 Vias

Dielectric thickness	3.0	mils
Dielectric constant	4.1	--
Loss Tangent	0.024	--
Nominal Impedance	50	Ohms
Trace Capacitance	1.39	pf/inch

Table 12-253 shows example trace lengths for the multi-drop topology of two and four represented in the figures that follow.

*

Table 12-5. Example Trace Lengths for Multi-Drop Topologies, 2 & 4

Multi-drop length parameter used in Figure 12-10 and Figure 12-11 .	Segment length example for multi drop configurations			
	Two drop configuration		Four drop configuration	
	Length (Inches)	Trace capacitance (Pf)	Length (Inches)	Trace capacitance (Pf)
L1	2	2.8	1.5	8.8
L2	4	5.6	2	16.1
L3	2	2.8	1	8.1
Total	8	11.1	6.5	35.6

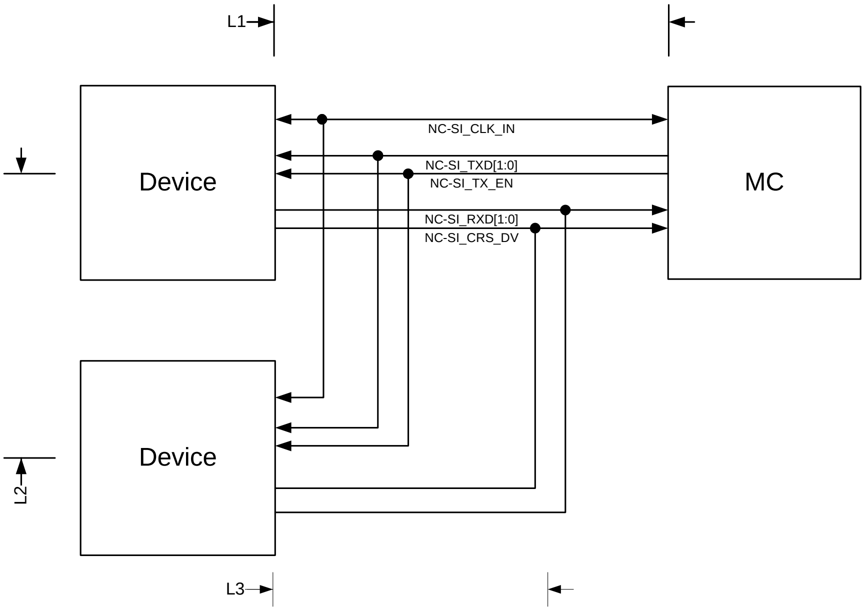


Figure 12-10. Example 2-Drop Topology

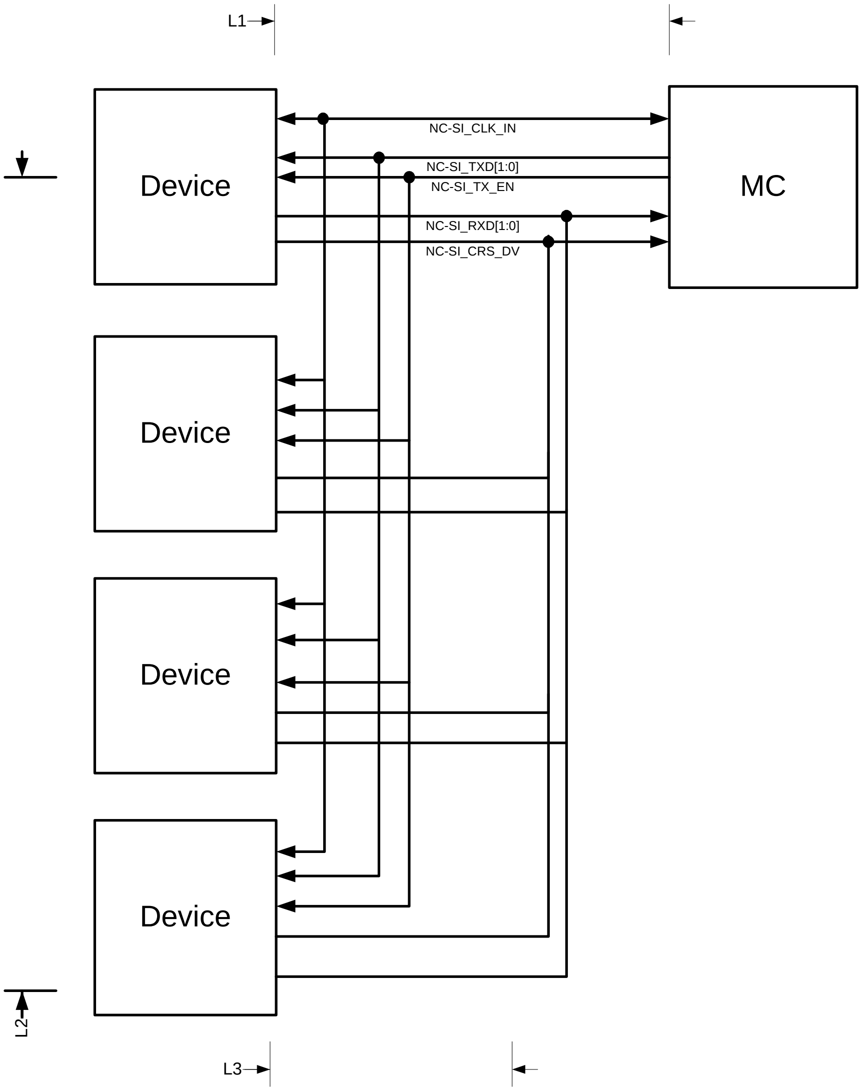


Figure 12-11. Example 4-Drop Topology

Table 12-6. Compliant NC-SI Maximum Length on a 50 ohm -10% Skew-board with Example Stack-up.

Topology	Total maximum compliant linear bus size (inches)	Number of vias	Approximate Net trace capacitance minus load capacitance (pf)
4 multi-drop	6.0	1	8.3
4 multi-drop	5.5	8	8.3
2 multi-drop	8.0	1	11.1

Table 12-6. Compliant NC-SI Maximum Length on a 50 ohm -10% Skew-board with Example Stack-up.

2 multi-drop	7.5	8	11.1
Point to point	9.0	1	12.5
Point to point	8.5	8	12.5

Extending NC-SI to a maximum 11ns rise time increases the maximum trace length.

Table 12-7. Functional NC SI maximum length on a 50 ohm -10% skew board with Example Stack-up (based on actual lab-measured solution)

Topology	Total maximum functional linear bus size (inches)	Number of vias	Approximate Net trace capacitance minus load capacitance (pf)
4 multi-drop	19	1	26.4
4 multi-drop	18	8	26.4
2 multi-drop	20	1	27.8
2 multi-drop	19	8	27.8
Point to point	22	1	30.6
Point to point	21	8	30.6

§ §

13.0 Thermal Management

This chapter provides methods for determining the operating temperature of the 82580 in a system based on case temperature. Case temperature is a function of the local ambient and internal temperatures of the component.

Properly designed solutions provide adequate cooling to maintain case temperature (Tcase) at or below what is listed in Table 13-258. This is accomplished by providing a low local ambient temperature, airflow, and creating a minimal thermal resistance to that local ambient temperature. Heatsinks and higher airflow may be required if temperatures exceed those listed.

13.1 Thermal Design Considerations

In a system, 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 surrounding the component that may limit the size of a thermal enhancement (heat sink).

Acomponent'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

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, 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.

Note: Operation outside the functional limit can degrade system performance, cause logic errors, or cause device and/or system damage. Temperatures exceeding the maximum operating limits may result in irreversible changes in the device operating characteristics. Sustained operation at component maximum temperature limit may affect long-term device reliability.

13.2 Terminology

The following terminology is used in this chapter:

- **PBGA Plastic Ball Grid Array**: A surface-mount package using a BGA structure whose PCB-interconnect method consists of Pb-free solder ball array on the interconnect side of the package and attached to a plastic substrate material. An integrated heat spreader (**IHS**) may be present for larger PBGA packages for enhanced thermal performance (but IHS is not present for the 82580).
- **Junction**: Refers to a P-N junction on the silicon. In this document, it is used as a temperature reference point (for example, Θ_{JA} refers to the "junction" to "ambient" thermal resistance).
- **Ambient**: Refers to local ambient temperature of the bulk air approaching the component. It can be measured by placing a thermocouple approximately 1"inch upstream from the component edge.
- **Lands**: The 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)**: The estimated maximum possible/expected power generated in a component by a realistic application. Use Maximum power requirements listed in Table 2.
- **LFM:** Linear feet per minute (airflow).
- **Θ_{JA} (Theta JA)**: Thermal resistance junction-to-ambient, °C/W.
- **Ψ_{JT} (Psi JT)**: Junction-to-top (of package) thermal characterization parameter, °C/W. Ψ_{JT} does not represent thermal resistance, but instead is a characteristic parameter that can be used to convert between Tj and Tcase when knowing the total TDP. Ψ_{JT} is easy to characterize in simulations or measurements, and is equal to Tj minus Tcase divided by the total TDP. This parameter can vary by environment conditions like heat sink and airflow.

13.3 Thermal Specifications

The thermal solution must maintain a case temperature at or below the values specified in Table 13-258. System-level or component-level thermal enhancements are required to dissipate the generated heat to ensure the case temperature never exceeds the maximum temperatures listed. Table 13-257 lists the thermal performance parameters per JEDEC JESD51-2 standard.

In Table 13-257, the Θ_{JA} values should be used as reference only and can vary by system environment. Ψ_{JT} values also can vary by system environment. They are given in Table 13-257 as the maximum value for 82580 simulations.

Analysis indicates that real applications are unlikely to cause the 82580 to be at Tcase-max for sustained periods of time, given that Tcase can reasonably be expected to be a distribution of temperatures. Sustained operation at Tcase-max may affect long-term reliability of the 82580 and the system; sustained operation at Tcase-max should be evaluated during the thermal design process and steps taken to further reduce the Tcase temperature.

Good system airflow is critical to dissipate the highest possible thermal power. The size and number of fans, vents, and/or ducts, and, their placement in relation to components and airflow channels within the system determine airflow. Acoustic noise constraints may limit the size and types of fans, vents and ducts that can be used in a particular design.

To develop a reliable, cost-effective thermal solution, all of the system variables must be considered. Use system-level thermal characteristics and simulations to account for individual component thermal requirements.

Table 13-1. Package Thermal Characteristics in Standard JEDEC Environment

Package	Θ_{JA} (°C/W)	Ψ_{JT} (°C/W)
17 mm PBGA[1]	22.6[7]	2.90[9]
17 mm PBGA - HS (19 x 6.3mm height)[2]	17.7[8]	2.90[9]
17 mm PBGA –HS (30 x 12mm height)[3]	16.6[8]	2.90[9]
17 mm PBGA–HS (25 x 7mm height)[4]	15.6[8]	2.90[9]
17 mm PBGA–HS (7 x 10mm height)[5]	14.1[8]	2.90[9]
17 mm PBGA–HS (40 x 10mm height)[6]	13.1[8]	2.90[9]

Notes:

1. Integrated Heat Spreader. The 82580 is a PBGA
2. Heat sink 19 x 19 x 6.3mm
3. Heat sink 30 x 30 x 12mm
4. Heat sink 25 x 25 x 7mm
5. Heat sink 27 x 27 x 10mm
6. Heat sink 40 x 40 x 10mm
7. 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.
8. Natural Convection (Still Air), Heat sink attached.
9. Psi_JT is given as maximum value for a worst-case 82580 scenario, and may vary to a lesser value in some scenarios.

Table 13-2. 82580 Line Absolute Thermal Maximum Rating (°C)

APPLICATION	Measured TDP (W)[1]	Tcase Max-hs[2] (°C)[3]
82580	4.0 @ 123 °C Tj_max	111

Notes:

1. Power value shown in Table 2 is measured maximum power, also known as Thermal Design Power (TDP). 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 - (Ψ_{JT} x Power) where Ψ_{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. Θ_{JA} is the thermal resistance junction-to-ambient of the package.

13.3.1 Case Temperature

The 82580 is designed to operate properly as long as Tcase rating is not exceeded. Section 6.1 discusses proper guidelines for measuring the case temperature.

13.4 Thermal Attributes

13.4.1 Designing for Thermal Performance

Section 13.9, “Heatsink and Attach Suppliers ” and Section 13.10, “PCB Guidelines ” document the PCB and system design recommendations required to achieve 82580 thermal performance.

13.4.2 Typical System Definition

A system with the following attributes was used to generate thermal characteristics data:

- A heatsink case, see Section 13.5.2, “Default Enhanced Thermal Solution ”.
- A JEDEC JESD 51-9 standard 2s2p Board.

Keep the following in mind when reviewing the data that is included in this document:

- All data is preliminary and is not validated against physical samples.
- Your system design may be significantly different.
- A larger board (more than six copper layers) may improve 82580 thermal performance.

13.4.3 Package Mechanical Attributes

For information on package attributes, see Chapter 11.0, Electrical/Mechanical Specification.

13.4.4 Package Thermal Characteristics

See the Table 13-259 and Table 13-260 for an aid in determining the optimum airflow and heatsink combination for the 82580. The table shows Tcase as a function of airflow and ambient temperature at the Thermal Design Power (TDP) for a typical system. The Red blocked value(s) indicate airflow/ ambient combinations that exceed the allowable case temperature for the 82580 line at 4.0 W.

Your system design may vary. Thermal models are available upon request (Flotherm* and Icepak*). Contact your local Intel representative.

Table 13-3. Quad Port

Case Temperature (Max = 111C)

No Heat Sink		Airflow (LFM)								
		0	50	100	150	200	250	300	350	400
Ambient Temperature (C)	45	120.9	117.2	113.8	111.2	109.1	107.3	106.0	104.9	103.8
	50	126.1	121.9	118.5	116.0	113.9	112.2	110.8	109.7	108.7
	55	131.2	126.7	123.3	120.8	118.7	117.0	115.7	114.6	113.6
	60	135.1	131.2	127.8	125.3	123.3	121.8	120.5	119.4	118.5
	65	139.3	135.9	132.5	130.1	128.1	126.4	125.1	124.0	123.0
	70	143.6	140.6	137.3	134.8	132.9	131.2	129.9	128.9	127.9
	75	147.9	145.2	142.0	139.6	137.7	136.0	134.8	133.7	132.8
	80	152.3	149.9	146.7	144.4	142.5	140.9	139.6	138.6	137.6
	85	156.7	154.6	151.4	149.1	147.3	145.7	144.4	143.4	142.5

Rose City Heat Sink		Airflow (LFM)								
		0	50	100	150	200	250	300	350	400
Ambient Temperature (C)	45	93.6	86.4	79.0	73.9	70.8	68.8	67.3	66.3	65.4
	50	98.4	91.2	83.9	78.9	75.7	73.7	72.3	71.2	70.4
	55	103.4	95.8	88.8	83.9	80.7	78.7	77.2	76.2	75.3
	60	107.5	100.5	93.2	88.3	85.4	83.3	82.0	81.1	80.3
	65	111.6	105.2	98.1	93.1	90.0	88.1	87.0	85.9	85.2
	70	116.0	109.8	103.0	98.0	94.9	93.0	91.7	90.7	90.1
	75	120.3	114.6	107.8	103.0	99.8	98.0	96.6	95.7	94.9
	80	124.5	119.3	112.6	107.9	104.8	102.9	101.6	100.6	99.9
	85	128.9	124.0	117.4	112.8	109.7	107.8	106.5	105.6	104.8

AAVID-Thermalloy Heat Sink		Airflow (LFM)								
		0	50	100	150	200	250	300	350	400
Ambient Temperature (C)	45	91.1	85.2	78.3	73.1	69.8	67.3	65.5	64.0	62.8
	50	96.6	89.9	82.7	78.1	74.7	72.0	70.1	68.7	67.7
	55	100.9	94.5	87.5	82.6	79.3	76.9	75.1	73.7	72.5
	60	105.1	99.0	92.3	87.5	84.2	81.8	80.0	78.6	77.5
	65	109.4	103.5	97.1	92.3	89.1	86.8	85.0	83.6	82.4
	70	113.8	108.0	101.9	97.2	94.0	91.7	90.0	88.5	87.4
	75	118.2	112.6	106.6	102.1	98.9	96.6	94.9	93.5	92.4
	80	122.6	117.1	111.4	107.0	103.8	101.6	99.8	98.5	97.4
	85	127.0	121.8	116.2	111.8	108.7	106.5	104.8	103.4	102.3

Table 13-3. Quad Port (Continued)

Alpha Heat Sink LPD40-10B		Airflow (LFM)								
		0	50	100	150	200	250	300	350	400
Ambient Temperature (C)	45	89.2	83.0	78.2	75.5	73.6	72.4	71.5	70.8	70.3
	50	93.3	87.6	83.0	80.4	78.6	77.4	76.5	75.8	75.2
	55	97.7	92.2	87.9	85.2	83.5	82.3	81.4	80.7	80.2
	60	102.1	96.8	92.7	90.1	88.2	87.1	86.2	85.7	85.2
	65	106.5	101.4	97.5	95.0	93.1	92.0	91.2	90.5	90.1
	70	111.0	106.0	102.3	99.9	98.1	96.9	96.1	95.5	95.0
	75	115.5	110.7	107.2	104.8	103.0	101.9	101.1	100.4	99.9
	80	120.0	115.4	112.0	109.7	108.0	106.8	106.0	105.4	104.9
	85	124.6	120.1	116.9	114.6	112.9	111.7	111.0	110.4	109.8

Alpha Heat Sink LPD25-7B		Airflow (LFM)								
		0	50	100	150	200	250	300	350	400
Ambient Temperature (C)	45	100.0	93.5	87.5	84.4	82.4	80.9	79.8	78.8	78.0
	50	104.4	98.2	92.3	89.3	87.1	85.8	84.7	83.7	83.0
	55	108.8	102.8	97.1	93.9	92.0	90.6	89.5	88.7	87.9
	60	113.0	107.4	101.9	98.8	96.9	95.5	94.4	93.5	92.8
	65	117.5	112.1	106.7	103.6	101.7	100.4	99.3	98.5	97.7
	70	122.0	116.7	111.6	108.5	106.6	105.3	104.2	103.4	102.6
	75	126.5	121.4	116.4	113.3	111.5	110.2	109.1	108.3	107.5
	80	131.0	126.0	121.2	118.2	116.3	115.0	114.0	113.2	112.5
	85	135.5	130.7	126.0	123.0	121.2	119.9	118.9	118.1	117.4

Alpha Heat Sink Z19-6.3B		Airflow (LFM)								
		0	50	100	150	200	250	300	350	400
Ambient Temperature (C)	45	107.6	101.6	97.1	94.1	91.7	89.7	88.2	86.8	85.7
	50	111.7	106.1	101.8	98.9	96.4	94.6	93.0	91.7	90.6
	55	115.9	110.7	106.5	103.6	101.2	99.4	97.8	96.6	95.5
	60	119.8	115.3	111.2	108.3	106.0	104.2	102.7	101.4	100.4
	65	124.1	120.0	115.9	113.1	110.9	109.0	107.5	106.3	105.2
	70	128.3	124.6	120.7	117.9	115.7	113.9	112.4	111.2	110.1
	75	132.7	129.2	125.4	122.7	120.5	118.7	117.3	116.0	115.0
	80	137.0	133.8	130.1	127.5	125.3	123.6	122.1	120.9	119.8
	85	141.4	138.5	134.8	132.2	130.1	128.4	127.0	125.8	124.7

Table 13-4. Dual Port

Case Temperature (Max = 111C)

No Heat Sink		Airflow (LFM)								
		0	50	100	150	200	250	300	350	400
Ambient Temperature (C)	45	98.3	94.2	91.7	90.1	89.0	88.0	87.2	86.4	85.8
	50	102.4	98.9	96.5	95.0	93.8	92.9	92.1	91.3	90.7
	55	106.7	103.6	101.3	99.8	98.7	97.8	96.9	96.2	95.6
	60	111.2	108.3	106.1	104.6	103.5	102.6	101.8	101.1	100.5
	65	115.7	113.1	110.9	109.5	108.4	107.5	106.7	106.0	105.4
	70	120.3	117.8	115.7	114.3	113.3	112.4	111.6	110.9	110.3
	75	124.9	122.6	120.5	119.2	118.1	117.2	116.5	115.8	115.2
	80	129.5	127.4	125.3	124.0	123.0	122.1	121.4	120.7	120.1
	85	134.1	132.1	130.1	128.8	127.8	127.0	126.3	125.6	125.0

Rose City Heat Sink		Airflow (LFM)								
		0	50	100	150	200	250	300	350	400
Ambient Temperature (C)	45	84.8	76.5	69.7	65.7	63.4	61.9	60.8	60.1	59.5
	50	89.1	81.2	74.6	70.8	68.4	66.8	65.8	65.1	64.5
	55	93.5	86.1	79.5	75.5	73.2	71.9	70.8	70.1	69.5
	60	98.0	90.9	84.5	80.5	78.2	76.7	75.9	75.1	74.5
	65	102.6	95.6	89.4	85.4	83.1	81.7	80.7	80.1	79.5
	70	107.2	100.5	94.3	90.4	88.1	86.7	85.7	84.9	84.5
	75	111.9	105.3	99.3	95.4	93.1	91.6	90.6	89.9	89.3
	80	116.5	110.1	104.2	100.3	98.1	96.6	95.6	94.9	94.3
	85	121.1	115.0	109.1	105.3	103.0	101.6	100.6	99.9	99.3

AAVID-Thermalloy Heat Sink		Airflow (LFM)								
		0	50	100	150	200	250	300	350	400
Ambient Temperature (C)	45	76.5	72.0	68.5	66.4	64.8	63.6	62.5	61.6	60.9
	50	80.7	76.6	73.2	71.3	69.7	68.5	67.4	66.6	65.8
	55	85.0	81.3	78.0	76.0	74.5	73.4	72.4	71.5	70.8
	60	89.5	86.0	82.8	80.8	79.4	78.2	77.3	76.4	75.7
	65	94.0	90.6	87.6	85.7	84.2	83.1	82.1	81.4	80.6
	70	98.6	95.3	92.4	90.5	89.1	88.0	87.0	86.2	85.5
	75	103.1	100.0	97.2	95.4	94.0	92.9	91.9	91.1	90.4
	80	107.7	104.7	102.0	100.2	98.9	97.8	96.8	96.0	95.4
	85	112.2	109.4	106.8	105.1	103.8	102.7	101.7	101.0	100.3

Table 13-4. Dual Port (Continued)

Alpha Heat Sink LPD40-10B		Airflow (LFM)								
		0	50	100	150	200	250	300	350	400
Ambient Temperature (C)	45	75.9	71.0	67.5	65.6	64.3	63.5	62.9	62.4	62.1
	50	80.0	75.7	72.3	70.5	69.3	68.5	67.9	67.4	67.0
	55	84.5	80.4	77.3	75.3	74.1	73.4	72.8	72.4	72.0
	60	89.0	85.1	82.2	80.2	79.0	78.3	77.7	77.3	77.0
	65	93.6	89.9	87.0	85.2	84.0	83.2	82.7	82.2	81.9
	70	98.2	94.6	91.9	90.1	88.9	88.2	87.6	87.2	86.8
	75	102.8	99.4	96.8	95.1	93.9	93.1	92.6	92.2	91.8
	80	107.4	104.1	101.7	100.1	98.8	98.1	97.6	97.1	96.8
	85	112.1	108.9	106.6	105.0	103.8	103.1	102.5	102.1	101.7

Alpha Heat Sink LPD25-7B		Airflow (LFM)								
		0	50	100	150	200	250	300	350	400
Ambient Temperature (C)	45	83.4	78.2	73.9	71.7	70.3	69.2	68.5	67.8	67.3
	50	87.7	82.9	78.7	76.6	75.2	74.2	73.4	72.8	72.3
	55	92.2	87.6	83.6	81.3	80.0	79.2	78.4	77.7	77.2
	60	96.7	92.4	88.4	86.2	84.9	84.0	83.2	82.7	82.2
	65	101.4	97.1	93.3	91.1	89.8	88.9	88.2	87.6	87.1
	70	106.0	101.9	98.2	96.0	94.7	93.8	93.1	92.5	92.0
	75	110.6	106.6	103.0	100.9	99.6	98.7	98.0	97.4	96.9
	80	115.2	111.4	107.9	105.8	104.5	103.7	103.0	102.4	101.9
	85	119.7	116.2	112.8	110.7	109.5	108.6	107.9	107.3	106.8

Alpha Heat Sink Z19-6.3B		Airflow (LFM)								
		0	50	100	150	200	250	300	350	400
Ambient Temperature (C)	45	88.4	83.7	80.5	78.3	76.6	75.3	74.2	73.3	72.5
	50	92.5	88.4	85.2	83.2	81.5	80.2	79.1	78.2	77.4
	55	96.9	93.1	90.0	87.9	86.3	85.0	84.0	83.1	82.4
	60	101.3	97.8	94.8	92.8	91.2	89.9	88.8	88.1	87.3
	65	105.8	102.5	99.6	97.6	96.0	94.8	93.8	92.9	92.1
	70	110.3	107.2	104.4	102.5	100.9	99.7	98.7	97.8	97.1
	75	114.9	112.0	109.2	107.3	105.8	104.6	103.6	102.7	102.0
	80	119.4	116.7	114.0	112.2	110.7	109.5	108.5	107.6	106.9
	85	124.0	121.5	118.8	117.0	115.5	114.3	113.4	112.5	111.8

13.5 Thermal Enhancements

One method 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.

13.5.1 Clearances

To be effective, a heatsink should have a pocket of air around it that is free of obstructions.

13.5.2 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 solutions (see Section 13.5.2, Default Enhanced Thermal Solution). These solutions replicate the performance defined in Section 13.4.4 at Thermal Design Power (TDP). If after implementing the Recommended Enhanced Thermal Solution, the case temperature continues to exceed allowable values additional cooling is needed. This additional cooling may be achieved by improving airflow to the component and/or adding additional thermal enhancements.

13.5.3 Extruded Heatsinks

If required, the following extruded heatsinks are the suggested. Figure 13-1 through Figure 13-5 shows the multiple profiles.

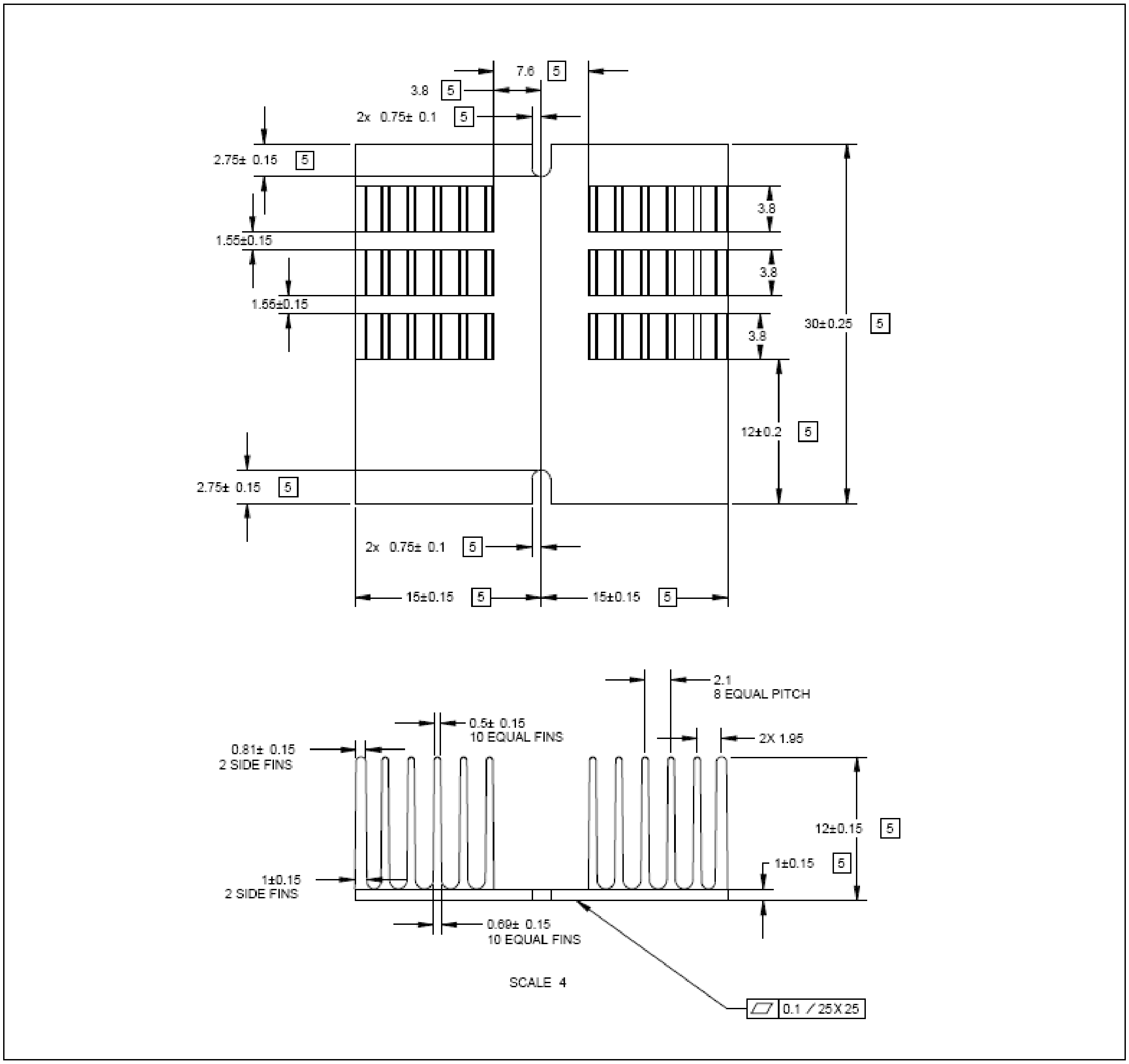
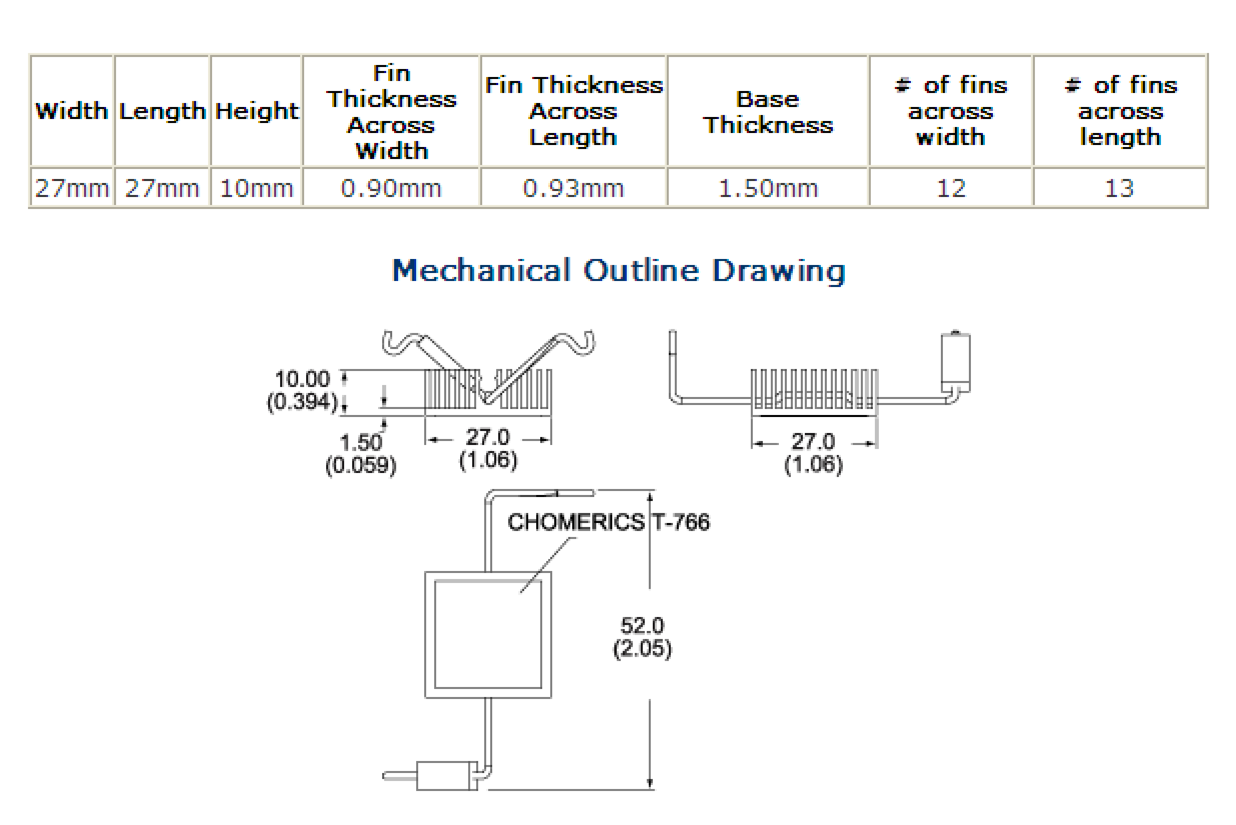


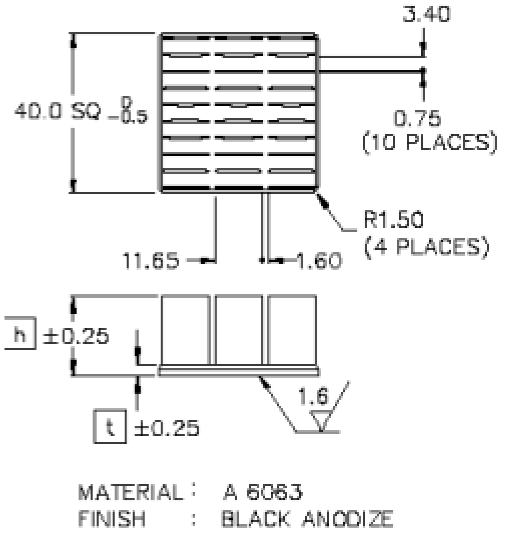
Figure 13-1. Rose City 12 mm Height Passive Heat Sink



Width	Length	Height	Fin Thickness Across Width	Fin Thickness Across Length	Base Thickness	# of fins across width	# of fins across length
27mm	27mm	10mm	0.90mm	0.93mm	1.50mm	12	13

Mechanical Outline Drawing

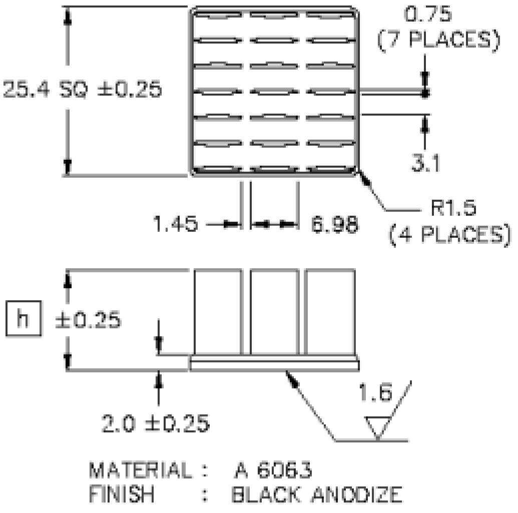
Figure 13-2. 10mm Height Passive Heat Sink (AAVID Thermalloy PN: 374324B60023G)



MODEL	HEIGHT h	THICKNESS t	WEIGHT (grams)
LPD40–3B	3	2.0	9.4
LPD40–4B	4		10.0
LPD40–5B	5		10.6
LPD40–6B	6		11.2
LPD40–7B	7		11.8
LPD40–10B	10	3.0	16.4
LPD40–15B	15		19.2
LPD40–20B	20		22.0
LPD40–25B	25		24.8
LPD40–30B	30		27.6
LPD40–35B	35		30.4

Dimensions : mm

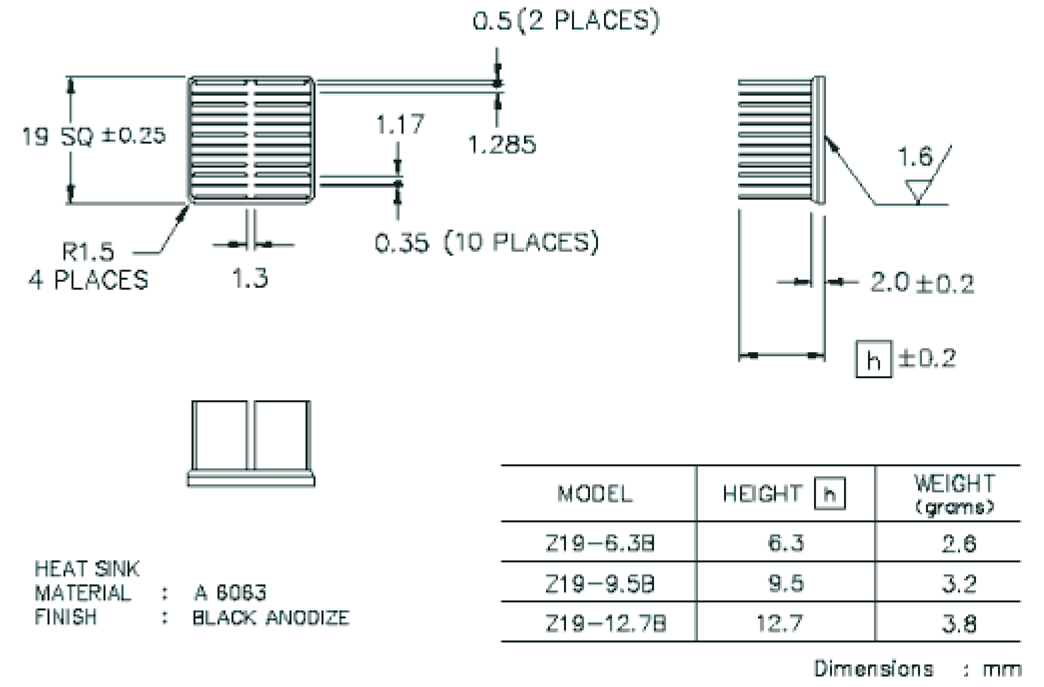
Figure 13-3. 10mm Height Passive Heat Sink (Alpha Novatech, Inc. PN: LPD40-10B)



MODEL	HEIGHT h	WEIGHT (grams)
LPD25–3B	3	4.1
LPD25–4B	4	4.3
LPD25–5B	5	4.5
LPD25–6B	6	4.7
LPD25–7B	7	4.9
LPD25–10B	10	5.5
LPD25–15B	15	6.5
LPD25–20B	20	7.5
LPD25–25B	25	8.5

Dimensions : mm

Figure 13-4. 7mm Height Passive Heat Sink (Alpha Novatech, Inc. PN: LPD25-7B)



MODEL	HEIGHT h	WEIGHT (grams)
Z19−6.3B	6.3	2.6
Z19−9.5B	9.5	3.2
Z19−12.7B	12.7	3.8

Figure 13-5. 6.3mm Height Passive Heat Sink (Alpha Novatech, Inc. PN: Z19-6.3B)

13.5.4 Attaching the Extruded Heatsink

An extruded heatsink may be attached using clips with a phase change thermal interface material. For attaching methods, contact the heatsink manufacturer.

13.5.4.1 Clips

A well-designed clip, in conjunction with a thermal interface material (tape, grease, etc.) often offers the best combination of mechanical stability and reworkability. Use of a clip requires significant advance planning as mounting holes are required in the PCB.

13.5.4.2 Thermal Interface (PCM45 Series)

The recommended thermal interface is the PCM45 series from Honeywell. PCM45 Series thermal interface pads are phase change materials formulated for use in high performance devices requiring minimum thermal resistance for maximum heat sink performance and component reliability. These pads consist of an electrically non-conductive, dry film that softens at device operating temperatures resulting in “greasy-like” performance. However, Intel has not fully validated the PCM45 Series TIM.

Each PCA, system and heatsink combination varies in attach strength. Carefully evaluate the reliability of double sided thermal interface tape attachments prior to high-volume use (see Section 5.5).

13.5.4.3 Maximum Static Normal Load

The 82580 package is capable of sustaining a maximum static normal load of 9.89lbf (44N). This load is an evenly distributed uniform compressive load in a direction perpendicular to the top surface of the. This mechanical load limit must not be exceeded during heatsink installation, mechanical stress testing, standard shipping conditions, and/or any other use condition. The PCB under the package must be fully supported during heat sink installation to prevent any deformation of the PCB. Note that the heat sink attach solution must not include continuous stress to the package, with the exception of a uniform load to maintain the heatsink-to-package thermal interface. This load specification is based on limited testing for design characterization, and is for the package only.

13.5.5 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 in Table 13-261.

Table 13-5. Reliability Validation

Test[1]	Requirement	Pass/Fail Criteria[2]
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-Temperature Life	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

Notes:

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.

13.6 Thermal Interface Management for Heat-Sink Solutions

To optimize heatsink design, it is important to understand the interface between the silicon die and the heatsink base. Thermal conductivity effectiveness depends on the following:

- Bond line thickness
- Interface material area
- Interface material thermal conductivity

13.6.1 Bond Line Management

The gap between the silicon die 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 the heatsink base, the silicon die, and the package encapsulent, plus the thickness of the thermal interface material (for example, PSA, thermal grease, epoxy) used to join the two surfaces.

13.6.2 Interface Material Performance

The following factors impact the performance of the interface material between the silicon die and the heatsink base:

- Thermal resistance of the material
- Wetting/filling characteristics of the material

13.6.2.1 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.

13.6.2.2 Wetting/Filling Characteristics of the Material

The wetting/filling characteristic of the thermal interface material is its ability to fill the gap between the package's 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.

13.7 Measurements for Thermal Specifications

Determining the thermal properties of the system requires careful case temperature measurements. Guidelines for measuring the 82580 case temperature are provided in Section 6.1.

13.7.1 Case Temperature Measurements

Maintain Tcase at or below the maximum case temperatures listed in Table 13-258 to ensure functionality and reliability. 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).

13.7.1.1 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.

Note: 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 (see Figure 13-2). This is the preferred method and is recommended for use with packages not having a heat sink.

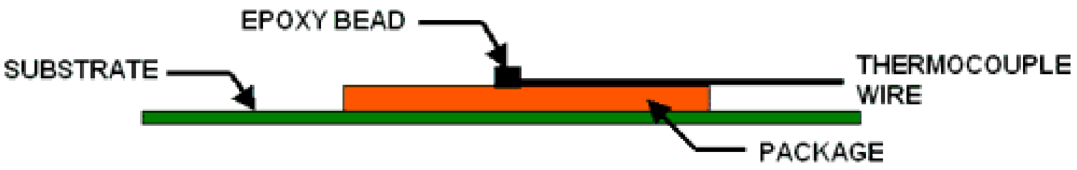


Figure 13-6. Technique for Measuring Tcase with 0° Angle Attachment, No Heatsink

13.7.1.2 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.

Note: 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 (see Figure 13-3). 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.

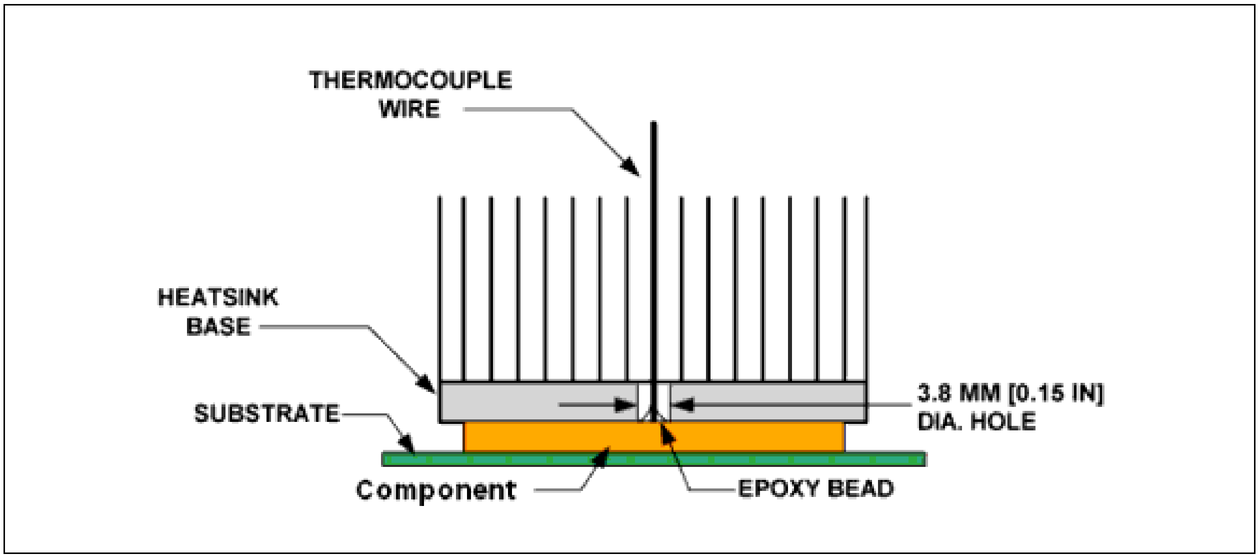


Figure 13-7. Technique for Measuring Tcase with 90° Angle Attachment

13.8 Thermal Diode

The 82580 incorporates an on-die diode that may be used to monitor the die temperature (junction temperature). A thermal sensor located on the motherboard or a stand-alone measurement kit, may monitor the die temperature of the 82580 for thermal management or characterization.

.

13.9 Heatsink and Attach Suppliers

Table 13-6. Heatsink and Attach Suppliers

Part	Part Number	Supplier	Contact
Alpha Heat Sinks	LPD40-10B LPD25-7B Z19-6.3B	Alpha Novatech, Inc	Sales Aplha Novatech, Inc. 408-567-8082 sales@alphanovtech.com
Aavid-Thermalloy Heat Sink	374324B60023G	Aavid Thermalloy	Harish Rutti 67 Primrose Dr. Suite 200 Laconia, NH 03246 Business: 972-633-9371 x27
Rose City Heat Sink	E66546-001	Intel Corp.	Please contact your Intel representative
PCM45 Series	PCM45F	Honeywell	North America Technical Contact: Paula Knoll 1349 Moffett Park Dr. Sunnyvale, CA 94089 Cell: 1-858-705-1274 Business: 858-279-2956 paula.knoll@honeywell.com

13.10 PCB Guidelines

The following general PCB design guidelines are recommended to maximize the thermal performance of PBGA 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 82580 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.

Note: The above information is provided as a general guideline to help maximize the thermal performance of the components.

§ §

14.0 Diagnostics

14.1 Customer Visible Features

14.1.1 JTAG Test Mode Description

The 82580 includes a JTAG (TAP) port compliant with the IEEE Standard Test Access Port and Boundary Scan Architecture 1149.6 Specification.

The TAP controller is accessed serially through the four dedicated pins TCK, TMS, TDI, and TDO. TMS, TDI, and TDO operate synchronously with TCK which is independent of all other clock within the 82580. This interface can be used for test and debug purposes. System board interconnects can be DC tested using the boundary scan logic in pads. Table 14-1 shows TAP controller related pin descriptions. Table 14-2 describes the TAP instructions supported by the 82580. The default instruction after JTAG reset is IDCODE.

Table 14-1. TAP Controller Pins

Signal	I/O	Description
TCK	In	Test clock input for the test logic defined by IEEE1149.1. Note: Signal should be connected to ground through a 1 kΩ pull-down resistor.
TDI	In	Test Data Input. Serial test instructions and data are received by the test logic at this pin. Note: Signal should be connected to VCC33 through a 1 kΩ pull-up resistor.
TDO	O/D	Test Data Output. The serial output for the test instructions and data from the test logic defined in IEEE1149.1. Note: Signal should be connected to VCC33 through a 1 kΩ pull-up resistor.
TMS	In	Test Mode Select input. The signal received at TMS is decoded by the TAP controller to control test operations. Note: Signal should be connected to VCC33 through a 1 kΩ pull-up resistor.

Reference: 321027-012EN

Table 14-2. TAP Instructions Supported

Instruction	Description	Comment
BYPASS	The BYPASS command selects the Bypass Register, a single bit register connected between TDI and TDO pins. This allows more rapid movement of test data to and from other components in the system.	IEEE 1149.1 Std. Instruction
EXTEST	The EXTEST Instruction allows circuitry or wiring external to the devices to be tested. Boundary-scan Register Cells at outputs are used to apply stimulus while Boundary-scan cells at input pins are used to capture data.	IEEE 1149.1 Std. Instruction
SAMPLE / PRELOAD	The SAMPLE/PRELOAD instruction is used to allow scanning of the boundary scan register without causing interference to the normal operation of the device. Two functions can be performed by use of the Sample/Preload instruction. SAMPLE – allows a snapshot of the data flowing into and out of a device to be taken without affecting the normal operation of the device. PRELOAD – allows an initial pattern to be placed into the boundary scan register cells. This allows initial known data to be present prior to the selection of another boundary-scan test operation.	IEEE 1149.1 Std. Instruction
IDCODE	The IDCODE instruction is forced into the parallel output latches of the instruction register during the Test-Logic-Reset TAP state. This allows the device identification register to be selected by manipulation of the broadcast TMS and TCK signals for testing purposes, as well as by a conventional instruction register scan operation. The ID code value for the 82580 A0 is 0x01509013 (Intel's Vendor ID = 0x13, Device ID = 0x1509, Rev ID = 0x0)	IEEE 1149.1 Std. Instruction
HIGHZ	The HIGHZ instruction is used to force all outputs of the device (except TDO) into a high impedance state. This instruction shall select the Bypass Register to be connected between TDI and TDO in the Shift-DR controller state.	IEEE 1149.1 Std. Instruction

§ §

Appendix A. Changes from the 82576

This appendix summarizes changes in the 82580 relative to the 82576.

Table A-1. Changes in the Programming Interface Relative to the 82576

Feature	Registers/Descriptors	Description
Queue Reduction to 8	MRQC.Def_Q field	Width of field changed to 3 in the 82580
	RETA table	Only 1 to 3 bits used for RSS indexing
	“RxTxQ” field in EICR, EICS, EIMS, EIMC and EIAM registers (when GPIE.Multiple_MSIX = 0)	Width of field changed to 3 in the 82580
Quad port support	New PCIe function 2 and Function 3	4 PCIe functions instead of 2 in the 82576.
	STATUS.LAN ID	Added ID values for LAN 2 (10b) and LAN 3 (11b). Updated initial values to be according to port number.
	MDICNFG and MDIC	• Added new MDICNFG (0x0A00; R/W) register to program MDIO configuration setup (PHY address, Internal/External PHY and Common MDIO usage). • Moved fields PHYADD (bits 25:21) and Destination (Bit 31) from MDIC register to MDICNFG register. PHYADD value loaded from EEPROM on initialization to support common MDIO configuration. • Added Com_MDIO bit (bit 30) to MDICNFG register to support shared MDIO usage in Multi port PHYs. Com_MDIO value loaded from EEPROM on initialization.
	EEDIAG	Added new bits LAN2 Disable, LAN3 Disable, LAN2 Disable Strap Behavior, LAN3 Disable Strap Behavior, LAN2 PCI Disable and LAN3 PCI Disable
	STRAP	Added to the diagnostic Strap Register bits “Lan2 disable” (bit 16) and “LAN3 disable” (bit 17).
	EEARBC	Added VALID_CORE2 (bit 13) and VALID_CORE3 (bit 14) bits.
	EEMNGCTL	Added CFG_DONE 2 (bit 20) and CFG_DONE 3 (bit 21) bits.
	FACTPS	Added new bits Func2 Power State, LAN2 Valid, Func2 Aux_En, Func3 Power State, LAN3 Valid and Func3 Aux_En. Modified functionality of “LAN Function Sel” bit.
	FWSM	Added indication of SerDes Configuration error for ports 2 and 3 (bits 30:29)
	SW_FW_SYNC	Added Firmware Software synchronization bits for ports 2 and 3
	PHPM	Added “Disable 100 in non-D0a” bit (bit 9) in PHY Power Management - PHPM register, to enable lower power consumption in low power state.
Shared DMA	CTRL	• Added *DEV_RST* bit to enable reset of all ports including DMA and other common logic. See Section 4.3.2 for expected Software behavior. • Setting the *GIO Master Disable* bit resets the internal DMA queues of the function. After clearing the *GIO Master Disable* bit driver should re-initialize the transmit and receive queues of this function.
	“DRSTA” and “TCP Timer” bit in ICR,ICS,IMS and IMC registers	• Added DRSTA bit (bit 30) that indicates CTRL.DEV_RST assertion. When CTRL.DEV_RST is asserted in one port all ports are reset. When ICS.DRSTA is set an interrupt is received on all ports. • Moved “TCP Timer” bit to bit 29 from bit 30.

Table A-1. Changes in the Programming Interface Relative to the 82576 (Continued)

Feature	Registers/Descriptors	Description
Shared DMA	SW_FW_SYNC and STATUS	• Added SW_FW_SYNC.SW_MB_SM semaphore bit (bit 8) to enable coordination of ownership of the SWMBWR mailbox write register, between drivers. • Added DEV_RST_SET bit (bit 20) to STATUS register. Bit indicates that a device driver initiated a device reset.
	IRPBS and ITPBS	Replaced RXPBS and TXPBS registers with register IRPBS and ITPBS, that don't support traffic classes. Buffer size value is loaded from EEPROM on reset. The 82580 supports utilization of idle ports memory by active ports in dual port or single port NICs. • Default Receive buffer size is 32KB for quad port systems. • Default Transmit buffer size is 20KB for quad port systems.
	RDHTE and TDHTE	Moved Freeze bit (bit 25) in RDHTE and TDHTE diagnostic registers to bit 31 and added 3 new bits in field 29:27. Added bit PBTCDPWR (bit 25) to TDHTE register.
	RXDCTL and TXDCTL	• Changed default value of *RXDCTL.Enable* of Q0 and *TXDCTL.Enable* of Q0 from 1 to 0, to avoid corner cases when FLR is asserted by OS or DEV_RST is asserted by other driver. • Transmit descriptors and Data are pre-fetched when "TXDCTL[n].Enable" bit is set and descriptors are available, even when TCTL.EN bit is cleared. • Receive descriptors are pre-fetched when "RXDCTL[n].Enable" bit is set and descriptors are available, even when RCTL.RXEN bit is cleared.
	Command Register.BME	De-assertion of Bus Master Enable (BME) resets LAN port. See Section 4.3.2.1 for expected software behavior.
	SWMBWR, SWMB0, SWMB1, SWMB2 and SWMB3	To support Software mailbox interface the SWMBWR, SWMB0, SWMB1, SWMB2 and SWMB3 registers where added. See Section 4.7.3.
	ICR, ICS, IMS and IMC	SWMB bit (bit 8) was added to the ICR, ICS, IMS and IMC registers to support Software mailbox interface
PCIe Gen2	PCIe BARs	Updated PCIe BARs configuration to be similar to the 82599. Default operating mode is 32 bit BAR mode.
	BARCTRL	Added BARCTRL register and updated relevant EEPROM fields to support the 82599 64-bit flexible CSR and Flash BAR Size allocation.
	GSCL_1	Added to GSCL_1 register following bits: "LBC Enable 0", "LBC Enable 1", "LBC Enable 2" and "LBC Enable 3" (bits 7:4).
	PPHY_CTL	Added PPHY_CTL register to support new PCIe GEN 2 circuitry.
	PICAUSE and PIENA	Added PICAUSE and PIENA registers to support generation of PCIe error interrupts.
	GCR and GCR_EXT	Updated GCR and GCR_EXT (PCIe Control Extended) register to support new PCIe Gen2 design.
	Link Status PCIe configuration register	Added 5 Gb/s support to "Link Status.Link speed" configuration register field (bits 3:0)
	Link CAP PCIe configuration register	Added 5 Gb/s support to "Link CAP.Max Link speed" configuration register field (bits 3:0)
	Link Control 2 and Link Status 2 PCIe configuration register	Added Link Control 2 (0xD0; RW) and Link Status 2 (0xD2; RW) PCIe configuration registers
	"Uncorrectable Error Status", "Uncorrectable Error Mask" and "Uncorrectable Error Severity" PCIe configuration registers	Added support for *ECRC error* reporting (bit 19) and *ACS Violation* reporting (bit 21) in "Uncorrectable Error Status", "Uncorrectable Error Mask" and "Uncorrectable Error Severity" PCIe configuration registers.
	RTIV, FUNCTAG and LTIV	Removed RTIV, FUNCTAG and LTIV Registers.
	GSCL_LBT[3:0]	Replaced GSCL_LBT registers with GSCL_5_8 registers.
	GSCL_3 and GSCL_4	Removed GSCL_3 and GSCL_4 registers.

Table A-1. Changes in the Programming Interface Relative to the 82576 (Continued)

Feature	Registers/Descriptors	Description
PCIe Gen2	CDQM and PCIESPARE	Added CDQM and PCIESPARE registers.
	GIOANACTL0, GIOANACTL1, GIOANACTL2, GIOANACTL3, GIOANACTLALL, CCMCTL, SCCTL and FUNCTO	Registers removed.
	PCIEPHYDAT and PCIEPHYADR	Added PCIEPHYDAT and PCIEPHYADR to access PCIe configuration circuitry.
	MREVID	Replaced "default Rev ID" with "Step Rev ID" (bits 15:8) in Mirrored Revision ID - MREVID register.
	PEIND and PEINDM	Removed ghost_parity_desc_rd_fatal bit (bit 23) and ghost_parity_data_rd_fatal bit (bit 24) from PEIND and PEINDM registers.
	IOADDR and IODATA	• Enabled access to CSRs via configuration address space using *IOADDR* (Configuration address 0x98) and *IODATA* (Configuration address 0x9C) registers. • Removed support of access to Flash in IO address space using the *IOADDR* and *IODATA* registers.
PHY/MAC interface	CTRL_EXT	Added support in CTRL_EXT.LINK_MODE field (Bits 23:22) for KX operation: • LINK_MODE = 01b - 1000BASE-KX Backplane link
	CTRL	Changed default value of CTRL.RFCE (bit 27 - receive flow control enable) to 1b to be the same as auto-negotiation advertisement.
	FCRTH0	Flow Control Receive Threshold High programmed to *FCRTH0.RTH* field changed due to new u-architecture. See Section 3.5.5.3.1 for information on how to calculate value.
	PCONF, PSTAT, PCONT, LINK, PFIFO, CHAN, PHPM, PHGDIS, PHMISC1, PHMISC2 and PHCSEL	Removed Gbe PHY registers: PCONF, PSTAT, PCONT, LINK, PFIFO, CHAN, PHPM, PHGDIS, PHMISC1, PHMISC2 and PHCSEL
	PHCTRL2, PHLBKC, PHRERRC, PHMIC, PHCNFG, PHCTRL1, PHINTM, PHINTR, PHSTAT, PHDIAG and PHDSTAT.	Added new GbE PHY Diagnostic registers: PHCTRL2, PHLBKC, PHRERRC, PHMIC, PHCNFG, PHCTRL1, PHINTM, PHINTR, PHSTAT, PHDIAG and PHDSTAT.
	PHPM	Added PHPM register that's external to the PHY, with same functionality as the 82576 PHPM register that was internal to the GbE PHY.
	ICR, ICS, IMS and IMC	Added GPHY (Interrupt from Internal PHY) interrupt bit (bit 10) to ICR, ICS, IMS and IMC registers.
Virtualization	PSRTYPE	PSRTYPE controls header split of a single queue and not two queues as in the 82576, since the 82580 supports single queue pools and not dual queue pools.
	VMOLR	VMOLR.RSSE (RSS Poll Enable) removed and bit 17 is reserved since RSS is not supported in virtualization mode
	FTQF	Changed VF field (Bits 11:9) and VF Mask (Bit 15) in FTQF (5 Tuple filter) register to reserved due to single queue support in Virtualization mode.
	SR-IOV registers	No support for Virtual Function address space due to SR-IOV removal. Registers defined in the 82576 "Virtual function Register Descriptions" section not supported • Removed following dedicated Virtual Function device registers due to no SR-IOV support: VTCTRL, VTStatus, VTFRTIMER, VTEICS, VTEIMS, VTEIMC, VTEIAC, VTEIAM, VTEICR, VTIVAR0, VTIVAR_MISC, VFGPRLBC, VFGPTLBC, VFGORLBC, VFGOTLBC and IOVCTL. Removed following registers due to removal of SR-IOV: VFMailbox[0 - 7], PFMailbox[0 - 7], VMBMEM, MBVFICR, MBVFIMR, VFLRE, VFRE, VFTE, QDE and VMVIR

Table A-1. Changes in the Programming Interface Relative to the 82576 (Continued)

Feature	Registers/Descriptors	Description
Virtualization	Virtualization TX switch buffer registers	Removed following registers due to removal of virtualization TX switch buffer: SWPBS, PBSWAC, DTXSWC, SWBFH, SWBFT, SWBFHS, SWBFTS, SWDFPC, SWPBECCSTS, SSVPC and SDPC.
	VMECM, SSVPC and WVBR	Registers removed due to removal of anti spoofing protection: VMECM, SSVPC and WVBR
	CIAA and CIAD	Removed CIAA and CIAD diagnostic registers due to removal of SR-IOV.
	GCR	GCR register changes due to removal of SR-IOV. Bit "IOV test mode" (bit 1) changed to reserved Bit "Ignore RID" (bit 0) changed to reserved
	DTXCTL	Bit SPOOF_INT (bit 6) changed to reserved, due to removal of anti spoofing protection.
	CTRL_EXT	PFRSTD (bit14) changed to reserved due to SR-IOV support removal.
	Advanced receive descriptors – Write Back format	Advanced receive descriptors – Write Back format LB (17) bit in "Extended Status" field changed to Reserved due to no support of VM to VM switch.
	STATUS	Changed "Num VFs" (bits 17:14) and "IOV mode" (bit 18) fields in the STATUS register to reserved.
	ICR, ICS, IMS and IMC registers	Changed *VMMB* bit (bit 8) and *XOUTSYNC* (bit 29) to reserved in *ICR*, *ICS*, *IMS* and *IMC* registers.
	PEIND	PEIND Register changes due to Switch support removal • "dbu_ecc_sw_pb_nfatal" (bit 0) changed to reserved • "dbu_ecc_sw_pb_fatal" (bit 27) changed to reserved
	PEINDM	PEINDM Register changes due to Switch support removal • "dbu_ecc_sw_pb_nfatal" (bit 0) changed to reserved • "dbu_ecc_sw_pb_fatal" (bit 27) changed to reserved
	VMRCTL	Changed DPME (bit 2) in VMRCTL register to reserved due to Switch removal
	VLVF	Bit *LVLAN* (Bit 20) in *VLVF* register changed to reserved due to removal of virtualization TX switch.
	PCIe capability Structures	• No support for "IOV Capability Structure" registers and "Alternative RID Interpretation (ARI) capability structure" in extended PCIe configuration address space due to SR-IOV removal. • Modified default values of "Next Capability pointer" field (Bits 31:20) in PCIe CAP ID (0x100; RO) PCIe configuration register due to SR-IOV/ARI support removal. • No support for "Virtual Functions (VF) Configuration Space" registers defined in "PCIe Programming Interface" chapter in the 82576 due to SR-IOV removal • Modified default value of "Next Capability Offset" field (Bits 31:20) in "Device Serial Number Enhanced Capability Header" PCIe configuration register due to SR-IOV/ARI support removal.

Table A-1. Changes in the Programming Interface Relative to the 82576 (Continued)

Feature	Registers/Descriptors	Description
DCE and BCN removal	DCE and BCN registers	Removed following registers due to no DCE support: • RTTDCS, RTTPCS, RTRPCS, RTRUP2TC, RTTUP2TC, RTTDTCRC, RTTPTCRC, RTRPTCRC, RTTDTCRS, RTTDTCRM, RTTPTCRS, RTTPTCRM, RTRPTCRS, RTRPTCRM, RTTDVMRM, RTTBCNRM, RTTDQSEL, RTTDVMRC, RTTDVMRS, RTTBCNR, RTTBCNRS, RTTBCNCR, RTTBCNTG, RTTBCNCP, RTTBCNRD, PFCTOP, RTTBCNIDX, RTTBCNACH, RTTBCNACL, FCRTL1, FCRTH1, FCSTS1 and PFCTOP • Removed following statistical registers due to no DCE support: PXONTXC [0-7], PXONRXC [0-7], PXOFFTXC [0-7], PXOFFRXC [0-7] and PXON2OFFC [0-1]
	FCTTV	Changed TTVTC1 field (Bits 31:16) in FCTTV register to reserved due to no DCE support.
DCE and BCN removal	Advanced receive descriptors – Write Back format	RTT (Round Trip Time) not supported in “RSS Hash Value/Fragment Checksum and IP identification” field due to removal of BCN support.
	Advanced Transmit Context Descriptor (DTYP = 2)	“BCNTLEN” (bits 35:30) field reserved due to BCN support removal
	Advanced Transmit Data Descriptor (DTYP = 3) – Read format:	“CC” (bit 39) field reserved due to BCN support removal.
	ETQF	Removed BCN frame (Bit 28 is reserved) from ETQF register.
	DRXMXOD	Removed DRXMXOD register due to no receive priority support
	MRQC	Following values of “Multiple Receive Queues Enable” field in MRQC register have been reserved. • 001 – Reserved due to DCE removal • 100 – Reserved due to DCE removal • 101 – Reserved due to no RSS support in virtualization • 110 – Reserved due to DCE removal MRQC.Def_Q values are ignored when above reserved values of MRQC. Multiple Receive Queues Enable field are used.
	CTRL_EXT	CTRL_EXT register changes: • TX_RT_EN (bit 29) changed to reserved due to DCE support removal. • RS_RT_EN (bit 27) changed to reserved due to DCE support removal.
	CTRL	Changed RPFCE (bit 14) and TPFCE (bit 15) in CTRL register to reserved due to priority flow control support removal.

Table A-1. Changes in the Programming Interface Relative to the 82576 (Continued)

Feature	Registers/Descriptors	Description
IPsec and LinkSec removal	LinkSec registers	Removed following registers due to no LinkSec support: • LSECTXCAP, LSECRXCAP, LSECTXCTRL, LSECRXCTRL, LSECTXSCL, LSECTXSCH, LSECTXSA, LSECTXPN0, LSECTXPN1, LSECTXKEY0, LSECTXKEY1, LSECRXSCL[n], LSECRXSCH[n], LSECRXSA, LSECRXSAPN and LSECRXKEY[n, m]. • LSECTXUT, LSECTXPKTE, LSECTXPKTP, LSECTXOCTE, LSECTXOCTP, LSECRXUT, LSECRXOCTE, LSECRXOCTP, LSECRXBAD, LSECRXNOSCI, LSECRXUNSCI, LSECRXUNCH, LSECRXDELAY, LSECRXOK[n], LSECRXINV[n], LSECRXNV[n], LSECRXUNSA and LSECRXNUSA • LSWFW
	IPSec registers	Removed following registers due to no IPSec support: • IPSRXCMD, IPSRXIDX, IPSRXIPADDR, IPSRXKEY, IPSRXSALT, IPSRXSPI, IPSTXIDX, IPSTXKEY, IPSTXSALT and IPSCTRL. Removed following diagnostic registers due to no IPSec support: • IPPBECCSTS and IPPBEEI
	Advanced receive descriptors – Write Back format	Advanced receive descriptors – Write Back format modifications: • Changed bits 10:8 of "Packet Type" field to Reserved. • SECP (17) bit in "Extended Status" field changed to Reserved. • SECERR (8:7) bits in "Extended Error" field changed to reserved.
	Advanced Transmit Context Descriptor (DTYP = 2)	Advanced Transmit Context Descriptor (DTYP = 2) modifications: • "IPsec SA Index" (bits 39:32) field changed to reserved due to IPsec support removal. • "IPSec ESP_LEN" (bits 8:0) field changed to reserved due to IPsec support removal. Removed following bits from TUCMD field due to IPsec support removal: • Encryption (bit 5) • IPSEC Type (bit 4)

Table A-1. Changes in the Programming Interface Relative to the 82576 (Continued)

Feature	Registers/Descriptors	Description
IPsec and LinkSec removal	Advanced Transmit Data Descriptor (DTYP = 3) read format	Advanced Transmit Data Descriptor (DTYP = 3) read format modifications: • ILSEC bit (bit 18) in MAC field changed to reserved due to no LinkSec support. • IPSEC bit in POPTS field (bit 2 in POPTS field) reserved due to no IPsec support.
	PEIND	PEIND Register changes due to IPSec support removal: • "mac_parity_secu_rx_sa_ fatal" bit (bit 9) changed to reserved • "mac_parity_secu_tx_sa_ fatal" bit (bit 10) changed to reserved • "mac_parity_secu_post_l 4cs_fatal" bit (bit 20) changed to reserved • "mac_parity_secu_pre_hd r_parse_fatal" bit (bit 21) changed to reserved • "mac_parity_secu_post_s ta_fatal" bit (bit 22) changed to reserved • "mac_ecc_secu_tx_pb_fat al" bit (bit 30) changed to reserved
	PEINDM	PEINDM Register changes due to IPSec support removal: • "mac_parity_secu_rx_sa_ fatal" bit (bit 9) changed to reserved • "mac_parity_secu_tx_sa_ fatal" bit (bit 10) changed to reserved • "mac_parity_secu_post_l 4cs_fatal" bit (bit 20) changed to reserved • "mac_parity_secu_pre_hd r_parse_fatal" bit (bit 21) changed to reserved • "mac_parity_secu_post_s ta_fatal" bit (bit 22) changed to reserved • "mac_ecc_secu_tx_pb_fat al" bit (bit 30) changed to reserved
	HHR	HHR register changes: RX_BYP (bit 6) changed to reserved due to IPSec support removal TX_BYP (bit 7) changed to reserved due to IPSec support removal
	SECUCTRL	Removed SECUCTRL control register due to IPsec and LinkSec removal
	RFCTL	Changed *RFCTL.Rx filter FIFO WaterMark* (bits 23:20) field to reserved due to removal of IPSec and LinkSec support.
MSIx vector reduction from 25 to 10	EITR, MSIXTADD, MSIXTUADD, MSIXTMSG and MSIXTVCTRL	Size of MSI-X table reduced from 25 entries to 10 entries as a result reduced number of following MSIx related registers from 25 to 10: • EITR, MSIXTADD, MSIXTUADD, MSIXTMSG and MSIXTVCTRL
	MSIXPBA	Changed number of "Pending Bits" in MSIXPBA register from 25 to 10 (bits 9:0).
	PBACL	Changed number of "PENBIT" bits in PBACL register from 25 to 10 (bits 9:0).
	IVAR and IVAR_MISC	Width of INT_Alloc[i] fields in IVAR and IVAR_MISC registers changed to 4 bits from 5 to support only 10 MSIx vectors instead of 25 MSIx vectors.
	EICR, EICS, EIMS, EIMC, EIAC and EIAM	Width of "MSIx" field in EICR, EICS, EIMS, EIMC, EIAC and EIAM registers (when GPIE.Multiple_MSIX = 1) changed to 10 bits instead of 25 bits due to MSIx vector reduction.

Table A-1. Changes in the Programming Interface Relative to the 82576 (Continued)

Feature	Registers/Descriptors	Description
TPH support	TXCTL	Added the following fields: • Tx Descriptor fetch TPH EN (bit 0) • Tx Descriptor writeback TPH EN (bit 1) • Tx Packet TPH EN (bit 3) Modified width of CPUID field to 8 bits (bits 31:24)
	RXCTL	Added the following fields: • RX Descriptor fetch TPH EN (bit 0) • RX Descriptor writeback TPH EN (bit 1) • Rx Header TPH EN (bit 2) • Rx Payload TPH EN (bit 3) Modified width of CPUID field to 8 bits (bits 31:24)
	DCA_CTRL	Added Desc_PH (bits 10:9) and Data_PH (bits 12:11) for TPH support.
	Device Capabilities 2	Added TPH bits in PCIe Device Capabilities 2 configuration register for TPH support.
	TPH Capability structure	Added PCIe TLP Processing Hint (TPH) Capability structure for TPH support.
Embedded features	FHFT_EXT and FHFT WoL Flex Filter Tables.	• Added 2 additional WoL Flex filters in *FHFT_EXT* Table to support queueing in D0 state. • Added Pool Select field (bits 15:8) in the DW before last of the WoL Flex Filters (*FHFT* and *FHFT_EXT*) to support queueing in D0 state.
	WUFC and WUS registers	• Added support for *FLX6* (bit 22) and *FLX7* (bit 23) in WUS and WUFC registers. • Added *FLEX_HQ* bit to *WUFC* register to enable usage of WoL Flex filters for queueing decisions in D0 state.
	SRRCTL	Added SRRCTL[n].Timestamp bit (bit 30) to enable time stamping of receive packets per queue.
	RDESC	New RDESC.TSIP bit (bit 15) in receive descriptor is set when packet is time stamped and placed in RX buffer. HDR_LEN and PKT_LEN are updated with additional timestamp bytes.
	TSAUXC	Fix the max allowed size of the TSAUXC.Mask to 24 and not 16 so it can cover the max IncValue size.
5-tuple filter removal	SAQF, DAQF, SPQF and FTQF/TTQF	Replaced 5-Tuple filtering support with 2-Tuple filtering support. Only TCP Destination Port and IP L4 protocol fields are checked. Removed registers SAQF, DAQF and SPQF registers. Functionality of FTQF register modified and register name changed to TTQF.
Manageability	MANC	MANC register changes due to exclusive management filtering change: • *RCV_ALL* (bit 19): Debug mode only. Changed to promiscuous mode. • *RCV_ALL_MCST* (bit 20), *MNG2HOST_EN* (bit 21), *Bypass VLAN* (bit 22) and *METF_MODE* (bit 27) changed to reserved. • FW_RESET (bit 14) - Added bit that indicates occurrence of Firmware reset due to BMC Fimrware reset command. •
	MANC2HOST	Removed MANC2HOST (0x5860) register due to exclusive management filtering change.
	MNGONLY	Added MNGONLY (0x5864) register due to exclusive management filtering change.
	FWSM	Updated FWSM.Ext_Err_Ind Firmware error indications.

Table A-1. Changes in the Programming Interface Relative to the 82576 (Continued)

Feature	Registers/Descriptors	Description
Manageability	MFUTP	Reduced MFUTP (Management UDP/TCP flex port filters) registers to 4 from 8, to support only 8 such filters per port instead of 16.
	MDEF, MDEF_EXT	• MDEF.Flex port width reduced to 8 bits (bits 19:12) to support 8 manageability Port Flex filters instead of 16. • MDEF.Flex TCO width reduced to 1 bit (bit 28) to support single manageability TCO flex filter per port instead of 4 per port. • Updated MDEF and MDEF_EXT registers to enable filtering when only certain combinations of VLAN and MAC addresses match. • Added to MDEF_EXT decision filter register the Traffic Source (Bit 31) bit to support OS to BMC traffic. • Added NC-SI Discard bit (bit 28) and Flow Control Discard (bit 29) to MDEF_EXT register. • Added NC-SI Discard bit (bit 28) and Flow Control Discard (bit 29) to MDEF_EXT register.
	MMAH, MMAL	Reduced number of manageability MAC Address filters (MMAH, MMAL) to 2 per port instead of 4 per port.
	FTFT	Reduced number of manageability TCO Flex filters (*FTFT*) to 1per port instead of 4 per port.
	BUPRC, BMPRC, BBPRC, BUPTC, BMPTC, BBPTC, BCRCERRS, BALGNERRC, BXONRXC, BXOFFRXC, BXONTXC, BXOFFTXC, BSCC and BMCC.	Added new BMC statistical registers BUPRC, BMPRC, BBPRC, BUPTC, BMPTC, BBPTC, BCRCERRS, BALGNERRC, BXONRXC, BXOFFRXC, BXONTXC, BXOFFTXC, BSCC and BMCC.
	SWSM	Removed WMNG (bit 3) and EEUR (bit 4) bits from SWSM register.
	MFVAL	Removed MFVAL register due to MDEF and MDEF_EXT per filter change.
Power saving	SRRCTL	Added SRRCTL.DMACQ_Dis bit (Bit 7) to support DMA Coalescing operation.
	DMACR, FCRTC, DMCRTRH, DMCTLx, DMCTXTH, and DMCCN	Added DMACR,FCRTC DMCRTRH, DMCTLx, DMCTXTH and DMCCN registers to support DMA Coalescing.
	LTRMINV, LTRMAXV and LTRC	Added LTRMINV, LTRMAXV and LTRC registers for PCIe LTR (Latency Tolerance Reporting) support
	Device Capabilities 2	• Added bit 9 in Device Capabilities 2 configuration register for LTR support.
	Device Control 2	• Added bit 10 to Device Control 2 Configuration register for LTR support.
	Latency Tolerance Requirement Reporting (LTR) Capability structure	Added Latency Tolerance Requirement Reporting (LTR) Capability structure in address 0x1C0.
	CTRL	Removed bit ADVD3WUC (bit 20) from CTRL register. The D3COLD_WAKEUP_ADVEN EEPROM bit controls report of D3 cold wake-up support in PMC configuration register, when AUX_PWR is available.
Memory Check	ICR, ICS, IMS and IMC registers	Removed NFER bit (bit 23) from ICR, ICS, IMS and IMC registers.
	PEIND and PEINDM	Updated PEIND and PEINDM functionality to reflect detection of memory parity errors in different sections of the 82580.

Table A-1. Changes in the Programming Interface Relative to the 82576 (Continued)

Feature	Registers/Descriptors	Description
TimeSync	SYSTIMR	Added SYSTIMR register.
	TIMINCA	Removed Incperiod field from TIMINCA register, added Increment Sign bit (bit 31) and increased Incvalue field width to 31 bits.
	SYTIMH, TIMADJH, RXSTMPH, TXSTMPH, TRGTTIMH0, TRGTTIMH1, AUXSTMPH0 and AUXSTMPH1	Reduced width of SYTIMH.STH, TIMADJH.TADJH, RXSTMPH.RTSH, TXSTMPH.TTSH, TRGTTIMH0.TTH, TRGTTIMH1.TTH, AUXSTMPH0.TSTH and AUXSTMPH1.TSTH fields to 8 bits.
	TSAUXC	TSAUXC register modified): - TSAUXC.Mask field (Bits 16:12) removed due to change in clock generation logic. - Bits TSAUXC.UTT0 (bit 3) and TSAUXC.UTT1 (bit 6) removed due to change in clock generation logic. - Added PLSG0 (bit 17), PLSNeg0 (bit 18), PLSG0 (bit 19) and PLSNeg0 (bit 20) to enable pulse generation. - Added "Disable systime" bit (bit 31) to TSAUXC register, to keep constant value in SYSTIMH/L/R registers during write.
	FREQOUT0 and FREQOUT1	Changed functionality of RLD field and renamed it to CHCT. Width of field changed to 8 bits (bits 7:0) in FREQOUT0 and FREQOUT1 registers.
Miscellaneous	DTXMXPKTSZ	Added new DTXMXPKTSZ register to define maximum allowable transmit packet size. Smaller packets allow better utilization of transmit buffer.
	FCRTH0	Value of FCRTH0.RTH field needs to be updated to take into account increased internal RX buffer size of 32 KB (See Section 4.6.6).
	LVMMC	LVMMC register modified for improved malicious driver detection. IPSec, LinkSec, RT, IOV and Spoof detect related bits removed.
	TXDCTL	TXDCTL register changes • Added new "Priority" bit (bit 27) to define low or high priority for a specific transmit queue to support applications that require low latency. • Added *TXDCTL. HWBTHRESH* field to support transmit head write-back coalescing to reduce PCIe utilization.
	ICR,ICS,IMS and IMC	• Changed name of RXO bit (bit 6) to *RX Miss* in ICR, ICS, IMS and IMC registers. • Renamed CSRTO bit (bit 24) to *PCI Exception* in ICR, ICS, IMS and IMC registers.
	RQDPC	RQDPC updated to 32 bits (12 bits in the 82576) to support longer duration between polls. Register is read by clear and can be written to, for diagnostic purposes.
	I2CPARAMS	Added CLK_OE_N bit (bit 13), CLK_IN bit (bit 14) and clk_stretch_dis bit (bit 15) to I2CPARAMS register to support clock stretching
	CONNSW	Added PHY_PDN bit (bit 11) to the "Copper/Fiber Switch Control" - CONNSW (0x0034; R/W) register to indicate internal GbE PHY in power down.
	DTXCTL	Changed "DTXCTL.NoSnoop_LSO_hdr_buf" (bit 1) and "DTXCTL.Add VLAN location" (bit 3) to reserved.
	EEC	Added EE_BLOCKED bit (bit 15) to EEC register, to support detection of EEPROM bit-banging access blocked occurrence. See Section 3.3.1.4.

§ §