Architectural Overview 2

2.1 Introduction

This section provides an overview of the PCI/PCI-X Family of Gigabit Ethernet Controllers. The following sections give detailed information about the Ethernet controller's functionality, register description, and initialization sequence. All major interfaces of the Ethernet controllers are described in detail.

The following principles shaped the design of the PCI/PCI-X Family of Gigabit Ethernet Controllers:

1. Provide an Ethernet interface containing a 10/100/1000 Mb/s PHY that also supports 1000 Base-X implementations.
2. Provide the highest performance solution possible, based on the following:
    - — Provide direct access to all memory without using mapping registers
    - — Minimize the PCI target accesses required to manage the Ethernet controller
    - — Minimize the interrupts required to manage the Ethernet controller
    - — Off-load the host processor from simple tasks such as TCP checksum calculations
    - — Maximize PCI efficiency and performance
    - — Use mixed signal processing to assure physical layer characteristics surpass specifications for UTP copper media
3. Provide a simple software interface for basic operations.
4. Provide a highly configurable design that can be used effectively in different environments.

The PCI/PCI-X Family of Gigabit Ethernet Controllers architecture is a derivative of the 82542 and 82543 designs. They take the MAC functionality and integrated copper PHY from their predecessors and adds SMBus-based manageability and integrated ASF controller functionality to the MAC[1]. In addition, the **82546GB/EB** features this architecture in an integrated dual-port solution comprised of two distinct MAC/PHY instances.

1. Not applicable to the **82544GC/EI** or **82541ER**.

架构概述 2

2.1 简介

本节提供了 PCI/PCI-X 千兆以太网控制器系列的概述。以下各节将详细说明以太网控制器的功能、寄存器描述和初始化序列。以太网控制器的所有主要接口都进行了详细描述。

以下原则塑造了 PCI/PCI-X 千兆以太网控制器系列的设计:

1. 提供一个包含 10/100/1000 Mbps PHY 的以太网接口，该接口还支持 1000 Base-X 实现。
2. 提供基于以下条件的最高性能解决方案:
    - — 提供对内存的直接访问，而无需使用映射寄存器
    - — 最小化管理以太网控制器所需的 PCI 目标访问
    - — 最小化管理以太网控制器所需的中断
    - — 将主机处理器从简单的任务（如 TCP 校验和计算）中卸载
    - — 最大程度提高PCI效率和性能
    - — 使用混合信号处理，确保物理层特性超过UTP铜介质的规格
3. 提供一个用于基本操作的简单软件接口。
4. 提供一个高度可配置的设计，可以在不同环境中有效使用。

PCI/PCI-X 千兆以太网控制器架构是82542和82543设计的衍生品。它们继承了前代的MAC功能和集成铜PHY，并将基于SMBus的管理能力和集成ASF控制器功能添加到MAC[1]。此外，82546GB/EB采用这种架构，以一个由两个独立的MAC/PHY实例组成的集成双端口解决方案呈现。

1. 不适用于82544GC/EI或82541ER。

2.2 External Architecture

Figure 2-1 shows the external interfaces to the **82546GB/EB**.

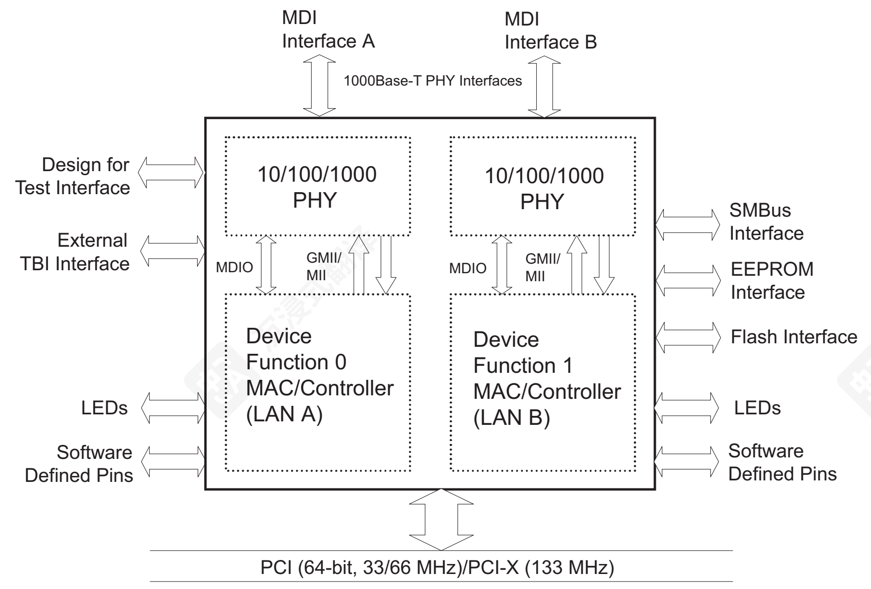
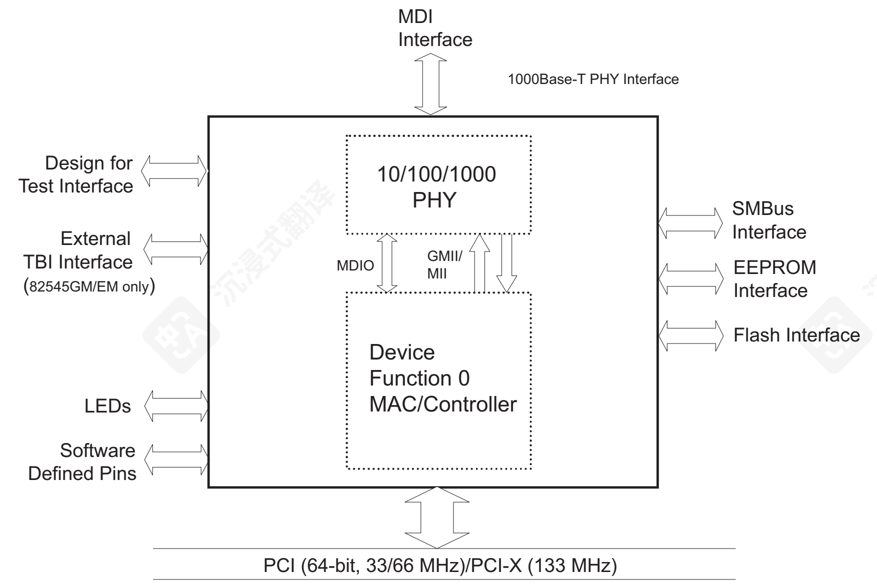


Figure 2-1. 82546GB/EB External Interface

Figure 2-2 shows the external interfaces to the **82545GM/EM**, **82544GC/EI**, **82540EP/EM**, and **82541xx**.


MDI Interface
1000Base-T PHY Interface
Design for Test Interface
External TBI Interface (82545GM/EM only)
LEDs
Software Defined Pins
10/100/1000 PHY
MDIO
GMII/ MII
Device Function 0 MAC/Controller
SMBus Interface
EEPROM Interface
Flash Interface
PCI (64-bit, 33/66 MHz)/PCI-X (133 MHz)

Note: 82540EP/EM and 82541xx do not support PCI-X; 82544GC/EI and 82541ER do not support SMBus interface

Figure 2-2. 82545GM/EM, 82544GC/EI, 82540EP/EM, and 82541xx External Interface

2.2 外部架构

图2-1显示了82546GB/EB的外部接口。

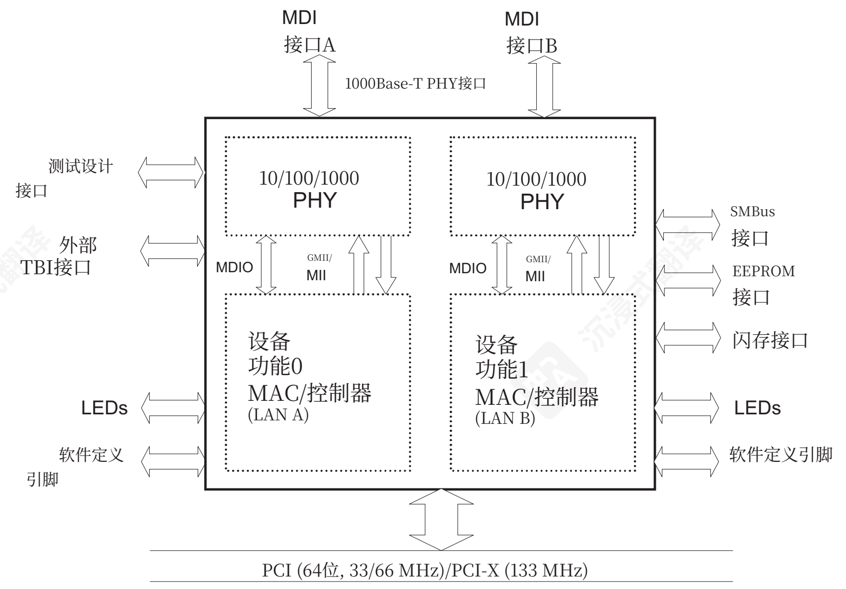
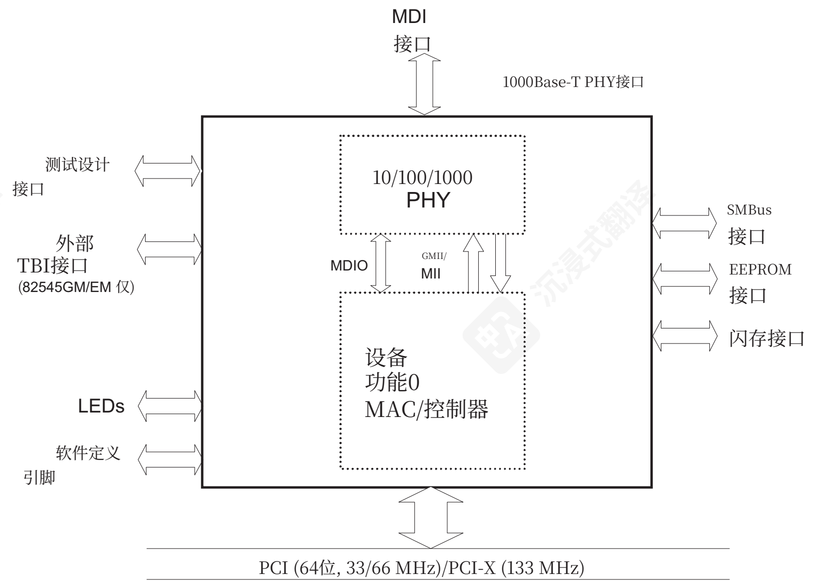


图2-1. 82546GB/EB外部接口

图2-2显示了82545GM/EM、82544GC/EI、82540EP/EM和82541xx的外部接口。



注意：82540EP/EM 和 82541xx 不支持 PCI-X；82544GC/EI 和 82541ER 不支持 SMBus 接口

图2-2。82545GM/EM、82544GC/EI、82540EP/EM和82541xx外部接口

Figure 2-3 shows the external interfaces to the **82547GI/EI**.

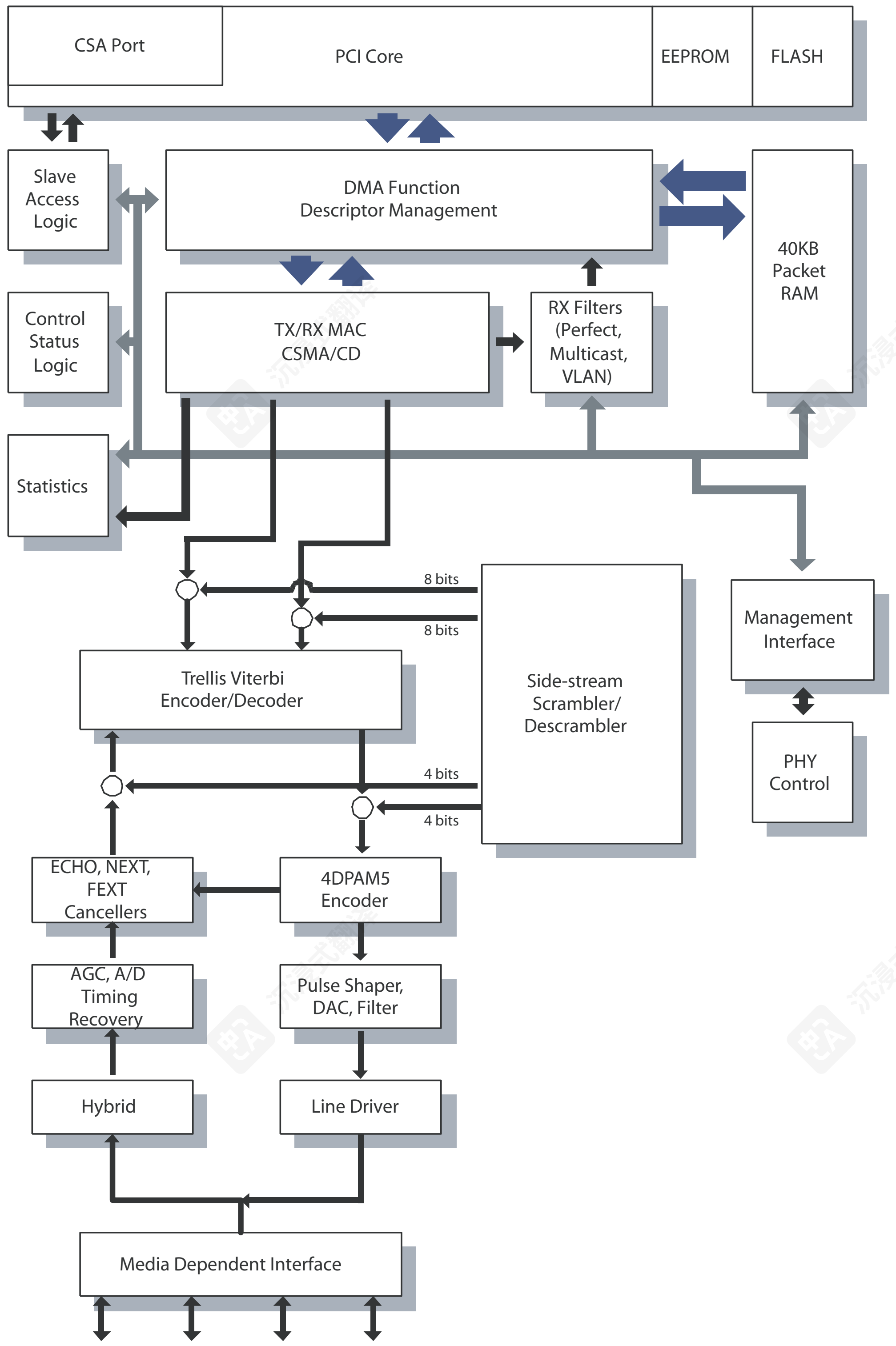


Figure 2-3. 82547GI(EI) External Interface

图2-3显示了82547GI/EI的外部接口。

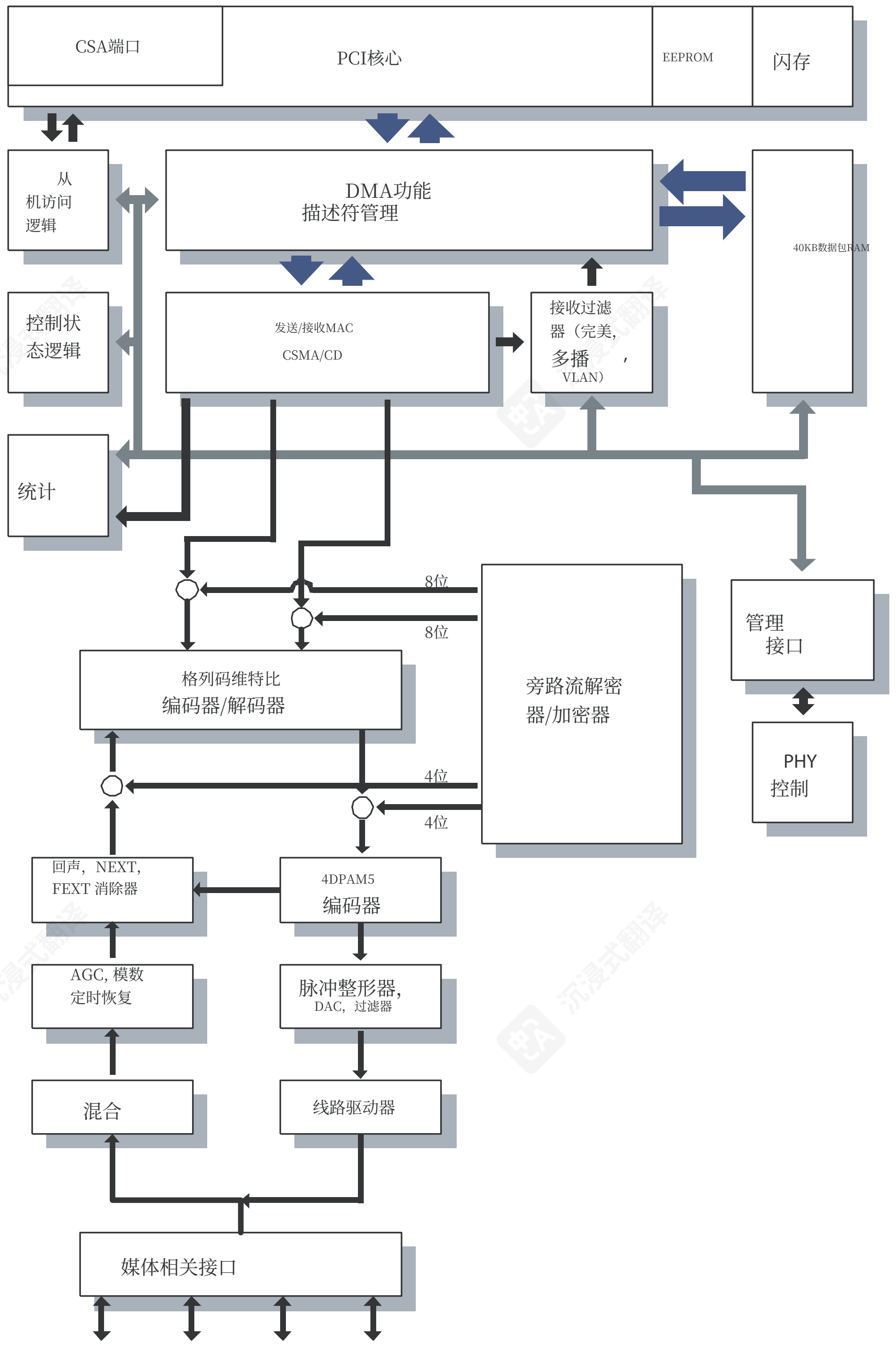


图2-3. 82547GI(EI)外部接口

2.3 Microarchitecture

Compared to its predecessors, the PCI/PCI-X Family of Gigabit Ethernet Controller's MAC adds improved receive-packet filtering to support SMBus-based manageability, as well as the ability to transmit SMBus-based manageability packets. In addition, an ASF-compliant TCO controller is integrated into the controller's MAC for reduced-cost basic ASF manageability.

Note: The **82544GC/EI** and **82541ER** do not support SMBus-based manageability.

For the **82546GB/EB**, this new functionality is packaged in an integrated dual-port combination. The architecture includes two instances of both the MAC and PHY along with a single PCI/PCI-X interface. As a result, each of the logical LAN devices appear as a distinct PCI/PCI-X bus device.

The following sections describe the hardware building blocks. Figure 2-4 shows the internal microarchitecture.

2.3.1 PCI/PCI-X Core Interface

The PCI/PCI-X core provides a complete glueless interface to a 33/66 MHz, 32/64-bit PCI bus or a 33/66/133 MHz, 32/64 bit PCI-X bus. It is compliant with the PCI Bus Specification Rev 2.2 or 2.3 and the PCI-X Specification Rev. 1.0a. The Ethernet controllers provide 32 or 64 bits of addressing and data, and the complete control interface to operate on a 32-bit or 64-bit PCI or PCI-X bus. In systems with a dedicated bus for the Ethernet controller, this provides sufficient bandwidth to support sustained 1000 Mb/s full-duplex transfer rates. Systems with a shared bus (especially the 32-bit wide interface) might not be able to maintain 1000 Mb/s, but can sustain multiple hundreds of Mbps.

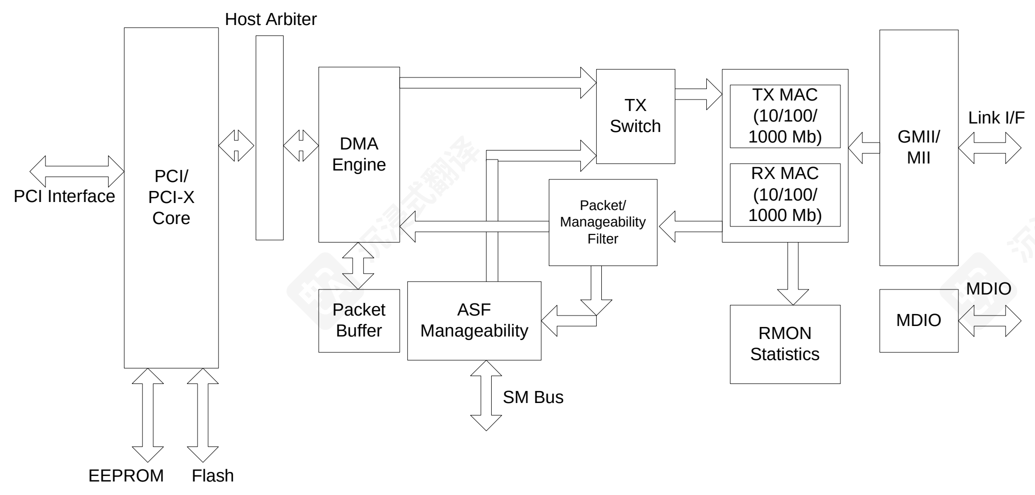


Figure 2-4. Internal Architecture Block Diagram

2.3 微架构

与它的前辈相比，千兆以太网控制器的PCI/PCI-X系列的MAC增加了改进的接收数据包过滤，以支持基于SMBus的管理能力，以及传输基于SMBus的管理能力数据包的能力。此外，一个符合ASF规范的TCO控制器被集成到控制器的MAC中，以降低成本的基础ASF管理能力。

注意：82544GC/EI和82541ER不支持基于SMBus的管理能力。

对于 82546GB/EB，这项新功能被集成在一个双端口组合中。该架构包括两个 MAC 和 PHY 的实例，以及一个 PCI/PCI-X 接口。因此，每个逻辑 LAN 设备都表现为一个独立的 PCI/PCI-X 总线设备。

以下部分描述了硬件构建模块。图 2-4 显示了内部微架构。

2.3.1 PCI/PCI-X 核心

PCI/PCI-X核心为33/66 MHz、32/64位PCI总线或33/66/133 MHz、32/64位PCI-X总线提供完整的无粘合接口。它符合PCI总线规范Rev 2.2或2.3和PCI-X规范Rev. 1.0a。以太网控制器提供32或64位的寻址和数据，以及完整的控制接口，用于在32位或64位PCI或PCI-X总线上操作。在以太网控制器具有专用总线的情况下，这提供了足够的带宽以支持持续的1000 Mb/s全双工传输速率。具有共享总线（尤其是32位宽接口）的系统可能无法维持1000 Mb/s，但可以持续支持数百Mbps。

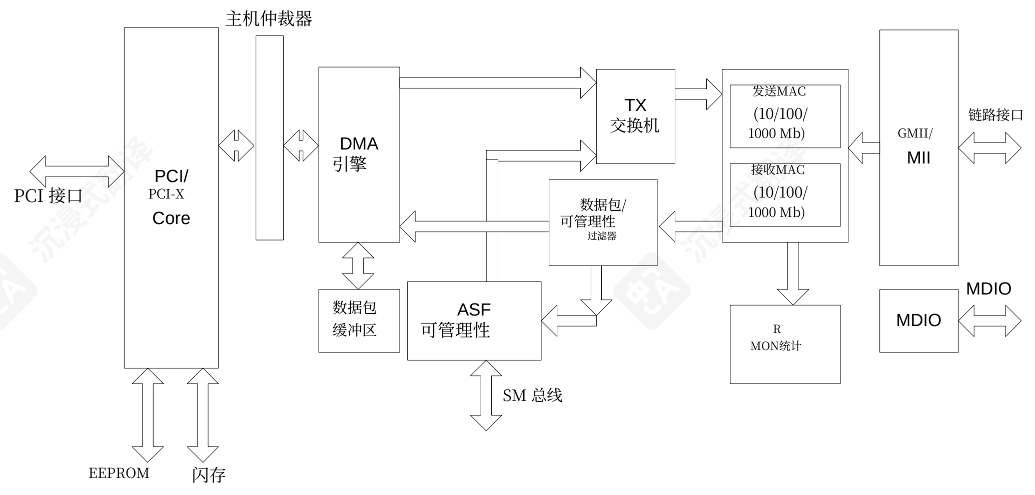


图 2-4。内部架构框图

When the Ethernet controller serves as a PCI target, it follows the PCI configuration specification, which allows all accesses to it to be automatically mapped into free memory and I/O space at initialization of the PCI system.

When processing transmit and receive frames, the Ethernet controller operates as master on the PCI bus. As a master, transaction burst length on the PCI bus is determined by several factors, including the PCI latency timer expiration, the type of bus transfer being made, the size of the data transfer, and whether the data transfer is initiated by receive or transmit logic.

The PCI/PCI-X bus interfaces to the DMA engine.

2.3.2 82547GI/EI CSA Interface

CSA is derived from the Intel® Hub Architecture. The 82547EI Controller CSA port consists of 11 data and control signals, two strobes, a 66 MHz clock, and driver compensation resistor connections. The operating details of these signals and the packet data protocol that accompanies them are proprietary. The CSA port has a theoretical bandwidth of 266 MB/s — approximately twice the peak bandwidth of a 32-bit 33 MHz PCI bus.

The CSA port architecture is invisible to both system software and the operating system, allowing conventional PCI-like configuration.

2.3.3 DMA Engine and Data FIFO

The DMA engine handles the receive and transmit data and descriptor transfers between the host memory and the on-chip memory.

In the receive path, the DMA engine transfers the data stored in the receive data FIFO buffer to the receive buffer in the host memory, specified by the address in the descriptor. It also fetches and writes back updated receive descriptors to host memory.

In the transmit path, the DMA engine transfers data stored in the host memory buffers to the transmit data FIFO buffer. It also fetches and writes back updated transmit descriptors.

The Ethernet controller data FIFO block consists of a 64 KB (40 KB for the **82547GI/EI**) on-chip buffer for receive and transmit operation. The receive and transmit FIFO size can be allocated based on the system requirements. The FIFO provides a temporary buffer storage area for frames as they are received or transmitted by the Ethernet controller.

The DMA engine and the large data FIFOs are optimized to maximize the PCI bus efficiency and reduce processor utilization by:

- Mitigating instantaneous receive bandwidth demands and eliminating transmit underruns by buffering the entire out-going packet prior to transmission
- Queuing transmit frames within the transmit FIFO, allowing back-to-back transmission with the minimum interframe spacing
- Allowing the Ethernet controller to withstand long PCI bus latencies without losing incoming data or corrupting outgoing data
- Allowing the transmit start threshold to be tuned by the transmit FIFO threshold. This adjustment to system performance is based on the available PCI bandwidth, wire speed, and latency considerations

当以太网控制器作为PCI目标时，它遵循PCI配置规范，该规范允许所有对其访问在PCI系统初始化时自动映射到空闲内存和I/O空间。

在处理传输和接收帧时，以太网控制器在PCI总线上作为主设备运行。作为主设备，PCI总线上的事务突发长度由多个因素决定，包括PCI延迟计时器超时、正在进行的总线传输类型、数据传输的大小，以及数据传输是由接收逻辑还是传输逻辑发起。

PCI/PCI-X总线接口到DMA引擎。

2.3.2 82547GI/EI CSA接口

CSA源自Intel®芯片组架构。82547EI控制器CSA端口包括11个数据和控制信号、两个选通信号、一个66 MHz时钟和驱动补偿电阻连接。这些信号的操作细节以及伴随它们的包数据协议是专有的。CSA端口的理论带宽为266 MB/s，大约是32位33 MHz PCI总线峰值带宽的两倍。

CSA端口架构对系统软件和操作系统都是不可见的，允许传统的PCI-like配置。

2.3.3 DMA引擎和数据 FIFO

DMA引擎处理主机内存和片上内存之间的接收和传输数据以及描述符传输。

在接收路径中，DMA引擎将接收数据 FIFO 缓冲区中存储的数据传输到主机内存中由描述符地址指定的接收缓冲区。它还获取并写回更新后的接收描述符到主机内存。

在传输路径中，DMA引擎将主机内存缓冲区中存储的数据传输到传输数据 FIFO 缓冲区。它还获取并写回更新后的传输描述符。

以太网控制器数据 FIFO 模块由一个64 KB（82547GI/EI为40 KB）的片上缓冲区组成，用于接收和传输操作。接收和传输 FIFO 的大小可以根据系统需求进行分配。FIFO为以太网控制器接收或传输的帧提供临时缓冲区存储区域。

DMA引擎和大型数据FIFO被优化以最大化PCI总线效率并减少处理器利用率，具体如下：

- 通过在传输前缓冲整个外发数据包来缓解瞬时接收带宽需求并消除传输欠载
- 在发送FIFO中排队发送帧，允许以最小帧间间隔进行连续传输
- 允许以太网控制器在不会丢失入站数据或损坏出站数据的情况下承受较长的PCI总线延迟
- 允许发送启动阈值由发送FIFO阈值调整。此系统性能调整基于可用的PCI带宽、线速和延迟考虑

- Offloading the receiving and transmitting IP and TCP/UDP checksums
- Directly retransmitting from the transmit FIFO any transmissions resulting in errors (collision detection, data underrun), thus eliminating the need to re-access this data from host memory

2.3.4 10/100/1000 Mb/s Receive and Transmit MAC Blocks

The controller's CSMA/CD unit handles all the IEEE 802.3 receive and transmit MAC functions while interfacing between the DMA and TBI/internal SerDes/MII/GMII interface block. The CSMA/CD unit supports IEEE 802.3 for 10 Mb/s, IEEE 802.3u for 100 Mb/s and IEEE 802.3z and IEEE 802.3ab for 1000 Mb/s.

The Ethernet controller supports half-duplex 10/100 Mb/s MII or 1000 Mb/s GMII mode and all aspects of the above specifications in full-duplex operation. In half-duplex mode, the Ethernet controller supports operation as specified in IEEE 802.3z specification. In the receive path, the Ethernet controller supports carrier extended packets and packets generated during packet bursting operation. The **82554GC/EI**, in the transmit path, also supports carrier extended packets and can be configured to transmit in packet burst mode.

The Ethernet controller offers various filtering capabilities that provide better performance and lower processor utilization as follows:

- Provides up to 16 addresses for exact match unicast/multicast address filtering.
- Provides multicast address filtering based on 4096 bit vectors. Promiscuous unicast and promiscuous multicast filtering are supported as well.
- The Ethernet controller strips IEEE 802.1q VLAN tag and filter packets based on their VLAN ID. Up to 4096 VLAN tags are supported[1].

In the transmit path, the Ethernet controller supports insertion of VLAN tag information, on a packet-by-packet basis.

The Ethernet controller implements the flow control function as defined in IEEE 802.3x, as well as specific operation of asymmetrical flow control as defined by IEEE 802.3z. The Ethernet controller also provides external pins for controlling the flow control function through external logic.

2.3.5 MII/GMII/TBI/Internal SerDes Interface Block

The Ethernet controller provides the following serial interfaces:

- A GMII/MII interface to the internal PHY.
- Internal SerDes interface[2] (**82546GB/EB** and **82545GM/EM**)/Ten Bit Interface (TBI)[2] for the **82544GC/EI**: The Ethernet controller implements the 802.3z PCS function, the Auto-Negotiation function and 10-bit data path interface (TBI) for both receive and transmit operations. It is used for 1000BASE-SX, -LX, and -CX configurations, operating only at 1000 Mb/s full-duplex. The on-chip PCS circuitry is only used when the link interface is configured for TBI mode and it is bypassed in internal PHY modes.

1. Not applicable to the **82541ER**.
2. Not applicable to the **82544GC/EI**, **82540EP/EM**, **82541xx**, and **82547GI/EI**.

- 卸载接收和传输IP及TCP/UDP校验和
- 直接从发送FIFO重新传输导致错误的传输（冲突检测、数据下溢），从而消除从主机内存重新访问此数据的需求

2.3.4 10/100/1000 Mbps接收和传输MAC模块

控制器的CSMA/CD单元处理所有IEEE 802.3接收和传输MAC功能，同时在DMA和TBI/内部SerDes/MII/GMII接口块之间进行接口。CSMA/CD单元支持IEEE 802.3用于10 Mb/s, IEEE 802.3u用于100 Mb/s，以及IEEE 802.3z和IEEE 802.3ab用于1000 Mb/s。

以太网控制器支持半双工 10/100 Mb/s MII 或 1000 Mb/s GMII 模式，并在全双工操作中支持上述所有规范。在半双工模式下，以太网控制器支持 IEEE 802.3z 规范中规定的操作。在接收路径中，以太网控制器支持载波扩展包和数据包突发操作期间生成的数据包。在传输路径中，82554GC/EI 也支持载波扩展包，并且可以配置为以数据包突发模式传输。

以太网控制器提供各种过滤功能，以提供更好的性能和更低的处理器利用率，如下所示：

- 提供最多 16 个地址，用于精确匹配单播/多播地址过滤。
- 基于 4096 位向量提供多播地址过滤。同时支持混杂单播和混杂多播过滤。
- 以太网控制器剥离 IEEE 802.1q VLAN 标签，并根据其 VLAN ID 过滤数据包。最多支持 4096 个 VLAN 标签[1]。

在传输路径中，以太网控制器支持逐包插入VLAN标签信息。

以太网控制器实现了IEEE 802.3x中定义的流控制功能，以及IEEE 802.3z定义的非对称流控制特定操作。以太网控制器还提供外部引脚，通过外部逻辑控制流控制功能。

2.3.5 MII/GMII/TBI/内部SerDes接口块

以太网控制器提供以下串行接口：
- 一个到内部PHY的GMII/MII接口。
- 内部SerDes接口2（82546GB/EB和82545GM/EM）/十位接口（TBI）2用于82544GC/EI：以太网控制器实现了802.3z PCS功能、自动协商功能和10位数据路径接口（TBI），用于接收和传输操作。它用于1000BASE-SX、-LX和-CX配置，仅以1000 Mb/s全双工运行。片上PCS电路仅在链路接口配置为TBI模式时使用，并在内部PHY模式下被绕过。

1. 不适用于 82541ER。 2. 不适用于 82544GC/EI、82540EP/EM、82541xx 和 82547GI/EI。

Note: Refer to the Extended Device Control Register (bits 23:22) for mode selection (see Section 13.4.6).

The link can be configured by several methods. Software can force the link setting to Auto-Negotiation by setting either the MAC in TBI mode (internal SerDes for the **82546GB/EB** and **82545GM/EM**), or the PHY in internal PHY mode.

The speed of the link in internal PHY mode can be determined by several methods:

- Auto speed detection based on the receive clock signal generated by the PHY.
- Detection of the PHY link speed indication.
- Software forcing the configuration of link speed.

2.3.6 10/100/1000 Ethernet Transceiver (PHY)

The Ethernet controller provides a full high-performance, integrated transceiver for 10/100/1000 Mb/s data communication. The physical layer (PHY) blocks are 802.3 compliant and capable of operating in half-duplex or full-duplex modes.

Highlights of the PHY blocks are as follows:

- Data stream serializers and encoders. Encoding techniques include Manchester, 4B/5B and 4D/PAM5. These blocks also perform data scrambling for 100/1000 Mb/s transmission as a technique to minimize radiated Electromagnetic Interference (EMI).
- A multi-mode transmit digital to analog converter, which produces filtered waveforms appropriate for the 10BASE-T, 100BASE-TX or 1000BASE-T Ethernet standards.
- Receiver Analog-to-Digital Converter (ADC). The ADC uses a 125 MHz sampling rate.
- Receiver decoders. These blocks perform the inverse operations of serializers, encoders and scramblers.
- Active hybrid and echo canceller blocks. The active hybrid and echo canceller blocks reduce the echo effect of transmitting and receiving simultaneously on the same analog pairs.
- NEXT canceller. This unit removes high frequency Near End Crosstalk induced among adjacent signal pairs.
- Additional wave shaping and slew rate control circuitry to reduce EMI.

Because the Ethernet controller is IEEE-compliant, the PHY blocks communicate with the MAC blocks through an internal GMII/MII bus operating at clock speeds of 2.5 MHz up to 125 MHz.

The Ethernet controller also uses an IEEE-compliant internal Management Data interface to communicate control and status information to the PHY.

2.3.7 EEPROM Interface

The PCI/PCI-X Family of Gigabit Ethernet Controllers provide a four-wire direct interface to a serial EEPROM device such as the 93C46 or compatible for storing product configuration information. Several words of the data stored in the EEPROM are automatically accessed by the Ethernet controller, after reset, to provide pre-boot configuration data to the Ethernet controller before it is accessible by the host software. The remainder of the stored information is accessed by various software modules to report product configuration, serial number and other parameters.

注意：请参考扩展设备控制寄存器（位23:22）进行模式选择（参见第13.4.6节）。

链路可以通过多种方法进行配置。软件可以通过设置TBI模式下的MAC（82546GB/EB和82545GM/EM的内部SerDes），或内部PHY模式下的PHY，强制链路设置为自动协商。

内部PHY模式下的链路速度可以通过多种方法确定：

- 基于PHY生成的接收时钟信号进行自动速度检测。
- 检测PHY链路速度指示。
- 软件强制配置链路速度。

2.3.6 10/100/1000 以太网收发器 (PHY)

以太网控制器为 10/100/ 1000 Mb/s 数据通信提供全高性能集成收发器。物理层 (PHY) 模块符合 802.3 标准，并能够在半双工或全双工模式下运行。

PHY 模块的特点如下：

- 数据流序列化器和编码器。编码技术包括曼彻斯特编码、4B/5B 编码和 4D/PAM5 编码。这些模块还执行数据扰码，以 100/1000 Mb/s 传输作为减少辐射电磁干扰（EMI）的技术。
- 多模式发射数模转换器，产生适用于 10BASE-T、100BASE-TX 或 1000BASE-T 以太网标准的滤波波形。
- 接收器模数转换器 (ADC)。ADC 使用 125 MHz 采样率。
- 接收器解码器。这些模块执行序列化器、编码器和扰码器的逆操作。
- 主动混合和回波消除器模块。主动混合和回波消除器模块减少在同一对模拟线路上同时发送和接收的回声效应。
- NEXT 消除器。该单元消除相邻信号对之间产生的高频近端串扰。
- 附加波形整形和压摆率控制电路以减少电磁干扰。

因为以太网控制器符合 IEEE 标准，PHY 模块通过内部 GMII/MII 总线与 MAC 模块通信，该总线时钟速度为 2.5 MHz 至 125 MHz。

以太网控制器还使用一个符合IEEE标准的内部管理数据接口来与PHY通信控制和状态信息。

2.3.7 EEPROM接口

PCI/PCI-X系列千兆以太网控制器提供四线直接接口到串行EEPROM设备（如93C46或兼容设备）用于存储产品配置信息。在复位后，以太网控制器会自动访问存储在EEPROM中的若干字节数据，以向以太网控制器提供预启动配置数据，在主机软件可以访问之前。存储信息的其余部分被各种软件模块访问以报告产品配置、序列号和其他参数。

2.3.8 FLASH Memory Interface

The Ethernet controller provides an external parallel interface to a FLASH device. Accesses to the FLASH are controlled by the Ethernet controller and are accessible to software as normal PCI reads or writes to the FLASH memory mapping area. The Ethernet controller supports FLASH devices with up to 512 KB of memory.

Note: The **82540EP/EM** provides an external interface to a serial FLASH or Boot EEPROM device. See Appendix B for more information.

2.4 DMA Addressing

In appropriate systems, all addresses mastered by the Ethernet controller are 64 bits in order to support systems that have larger than 32-bit physical addressing. Providing 64-bit addresses eliminates the need for special segment registers.

Note: The PCI 2.2 or 2.3 Specification requires that any 64-bit address whose upper 32 bits are all 0b appear as a 32-bit address cycle. The Ethernet controller complies with the PCI 2.2 or 2.3 Specification.

PCI is little-endian; however, not all processors in systems using PCI treat memory as little-endian. Network data is fundamentally a byte stream. As a result, it is important that the processor and Ethernet controller agree about the representation of memory data. The default is little-endian mode.

Descriptor accesses are not byte swapped.

The following example illustrates data-byte ordering for little endian. Bytes for a receive packet arrive in the order shown from left to right.

```
01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e
```

Example 2-1. Byte Ordering

There are no alignment restrictions on packet-buffer addresses. The byte address for the major words is shown on the left. The byte numbers and bit numbers for the PCI bus are shown across the top.

Table 2-1. Little Endian Data Ordering

		63							0
		7	6	5	4	3	2	1	0
Byte	0	08	07	06	05	04	03	02	01
Address	8	10	0f	0e	0d	0c	0b	0a	09
	10	18	17	16	15	14	13	12	11
	18	20	1f	1e	1d	1c	1b	1a	19

2.3.8 闪存接口

以太网控制器提供一个外部并行接口到闪存设备。对闪存的访问由以太网控制器控制，并且软件可以像正常PCI读或写到闪存内存映射区域一样访问。以太网控制器支持内存高达512 KB的闪存设备。

注意：82540EP/EM 为串行闪存或启动EEPROM设备提供外部接口。有关更多信息，请参阅附录B。

2.4 DMA寻址

在适当的系统中，以太网控制器管理的所有地址都是64位，以支持具有大于32位物理寻址的系统。提供64位地址消除了对特殊段寄存器的需求。

注意：PCI 2.2或2.3规范要求任何上32位全为0b的64位地址应表现为32位地址周期。以太网控制器符合PCI 2.2或2.3规范。

PCI采用小端模式；然而，在使用PCI的系统中的所有处理器并不都将内存视为小端模式。网络数据本质上是一个字节流。因此，处理器和以太网控制器就内存数据的表示方式达成一致非常重要。默认为小端模式。

描述符访问不会进行字节交换。

以下示例说明了小端模式下的数据字节顺序。接收数据包的字节从左到右按所示顺序到达。

01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e

示例 2-1。字节排序

数据包缓冲地址没有对齐限制。主要字的字节地址显示在左侧。PCI总线的字节编号和位编号显示在顶部。

表 2-1。小端数据排序

		63							0
		7	6	5	4	3	2	1	0
Byte	0	08	07	06	05	04	03	02	01
地址	8	10	0f	0e	0d	0c	0b	0a	09
	10	18	17	16	15	14	13	12	11
	18	20	1f	1e	1d	1c	1b	1a	19

2.5 Ethernet Addressing

Several registers store Ethernet addresses in the Ethernet controller. Two 32-bit registers make up the address: one is called “high”, and the other is called “low”. For example, the Receive Address Register is comprised of Receive Address High (RAH) and Receive Address Low (RAL). The least significant bit of the least significant byte of the address stored in the register (for example, bit 0 of RAL) is the multicast bit. The LS byte is the first byte to appear on the wire. This notation applies to all address registers, including the flow control registers.

Figure 2-5 shows the bit/byte addressing order comparison between what is on the wire and the values in the unique receive address registers.

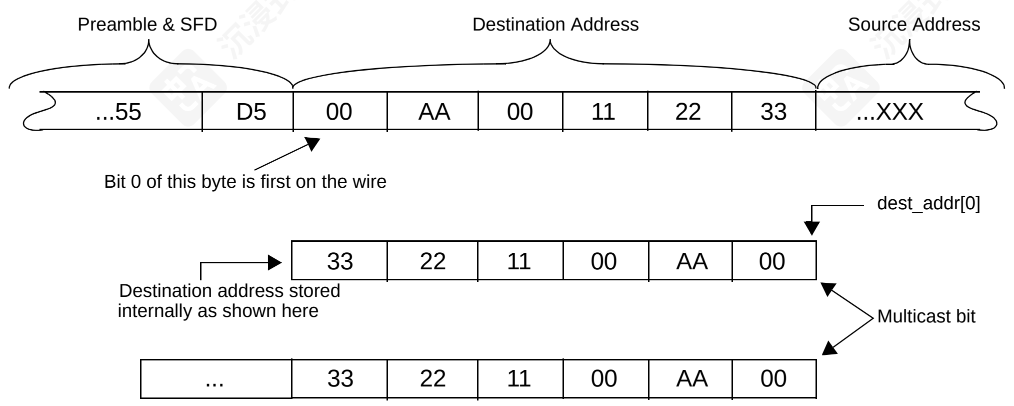


Figure 2-5. Example of Address Byte Ordering

The address byte order numbering shown in Figure 2-5 maps to Table 2-2. Byte #1 is first on the wire.

Table 2-2. Intel® Architecture Byte Ordering

IA Byte #	1 (LSB)	2	3	4	5	6 (MSB)
Byte Value (Hex)	00	AA	00	11	22	33

Note: The notation in this manual follows the convention shown in Table 2-2. For example, the address in Table 2-2 indicates 00_AA_00_11_22_33h, where the first byte (00h_) is the first byte on the wire, with bit 0 of that byte transmitted first.

2.5 以太网寻址

几个寄存器在以太网控制器中存储以太网地址。地址由两个32位寄存器组成：一个是“高”，另一个是“低”。例如，接收地址寄存器由接收地址高（RAH）和接收地址低（RAL）组成。寄存器中存储的地址（例如RAL的位0）的最低字节的最小位是多播位。LS字节是第一个出现在线上的字节。此符号适用于所有地址寄存器，包括流控制寄存器。

图2-5显示了线上传输的内容与唯一接收地址寄存器中的值之间的位/字节寻址顺序比较。

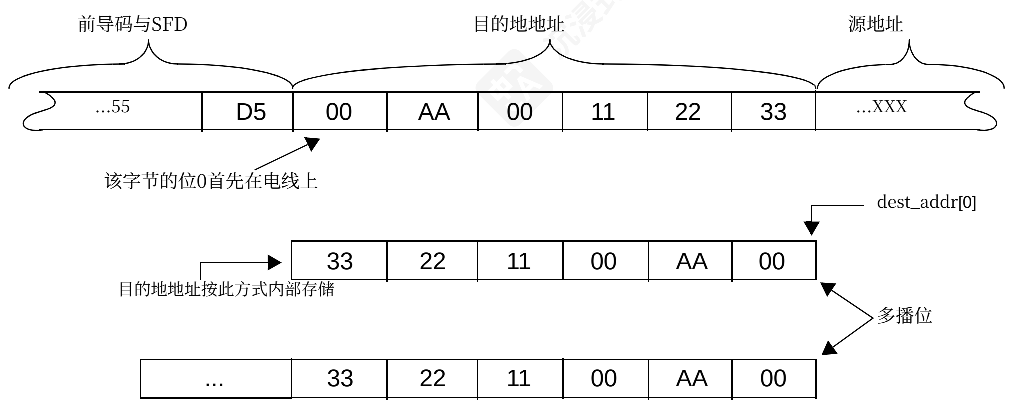


图2-5。地址字节排序示例

图2-5中显示的地址字节顺序编号映射到表2-2。字节#1首先出现在线上。

表2-2。英特尔®架构字节顺序

IA字节 #	1 (LSB)	2	3	4	5	6 (MSB)
字节值 (十六进制)	00	AA	00	11	22	33

注意：本手册中的符号遵循表2-2中所示约定。例如，表2-2中的地址指示00_AA_00_11_22_33h，其中第一个字节（00h_）是线上的第一个字节，该字节的位0首先传输。

2.6 Interrupts

The Ethernet controller provides a complete set of interrupts that allow for efficient software management. The interrupt structure is designed to accomplish the following:

- Make accesses “thread-safe” by using ‘set’ and ‘clear-on-read’ rather than ‘read-modify-write’ operations.
- Minimize the number of interrupts needed relative to work accomplished.
- Minimize the processing overhead associated with each interrupt.

Intel accomplished the first goal by an interrupt logic consisting of four interrupt registers. More detail about these registers is given in sections 13.4.17 through 13.4.21.

- Interrupt Cause ‘Set’ and ‘Read’ Registers

  The Read register records the cause of the interrupt. All bits set at the time of the read are auto-cleared. The cause bit is set for each bit written as a 1b in the Set register. If there is a race between hardware setting a cause and software clearing an interrupt, the bit remains set. No race condition exists on writing the Set register. A ‘set’ provides for software posting of an interrupt. A ‘read’ is auto-cleared to avoid expensive write operations. Most systems have write buffering, which minimizes overhead, but typically requires a read operation to guarantee that the write operation has been flushed from the posted buffers. Without auto-clear, the cost of clearing an interrupt can be as high as two reads and one write.
- Interrupt Mask ‘Set’ (Read) and ‘Clear’ Registers

  Interrupts appear on PCI only if the interrupt cause bit is a 1b, and the corresponding interrupt mask bit is a 1b. Software can block assertion of the interrupt wire by clearing the bit in the mask register. The cause bit stores the interrupt event regardless of the state of the mask bit. The Clear and Set operations make this register more “thread-safe” by avoiding a ‘read-modify-write’ operation on the mask register. The mask bit is set to a 1b for each bit written in the Set register, and cleared for each bit written in the Clear register. Reading the Set register returns the current value.

Intel accomplished the second goal (minimizing interrupts) by three actions:

- Reducing the frequency of all interrupts (see Section 13.4.17). Not applicable to the **82544GC/EI**.
- Accepting multiple receive packets before signaling an interrupt (see Section 3.2.3)
- Eliminating (or at least reducing) the need for interrupts on transmit (see Section 3.2.7)

The third goal is accomplished by having one interrupt register consolidate all interrupt information. This eliminates the need for multiple accesses.

Note that the Ethernet controller also supports Message Signaled Interrupts as defined in the PCI 2.2, 2.3, and PCI-X specifications. See Section 4.1.3.1 for details.

2.6 中断

以太网控制器提供了一套完整的中断，允许进行高效的软件管理。中断结构设计用于完成以下功能：

- 通过使用“设置”和“读清除”而不是“读改写”操作，使访问“线程安全”。
- 相对于完成的工作，尽量减少所需的中断数量。
- 最小化每个中断相关的处理开销。

英特尔通过一个由四个中断寄存器组成的中断逻辑完成了第一个目标。关于这些寄存器的更多详细信息在 13.4.17 节到 13.4.21 节中给出。

- 中断原因‘设置’和‘读’寄存器

  读寄存器记录了中断的原因。在读取时设置的位会自动清除。对于在设置寄存器中写入为 1b 的每个位，都会设置原因位。如果硬件设置原因和软件清除中断之间存在竞争，该位将保持设置。在写入设置寄存器时不存在竞争条件。‘设置’为软件发布中断提供支持。‘读取’会自动清除以避免昂贵的写操作。大多数系统都有写缓冲区，这可以最小化开销，但通常需要执行读取操作来保证写操作已从发布的缓冲区中刷新。如果没有自动清除，清除中断的成本可能高达两次读取和一次写入。
- 中断屏蔽‘设置’（读）和‘清除’寄存器

  中断仅在PCI上出现，如果中断原因位是1b，并且相应的中断屏蔽位也是1b。软件可以通过清除屏蔽寄存器中的位来阻止中断线的断言。原因位会存储中断事件，而不管屏蔽位的状态如何。清除和设置操作通过避免对屏蔽寄存器进行‘读-修改-写’操作，使该寄存器更‘线程安全’。设置寄存器中写入的每个位都会将屏蔽位设置为1b，而清除寄存器中写入的每个位都会清除屏蔽位。读取设置寄存器会返回当前值。

英特尔通过三个操作完成了第二个目标（最小化中断）：

- 减少所有中断的频率（见第13.4.17节）。不适用于82544GC/EI。
- 在接受多个接收数据包后再发出中断信号（见第3.2.3节）
- 消除（或至少减少）传输时对中断的需求（参见第3.2.7节）

第三个目标是通过一个中断寄存器整合所有中断信息来实现的。这消除了对多次访问的需求。

注意：以太网控制器还支持PCI 2.2、2.3和PCI-X规范中定义的消息触发中断。详情请参见第4.1.3.1节。

2.7 Hardware Acceleration Capability

The Ethernet controller provides the ability to offload IP, TCP, and UDP checksum for transmit. The functionality provided by these features can significantly reduce processor utilization by shifting the burden of the functions from the driver to the hardware.

The checksum offloading feature is briefly outlined in the following sections. More detail about all of the hardware acceleration capabilities is provided in Section 3.2.9.

2.7.1 Checksum Offloading

The Ethernet controller provides the ability to offload the IP, TCP, and UDP checksum requirements from the software device driver. For common frame types, the hardware automatically calculates, inserts, and checks the appropriate checksum values normally handled by software.

For transmits, every Ethernet packet might have two checksums calculated and inserted by the Ethernet controller. Typically, these would be the IP checksum, and either the TCP or UDP checksum. The software device driver specifies which portions of the packet are included in the checksum calculations, and where the calculated values are inserted via descriptors (refer to Section 3.3.5 for details).

For receives, the hardware recognizes the packet type and performs the checksum calculations and error checking automatically. Checksum and error information is provided to software through the receive descriptors (refer to Section 3.2.9 for details).

2.7.2 TCP Segmentation

The Ethernet controller implements a TCP segmentation capability for transmits that allows the software device driver to offload packet segmentation and encapsulation to the hardware. The software device driver can send the Ethernet controller the entire IP, TCP or UDP message sent down by the Network Operating System (NOS) for transmission. The Ethernet controller segments the packet into legal Ethernet frames and transmit them on the wire. By handling the segmentation tasks, the hardware alleviates the software from handling some of the framing responsibilities. This reduces the overhead on the CPU for the transmission process thus reducing overall CPU utilization. See Section 3.5 for details.

2.8 Buffer and Descriptor Structure

Software allocates the transmit and receive buffers, and also forms the descriptors that contain pointers to, and the status of, those buffers. A conceptual ownership boundary exists between the driver software and the hardware of the buffers and descriptors. The software gives the hardware ownership of a queue of buffers for receives. These receive buffers store data that the software then owns once a valid packet arrives.

For transmits, the software maintains a queue of buffers. The driver software owns a buffer until it is ready to transmit. The software then commits the buffer to the hardware; the hardware then owns the buffer until the data is loaded or transmitted in the transmit FIFO.

2.7 硬件加速能力

以太网控制器提供传输IP、TCP和UDP校验和卸载的能力。这些功能提供的功能可以通过将功能的负担从驱动程序转移到硬件来显著降低处理器利用率。

校验和卸载功能在以下章节中简要概述。所有硬件加速能力的详细信息在3.2.9节中提供。

2.7.1 校验和卸载

以太网控制器提供从软件设备驱动程序卸载IP、TCP和UDP校验和需求的能力。对于常见帧类型，硬件会自动计算、插入和检查软件通常处理的适当校验和值。

对于传输，每个以太网数据包可能由以太网控制器计算并插入两个校验和。通常，这些将是IP校验和，以及TCP或UDP校验和之一。软件设备驱动程序指定数据包的哪些部分包含在校验和计算中，并通过描述符插入计算出的值（有关详细信息，请参阅第3.3.5节）。

对于接收，硬件识别数据包类型并自动执行校验和计算和错误检查。校验和和错误信息通过接收描述符提供给软件（有关详细信息，请参阅第3.2.9节）。

2.7.2 TCP分段

以太网控制器为传输实现了TCP分段功能，允许软件设备驱动程序将数据包分段和封装卸载到硬件。软件设备驱动程序可以向以太网控制器发送网络操作系统（NOS）向下传输的整个IP、TCP或UDP消息。以太网控制器将数据包分段为合法的以太网帧并在线路上传输。通过处理分段任务，硬件减轻了软件处理部分帧处理责任的压力。这减少了传输过程中CPU的开销，从而降低了整体CPU利用率。有关详细信息，请参阅第3.5节。

2.8 缓冲区和描述符结构

软件分配发送和接收缓冲区，并形成包含指向这些缓冲区的指针及其状态的描述符。驱动软件和缓冲区及描述符的硬件之间存在一个概念上的所有权边界。软件将接收缓冲区的队列的所有权交给硬件。这些接收缓冲区存储数据，一旦收到一个有效的数据包，软件就拥有这些数据。

对于发送，软件维护一个缓冲区队列。驱动软件拥有一个缓冲区，直到它准备好发送。然后软件将该缓冲区提交给硬件；硬件随后拥有该缓冲区，直到数据被加载或传输到发送FIFO中。

Descriptors store the following information about the buffers:

- The physical address
- The length
- Status and command information about the referenced buffer

Descriptors contain an end-of-packet field that indicates the last buffer for a packet. Descriptors also contain packet-specific information indicating the type of packet, and specific operations to perform in the context of transmitting a packet, such as those for VLAN or checksum offload.

Section 3 provides detailed information about descriptor structure and operation in the context of packet transmission and reception.

描述符存储以下关于缓冲区的信息:

- 物理地址
- 长度
- 关于引用缓冲区的状态和命令信息

描述符包含一个数据包结束字段，该字段指示数据包的最后一个缓冲区。描述符还包含数据包特定信息，指示数据包的类型，以及在传输数据包的上下文中执行特定操作，例如VLAN或校验和卸载。

第3节提供了关于描述符结构和操作在数据包传输和接收环境下的详细信息。

Receive and Transmit Description 3

3.1 Introduction

This section describes the packet reception, packet transmission, transmit descriptor ring structure, TCP segmentation, and transmit checksum offloading for the PCI/PCI-X Family of Gigabit Ethernet Controllers.

Note: The **82544GC/EI** does not support IPv6.

3.2 Packet Reception

In the general case, 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 a receive buffer in host memory, and updating the state of a receive descriptor.

3.2.1 Packet Address Filtering

Hardware stores incoming packets in host memory subject to the following filter modes. If there is insufficient space in the receive FIFO, hardware drops them and indicates the missed packet in the appropriate statistics registers.

The following filter modes are supported:

- ***Exact Unicast/Multicast*** — The destination address must exactly match one of 16 stored addresses. These addresses can be unicast or multicast.
- ***Promiscuous Unicast*** — Receive all unicasts.
- ***Multicast*** **—** The upper bits of the incoming packet's destination address index a bit vector that indicates whether to accept the packet; if the bit in the vector is one, accept the packet, otherwise, reject it. The controller provides a 4096 bit vector. Software provides four choices of which bits are used for indexing. These are [47:36], [46:35], [45:34], or [43:32] of the internally stored representation of the destination address.
- ***Promiscuous Multicast*** **—** Receive all multicast packets.
- ***VLAN*** — Receive all VLAN[1] packets that are for this station and have the appropriate bit set in the VLAN filter table. A detailed discussion and explanation of VLAN packet filtering is contained in Section 9.3.

Normally, only good packets are received. These are defined as those packets with no CRC error, symbol error, sequence error, length error, alignment error, or where carrier extension or receive errors are detected. However, if the store–bad–packet bit is set in the Device Control register (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 (RCTL.UPE/MPE) and the store–bad–packet bit (RCTL.SBP).

1. Not applicable to the **82541ER**.

接收和传输描述 3

3.1 简介

本节描述了PCI/PCI-X系列千兆以太网控制器的数据包接收、数据包传输、传输描述符环结构、TCP分段和传输校验和卸载。

注意：82544GC/EI 不支持 IPv6。

3.2 数据包接收

在一般情况下，数据包接收包括识别线路上是否存在数据包、执行地址过滤、将数据包存储在接收数据FIFO中、将数据传输到主机内存中的接收缓冲区，以及更新接收描述符的状态。

3.2.1 数据包地址过滤

硬件根据以下过滤模式将入站数据包存储在主机内存中。如果接收FIFO中没有足够的空间，硬件将丢弃它们，并在相应的统计寄存器中指示丢失的数据包。

支持以下过滤模式：

- 精确单播/多播 — 目的地地址必须与存储的16个地址之一完全匹配地址。这些地址可以是单播或多播。
- 混杂单播 — 接收所有单播数据包。
- 多播 — 入站数据包的目的地地址的上位比特索引一个位向量，该位向量指示是否接受数据包；如果向量中的位为1，则接受数据包，否则拒绝。控制器提供一个4096比特向量。软件提供四种选择，用于索引哪些位被使用。这些是 [47:36], [46:35], [45:34], 或 [43:32] ，它们是内部存储的目的地地址表示中的比特。
- 混杂多播 — 接收所有多播数据包。
- VLAN — 接收所有为该站点的VLAN1数据包，并且VLAN过滤器表中设置了适当的位。VLAN数据包过滤的详细讨论和解释包含在9.3节中。

通常，只有正常的数据包会被接收。这些数据包被定义为没有CRC错误、符号错误、序列错误、长度错误、对齐错误，或者检测到载波扩展或接收错误的数据包。然而，如果设备控制寄存器（RCTL.SBP）中的store–bad–packet位被设置，那么通过过滤器函数的坏数据包会被存储在主机内存中。数据包错误由接收描述符（RDESC.ERRORS）中的错误位指示。通过设置混杂使能（RCTL.UPE/MPE）和store–bad–packet位（RCTL.SBP），可以接收所有数据包，无论它们是否为坏数据包。

1. 不适用于82541ER。

If manageability is enabled and if RCMCP is enabled then ARP request packets can be directed over the SMBus or processed internally by the ASF controller rather than delivered to host memory (not applicable to the **82544GC/EI** or **82541ER**.

3.2.2 Receive Data Storage

Memory buffers pointed to by descriptors store packet data. Hardware supports seven receive buffer sizes:

- 256 B
- 512 B
- 1024 B
- 2048 B
- 4096 B
- 8192 B
- 16384 B

Buffer size is selected by bit settings in the Receive Control register (RCTL.BSIZE & RCTL.BSEX). See Section 13.4.22 for details.

The Ethernet controller places no alignment restrictions on packet 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 may have no knowledge of a specific Ethernet controller's buffer alignment requirements.

Although alignment is completely unrestricted, it is highly recommended that software allocate receive buffers on at least cache-line boundaries whenever possible.

3.2.3 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. Table 3-1 lists where the shaded areas indicate fields that are modified by hardware upon packet reception.

Table 3-1. Receive Descriptor (RDESC) Layout

	63 — 48	47 — 40	39 — 32	31 — 16	15 — 0
0	Buffer Address [63:0]				
8	Special	Errors	Status	Packet Checksum (See Note)	Length

82544GC/EI only

	63 — 48	47 — 40	39 — 32	31 — 16	15 — 0
0	Buffer Address [63:0]				
8	Reserved	Errors	Status	Reserved	Length

Note: The checksum indicated here is the unadjusted "16 bit ones complement" of the packet. A software assist may be required to back out appropriate information prior to sending it to upper software

如果可管理性被启用并且RCMCP被启用，那么ARP请求数据包可以经由SMBus传输，或者由ASF控制器内部处理，而不是发送到主机内存（不适用于82544GC/EI或82541ER。）

3.2.2 接收数据存储

描述符指向的内存缓冲区存储数据包数据。硬件支持七种接收缓冲区大小：

- 256 B
- 512 B
- 1024 B
- 2048 B
- 4096 B
- 8192 B
- 16384 B

缓冲区大小由接收控制寄存器（RCTL.BSIZE & RCTL.BSEX）中的位设置选择。详细信息请参阅第13.4.22节。

以太网控制器对数据包缓冲区地址不施加对齐限制。在接收缓冲区由网络软件堆栈中较高层分配的情况下，这是可取的，因为这些较高层可能不知道特定以太网控制器的缓冲区对齐要求。

尽管对齐完全不受限制，但强烈建议软件在可能的情况下，在至少缓存行边界上分配接收缓冲区。

3.2.3 接收描述符格式

接收描述符是一种数据结构，其中包含接收数据缓冲区地址和硬件用于存储数据包信息的字段。表3-1列出了其中阴影区域表示在数据包接收时由硬件修改的字段。

表3-1。接收描述符（RDESC）布局

	63 — 48	47 — 40	39 — 32	31 — 16	15 — 0
0	缓冲地址 [63:0]				
8	特殊	错误	状态	数据包校验和（参见注意）	长度

82544GC/EI 仅

	63 — 48	47 — 40	39 — 32	31 — 16	15 — 0
0	缓冲地址 [63:0]				
8	保留	错误	状态	保留	长度

注意：此处指示的校验和是数据包的未调整“16位一补码”。一个软件可能需要协助，以便在将其发送到上层软件之前撤回适当的信息

layers. The packet checksum is always reported in the first descriptor (even in the case of multi-descriptor packets).

Upon receipt of a packet for Ethernet controllers, hardware stores the packet data into the indicated buffer and writes the length, Packet Checksum, status, errors, and status fields. Length covers the data written to a receive buffer including CRC bytes (if any). Software must read multiple descriptors to determine the complete length for packets that span multiple receive buffers.

For standard 802.3 packets (non-VLAN) the Packet Checksum is by default computed over the entire packet from the first byte of the DA through the last byte of the CRC, including the Ethernet and IP headers. Software may modify the starting offset for the packet checksum calculation by means of the Receive Control Register. This register is described in Section 13.4.22. To verify the TCP checksum using the Packet Checksum, software must adjust the Packet Checksum value to back out the bytes that are not part of the true TCP Checksum.

3.2.3.1 Receive Descriptor Status Field

Status information indicates whether the descriptor has been used and whether the referenced buffer is the last one for the packet. Refer to Table 3-2 for the layout of the status field. Error status information is shown in Table 3-3.

For multi-descriptor packets, packet status is provided in the final descriptor of the packet (EOP set). If EOP is not set for a descriptor, only the Address, Length, and DD bits are valid.

Table 3-2. Receive Status (RDESC.STATUS) Layout

7	6	5	4	3	2	1	0
PIF	IPCS	TCPCS	RSV	VP	IXSM	EOP	DD

Receive Descriptor Status Bits	Description
PIF (bit 7)	Passed in-exact filter Hardware supplies the PIF field to expedite software processing of packets. Software must examine any packet with PIF set to determine whether to accept the packet. If PIF is clear, then the packet is known to be for this station, so software need not look at the packet contents. Packets passing only the Multicast Vector has PIF set.
IPCS (bit 6)	IP Checksum Calculated on Packet When Ignore Checksum Indication is deasserted (IXSM = 0b), IPCS bit indicates whether the hardware performed the IP checksum on the received packet. 0b = Do not perform IP checksum 1b = Perform IP checksum Pass/Fail information regarding the checksum is indicated in the error bit (IPE) of the descriptor receive errors (RDESC.ERRORS) IPv6 packets do not have the IPCS bit set. Reads as 0b.

层。数据包校验和始终报告在第一个描述符中（即使在多描述符数据包的情况下也是如此）。

当以太网控制器收到数据包时，硬件将数据包数据存储到指定的缓冲区，并写入长度、数据包校验和、状态、错误和状态字段。长度包括写入接收缓冲区的数据，包括CRC字节（如果有）。软件必须读取多个描述符来确定跨越多个接收缓冲区的数据包的完整长度。

对于标准802.3数据包（非VLAN），数据包校验和默认情况下是从DA的第一个字节到CRC的最后一个字节计算整个数据包，包括以太网和IP头部。软件可以通过接收控制寄存器修改数据包校验和计算的起始偏移量。该寄存器在13.4.22节中描述。要使用数据包校验和验证TCP校验和，软件必须调整数据包校验和值以减去不属于真实TCP校验和的字节。

3.2.3.1 接收描述符状态字段

状态信息指示描述符是否已被使用，以及引用的缓冲区是否是数据包的最后一个。参考表3-2了解状态字段的结构。错误状态信息显示在表3-3中。

对于多描述符数据包，数据包状态在数据包的最后一个描述符中提供（EOP设置）。如果一个描述符没有设置EOP，只有地址、长度和DD位是有效的。

表3-2。接收状态（RDESC.STATUS）布局

7	6	5	4	3	2	1	0
PIF	IPCS	TCPCS	RSV	VP	IXSM	EOP	DD

接收描述符状态位	描述
PIF（位7）	通过不精确过滤器 硬件为PIF字段提供信息以加速软件对数据包的处理。软件必须检查任何PIF字段被设置的包，以确定是否接受该包。如果PIF字段为清除状态，则该包已知是为此站点的，因此软件无需查看包内容。仅通过多播向量的包具有PIF字段被设置。
IPCS（位6）	当忽略校验和指示为未置位（IXSM = 0b）时，在包上计算IP校验和，IPCS位指示 硬件是否在接收到的数据包上执行了IP校验和。 0b = 不执行IP校验和 1b = 执行IP校验和 关于校验和的通过/失败信息指示在错误位（IPE）中 在描述符接收错误（RDESC.ERRORS）中 IPv6数据包没有设置IPCS位。读取为0b。

Receive Descriptor Status Bits	Description
TCPCS (bit 5)	TCP Checksum Calculated on Packet When Ignore Checksum Indication is deasserted (IXSM = 0b), TCPCS bit indicates whether the hardware performed the TCP/UDP checksum on the received packet. 0b = Do not perform TCP/UDP checksum; 1b = Perform TCP/UDP checksum Pass/Fail information regarding the checksum is indicated in the error bit (TCPE) of the descriptor receive errors (RDESC.ERRORS). IPv6 packets may have this bit set if the TCP/UDP packet was recognized. Reads as 0b.
RSV (bit 4)	Reserved Reads as 0b.
VP (bit 3)	Packet is 802.1Q (matched VET) Indicates whether the incoming packet's type matches VET (i.e., if the packet is a VLAN (802.1q) type). It is set if the packet type matches VET and CTRL.VME is set. For a further description of 802.1q VLANs, see Chapter 9. Reads as 0b.
IXSM (bit 2)	Ignore Checksum Indication When IXSM = 1b, the checksum indication results (IPCS, TCPCS bits) should be ignored. When IXSM = 0b the IPCS and TCPCS bits indicate whether the hardware performed the IP or TCP/UDP checksum(s) on the received packet. Pass/Fail information regarding the checksum is indicated in the status bits as described below for IPE and TCPE. Reads as 1b.
EOP (bit 1)	End of Packet EOP indicates whether this is the last descriptor for an incoming packet.
DD (bit 0)	Descriptor Done Indicates whether hardware is done with the descriptor. When set along with EOP, the received packet is complete in main memory.

Note: See Table 3-5 for a description of supported packet types for receive checksum offloading. Unsupported packet types either have the IXSM bit set, or they don't have the TCPCS bit set.

3.2.3.2 Receive Descriptor Errors Field

Most error information appears only when the Store Bad Packets bit (RCTL.SBP) is set and a bad packet is received. Refer to Table 3-3 for a definition of the possible errors and their bit positions.

The error bits are valid only when the EOP and DD bits are set in the descriptor status field (RDESC.STATUS)

接收描述符状态位	描述
TCPCS（位5）	TCP校验和计算在包上 当Ignore Checksum Indication未置位（IXSM = 0b）时，TCPCS位指示硬件是否在接收到的数据包上执行了TCP/UDP校验和。 0b = 不执行TCP/UDP校验和；1b = 执行TCP/UDP校验和 Pass/Fail信息关于校验和指示在描述符接收错误（RDESC.ERRORS）的错误位（TCPE）中。IPv6数据包如果TCP/UDP包被识别，则可能设置此位。读取为0b。
RSV (位4)	保留 读作0b。
VP (位3)	数据包是802.1Q (匹配VET) 指示入站数据包的类型是否匹配VET (即，如果数据包是VLAN (802.1q) 类型)。如果数据包类型匹配VET并且CTRL.VME被设置，则会被设置。有关802.1q VLAN的进一步描述，请参阅第9章。读作0b。
IXSM (bit 2)	忽略校验和指示 当 IXSM = 1b 时，校验和指示结果（IPCS、TCPCS 位）应被忽略。 当 IXSM = 0b 时，IPCS 和 TCPCS 位指示硬件是否对接收到的数据包执行了 IP 或 TCP/UDP 校验和。关于校验和的 Pass/Fail 信息以 IPE 和 TCPE 描述的方式在状态位中指示。 读取为 1b。
EOP (bit 1)	数据包结束 EOP 指示这是否是入站数据包的最后一个描述符。
DD (位0)	描述符完成 指示硬件是否已完成该描述符。当与 EOP 同时设置时，接收到的数据包在主内存中已完成。

注意：请参见表3-5，了解支持接收校验和卸载的数据包类型描述。不受支持的数据包类型要么设置了IXSM位，要么没有设置TCPCS位。

3.2.3.2 接收描述符错误字段

大多数错误信息仅在Store Bad Packets位（RCTL.SBP）被设置且接收到一个坏包时才出现。有关可能的错误及其位位置的说明，请参阅表3-3。

错误位仅在描述符状态字段（RDESC.STATUS）中的EOP和DD位被设置时才有效。

Table 3-3. Receive Errors (RDESC.ERRORS) Layout

7	6	5	4	3	2	1	0
RXE	IPE	TCPE	RSV CXE[a]	RSV	SEQ RSV[b]	SE RSV[b]	CE

a. **82544GC/EI** only.
b. **82541xx**, **82547GI/EI**, and **82540EP/EM** only.

Receive Descriptor Error bits	Description
RXE (bit 7)	RX Data Error Indicates that a data error occurred during the packet reception. A data error in TBI[a] mode (**82544GC/EI**)/internal SerDes (**82546GB/EB** and **82545GM/EM**) refers to the reception of a /V/ code (see Section 8.2.1.3). In GMII or MII mode, the assertion of I_RX_ER during data reception indicates a data error. This bit is valid only when the EOP and DD bits are set; it is not set in descriptors unless RCTL.SBP (Store Bad Packets) control bit is set.
IPE (bit 6)	IP Checksum Error When set, indicates that IP checksum error is detected in the received packet. Valid only when the IP checksum is performed on the receive packet as indicated via the IPCS bit in the RDESC.STATUS field. If receive IP checksum offloading is disabled (RXCSUM.IPOFL), the IPE bit is set to 0b. It has no effect on the packet filtering mechanism. Reads as 0b.
TCPE (bit 5)	TCP/UDP Checksum Error When set, indicates that TCP/UDP checksum error is detected in the received packet. Valid only when the TCP/UDP checksum is performed on the receive packet as indicated via TCPCS bit in RDESC.STATUS field. If receive TCP/UDP checksum offloading is disabled (RXCSUM.TUOFL), the TCPE bit is set to 0b. It has no effect on the packet filtering mechanism. Reads as 0b.
CXE RSV (bit 4)	Carrier Extension Error When set, indicates a packet was received in which the carrier extension error was signaled across the GMII interface. A carrier extension error is signaled by the PHY by the encoding of 1Fh on the receive data inputs while I_RX_ER is asserted. Valid only while working in 1000 Mb/s half-duplex mode of operation. This bit is reserved for all Ethernet controllers except the **82544GC/EI**.
RSV (Bit 3)	Reserved Reads as 0b.

表3-3。接收错误（RDESC.ERRORS）布局

7	6	5	4	3	2	1	0
RXE	IPE	TCPE	RSV CXE[a]	RSV	SEQ RSV[b]	SE RSV[b]	CE

a. 仅限82544GC/EI。
b. 仅限82541xx、82547GI/EI和82540EP/EM。

接收描述符错误位	描述
RXE (位7)	接收数据错误 指示在数据包接收过程中发生了数据错误。在TBI[a]模式（82544GC/EI）/内部SerDes（82546GB/EB和82545GM/EM）中的数据错误是指接收到了/V/码（参见第8.2.1.3节）。在GMII或MII模式下，数据接收期间I_RX_ER的置位表示数据错误。该位仅在EOP和DD位被设置时有效；除非RCTL.SBP（存储错误数据包）控制位被设置，否则在描述符中不会被设置。
IPE (位6)	IP校验和错误 当设置时，指示在接收到的数据包中检测到IP校验和错误。仅在接收数据包执行IP校验和时有效，如RDESC.STATUS字段中的IPCS位所示。 如果禁用接收IP校验和卸载（RXCSUM.IPOFL），则IPE位设置为0b。它对数据包过滤机制没有影响。 读取为0b。
TCPE（位5）	TCP/UDP校验和错误 当设置时，指示在接收到的数据包中检测到TCP/UDP校验和错误。 仅在接收数据包上执行TCP/UDP校验和时才有效，如RDESC.STATUS字段中的TCPCS位所示。 如果禁用接收TCP/UDP校验和卸载（RXCSUM.TUOFL），则TCPE位设置为0b。 这对数据包过滤机制没有影响。 读取为0b。
CXE RSV（位4）	载波扩展错误 当设置时，指示一个数据包在GMII接口上传输了载波扩展错误。载波扩展错误由PHY通过在接收数据输入上编码1Fh并在I_RX_ER被置位时信号表示。 仅在1000 Mbps半双工模式下有效。 此位保留用于除82544GC/EI以外的所有以太网控制器。
RSV（位3）	保留 读取为0b。

Receive Descriptor Error bits	Description
SEQ (bit 2)	Sequence Error When set, indicates a received packet with a bad delimiter sequence (in TBI mode/ internal SerDes). In other 802.3 implementations, this would be classified as a framing error. A valid delimiter sequence consists of: idle →start-of-frame (SOF) → data, →pad (optional) → end-of-frame (EOF) → fill (optional) → idle.
SE (bit 1)	Symbol Error When set, indicates a packet received with bad symbol. Applicable only in TBI mode/ internal SerDes.
CE (bit 0)	CRC Error or Alignment Error CRC errors and alignment errors are both indicated via the CE bit. Software may distinguish between these errors by monitoring the respective statistics registers.

a. Not applicable to the **82540EP/EM**, **82541xx**, or **82547GI/EI**.

3.2.3.3 Receive Descriptor Special Field

Hardware stores additional information in the receive descriptor for 802.1q packets. If the packet type is 802.1q, determined when a packet type field matches the VLAN[1] Ethernet Register (VET) and RCTL.VME = 1b, then the special field records the VLAN information and the four byte VLAN information is stripped from the packet data storage. The Ethernet controller stores the Tag Control Information (TCI) of the 802.1q tag in the Special field. Otherwise, the special field contains 0000h.

Table 3-4. Special Descriptor Field Layout

802.1q Packets

15 13	12	11 0
PRI	CFI	VLAN

All Other Packets

15 8	7 0
00	00

Receive Descriptor Special Field	Description
VLAN	VLAN Identifier 12 bits that records the packet VLAN ID number
CFI	Canonical Form Indicator 1 bit that records the packet's CFI VLAN field
PRI	User Priority 3 bits that records the packet's user priority field.

1. Not applicable to the **82541ER**.

接收描述符错误位	描述
SEQ（位2）	序列错误 当被设置时，指示一个具有无效分隔符序列的接收数据包（在TBI模式/内部SerDes中）。在其他802.3实现中，这将被归类为帧错误。 有效的分隔符序列包括： 空闲 →帧开始（SOF）→ 数据, →填充（可选）→ 帧结束（EOF）→ 填充（可选）→ 空闲。
SE (bit 1)	符号错误 当设置时，指示接收到的数据包存在符号错误。仅适用于TBI模式/内部SerDes。
CE (bit 0)	CRC错误或对齐错误 CRC错误和对齐错误都通过CE位指示。软件可以通过监控相应的统计寄存器来区分这些错误。

a. 不适用于 82540EP/EM、82541xx 或 82547GI/EI。

3.2.3.3 接收描述符特殊字段

硬件在接收描述符中为802.1q数据包存储附加信息。如果数据包类型是802.1q，当数据包类型字段与VLAN[1] 以太网寄存器（VET）和RCTL.VME = 1b匹配时确定，则特殊字段记录VLAN信息，并且四个字节的VLAN信息从数据包数据存储中剥离。以太网控制器将802.1q标签的标签控制信息（TCI）存储在特殊字段中。否则，特殊字段包含0000h。

表3-4. 特殊描述符字段布局

802.1q 数据包

15 13	12	11 0
PRI	CFI	VLAN

其他所有数据包

15 87 0	
00	00

接收描述符特殊字段	描述
VLAN	VLAN标识符 12位，记录数据包的VLAN ID编号
CFI	规范形式指示符 1位，记录数据包的CFI VLAN字段
PRI	用户优先级 3 位，记录数据包的用户优先级字段。

1. 不适用于82541ER。

3.2.4 Receive Descriptor Fetching

The descriptor fetching strategy is designed to support large bursts across the PCI bus. This is made possible by using 64 on-chip receive descriptors and an optimized fetching algorithm. The fetching algorithm attempts to make the best use of PCI bandwidth by fetching a cache line (or more) descriptors with each burst. The following paragraphs briefly describe the descriptor fetch algorithm and the software control provided.

When the on-chip buffer is empty, a fetch happens as soon as any descriptors are made available (software writes to the tail pointer). When the on-chip buffer is nearly empty (RXDCTL.PTHRESH), a prefetch is performed whenever enough valid descriptors (RXDCTL.HTHRESH) are available in host memory and no other PCI 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 chip may elect to perform a fetch which is not a multiple of cache line size. The hardware performs this non-aligned fetch if doing so results in the next descriptor fetch being aligned on a cache line boundary. This mechanism provides the highest efficiency in cases where fetches fall behind software.

Note: The Ethernet controller **never** fetches descriptors beyond the descriptor TAIL pointer.

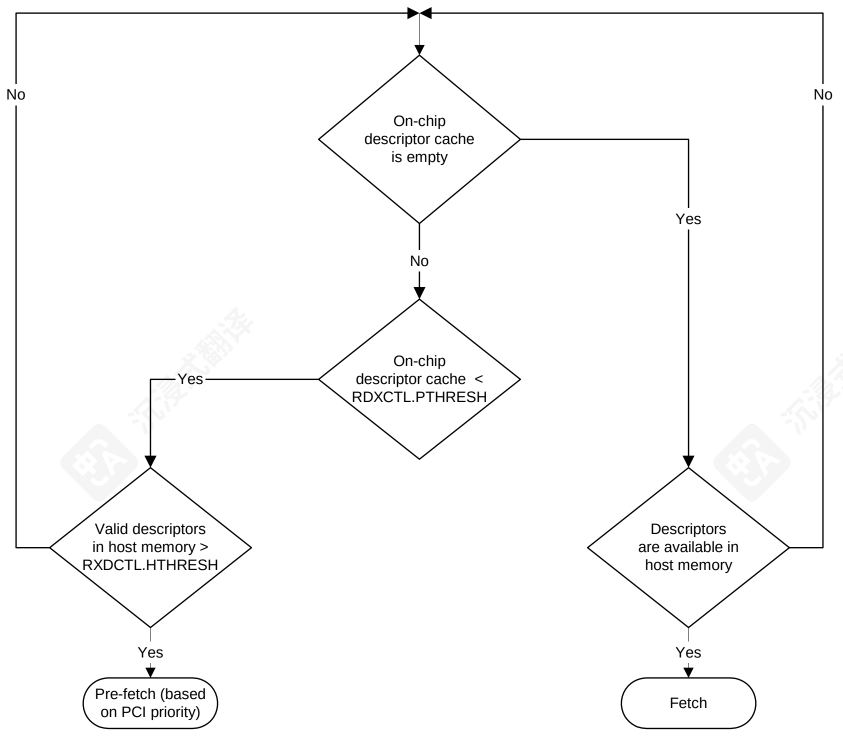

No
No
On-chip descriptor cache is empty
Yes
No
On-chip descriptor cache < RDXCTL.PTHRESH
Yes
Valid descriptors in host memory > RXDCTL.HTHRESH
Descriptors are available in host memory
Yes
Yes
Pre-fetch (based on PCI priority)
Fetch

Figure 3-1. Receive Descriptor Fetching Algorithm

3.2.4 接收描述符获取

描述符获取策略旨在支持 PCI 总线上的大突发传输。这是通过使用 64 个片上接收描述符和优化的获取算法实现的。获取算法尝试通过每次突发获取缓存行（或更多）描述符来充分利用 PCI 带宽。以下段落简要描述了描述符获取算法和提供的软件控制。

当片上缓冲区为空时，只要任何描述符可用（软件写入尾指针），就会立即发生获取。当片上缓冲区几乎为空（RXDCTL.PTHRESH）时，只要主机内存中有足够的有效描述符（ RXDCTL.HTHRESH），并且没有更高优先级的其他PCI活动挂起（描述符获取和写回或数据包数据传输），就会执行预取。

当主机内存中的描述符数量大于可用的片上描述符存储时，芯片可以选择执行不是缓存行大小倍数的获取。如果这样做会导致下一个描述符获取在缓存行边界上对齐，硬件会执行这种非对齐获取。这种机制在获取落后于软件的情况下提供了最高效率。

注意：以太网控制器永远不会获取描述符 TAIL 指针之外的描述符。

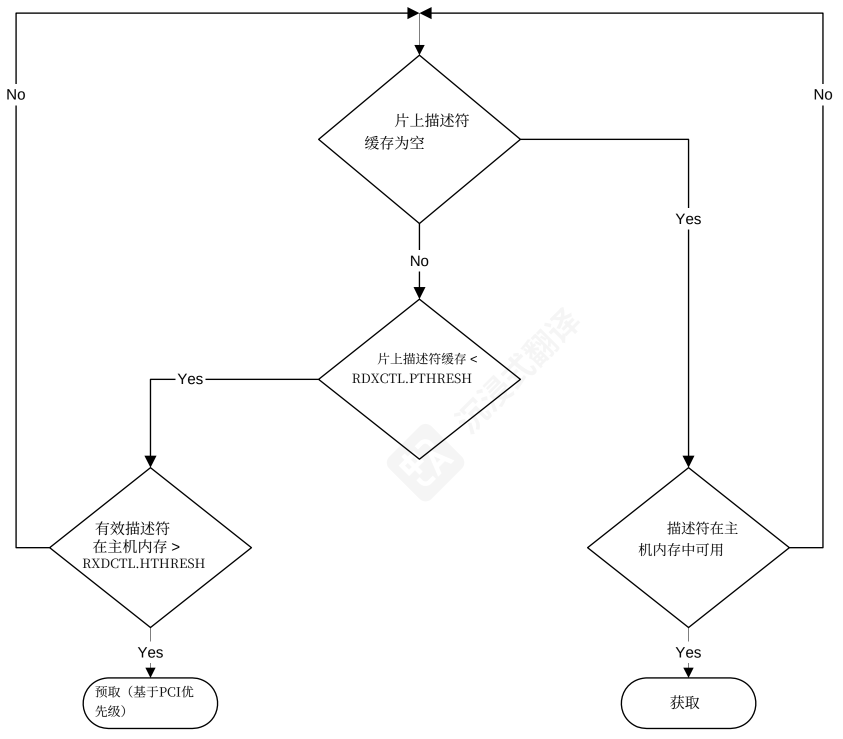


图3-1。接收描述符获取算法

3.2.5 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. Two mechanisms minimize the occurrence of partial line write backs:

- Receive descriptor packing
- Null descriptor padding

The following sections explain these mechanisms.

3.2.5.1 Receive Descriptor Packing

To maximize memory efficiency, receive descriptors are “packed” together and written as a cache line whenever possible. Descriptors accumulate and are written out in one of three conditions:

- RXDCTL.WTHRESH descriptors have been used (the specified max threshold of unwritten used descriptors has been reached)
- The receive timer expires (RADV or RDTR)
- Explicit software flush (RDTR.FPD)

For the first condition, if the number of descriptors specified by RXDCTL.WTHRESH are used, they are written back, regardless of cacheline alignment. It is therefore recommended that WTHRESH be a multiple of cacheline sizes.

In the second condition, a timer (RDTR or RADV) expiration causes all used descriptors to be written back prior to initiating an interrupt.

In the second condition for the **82544GC/EI**, a timer (RDTR) is included 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 (receiver timer interrupt). In general, the arrival rate is sufficiently fast enough that packing is the common case under load.

For the final condition, software may explicitly flush accumulated descriptors by writing the timer register with the high order bit set.

3.2.5.2 Null Descriptor Padding

Hardware stores no data in descriptors with a null data address. Software can make use of this property to cause the first condition under receive descriptor packing to occur early. Hardware writes back null descriptors with the DD bit set in the status byte and all other bits unchanged.

3.2.6 Receive Descriptor Queue Structure

Figure 3-2 shows the structure of the receive descriptor ring. Hardware maintains a circular ring of descriptors and writes back used descriptors just prior to advancing the head pointer. Head and tail pointers wrap back to base when “size” descriptors have been processed.

Software adds receive descriptors by writing the tail pointer with the index of the entry beyond the last valid descriptor. As packets arrive, they are stored in memory and the head pointer is incremented by hardware. When the head pointer is equal to the tail pointer, the ring is empty. Hardware stops storing packets in system memory until software advances the tail pointer, making more receive buffers available.

3.2.5 接收描述符回写

处理器具有大于接收描述符大小（16 字节）的缓存行大小。因此，为每个接收到的数据包回写描述符信息会导致昂贵的部分缓存行更新。两种机制最小化了部分行写回的发生：

- 接收描述符打包
- 空描述符填充

以下各节将解释这些机制。

3.2.5.1 接收描述符打包

为最大化内存效率，接收描述符在可能的情况下会“打包”在一起并以缓存行形式写入。描述符会累积，并在以下三种情况下被写出：

- RXDCTL.WTHRESH 描述符已被使用（指定的未写入使用描述符的最大阈值已达到）
- 接收计时器到期（RADV 或 RDTR）
- 显式软件刷新 (RDTR.FPD)

对于第一个条件，如果RXDCTL.WTHRESH指定的描述符数量被使用，它们会被写回，无论缓存行是否对齐。因此建议WTHRESH是缓存行大小的倍数。

在第二个条件中，计时器（RDTR或RADV）超时会导致在触发中断之前将所有使用的描述符写回。

对于82544GC/EI的第二个条件，计时器（RDTR）被包含以强制描述符的及时写回。计时器初始化后的第一个数据包启动计时器。计时器超时会刷新任何累积的描述符并设置一个中断事件（接收器定时器中断）。通常，到达速率足够快，负载下打包是常见情况。

对于最终条件，软件可以通过将计时器寄存器的高位设置来显式刷新累积的描述符。

3.2.5.2 空描述符填充

硬件在具有空数据地址的描述符中不存储数据。软件可以利用此特性，使接收描述符打包下的第一个条件提前发生。硬件在状态字节中设置DD位并保持其他位不变的情况下，回写空描述符。

3.2.6 接收描述符队列结构

图3-2显示了接收描述符环的结构。硬件维护一个循环的描述符环，并在移动头指针之前回写已使用的描述符。当“大小”描述符被处理后，头指针和尾指针会回绕到基础。

软件通过用最后一个有效描述符之后的条目索引写入尾指针来添加接收描述符。当数据包到达时，它们被存储在内存中，并且头指针由硬件递增。当头指针等于尾指针时，环是空的。硬件停止在系统内存中存储数据包，直到软件移动尾指针，从而提供更多的接收缓冲区。

The receive descriptor head and tail pointers reference 16-byte blocks of memory. Shaded boxes in the figure represent descriptors that have stored incoming packets but have not yet been recognized by software. Software can determine if a receive buffer is valid by reading descriptors in memory rather than by I/O reads. Any descriptor with a non-zero status byte has been processed by the hardware, and is ready to be handled by the software.

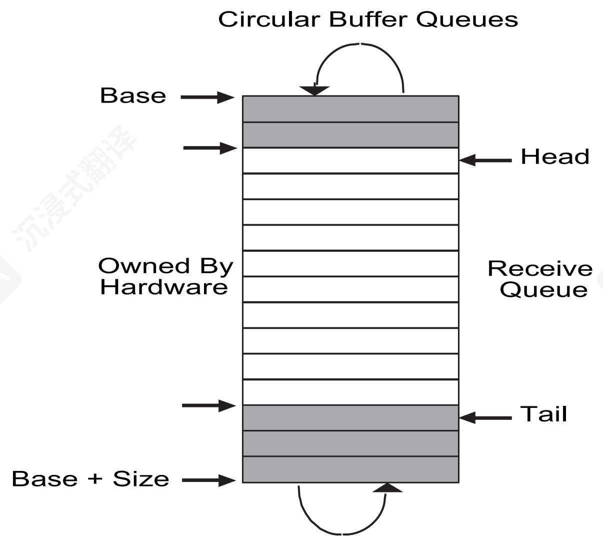


Figure 3-2. Receive Descriptor Ring Structure

Note: The head pointer points to the next descriptor that is written back. At the completion of the descriptor write-back operation, this pointer is incremented by the number of descriptors written back. **HARDWARE OWNS ALL DESCRIPTORS BETWEEN [HEAD AND TAIL]**. Any descriptor not in this range is owned by software.

The receive descriptor ring is described by the following registers:

- Receive Descriptor Base Address registers (RDBAL and RDBAH)

  These registers indicate 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. RDBAL contains the lower 32-bits; RDBAH contains the upper 32 bits. Hardware ignores the lower 4 bits in RDBAL.

- Receive Descriptor Length register (RDLEN)

  This register determines the number of bytes allocated to the circular buffer. This value must be a multiple of 128 (the maximum cache line size). Since each descriptor is 16 bytes in length, the total number of receive descriptors is always a multiple of 8.

- Receive Descriptor Head register (RDH)

  This register holds a value that is an offset from the base, and indicates the in–progress descriptor. There can be up to 64K descriptors in the circular buffer. Hardware maintains a shadow copy that includes those descriptors completed but not yet stored in memory.

接收描述符头和尾指针引用内存中的16字节块。图中阴影框表示已存储传入数据包但尚未被软件识别的描述符。软件可以通过读取内存中的描述符来确定接收缓冲区是否有效，而不是通过I/O读取。任何状态字节非零的描述符都已被硬件处理，并准备好由软件处理。

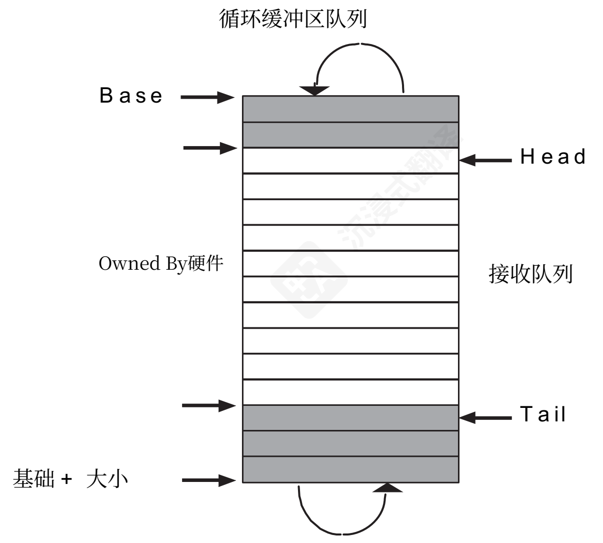


图3-2。接收描述符环结构

注意：头指针指向下一个写回的描述符。在描述符写回操作完成后，此指针会根据写回的描述符数量增加。硬件拥有[HEAD和TAIL]之间的所有描述符。不在该范围内的任何描述符都属于软件。

接收描述符环由以下寄存器描述：

- 接收描述符基地址寄存器（RDBAL和RDBAH）

  这些寄存器指示描述符环缓冲区的起始位置。这个64位地址对齐在16字节边界上，并存储在两个连续的32位寄存器中。RDBAL包含低32位；RDBAH包含高32位。硬件忽略RDBAL中的低4位。

- 接收描述符长度寄存器 (RDLEN)

  该寄存器确定分配给循环缓冲区的字节数。此值必须是128的倍数（最大缓存行大小）。由于每个描述符长度为16字节，接收描述符的总数始终是8的倍数。

- 接收描述符头寄存器 (RDH)

  该寄存器保存一个相对于基础偏移量的值，并指示正在处理的描述符。循环缓冲区中最多可以有64K个描述符。硬件维护一个影子副本，其中包含那些已完成但尚未存储在内存中的描述符。

- Receive Descriptor Tail register (RDT)
  This register holds a value that is an offset from the base, and identifies the location beyond the last descriptor hardware can process. Note that tail should still point to an area in the descriptor ring (somewhere between RDBA and RDBA + RDLEN). This is because tail points to the location where software writes the first new descriptor.

If software statically allocates buffers, and uses memory read to check for completed descriptors, it simply has to zero the status byte in the descriptor to make it ready for reuse by hardware. This is not a hardware requirement (moving the hardware tail pointer is), but is necessary for performing an in–memory scan.

3.2.7 Receive Interrupts

The Ethernet controller can generate four receive-related interrupts:

- Receiver Timer Interrupt (ICR.RXT0)
- Small Receive Packet Detect (ICR.SRPD)
- Receive Descriptor Minimum Threshold (ICR.RXDMT0)
- Receiver FIFO Overrun (ICR.RX0)

3.2.7.1 Receive Timer Interrupt

The Receive Timer Interrupt is used to signal most packet reception events (the Small Receive Packet Detect interrupt is also used in some cases as described later in this section). In order to minimize the interrupts per work accomplished, the Ethernet controller provides two timers to control how often interrupts are generated.

3.2.7.1.1 Receive Interrupt Delay Timer / Packet Timer (RDTR)

The Packet Timer minimizes the number of interrupts generated when many packets are received in a short period of time. The packet timer is started once a packet is received and transferred to host memory (specifically, after the last packet data byte is written to memory) and is reinitialized (to the value defined in RDTR) and started EACH TIME a new packet is received and transferred to the host memory. When the Packet Timer expires (e.g. no new packets have been received and transferred to host memory for the amount of time defined in RDTR) the Receive Timer Interrupt is generated.

Setting the Packet Timer to 0b disables both the Packet Timer and the Absolute Timer (described below) and causes the Receive Timer Interrupt to be generated whenever a new packet has been stored in memory.

Writing to RDTR with its high order bit (FPD) set forces an explicit writeback of consumed descriptors (potentially a partial cache lines amount of descriptors), causes an immediate expiration of the Packet Timer and generates a Receive Timer Interrupt.

The Packet Timer is reinitialized (but not started) when the Receive Timer Interrupt is generated due to an Absolute timer expiration or Small Receive Packet Detect Interrupt.

See section Section 13.4.30 for more details on the Packet Timer.

- 接收描述符尾寄存器 (RDT)
  该寄存器保存一个相对于基础的值，并标识硬件可以处理的最后一个描述符之后的位置。注意，尾部（tail）仍然应指向描述符环（descriptor ring）中的一个区域（RDBA和RDBA + RDLEN之间）。这是因为尾部指向软件写入第一个新描述符的位置。

如果软件静态分配缓冲区，并使用内存读取来检查完成的描述符，它只需将描述符中的状态字节清零，以便硬件可以重新使用。这不是硬件要求（移动硬件尾指针是），但对于执行内存扫描是必要的。

3.2.7 接收中断

以太网控制器可以生成四个与接收相关的中断：

- 接收器定时器中断 (ICR.RXT0)
- 小型接收数据包检测 (ICR.SRPD)
- 接收描述符最小阈值 (ICR.RXDMT0)
- 接收器FIFO溢出 (ICR.RX0)

3.2.7.1 接收定时器中断

接收定时器中断用于信号大多数数据包接收事件（在某些情况下，小型接收数据包检测中断也用于此，如本节后面所述）。为了尽量减少每项工作产生的中断次数，以太网控制器提供了两个定时器来控制中断生成的频率。

3.2.7.1.1 接收中断延迟定时器 / 数据包定时器 (RDTR)

数据包定时器在短时间内接收到大量数据包时，可最小化产生的中断数量。数据包定时器在数据包被接收并传输到主机内存（具体来说，在最后一个数据包数据字节写入内存之后）时启动，并在每次新数据包被接收并传输到主机内存时重新初始化（设置为RDTR中定义的值）并启动。当数据包定时器超时（例如，在RDTR中定义的时间内没有新数据包被接收并传输到主机内存）时，将生成接收定时器中断。

将数据包定时器设置为0b会禁用数据包定时器和绝对定时器（如下所述），并导致每当有新数据包存储在内存中时生成接收定时器中断。

向RDTR写入其高位（FPD）被设置时，会强制回写已消耗的描述符（可能是一部分缓存行数量的描述符），导致数据包定时器立即超时并生成接收定时器中断。

数据包计时器在绝对定时器超时或小型接收包检测中断导致接收定时器中断生成时被重新初始化（但未启动）。

有关数据包计时器的更多详细信息，请参阅第13.4.30节。

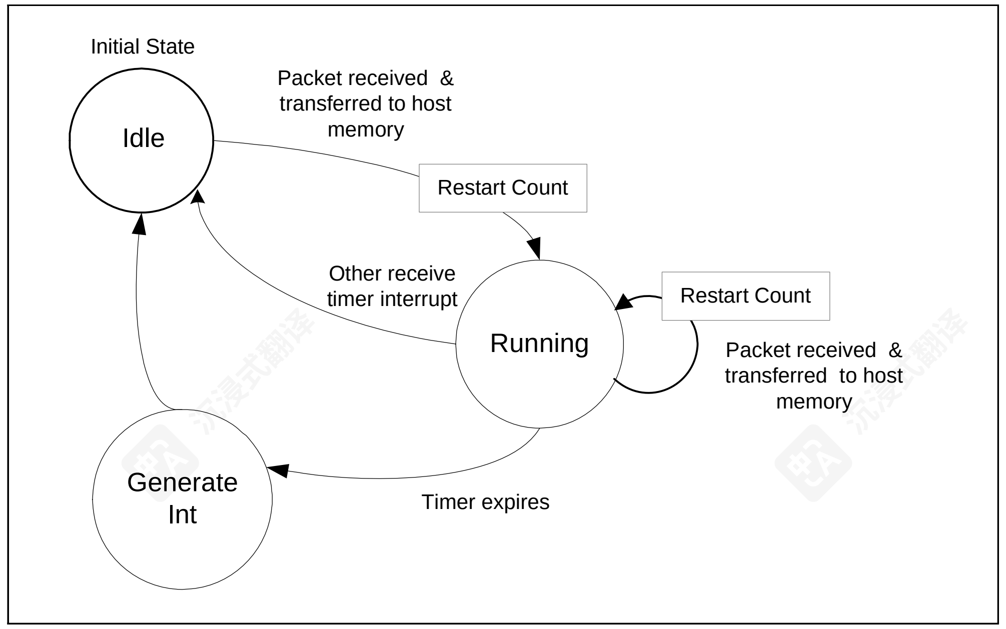

Initial State
Idle
Packet received & transferred to host memory
Restart Count
Other receive timer interrupt
Running
Restart Count
Packet received & transferred to host memory
Generate Int
Timer expires

Figure 3-3. Packet Delay Timer Operation (State Diagram)

3.2.7.1.2 Receive Interrupt Absolute Delay Timer (RADV)

The Absolute Timer ensures that a receive interrupt is generated at some predefined interval after the first packet is received. The absolute timer is started once a packet is received and transferred to host memory (specifically, after the last packet data byte is written to memory) but is NOT reinitialized / restarted each time a new packet is received. When the Absolute Timer expires (no receive interrupt has been generated for the amount of time defined in RADV) the Receive Timer Interrupt is generated.

Setting RADV to 0b or RDTR to 0b disables the Absolute Timer. To disable the Packet Timer only, RDTR should be set to RADV + 1b.

The Absolute Timer is reinitialized (but not started) when the Receive Timer Interrupt is generated due to a Packet Timer expiration or Small Receive Packet Detect Interrupt.

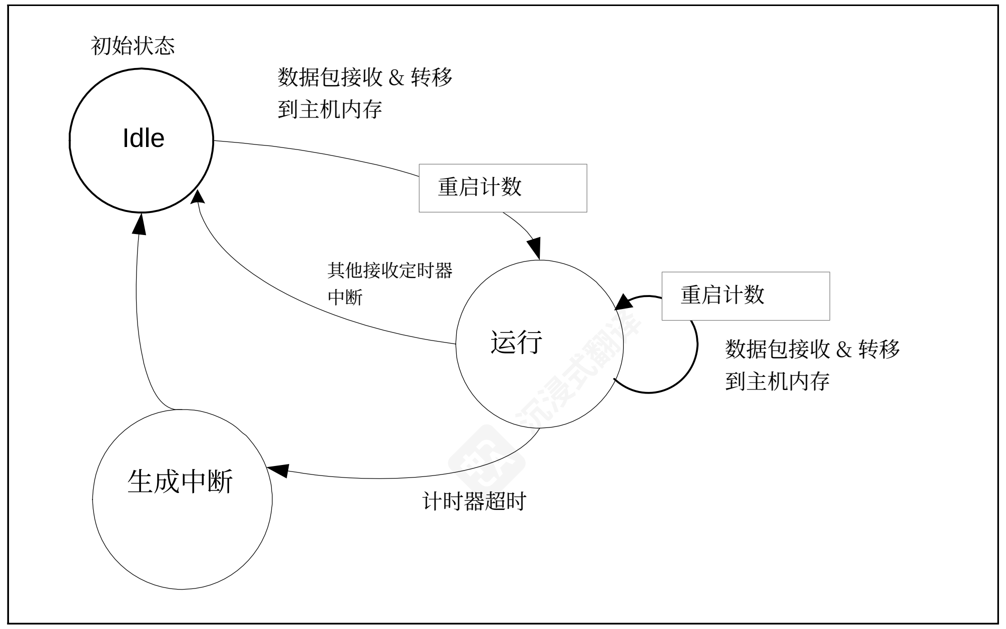


图3-3。数据包延迟计时器操作（状态图）

3.2.7.1.2 接收中断绝对延迟定时器（RADV）

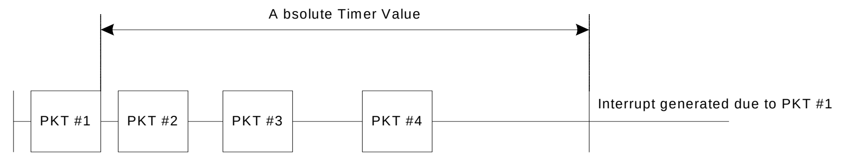
绝对定时器确保在第一个数据包接收后，在预定义间隔内生成接收中断。绝对定时器在数据包接收并传输到主机内存（具体是在最后一个数据包数据字节写入内存之后）时启动，但在每次接收到新数据包时不会重新初始化/重启。当绝对定时器到期时（在RADV定义的时间量内未生成接收中断），将生成接收定时器中断。

将RADV设置为0b或RDTR设置为0b会禁用绝对定时器。要仅禁用数据包计时器，应将RDTR设置为RADV + 1 b。

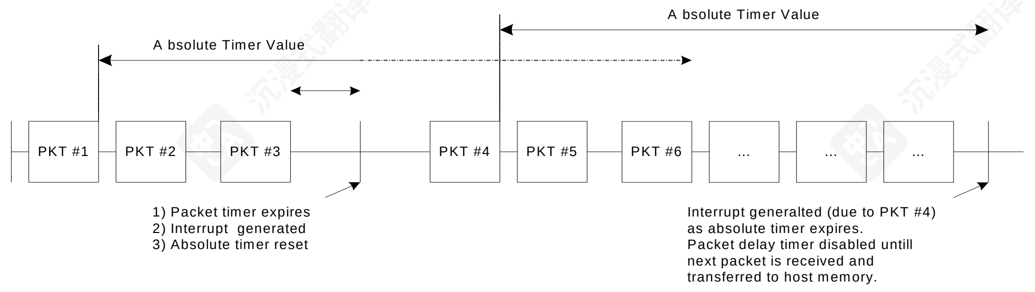
绝对定时器在接收定时器中断由于数据包计时器过期或小型接收数据包检测中断生成时被重新初始化（但未启动）。

The diagrams below show how the Packet Timer and Absolute Timer can be used together:

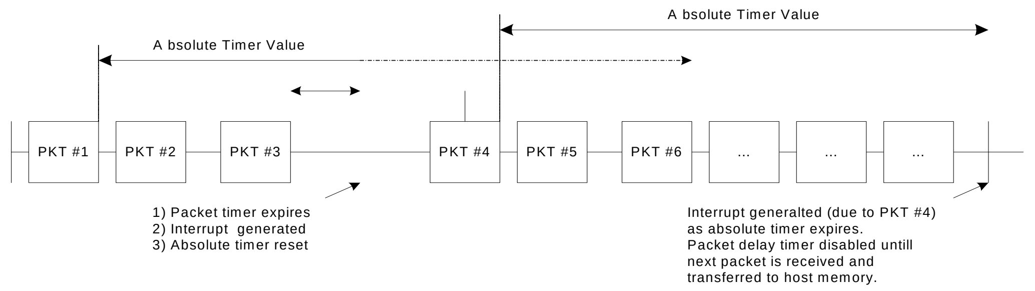
Case A: Using only an absolute timer



Case B: Using an absolute time in conjunction with the Packet timer



Case C: Packet timer expiring while a packet is transferred to host memory.
Illustrates that packet timer is re-started only after a packet is transferred to host memory.



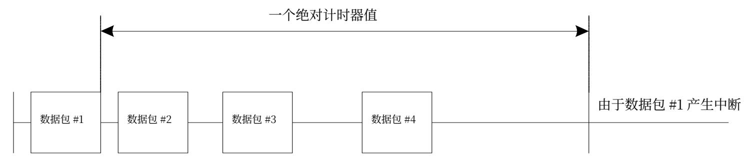
3.2.7.2 Small Receive Packet Detect

A Small Receive Packet Detect interrupt (ICR.SRPD) is asserted when small-packet detection is enabled (RSRPD is set with a non-zero value) and a packet of (size ≤ RSRPD.SIZE) has been transferred into the host memory. When comparing the size the headers and CRC are included (if CRC stripping is not enabled). CRC and VLAN headers are not included if they have been stripped. A receive timer interrupt cause (ICR.RXT0) is also noted when the Small Packet Detect interrupt occurs.

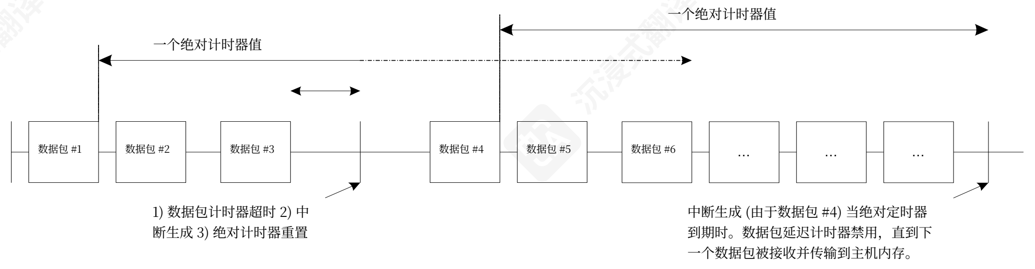
For the **82541xx** and **82547GI/EI**, receiving a small packet does not clear the absolute or packet delay timers, so one packet might generate two interrupts, one due to small packet reception and one due to timer expiration.

下图显示了数据包计时器和绝对定时器如何一起使用：

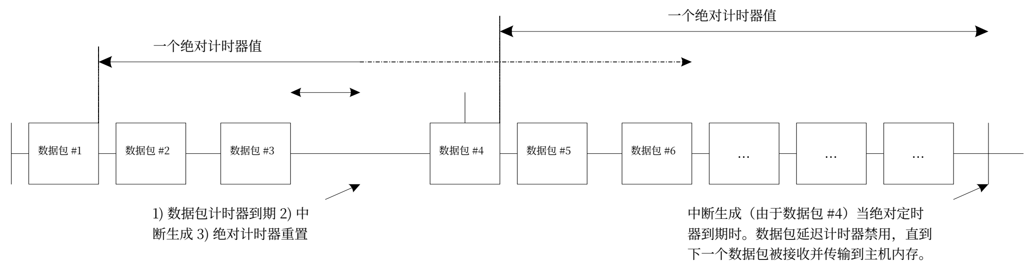
情况 A：仅使用绝对定时器



情况 B：使用绝对时间与数据包计时器结合


一个绝对计时器值
一个绝对计时器值
数据包 #1
数据包 #2
数据包 #3
数据包 #4
数据包 #5
数据包 #6
...
...
...
1) 数据包计时器超时 2) 中断生成 3) 绝对计时器重置
中断生成 (由于数据包 #4) 当绝对定时器到期时。数据包延迟计时器禁用，直到下一个数据包被接收并传输到主机内存。

情况 C：数据包正在传输到主机内存时数据包计时器过期。说明数据包计时器仅在数据包传输到主机内存后才重新启动。


一个绝对计时器值
一个绝对计时器值
数据包 #1
数据包 #2
数据包 #3
数据包 #4
数据包 #5
数据包 #6
...
...
...
1) 数据包计时器到期 2) 中断生成 3) 绝对计时器重置
中断生成（由于数据包 #4）当绝对定时器到期时。数据包延迟计时器禁用，直到下一个数据包被接收并传输到主机内存。

3.2.7.2 小型接收数据包检测

当启用小型数据包检测时（RSRPD使用非零值设置），并且一个大小为（≤ RSRPD.SIZE）的数据包已传输到主机内存时，会触发小型接收数据包检测中断（ICR.SRPD）。在比较大小时，会包含头部和CRC（如果未启用CRC剥离）。如果已剥离，则不包含CRC和VLAN头部。当小型数据包检测中断发生时，也会注意到接收定时器中断触发（ICR.RXT0）。

对于 82541xx 和 82547GI/EI，接收一个小数据包不会清除绝对或数据包延迟定时器，因此一个数据包可能会产生两个中断，一个是由于小数据包接收，另一个是由于定时器超时。

3.2.7.3 Receive Descriptor Minimum Threshold (ICR.RXDMT)

The minimum descriptor threshold helps avoid descriptor under-run by generating an interrupt when the number of free descriptors becomes equal to the minimum amount defined in RCTL.RDMTS (measured as a fraction of the receive descriptor ring size).

3.2.7.4 Receiver FIFO Overrun

FIFO overrun occurs when hardware attempts to write a byte to a full FIFO. An overrun could indicate that software has not updated the tail pointer to provide enough descriptors/buffers, or that the PCI bus is too slow draining the receive FIFO. Incoming packets that overrun the FIFO are dropped and do not affect future packet reception.

3.2.8 82544GC/EI Receive Interrupts

The presence of new packets is indicated by the following:

- Absolute timer (RDTR) — A predetermined amount of time has elapsed since the first packet received after the hardware timer was written (specifically, after the last packet data byte was written to memory); this also flushes any accumulated descriptors to memory. Software can set the timer value to 0b if it wants to be notified each time a new packet has been stored in memory.

  Writing the absolute timer with its high order bit 1 forces an explicit flush of any partial cache lines. Hardware writes all used descriptors to memory and updates the globally visible value of the head pointer.

In addition, hardware provides the following interrupts:

- Receive Descriptor Minimum Threshold (ICR.RXDMT)

  The minimum descriptor threshold helps avoid descriptor underrun by generating an interrupt when the number of free descriptors becomes equal to the minimum. It is measured as a fraction of the receive descriptor ring size.
- Receiver FIFO Overrun (ICR.RXO)

  FIFO overrun occurs when hardware attempts to write a byte to a full FIFO. An overrun could indicate that software has not updated the tail pointer to provide enough descriptors/buffers, or that the PCI bus is too slow draining the receive FIFO. Incoming packets that overrun the FIFO are dropped and do not affect future packet reception.

3.2.9 Receive Packet Checksum Offloading

The Ethernet controller supports the offloading of three receive checksum calculations: the Packet Checksum, the IP Header Checksum, and the TCP/UDP Checksum.

Note: IPv6 packets do not have IP checksums.

3.2.7.3 接收描述符最小阈值 (ICR.RXDMT)

最小描述符阈值通过在空闲描述符数量等于 RCTL.RDMTS 中定义的最小值（以接收描述符环大小的一部分来衡量）时生成中断，来避免描述符欠载。

3.2.7.4 接收器FIFO溢出

FIFO溢出发生在硬件尝试向满的FIFO写入一个字节时。溢出可能表示软件未更新尾指针以提供足够的描述符/缓冲区，或者PCI总线清空接收FIFO太慢。溢出FIFO的传入数据包会被丢弃，并且不会影响未来数据包接收。

3.2.8 82544GC/EI接收中断

新数据包的存在通过以下方式指示：

- 绝对定时器 (RDTR) — 自硬件定时器写入后（具体来说，自最后一个数据包数据字节写入内存后）收到的第一个数据包以来经过了一预定的时长；这也将所有累积的描述符刷新到内存。如果软件希望每次新数据包存储到内存时都收到通知，可以将定时器值设置为0b。

  将绝对定时器的高位设置为1会强制刷新任何未完成的缓存行。硬件将所有使用的描述符写入内存并更新全局可见的头指针值。

此外，硬件还提供以下中断：

- 接收描述符最小阈值 (ICR.RXDMT)

  最小描述符阈值通过在空闲描述符数量等于最小值时生成中断来帮助避免描述符下溢。它以接收描述符环大小的分数形式进行测量。
- 接收器FIFO溢出 (ICR.RXO)

  当硬件尝试向满的FIFO写入一个字节时会发生FIFO溢出。溢出可能
  表明软件未更新尾指针以提供足够的描述符/缓冲区，或者PCI总线清空接收FIFO太慢。入站数据包如果溢出
  FIFO被丢弃，并且不会影响未来数据包接收。

3.2.9 接收数据包校验和卸载

以太网控制器支持三种接收校验和计算卸载：数据包校验和、IP头校验和以及TCP/UDP校验和。

注意：IPv6 数据包没有 IP 校验和。

The Packet checksum is the one's complement over the receive packet, starting from the byte indicated by RXCSUM.PCSS (0b corresponds to the first byte of the packet), after stripping. For example, for an Ethernet II frame encapsulated as an 802.3ac VLAN packet and with RXCSUM.PCSS set to 14 decimal, 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 back out 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 may be offloaded to the Ethernet controller. If RXCSUM.IPOFLD is set to 1b, the controller calculates the IP checksum and indicates a pass/fail condition to software by means of the IP Checksum Error bit (RDESC.IPE) in the ERROR field of the receive descriptor. Similarly, if the RXCSUM.TUOFLD is set to 1b, the Ethernet controller calculates the TCP or UDP checksum and indicates a pass/fail condition to software by means of the TCP/UDP Checksum Error bit (RDESC.TCPE). These error bits are valid when the respective status bits indicate the checksum was calculated for the packet (RDESC.IPCS and RDESC.TCPCS).

If neither RXCSUM.IPOFLD nor RXCSUM.TUOFLD is set, the Checksum Error bits (IPE and TCPE) is 0b for all packets.

Supported Frame Types include:

- Ethernet II
- Ethernet SNAP

Note: See Table 3-6 for the **82544GC/EI** supported receive checksum capabilities.

Table 3-5. Supported Receive Checksum Capabilities

Packet Type	HW IP Checksum Calculation	HW TCP/UDP 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 Fragment	 No (n/a) No (n/a) No (n/a) No (n/a)	 Yes Yes Yes No
IPv4 tunnels: IPv4 packet in an IPv4 tunnel IPv6 packet in an IPv4 tunnel	 No Yes (IPv4)	 No Yes[a]
IPv6 tunnels: IPv4 packet in an IPv6 tunnel IPv6 packet in an IPv6 tunnel	 No No	 No No
Packet is an IPv4 fragment	Yes	No
Packet is greater than 1552 bytes; (LPE=1b)[b]	Yes	Yes
Packet has 802.3ac tag	Yes	Yes

数据包校验和是对接收数据包进行的一补码运算，从 RXCSUM.PCSS 指示的字节开始（0b 对应数据包的第一个字节），在剥离后。例如，对于一个封装为 802.3ac VLAN 数据包的以太网II帧，且 RXCSUM.PCSS 设置为 14 十进制，数据包校验和将包括整个封装的帧，不包括 14 字节的以太网头部（DA,SA,类型/长度）和 4 字节的 q 标签。如果 RCTL.SECRC 位被设置，数据包校验和不包括以太网 CRC。

软件必须在进行数据包校验和与数据包中存储的TCP校验和比较之前，执行所需的偏移计算（以排除不应包含的字节并包含伪头部）。

对于支持的数据包/帧类型，整个校验和计算可以卸载到以太网控制器。如果 RXCSUM.IPOFLD设置为1b，控制器计算IP校验和，并通过接收描述符ERROR字段的 IP校验和错误位（RDESC.IPE）向软件指示通过/失败条件。类似地，如果 RXCSUM.TUOFLD设置为1b，以太网控制器计算TCP或UDP校验和，并通过 TCP/UDP校验和错误位（RDESC.TCPE）向软件指示通过/失败条件。当相应的状态位指示已为数据包计算校验和时（RDESC.IPCS和RDESC.TCPCS），这些错误位有效。

如果RXCSUM.IPOFLD和RXCSUM.TUOFLD都没有设置，则所有数据包的校验和错误位（IPE和TCPE）为0b。

支持的帧类型包括:

- 以太网II
- 以太网SNAP

注意：有关 82544GC/EI 支持的接收校验和功能，请参阅表 3-6。

表3-5。支持的接收校验和功能

数据包类型	硬件IP校验和计算	硬件 TCP/UDP 校验和计算
IPv4 数据包	Yes	Yes
IPv6 数据包	不 (不适用)	Yes
IPv6 数据包带有下一个头部选项: 逐跳选项 目的选项 路由 分片	 不 (不适用) 不 (不适用) 不 (不适用) 不 (不适用)	 Yes Yes Yes No
IPv4 隧道: IPv4 数据包在 IPv4 隧道中 IPv6 数据包在 IPv4 隧道中	 No 是 (IPv4)	 No Yes[a]
IPv6隧道: IPv6隧道中的IPv4数据包 IPv6隧道中的IPv6数据包	 No No	 No No
数据包是IPv4碎片	Yes	No
数据包大于1552字节; (LPE=1b)[b]	Yes	Yes
数据包有802.3ac标签	Yes	Yes

Table 3-5. Supported Receive Checksum Capabilities

Packet Type	HW IP Checksum Calculation	HW TCP/UDP Checksum Calculation
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 # other than TCP or UDP.	Yes	No

a. The IPv6 header portion can include supported extension headers as described in the IPv6 Filter section.
b.For the **82541xx** and **82547GI/EI**, frame sizes greater than 2 KB require full-duplex operation.

Table 3-6. 82544GC/EI Supported Receive Checksum Capabilities

Packet Type	HW IP Checksum Calculation	HW TCP/UDP Checksum Calculation
IP v4 packets	Yes	Yes
IP v6 packets (no IP checksum in IPv6)	No	No
Packet is an IP fragment	Yes	No
Packet is greater than 1552 bytes; (LPE=1)	Yes	Yes
Packet has 802.3ac tag	Yes	Yes
Packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet has TCP or UDP options	Yes	Yes
IP header's protocol field contains a protocol other than TCP or UDP.	Yes	No

Table 3-5 lists the general details about what packets are processed. In more detail, the packets are passed through a series of filters (Section 3.2.9.1 through Section 3.2.9.5) to determine if a receive checksum is calculated.

Note: (Section 3.2.9.1 through Section 3.2.9.5) does not apply to the **82544GC/EI**.

3.2.9.1 MAC Address Filter

This filter checks the MAC destination address to be sure it is valid (IA match, broadcast, multicast, etc.). The receive configuration settings determine which MAC addresses are accepted. See the various receive control configuration registers such as RCTL (RCTL.UPE, RCTL.MPE, RCTL.BAM), MTA, RAL, and RAH.

表3-5。支持的接收校验和功能

数据包类型	硬件IP校验和计算	硬件 TCP/UDP 校验和计算
IPv4数据包有IP选项 (IP头部超过20字节)	Yes	Yes
数据包包含TCP或UDP选项	Yes	Yes
IP头部协议字段包含除TCP或UDP以外的协议#。	Yes	No

a. IPv6头部部分可以包含IPv6过滤器中描述的受支持扩展头部。
部分。
b. 对于82541xx和82547GI/EI，大于2 KB的帧大小需要全双工操作。

表 3-6。 82544GC/EI 支持的接收校验和功能

数据包类型	硬件IP校验和计算	硬件 TCP/UDP 校验和计算
IPv4 数据包	Yes	Yes
IPv6 数据包 (IPv6中无IP校验和)	No	No
数据包是IP分片	Yes	No
数据包大于1552字节；（LPE=1)	Yes	Yes
数据包有 802.3ac 标签	Yes	Yes
数据包有 IP 选项 (IP 头部超过 20 字节)	Yes	Yes
数据包有 TCP 或 UDP 选项	Yes	Yes
IP 头部的协议字段包含除 TCP 或 UDP 之外的协议。	Yes	No

表 3-5 列出了关于处理的数据包的详细信息。更详细地说，数据包会通过一系列过滤器（第3节3.2.9.1至第3节3.2.9.5）以确定是否计算接收校验和。

注意：第 3.2.9.1 节至第 3.2.9.5 节不适用于 82544GC/EI。

3.2.9.1 MAC地址过滤器

此过滤器检查MAC目标地址以确保其有效（IA匹配、广播、多播等）。接收配置设置决定接受哪些MAC地址。请参阅各种接收控制配置寄存器，例如RCTL（RTCL.UPE、RCTL.MPE、RCTL.BAM）、MTA、RAL和RAH。

3.2.9.2 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.

3.2.9.3 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 5 (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.

3.2.9.4 IPv6 Filter

This filter checks for valid IPv6 headers, which are a fixed size and have no checksum. The IPv6 extension headers accepted are: Hop-by-Hop, Destination Options, and Routing. The maximum size next header accepted is 16 dwords (64 bytes).

All of the IPv6 extension headers supported by the Ethernet controller have the same header structure:

Byte 0	Byte 1	Byte 2	Byte 3
Next Header	Hdr Ext Len		

- NEXT HEADER is a value that identifies the header type. The supported IPv6 next headers values are:
  - — Hop-by-Hop = 00h
  - — Destination Options = 3Ch
  - — Routing = 2Bh
- HDR EXT LEN is the 8 byte count of the header length, not including the first 8 bytes. For example, a value of 3 means that the total header size including the NEXT HEADER and HDR EXT LEN fields is 32 bytes (8 + 3*8).
  - — The RXCSUM.IPV6OFL bit must be set for this filter to pass.

3.2.9.5 UDP/TCP Filter

This filter checks for a valid UDP or TCP header. The prototype next header values are 11h and 06h, respectively. The RXCSUM.TUOFL bit must be set for this filter to pass.

3.3 Packet Transmission

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.

3.2.9.2 SNAP/VLAN过滤器

此过滤器检查下一个头部以查找IP头部。它能够解码以太网II、以太网SNAP和IEEE 802.3ac头部。它会跳过这些中间头部并查找IP头部。接收配置设置决定接受哪些下一个头部。请参阅各种接收控制配置寄存器，例如RCTL（RCTL.VFE）、VET和VFTA。

3.2.9.3 IPv4过滤器

此过滤器检查有效的IPv4头部。版本字段检查是否为正确值（4）。IPv4头部如果大小大于或等于5（dwords），则被接受。如果IPv4头部被正确解码，则检查IP校验和的有效性。RXCSUM.IPOFL位必须被设置，此过滤器才能通过。

3.2.9.4 IPv6过滤器

此过滤器检查有效的IPv6头部，IPv6头部大小固定且无校验和。接受的IPv6扩展头部有：逐跳选项、目的选项和路由。接受的最大下一个头部大小为16 dwords（64字节）。

以太网控制器支持的IPv6扩展头部都具有相同的头部结构：

字节0	字节1	字节2	字节3
下一个头部	头部扩展长度		

- 下一个头部是一个标识头部类型的值。支持的IPv6下一个头部值有：
  - — 逐跳选项 = 00h
  - — 目的选项 = 3Ch
  - — 路由 = 2Bh
- 头部扩展长度是头部长度的8字节计数，不包括前8字节。例如，值为3表示包括NEXT HEADERS和HDR EXT LEN字段的整个头部长度为32字节(8 + 3*8)。
  - — 此过滤器要生效，必须设置RXCSUM.IPV6OFL位。

3.2.9.5 UDP/TCP过滤器

此过滤器检查有效的UDP或TCP头部。原型下一个头部值分别为11h和06h。RXCSUM.TUOFL位必须被设置此过滤器才能通过。

3.3 数据包传输

常规（非TCP分段数据包）的传输过程包括：

- 协议栈从应用程序接收一个要传输的数据块。

- 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 the driver to send the individual packet.
  - — The driver gets the frame and interfaces with the hardware.
  - — The hardware reads the packet from host memory (via DMA transfers).
  - — The driver returns ownership of the packet to the Network Operating System (NOS) when the hardware has completed the DMA transfer of the frame (indicated by an interrupt).

Output packets are made up of pointer–length pairs constituting a descriptor chain (so called descriptor based transmission). Software forms transmit packets by assembling the list of pointer–length pairs, storing this information in the transmit descriptor, and then updating the on–chip transmit tail pointer to the descriptor. The transmit descriptor and buffers are stored in host memory. Hardware typically transmits the packet only after it has completely fetched all packet data from host memory and deposited it into the on-chip transmit FIFO. This permits TCP or UDP checksum computation, and avoids problems with PCI underruns.

3.3.1 Transmit Data Storage

Data are stored in buffers pointed to by the descriptors. Alignment of data is on an arbitrary byte boundary with the maximum size per descriptor limited only to the maximum allowed packet size (16288 bytes). A packet typically consists of two (or more) descriptors, one (or more) for the header and one for the actual data. Some software implementations copy the header(s) and packet data into one buffer and use only one descriptor per transmitted packet.

3.3.2 Transmit Descriptors

The Ethernet controller provides three types of transmit descriptor formats.

The original descriptor is referred to as the “legacy” descriptor format. The two other descriptor types are collectively referred to as extended descriptors. One of them is similar to the legacy descriptor in that it points to a block of packet data. This descriptor type is called the TCP/IP Data Descriptor and is a replacement for the legacy descriptor since it offers access to new offloading capabilities. The other descriptor type is fundamentally different as it does not point to packet data. It merely contains control information which is loaded into registers of the controller and affect the processing of future packets. The following sections describe the three descriptor formats.

The extended descriptor types are accessed by setting the TDESC.DEXT bit to 1b. If this bit is set, the TDESC.DTYP field is examined to control the interpretation of the remaining bits of the descriptor. Table 3-7 shows the generic layout for all extended descriptors. Fields marked as NR are not reserved for any particular function and are defined on a per-descriptor type basis. Notice that the DEXT and DTYP fields are non-contiguous in order to accommodate legacy mode operation. For legacy mode operation, bit 29 is set to 0b and the descriptor is defined in Section 3.3.3.

- 协议栈根据媒体MTU大小和所需数据包头计算传输此数据块所需的包数量。媒体MTU大小和所需数据包头。
- 对于数据块中的每个数据包：
  - – 以太网、IP和TCP/UDP头由栈准备。
  - – 栈与软件设备驱动程序接口，并命令驱动程序发送单个数据包。
  - – 驱动程序获取帧并与硬件接口。
  - – 硬件从主机内存读取数据包（通过DMA传输）。
  - – 当硬件完成帧的DMA传输时（通过中断指示），驱动程序将数据包的所有权返回给网络操作系统（NOS）。硬件完成帧的DMA传输时（通过中断指示）。

输出数据包由构成描述符链的指针-长度对组成（即基于描述符的传输）。软件通过组装指针-长度对列表，将此信息存储在传输描述符中，然后更新片上传输尾指针到该描述符。传输描述符和缓冲区存储在主机内存中。硬件通常只有在完全从主机内存中获取所有数据包数据并将其存入片上传输FIFO后才会传输数据包。这允许进行TCP或UDP校验和计算，并避免PCI欠载问题。

3.3.1 传输数据存储

数据存储在由描述符指向的缓冲区中。数据对齐在任意字节边界上，每个描述符的大小仅受最大允许数据包大小（16288字节）的限制。一个数据包通常由两个（或更多）描述符组成，一个（或更多）用于头部，一个用于实际数据。一些软件实现将头部和数据包数据复制到一个缓冲区中，并且每个传输的数据包只使用一个描述符。

3.3.2 传输描述符

以太网控制器提供三种类型的传输描述符格式。

原始描述符称为“传统”描述符格式。其他两种描述符类型统称为扩展描述符。其中一种是类似于传统描述符的，它指向一个数据包数据块。这种描述符类型称为TCP/IP数据描述符，它是传统描述符的替代品，因为它提供了新的卸载功能。另一种描述符类型根本不同，因为它不指向数据包数据。它仅包含控制信息，这些信息被加载到控制器的寄存器中，并影响未来数据包的处理。以下几节将描述三种描述符格式。

通过将TDESC.DEXT位设置为1b来访问扩展描述符类型。如果此位被设置，将检查TDESC.DTYP字段以控制描述符其余位的解释。表3-7显示了所有扩展描述符的通用布局。标记为NR的字段不保留用于任何特定功能，并且基于每个描述符类型定义。请注意，DEXT和DTYP字段是非连续的，以便适应传统模式操作。对于传统模式操作，位29设置为0b，并且描述符在第3节3.3.3中定义。

Table 3-7. Transmit Descriptor (TDESC) Layout

	63	30 29	28 24	23 20	19 0
0	Buffer Address [63:0]				
8	NR	DEXT	NR	DTYP	NR

3.3.3 Legacy Transmit Descriptor Format

To select legacy mode operation, bit 29 (TDESC.DEXT) should be set to 0b. In this case, the descriptor format is defined as shown in Table 3-8. The address and length must be supplied by software. Bits in the command byte are optional, as are the Checksum Offset (CSO), and Checksum Start (CSS) fields.

Table 3-8. Transmit Descriptor (TDESC) Layout – Legacy Mode

	63 48	47 40	39 36	35 32	31 24	23 16	15 0
0	Buffer Address [63:0]						
8	Special	CSS	RSV	STA	CMD	CSO	Length

Table 3-9. Transmit Descriptor Legacy Descriptions

Transmit Descriptor Legacy	Description
Buffer Address	Buffer Address Address of the transmit descriptor in the host memory. Descriptors with a null address transfer no data. If they have the RS bit in the command byte set (TDESC.CMD), then the DD field in the status word (TDESC.STATUS) is written when the hardware processes them.
Length	Length is per segment. The maximum length associated with any single legacy descriptor is 16288 bytes. Although a buffer as short as one byte is allowed, the total length of the packet, before padding and CRC insertion must be at least 48 bytes. Length can be up to a default value of 16288 bytes per descriptor, and 16288 bytes total. In other words, the length of the buffer pointed to by one descriptor, or the sum of the lengths of the buffers pointed to by the descriptors can be as large as the maximum allowed transmit packet. Descriptors with zero length transfer no data. If they have the RS bit in the command byte set (TDESC.CMD), then the DD field in the status word (TDESC.STATUS) is written when the hardware processes them.
CSO	Checksum Offset The Checksum offset field indicates where, relative to the start of the packet, to insert a TCP checksum if this mode is enabled. (Insert Checksum bit (IC) is set in TDESC.CMD). Hardware ignores CSO unless EOP is set in TDESC.CMD. CSO is provided in unit of bytes and must be in the range of the data provided to the Ethernet controller in the descriptor. (CSO < length - 1). Should be written with 0b for future compatibility.

表3-7。传输描述符（TDESC）布局

	63 30	29	28 24	23 20	19 0
0	缓冲地址 [63:0]				
8	NR	DEXT	NR	DTYP	NR

3.3.3 传统传输描述符格式

要选择传统模式操作，位29（TDESC.DEXT）应设置为0b。在这种情况下，描述符格式定义如表3-8所示。地址和长度必须由软件提供。命令字中的位是可选的，校验和偏移（CSO）和校验和开始（CSS）字段也是可选的。

表3-8。传输描述符（TDESC）布局——传统模式

	63 48	47 40	39 36	35 32	31 24	23 16	15 0
0	缓冲地址 [63:0]						
8	特殊	CSS	RSV	STA	CMD	CSO	长度

表3-9. 传输描述符传统描述

传输描述符传统	描述
缓冲地址	缓冲地址 主机内存中的传输描述符地址。地址为空的描述符不传输数据。如果它们在命令字节中设置了RS位（TDESC.CMD），则在硬件处理它们时，状态字（TDESC.STATUS）中的DD字段会被写入。
长度	长度是按分段计算的。 任何单个传统描述符关联的最大长度是16288字节。尽管允许最短为1字节的缓冲区，但数据包的总长度 在填充和CRC插入之前必须至少为48字节。 长度可以达到每个描述符16288字节的默认值，以及总共16288字节。换句话说，一个描述符指向的缓冲区的长度，或多个描述符指向的缓冲区的长度之和可以达到最大允许传输数据包的大小。 零长度传输的描述符没有数据。如果它们在命令字节的RS位被设置（TDESC.CMD），则状态字中的DD字段 (TDESC.STATUS) 由硬件在处理时写入。
CSO	校验和偏移 校验和偏移字段指示，如果启用此模式，应相对于数据包的起始位置插入TCP校验和。（TDESC.CMD中的插入校验和位（IC）被设置）。硬件忽略CSO，除非TDESC.CMD中设置了EOP。CSO以字节为单位提供，并且必须在描述符提供给以太网控制器数据的范围内。（CSO < 长度 - 1)。 应使用0b写入以保持未来兼容性。

Transmit Descriptor Legacy	Description
CMD	Command field See Section 3.3.3.1 for a detailed field description.
STA	Status field See Section 3.3.3.2 for a detailed field description.
RSV	Reserved Should be written with 0b for future compatibility.
CSS	Checksum Start Field The Checksum start field (TDESC.CSS) indicates where to begin computing the checksum. The software must compute this offset to back out the bytes that should not be included in the TCP checksum. CSS is provided in units of bytes and must be in the range of data provided to the Ethernet controller in the descriptor (CSS < length). For short packets that ar padded by the software, CSS must be in the range of the unpadded data length. A value of 0b corresponds to the first byte in the packet. CSS must be set in the first descriptor of the packet.
Special	Special Field See the notes that follow this table for a detailed field description.

Notes:

1. Even though CSO and CSS are in units of bytes, the checksum calculation typically works on 16-bit words. Hardware does not enforce even byte alignment.
2. 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 on the encapsulated packet rather than on the added fields.
3. Although the Ethernet controller can be programmed to calculate and insert TCP checksum using the legacy descriptor format as described above, it is recommended that software use the newer TCP/IP Context Transmit Descriptor Format. This newer descriptor format allows the hardware to calculate both the IP and TCP checksums for outgoing packets. See Section 3.3.5 for more information about how the new descriptor format can be used to accomplish this task.

传输描述符传统	描述
CMD	命令字段 请参阅第3.3.3.1节以获取详细字段描述。
STA	状态字段 请参阅第3.3.3.2节以获取详细字段描述。
RSV	保留 应使用0b编写以保持未来兼容性。
CSS	校验和起始字段 校验和起始字段 (TDESC.CSS) 指示从何处开始计算校验和。软件必须计算此偏移量，以排除不应包含在 TCP 校验和中的字节。CSS 以字节为单位提供，并且必须在描述符提供给以太网控制器（CSS < 长度）的数据范围内。对于由软件填充的短数据包，CSS 必须在未填充数据长度范围内。0b 的值对应数据包中的第一个字节。 CSS 必须在数据包的第一个描述符中设置。
特殊	特殊字段 请参阅此表格下方的注释，以获取字段的详细描述。

注意：

1. 即使CSO和CSS以字节为单位，校验和计算通常在16位字上工作。硬件不会强制执行偶数字节对齐。
2. 硬件不会将802.1Q以太类型或其后的VLAN字段添加到校验和中。因此，对于VLAN数据包，软件只能在封装的数据包上计算值，而不是在添加的字段上。
3. 尽管以太网控制器可以被编程为使用上述传统描述符格式计算并插入TCP校验和，但建议软件使用更新的TCP/IP上下文传输描述符格式。这种新的描述符格式允许硬件计算出站数据包的IP和TCP校验和。有关如何使用新的描述符格式完成此任务的更多信息，请参阅第3.3.5节。

3.3.3.1 Transmit Descriptor Command Field Format

The CMD byte stores the applicable command and has fields shown in Table 3-10.

Table 3-10. Transmit Command (TDESC.CMD) Layout

7	6	5	4	3	2	1	0
IDE	VLE	DEXT	RSV RPS[a]	RS	IC	IFCS	EOP

a. **82544GC/EI** only.

TDESC.CMD	Description
IDE (bit 7)	Interrupt Delay Enable When set, activates the transmit interrupt delay timer. The Ethernet controller loads a countdown register when it writes back a transmit descriptor that has RS and IDE set. The value loaded comes from the IDV field of the Interrupt Delay (TIDV) register. When the count reaches 0, a transmit interrupt occurs if transmit descriptor write-back interrupts (IMS.TXDW) are enabled. Hardware always loads the transmit interrupt counter whenever it processes a descriptor with IDE set even if it is already counting down due to a previous descriptor. If hardware encounters a descriptor that has RS set, but not IDE, it generates an interrupt immediately after writing back the descriptor. The interrupt delay timer is cleared.
VLE (bit 6)	VLAN Packet Enable When set, indicates that the packet is a VLAN packet and the Ethernet controller should add the VLAN Ethertype and an 802.1q VLAN tag to the packet. The Ethertype field comes from the VET register and the VLAN tag comes from the special field of the TX descriptor. The hardware inserts the FCS/CRC field in that case. When cleared, the Ethernet controller sends a generic Ethernet packet. The IFCS controls the insertion of the FCS field in that case. In order to have this capability CTRL.VME bit should also be set, otherwise VLE capability is ignored. VLE is valid only when EOP is set.
DEXT (bit 5)	Extension (0b for legacy mode). Should be written with 0b for future compatibility.
RPS RSV (bit 4)	Report Packet Sent When set, the **82544GC/EI** defers writing the DD bit in the status byte (DESC.STATUS) until the packet has been sent, or transmission results in an error such as excessive collisions. It is used is cases where the software must know that the packet has been sent, and not just loaded to the transmit FIFO. The **82544GC/EI** might continue to prefetch data from descriptors logically after the one with RPS set, but does not advance the descriptor head pointer or write back any other descriptor until it sent the packet with the RPS set. RPS is valid only when EOP is set. This bit is reserved and should be programmed to 0b for all Ethernet controllers except the **82544GC/EI**.
RS (bit 3)	Report Status When set, the Ethernet controller needs to report the status information. This ability may be used by software that does in-memory checks of the transmit descriptors to determine which ones are done and packets have been buffered in the transmit FIFO. Software does it by looking at the descriptor status byte and checking the Descriptor Done (DD) bit.

3.3.3.1 传输描述符命令字段格式

CMD字节存储适用的命令，其字段如表3-10所示。

表3-10。传输命令（TDESC.CMD）布局

7	6	5	4	3	2	1	0
IDE	VLE	DEXT	RSV RPS[a]	RS	IC	IFCS	EOP

a. 仅82544GC/EI。

TDESC.CMD	描述
IDE (bit 7)	中断延迟使能 当设置时，激活传输中断延迟定时器。以太网控制器在回写具有RS和IDE设置的传输描述符时，会加载一个倒计时寄存器。加载的值来自中断延迟（TIDV）寄存器的IDV字段。当计数器达到0时，如果传输描述符写回中断（IMS.TXDW）被使能，则会发生传输中断。硬件在处理具有IDE设置的描述符时，无论是否由于先前的描述符正在倒数，始终会加载传输中断计数器。如果硬件遇到具有RS设置但IDE未设置的描述符，它会在回写描述符后立即生成中断。中断延迟定时器被清除。
VLE (bit 6)	VLAN数据包使能 当设置时，指示该数据包是VLAN数据包，并且以太网控制器应该向数据包中添加VLAN以太类型和802.1q VLAN标签。以太类型字段来自VET寄存器，VLAN标签来自TX描述符的特殊字段。在这种情况下，硬件插入FCS/CRC字段。 当清除时，以太网控制器发送通用以太网数据包。IFCS控制该情况下FCS字段的插入。 为了具有此功能，CTRL.VME位也应该设置，否则VLE功能被忽略。VLE仅当EOP设置时有效。
DEXT（位5）	扩展（传统模式为0b）。 应使用0b编写以保持未来兼容性。
RPS RSV（位4）	报告包发送 当设置时，82544GC/EI会推迟在状态字节（DESC.STATUS）中写入DD位，直到数据包已发送，或传输结果导致错误，如过度冲突。它用于软件必须知道数据包已发送的情况，而不仅仅是加载到发送FIFO的情况。82544GC/EI可能会继续从逻辑上具有RPS设置的描述符中预取数据，但不会移动描述符头指针或回写任何其他描述符，直到它发送了具有RPS设置的包。RPS仅在EOP设置时有效。 此位是保留位，除82544GC/EI外，所有以太网控制器都应编程为0b。
RS（位3）	报告状态 当设置时，以太网控制器需要报告状态信息。这种能力可能被用于对传输描述符进行内存检查的软件，以确定哪些描述符已完成，并且数据包已缓冲在发送FIFO中。软件通过查看描述符状态字节并检查描述符完成（DD）位来进行操作。

TDESC.CMD	Description
IC (bit 2)	Insert Checksum When set, the Ethernet controller needs to insert a checksum at the offset indicated by the CSO field. The checksum calculations are performed for the entire packet starting at the byte indicated by the CCS field. IC is ignored if CSO and CCS are out of the packet range. This occurs when (CSS ≥ length) OR (CSO ≥ length - 1). IC is valid only when EOP is set.
IFCS (bit 1)	Insert FCS Controls the insertion of the FCS/CRC field in normal Ethernet packets. IFCS is valid only when EOP is set.
EOP (bit 0)	End Of Packet When set, indicates the last descriptor making up the packet. One or many descriptors can be used to form a packet.

Notes:

1. VLE, IFCS, and IC are qualified by EOP. That is, hardware interprets these bits ONLY when EOP is set.
2. Hardware only sets the DD bit for descriptors with RS set.
3. Descriptors with the null address (0b) or zero length transfer no data. If they have the RS bit set then the DD field in the status word is written when hardware processes them.
4. Although the transmit interrupt may be delayed, the descriptor write-back requested by setting the RS bit is performed without delay unless descriptor write-back bursting is enabled.

3.3.3.2 Transmit Descriptor Status Field Format

The STATUS field stores the applicable transmit descriptor status and has the fields shown in Table 3-11.

The transmit descriptor status field is only present in cases where RS (or RPS for the **82544GC/EI** only) is set in the command field.

Table 3-11. Transmit Status Layout

3	2	1	0
RSV TU[a]	LC	EC	DD

a. **82544GC/EI** only.

TDESC.CMD	描述
IC（位2）	插入校验和 当设置时，以太网控制器需要在CSO字段指示的偏移量处插入校验和。校验和计算从CCS字段指示的字节开始对整个数据包进行。如果CSO和CCS超出数据包范围，则忽略IC。这发生在(CSS ≥ 长度) OR (CSO ≥ 长度 - 1)时。IC仅当EOP设置时有效。
IFCS (位0)	插入FCS 控制普通以太网数据包中FCS/CRC字段的插入。IFCS仅当EOP设置时有效。
EOP (位0)	数据包结束 当设置时，指示构成数据包的最后一个描述符。一个或多个描述符可用于形成数据包。

注意：

1. VLE、IFCS 和 IC 由 EOP 限定。也就是说，硬件仅在 EOP 设置时才解释这些位。
2. 硬件仅对 RS 设置的描述符设置 DD 位。
3. 具有空地址（0b）或零长度传输的描述符不传输数据。如果它们设置了 RS 位，则在硬件处理它们时，状态字中的 DD 字段会被写入。
4. 虽然传输中断可能会延迟，但通过设置RS位请求的描述符回写不会延迟，除非启用了描述符回写突发。

3.3.3.2 传输描述符状态字段格式

STATUS字段存储适用的传输描述符状态，其字段如表3-11所示。

传输描述符状态字段仅在命令字段中设置了RS（对于82544GC/EI仅设置RPS）时存在。

表3-11。传输状态布局

3	2	1	0
RSV TU[a]	LC	EC	DD

a. 82544GC/EI 仅限。

TDESC.STATUS	Description
TU RSV (bit 3)	Transmit Underrun Indicates a transmit underrun event occurred. Transmit Underrun might occur if Early Transmits are enabled (based on ETT.Txthreshold value) and the **82544GC/EI** was not able to complete the early transmission of the packet due to lack of data in the packet buffer. This does not necessarily mean the packet failed to be eventually transmitted. The packet is successfully re-transmitted if the TCTL.NRTU bit is cleared (and excessive collisions do not occur). This bit is reserved and should be programmed to 0b for all Ethernet controllers except the **82544GC/EI**.
LC (bit 2)	Late Collision Indicates that late collision occurred while working in half-duplex mode. It has no meaning while working in full-duplex mode. Note that the collision window is speed dependent: 64 bytes for 10/100 Mb/s and 512 bytes for 1000 Mb/s operation.
EC (bit 1)	Excess Collisions Indicates that the packet has experienced more than the maximum excessive collisions as defined by TCTL.CT control field and was not transmitted. It has no meaning while working in full-duplex mode.
DD (bit 0)	Descriptor Done Indicates that the descriptor is finished and is written back either after the descriptor has been processed (with RS set) or for the **82544GC/EI**, after the packet has been transmitted on the wire (with RPS set).

Note: The DD bit reflects status of all descriptors up to and including the one with the RS bit set (or RPS for the **82544GC/EI**).

3.3.4 Transmit Descriptor Special Field Format

The SPECIAL field is used to provide the 802.1q/802.1ac tagging information.

When CTRL.VME is set to 1b, all packets transmitted from the Ethernet controller that have VLE set in the TDESC.CMD are sent with an 802.1Q header added to the packet. The contents of the header come from the transmit descriptor special field and from the VLAN type register. The special field is ignored if the VLE bit in the transmit descriptor command field is 0b. The special field is valid only for descriptors with EOP set to 1b in TDESC.CMD.

Table 3-12. Special Field (TDESC.SPECIAL) Layout

15 13	12	11 0
PRI	CFI	VLAN

TDESC.SPECIAL	Description
PRI	User Priority 3 bits that provide the VLAN user priority field to be inserted in the 802.1Q tag.
CFI	Canonical Form Indicator.
VLAN	VLAN Identifier 12 bits that provide the VLAN identifier field to be inserted in the 802.1Q tag.

TDESC.STATUS	描述
TU RSV（位3）	传输欠载 指示发生了传输欠载事件。如果启用了早期传输（基于ETT.Txthreshold值），并且由于数据包缓冲区中缺少数据，82544GC/EI无法完成数据包的早期传输，则可能会发生传输欠载。这并不一定意味着数据包最终未能成功传输。如果TCTL.NRTU位被清除（并且没有发生过度冲突），则数据包会成功重传。 该位为保留位，除82544GC/EI以外的所有以太网控制器应编程为0b。
LC（位2）	晚期冲突 指示在半双工模式下发生了晚期冲突。在全双工模式下无意义。请注意，冲突窗口与速度相关：10/100 Mb/s操作时为64字节，1000 Mb/s操作时为512字节。
EC（位1）	过度冲突 指示该数据包已超过最大过度 根据TCTL.CT控制字段定义的冲突未被传输。它没有 在全双工模式下工作的含义。
数据描述（位0）	描述符完成 指示描述符已完成，并且已写回，要么是在描述符被处理（设置了RS）之后，要么是对于82544GC/EI，在数据包在网络上传输（设置了RPS）之后。

注意： DD位反映了所有描述符（包括RS位被设置的描述符（或RPS）的状态对于82544GC/EI）。

3.3.4 传输描述符特殊字段格式

SPECIAL字段用于提供802.1q/802.1ac的标记信息。

当CTRL.VME设置为1b时，所有从以太网控制器传输且在TDESC.CMD中设置了VLE的数据包都会在数据包中添加802.1Q标题。标题的内容来自传输描述符特殊字段和VLAN类型寄存器。如果传输描述符命令字段中的VLE位为0b，则忽略特殊字段。特殊字段仅对TDESC.CMD中EOP设置为1b的描述符有效。

表3-12。特殊字段（TDESC.SPECIAL）布局

15 13	12	11 0
PRI	CFI	VLAN

TDESC.SPECIAL	描述
PRI	用户优先级 3位，用于提供要插入到802.1Q标签中的VLAN用户优先级字段。
CFI	规范形式指示符。
VLAN	VLAN标识符 12位，用于提供要插入到802.1Q标签中的VLAN标识符字段。

3.3.5 TCP/IP Context Transmit Descriptor Format

The TCP/IP context transmit descriptor provides access to the enhanced checksum offload facility available in the Ethernet controller. This feature allows 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.

The TCP/IP context transmit descriptor does not point to packet data as a data descriptor does. Instead, this descriptor provides access to an on-chip context that supports the transmit checksum offloading feature of the controller. A “context” refers to a set of registers loaded or unloaded as a group to provide a particular function.

The context is explicit and directly accessible via the TCP/IP context transmit descriptor. The context is used to control the checksum offloading feature for normal packet transmission.

The Ethernet controller automatically selects the appropriate legacy or normal context to use based on the current packet transmission.

While the architecture supports arbitrary ordering rules for the various descriptors, there are restrictions including:

- Context descriptors should not occur in the middle of a packet.
- Data descriptors of different packet types (legacy or normal) should not be intermingled except at the packet level.

All contexts control calculation and insertion of up to two checksums. This portion of the context is referred to as the checksum context.

In addition to checksum context, the segmentation context adds information specific to the segmentation capability. This additional information includes the total payload for the message (TDESC.PAYLEN), the total size of the header (TDESC.HDRLEN), the amount of payload data that should be included in each packet (TDESC.MSS), and information about what type of protocol (TCP, IPv4, IPv6, etc.) is used. This information is specific to the segmentation capability and is therefore ignored for context descriptors that do not have the TSE bit set.

Because there are dedicated resources on-chip for the normal context, the context remains constant until it is 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 may be completely unnecessary to load a new context for each packet. For example, if most traffic generated from a given node is standard TCP frames, this context could be set up once and used for many frames. Only when some other frame type is required would a new context need to be loaded by software. After the “non-standard” frame is transmitted, the “standard” context would be setup once more by software. This method avoids the “extra descriptor per packet” penalty for most frames. The penalty can be eliminated altogether if software elects to use TCP/IP checksum offloading only for a single frame type, and thus performs those operations in software for other frame types.

This same logic can also be applied to the segmentation context, though the environment is a more restrictive one. In this scenario, the host is commonly asked to send a message of the same type, TCP/IP for instance, and these messages also have the same total length and same maximum segment size (MSS). In this instance, the same segmentation context could be used for multiple TCP messages that require hardware segmentation. The limitations of this scenario and the relatively small performance advantage make this approach unlikely; however, it is useful in understanding the underlying mechanism.

3.3.5 TCP/IP上下文传输描述符格式

TCP/IP上下文传输描述符提供了对以太网控制器中可用的增强校验和卸载功能的访问。此功能允许通过在硬件中执行额外工作来更高效地处理TCP和UDP数据包类型，从而减少为准备这些数据包进行传输而相关的软件开销。

TCP/IP上下文传输描述符不指向数据描述符所指向的数据包数据。相反，此描述符提供对支持控制器传输校验和卸载功能的片上上下文的访问。“上下文”是指作为一组加载或卸载的寄存器集，以提供特定功能。

上下文是明确的，并且可以通过TCP/IP上下文传输描述符直接访问。上下文用于控制正常数据包传输的校验和卸载功能。

以太网控制器根据当前数据包传输自动选择合适的传统或正常上下文使用。

虽然架构支持各种描述符的任意排序规则，但存在限制，包括：

- 上下文描述符不应出现在数据包中间。
- 不同数据包类型（传统或正常）的数据描述符不应混用
  除非在数据包级别。

所有上下文控制最多两个校验和的计算和插入。这部分上下文称为校验和上下文。

除了校验和上下文，分段上下文还添加了与分段能力相关的特定信息。这些附加信息包括消息的总有效载荷（TDESC.PAYLEN）、头部总大小（TDESC.HDRLEN）、每个数据包应包含的有效载荷数据量（TDESC.MSS）以及所使用的协议类型（TCP、IPv4、IPv6 等）的信息。这些信息是针对分段能力的特定信息，因此对于没有设置 TSE 位的上下文描述符会被忽略。

由于在片上为正常上下文分配了专用资源，上下文保持不变，直到被另一个上下文描述符修改。这意味着一个上下文可以用于多个数据包（或多个分段块），除非在每个新数据包之前加载新的上下文。根据环境，可能完全不需要为每个数据包加载新的上下文。例如，如果从给定节点生成的流量主要是标准 TCP 帧，这个上下文可以设置一次并用于许多帧。只有当需要其他帧类型时，软件才需要加载新的上下文。在传输“非标准”帧后，软件会再次设置“标准”上下文。这种方法避免了大多数帧的“每包额外描述符”惩罚。如果软件选择仅对单个帧类型使用 TCP/IP 校验和卸载，并且因此对其他帧类型在软件中执行这些操作，则可以完全消除这种惩罚。

这种相同的逻辑也可以应用于分段上下文，尽管环境是一个更严格的上下文。在这种情况下，主机通常被要求发送相同类型的消息，例如TCP/IP，并且这些消息具有相同的总长度和相同的最小分段大小（MSS）。在这种情况下，相同的分段上下文可以用于需要硬件分段的多个TCP消息。这种场景的限制以及相对较小的性能优势使得这种方法不太可能；然而，它对于理解底层机制是有用的。

3.3.6 TCP/IP Context Descriptor Layout

The following section describes the layout of the TCP/IP context transmit descriptor.

To select this descriptor format, bit 29 (TDESC.DEXT) must be set to 1b and TDESC.DTYP must be set to 0000b. In this case, the descriptor format is defined as shown in Table 3-13.

Note that the TCP/IP context descriptor does not transfer any packet data. It merely prepares the checksum hardware for the TCP/IP Data descriptors that follow.

Table 3-13. Transmit Descriptor (TDESC) Layout – (Type = 0000b)

	63 48	47 40	39	32	31		16	15 8	7 0
0	TUCSE	TUCSO	TUCSS		IPCSE			IPCSO	IPCSS
8	MSS	HDRLEN	RSV	STA	TUCMD	DTYP	PAYLEN		
	63 48	47 40	39 36	35 32	31 24	23 20	19		0

Note: The first quadword of this descriptor type contains parameters used to calculate the two checksums which may be offloaded.

3.3.6 TCP/IP上下文描述符布局

下一节描述了TCP/IP上下文传输描述符的布局。

要选择此描述符格式，位29（TDESC.DEXT）必须设置为1b，TDESC.DTYP必须设置为0000b。在这种情况下，描述符格式定义如表3-13所示。

请注意，TCP/IP上下文描述符不会传输任何数据包数据。它只是为后续的TCP/IP数据描述符准备校验和硬件。

表 3-13。传输描述符（TDESC）布局 – (类型 = 0000b)

	63 48	47 40	39	32	31		16	15 8	7 0
0	TUCSE	TUCSO	TUCSS		IPCSE			IPCSO	IPCSS
8	MSS	HDRLEN	RSV	STA	TUCMD	DTYP	PAYLEN		
	63 48	47 40	39 36	35 32	31 24	23 20	19		0

注意：此描述符类型的第一个四字节数据包含用于计算可能卸载的两个校验和的参数。

Table 3-14. Transmit Descriptor (TDESC) Layout

Transmit Descriptor Offload	Description
TUCSE	TCP/UDP Checksum Ending Defines the ending byte for the TCP/UDP checksum offload feature. Setting TUCSE field to 0b indicates that the checksum covers from TUCCS to the end of the packet.
TUCSO	TCP/UDP Checksum Offset Defines the offset where to insert the TCP/UDP checksum field in the packet data buffer. This is used in situations where the software needs to calculate partial checksums (TCP pseudo-header, for example) to include bytes which are not contained within the range of start and end. If no partial checksum is required, software must write a value of 0b.
TUCSS	TCP/UDP Checksum Start Defines the starting byte for the TCP/UDP checksum offload feature. It must be defined even if checksum insertion is not desired for some reason. When setting the TCP segmentation context, TUCSS is used to indicate the start of the TCP header.
IPCSE	IP Checksum Ending Defines the ending byte for the IP checksum offload feature. It specifies where the checksum should stop. A 16-bit value supports checksum offloading of packets as large as 64KB. Setting IPCSE field to 0b indicates that the checksum covers from IPCCS to the end of the packet. In this way, the length of the packet does not need to be calculated.
IPCSO	IP Checksum Offset The IPCSO field specifies where the resulting IP checksum should be placed. It is limited to the first 256 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 checksum is not inserted.
IPCSS	IP Checksum Start IPCSS specifies the byte offset from the start of the transferred data to the first byte in be included in the checksum. Setting this value to 0b means the first byte of the data would be included in the checksum. Note that the maximum value for this field is 255. This is adequate for typical applications. The IPCSS value needs to be less than the total transferred length of the packet. If this is not the case, the results are unpredictable. IPCSS must be defined even if checksum insertion is not desired for some reason. When setting the TCP segmentation context, IPCSS is used to indicate the start of the IP header.
MSS	Maximum Segment Size Controls the Maximum Segment Size. This specifies the maximum TCP or UDP payload “segment” sent per frame, not including any header. The total length of each frame (or “section”) sent by the TCP Segmentation mechanism (excluding 802.3ac tagging and Ethernet CRC) is MSS bytes + HRDLEN. The one exception is the last packet of a TCP segmentation context which is (typically) shorter than “MSS+HDRLEN”. This field is ignored if TDESC.TSE is not set.
HDRLEN	Header Length Specifies the length (in bytes) of the header to be used for each frame (or “section”) of a TCP Segmentation operation. The first HDRLEN bytes fetched from data descriptor(s) are stored internally and used as a prototype header for each section, and are pre-pended to each payload segment to form individual frames. For UDP packets this is normally equal to “UDP checksum offset + 2”. For TCP packets it is normally equal to “TCP checksum offset + 4 + TCP header option bytes”. This field is ignored if TDESC.TSE is not set.

表3-14. 传输描述符（TDESC）布局

传输描述符卸载	描述
TUCSE	TCP/UDP校验和结束 定义了TCP/UDP校验和卸载功能的结束字节。将TUCSE字段设置为0b表示校验和从TUCCS开始到数据包的末尾。
TUCSO	TCP/UDP校验和偏移 定义了TCP/UDP校验和字段在数据包数据缓冲区中的插入偏移量。这用于软件需要计算部分校验和（例如TCP伪头部）的情况，以包含起始和结束范围之外的字节。如果不需要部分校验和，软件必须写入0b的值。
TUCSS	TCP/UDP校验和开始 定义了TCP/UDP校验和卸载功能的起始字节。 即使出于某种原因不希望进行校验和插入，也必须定义它。在设置TCP分段上下文时，TUCSS用于指示TCP头部的开始。
IPCSE	IP校验和结束 定义了IP校验和卸载功能的结束字节。 它指定了校验和应停止的位置。一个16位值支持对最大为64KB的数据包进行校验和卸载。将IPCSE字段设置为0b表示校验和从IPCCS开始到数据包的末尾。这样，就不需要计算数据包的长度。
IPCSO	IP校验和偏移 IPCSO字段指定结果IP校验和应放置的位置。它仅限于数据包的前256字节，并且必须小于或等于给定数据包的总长度。如果不是这样，校验和将不会被插入。
IPCSS	IP校验和开始 IPCSS指定从传输数据的开始到第一个包含在校验和中的字节的字节偏移。将此值设置为0b意味着数据的第一字节将被包含在校验和中。 注意，此字段的最大值是255。这对于典型应用来说是足够的。 IPCSS值需要小于数据包的总传输长度。如果不是这种情况，结果是不可预测的。 即使出于某种原因不希望进行校验和插入，IPCSS也必须定义。在设置TCP分段上下文时，IPCSS用于指示IP头部的开始。
MSS	最大分段大小 控制最大分段大小。这指定了每个帧发送的TCP或UDP有效载荷“分段”，不包括任何头部。TCP分段机制发送的每个帧（或“部分”）的总长度（不包括802.3ac标记和以太网CRC）是MSS字节+ HRDLEN。唯一的例外是TCP分段上下文中的最后一个数据包，它通常比“MSS+HDRLEN”短。如果TDESC.TSE未设置，则忽略此字段。
HDRLEN	头部长度 指定TCP分段操作中每个帧（或“部分”）所使用的头部长度（以字节为单位）。从数据描述符中获取的前HDRLEN个字节将内部存储，并用作每个部分的原型头部，并预置于每个有效载荷分段以形成单个帧。对于UDP数据包，这通常等于“UDP校验和偏移 + 2”。对于TCP数据包，它通常等于“TCP校验和偏移 + 4 + TCP头部选项字节”。如果TDESC.TSE未设置，则忽略此字段。

Transmit Descriptor Offload	Description
RSV	Reserved Should be programmed to 0b for future compatibility.
STA	TCP/UDP Status field Provides transmit status indication. Section 3.3.6.2 provides the bit definition for the TDESC.STA field.
TUCMD	TCP/UDP command field The command field provides options that control the checksum offloading, along with some of the generic descriptor processing functions. Section 3.3.6.1 provides the bit definitions for the TDESC.TUCMD field.
DTYP	Descriptor Type Set to 0000b for TCP/IP context transmit descriptor type.
PAYLEN	The packet length field (TDESC.PAYLEN) is the total number of payload bytes for this TCP Segmentation offload context (i.e., the total number of payload bytes that could be distributed across multiply frames after TCP segmentation is performed). Following the fetch of the prototype header, PAYLEN specifies the length of data that is fetched next from data descriptor(s). This field is also used to determine when "last-frame" processing needs to be performed. Typically, a new data descriptor is used to denote the start of the payload data buffer(s), but this is not required. PAYLEN specification should not include any header bytes. There is no restriction on the overall PAYLEN specification with respect to the transmit FIFO size, once the MSS and HDRLEN specifications are legal. This field is ignored if TDESC.TSE is not set. Refer to Section 3.5 for details on the TCP Segmentation off-loading feature.

Notes:

1. A number of the fields are ignored if the TCP Segmentation enable bit (TDESC.TSE) is cleared, denoting that the descriptor does not refer to the TCP segmentation context.
2. Maximum limits for the HDRLEN and MSS fields are dictated by the lengths variables. However, there is a further restriction that for any TCP Segmentation operation, the hardware must be capable of storing a complete section (completely-built frame) in the transmit FIFO prior to transmission. Therefore, the sum of MSS + HDRLEN must be at least 80 bytes less than the allocated size of the transmit FIFO.

3.3.6.1 TCP/UDP Offload Transmit Descriptor Command Field

The command field (TDESC.TUCMD) provides options to control the TCP segmentation, along with some of the generic descriptor processing functions.

传输描述符卸载	描述
RSV	保留 应编程为0b以实现未来兼容性。
STA	TCP/UDP状态字段 提供传输状态指示。 第3节3.3.6.2提供了TDESC.STA字段的位定义。
TUCMD	TCP/UDP命令字段 命令字段提供了一些用于控制校验和卸载以及通用描述符处理功能的选项。 第3节3.3.6.1提供了TDESC.TUCMD字段的位定义。
DTYP	描述符类型 设置为0000b，用于TCP/IP上下文传输描述符类型。
PAYLEN	数据包长度字段（TDESC.PAYLEN）是此TCP分段卸载上下文（即，TCP分段完成后可在多个帧中分布的有效载荷字节数）的有效载荷字节数总和。在获取原型头部后，PAYLEN指定从数据描述符中获取的下一次数据长度。此字段还用于确定何时需要执行"最后一帧"处理。通常，一个新的数据 描述符用于表示有效载荷数据缓冲区（s）的开始，但这不是必需的。PAYLEN规范不应包含任何头字节。一旦MSS和HDRLEN规范合法，PAYLEN规范在整体上不受传输FIFO大小的限制。如果TDESC.TSE未设置，此字段将被忽略。有关TCP分段卸载功能的详细信息，请参阅第3.5节。

备注：

1. 如果TCP分段启用位（TDESC.TSE）被清除，则某些字段将被忽略，表示该描述符不指向TCP分段上下文。
2. HDRLEN和MSS字段的最高限制由长度变量决定。然而，还有进一步的限制，即对于任何TCP分段操作，硬件必须在传输之前能够在发送FIFO中存储一个完整部分（完全构建的帧）。因此，MSS + HDRLEN的总和必须至少比发送FIFO的分配大小少80字节。

3.3.6.1 TCP/UDP卸载发送描述符命令字段

命令字段（TDESC.TUCMD）提供选项以控制TCP分段，以及一些通用描述符处理功能。

Table 3-15. Command Field (TDESC.TUCMD) Layout

7	6	5	4	3	2	1	0
IDE	RSV	DEXT	RSV	RS	TSE	IP	TCP

TDESC.TUCMD	Description
IDE (bit 7)	Interrupt Delay Enable IDE activates the transmit interrupt delay timer. Hardware loads a countdown register when it writes back a transmit descriptor that has the RS bit and the IDE bit set. The value loaded comes from the IDV field of the Interrupt Delay (TIDV) register. When the count reaches 0, a transmit interrupt occurs. Hardware always loads the transmit interrupt counter whenever it processes a descriptor with IDE set even if it is already counting down due to a previous descriptor. If hardware encounters a descriptor that has RS set, but not IDE, it generates an interrupt immediately after writing back the descriptor. The interrupt delay timer is cleared.
RSV (Bit 6)	Reserved. Set to 0b for future compatibility.
DEXT(Bit 5)	Descriptor Extension Must be 1b for this descriptor type.
RSV (Bit 4)	Reserved. Set to 0b for future compatibility.
RS (Bit 3)	Report Status RS tells the hardware to report the status information for this descriptor. Because this descriptor does not transmit data, only the DD bit in the status word is valid. Refer to Section 3.3.6.2 for the layout of the status field.
TSE (Bit 2)	TCP Segmentation Enable TSE indicates that this descriptor is setting the TCP segmentation context. If this bit is not set, the checksum offloading context for normal (non-"TCP Segmentation") packets is written. When a descriptor of this type is processed the Ethernet controller immediately updates the context in question (TCP Segmentation or checksum offloading) with values from the descriptor. This means that if any normal packets or TCP Segmentation packets are in progress (a descriptor with EOP set has not been received for the given context), the results are likely to be undesirable.
IP (Bit 1)	Packet Type (IPv4 = 1b, IPv6 = 0b) Identifies what type of IP packet is used in the segmentation process. This is necessary for hardware to know where the IP Payload Length field is located. This does not override the checksum insertion bit, IXSM.
IP (Bit 1) **82544GC/EI** only	Packet Type (IP = 1b) Identifies the packet as an IP packet. The purpose of this bit is to enable/disable the updating of the IP header during the segmentation process. This does not override the checksum insertion bit, IXSM.
TCP (bit 0)	Packet Type (TCP = 1b) Identifies the packet as either TCP or UDP (non-TCP). This affects the processing of the header information.

Note:

1. The IDE, DEXT, and RS bits are valid regardless of the state of TSE. All other bits are ignored if TSE = 0b.
2. The TCP Segmentation feature also provides access to a generic block send function and may be useful for performing "segmentation offload" in which the header information is constant. By clearing both the TCP and IP bits, a block of data may be broken down into frames of a given size, a constant, arbitrary length header may be pre-pended to each frame, and two checksums optionally added.

表 3-15。命令字段 (TDESC.TUCMD) 布局

7	6	5	4	3	2	1	0
IDE	RSV	DEXT	RSV	RS	TSE	IP	TCP

TDESC.TUCMD	描述
IDE (位7)	中断延迟使能 IDE激活发送中断延迟定时器。硬件在写回具有RS位和IDE位设置的传输描述符时加载倒计时寄存器。加载的值来自中断延迟（TIDV）寄存器的IDV字段。当计数达到0时，发生传输中断。硬件在处理具有IDE设置的描述符时始终加载传输中断计数器，即使它由于先前的描述符已经在倒计时。如果硬件遇到具有RS设置但IDE未设置的描述符，它会在写回描述符后立即生成中断。中断延迟定时器被清除。
RSV (位6)	保留。设置为0b以实现未来兼容性。
DEXT(位5)	描述符扩展对于此描述符类型必须为 1b。
RSV (位4)	保留。设置为0b以实现未来兼容性。
RS (位3)	报告状态 RS 告知硬件报告此描述符的状态信息。由于此描述符不传输数据，因此状态字中的 DD 位有效。有关状态字段的布局，请参阅第 3.3.6.2 节。
TSE（位 2）	TCP 分段启用 TSE 指示此描述符正在设置 TCP 分段上下文。如果此位未设置，则针对正常（非"TCP 分段"）数据包的校验和卸载上下文将被写入。当处理此类类型的描述符时，以太网控制器会立即用描述符中的值更新相关上下文（TCP 分段或校验和卸载）。这意味着如果任何正常数据包或 TCP 分段数据包正在进行中（对于给定上下文尚未收到已设置 EOP 的描述符），结果可能不佳。
IP（位 1）	数据包类型 (IPv4 = 1b, IPv6 = 0b) 用于识别分段过程中使用的IP数据包类型。这是 必要的，以便硬件知道IP有效载荷长度字段的位置。这是 不会覆盖校验和插入位，IXSM。
IP (位1) 82544GC/EI仅限	数据包类型 (IP = 1b) 将数据包标识为IP数据包。此位的目的是启用/禁用 分段过程中IP头部的更新。这不会覆盖 校验和插入位，IXSM。
TCP (位0)	数据包类型 (TCP = 1b) 将数据包识别为TCP或UDP（非TCP）。这会影响头部信息的处理。

注意：

1. IDE、DEXT 和 RS 位无论 TSE 的状态如何都有效。如果 TSE = 0b，则忽略所有其他位。
2. TCP 分段功能还提供对通用块发送函数的访问，并且在进行"分段卸载"时可能很有用，其中头部信息是恒定的。通过清除 TCP 和 IP 位，可以将数据块分解为给定大小的帧，为每个帧预置一个恒定、任意长度的头部，并可选地添加两个校验和。

3.3.6.2 TCP/UDP Offload Transmit Descriptor Status Field

Four bits are reserved to provide transmit status, although only one is currently assigned for this specific descriptor type. The status word is only written back to host memory in cases where the RS is set in the command.

Table 3-16. Transmit Status Layout

3	2	1	0
RSV			DD

TDESC.STA	Description
RSV	Reserved Reserved for future use. Reads as 0b.
DD (bit 0)	Descriptor Done Indicates that the descriptor is finished and is written back after the descriptor has been processed.

3.3.7 TCP/IP Data Descriptor Format

The TCP/IP data descriptor is the companion to the TCP/IP context transmit descriptor described in the previous section. This descriptor type provides similar functionality to the legacy mode descriptor but also integrates the checksum offloading and TCP Segmentation feature.

To select this descriptor format, bit 29 in the command field (TDESC.DEXT) must be set to 1b and TDESC.DTYP must be set to 0001b. In this case, the descriptor format is defined as shown in Table 3-17.

3.3.6.2 TCP/UDP卸载传输描述符状态字段

有四位保留用于提供传输状态，尽管目前仅为此特定描述符类型分配了一位。状态字仅在命令中RS被设置的情况下才写回主机内存。

表3-16. 传输状态布局

3	2	1	0
RSV			DD

TDESC.STA	描述
RSV	保留 保留用于将来使用。读作0b。
DD (位0)	描述符完成 指示描述符已完成，并且在描述符处理后将写回。 已处理。

3.3.7 TCP/IP 数据描述符格式

TCP/IP 数据描述符是上一节中描述的 TCP/IP 上下文传输描述符的配套描述符。这种描述符类型提供了与传统模式描述符类似的功能，但也集成了校验和卸载和 TCP 分段功能。

要选择此描述符格式，命令字段（TDESC.DEXT）中的位 29 必须设置为 1b，且 TDESC.DTYP 必须设置为 0001b。在这种情况下，描述符格式定义如表 3-17 所示。

Table 3-17. Transmit Descriptor (TDESC) Layout – (Type = 0001b)

0	Address [63:0]						
8	Special	POPTS	RSV	STA	DCMD	DTYP	DTALEN
0	63	48 47	40 39	36 35	32 31	24 23	20 19 0

Transmit Descriptor	Description
Address	Data buffer address Address of the data buffer in the host memory which contains a portion of the transmit packet.
DTALEN	Data Length Field Total length of the data pointed to by this descriptor, in bytes. For data descriptors not associated with a TCP Segmentation operation (TDESC.TSE not set), the descriptor lengths are subject to the same restrictions specified for legacy descriptors (the sum of the lengths of the data descriptors comprising a single packet must be at least 80 bytes less than the allocated size of the transmit FIFO.)
DTYP	Data Type Set to 0001b to identify this descriptor as a TCP/IP data descriptor.
DCMD	Descriptor Command Field Provides options that control some of the generic descriptor processing features. Refer to Section 3.3.7.1 for bit definitions of the DCMD field.
STA	TCP/IP Status field Provides transmit status indication. Section 3.3.7.2 provides the bit definition for the TDESC.STA field.
RSV	Reserved Set to 0b for future compatibility.
POPTS	Packet Option Field Provides a number of options which control the handling of this packet. This field is ignored except on the first data descriptor of a packet. Section 3.3.7.3 provides the bit definition for the TDESC.POPTS field.
Special	Special field The Special field is used to provide 802.1q tagging information. This field is only valid in the last descriptor of the given packet (qualified by the EOP bit).

表 3-17. 传输描述符（TDESC）布局 – （类型 = 0001b）

0	地址 [63:0]						
8	特殊	POPTS	RSV	STA	DCMD	DTYP	DTALEN
0	63	48 47	40 39	36 35	32 31	24 23	20 19 0

传输描述符	描述
地址	数据缓冲区地址 主机内存中包含传输数据包一部分的数据缓冲区的地址。
DTALEN	数据长度字段 此描述符指向的数据的总长度，以字节为单位。 对于与TCP分段操作（TDESC.TSE未设置）无关的数据描述符，描述符长度受限于传统描述符指定的相同限制（构成单个数据包的数据描述符的总长度必须小于传输FIFO的分配大小80字节）。
DTYP	数据类型 设置为0001b以标识此描述符为TCP/IP数据描述符。
DCMD	描述符命令字段 提供控制某些通用描述符处理功能的选项。有关DCMD字段的位定义，请参阅第3.3.7.1节。
STA	TCP/IP状态字段 提供传输状态指示。 第3节3.3.7.2提供了TDESC.STA字段的位定义。
RSV	保留 设置为0b以保持未来兼容性。
POPTS	数据包选项字段 提供了一些选项，用于控制此数据包的处理。此字段仅在数据包的第一个数据描述符中有效。 第3.3.7.3节提供了TDESC.POPTS字段的位定义。
特殊	特殊字段 特殊字段用于提供802.1q标记信息。 此字段仅在给定数据包的最后一个描述符中有效（由EOP位限定）。

3.3.7.1 TCP/IP Data Descriptor Command Field

The Command field provides options that control checksum offloading and TCP segmentation features along with some of the generic descriptor processing features.

Table 3-18. Command Field (TDESC.DCMD) Layout

7	6	5	4	3	2	1	0
IDE	VLE	DEXT	RSV RPS[a]	RS	TSE	IFCS	EOP

a. **82544GC/EI** only.

TDESC.DCMD	Description
IDE (bit 7)	Interrupt Delay Enable When set, activates the transmit interrupt delay timer. Hardware loads a countdown register when it writes back a transmit descriptor that has RS and IDE set. The value loaded comes from the IDV field of the Interrupt Delay (TIDV) register. When the count reaches 0, a transmit interrupt occurs if enabled. Hardware always loads the transmit interrupt counter whenever it processes a descriptor with IDE set even if it is already counting down due to a previous descriptor. If hardware encounters a descriptor that has RS set, but not IDE, it generates an interrupt immediately after writing back the descriptor. The interrupt delay timer is cleared.
VLE (bit 6)	VLAN Enable When set, indicates that the packet is a VLAN packet and the hardware should add the VLAN Ethertype and an 802.1q VLAN tag to the packet. The Ethertype should come from the VET register and the VLAN data comes from the special field of the TX descriptor. The hardware in that case appends the FCS/CRC. Note that the CTRL.VME bit should also be set. If the CTRL.VME bit is not set, the Ethernet controller does not insert VLAN tags on outgoing packets and it sends generic Ethernet packets. The IFCS controls the insertion of the FCS/CRC in that case. VLE is only valid in the last descriptor of the given packet (qualified by the EOP bit).
DEXT (Bit 5)	Descriptor Extension Must be 1b for this descriptor type
RPS RSV (bit 4)	Report Packet Sent RPS is used in cases where software must know that a packet has been sent on the wire, not just that it has been loaded into the **82544GC/EI** controller's internal packet buffer. When set, hardware defers writing the DD bit in the status byte until the packet has been sent, or transmission results in an error such as excess collisions. Hardware can continue to pre-fetch data from descriptors logically after the one with RPS set, but does not advance the head pointer or write back any other descriptors until it has sent the packet with RPS set. For a TCP Segmentation context, the RPS bit indicates to the **82544GC/EI** that the descriptor status should only be written back once all packets that make up the given TCP Segmentation context had been sent. This bit is reserved and should be programmed to 0b for all Ethernet controllers except the **82544GC/EI**.
RS (bit 3)	Report Status When set, tells the hardware to report the status information for this descriptor as soon as the corresponding data buffer has been fetched and stored in the controller's internal packet buffer.

3.3.7.1 TCP/IP数据描述符命令字段

命令字段提供选项，用于控制校验和卸载和TCP分段功能，以及通用描述符处理功能中的一些功能。

表3-18。命令字段（TDESC.DCMD）布局

7	6	5	4	3	2	1	0
IDE	VLE	DEXT	RSV RPS[a]	RS	TSE	IFCS	EOP

a. 82544GC/EI 仅限。

TDESC.DCMD	描述
IDE (bit 7)	中断延迟使能 当设置时，激活传输中断延迟定时器。硬件在写回具有RS和IDE设置的传输描述符时，会加载一个倒计时寄存器。加载的值来自中断延迟（TIDV）寄存器的IDV字段。当计数器达到0时，如果已使能，则发生传输中断。硬件在处理具有IDE设置的描述符时，无论是否由于先前的描述符正在倒计时，都会始终加载传输中断计数器。如果硬件遇到具有RS设置但IDE未设置的描述符，它会在写回描述符后立即生成中断。中断延迟定时器被清除。
VLE (bit 6)	VLAN使能 当设置时，指示该数据包是VLAN数据包，硬件应向数据包中添加VLAN以太类型和802.1q VLAN标签。以太类型应来自VET寄存器，VLAN数据来自TX描述符的特殊字段。在这种情况下，硬件会附加FCS/CRC。 注意CTRL.VME位也应设置。如果CTRL.VME位未设置，以太网控制器不会在出站数据包中插入VLAN标签，并发送通用以太网数据包。IFCS控制该情况下FCS/CRC的插入。 VLE仅在给定数据包的最后一个描述符中有效（由EOP位限定）。
DEXT (位5)	描述符扩展 Must be 1b for this descriptor type
RPS RSV (位4)	报告包发送 RPS 用于软件必须知道数据包已在网络上发送的情况，而不仅仅是加载到 82544GC/EI 控制器的内部数据包缓冲区的情况。 当设置时，硬件将状态字节中的 DD 位延迟写入，直到数据包已发送，或传输结果导致错误（如过度冲突）。硬件可以继续从具有 RPS 设置的逻辑描述符之后预取数据，但直到它发送了具有 RPS 设置的数据包之前，不会前进头指针或写回任何其他描述符。 对于 TCP 分段上下文，RPS 位指示 82544GC/EI 应该仅当组成给定 TCP 分段上下文的所有数据包都已发送后，才写回描述符状态。 此位为保留位，除82544GC/EI以外的所有以太网控制器应编程为0b。
RS（位3）	报告状态 当设置时，指示硬件在相应的数据缓冲区被获取并存储在控制器内部的数据包缓冲区后，立即报告此描述符的状态信息。

TDESC.DCMD	Description
TSE (bit 2)	TCP Segmentation Enable TSE indicates that this descriptor is part of the current TCP Segmentation context. If this bit is not set, the descriptor is part of the "normal" context.
IFCS (Bit 1)	Insert IFCS Controls the insertion of the FCS/CRC field in normal Ethernet packets. IFCS is only valid in the last descriptor of the given packet (qualified by the EOP bit).
EOP (Bit 0)	End Of Packet The EOP bit indicates that the buffer associated with this descriptor contains the last data for the packet or for the given TCP Segmentation context. In the case of a TCP Segmentation context, the DTALEN length of this descriptor should match the amount remaining of the original PAYLEN. If it does not, the TCP Segmentation context is terminated but the end of packet processing may be incorrectly performed. These abnormal termination events are counted in the TSCTFC statistics register.

Note: The VLE, IFCS, and VLAN fields are only valid in certain descriptors. If TSE is enabled, the VLE, IFCS, and VLAN fields are only valid in the first data descriptor of the TCP segmentation context. If TSE is not enabled, then these fields are only valid in the last descriptor of the given packet (qualified by the EOP bit).

3.3.7.2 TCP/IP Data Descriptor Status Field

Four bits are reserved to provide transmit status, although only the DD is valid[1]. The status word is only written back to host memory in cases where the RS bit is set in the command field. The DD bit indicates that the descriptor is finished and is written back after the descriptor has been processed.

Table 3-19. Transmit Status Layout

3	2	1	0
RSV TU[a]	LC	EC	DD

a. **82544GC/EI** only.

TDESC.STA	Description
Reserved	Reserved

1. Unless the RPS bit is set in the descriptor (**82544GC/EI** only).

TDESC.DCMD	描述
TSE (bit 2)	TCP分段启用TSE指示该描述符是当前TCP分段上下文的一部分。如果此位未设置，则该描述符是"正常"上下文的一部分。
IFCS（位1）	插入IFCS 控制FCS/CRC字段在普通以太网数据包中的插入。 IFCS仅在给定数据包的最后一个描述符中有效（由EOP位限定）。
EOP（位0）	数据包结束 EOP位指示与该描述符关联的缓冲区包含数据包的最后一个数据或给定TCP分段上下文的数据。在TCP分段上下文中，该描述符的DTALEN长度应与原始PAYLEN剩余量匹配。如果不匹配，TCP分段上下文将被终止，但数据包结束处理可能被错误执行。这些异常终止事件计入TSCTFC统计寄存器。

注意：VLE、IFCS 和 VLAN 字段仅在特定描述符中有效。如果 TSE 被启用，VLE、IFCS 和 VLAN 字段仅在 TCP 分段上下文中的第一个数据描述符中有效。如果 TSE 没有被启用，那么这些字段仅在给定数据包的最后一个描述符中有效（由 EOP 位限定）。

3.3.7.2 TCP/IP 数据描述符状态字段

保留四位以提供传输状态，尽管只有 DD 有效[1]。状态字仅在命令字段中的 RS 位被设置的情况下才写回主机内存。DD 位指示描述符已完成，并且在描述符被处理后写回。

表 3-19。传输状态布局

3	2	1	0
RSV TU[a]	LC	EC	DD

a. 仅82544GC/EI。

TDESC.STA	描述
保留	保留

1. 除非在描述符中设置了RPS位（仅限82544GC/EI）。

TDESC.STA	Description
LC (bit2)	Late Collision Indicates that late collision occurred while working in half-duplex mode. It has no meaning while working in full-duplex mode. Note that the collision window is speed dependent: 64 bytes for 10/100 Mb/s and 512 bytes for 1000 Mb/s operation.
EC (bit 1)	Excess Collision Indicates that the packet has experienced more than the maximum excessive collisions as defined by TCTL.CT control field and was not transmitted. Is has no meaning while working in full-duplex mode.
DD (bit 0)	Descriptor Done Indicates that the descriptor is done and is written back either after the descriptor has been processed (with RS set), or for the **82554GC/EI** only, after the packet has been transmitted on the wire (with RPS set).

3.3.7.3 TCP/IP Data Descriptor Option Field

The POPTS field provides a number of options which control the handling of this packet. This field is ignored except on the first data descriptor of a packet.

Table 3-20. Packet Options Field (TDESC.POPTS) Layout

7	6	5	4	3	2	1	0
RSV	RSV	RSV	RSV	RSV	RSV	TXSM	IXSM

TDESC.POPTS	Description
RSV (bit 2-7)	Reserved Should be written with 0b for future compatibility.
TXSM (bit1)	Insert TCP/UDP Checksum Controls the insertion of the TCP/UDP checksum. If not set, the value placed into the checksum field of the packet data is not modified, and is placed on the wire. When set, TCP/UDP checksum field is modified by the hardware. Valid only in the first data descriptor for a given packet or TCP Segmentation context.
IXSM (bit 0)	Insert IP Checksum Controls the insertion of the IP checksum. If not set, the value placed into the checksum field of the packet data is not modified and is placed on the wire. When set, the IP checksum field is modified by the hardware. Valid only in the first data descriptor for a given packet or TCP Segmentation context.

3.3.7.4 TCP/IP Data Descriptor Special Field

The SPECIAL field is used to provide the 802.1q/802.3ac tagging information.

TDESC.STA	描述
LC（位2）	晚期冲突 指示在半双工模式下发生了晚期冲突。 在全双工模式下无意义。注意：冲突窗口与速度相关：10/100 Mb/s为64字节，1000 Mb/s操作为512字节。
EC（位1）	多余冲突 指示数据包经历了超过最大额外 由TCTL.CT控制字段定义的冲突，并且未被传输。 在全双工模式下，此设置无意义。
DD (位0)	描述符完成 指示描述符已完成，并且已写回，无论是描述符处理完成（设置RS）之后，还是对于82554GC/EI而言，在数据包在网络上传输完成（设置RPS）之后。

3.3.7.3 TCP/IP数据描述符选项字段

POPTS字段提供了一些选项，用于控制此数据包的处理。此字段仅在数据包的第一个数据描述符中被忽略。

表3-20。数据包选项字段（TDESC.POPTS）布局

7	6	5	4	3	2	1	0
RSV	RSV	RSV	RSV	RSV	RSV	TXSM	IXSM

TDESC.POPTS	描述
RSV (bit 2-7)	保留 应使用 0b 以确保未来兼容性。
TXSM (位1)	插入TCP/UDP校验和 控制TCP/UDP校验和的插入。 若未设置，则数据包数据校验和字段中的值不会被修改，并直接传输。当设置时，硬件会修改TCP/UDP校验和字段。 仅适用于给定数据包或TCP分段上下文中的第一个数据描述符。
IXSM (位0)	插入IP校验和 控制IP校验和的插入。 若未设置，则数据包数据校验和字段中的值不会被修改，并直接传输。当设置时，硬件会修改IP校验和字段。 仅适用于给定数据包或TCP分段上下文中的第一个数据描述符。

3.3.7.4 TCP/IP数据描述符特殊字段

特殊字段用于提供 802.1q/802.3ac 标记信息。

When CTRL.VME is set to 1b, all packets transmitted from the Ethernet controller that has VLE set in the DCMD field is sent with an 802.1Q header added to the packet. The contents of the header come from the transmit descriptor special field and from the VLAN type register. The special field is ignored if the VLE bit in the transmit descriptor command field is 0b. The special field is valid only when EOP is set.

Table 3-21. Special Field (TDESC.SPECIAL) Layout

15 13	12	11 0
PRI	CFI	VLAN

TDESC.SPECIAL	Description
PRI	User Priority Three bits that provide the VLAN user priority field to be inserted in the 802.1Q tag.
CFI	Canonical Form Indicator
VLAN	VLAN Identifier 12 bits that provide the VLAN identifier field to be inserted in the 802.1Q tag.

3.4 Transmit Descriptor Ring Structure

The transmit descriptor ring structure is shown in Figure 3-4. A pair of hardware registers maintains the transmit queue. New descriptors are added to the ring by writing descriptors into the circular buffer memory region and moving the ring's tail pointer. The tail pointer points one entry beyond the last hardware owned descriptor (but at a point still within the descriptor ring). 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.

当 CTRL.VME 设置为 1b 时，所有从以太网控制器传输且 DCMD 字段中设置了 VLE 的数据包都会在数据包中添加 802.1Q 标题。标题的内容来自传输描述符特殊字段和 VLAN 类型寄存器。如果传输描述符命令字段中的 VLE 位为 0b，则忽略特殊字段。特殊字段仅在 EOP 设置时有效。

表3-21。特殊字段（TDESC.SPECIAL）布局

15 13	12	11 0
PRI	CFI	VLAN

TDESC.SPECIAL	描述
PRI	用户优先级 三个比特位，用于在802.1Q标签中插入VLAN用户优先级字段。
CFI	规范形式指示符
VLAN	VLAN标识符 提供802.1Q标签中要插入的VLAN标识符字段的12位。

3.4 传输描述符环结构

传输描述符环结构如图3-4所示。一对硬件寄存器维护传输队列。通过将描述符写入循环缓冲区内存区域并将环的尾指针移动，将新的描述符添加到环中。尾指针指向最后一个硬件拥有的描述符之后的一个条目（但仍在描述符环内）。传输继续，直到头等于尾的描述符处，此时队列为空。

传递给硬件的描述符在头指针移动到它们之前不应被软件操作。

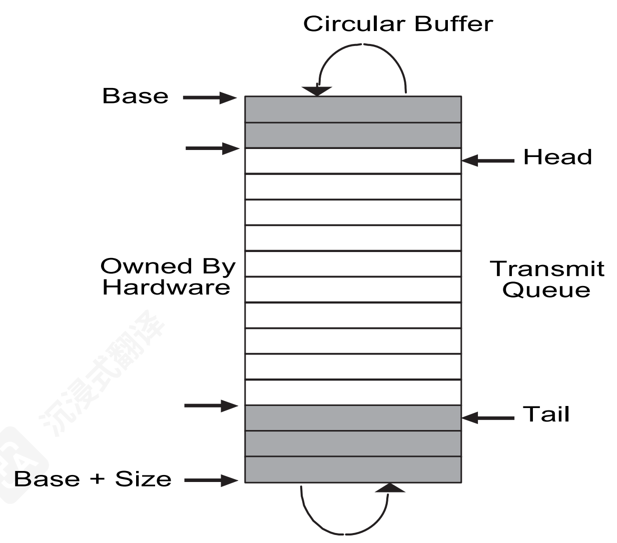


Figure 3-4. Transmit Descriptor Ring Structure

Shaded boxes in Figure 3-4 represent descriptors that have been transmitted but not yet reclaimed by software. Reclaiming involves freeing up buffers associated with the descriptors.

The transmit descriptor ring is described by the following registers:

- Transmit Descriptor Base Address registers (TDBAL and TDBAH)

  These registers indicate 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. TDBAL contains the lower 32-bits; TDBAH contains the upper 32 bits. Hardware ignores the lower 4 bits in TDBAL.
- Transmit Descriptor Length register (TDLEN)

  This register determines the number of bytes allocated to the circular buffer. This value must be 128 byte aligned.
- Transmit Descriptor Head register (TDH)

  This register holds a value which is an offset from the base, and indicates the in–progress descriptor. There can be up to 64K descriptors in the circular buffer. Reading this register returns the value of “head” corresponding to descriptors already loaded in the output FIFO.
- Transmit Descriptor Tail register (TDT)

  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 base register indicates the start of the circular descriptor queue and the length register indicates the maximum size of the descriptor ring. The lower seven bits of length are hard–wired to 0b. Byte addresses within the descriptor buffer are computed as follows:

*address = base + (ptr * 16)*, where *ptr* is the value in the hardware head or tail register.

The size chosen for the head and tail registers permit a maximum of 64 K descriptors, or approximately 16 K packets for the transmit queue given an average of four descriptors per packet.

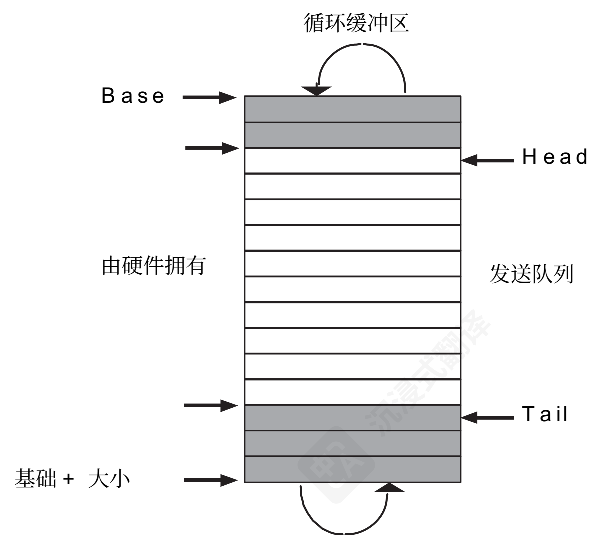


图3-4。传输描述符环结构

图3-4中的阴影框表示已被传输但尚未被软件回收的描述符。回收涉及释放与描述符关联的缓冲区。

传输描述符环由以下寄存器描述：

- 传输描述符基地址寄存器（TDBAL和TDBAH）

  这些寄存器指示描述符环缓冲区的起始位置。这个64位地址是按对齐的
  一个16字节的边界，并存储在两个连续的32位寄存器中。TDBAL包含低32位；
  TDBAH包含高32位。硬件忽略TDBAL中的低4位。
- 传输描述符长度寄存器 (TDLEN)

  此寄存器确定分配给循环缓冲区的字节数。此值必须对齐128字节。
- 传输描述符头寄存器 (TDH)

  此寄存器存储一个相对于基础的偏移量，并指示正在处理的描述符。循环缓冲区中最多可以有64K个描述符。读取此寄存器将返回输出FIFO中已加载的描述符对应的“头部”值。
- 传输描述符尾寄存器 (TDT)

  此寄存器存储一个相对于基础的偏移量，并指示硬件能够处理的最后一个描述符之后的位置。这是软件写入第一个新描述符的位置。

基础寄存器指示循环描述符队列的起始位置，长度寄存器指示描述符环的最大大小。长度的低七位硬接为0b。描述符缓冲区内的字节地址计算方法如下：

地址 = 基础 + (ptr * 16)，其中ptr是硬件头部或尾部寄存器的值。

头部和尾部寄存器选择的大小允许最多64 K个描述符，或给定每个数据包平均四个描述符的情况下，传输队列大约为16 K个数据包。

Once activated, hardware fetches the descriptor indicated by the hardware head register. The hardware tail register points one beyond the last valid descriptor.

Software can determine if a packet has been sent by setting the RS bit (or the RPS bit for the **82544GC/EI** only) in the transmit descriptor command field. Checking the transmit descriptor DD bit in memory eliminates a potential race condition. All descriptor data is written to the IO bus prior to incrementing the head register, but a read of the head register could “pass” the data write in systems performing IO write buffering. Updates to transmit descriptors use the same IO write path and follow all data writes. Consequently, they are not subject to the race condition. Other potential conditions also prohibit software reading the head pointer.

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[1].

The process of checking for completed packets consists of one of the following:

- Scan memory for descriptor status write-backs.
- Take an interrupt. An interrupt condition can be generated whenever a transmit queue goes empty (ICR.TXQE). Interrupts can also be triggered in other ways.

3.4.1 Transmit Descriptor Fetching

The descriptor processing strategy for transmit descriptors is essentially the same as for receive descriptors except that a different set of thresholds are used. As for receives, the number of on-chip transmit descriptors buffer space is 64 descriptors.

When the on-chip buffer is empty, a fetch happens as soon as any descriptors are made available (software writes to the tail pointer). When the on-chip buffer is nearly empty (TXDCTL.PTHRESH), a prefetch is performed whenever enough valid descriptors (TXDCTL.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).

The descriptor prefetch policy is aggressive to maximize performance. If descriptors reside in an external cache, the system must ensure cache coherency before changing the tail pointer.

When the number of descriptors in host memory is greater than the available on-chip descriptor storage, the chip may elect to perform a fetch which is not a multiple of cache line size. The hardware performs this non-aligned fetch if doing so results in the next descriptor fetch being aligned on a cache line boundary. This allows the descriptor fetch mechanism to be most efficient in the cases where it has fallen behind software.

3.4.2 Transmit Descriptor Write-back

The descriptor write-back policy for transmit descriptors is similar to that for receive descriptors with a few additional factors. First, since transmit descriptor write-backs are optional (controlled by RS[2] in the transmit descriptor), only descriptors which have one (or both) of these bits set starts the accumulation of write-back descriptors. Secondly, to preserve backward compatibility with the 82542, if the TXDCTL.WTHRESH value is 0b, the Ethernet controller writes back a single byte of the descriptor (TDESCR.STA) and all other bytes of the descriptor are left unchanged.

1. With the RPS bit set, the head is not advanced until after the packet is transmitted or rejected due to excess collisions (**82544GC/EI** only).
2. And RPS for the **82544GC/EI** only.

一旦激活，硬件获取硬件头寄存器指示的描述符。硬件尾寄存器指向最后一个有效描述符之后的位置。

软件可以通过在传输描述符命令字段中设置RS位（对于82544GC/EI，仅设置RPS位）来确定数据包是否已发送。在内存中检查传输描述符DD位可以消除潜在的竞争条件。所有描述符数据在增加头寄存器之前都写入IO总线，但在执行IO写入缓冲的系统中，读取头寄存器可能会“跳过”数据写入。传输描述符的更新使用相同的IO写入路径，并遵循所有数据写入。因此，它们不受竞争条件的影响。其他潜在条件也禁止软件读取头指针。

通常情况下，硬件在传输之前预取数据包数据。硬件通常在将数据存储到发送FIFO[1]后更新头指针的值。

检查已完成数据包的过程包括以下一项:

- 扫描内存中的描述符状态写回。
- 触发中断。当传输队列变为空 (ICR.TXQE) 时，可能会生成中断条件。中断也可以通过其他方式触发。

3.4.1 传输描述符获取

3.4.1 传输描述符获取

当片上缓冲区为空时，一旦有描述符可用（软件写入尾指针），就会立即发生获取。当片上缓冲区几乎为空（TXDCTL.PTHRESH）时，只要主机内存中有足够的有效描述符（TXDCTL.HTHRESH）且没有更高优先级的其他DMA活动（描述符获取和写回或数据包数据传输），就会执行预取。

当片上缓冲区为空时，一旦有描述符可用（软件写入尾指针），就会立即发生获取。当片上缓冲区几乎为空（TXDCTL.PTHRESH）时，只要主机内存中有足够的有效描述符（TXDCTL.HTHRESH）且没有更高优先级的其他DMA活动（描述符获取和写回或数据包数据传输），就会执行预取。

当片上缓冲区为空时，一旦有描述符可用（软件写入尾指针），就会立即发生获取。当片上缓冲区几乎为空（TXDCTL.PTHRESH）时，只要主机内存中有足够的有效描述符（TXDCTL.HTHRESH）且没有更高优先级的其他DMA活动（描述符获取和写回或数据包数据传输），就会执行预取。

描述符预取策略是激进的，以最大化性能。如果描述符驻留在外部缓存中，系统必须在更改尾指针之前确保缓存一致性。

当主机内存中的描述符数量大于可用的片上描述符存储时，芯片可能会选择执行一个不是缓存行大小倍数的获取。如果这样做会导致下一个描述符获取在缓存行边界上对齐，硬件会执行这种非对齐获取。这允许描述符获取机制在它落后于软件的情况下最有效率。

3.4.2 传输描述符写回

传输描述符的描述符回写策略与接收描述符相似，但有一些额外的因素。首先，由于传输描述符回写是可选的（由传输描述符中的 RS[2] 控制），只有那些具有其中一个（或两个）位被设置的描述符才会开始回写描述符的累积。其次，为了与 82542 保持向后兼容性，如果 TXDCTL.WTHRESH 值为 0b，以太网控制器会回写描述符（TDESCR.STA）的一个字节，而描述符的其他字节保持不变。

1. 设置RPS位后，头部不会前进，直到数据包被传输或由于多余冲突（仅限82544GC/EI）而被拒绝。 2. 仅限82544GC/EI的RPS。

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

Descriptors are written back in one of three conditions:

- TXDCTL.WTHRESH = 0b and a descriptor which has RS[1] set is ready to be written back
- Transmit Interrupt Delay timer expires
- TXDCTL.WTHRESH > 0b and TXDCTL.WTHRESH descriptors have accumulated

For the first condition, write-backs are immediate. This is the default operation and is backward compatible. For this case, the Transmit Interrupt delay function works as described in Section 3.4.3.1.

The other two conditions are only valid if descriptor bursting is enabled (see Section 13.4.44). In the second condition, the Transmit Interrupt Delay timer (TIDV) 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 final condition, if TXDCTL.WTHRESH descriptors are ready for write-back, the write-back is performed.

3.4.3 Transmit Interrupts

Hardware supplies three transmit interrupts. These interrupts are initiated through the following conditions:

- Transmit queue empty (TXQE) — All descriptors have been processed. The head pointer is equal to the tail pointer.
- Descriptor done [Transmit Descriptor Write-back (TXDW)] — Set when hardware writes back a descriptor with RS[1] set. This is only expected to be used in cases where, for example, the streams interface has run out of descriptors and wants to be interrupted whenever progress is made.
- Transmit Delayed Interrupt (TXDW) — In conjunction with IDE (Interrupt Delay Enable), the TXDW indication is delayed by a specific time per the TIDV register. This interrupt is set when the transmit interrupt countdown register expires. The countdown register is loaded with the value of the IDV field of the TIDV register, when a transmit descriptor with its RS[1] bit and the IDE bit are set, is written back. When a Transmit Delayed Interrupt occurs, the TXDW interrupt cause bit is set (just as when a Transmit Descriptor Write-back interrupt occurs). This interrupt may be masked in the same manner as the TXDW interrupt. This interrupt is used frequently by software that performs dynamic transmit chaining, by adding packets one at a time to the transmit chain.

Note: The transmit delay interrupt is indicated with the same interrupt bit as the transmit write-back interrupt, TXDW. The transmit delay interrupt is only delayed in time as discussed above.

1. Or RPS for the **82544GC/EI** only.

由于延迟然后突发传输描述符回写的收益最小，阈值很可能保持默认值（0b），以强制立即回写传输描述符并保持向后兼容性。

描述符在以下三种情况下被写回：

- TXDCTL.WTHRESH = 0b 并且一个设置了RS1的描述符准备好被写回
- 传输中断延迟定时器到期
- TXDCTL.WTHRESH > 0b 并且TXDCTL.WTHRESH描述符已经积累

对于第一种情况，写回是立即的。这是默认操作，并且向后兼容。对于这种情况，传输中断延迟功能按第3.4.3.1节中描述的方式工作。

其他两种情况只有在启用描述符突发的情况下才有效（参见第13.4.44节）。在第二种情况下，传输中断延迟定时器（TIDV）用于强制及时写回描述符。定时器初始化后的第一个数据包启动定时器。定时器超时会刷新任何累积的描述符并设置一个中断事件（TXDW）。

对于最后一种情况，如果TXDCTL.WTHRESH描述符准备好写回，则执行写回。

3.4.3 传输中断

硬件提供三种传输中断。这些中断通过以下条件触发：

- 传输队列空（TXQE）— 所有描述符已处理。头指针等于尾指针。
- 描述符完成 [传输描述符写回 (TXDW)] — 当硬件写回带有RS1设置的描述符时设置。这仅预期在例如流接口用完描述符并希望在每次进度时被中断的情况下使用。
  流接口用完描述符并希望在每次进度时被中断。
- 传输延迟中断 (TXDW) — 与IDE（中断延迟使能）结合，TXDW指示会根据TIDV寄存器延迟特定时间。当传输中断倒计时寄存器到期时设置此中断。当带有其RS1位和IDE位的传输描述符被写回时，倒计时寄存器被加载为TIDV寄存器IDV字段的值。当发生传输延迟中断时，TXDW中断原因位被设置（就像发生传输描述符写回中断时一样）。此中断可以像TXDW中断一样被屏蔽。此中断经常被执行动态传输链的软件使用，通过一次添加一个数据包到传输链中。

注意：发送延迟中断与发送回写中断 TXDW 使用相同的中断位指示。发送延迟中断仅在时间上延迟，如上所述。

1. 或仅适用于 82544GC/EI 的 RPS。

- Link status change (LSC) - Set when the link status changes. When using the internal PHY, link status changes are determined and indicated by the PHY via a change in its LINK indication.

  When using an external TBI device (**82544GC/EI** only), the device might indicate a link status change using its LOS (loss of sync) indication. In this TBI mode, if HW Auto-Negotiation is enabled, the MAC can also detect and signal a link status change if the Configuration Base Page register is received (0b), or if either the LRST or ANE bits are changed by software.
- Transmit Descriptor Ring Low Threshold Hit (TXD_LOW) (not applicable to the **82544GC/EI**) - Set when the total number of transmit descriptors available (as measured by the difference between the Tx descriptor ring Head and Tail pointer) hits the low threshold specified in the TXDCTL.LWTHRESH field.

3.4.3.1 Delayed Transmit Interrupts

This mechanism allows software the flexibility of delaying transmit interrupts until no more descriptors are added to a transmit chain for a certain amount of time, rather than when the Ethernet controller's head pointer catches the tail pointer. This occurs if the Ethernet controller is processing packets slightly faster than the software, a likely scenario for gigabit operations.

A software driver usually has no knowledge of when it is going to be asked to send another frame. For performance reasons, it is best to generate only one transmit interrupt after a burst of packets have been sent.

Refer to Section 3.3.3.1 for specific details.

3.5 TCP Segmentation

Hardware TCP Segmentation is one of the off-loading options of most modern TCP/IP stacks. This is often referred to as "Large Send" offloading. This feature enables the TCP/IP stack to pass to the Ethernet controller software driver a message to be transmitted that is bigger than the Maximum Transmission Unit (MTU) of the medium. It is then the responsibility of the software driver and hardware to carve 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 is constant for all packets associated with the TCP message.

The offloading of these processes from the software driver to the Ethernet controller saves significant CPU cycles. The software driver shares the additional tasks to support these options with the Ethernet controller.

Although the Ethernet controller's TCP segmentation offload implementation was specifically designed to take advantage of new "TCP Segmentation offload" features, the hardware implementation was made generic enough so that it could also be used to "segment" traffic from other protocols. For instance this feature could be used any time it is desirable for hardware to segment a large block of data for transmission into multiple packets that contain the same generic header.

- 链路状态改变（LSC）- 当链路状态改变时设置。在使用内部PHY时，链路状态改变由PHY通过其LINK指示的变化来确定和指示。

  当使用外部 TBI 设备（仅限 82544GC/EI）时，设备可能使用其 LOS（丢失同步）指示来指示链路状态变化。在此 TBI 模式下，如果启用了硬件自动协商，MAC 还可以检测并信号链路状态变化，如果接收到配置基页寄存器（0b），或者 LRST 或 ANE 位中任一位由软件改变。
- 传输描述符环低阈值命中 (TXD_LOW)（不适用于82544GC/ EI）- 当可用的传输描述符总数（通过Tx描述符环头指针和尾指针之间的差值测量）达到TXDCTL.LWTHRESH字段中指定的低阈值时设置。

3.4.3.1 延迟传输中断

这种机制允许软件在一段时间内延迟传输中断，直到没有更多描述符被添加到传输链中，而不是当以太网控制器头指针捕获尾指针时。如果以太网控制器处理数据包的速度略快于软件，这种情况就会发生，这在千兆操作中很常见。

软件驱动程序通常不知道何时会被要求发送另一个帧。出于性能原因，最好在发送数据包突发后只生成一个传输中断。

参考第3.3.3.1节以获取具体细节。

3.5 TCP分段

硬件TCP分段是大多数现代TCP/IP栈卸载选项之一。这通常被称为“大发送”卸载。此功能使TCP/IP栈能够将一个大于介质最大传输单元（MTU）的传输消息传递给以太网控制器软件驱动程序。然后，软件驱动程序和硬件的责任是将TCP消息分割成具有适当层2（以太网）、3（IP）和4（TCP）头部的MTU大小帧。这些头部必须包括序列号、校验和字段、选项和标志值。请注意，这些值中的某些值（如校验和值）对TCP消息的每个数据包都是唯一的，而其他字段（如源IP地址）对所有与TCP消息相关的数据包都是恒定的。

将这些进程从软件驱动程序卸载到以太网控制器可以节省大量的CPU周期。软件驱动程序将与以太网控制器共享额外的任务以支持这些选项。

虽然以太网控制器的TCP分段卸载实现是专门设计来利用新的“TCP分段卸载”功能，但硬件实现足够通用，以至于它也可以用来“分段”其他协议的流量。例如，当硬件需要将大块数据分段传输为包含相同通用头部多个数据包时，此功能都可以使用。

3.5.1 Assumptions

The following assumption applies to the TCP Segmentation implementation in the Ethernet controller:

- The RS bit operation is not changed. Interrupts are set after data in buffers pointed to by individual descriptors is transferred to hardware.
- Checksums are not accurate above a 12 K frame size.
- The function of the RPS[1] bit in the Transmit Descriptor is applicable to all of the packets that make up the "TCP Segmentation" context, not the individual packets segmented by hardware.

3.5.2 Transmission Process

The transmission process for regular (non-TCP Segmentation packets) involves:

- The protocol stack receives from an application a block of data that is to be transmitted.
- The protocol stack calculates the number of packets required to transmit this block based on the MTU size of the media and required packet headers.
- For each packet of the data block:
- Ethernet, IP and TCP/UDP headers are prepared by the stack.
- The stack interfaces with the software device driver and commands the driver to send the individual packet.
- The driver gets the frame and interfaces with the hardware.
- The hardware reads the packet from host memory (via DMA transfers).
- The driver returns ownership of the packet to the operating system when the hardware has completed the DMA transfer of the frame (indicated by an interrupt).

The transmission process for the Ethernet controller TCP segmentation offload implementation involves:

- The protocol stack receives from an application a block of data that is to be transmitted.
- The stack interfaces to the software device driver and passes the block down with the appropriate header information.
- The software device driver sets up the interface to the hardware (via descriptors) for the TCP Segmentation context.
- The hardware transfers the packet data and performs the Ethernet packet segmentation and transmission based on offset and payload length parameters in the TCP/IP context descriptor including:
  — Packet encapsulation
  — Header generation & field updates including IP and TCP/UDP checksum generation
  — The driver returns ownership of the block of data to the operating system when the hardware has completed the DMA transfer of the entire data block (indicated by an interrupt).

1. **82544GC/EI** only.

3.5.1 假设

以下假设适用于以太网控制器中的TCP分段实现：

- RS位操作未更改。中断是在指向各个描述符的缓冲区中的数据传输到硬件后设置的。
- 校验和在12 K帧大小以上的情况下不准确。
- 传输描述符中的RPS1位的函数适用于构成“TCP分段”上下文的所有数据包，而不是硬件分段的单个数据包。

3.5.2 传输过程

常规（非TCP分段数据包）的传输过程包括：

- 协议栈从应用程序接收一个要传输的数据块。
- 协议栈根据媒体MTU大小和所需的包头计算传输此数据块所需的数据包数量。
- 对于数据块中的每个数据包：
- 以太网、IP 和 TCP/UDP 头部由栈准备。
- 栈与软件设备驱动程序接口，并命令驱动程序发送单个数据包。
- 驱动程序获取帧并与硬件接口。
- 硬件从主机内存中读取数据包（通过 DMA 传输）。
- 当硬件需要时，驱动程序将数据包的所有权返回给操作系统完成了帧的DMA传输（通过中断指示）。

以太网控制器TCP分段卸载实现的传输过程涉及：

协议栈从应用程序接收一个要传输的数据块。
- 栈接口与软件设备驱动程序相连，并附带适当的头部信息向下传递数据块。
- 软件设备驱动程序为TCP分段上下文设置与硬件（通过描述符）的接口。
- 硬件根据TCP/IP上下文描述符中的偏移量和有效载荷长度参数，传输数据包数据，并执行以太网数据包分段和传输。
  — 数据封装
  — 头部生成 & 字段更新包括IP和TCP/UDP校验和生成
  — 当硬件完成整个数据块的DMA传输（通过中断指示）时，驱动程序将数据块的所有权返回给操作系统。

1. 82544GC/EI仅适用。

3.5.2.1 TCP Segmentation Data Fetch Control

To perform TCP Segmentation in the Ethernet controller, the DMA unit must ensure that the entire payload of the segmented packet fits into the available space in the on-chip Packet Buffer. The segmentation process is performed without interruption. The DMA performs various comparisons between the payload and the Packet Buffer to ensure that no interruptions occur. The TCP Segmentation Pad & Minimum Threshold (TSPMT) register is used to allow software to program the minimum threshold required for a TCP Segmentation payload. Consideration should be made for the MTU value when writing this field. The TSPMT register is also used to program the threshold padding overhead. This padding is necessary due to the indeterminate nature of the MTU and the associated headers.

3.5.3 TCP Segmentation Performance

Performance improvements for a hardware implementation of TCP Segmentation offload mean:

- The operating system stack does not need to partition the block to fit the MTU size, saving CPU cycles.
- The operating system stack only computes one Ethernet, IP, and TCP header per segment, saving CPU cycles.
- The operating system stack interfaces with the software device driver only once per block transfer, instead of once per frame.
- Larger PCI bursts are used which improves bus efficiency.
- Interrupts are easily reduced to one per TCP message instead of one per packet.
- Fewer I/O accesses are required to command the hardware.

3.5.4 Packet Format

Typical TCP/IP transmit window size is 8760 bytes (about 6 full size frames). A TCP message can be as large as 64 KB and is generally fragmented across multiple pages in host memory. The Ethernet controller partitions the data packet into standard Ethernet frames prior to transmission. The Ethernet controller supports calculating the Ethernet, IP, TCP, and even UDP headers, including checksum, on a frame by frame basis.

Ethernet	IPv4	TCP/UDP	DATA	FCS

Figure 3-5. TCP/IP Packet Format

Frame formats supported by the Ethernet controller’s TCP segmentation include:

- Ethernet 802.3
- IEEE 802.1q VLAN (Ethernet 802.3ac)
- Ethernet Type 2
- Ethernet SNAP
- IPv4 headers with options
- IPv6 headers with IP option next headers
- IPv6 packet tunneled in IPv4

3.5.2.1 TCP分段数据获取控制

要在以太网控制器中执行TCP分段，DMA单元必须确保分段数据包的有效载荷完全适合片上数据包缓冲区中的可用空间。分段过程在不中断的情况下执行。DMA在有效载荷和数据包缓冲区之间进行各种比较，以确保不会发生中断。TCP分段填充和最小阈值（TSPMT）寄存器用于允许软件编程TCP分段有效载荷所需的最小阈值。在写入此字段时应考虑MTU值。TSPMT寄存器也用于编程阈值填充开销。由于MTU的不确定性以及相关的头部，这种填充是必要的。

3.5.3 TCP分段性能

TCP分段卸载的硬件实现的性能提升意味着：

- 操作系统堆栈无需将块分区以适应MTU大小，从而节省CPU周期。
- 操作系统堆栈每个分段仅计算一次以太网、IP和TCP头部，从而节省CPU周期。
- 操作系统堆栈每次块传输仅与软件设备驱动程序接口一次，而不是每次帧一次。
- 使用更大的PCI突发，从而提高总线效率。
- 中断可以轻松减少到每个TCP消息一次，而不是每个数据包一次。
- 需要较少的I/O访问来控制硬件。

3.5.4 数据包格式

典型的TCP/IP传输窗口大小为8760字节（约6个完整大小的帧）。一个TCP消息可以最大为64 KB，并且通常在主机内存中的多个页面之间进行分片。以太网控制器在传输之前将数据包划分为标准的以太网帧。以太网控制器支持逐帧计算以太网、IP、TCP甚至UDP头部，包括校验和。

以太网	IPv4	TCP/UDP	DATA	FCS

图3-5。TCP/IP数据包格式

以太网控制器TCP分段的帧格式支持：

- 以太网802.3
- IEEE 802.1q VLAN（以太网802.3ac）
- 以太网类型2
- 以太网SNAP
- IPv4头部带选项
- IPv6头部带IP选项下一头部
- IPv6包隧道传输IPv4

- TCP with options
- UDP with limitations.

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 Ethernet controller has the ability to segment UDP traffic (in addition to TCP traffic). This process has limited usefulness.

IP tunneled packets are not supported for TCP Segmentation operation[1].

3.5.5 TCP Segmentation Indication

Software indicates a TCP Segmentation transmission context to the hardware by setting up a TCP/IP Context Transmit Descriptor. The purpose of this descriptor is to provide information to the hardware to be used during the TCP segmentation offload process. The layout of this descriptor is reproduced in Section 3.3.6.

	63 48	47 40	39 32	31 16	15 8	7 0
0	TUCSE	TUCS0	TUCSS	IPCSE	IPCS0	IPCSS

	63 48	47 40	39 36	35 32	31 24	23 20	19 0
8	MSS	HDRLEN	RSV	STA	TUCMD	DTYP	PAYLEN

7	6	5	4	3	2	1	0
IDE	RSV	DEXT	RSV	RS	TSE	IP	TCP

Figure 3-6. TCP/IP Context Transmit Descriptor & Command Layout

Setting the TSE bit in the Command field to 1b indicates that this descriptor refers to the TCP Segmentation context (as opposed to the normal checksum offloading context). This causes the checksum offloading, packet length, header length, and maximum segment size parameters to be loaded from the descriptor into the Ethernet controller.

The TCP Segmentation prototype header is taken from the packet data itself. Software must identity the type of packet that is being sent (IP/TCP, IP/UDP, other), calculate appropriate checksum offloading values for the desired checksums, and calculate the length of the header which is pre-pended. The header may 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 may 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:

- 带选项的TCP
- 带限制的UDP

UDP（与TCP不同）不是“可靠协议”，在UDP层不支持分片。大于给定网络介质MTU大小的UDP消息通常在IP层进行分片。这与TCP不同，在TCP中，大型TCP消息可以在IP层或TCP层进行分片，具体取决于软件实现。以太网控制器具有分割UDP流量（除TCP流量外）的能力。此过程实用性有限。

IP隧道数据包不支持TCP分段操作[1]。

3.5.5 TCP分段指示

软件通过设置TCP/IP上下文传输描述符，向硬件指示TCP分段传输上下文。此描述符的目的是为硬件提供信息，以便在TCP分段卸载过程中使用。此描述符的布局在3.3.6节中重新呈现。

	63 48	47 40	39 32	31 16	15 8	7 0
0	TUCSE	TUCS0	TUCSS	IPCSE	IPCS0	IPCSS

	63 48	47 40	3936	35 32	31 24	23 20	19 0
8	MSS	HDRLEN	RSV	STA	TUCMD	DTYP	PAYLEN

7	6	5	4	3	2	1	0
IDE	RSV	DEXT	RSV	RS	TSE	IP	TCP

图3-6。TCP/IP上下文传输描述符和命令布局

将命令字段的TSE位设置为1b表示此描述符引用TCP分段上下文（相对于正常校验和卸载上下文）。这会导致校验和卸载、数据包长度、头部长度和最大分段大小参数从描述符加载到以太网控制器中。

TCP分段原型头取自数据包本身。软件必须识别正在发送的数据包类型（IP/TCP、IP/UDP、其他），为所需的校验和计算适当的校验和卸载值，并计算预置头部长度。头部长度最多可达240字节。

一旦TCP分段上下文被设置，下一个描述符提供传输的初始数据。这个第一个描述符必须指向类型指示的数据包。此外，它指向的数据可能需要由软件修改，因为它作为TCP分段上下文中所有数据包的原型头部。以下部分描述了支持的数据包类型以及硬件执行的各种更新。这应作为指南，以确定必须修改原始数据包头部的内容，使其成为合适的原型头部。

以下总结了驱动程序在构建原型头部时考虑的用于修改的字段：

- IPv4 Header
  - — Length should be set to zero
  - — 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
- IPv6 Header
  - — Length should be set to zero
- TCP Header
  - — Sequence Number should be set as appropriate for first packet of send (if not already)
  - — PSH, and FIN flags should be set as appropriate for last packet of send
  - — TCP Checksum should be set to the partial pseudo-header checksum as follows:

IP Source Address		
IP Destination Address		
Zero		
Zero		Next Header
Zero[a]	Layer 4 Protocol[a]	Zero[a]

a. **82544GC/EI** only

Figure 3-7. TCP Partial Pseudo-Header Checksum

- UDP Header
  - — Checksum should be set as in TCP header, above

The Ethernet controller's DMA function fetches the ethernet, IP, and TCP/UDP prototype header information from the initial descriptor(s) and save them on-chip for individual packet header generation. The following sections describe the updating process performed by the hardware for each frame sent using the TCP Segmentation capability.

3.5.6 TCP Segmentation Use of Multiple Data Descriptors

TCP Segmentation enables a packet to be segmented to describe more than one data descriptor. A large packet contained in a single virtual-address buffer is better described as a series of data descriptors, each referencing a single physical address page.

The only requirement for this use is if multiple data descriptors for TCP segmentation follows this guideline:

- If multiple data descriptors are used to describe the IP/TCP/UDP header section, each descriptor must describe one or more complete headers; descriptors referencing only parts of headers are not supported.

- IPv4头部
  - — 长度应设置为零
  - — 标识字段应设置为发送第一个数据包时适当的值（如果尚未设置）
  - — 头部校验和应清零，除非驱动程序需要调整
- IPv6头部
  - — 长度应设置为零
- TCP头部
  - — 序列号应设置为适当值，适用于发送的第一个数据包（如果尚未设置）
  - — PSH和FIN标志应设置为适当值，适用于发送的最后一个数据包
  - — TCP校验和应设置为如下部分伪头部校验和：

IP源地址		
IP目标地址		
Zero		
Zero		下一个头部
零[a]	第4层 协议[a]	零[a]

a. 仅82544GC/EI

图3-7。TCP部分伪头部校验和

- UDP头部
  - — 校验和应与TCP头部中的设置相同，如上所示

以太网控制器的DMA功能从初始描述符中获取以太网、IP和TCP/UDP原型头部信息，并将它们保存在片上以进行单个数据包头部的生成。以下部分描述了硬件针对使用TCP分段能力发送的每个帧执行更新过程。

3.5.6 TCP分段使用多个数据描述符

TCP分段允许一个数据包被分段以描述多个数据描述符。一个包含在单个虚拟地址缓冲区中的大数据包最好被描述为一系列数据描述符，每个描述符引用一个物理地址页。

此用途的唯一要求是，如果多个TCP分段数据描述符遵循此指南：

- 如果使用多个数据描述符来描述IP/TCP/UDP头部部分，每个描述符必须描述一个或多个完整的头部；仅引用头部部分的描述符不受支持。

Note: It is recommended that the entire header section, as described by the TCP Context Descriptor HDRLEN field, be coalesced into a single buffer and described using a single data descriptor.

3.5.7 IP and TCP/UDP Headers

This section outlines the format and content for the IP, TCP and UDP headers. The Ethernet controller requires baseline information from the software device driver in order to construct the appropriate header information during the segmentation process.

Header fields that are modified by the Ethernet controller are highlighted in the figures that follow.

The IPv4 header is first shown in the traditional (RFC 791) representation, and because byte and bit ordering is confusing in that representation, the IP header is also shown in little-endian format. The actual data is fetched from memory in little-endian format.

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 34 5 6 7 8 9 0 1

Version	IP Hdr Length	TYPE of service	Total length	
Identification			Flags	Fragment Offset
Time to Live		Layer 4 Protocol ID	Header Checksum	
Source Address				
Destination Address				
Options				

Figure 3-8. IPv4 Header (Traditional Representation)

注意：建议将整个头部区域（如TCP上下文描述符HDRLEN字段所述）合并到一个缓冲区中，并使用一个数据描述符进行描述。

3.5.7 IP和TCP/UDP头

本节概述了IP、TCP和UDP头的格式和内容。以太网控制器需要在分段过程中根据软件设备驱动程序提供的基础信息来构建适当的头部信息。

以太网控制器修改的头部字段在后续的图中突出显示。

IPv4头部首先以传统（RFC 791）的形式显示，并且由于字节和位顺序在该表示中令人困惑，IP头部也以小端格式显示。实际数据以小端格式从内存中获取。

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

版本	IP头长度	服务类型	总长度	
标识			标志	分片偏移
生存时间		第4层协议ID	头部校验和	
源地址				
目的地地址				
选项				

图3-8。IPv4头（传统表示）

Byte 3	Byte 2				Byte 1	Byte 0	
7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0				7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	
LSB Total length MSB					TYPE of service	Version	IP Hdr Length
Fragment Offset Low	R E S	N F	M F	Fragment Offset High	LSB Identification MSB		
Header Checksum					Layer 4 Protocol ID	Time to Live	
Source Address							
Destination Address							
Options							

Figure 3-9. IPv4 Header (Little-Endian Order)

Flags Field Definition:

The Flags field is defined below. Note that hardware does not evaluate or change these bits.

• MF	More Fragments
• NF	No Fragments
• Reserved	

Note: The IPv6 header is first shown in the traditional (RFC 2460), big-endian representation. The actual data is fetched from memory in little-endian format.

0 1 2 3 4 5 6 7 8 9 ^{1}0 1 2 3 4 5 6 7 8 9 ^{2}0 1 2 3 4 5 6 7 8 9 ^{3}0 1			
Version	Traffic Class	Flow Label	
Payload Length		Next Header	Hop Limit
Source Address			
Destination Address			

Figure 3-10. IPv6 TCP Header (Traditional Representation)

A TCP or UDP frame uses a 16 bit wide one's complement checksum. The checksum word is computed on the outgoing TCP or UDP header and payload, and on the Pseudo Header. Details on checksum computations are provided in Section 3.5. TCP requires the use of checksum, where it is optional for UDP.

字节3	字节2				字节1	字节0	
7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0							
LSB 总长度 MSB					服务类型	版本	IP头长度
分片偏移低	R E S	N F	M F	片段偏移 高位	LSB标识MSB		
头部校验和					第4层协议ID	生存时间	
源地址							
目的地地址							
选项							

图3-9。IPv4头部（小端序）

标志字段定义：

标志字段定义如下。注意，硬件不会评估或更改这些位。

• MF	更多片段
• NF	无片段
• 保留	

注意：IPv6头部首先以传统（RFC 2460）的大端序表示显示。实际数据以小端格式从内存中获取。

01234567891012345678920123456789301			
版本	流量类别	流标签	
有效载荷长度		下一个头部	跳数限制
源地址			
目的地地址			

图3-10。IPv6 TCP头部（传统表示）

一个TCP或UDP帧使用一个16位宽的一补码校验和。校验和字是在出站TCP或UDP头部和有效载荷以及伪头部上计算的。校验和计算的详细信息在第3.5节中提供。TCP需要使用校验和，而UDP则可选。

The TCP header is first shown in the traditional (RFC 793) representation. Because byte and bit ordering is confusing in that representation, the TCP header is also shown in little-endian format. The actual data is fetched from memory in little-endian format.

0 1 2 3 4 5 6 7 8 9	1 0 1 2 3 4 5 6 7 8 9	2 0 1 2 3 4 5 6 7 8 9	3 0 1					
Source Port								Destination Port
Sequence Number								
Acknowledgement Number								
TCP Header Length	Reserved	U R G	A C K	P S H	R S T	S Y N	F I N	Window
Checksum								Urgent Pointer
Options								

Figure 3-11. TCP Header (Traditional Representation)

Byte3	Byte2						Byte1		Byte0
7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0						7 6 5 4 3 2 1 0		7 6 5 4 3 2 1 0
Destination Port							Source Port		
LSB	Sequence Number								MSB
Acknowledgement Number									
Window	RE S	U R G	A C K	P S H	R S T	S Y N	FI N	TCP Header Length	Reserved
Urgent Pointer							Checksum		
Options									

Figure 3-12. TCP Header (Little-Endian)

The TCP header is always a multiple of 32 bit words. TCP options may occupy space at the end of the TCP header and are a multiple of 8 bits in length. All options are included in the checksum.

The checksum also covers a 96-bit pseudo header conceptually prefixed to the TCP Header (see Figure 3-13 and Figure 3-14). The IPv4 pseudo header contains the IPv4 Source Address, the IPv4 Destination Address, the IPv4 Protocol field, and TCP Length. The IPv6 pseudo header contains the IPv6 Source Address, the IPv6 Destination Address, the IPv6 Payload Length, and the IPv6 Next Header field. Software pre-calculates the <u>partial</u> pseudo header sum, which includes IPv4 SA, DA and protocol types, but <u>not</u> the TCP length, and stores this value into the TCP checksum field of the packet.

The Protocol ID field should always be added the least significant byte (LSB) of the 16 bit pseudo header sum, where the most significant byte (MSB) of the 16 bit sum is the byte that corresponds to the first checksum byte out on the wire.

The TCP Length field is the TCP Header Length including option fields plus the data length in bytes, which is calculated by hardware on a frame by frame basis. The TCP Length does not count the 12 bytes of the pseudo header. The TCP length of the packet is determined by hardware as:

TCP头部首先以传统（RFC 793）的形式显示。由于在该表示中字节和位顺序令人困惑，TCP头部也以小端格式显示。实际数据以小端格式从内存中获取。

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								
源端口								目标端口
序列号								
确认号								
TCP头部长度	保留	U R G	A C K	P S H	R S T	S Y N	FIN	窗口
校验和								紧急指针
选项								

图3-11。TCP头部（传统表示）

字节3	字节2						字节1		字节0
7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0									
目标端口							源端口		
LSB序列号MSB									
确认号									
窗口	R ES	U R G	A C K	P S H	R S T	S Y N	FI N	TCP 头部长度	保留
紧急指针							校验和		
选项									

图3-12。TCP头部（小端模式）

TCP头部始终是32位字的倍数。TCP选项可能占据TCP头部末尾的空间，并且长度是8位的倍数。所有选项都包含在校验和中。

校验和也涵盖了在TCP头部概念性前缀的96位伪头部（参见图3-13和图3-14）。IPv4伪头部包含IPv4源地址、IPv4目的地址、IPv4协议字段和TCP长度。IPv6伪头部包含IPv6源地址、IPv6目的地址、IPv6有效载荷长度和IPv6下一个头部字段。软件预先计算部分伪头部和，包括IPv4 SA、DA和协议类型，但不包括TCP长度，并将此值存储到数据包的TCP校验和字段中。

协议ID字段应始终添加到16位伪头部和的最低有效位字节（LSB）中，其中16位和的最高有效位字节（MSB）是线路上第一个校验和字节对应的字节。

TCP长度字段是TCP头部长度（包括选项字段）加上以字节为单位的数据长度，该值由硬件逐帧计算。TCP长度不包括伪头部的12个字节。数据包的TCP长度由硬件确定：

TCP Length = Payload + HDRLEN - TUCSS

“Payload” is normally MSS except for the last packet where it represents the remainder of the payload.

0		31
IP Source Address		
IP Destination Address		
Zero	Layer 4 Protocol ID	TCP Length

Figure 3-13. TCP Pseudo Header Content (Traditional Representation)

IP Source Address	
IP Destination Address	
Upper Layer Packet Length	
Zero	Next Header

Figure 3-14. TCP PseudoHeader Content for IPv6

Note: The IP Destination address is the final destination of the packet. Therefore, if a routing header is used, the last address in the route list is used in this calculation. The upper-layer packet length is the length of the TCP header and the TCP payload.

The UDP header is always 8 bytes in size with no options.

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
Length	Checksum

Figure 3-15. UDP Header (Traditional Representation)

Byte3	Byte2	Byte1	Byte0
0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
Destination Port		Source Port	
Checksum		Length	

Figure 3-16. UDP Header (Little-Endian Order)

TCP Length = Payload + HDRLEN - TUCSS

"有效载荷"通常表示MSS，但在最后一个数据包中，它表示有效载荷的剩余部分。

0		31
IP源地址		
IP目标地址		
Zero	第4层协议ID	TCP长度

图3-13。TCP伪头内容（传统表示）

IP源地址	
IP目标地址	
上层数据包长度	
Zero	下一个头部

图3-14。IPv6的TCP伪头内容

注意：IP目标地址是数据包的最终目的地。因此，如果使用路由头，则在此计算中使用路由列表中的最后一个地址。上层报文长度是TCP头部和TCP有效载荷的长度。

UDP头部始终为8字节大小，且没有选项。

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	
源端口	目标端口
长度	校验和

图3-15。UDP头部（传统表示）

字节3	字节2	字节1	字节0
0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7			
目标端口		源端口	
校验和		长度	

图3-16。UDP头部（小端模式）

UDP pseudo header has the same format as the TCP pseudo header. The IPv4 pseudo header conceptually prefixed to the UDP header contains the IPv4 source address, the IPv4 destination address, the IPv4 protocol field, and the UDP length (same as the TCP Length discussed above). The IPv6 pseudo header for UDP is the same as the IPv6 pseudo header for TCP. This checksum procedure is the same as is used in TCP.

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

Figure 3-17. UDP Pseudo Header Diagram for IPv4

IP Source Address	
IP Destination Address	
Upper Layer Packet Length	
Zero	Next Header

Figure 3-18. UDP PseudoHeader Diagram for IPv6

Note: The IP Destination Address is the final destination of the packet. Therefore, if a routing header is used, the last address in the route list is used in this calculation. The upper-layer packet length is the length of the UDP header and UDP payload.

Unlike the TCP checksum, the UDP checksum is optional. Software must set the TXSM bit in the TCP/IP Context Transmit Descriptor to indicate that a UDP checksum should be inserted. Hardware does not overwrite the UDP checksum unless the TXSM bit is set.

3.5.8 Transmit Checksum Offloading with TCP Segmentation

The Ethernet controller supports checksum off-loading as a component of the TCP Segmentation offload feature and as a standalone capability. Section 3.5.8 describes the interface for controlling the checksum off-loading feature. This section describes the feature as it relates to TCP Segmentation.

The Ethernet controller supports IP and TCP/UDP header options in the checksum computation for packets that are derived from the TCP Segmentation feature. The Ethernet controller 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 has to compute all but one IP header checksum. The Ethernet controller calculates checksums 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.

UDP伪头部与TCP伪头部格式相同。概念上位于UDP头部之前的IPv4伪头部包含IPv4源地址、IPv4目的地址、IPv4协议字段和UDP长度（与上述讨论的TCP长度相同）。UDP的IPv6伪头部与TCP的IPv6伪头部相同。此校验和过程与TCP中使用的相同。

IP源地址		
IP目标地址		
Zero	第4层协议ID	UDP长度

图3-17。IPv4的UDP伪头图

IP源地址	
IP目标地址	
上层数据包长度	
Zero	下一个头部

图3-18。IPv6的UDP伪头图

注意：IP目标地址是数据包的最终目的地。因此，如果使用路由头，则在此计算中使用路由列表中的最后一个地址。上层报文长度是UDP头部和UDP有效载荷的长度。

与TCP校验和不同，UDP校验和是可选的。软件必须在TCP/IP上下文传输描述符中设置TXSM位，以指示应插入UDP校验和。硬件不会覆盖UDP校验和，除非TXSM位被设置。

3.5.8 使用TCP分段传输校验和卸载

以太网控制器支持校验和卸载作为TCP分段卸载功能的一部分，以及作为独立功能。第3.5.8节描述了控制校验和卸载功能的接口。本节描述了该功能与TCP分段的关系。

以太网控制器支持IP和TCP/UDP头选项在数据包的校验和计算中，这些数据包源自TCP分段功能。以太网控制器能够计算IP头校验和的一个级别和TCP/UDP头和有效载荷校验和。在多个IP头的情况下，驱动程序必须计算除一个之外的所有IP头校验和。以太网控制器逐帧动态计算校验和，并将结果插入每个帧的IP/TCP/UDP头部。TCP和UDP校验和是通过对有效载荷的所有字节和伪头部进行校验和计算得出的。

Three specific types of checksum are supported by the hardware in the context of the TCP Segmentation offload feature:

- IPv4 checksum (IPv6 does not have a checksum)
- TCP checksum
- UDP checksum

Each packet that is sent via the TCP segmentation offload feature optionally includes the IPv4 checksum and either the TCP or UDP checksum.

All checksum calculations use a 16-bit wide one's complement checksum. The checksum word is calculated on the outgoing data. The checksum field is written with the 16 bit one's complement of the one's complement sum of all 16-bit words in the range of CSS to CSE, including the checksum field itself.

3.5.9 IP/TCP/UDP Header Updating

IP/TCP/UDP header is updated for each outgoing frame based on the IP/TCP header prototype which hardware transfers from the first descriptor(s) and stores on chip. The IP/TCP/UDP headers are fetched from host memory into an on-chip 240 byte header buffer once for each TCP segmentation context (for performance reasons, this header is not fetched again for each additional packet that is derived from the TCP segmentation process). The checksum fields and other header information are later updated on a frame by frame basis. The updating process is performed concurrently with the packet data fetch.

The following sections define which fields are modified by hardware during the TCP Segmentation process by the Ethernet controller. Figure 3-19 illustrates the overall data flow.

在TCP分段卸载功能上下文中，硬件支持三种特定的校验和类型:

- IPv4校验和（IPv6没有校验和）
- TCP校验和
- UDP校验和

通过TCP分段卸载功能发送的每个数据包可以选择性地包含IPv4校验和，以及TCP或UDP校验和中的一种。

所有校验和计算都使用16位宽的一补码校验和。校验和字是在出站数据上计算的。校验和字段被写入CSS到CSE范围内的所有16位字的补码和的一补码，包括校验和字段本身。

3.5.9 IP/TCP/UDP头部更新

IP/TCP/UDP头部是根据硬件从第一个描述符(s)传输并存储在片上的IP/TCP头部原型进行更新的。IP/TCP/UDP头部从主机内存中获取到片上240字节头部缓冲区一次，每个TCP分段上下文（出于性能原因，这个头部不会为TCP分段过程中派生的每个附加数据包再次获取）。校验和字段和其他头部信息稍后逐帧更新。更新过程与数据包数据获取同时进行。

以下小节定义了以太网控制器在TCP分段过程中修改哪些字段。图3-19说明了整体数据流。

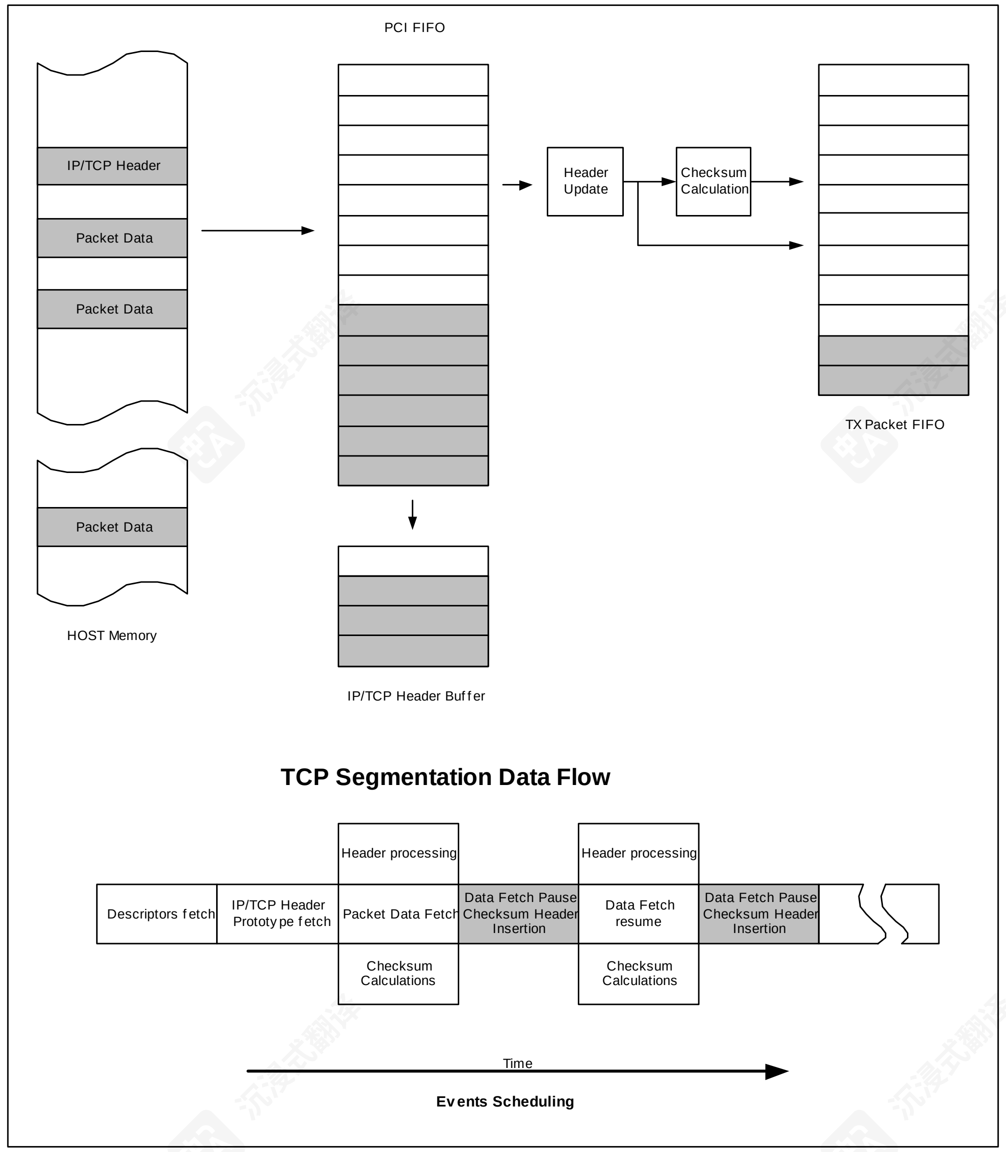


Figure 3-19. Overall Data Flow

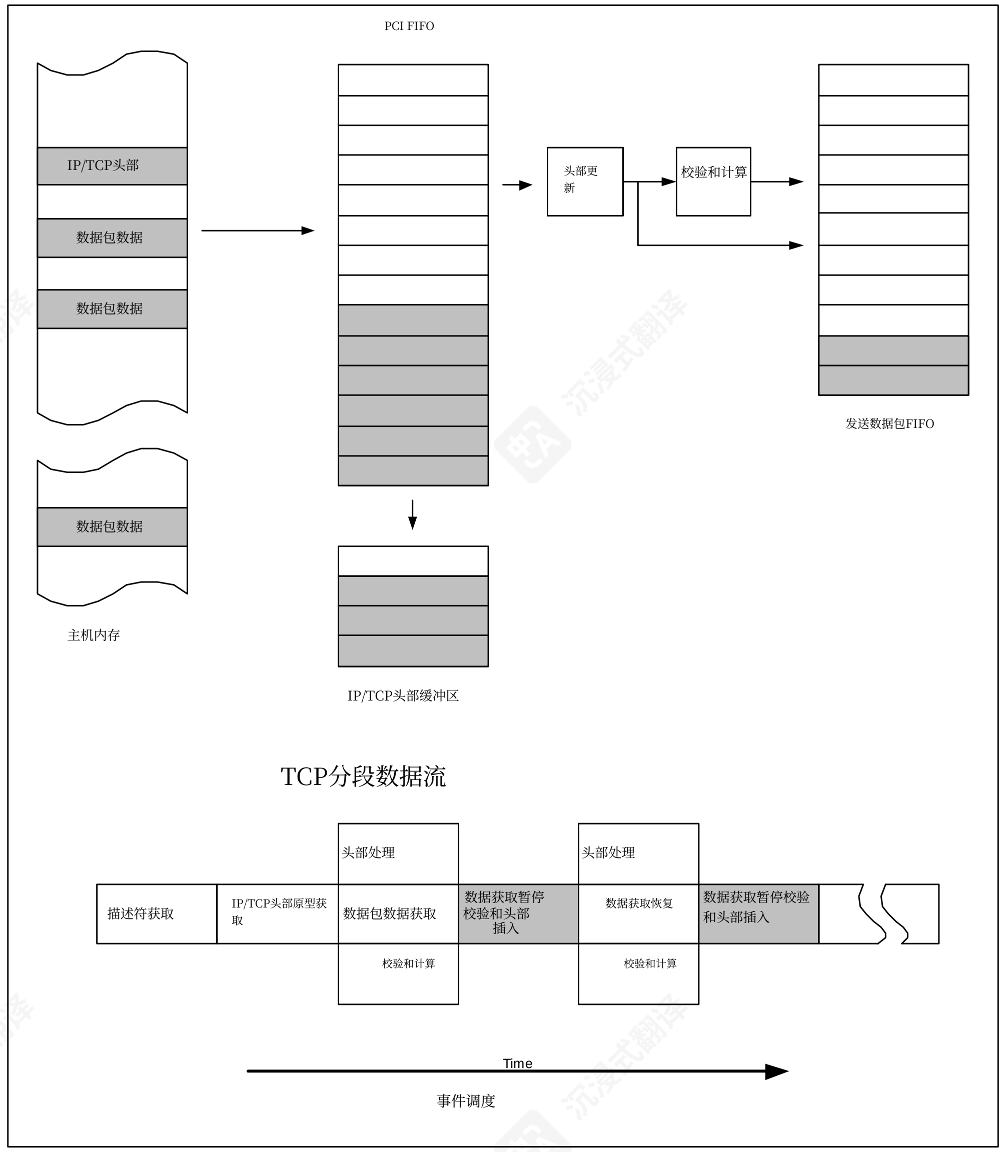


图3-19。整体数据流

3.5.9.1 TCP/IP/UDP Header for the First Frame

The hardware makes the following changes to the headers of the first packet that is derived from each TCP segmentation context.

- IPv4 Header
  - — IP Total Length = MSS + HDRLEN – IPCSS
  - — IP Checksum
  - — IPv6 Header
  - — Payload Length = MSS + HDRLEN - IPCSS
- TCP Header
  - — Sequence Number: The value is the Sequence Number of the first TCP byte in this frame.
  - — If FIN flag = 1b, it is cleared in the first frame.
  - — If PSH flag =1b, it is cleared in the first frame.
  - — TCP Checksum
- UDP Header
  - — UDP length: MSS + HDRLEN - TUCSS
  - — UDP Checksum

3.5.9.2 TCP/IP/UDP 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:

Note: Number of bytes left for transmission = PAYLEN – (N * MSS). Where N is the number of frames that have been transmitted.

- IPv4 Header
  - — IP Identification: incremented from last value (wrap around)
  - — IP Total Length = MSS + HDRLEN – IPCSS
  - — IP Checksum
- IPv6 Header
- Payload Length = MSS + HRDLEN - IPCSS
- TCP Header
  - — Sequence Number update: Add previous TCP payload size to the previous sequence number value. This is equivalent to adding the MSS to the previous sequence number.
  - — If FIN flag = 1b, it is cleared in these frames.
  - — If PSH flag =1b, it is cleared in these frames.
  - — TCP Checksum
- UDP Header
  - — UDP Length: MSS + HDRLEN – TUCSS
  - — UDP Checksum

3.5.9.1 第一个帧的TCP/IP/UDP头部

硬件对每个TCP分段上下文派生的第一个数据包的头部进行了以下更改。

- IPv4头部
  - – IP总长度 = MSS + HDRLEN – IPCSS
  - – IP校验和
  - – IPv6头部
  - – 有效载荷长度 = MSS + HDRLEN - IPCSS
- TCP头部
  - – 序列号：该值是此帧中第一个TCP字节的序列号。
  - – 如果FIN标志 = 1b，它在第一个帧中被清除。
  - – 如果PSH标志 =1b，它在第一个帧中被清除。
  - – TCP校验和
- UDP头部
  - – UDP长度：MSS + HDRLEN - TUCSS
  - – UDP校验和

3.5.9.2 后续帧的 TCP/IP/UDP 头部

硬件对作为 TCP 分段上下文一部分派生的后续数据包的头部进行以下更改：

注意：剩余用于传输的字节数 = PAYLEN – (N * MSS)。其中 N 是已传输的帧数。

- IPv4 头部
  - – IP标识：从上一个值递增（回绕）
  - – IP总长度 = MSS + HDRLEN – IPCSS
  - – IP校验和
- IPv6头部
- 有效载荷长度 = MSS + HRDLEN - IPCSS
- TCP头部
  - – 序列号更新：将前一个TCP有效载荷大小加到前一个序列号值上。这相当于将MSS加到前一个序列号上。
  - – 如果FIN标志 = 1b，则在这些帧中清除。
  - – 如果PSH标志 =1b，则在这些帧中清除。
  - – TCP校验和
- UDP头部
  - – UDP长度: MSS + HDRLEN – TUCSS
  - – UDP校验和

3.5.9.3 TCP/IP/UDP Header for the Last Frame

The controller makes the following changes to the headers for the last frame of a TCP segmentation context:

Note: Last frame payload bytes = PAYLEN – (N * MSS)

- IPv4 Header
  - — IP Total Length = (last frame payload bytes + HDRLEN) – IPCSS
  - — IP Identification: incremented from last value (wrap around)
  - — IP Checksum
- IPv6 Header
- Payload Length = MSS + HDRLEN - IPCSS
- TCP Header
  - — Sequence Number update: Add previous TCP payload size to the previous sequence number value. This is equivalent to adding the MSS to the previous sequence number.
  - — If FIN flag = 1b, set it in this last frame
  - — If PSH flag =1b, set it in this last frame
  - — TCP Checksum
- UDP Header
  - — UDP length: (last frame payload bytes + HDRLEN) - TUCSS
  - — UDP Checksum

3.6 IP/TCP/UDP Transmit Checksum Offloading

The previous section on TCP Segmentation offload describes the IP/TCP/UDP checksum offloading mechanism used in conjunction with TCP Segmentation. The same underlying mechanism can also be applied as a standalone feature. The main difference in normal packet mode (non-TCP Segmentation) is that only the checksum fields in the IP/TCP/UDP headers need to be updated.

Before taking advantage of the Ethernet controller's enhanced checksum offload capability, a checksum context must be initialized. For the normal transmit checksum offload feature, this task is performed by providing the Ethernet controller with a TCP/IP Context Descriptor with TSE = 0b to denote a non-segmentation context. For additional details on contexts, refer to Section 3.3.5. Enabling the checksum offloading capability without first initializing the appropriate checksum context leads to unpredictable results. Once the checksum context has been set, that context, is used for all normal packet transmissions until a new context is loaded. Also, since checksum insertion is controlled on a per packet basis, there is no need to clear/reset the context.

The Ethernet controller is capable of performing two transmit checksum calculations. Typically, these would be used for TCP/IP and UDP/IP packet types, however, the mechanism is general enough to support other checksums as well. Each checksum operates independently and provides identical functionality. Only the IP checksum case is discussed as follows.

3.5.9.3 最后一个帧的TCP/IP/UDP头部

控制器对 TCP 分段上下文最后帧的头部进行以下更改:

注意：最后一个帧的有效载荷字节= PAYLEN – (N * MSS)

- IPv4头部
  - — IP总长度 = (最后一个帧的有效载荷字节 + HDRLEN) – IPCSS
  - — IP标识：从上一个值递增（回绕）
  - — IP校验和
- IPv6头部
- 有效载荷长度 = MSS + HDRLEN - IPCSS
- TCP头部
  - — 序列号更新：将前一个TCP有效载荷大小添加到前一个序列号值中。这相当于将MSS添加到前一个序列号。
  - — 如果FIN标志 = 1b，将其设置在这个最后一个帧中
  - — 如果PSH标志 =1b，将其设置在这个最后一个帧中
  - — TCP校验和
- UDP头部
  - — UDP长度：（最后一个帧的有效载荷字节 + HDRLEN) - TUCSS
  - — UDP校验和

3.6 IP/TCP/UDP发送校验和卸载

上一节关于TCP分段卸载描述了与TCP分段配合使用的IP/TCP/UDP校验和卸载机制。相同的底层机制也可以作为独立功能应用。在正常数据包模式（非TCP分段）下的主要区别是只需要更新IP/TCP/UDP头中的校验和字段。

在利用以太网控制器的增强型校验和卸载功能之前，必须初始化校验和上下文。对于正常发送校验和卸载功能，此任务是通过向以太网控制器提供带有TSE = 0b的TCP/IP上下文描述符来完成的，以表示非分段上下文。有关上下文的更多详细信息，请参阅第3节第3.3.5节。在未首先初始化适当的校验和上下文的情况下启用校验和卸载功能会导致不可预测的结果。一旦设置了校验和上下文，该上下文将用于所有正常数据包传输，直到加载新的上下文为止。此外，由于校验和插入是按包控制的，因此无需清除/重置上下文。

以太网控制器能够执行两种发送校验和计算。通常，这些用于TCP/IP和UDP/IP数据包类型，但是，该机制足够通用，以支持其他校验和。每个校验和独立运行并提供相同的功能。仅讨论IP校验和的情况如下。

Three fields in the TCP/IP Context Descriptor set the context of the IP checksum offloading feature:

- IPCSS

  This field specifies the byte offset form the start of the transferred data to the first byte to be included in the checksum. Setting this value to 0b means that the first byte of the data is included in the checksum. The maximum value for this field is 255. This is adequate for typical applications.

Note: The IPCSS value needs to be less than the total DMA length to a packet. If this is not the case, the result will be unpredictable.

- IPCSO

  This field specifies where the resulting checksum should be placed. Again, this is limited to the first 256 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 checksum is not inserted.

- IPCSE

  This field specifies where the checksum should stop. A 16-bit value supports checksum offloading of packets as large as 64KB. Setting the IPCSE field to all zeros means End-of-Packet. In this way, the length of the packet does not need to be calculated.

As mentioned above, 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 of the offload feature only for a particular traffic type, thereby avoiding all context descriptors except for the initial one.

TCP/IP上下文描述符中的三个字段设置了IP校验和卸载功能的环境:

- IPCS

  该字段指定从传输数据的开始位置到校验和中包含的第一个字节的字节偏移量。将此值设置为0b意味着数据的第一个字节包含在校验和中。此字段的最大值是255。这对于典型应用来说是足够的。

注意：IPCS值需要小于到数据包的总DMA长度。如果不是这样，结果将是不可预测的。

- IPCSO

  此字段指定校验和应放置的位置。同样，这仅限于数据包的前256字节，并且必须小于或等于给定数据包的总长度。如果不是这种情况，则不会插入校验和。

- IPCSE

  此字段指定校验和应停止的位置。一个16位值支持卸载最大为64KB的数据包。将IPCSE字段设置为全零表示数据包结束。通过这种方式，无需计算数据包的长度。

如上所述，对于每个新数据包，没有必要设置新的上下文。在许多情况下，相同的校验和上下文可用于大部分数据包流。在这种情况下，某些卸载功能仅针对特定的流量类型，从而避免除初始一个之外的所有上下文描述符。

Note: This page intentionally left blank.

注意：本页故意留空。

General Initialization and Reset Operation 14

14.1 Introduction

This section lists all necessary initializations and describes the reset commands for the PCI/PCI-X Family of Gigabit Ethernet Controllers.

Note: TBI mode is used by the **82544GC/EI**. Internal SerDes is used by the **82546GB/EB** and **82545GM/EM**.

14.2 Power Up State

At power up, the Ethernet controller is not automatically configured by the hardware for normal operation. Software initialization is required before normal operation can continue. In general, the Ethernet controller is considered non-functional until the software driver successfully loads and sets up the hardware. However, Auto-Negotiation can start at power up or upon receipt of an assertion of PCI reset if configured to do so by the EEPROM.

14.3 General Configuration

Several values in the Device Control Register (CTRL) need to be set upon power up or after an Ethernet controller reset for normal operation.

- Speed and duplex are determined via Auto-Negotiation by the PHY, Auto-Negotiation by the MAC for internal SerDes[1] mode, or forced by software if the link is forced. In internal PHY mode, the Ethernet controller can be configured automatically by hardware or forced by software to the same configuration as the PHY.
- In internal PHY mode, the Auto-Speed Detection Enable (CTRL.ASDE) bit, when set to 1b, detects the resolved speed and duplex of the link and self-configure the MAC appropriately. This bit should be set in conjunction with the Set Link Up (CTRL.SLU) bit.
- The MAC can also be forced to a specific Speed/Duplex combination. This is accomplished by setting the Set Link Up (CTRL.SLU), Force Speed (CTRL. FRCSPD) and Force Duplex (CTRL.FRCDPLX) bits. Once speed and duplex are determined (either via Auto-Negotiation or forced by software), speed is forced by setting the appropriate Speed Selection (CTRL.SPEED) bits and duplex is forced by updating the Full Duplex (CTRL.FD) bit.
- For the **82541xx** and **82547GI/EI**, configure the LED behavior through LEDCTRL.
- Link Reset (CTRL.LRST) should be set to 0b (normal). The Ethernet controller defaults to LRST = 1b which disables Auto-Negotiation. A transition to 0b initiates the Auto-Negotiation function. LRST can be defined in the EEPROM. This bit is only valid in internal SerDes mode and has no effect in internal PHY mode.

1. The **82540EP/EM**, **82541xx**, and **82547GI/EI** do not support any SerDes functionality.

通用初始化和复位操作 14

14.1 简介

本节列出了所有必要的初始化操作，并描述了 PCI/PCI-X 千兆以太网控制器系列的复位命令。

注意：82544GC/EI 使用 TBI 模式。82546GB/EB 和 82545GM/EM 使用内部 SerDes。

14.2 上电状态

在上电时，以太网控制器不会由硬件自动配置以进行正常操作。在正常操作继续之前需要软件初始化。通常，在软件驱动成功加载并设置硬件之前，以太网控制器被认为无法正常工作。然而，如果 EEPROM 配置为这样做，自动协商可以在上电时或收到 PCI 复位断言时开始。

14.3 一般配置

设备控制寄存器 (CTRL) 中的几个值需要在启动时或以太网控制器重置后进行设置以实现正常操作。

- 速度和全双工是通过由 PHY 进行的自动协商、由 MAC 进行的自动协商（用于内部 SerDes1 模式），或如果链路被强制，则由软件强制确定。在内部 PHY 模式下，以太网控制器可以由硬件自动配置或由软件强制配置为与 PHY 相同的配置。
- 在内部 PHY 模式下，当自动速度检测使能 (CTRL.ASDE) 位设置为 1b 时，会检测链路的解析速度和全双工，并相应地自配置 MAC。此位应与设置链路开启 (CTRL.SLU) 位一起设置。
- MAC也可以被强制设置为特定的速度/全双工组合。这是通过设置设置链路（CTRL.SLU）、强制速度（CTRL. FRCSPD）和强制全双工（CTRL.FRCDPLX）位来完成的。一旦速度和全双工确定（无论是通过自动协商还是由软件强制），速度通过设置适当的速度选择（CTRL.SPEED）位来强制，全双工通过更新全双工（CTRL.FD）位来强制。
- 对于82541xx和82547GI/EI，通过LEDCTRL配置LED行为。
- 链路复位（CTRL.LRST）应设置为0b（正常）。以太网控制器默认为LRST = 1b，这将禁用自动协商。过渡到0b会启动自动协商功能。LRST可以在EEPROM中定义。此位仅在内部SerDes模式下有效，在内部PHY模式下无效。

1. 82540EP/EM、82541xx 和 82547GI/EI 不支持任何 SerDes 功能。

- PHY Reset (CTRL.PHY_RST) should be set to 0b. Setting this bit to 1b resets the PHY without accessing the PHY registers. This bit is ignored in internal SerDes mode.
- CTRL.ILOS should be set to 0b (not applicable to the **82541xx** and **82547GI/EI**).
- If Flow Control is desired, program the FCAH, FCAL, FCT and FCTTV registers. If not, they should be written with 0b. To enable XON frame transmission, the XON Enable (FCTRL.XONE) bit must be set. Advertising Flow Control capabilities during the Auto-Negotiation process is dependent on whether the Ethernet controller is operating in internal SerDes or internal PHY mode. In internal SerDes mode, the TXCW register must be set up prior to starting the Auto-Negotiation process. In internal PHY mode, the appropriate PHY registers must be set up properly to advertise desired capabilities prior to starting or re-starting the Auto-Negotiation process. The Receive Flow Control Enable (CTRL.RFCE) and Transmit Flow Control Enable (CTRL.TFCE) bits need to be explicitly set by software in internal PHY mode (because Auto-Negotiation is managed by PHY rather than the MAC), or when a fiber connection is desired but link was forced rather than Auto-Negotiated.
- If VLANs are not used, software should clear VLAN Mode Enable (CTRL.VME) bit. In this instance, there is no need then to initialize the VLAN Filter Table Array (VFTA). If VLANs are desired, the VFTA should be both initialized and loaded with the desired information.
- For the **82541xx** and **82547GI/EI**, clear all statistical counters.

14.4 Receive Initialization

Program the Receive Address Register(s) (RAL/RAH) with the desired Ethernet addresses. RAL[0]/RAH[0] should always be used to store the Individual Ethernet MAC address of the Ethernet controller. This can come from the EEPROM or from any other means (for example, on some machines, this comes from the system PROM not the EEPROM on the adapter port).

Initialize the MTA (Multicast Table Array) to 0b. Per software, entries can be added to this table as desired.

Program the Interrupt Mask Set/Read (IMS) register to enable any interrupt the software driver wants to be notified of when the event occurs. Suggested bits include RXT, RXO, RXDMT, RXSEQ, and LSC. There is no immediate reason to enable the transmit interrupts.

If software uses the Receive Descriptor Minimum Threshold Interrupt, the Receive Delay Timer (RDTR) register should be initialized with the desired delay time.

Allocate a region of memory for the receive descriptor list. Software should insure this memory is aligned on a paragraph (16-byte) boundary. Program the Receive Descriptor Base Address (RDBAL/RDBAH) register(s) with the address of the region. RDBAL is used for 32-bit addresses and both RDBAL and RDBAH are used for 64-bit addresses.

Set the Receive Descriptor Length (RDLEN) register to the size (in bytes) of the descriptor ring. This register must be 128-byte aligned.

The Receive Descriptor Head and Tail registers are initialized (by hardware) to 0b after a power-on or a software-initiated Ethernet controller reset. Receive buffers of appropriate size should be allocated and pointers to these buffers should be stored in the receive descriptor ring. Software initializes the Receive Descriptor Head (RDH) register and Receive Descriptor Tail (RDT) with the appropriate head and tail addresses. Head should point to the first valid receive descriptor in the descriptor ring and tail should point to one descriptor beyond the last valid descriptor in the descriptor ring.

- PHY复位（CTRL.PHY_RST）应设置为0b。将此位设置为1b可重置PHY，而无需访问PHY寄存器。在内部SerDes模式下，此位将被忽略。
- CTRL.ILOS应设置为0b（不适用于82541xx和82547GI/EI）。
- 如果需要流控制，应编程FCAH、FCAL、FCT和FCTTV寄存器。如果不需要，则应使用0b写入。要启用XON帧传输，必须设置XON启用（FCTRL.XONE）位。在自动协商过程中是否宣传流控制能力，取决于以太网控制器是在内部SerDes模式还是内部PHY模式下运行。在内部SerDes模式下，必须在开始自动协商过程之前设置TXCW寄存器。在内部PHY模式下，必须在开始或重新开始自动协商过程之前正确设置适当的PHY寄存器，以宣传所需能力。在内部PHY模式下，接收流控制启用（CTRL.RFCE）和发送流控制启用（CTRL.TFCE）位需要由软件显式设置（因为自动协商由PHY管理而不是MAC），或者当希望使用光纤连接但链路被强制而不是自动协商时。
- 如果未使用VLAN，软件应清除VLAN模式启用（CTRL.VME）位。在这种情况下，则无需初始化VLAN过滤表数组（VFTA）。如果需要使用VLAN，则应初始化VFTA并加载所需信息。
- 对于82541xx和82547GI/EI，清除所有统计计数器。

14.4 接收初始化

使用所需的以太网地址编程接收地址寄存器 (RAL/RAH)。RAL[0]/RAH[0] 应始终用于存储以太网控制器的单个以太网 MAC 地址。这可以来自 EEPROM 或任何其他方式（例如，在某些机器上，这来自系统 PROM 而不是适配器端口上的 EEPROM）。

初始化MTA（多播表数组）为0b。根据软件要求，可以按需向此表添加条目。

将中断掩码设置/读取（IMS）寄存器编程为启用软件驱动程序希望在事件发生时接收通知的任何中断。建议的位包括RXT、RXO、RXDMT、RXSEQ和LSC。立即启用传输中断没有直接理由。

如果软件使用接收描述符最小阈值中断，则应使用所需延迟时间初始化接收延迟定时器（RDTR）寄存器。

为接收描述符列表分配一个内存区域。软件应确保此内存对齐在段落（16字节）边界上。使用该区域的地址编程接收描述符基地址（RDBAL/RDBAH）寄存器。RDBAL用于32位地址，而RDBAL和RDBAH都用于64位地址。

将接收描述符长度（RDLEN）寄存器设置为描述符环的大小（以字节为单位）。此寄存器必须对齐128字节。

接收描述符头和尾寄存器在上电或软件发起的以太网控制器复位后由硬件初始化为0b。应分配适当大小的接收缓冲区，并将这些缓冲区的指针存储在接收描述符环中。软件使用适当的首地址和尾地址初始化接收描述符头（RDH）寄存器和接收描述符尾（RDT）寄存器。首地址应指向描述符环中第一个有效接收描述符，尾地址应指向描述符环中最后一个有效接收描述符之后的那个描述符。

Program the Receive Control (RCTL) register with appropriate values for desired operation to include the following:

- Set the receiver Enable (RCTL.EN) bit to 1b for normal operation. However, it is best to leave the Ethernet controller receive logic disabled (RCTL.EN = 0b) until after the receive descriptor ring has been initialized and software is ready to process received packets.
- Set the Long Packet Enable (RCTL.LPE) bit to 1b when processing packets greater than the standard Ethernet packet size. For example, this bit would be set to 1b when processing Jumbo Frames.
- Loopback Mode (RCTL.LBM) should be set to 00b for normal operation.
- Configure the Receive Descriptor Minimum Threshold Size (RCTL.RDMTS) bits to the desired value.
- Configure the Multicast Offset (RCTL.MO) bits to the desired value.
- Set the Broadcast Accept Mode (RCTL.BAM) bit to 1b allowing the hardware to accept broadcast packets.
- Configure the Receive Buffer Size (RCTL.BSIZE) bits to reflect the size of the receive buffers software provides to hardware. Also configure the Buffer Extension Size (RCTL.BSEX) bits if receive buffer needs to be larger than 2048 bytes.
- Set the Strip Ethernet CRC (RCTL.SECRC) bit if the desire is for hardware to strip the CRC prior to DMA-ing the receive packet to host memory.
- For the **82541xx** and **82547GI/EI**, program the Interrupt Mask Set/Read (IMS) register to enable any interrupt the driver wants to be notified of when the even occurs. Suggested bits include RXT, RXO, RXDMT, RXSEQ, and LSC. There is no immediate reason to enable the transmit interrupts. Plan to optimize interrupts later, including programming the interrupt moderation registers TIDV, TADV, RADV and IDTR.
- For the **82541xx** and **82547GI/EI**, if software uses the Receive Descriptor Minimum Threshold Interrupt, the Receive Delay Timer (RDTR) register should be initialized with the desired delay time.

14.5 Transmit Initialization

Allocate a region of memory for the transmit descriptor list. Software should insure this memory is aligned on a paragraph (16-byte) boundary. Program the Transmit Descriptor Base Address (TDBAL/TDBAH) register(s) with the address of the region. TDBAL is used for 32-bit addresses and both TDBAL and TDBAH are used for 64-bit addresses.

Set the Transmit Descriptor Length (TDLEN) register to the size (in bytes) of the descriptor ring. This register must be 128-byte aligned.

The Transmit Descriptor Head and Tail (TDH/TDT) registers are initialized (by hardware) to 0b after a power-on or a software initiated Ethernet controller reset. Software should write 0b to both these registers to ensure this.

Initialize the Transmit Control Register (TCTL) for desired operation to include the following:

- Set the Enable (TCTL.EN) bit to 1b for normal operation.
- Set the Pad Short Packets (TCTL.PSP) bit to 1b.

使用适当的值编程接收控制（RCTL）寄存器以包括所需的操作：

- 将接收器启用（RCTL.EN）位设置为1b以进行正常操作。但是，最好在接收描述符环初始化后且软件准备好处理接收到的数据包之前，保持以太网控制器接收逻辑禁用（RCTL.EN = 0b）。
- 在处理大于标准以太网数据包大小的数据包时，将长包启用（RCTL.LPE）位设置为1b。例如，在处理Jumbo Frames时，此位将被设置为1b。
- 回环模式（RCTL.LBM）应设置为00b以进行正常操作。
- 将接收描述符最小阈值大小（RCTL.RDMTS）位配置为所需值。
- 将多播偏移（RCTL.MO）位配置为所需值。
- 将广播接受模式（RCTL.BAM）位设置为1b，允许硬件接受广播数据包。
- 将接收缓冲区大小（RCTL.BSIZE）位配置为反映接收缓冲区软件提供给硬件的大小。如果接收缓冲区需要大于2048字节，请配置缓冲区扩展大小（RCTL.BSEX）位。
- 如果希望硬件在将接收数据包DMA传输到主机内存之前剥离CRC，请设置剥离以太网CRC（RCTL.SECRC）位。
- 对于82541xx和82547GI/EI，编程中断掩码设置/读取（IMS）寄存器以启用驱动程序希望在事件发生时通知的任何中断。建议的位包括RXT、RXO、RXDMT、RXSEQ和LSC。没有立即启用传输中断的理由。计划稍后优化中断，包括编程中断调节寄存器TIDV、TADV、RADV和IDTR。
- 对于 82541xx 和 82547GI/EI，如果软件使用接收描述符最小阈值中断，接收延迟定时器（RDTR）寄存器应使用所需的延迟时间进行初始化。

14.5 传输初始化

为传输描述符列表分配一个内存区域。软件应确保此内存对齐在段落（16字节）边界上。将传输描述符基地址（TDBAL/TDBAH）寄存器编程为该区域的地址。TDBAL 用于 32 位地址，而 TDBAL 和 TDBAH 都用于 64 位地址。

将传输描述符长度（TDLEN）寄存器设置为描述符环的大小（以字节为单位）。此寄存器必须对齐 128 字节。

传输描述符头和尾（TDH/TDT）寄存器在上电或软件发起的以太网控制器复位后（由硬件）被初始化为0b。软件应向这两个寄存器写入0b以确保这一点。

初始化传输控制寄存器（TCTL）以进行所需操作，包括以下内容：
- 将使能（TCTL.EN）位设置为 1b 以进行正常操作。
- 将填充短数据包（TCTL.PSP）位设置为1b。

- Configure the Collision Threshold (TCTL.CT) to the desired value. Ethernet standard is 10h. This setting only has meaning in half duplex mode.
- Configure the Collision Distance (TCTL.COLD) to its expected value. For full duplex operation, this value should be set to 40h. For gigabit half duplex, this value should be set to 200h. For 10/100 half duplex, this value should be set to 40h.

Program the Transmit IPG (TIPG) register with the following decimal values to get the minimum legal Inter Packet Gap:

	Fiber	**Copper**	**Fiber (82544GC/EI**	**Copper (82544GC/EI**
IPGT	10	10	6	8
IPGR1	10	10	8[a]	8[a]
IPGR2	10	10	6[a]	6[a]

a. Applicable to the **82541xx** and **82547GI/EI**.

- 将碰撞阈值 (TCTL.CT) 设置为所需值。以太网标准是 10h。此设置仅在半双工模式下有效。
- 将碰撞距离 (TCTL.COLD) 设置为其预期值。对于全双工操作，此值应设置为 40h。对于千兆半双工，此值应设置为 200h。对于 10/100 半双工，此值应设置为 40h。

使用以下十进制值编程传输IPG (TIPG) 寄存器以获得最小的合法帧间间隙:

	光纤	铜缆	光纤 (82544GC/EI	铜缆 (82544GC/EI
IPGT	10	10	6	8
IPGR1	10	10	8[a]	8[a]
IPGR2	10	10	6[a]	6[a]

a. 适用于 82541xx 和 82547GI/EI。

Note: IPGR1 and IPGR2 are not needed in full duplex, but are easier to always program to the values shown.

Table 14-1. Signal Descriptions

Signal	Ball	Name and Function
LOS / LINK	A10	**Loss of Signal (TBI) / Link Indication.** Loss of signal (high for lost signal) from the optical transceiver when LINK_MODE equals 11b; active high link indication from PHY in GMII/MII mode.
TX_DATA[9] / TX_ER TX_DATA[8] / TX_EN TX_DATA[7] TX_DATA[6] TX_DATA[5] TX_DATA[4] TX_DATA[3] TX_DATA[2] TX_DATA[1] TX_DATA[0]	C7 D7 E6 B5 E5 C5 E4 C4 D5 D4	**Transmit Data.** TBI: TX_DATA[9:0] for transmit data bus. GMII: TX_DATA[7:0] for transmit data bus. TX_ER forces propagation of transmit errors and is used for carrier extension. TX_EN is asserted to indicate transmission of data on the interface. MII: TX_DATA[3:0] for transmit data bus. TX_ER is not used. TX_EN is used for transmit enable signal.
GTX_CLK	C6	**Transmit Clock.** TBI: 125 MHz transmit clock. GMII: Operates at 125 MHz. MII: Undefined.
EWRAP	E10	**Enable Wrap.** TBI: EWRAP is low in normal operation. When it is high, the SerDes device is forced to transceiver loopback the serialized transmit data to the receiver. This pin is tri-stated during EEPROM read. In order to avoid a floating input in an external SerDes, a weak external pull-down should be connected to this pin. GMII / MII: Not used.
COL	E7	**Collision.** TBI: Undefined. GMII / MII: This signal indicates that a collision was detected on the medium by the PHY. This signal remains asserted while the collision persists. For half-duplex transceivers, this signal indicates simultaneous transmission and reception. This signal is ignored in full-duplex mode. Normal Mode: This signal must be connected to VSS except for test mode.

注意：在全双工模式下，IPGR1 和 IPGR2 不需要，但始终编程为所示值更容易。

表14-1。信号描述

信号	Ball	名称和功能
LOS / LINK	A10	信号丢失 (TBI) / 链路指示。当 LINK_MODE 等于 11b 时，光收发器出现信号丢失（丢失信号为高电平）；在 GMII/MII 模式下，PHY 发出高电平链路指示。
传输数据[9] / TX_ER 传输数据[8] / TX_EN 传输数据[7] 传输数据[6] 传输数据[5] 传输数据[4] 传输数据[3] 传输数据[2] 传输数据[1] 传输数据[0]	C7 D7 E6 B5 E5 C5 E4 C4 D5 D4	传输数据。 TBI: TX_DATA[9:0] for 传输数据总线。 GMII: TX_DATA[7:0] for 传输数据总线。 TX_ER 强制传输错误的传播，并用于 载波扩展。TX_EN 用于指示接口上的 数据传输。 MII: TX_DATA[3:0] for 传输数据总线。 TX_ER 未使用。TX_EN 用于 传输使能信号。
GTX_CLK	C6	传输时钟。 TBI: 125 MHz 传输时钟。 GMII: 运行在 125 MHz。 MII: 未定义。
EWRAP	E10	启用回环。 TBI: EWRAP 在正常操作中为低电平。当其为高电平时，SerDes 设备被强制将串行发送数据通过收发器环回传输到接收器。 在此引脚上，EEPROM 读操作期间处于三态。为了防止外部 SerDes 出现浮动输入，应将一个弱外部下拉连接到此引脚。 GMII / MII: 未使用。
COL	E7	冲突。 TBI: 未定义。 GMII / MII: 该信号指示 PHY 在介质上检测到冲突。当冲突发生时，该信号保持有效。 持续存在。对于半双工收发器，该信号指示同时发送和接收。在全双工模式下，该信号被忽略。 正常模式：该信号必须连接到VSS，测试模式除外。

Signal	Ball	Name and Function
CRS	A6	**Carrier Sense.** TBI: Undefined. GMII / MII: This signal indicates traffic activity on the cable, either incoming or outgoing. This signal is driven by the PHY. CS is not required to transition synchronously with respect to the RX or TX clocks. This signal is ignored in full-duplex mode. Normal Mode: This signal must be connected to VSS except for test mode.
RX_DATA[9] / RX_ER RX_DATA[8] / RX_DV RX_DATA[7] RX_DATA[6] RX_DATA[5] RX_DATA[4] RX_DATA[3] RX_DATA[2] RX_DATA[1] RX_DATA[0]	A9 D10 B9 C9 D9 E9 E8 C8 A7 B7	**Receive Data.** TBI: RX_DATA[9:0] for receive data bus GMII: RX_DATA[7:0] for receive data bus. RX_ER signals a receive error. RX_DV is asserted to indicate data is valid on the interface. MII: RX_DATA[3:0] for receive data bus. RX_ER signals a receive error. RX_DV indicates data is valid on the interface.
RBC0 / RX_CLK	C11	**Receive Clock 0.** TBI: RBC0 is receive clock (62.5 Mbps). GMII: RX_CLK is receive clock (125 Mbps). MII: RX_CLK is receive clock for 100 Mbps operation (25 Mbps) and for 10 Mbps operation (2.5 Mbps).

14.5.1 Signal Interface

The external GMII/MII interface is similar in function to the interface used to communicate between the MAC and internal PHY. As with use of the internal PHY, the external GMII/MII interface supports 10/100/1000 Mbps operation, with both half- and full-duplex operation at 10/100 Mbps, and full-duplex operation at 1000 Mbps. Unlike the communication path to the internal PHY, the external interface does not provide certain additional control/status interfaces for automatic hardware link setup and/or power-management

Table 14-2lists the signals, functions, and pins used to provide this interface.

信号	Ball	名称和功能
CRS	A6	载波侦测。 TBI: 未定义。 GMII / MII: 该信号指示电缆上的流量活动，无论是 入站还是出站。该信号由PHY驱动。CS无需 与RX或TX时钟同步切换。在全双工模式下，该信号被忽略。 正常模式：该信号必须连接到VSS，测试模式除外。
RX_DATA[9] / RX_ER RX_DATA[8] / RX_DV 接收数据[7] 接收数据[6] 接收数据[5] 接收数据[4] 接收数据[3] 接收数据[2] 接收数据[1] 接收数据[0]	A9 D10 B9 C9 D9 E9 E8 C8 A7 B7	接收数据。 TBI: 接收数据[9:0] 用于接收数据总线 GMII: 接收数据[7:0] 用于接收数据总线。 RX_ER 信号表示接收错误。RX_DV 用于指示接口上的数据有效。 MII: RX_DATA[3:0] 用于接收数据总线。 RX_ER信号指示接收错误。RX_DV指示数据在 接口上有效。
RBC0 / RX_CLK	C11	接收时钟0。 TBI: RBC0是接收时钟（62.5 Mbps）。 GMII: RX_CLK 是接收时钟 (125 Mbps)。 MII: RX_CLK 是 100 Mbps 操作的接收时钟 (25 Mbps) 和 10 Mbps 操作的接收时钟 (2.5 Mbps)。

14.5.1 信号接口

外部GMII/MII接口在功能上与用于MAC和内部PHY之间通信的接口相似。与使用内部PHY一样，外部GMII/MII接口支持10/100/1000 Mbps操作，在10/100 Mbps时支持半双工和全双工操作，在1000 Mbps时支持全双工操作。与到内部PHY的通信路径不同，外部接口不提供某些额外的控制/状态接口，用于自动硬件链路设置和/或电源管理

表14-2列出了用于提供此接口的信号、功能和使用引脚。

Table 14-2. Signal Functions

Signal	Function	Pin
GMII (1000 Mbps) Operations		
CRS	Carrier Sense	CRS
COL	Collision Detect	COL
TX_ER	Transmit Code Error	TX_DATA[9]/TX_ER
TX_EN	Transmit Enable	TX_DATA[8]/TX_EN
GTX_CLK	Transmit Data Clock (125 MHz)	GTX_CLK
TX_DATA	Transmit Data	TX_DATA[7:0]
RX_CLK	Receive Data Clock (125 MHz)	RBC0/RX_CLK
RX_DATA	Receive Data	RX_DATA[7:0]
RX_ER	Receive Error	RX_DATA[9]/RX_ER
RX_DV	Receive Data Valid	RX_DATA[8]/RX_DV
LINK	PHY Link Indication	LOS/LINK
MII (10/100 Mbps) Differences		
MTX_CLK	Transmit Data Clock (25/2.5 MHz)	RBC1/MTX_CLK
TX_DATA	Transmit Data	TX_DATA[3:0]
RX_CLK	Receive Data Clock (25/2.5 MHz)	RBC0/RX_CLK
RX_DATA	Receive Data	RX_DATA[3:0]

14.5.2 GMII/MII Features not Supported

Table 14-3 lists the signals and functions not provided by this interface.

表14-2. 信号功能

信号	功能	Pin
GMII（1000 Mbps）操作		
CRS	载波侦测	CRS
COL	碰撞检测	COL
TX_ER	传输码错误	传输数据[9]/TX_ER
TX_EN	传输使能	传输数据[8]/传输使能
GTX_CLK	传输数据时钟（125 MHz）	GTX_CLK
传输数据	传输数据	传输数据[7:0]
接收时钟	接收数据时钟 (125 MHz)	RBC0/接收时钟
接收数据	接收数据	接收数据[7:0]
RX_ER	接收错误	接收数据[9]/RX_ER
RX_DV	接收数据有效	RX_DATA[8]/RX_DV
LINK	物理链路指示	丢失/链路
MII (10/100 Mbps) 差异		
发射时钟	发射数据时钟 (25/2.5 MHz)	RBC1/发射时钟
传输数据	传输数据	传输数据[3:0]
接收时钟	接收数据时钟 (25/2.5 MHz)	RBC0/接收时钟
接收数据	接收数据	接收数据[3:0]

14.5.2 不支持的GMII/MII功能

表14-3列出了此接口未提供的信号和功能。

Table 14-3. Signal Functions Not Supported

Signal	Function	Ramifications
MII Management Interface (PHY Register Access)		
MDC	Management Data Clock	No support/access to MII register set.
MDI/O	Management Data I/O	
Direct PHY Indications to MAC		
FDX	PHY-negotiated full/half duplex indication	Can limit use to specific known duplex setting.
SPD_IND	PHY-negotiated speed (10/100/1000 Mbps)	Can limit use to specific known speed or require use of auto-speed detection.

14.5.3 Avoiding GMII Test Mode(s)

Note that the Ethernet controller contains a set of test modes that use this interface for component manufacturing and/or diagnostic test. To avoid accidental engagement of unexpected test mode(s) when using the external GMII (or TBI), the TEST_GMII[2:0] test pins must remain de-asserted (low) and the TEST_DM_N pin must remain de-asserted (high).

14.5.4 MAC Configuration

The Ethernet controller MAC operates in a GMII/MII mode when operating with the internal PHY; this mode is similar to the GMII/MII mode of the standalone 82543 MAC components and others. In GMII/MII mode, the MAC operates assuming use of a GMII/MII interface communication, variable duplex & speed configuration (unless forced or auto-detected). For the Ethernet controller, to use this external interface as a GMII/MII interface and have the MAC operate in this GMII/MII Mode, the LINK_MODE must be set to 01b.

It is likely that the MAC might be required to be configured in a forced-duplex configuration, as no means is provided (either the MDI/O access or direct PHY-to-MAC signaling) of any duplex configuration that might be negotiated between the attached Ethernet controller/transceiver and its link partner.

The MAC can further be required to be configured in a forced-speed configuration, as no direct speed indication is available via the external interface (compared to the SPD_IND signals provided by the internal PHY). The Auto-Speed detection (ASD) can be potentially useful in automatically calculating and configuring a speed setting based in the interface signals that are provided.

The MAC is unable to provide any access to MII Management registers through the MDIC register, as no explicit MDI/O signals are included in this interface. However, it is possible that software-definable pins (SDP) can be capable of providing the necessary access capability.

表14-3。不支持的信号功能

信号	功能	影响
MII 管理接口 (PHY 寄存器访问)		
MDC	管理数据时钟	不支持/无法访问 MII 寄存器集。
MDI/O	管理数据 I/O	
直接 PHY 指示到 MAC		
FDX	PHY 协商全/半双工指示	可以限制为特定的已知双工设置。
SPD_IND	PHY 协商速度 (10/100/1000 Mbps)	可以限制使用特定已知速度或要求使用自动速度检测。

14.5.3 避免GMII测试模式

注意，以太网控制器包含一组用于组件制造和/或诊断测试的测试模式，这些测试模式使用此接口。在使用外部GMII（或TBI）时，为避免意外激活意外的测试模式，TEST_GMII[2:0] 测试引脚必须保持非激活（低），TEST_DM_N引脚必须保持非激活（高）。

14.5.4 MAC配置

以太网控制器MAC在与内部PHY配合工作时，会以GMII/MII模式运行；此模式与独立82543 MAC组件及其他组件的GMII/MII模式相似。在GMII/MII模式下，MAC假设使用GMII/MII接口通信，可变全双工和速度配置（除非强制或自动检测）。对于以太网控制器，要使用此外部接口作为GMII/MII接口并让MAC在此GMII/MII模式下运行，LINK_MODE必须设置为01b。

MAC可能需要配置为强制全双工模式，因为没有提供任何双工配置手段（无论是MDI/O访问还是直接PHY到MAC信号），而连接的以太网控制器/收发器与其链路伙伴之间可能协商的双工配置。

MAC可能还需要配置为强制速度模式，因为外部接口没有直接的速度指示（与内部PHY提供的SPD_IND信号相比）。自动速度检测(ASD)可以潜在地通过接口信号自动计算和配置速度设置。

MAC无法通过MDIC寄存器提供对MII管理寄存器的任何访问，因为此接口不包含明确的MDI/O信号。但是，软件定义引脚(SDP)可能能够提供必要的访问能力。

14.5.5 Link Setup

The following examples are provided as suggestions for configuring common settings between the MAC and an Ethernet controller attached in the GMII/MII mode.

- MAC duplex and speed settings forced by software based on resolution of PHY
  (CTRL.FRCDPLX = 1b, CTRL.FRCSPD = 1b, CTRL.ASDE = don't care)
  CTRL.FD Set by software based on reading PHY status register after PHY has autonegotiated a successful link-up.
  CTRL.SLU Must be set to 1b by software to enable communications between MAC and PHY
  CTRL.RFCE Must be set by S/W after reading flow control resolution from PHY registers
  CTRL.TFCE - Must be set by S/W after reading flow control resolution from PHY registers
  CTRL.SPEED Set by software based on reading PHY status register after PHY has autonegotiated 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 1b)
  STATUS.SPEED Reflects MAC forced speed setting written in CTRL.SPEED

- MAC duplex setting forced by software based on resolution of PHY; speed auto-detected by MAC
  (CTRL.FRCDPLX = 1b, CTRL.FRCSPD = 0b, CTRL.ASDE = 1b)
  CTRL.FD Set by software based on reading PHY status register after PHY has autonegotiated a successful link-up.
  CTRL.SLU Must be set to 1b by software to enable communications between MAC and PHY
  CTRL.RFCE Must be set by S/W after reading flow control resolution from PHY registers
  CTRL.TFCE .... Must be set by S/W after reading flow control resolution from PHY registers
  CTRL.SPEED Don't care; speed setting is calculated by the MAC based on signals from the PHY after PHY has autonegotiated 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 1b)
  STATUS.SPEED Reflects actual speed setting calculated by MAC ASD function

14.5.5 链路设置

以下示例作为建议，用于配置MAC和连接在GMII/MII模式下的以太网控制器之间的常见设置。

- MAC 上下行和速度设置由软件根据 PHY 的分辨率强制设置
  (CTRL.FRCDPLX = 1b, CTRL.FRCSPD = 1b, CTRL.ASDE = 不用关心)
  CTRL.FD 由软件根据读取PHY状态寄存器后PHY已自动协商成功链路建立。
  CTRL.SLU必须由软件设置为1b以启用MAC和PHY之间的通信
  CTRL.RFCE必须由软件在从PHY寄存器读取流控制分辨率后设置
  CTRL.TFCE - 必须由软件在从PHY寄存器读取流控制分辨率后设置
  CTRL.SPEED由软件根据 PHY 自动协商成功链路建立后读取 PHY 状态寄存器来设置。
  STATUS.FD 反映写入到 CTRL.FD 的 MAC 强制双工设置；STATUS.LU 反映 PHY 提供的链路指示 (LINK)，并由 CTRL.SLU（设置为 1b）进行限定。
  STATUS.SPEED 反映写入到 CTRL.SPEED 的 MAC 强制速度设置。

- MAC 双工设置由软件根据 PHY 的解析结果强制设置；速度由自动检测。MAC
  (CTRL.FRCDPLX = 1b, CTRL.FRCSPD = 0b, CTRL.ASDE = 1b)
  CTRL.FD 由软件根据 PHY 自动协商成功链路建立后读取 PHY 状态寄存器来设置。
  CTRL.SLU 必须由软件设置为 1b 以启用 MAC 和 PHY 之间的通信
  CTRL.RFCE 必须由 S/W 在从 PHY 寄存器读取流控制分辨率后设置
  CTRL.TFCE .... 必须由 S/W 在从 PHY 读取流控制分辨率后设置寄存器
  CTRL.SPEED 不用管；速度设置由MAC根据PHY在自动协商成功建立链路后的信号计算得出 PHY在自动协商成功建立链路后的信号计算得出
  STATUS.FD反映MAC写入CTRL.FD的强制双工设置 STATUS.LU反映PHY的链路指示（LINK），由CTRL.SLU（设置为1b）限定
  STATUS.SPEED反映由MAC ASD功能计算的实际速度设置

- 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 1b 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 by S/W to desired flow-control operation (must match flow-control settings of PHY)
CTRL.TFCE	Must be set by S/W 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 1b (positive link indication LINK from PHY qualified with CTRL.SLU). Note: 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

Note: It is important to note that for the Ethernet controller's link indication (LINK) to be noted by the MAC, the MAC control bit CTRL.SLU must be set to 1b. Normal MAC/PHY speed and duplex configuration are based on observing events on this link indication from the Ethernet controller.

14.6 PHY Initialization (10/100/1000 Mb/s Copper Media)

Software needs to determine the PHY address at which the PHY actually resides. This number can be anywhere from 0 to 31.The PHY address is programmable. Board designers can then choose at what PHY address the PHY resides. Software needs to identify the PHY address so that the PHY can be accessed successfully.

To accomplish read and write access to any of the PHY registers, software must program the MDI Control Register (MDIC) with the appropriate data. A PHY is reset at power-up and is enabled to Auto-Negotiate by default. Typically in most environments, by the time the software driver is loaded, the Auto-Negotiation process has completed. However, the PHY might or might not advertise the appropriate capabilities desired by the design. In this instance, it is up to the software to insure that the PHY registers are set up properly to advertise the appropriate Ethernet controller capabilities. For example, by default the Ethernet controller advertises no flow control capabilities in its Auto-Negotiation Advertisement Register (MII Register 4). In order to advertise TX and/or RX Pause capabilities, this register must be modified and Auto-Negotiation re-started to advertise these capabilities to the link partner.

The MII Status Register (PHY Register 1) should be used to check link status.

Software can also force the speed/duplex of a PHY via MII/GMII register access. Note that forcing gigabit speed in a copper environment is not allowed per IEEE specification. Only 10/100 speed and duplex should be forced in the PHY.

- MAC/PHY全双工和速度设置均由软件强制（完全强制链路设置）
  (CTRL.FRCDPLX = 1b, CTRL.FRCSPD = 1b, CTRL.SLU = 1b)

CTRL.FD.由软件设置为所需的 全双工/半双工操作（必须与 PHY 的双工设置匹配）

CTRL.SLU................必须由软件设置为1b以启用MAC和PHY之间的通信。PHY也必须强制/配置为向MAC指示正向链路指示（LINK）

CTRL.RFCE ....必须由软件设置为所需的流控操作（必须与PHY的流控设置匹配）

CTRL.TFCE ....必须由软件设置为所需的流控操作（必须与PHY的流控设置匹配）

CTRL.SPEED ....由软件设置为所需链路速度（必须与PHY的速度设置匹配）

STATUS.FD ....反映软件写入CTRL.FD的MAC双工设置
STATUS.LU..............反映1b（来自PHY且经CTRL.SLU合格的物理链路指示LINK）。注意：由于CTRL.SLU和PHY链路指示LINK均被强制，此位设置并不能保证链路操作已真正建立。

STATUS.SPEED........Reflects MAC forced speed setting written in CTRL.SPEED

注意：需要注意的是，要让MAC能够注意到以太网控制器的链路指示（LINK），MAC控制位CTRL.SLU必须设置为1b。正常的MAC/PHY速度和全双工配置是基于观察以太网控制器在此链路指示上发生的事件。

14.6 PHY 初始化（10/100/1000 Mbps 铜缆媒体）

软件需要确定PHY实际所在的PHY地址。该地址范围可以从0到31。PHY地址是可编程的。板级设计者可以随后选择PHY所在的PHY地址。软件需要识别PHY地址，以便成功访问PHY。

要完成对任何 PHY 寄存器的读写访问，软件必须使用适当的数据编程 MDI 控制寄存器 (MDIC)。PHY 在上电时被复位，并默认启用自动协商。在大多数环境中，当软件驱动加载时，自动协商过程通常已完成。然而，PHY 可能会或可能不会宣传设计所需的相关功能。在这种情况下，需要由软件确保 PHY 寄存器正确设置以宣传适当的以太网控制器功能。例如，默认情况下，以太网控制器在其自动协商通告寄存器 (MII 寄存器 4) 中不宣传任何流控制功能。为了宣传 TX 和/或 RX 暂停功能，必须修改此寄存器并重新启动自动协商，以向链路伙伴宣传这些功能。

应使用 MII 状态寄存器 (PHY 寄存器 1) 来检查链路状态。

软件也可以通过 MII/GMII 寄存器访问来强制 PHY 的速度/全双工。请注意，根据 IEEE 规范，在铜缆环境中不允许强制千兆速度。仅在 PHY 中应强制 10/100 速度和全双工。

Once link is achieved by the PHY, software is notified when a Link Status Change (LSC) interrupt is generated by the Ethernet controller. This only occurs if software enabled the LSC bit in the Interrupt Mask Set/Read (MS) Register.

14.7 Reset Operation

The following reset signals affect the Ethernet controller in different ways. RST# is the only external signal. Other reset events are asserted by performing slave writes to specific bits in the control registers.

Values indicated as "?" imply the default value is either unknown or is read from the EEPROM.

Note: In situations where the TX block is reset, the TX data lines are forced to all 0b's. This causes a substantial number of symbol errors to be detected by the link partner. In TBI mode (**82544GC/EI**)/internal SerDes (**82546GB/EB** and **82545GM/EM**), if the duration is long enough, the link partner can restart the Auto-Negotiation process by sending "break-link" (/C/ codes with the configuration register value set to all 0b's).

LAN_PWR_GOOD:

Deasserting LAN_PWR_GOOD resets all resettable registers in the Ethernet controller. The signal is level-sensitive, and the Ethernet controller is held in reset until LAN_PWR_GOOD is asserted. While asserted, all PCI signals are forced to a high impedance state.

General Registers:	Reset to power-on values.
Interrupt Registers:	Reset to power-on values.
Receive Registers:	Reset to power-on values (exceptions are the RAH/RAL, MTA, VFTA and RDBAH/RDBAL registers, which are not reset to any preset value. The valid bit of the RAH register is cleared).
Transmit Registers:	Reset to power-on values (exceptions are the TDBAH/TDBAL registers, which are not reset to any preset value).
Statistics Registers:	Reset to power-on values.
Wakeup Registers:	The WUC (except for the *PME_En* and *PME_Status* bits if AUX_POWER = 1b), WUFC, IPAV, and FFLT registers are reset to their default value.
Diagnostic Registers:	Reset to power-on values (exception is the PBM memory, which is not reset to any preset value).
PCI Config Space:	Context Lost; requires initialization.
PHY:	RST# is asserted to reset the PHY while LAN_PWR_GOOD is deasserted.

In addition, the Ethernet controller automatically reads certain values from the EEPROM and configures itself to use those EEPROM settings.

一旦 PHY 实现了链路，当以太网控制器生成链路状态改变（LSC）中断时，软件会收到通知。这仅发生在软件在中断屏蔽设置/读取（MS）寄存器中启用了 LSC 位的情况下。

14.7 复位操作

以下复位信号以不同方式影响以太网控制器。RST# 是唯一的外部信号。其他复位事件是通过向控制寄存器中的特定位执行从设备写入来断言的。

标有“？ ”的值表示默认值要么未知，要么是从 EEPROM 中读取的。

注意：在 TX 块复位的情况下，TX 数据线被强制设置为全 0b。这会导致链路伙伴检测到大量符号错误。在 TBI 模式（82544GC/ EI）/内部 SerDes（82546GB/EB 和 82545GM/EM）中，如果持续时间足够长，链路伙伴可以通过发送“断开链路”（/C/ 代码，配置寄存器值设置为全 0b）来重新启动自动协商过程。

LAN_PWR_GOOD:

撤销LAN_PWR_GOOD会重置以太网控制器中所有可重置的寄存器。该信号是电平敏感的，以太网控制器会保持在上电复位状态，直到LAN_PWR_GOOD被设置。在设置期间，所有PCI信号都会被强制置为高阻抗状态。

通用寄存器:	重置为上电值。
中断寄存器:	重置为上电值。
接收寄存器:	重置为上电值（例外是RAH/RAL、MTA、VFTA和RDBAH/RDBAL寄存器，它们不重置为任何预设值。RAH寄存器的有效位被清除）。
发送寄存器:	重置为上电值（例外是TDBAH/TDBAL寄存器，它们不重置为任何预设值）。这些，没有被重置为任何预设值)。
统计寄存器:	重置为上电值。
唤醒寄存器:	WUC（如果AUX_POWER = 1b，则不包括PME_En和PME_Status位），WUFC、IPAV和FFLT寄存器被重置为其默认值。
诊断寄存器:	重置为上电值（PBM内存除外，它不会被重置为任何预设值）。重置为任何预设值)。
PCI配置空间:	上下文丢失；需要初始化。
PHY:	RST#被置位以复位PHY，而LAN_PWR_GOOD被释放。

此外，以太网控制器会自动从EEPROM中读取某些值，并配置自身以使用这些EEPROM设置。

RST#:

When asserted, all PCI signals are forced to a high impedance state. Upon deassertion, the Ethernet controller's internal registers, excluding the following exceptions, are reset.

General Registers:	Reset to power-on values.
Interrupt Registers:	Reset to power-on values.
Receive Registers:	Reset to power-on values (exceptions are the RAH/RAL, MTA, VFTA and RDBAH/RDBAL registers, which are not reset to any preset value. The valid bit of the RAH register is cleared).
Transmit Registers:	Reset to power-on values (exceptions are the TDBAH/TDBAL, and TIPG registers, which is not reset to any preset value).
Statistics Registers:	Reset to power-on values.
Wakeup Registers:	The WUC (except for the *PME_En* and *PME_Status* bits if AUX_POWER = 1b), WUFC, IPAV, and FFLT registers are reset to their default value.
Diagnostic Registers:	Reset to power-on values (exception is the PBM memory, which is not reset to any preset value).
PCI Config Space:	Context Lost; requires initialization. If AUX_POWER = 1b then the *PME_En* and *PME_Status* bits of the Power Management Control/Status Register are preserved.
PHY:	RST# is asserted for 400 ns after deassertion of RST#.

Asserting RST# puts the Ethernet controller into the "Dr" Power Management state. See Section 6.3.1.1 for details on the power states, and Section 6.3.2.4 for reset related timing.

Deasserting RST# also causes the EEPROM to be re-read and the registers that get values from the EEPROM to be re-loaded.

Global Reset:

Bit 26 of the Device Control Register (CTRL.RST) performs an Ethernet controller reset of all functions to their equivalent power on state similar to asserting RST#, except that the state of the PCI core and PCI configuration space is not affected.

General Registers:	Reset to power-on values.
Interrupt Registers:	Reset to power-on values.
Receive Registers:	Reset to power-on values (exceptions are the RAH/RAL, MTA, VFTA and RDBAH/RDBAL registers, which are not reset to any preset value. The valid bit of the RAH register is cleared).
Transmit Registers:	Reset to power-on values (exceptions are the TDBAH/TDBAL, and TIPG registers).
Statistics Registers:	Reset to power-on values.
Wakeup Registers:	The WUC (except for the *PME_En* and *PME_Status* bits), WUFC, IPAV, and FFLT registers are reset to their default value.
Diagnostic Registers:	Reset to power-on values (exception is the PBM memory, which is not reset to any preset value).
PCI Config Space:	No Change.
PHY:	No effect.

RST#:

当被置位时，所有PCI信号都会被强制置为高阻抗状态。当被复位时，以太网控制器的内部寄存器（除以下例外外）会被重置。

通用寄存器:	重置为上电值。
中断寄存器:	重置为上电值。
接收寄存器:	重置为上电值（例外是RAH/RAL、MTA、VFTA和RDBAH/RDBAL寄存器，它们不重置为任何预设值。RAH寄存器的有效位被清除）。
发送寄存器:	重置为上电值（例外是TDBAH/TDBAL、和TIPG寄存器，它们不重置为任何预设值）。
统计寄存器:	重置为上电值。
唤醒寄存器:	WUC（若AUX_POWER = 1b，则PME_En和PME_Status位除外），WUFC、IPAV和FFLT寄存器被重置为其默认值。
诊断寄存器:	重置为上电值（PBM内存除外，它不会被重置为任何预设值）。
PCI配置空间:	上下文丢失；需要初始化。如果AUX_POWER = 1b，则电源管理控制/状态寄存器的PME_En和PME_Status位被保留。
PHY:	撤销RST#后，RST#会保持 asserted 状态 400 ns。

设置RST#会将以太网控制器置于“Dr”电源管理状态。有关电源状态的详细信息，请参阅6.3.1.1节，有关复位相关时序的详细信息，请参阅6.3.2.4节。

撤销RST#还会导致EEPROM重新读取，以及从EEPROM获取值的寄存器重新加载。

全局复位:

设备控制寄存器 (CTRL.RST) 的第 26 位执行所有功能的以太网控制器复位，使其恢复到类似于断言 RST# 的开机状态，但 PCI 核心和 PCI 配置空间的状态不受影响。

通用寄存器:	重置为上电值。
中断寄存器:	重置为上电值。
接收寄存器:	重置为上电值（例外情况是RAH/RAL、MTA、VFTA和RDBAH/RDBAL寄存器，它们不会被重置为任何预设值。RAH寄存器的有效位被清除）。
发送寄存器:	重置为上电值（TDBAH/TDBAL和TIPG寄存器除外）。
统计寄存器:	重置为上电值。
唤醒寄存器:	WUC（PME_En和PME_Status位除外）、WUFC、IPAV和FFLT寄存器被重置为其默认值。
诊断寄存器:	重置为上电值（例外是PBM内存，它不会被重置为任何预设值）。
PCI配置空间:	无变化。
PHY:	无影响。

Default values for certain bits of the Device Control Register must be read out of the EEPROM and appropriately set by software if an EEPROM is used.

Global Reset does NOT affect the direction of the software programmable pins.

Link_Reset:

When LRST (bit 3 of the Device Control register) is written as a logic 1b, the Ethernet controller is forced into a link reset state. When LRST is set to 1b the Auto-Negotiation function is disabled. The Auto-Negotiation logic is initiated/restarted when LRST is transitions to 0b. A link reset is only relevant in TBI mode/internal SerDes (not applicable to the **82540EP/EM**, **82541xx** and **82547GI/EI**).

The transmitter sends /C/ ordered_sets when LRST is asserted.

General Registers:	No change.
Interrupt Registers:	No change.
Receive Registers:	The RXCW register is cleared.
Transmit Registers:	No change.
Statistics Registers:	No change.
Wakeup Registers:	No change.
Diagnostic Registers:	No change.
PHY:	No effect.

EE_RST (Extended Device Control Register):

EEPROM reset bit. Initiates a “reset-like” event to the EEPROM function that causes the EEPROM to be read again. Control registers bits are not affected other than those read from the EEPROM.

PHY_RST (Device Control Register):

PHY reset bit in the Device Control Register. By writing a 1b to this bit the software forces the assertion of an internal signal output to reset the PHY device without accessing the PHY registers through the MII management interface (MDI/O & MDC). Internal states of the Ethernet controller are not impacted. To release the PHY reset the software must write a 0b to the bit.

In situations where the Ethernet controller is reset using the software reset CTRL.RST, the TX data lines are forced to all 0b’s. This causes a substantial number of symbol errors to be detected by the link partner. In TBI mode/internal SerDes, if the duration is long enough, the link partner can restart the Auto-Negotiation process by sending “break-link” (/C/ codes with the config register value set to all 0b’s).

Some registers mentioned above within the Ethernet controller are treated specially. The RAH/RAL[n], MTA[n], VFTA[n], WUPM[n], FFMT[n], FFVT[n], TDBAH/TDBAL, and RDBAH/RDBAL registers have no default value and 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 Ethernet controller. Bit 31, the valid bit, of the RAH[n] registers is the exception and is reset with the LAN_PWR_GOOD and RST# and software reset (CTRL.RST) bit.

设备控制寄存器某些位的默认值必须从 EEPROM 中读出，并在使用 EEPROM 时由软件适当设置。

全局复位不会影响软件可编程引脚的方向。

Link_Reset：

当LRST（设备控制寄存器的位3）被写入逻辑1b时，以太网控制器会被强制进入链路复位状态。当LRST设置为1b时，自动协商功能会被禁用。当LRST从1b转换到0b时，自动协商逻辑会被启动/重启。链路复位仅在TBI模式/内部SerDes中相关（不适用于82540EP/EM、82541xx和82547GI/EI）。

当LRST被置位时，发射器发送/C/有序集。

通用寄存器：	无变化。
中断寄存器：	无变化。
接收寄存器：	RXCW寄存器被清除。
发送寄存器：	无变化。
统计寄存器：	无变化。
唤醒寄存器：	无变化。
诊断寄存器：	无变化。
PHY:	无影响。

EE_RST（扩展设备控制寄存器）：

EEPROM复位位。向EEPROM功能发起一个“复位类”事件，导致EEPROM被重新读取。控制寄存器位除从EEPROM读取的外，均不受影响。

PHY_RST（设备控制寄存器）：

设备控制寄存器中的PHY重置位。通过向该位写入1b，软件强制内部信号输出以重置PHY设备，而无需通过MII管理接口（MDI/O & MDC）访问PHY寄存器。以太网控制器内部状态不受影响。要释放PHY复位，软件必须将该位写入0b。

在以太网控制器使用软件复位CTRL.RST重置的情况下，TX数据线被强制设置为所有0b。这会导致链路伙伴检测到大量符号错误。在TBI模式/内部SerDes中，如果持续时间足够长，链路伙伴可以通过发送“断开链路”（/C/代码，配置寄存器值设置为所有0b）来重新启动自动协商过程。

上述以太网控制器中提到的某些寄存器被特殊处理。RAH/RAL[n], MTA[n], VFTA[n], WUPM[n], FFMT[n], FFVT[n], TDBAH/TDBAL， 以及RDBAH/RDBAL寄存器没有默认值，如果与寄存器相关的功能被启用，则必须由软件进行编程。一旦编程，只要以太网控制器通电，其值就会在所有复位过程中保持不变。RAH[n] 寄存器的位31，即有效位，是例外，它会在LAN_PWR_GOOD、RST#和软件复位（CTRL.RST）位时被复位。

Driver accessible Wakeup Status registers are excluded from all resets except for LAN_PWR_GOOD. This includes:

- Wakeup Status Register.
- Wakeup Packet Length.
- Wakeup Packet Memory.

Finally, the “Wakeup Context” as defined in the PCI Bus Power Management Interface Specification is reset on LAN_PWR_GOOD, and is also reset on the deassertion of RST# if AUX_POWER = 0b. This includes:

- PME_En bit of the Power Management Control/Status Register (PMCSR).
- PME_Status bit of the Power Management Control/Status Register (PMCSR).

The shadow copies of these bits in the Wakeup Control Register are treated identically.

14.8 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 D0$_{active}$ power state (when internal registers become accessible, as enabled by setting the Memory Access Enable of the PCI Command register), and is guaranteed to be completed within 1 µs of this transition. Access to statistics registers prior to this interval can return indeterminate values. Given typical system boot times and the software driver’s Ethernet controller initialization routines, no initialization of these registers through software should be necessary.

驱动可访问唤醒状态寄存器除LAN_PWR_GOOD外，会排除所有复位。这包括：

- 唤醒状态寄存器。
- 唤醒数据包长度。
- 唤醒数据包内存。

最后，根据PCI总线电源管理接口规范中定义的“唤醒上下文”，在LAN_PWR_GOOD时会被复位，如果AUX_POWER = 0b的RST#被撤销时也会被复位。这包括：

- 电源管理控制/状态寄存器（PMCSR）的PME_En位。
- 电源管理控制/状态寄存器（PMCSR）的PME_Status位。

唤醒控制寄存器中这些位的影子副本被同等对待。

14.8 统计初始化

统计寄存器由硬件初始化为每个特定寄存器描述中详细指定的值。这些寄存器的初始化在转换到D0活动电源状态时开始（此时内部寄存器变得可访问，如通过设置PCI命令寄存器的内存访问使能位所启用），并保证在此转换后的1 µs内完成。在此时间间隔之前访问统计寄存器可能会返回不确定值。考虑到典型的系统启动时间和软件驱动程序的以太网控制器初始化例程，不应需要通过软件对这些寄存器进行初始化。