Introduction to TCP/IP Offload Engine (TOE)

Version 1.0, April 2002

Authored By:

Eric Yeh, Hewlett Packard

Herman Chao, QLogic Corp.

Venu Mannem, Adaptec, Inc.

Joe Gervais, Alacritech

Bradley Booth, Intel

1300 Bristol Street North, Suite 160

Newport Beach, CA 92660

Tel: 949.250.7155\

Fax: 949.250.7159

www.10gea.org

Introduction

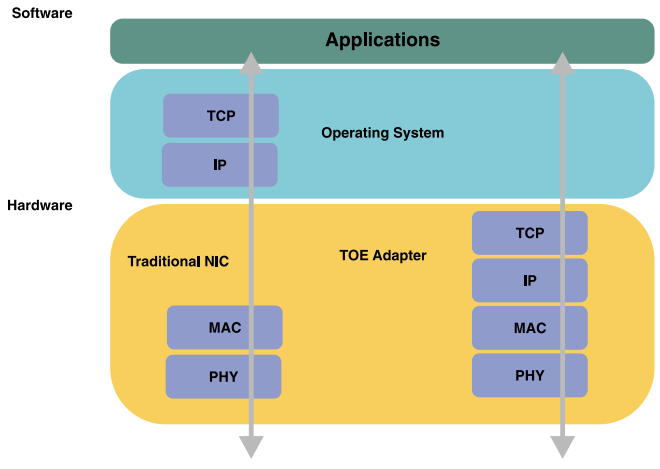
Ethernet has become the most popular networking protocol for local area networks (LANs). It is inexpensive and offers plug-and-play implementation. As networking has grown in popularity, Ethernet network link speeds have also been increasing. The growth of Ethernet from 10 Mbit/s to 10 Gbit/s has surpassed the growth of microprocessor performance in mainstream servers and computers. A fundamental obstacle to improving network performance is that servers were designed for computing rather than input and output (I/O). The Internet revolution has drastically changed server requirements, and I/O is becoming a major bottleneck in delivering high-speed computing. The main reason for the bottleneck is the TCP/IP stack being processed at a rate less than the network speed.

TCP/IP offload Engine (TOE) is one of the technologies that can reduce the amount of TCP/IP processing handled by micro-processor and server I/O subsystem, and thus ease server networking bottleneck. Deployment of TCP/IP offload in conjunction with high-speed Ethernet technologies enables applications to take full advantage of the networking capabilities.

Ethernet-based IP storage is a large opportunity for 10 Gigabit Ethernet. The major issue of IP storage is the high TCP/IP processing overhead, which constrains servers to performance levels that are unacceptable for block storage transport. Only TCP/IP offload technology can provide this level of performance. iSCSI is a good example of using TCP/IP offload to achieve high-performance IP storage. If you are interested in understanding more about iSCSI, please refer to the 10GEA iSCSI white paper.

The processing of TCP/IP over Ethernet is traditionally accomplished by software running on the central processor, CPU or microprocessor, of the server. As network connections scale beyond Gigabit Ethernet speeds, the CPU becomes burdened with the large amount of TCP/IP protocol processing required. Reassembling out-of-order packets, resource-intensive memory copies, and interrupts put a tremendous load on the host CPU. In high-speed networks, the CPU has to dedicate more processing to handle the network traffic than to the applications it is running. A rough estimate of the CPU processing required to handle a given Ethernet link speed is, for every one bit per second of network data processed, one hertz of CPU processing is required. (This general rule of thumb was first stated by PC Magazine in the mid 1990's, and is still used as a rule of thumb by computer companies today.) For instance, a 20 GHz CPU running at full utilization would be needed to drive a 10 Gbps Ethernet network link. TOE is emerging as a solution to limit the processing required by CPUs for networking links. A TOE may be embedded in a network interface card, NIC, or host bus adapter, HBA.

TCP/IP Offload Engine, TOE



TOE Technology and Various TOE Implementations

The basic idea of a TOE is to offload the processing of TCP/IP protocols from the host processor to the hardware on the adapter or in the system. Under the umbrella of the TCP/IP protocol stack, there are multiple protocols like TCP, IP, UDP, ICMP and others. Most end-user applications use TCP and IP protocols. A TOE can be implemented with a network processor and firmware, specialized ASICs, or a combination of both. TOE not only offload CPU processing to increase application performance, but enable Ethernet to address new markets, such as iSCSI storage area networks (SANs) and high-performance network-attached storage (NAS) applications. All TOE implementations available in the market are concentrating on offloading TCP and IP processing. In the following sections, we will explore different implementations of TOE.

As a precursor to TCP offloading, some operating systems supported features to offload some computing intensive features from the host to the underlying adapters. IP checksum offload implemented in some server network adapters is an example of a simple offload. But as Ethernet speeds increase beyond 100 Mbit/s as end-user performance expectations grow, the need for further protocol processing offloading has become a clear requirement.

TOE can be implemented in different ways depending on end-user preference between various factors like deployment flexibility and performance. Traditionally, processor-based solutions provided the flexibility to implement new features, while ASIC solutions provided performance but were not flexible enough to add new features. Today, there is a new breed of performance optimized ASICs utilizing multiple processing engines to provide ASIC-like performance with more deployment flexibility. In the following section, we will discuss various implementations.

In a discrete implementation, TOE is implemented using off-the-shelf components like a network processor or microprocessor running a real time operating system (RTOS) and a MAC/PHY. The protocol processing from the host CPU is offloaded to the protocol stack in the RTOS. This implementation not only offloads the TCP stack, but also offloads any other protocols that are supported by the embedded stack in RTOS, provided proper hooks are supplied by the hardware to offload those protocols from the host. The advantage of this implementation is the flexibility of the solution and wide availability of the components; however, the scalability to 10 Gigabit Ethernet and beyond is coming under scrutiny.

In an ASIC-based implementation, TCP/IP processing is offloaded to performance-optimized hardware. ASIC implementations are customized for TCP/IP protocol offload, offering better performance than discrete implementations. There is general agreement that the advantages of this implementation are performance and scalability, but at the expense of flexibility.

There are implementations that will try to take advantage of both the processor-based implementation and the ASIC-based implementation. The intent is to be able to provide scalability and flexibility while maintaining performance.

TCP protocol processing can be split into different phases:

- Connection establishment
- Data transmission/reception
- Connection tear-down (once the data is transmitted and received)
- Error handling

In environments where dropped packets are infrequent and connections are maintained for long periods of time, the bulk of the overhead in TCP/IP is in data transmission and reception. The offloading of this overhead is commonly referred to as data path offloading. Data path offloading eliminates the TCP/IP overhead in the data transmission/reception phase. The host stack maintains responsibility for the remaining phases (i.e. connection establishment, closing and error handling).

Full offloading executes all phases of the TCP stack in hardware. With full offload, a TOE relieves the host not only from processing data, but also from connection management tasks. In environments where connection management or error handling are intensive tasks, there is a definitive advantage to full offload solutions.

Depending on the end-user application, data path offload or full offload may be equally effective in lowering host CPU utilization and increasing the data throughput. By offloading TCP/IP protocol processing, the host processor can be relieved from the computing intensive protocol stacks and is free to focus its CPU cycles on applications. With 10 Gigabit Ethernet, it is clear that TOE offers substantial benefits to system performance in a variety of application environments.

Performance with TCP Offload

It is still early to describe definitive performance improvements or metrics of TCP/IP offload on 10 Gigabit Ethernet. However, we can evaluate TOE on three basic performance metrics: throughput, CPU utilization, and latency.

Throughput has always been used as a key indicator for network performance. Throughput measurement again is one of the performance differentiators among various TOE and non-TOE adapters.

CPU utilization is also an essential performance measurement for servers. Traditional adapters typically consume 1 hertz of processor for each bit of TCP/IP data moved. There are two major components that drive this baseline – the amount of data copied within the protocol stack as data is moved from network buffers to user buffers and back, and the number of transactions to move the data. With a traditional adapter using standard 1,500 byte frames, there are typically three transactions for every 3,000 bytes of data – two packets generated or received, and an acknowledgement back in the other direction.

TOE strives to address the data copy and transaction problems by interfacing to the system above the transport layer with a session layer interface. This usually entails a socket interface for sockets-based systems. Such an interface takes advantage of the fact that applications typically read and write in large blocks, typically tens of kilobytes or more. Emails with large attachments, Web pages with lots of multimedia elements, and servers serving up multi-megabyte applications define the typical traffic mix.

In large application reads and writes, the workload on the host system is drastically reduced. A typical 32KB read or write would be more than 30 transactions – over 20 data packets and 10 ACKs using conventional adapter technology. With a TOE adapter in the system, it dramatically reduces the number of transactions between the host and the TOE adapter, with all packet processing and ACK processing local to the TOE adapter. The data is moved to and from application buffers by hardware DMA engines.

Latency is another performance metric impacted by the use of TCP offload. In simple terms, the decrease in the number of transactions across the system I/O bus and memory bus helps reduce the wait time for the bus and decreases latency. By performing the TCP/IP processing in the adapter, the number of transactions across the system I/O bus and memory bus are reduced. With TOE, network adapters will be able to respond faster, therefore enabling a faster end-to-end communication.

Challenges

The implementation and market adoption of TOE faces some key challenges. Currently there is no standard driver interface for major operating systems and TOE adapters. The wide adoption of Ethernet adapters was fueled by the standardization of the TCP/IP networking software driver interface for major operating systems. Drivers for TOE implementations are interfacing to operating systems differently and in a variety of proprietary ways. Major operating systems are working on standard interfaces to TOE to facilitate simple support of TOE solutions.

Another challenge is the market expects network adapters to be inexpensive. TOE adapter implementations that contain an embedded processor or multiple specialized ASICs may be too expensive for the market to bear. Consequently, companies are working on integrated ASIC solutions to reduce cost. Single chip TOE implementations require putting a protocol typically handled in software, TCP/IP, into fixed function silicon. Implementing TCP/IP completely in hardware poses a significant technical challenge.

Beyond TOE specific challenges, terminating a 10 Gigabit Ethernet link in a server places strain on the overall server architecture. A balanced system is required to take full advantage of a high performance network connection. Memory bandwidth and bus bandwidth are just two of the most critical system dimensions that need to be monitored as systems adopt 10 Gigabit Ethernet. A typical server running Windows 2000 today uses PCI running at 64 bits / 66 MHz, with an effective bandwidth of around 350 MB/s. A server using PCI-X at 133 MHz has an effective bus bandwidth of around 800 MB/s. To fill a 10 Gigabit Ethernet link, on the order of 1.25GB/s of bandwidth is required in each direction, or a total of 2.5 GB/s, more than three times today's PCI-X implementations, and nearly seven times the bandwidth of a PCI 64/66 bus.

Typical memory subsystems will also need to advance to take full advantage of the higher speed networks. Many standard servers use SDRAM in a 128-bit wide mode, running at 133 MHz. This provides about 2 GB/s of memory bandwidth. Memory bandwidth needs to be several times faster than the I/O bandwidth, since bandwidth is required both to send and receive data on the network and to move that data to and from the CPU and/or to and from storage. Memory bandwidth on the order of 4 times the desired I/O bandwidth, or 10 Gigabytes per second per network interface wouldn't be an unreasonable requirement.

Applications

Some of the uses of TOE have been described above. If you look at what a TOE brings to the overall 10 Gigabit Ethernet systems, its focus is around the offloading the TCP/IP overhead from the processor. Given that TCP/IP is almost synonymous with Ethernet, the ability to remove or reduce the CPU requirements for handling TCP/IP messaging will permit Ethernet based technologies to venture into new areas. One such area as described above is the transportation of iSCSI or any IP storage traffic. TOE helps decrease the load on the host CPU in either a target (the device with the data) or an initiator (the device that wants the data).

IP storage and general Enterprise data traffic are not the only places that TCP/IP offloading can add benefit. The use of Ethernet technologies for backplanes (i.e. XAUI in 10 Gigabit Ethernet) can use TCP/IP as a means to provide network and transport layer management. TOE can assist this by providing TCP/IP functionality without the need of a CPU or with a lower performance, lower cost CPU or processor. Other possible applications are in thin clients where the processing power of the client is swamped with the processing of TCP/IP. Freeing up the clients processing power will permit it to focus its limited processing power on other tasks.

These are just some general applications that a TOE function could be used to provide assistance. As the technology grows and matures, other applications and possibly other markets for TCP/IP, TOE and Ethernet will emerge.