TECHNICAL WHITE PAPER

Integrated Network Acceleration Features

of Intel® I/O Acceleration Technology and Microsoft® Windows Server® 2008

TABLE OF CONTENTS

Ever-increasing network-traffic volumes make it mandatory for IT organizations to address I/O bottlenecks. This is especially true for data centers implementing server consolidation and virtualization, both of which can cause further increases in I/O bandwidth demands and traffic burdens on servers.

This white paper examines the key contributors to I/O bottlenecks. It then discusses how the tightly integrated features of Intel® I/O Acceleration Technology (Intel® I/OAT) and Microsoft® Windows Server® 2008 significantly reduce I/O bottlenecks, allowing network data to move more efficiently through multi-core processor-based servers. The fully integrated, system-wide solution provided by Intel I/OAT is optimized for multi-core processors and provides network acceleration that scales seamlessly across multiple Ethernet ports. It also provides a safe and flexible choice for IT managers due to its tight integration into popular operating systems such as Microsoft Windows Server 2008.

intel® | Microsoft®

Introduction

The demands for network throughput seem to grow relentlessly. Much of this growth is in terms of demands on Information Technology (IT) managers to support expanding network-based business applications, network-based storage and backup solutions, video conferencing, media streaming, and other high-value solutions. Additionally, new networking approaches, such as data center consolidation and virtualization, bring new challenges in handling increased network traffic.

Widespread deployment of Gigabit Ethernet (GbE), use of adapter teaming, and now increasing use of 10 Gigabit Ethernet (10GbE) have markedly increased bandwidth to accommodate network-traffic growth. However, increased network speed also increases the load on server resources needed to service the network connections. This leads to input/output (I/O) bottlenecks when data-movement demands push CPU utilization excessively high. These I/O bottlenecks have emerged as a significant challenge that keeps organizations from gaining full value for their server infrastructure investments.

To address these bottlenecks, Intel developed Intel® I/O Acceleration Technology[1] (Intel® I/OAT), which is included in PCI Express* Intel® Server Adapters and in Multi-Core Intel® Xeon® processor-based platforms. Similarly, Microsoft has developed complementary I/O acceleration technologies in Windows Server® 2008. Intel I/OAT along with Windows Server 2008 moves network data more efficiently through multi-core processor-based servers for fast, scalable, and reliable networking. In addition to providing network and data acceleration across the platform, Intel I/OAT also scales seamlessly across multiple Ethernet ports to improve application response times.

Tight integration of Intel I/OAT with Microsoft Windows Server 2008 makes these I/O gains possible. In particular, the Microsoft Windows 2008 feature of NetDMA recognizes the presence of Intel® QuickData Technology[2] (one of the features of Intel I/OAT). Together, they largely free the CPU from memory transfers associated with transmission control protocol/Internet protocol (TCP/IP) packet processing. Additionally, receive-side scaling (RSS) allows traffic load balancing across multiple processor cores. Still other features in Intel I/OAT – which include extended message-signaled interrupts (MSI-X), low-latency interrupts, stateless offloading, and others – provide additional acceleration from the server adapter through the chipset and CPU to the operating system and application. The result is a system-wide acceleration solution that moves data more efficiently through the server platform. Moreover, IT organizations gain immediate benefit from Intel I/OAT and Windows Server 2008 because neither requires changes to existing applications or network management tools.

To gain a deeper understanding of the features and benefits of Intel I/OAT and its integration with Microsoft Windows Server 2008, it is necessary to first understand the nature and causes of the I/O bottlenecks being resolved. Then, with the problems fully defined, the solutions provided by Intel I/OAT and Windows Server 2008 become more apparent.

What Causes I/O Bottlenecks?

TCP/IP packets carry the network traffic. With increasing application demands and available network bandwidth, both the volume and speed of traffic increases. This results in increasing TCP/IP processing burdens on the system. Anything within the packet-processing path that cannot carry the increased burden, or impedes the flow, results in a potential bottleneck.

Historically, the TCP/IP network stack took much of the blame for being the primary bottleneck. This is because the TCP/IP stack has undergone little technological advancement since the 1980s. As a result, increasing traffic causes significant CPU overhead since the CPU must process the non-optimized TCP protocol. One proposed solution to this is to offload TCP/IP processing to the network interface card (NIC) with a TCP offload engine (TOE).

However, Intel researchers felt that there was more to the problem than just the TCP/IP stack. To support this, Intel research and development teams performed detailed examinations of the entire packet-processing path (see Figure 1).

At a high level, once the data packet arrives at the system, the payload is stored in the system memory. When an application requests the data, the processor fetches it from the system memory and copies it to the user memory, placing the data in its cache. The data is now available to the application, satisfying the application request.

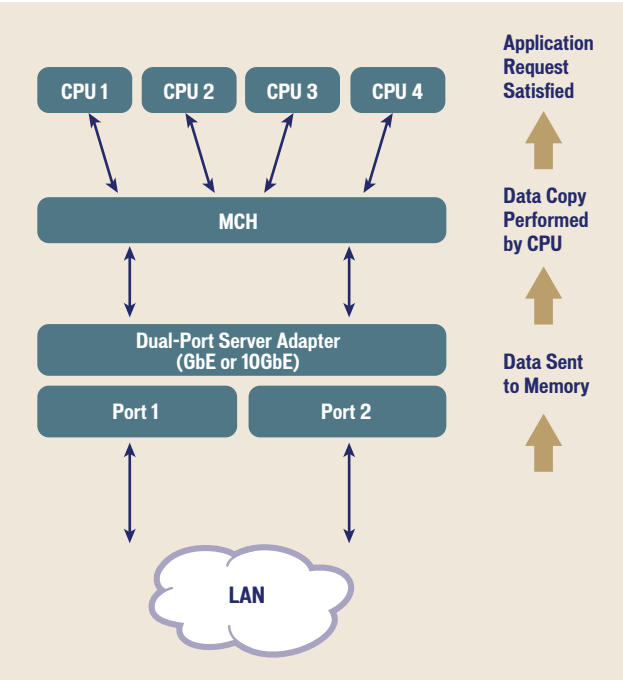

Figure 1. Data movement across the platform to satisfy application request.

Within the received packet process, the CPU must devote significant resources to fetching and copying data between memories. At the same time there may be multiple requests coming from applications, resulting in CPU cache misses. Such bottlenecks drop the service-level agreement (SLA) performance of the application and cause increases in CPU utilization levels.

The system-wide examination by Intel researchers revealed a wide range of causes for bottlenecks. The major sources include the following broad issues affecting performance:

Excessive context switching. Higher network speeds lead to a proportional increase in CPU interrupts. At GbE rates, full-size frames coming into the NIC at full network speed can cause 80,000 interrupts per second, which can have a significant impact on server performance and scalability. The host CPU is required to process interrupts from the network, and with increased interrupt rates, the amount of context switching between application and network traffic processing also increases.

Lack of effective scaling. In multi-processor or multi-core systems, limitations in network stack implementation can cause inefficient processing of inbound network traffic. Specifically, the inability to scale or load balance across multiple cores results in processing of received network packets on a single core or CPU. This inefficient processing of inbound traffic potentially reduces the realized value in multi-processor server investments.

Memory overhead and latency. In a conventional network stack implementation, received data often needs to be copied in memory between network buffers and application buffers. The CPU uses valuable cycles performing such copy operations. Additionally, the fact that memory speeds have not kept up with increases in CPU speeds worsens the CPU overhead associated with moving data within memory. Large numbers of CPU cycles may be spent waiting for memory accesses to complete, which increases CPU utilization and wastes processor cycles that could be used more effectively.

A Closer Look at Packet Processing

To further pinpoint the causes of I/O bottlenecks, it is helpful to examine the key events in packet processing. The following discussion provides this step-by-step view beginning with processing incoming packets received at the server adapter or LAN on motherboard (LOM). This is followed by a similar examination of outgoing packets generated as a response to the incoming packets.

Incoming Packet Processing. Going through the steps in the arrival and processing of an incoming packet provides insight into the role of interrupts, context switches, compulsory cache misses and other events affecting performance. The basic incoming packet processing steps are as follows:

1. Packet Arrives. The server adapter (NIC or LOM) receives incoming packets and performs a checksum validation on the packet data. It then uses a direct memory access (DMA) process to load information, including the packet descriptor, into system memory.

2. NIC Interrupts Host CPU. The NIC performs several such DMA operations for received packets. The NIC then interrupts the host CPU, which invokes an interrupt service routine (ISR).

3. ISR Makes Deferred Call. Typically, the ISR does a few tasks, such as interrupt acknowledgement, and then schedules a task referred to as a "deferred procedure call" to handle the interrupt.

4. Data Access Begins. Inside the deferred procedure call, once it is scheduled to execute on the CPU, data accesses begin. These accesses are required for retrieving the descriptor and header of received packets.

5. CPU Fetches Data from Memory. The data accesses result in a stall cycle from the perspective of the processor. This is because, when the NIC initiates the DMA action, the cache lines corresponding to the modified portion of memory are invalidated. Invalidating the cache subsequently forces the CPU to fetch the data from main memory, hence a stall. This is a compulsory cache miss on the CPU.

6. TCP/IP Processing Begins on the Header. After header-information retrieval, TCP/IP stack processing begins. As a part of this process, there is an access to memory for a Transport Control Block (TCB), which contains context information associated with the TCP/IP connection. Access to the TCB may or may not be a cache miss. There could be a cache miss if there is a conflict miss or capacity miss or for some reason the TCB was evicted from the CPU cache. After the TCB retrieval, TCP continues its processing.

7. Payload Transferred. If the incoming packet conforms to TCP/IP, the payload portion of the packet is copied from a kernel buffer to an application buffer.

This completes incoming packet processing. The same process repeats for each incoming packet.

Outgoing Packet Processing. Outgoing packet processing is similar to receiving TCP/IP data. The key events are as follows:

1. Send Call Made. When an application needs to transmit data, it typically issues a "send call" using the sockets interface. Assuming the send call executes in user mode, the operating system (OS) transitions into kernel mode, and the call comes down to the TCP/IP module that runs inside the OS kernel.

2. TCB Accessed. The first thing the TCP/IP process does is access a TCB. Similar to what happens in the receive data flow, the access to the TCB could result in a cache miss. In this case, the cache miss is most likely due to a conflict miss or a capacity miss.

3. Data Copied and Packet Created. After the TCB retrieval, TCP/IP continues to execute. The TCP/IP stack copies the data from an application buffer to a kernel buffer, which is used to create a network packet.

4. Packet Descriptor Created. Once the TCP/IP process creates a packet, it hands control over to the device driver, which creates a packet descriptor.

5. Memory-Mapped I/O Access Performed. The device driver creates the packet descriptor in pre-allocated buffers in shared memory and performs a memory mapped I/O access (also known as a doorbell write) to the NIC hardware.

6. NIC Reads Descriptor and Packet. When the NIC receives the doorbell write or memory-mapped I/O (MMI/O), it wakes up and performs a read of the packet descriptor. Then it performs a DMA transfer of the packet data into its hardware queue.

7. NIC Transmits Packet. After the NIC reads the packet, it sends the packet out on the wire.

8. NIC Sends Completion to CPU. After the NIC sends the packet on the wire, it issues a completion notification to the drive on the CPU that originated the send call.

9. Driver Frees Buffers. Once the driver receives the completion notice, it frees up the buffers.

As with processing received packets, these steps repeat for every transmitted packet.

In summary, the network data traverses multiple elements of the server platform. It traverses the NIC, the CPU, memory, and multiple layers within the operating system. To improve efficiency throughout the data path, improvements need to be made in each of the platform elements involved. This requires an I/O acceleration architecture that is platform-wide rather than just a point solution. Intel I/OAT and its tight integration with Windows Server 2008 provide such a platform-wide solution. This solution addresses TCP/IP processing and CPU overhead, and goes beyond to also address system overhead issues associated with packet I/O efficiency.

Overhead Beyond the CPU

A common question is the degree to which increasing processor capacity will alleviate network bottlenecks. While all compute-related elements of TCP/IP scale linearly with CPU speed, other tasks, including the memory-related operations, do not scale linearly. This is because CPU core speeds are faster than memory speeds.

When TCP/IP processing forces the CPU to access memory repeatedly, the CPU cannot execute at its core speed. This is why memory-dependent tasks cannot scale linearly with CPU core-speed increases.

However, as shown in Figure 2, TCP/IP processing, while scalable, represents only a small part of the total time required to transmit data. System overhead including descriptor and header accesses, TCB accesses, and memory copies involve activities outside the CPU. These activities introduce stalls, repetitions, and other delays that keep the CPU from executing at maximum throughput.

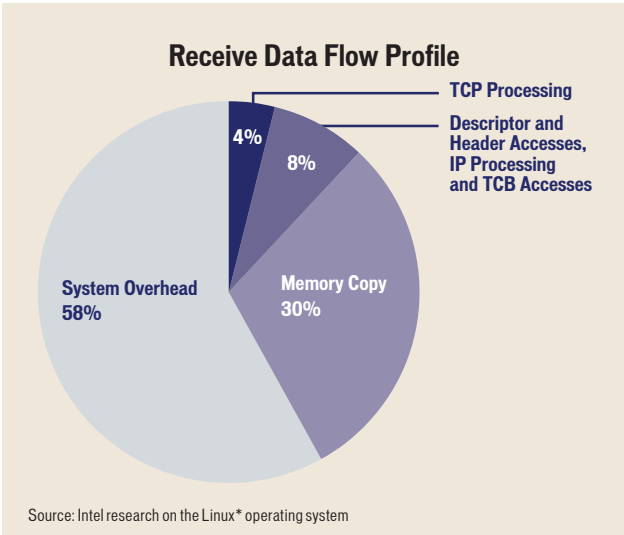

Figure 2. TCP processing represents only a small part of the total time required to receive data, while other system overheads represent the bulk of the time spent.

Adding to system overhead is the fact that many of the compute-related elements and memory operations occur serially in network data processing. The system first accesses the header and the descriptor, then performs some processing on the data, and then copies the data in a serial fashion for transmission. As a result, system overhead in total, presents the largest bottleneck for overall received data flow.

Intel I/OAT with Intel QuickData Technology and their tight integration with Microsoft Windows Server 2008 address most of the issues presented in Figure 2, from TCP/IP processing to the larger issue of overall system overhead. The techniques and technologies used to address these issues are the topics of the next section.

Network Acceleration Technologies

As discussed previously, I/O bottlenecks are a system-wide problem. Providing a system-wide solution to I/O bottlenecks is the fundamental premise of Intel I/OAT. This means the solution must be applied through the server NIC or LOM, the chipset, and the operating system. To do this, Intel I/OAT capability is incorporated in all PCI Express Intel Server Adapters and Ethernet Controllers. It is also incorporated in the chipset and the Multi-Core Intel Xeon Processors and includes Intel QuickData Technology for integration with operating systems. In particular, Intel QuickData Technology and Network Direct Memory Access (NetDMA) in Microsoft Windows Server 2008 work together to accelerate data and unburden the CPU from memory-to-memory copies. This and the other network acceleration features of Intel I/OAT and Window Server 2008 provide a system-wide solution to I/O bottlenecks.

Table 1. Summary of Network Acceleration Technologies

Technology	Windows Server® 2008	Multi-Core Intel® Xeon® Processor	Intel® Server Adapter or LOM
NetDMA	X (activates with Intel® QuickData Technology)	X	
Receive Side Scaling	X		X
TCP Chimney Offload	X		
Intel® QuickData Technology	X	X	X
Direct Cache Access	X	X	X
Receive Side Coalescing			X
Extended Message-Signaled Interrupts (MSI-X)	X		X
Low-Latency Interrupts			X
Header Splitting	X		X
TCP Checksum Offload and Segmentation Offload	X		X

Intel® I/OAT Features

Table 1 provides a summary of network acceleration features for reference. Each of the features listed here is discussed in greater detail in the rest of this section according to where they largely reside in the system–operating system, chipset, or server adapter.

Operating System Technologies

The operating system technologies discussed here apply to Microsoft Windows Server 2008. These technologies were also available in Windows Server 2003, but they have undergone further optimization in Windows Server 2008.

Network Direct Memory Access (NetDMA). Jointly designed by Intel and Microsoft, NetDMA enables memory-management efficiencies through a direct memory access (DMA) engine on servers equipped with Intel QuickData Technology. The Intel QuickData Technology (DMA engine), which is a part of Intel I/OAT, offloads the task of memory-to-memory data transfers from the host CPU. With Intel QuickData Technology on the server, NetDMA largely frees the CPU from handling the task of copying data between memory locations. This streamlines memory access, reduces system overhead, and reduces over-utilization of CPU resources for I/O operations.

Windows Server 2008 automatically invokes NetDMA when it detects supporting hardware, such as servers equipped with Intel QuickData Technology. Thus, NetDMA requires no setup or implementation effort from IT personnel.

Receive Side Scaling (RSS). Developed by Microsoft, RSS balances inbound (receive) network traffic across multiple CPUs or cores in a multi-core processor. This feature works in conjunction with RSS support in Intel I/OAT to dynamically balance received network traffic loads as either the system load or network conditions vary. This dynamic balancing prevents one core, or CPU, from being overburdened while another is idle or near-idle. The result is improved overall throughput. Any applications with significant inbound networking traffic and running on a multi-core processor-based server can benefit from RSS.

In operation, RSS directs incoming data packets to different queues without the need for reordering. The specific queue for a given packet is determined by calculation of a hash value from the fields in the packet header. The hash value serves as a lookup mechanism in a hash table in the NIC that determines into which queue the packet should be directed. Hash values also determine the specific processor for handling the packet flow to ensure that packets are handled in order.

RSS and NetDMA are complementary and can be used together. RSS is complementary with the Windows Server 2008 TCP Chimney Offload feature, as well; however, NetDMA and TCP Chimney Offload cannot be invoked simultaneously (see below).

TCP Chimney Offload. TCP Chimney Offload provides automated, stateful offload of TCP traffic processing to a specialized network adapter implementing a TCP Offload Engine (TOE). Such adapters are often referred to as TOE NICs. Stateful offload to a TOE NIC means it retains significant attributes of a connection, such as IP address, ports being used, and packet sequence numbers in the memory available on the NIC.

For long-lived connections with large-sized packet payloads and workloads that transfer large data sets via the network, and other content-heavy applications, TCP Chimney Offload reduces CPU overhead by delegating network packet processing tasks, including packet segmentation and reassembly to the TOE NIC. TCP Chimney Offload is best suited for scenarios involving large I/O sizes over long-lived connections.

If Windows Server 2008 detects a NIC with both DMA and TCP offload capability, TCP Chimney Offload will be preferentially used and NetDMA will not be used. Since NetDMA is essential for using Intel QuickData Technology, that capability becomes unavailable in a TCP offload environment. However, RSS can still be used across all TCP connections, including those that are offloaded through TCP Chimney Offload.

Server Chipset Technologies

Intel I/OAT comprises Intel QuickData Technology and DCA technologies on the chipset as well as other networking features on the server adapter. Multi-Core Intel Xeon processors incorporate these same technologies; thus, Intel I/OAT capability is available on any multi-core Intel Xeon processor-based server.

Intel QuickData Technology. Intel QuickData Technology is part of Intel I/OAT and has several functions. Primarily it is a data-acceleration engine using DMA. However, it also allows third-party networking and server vendors to take advantage of I/O acceleration in addition to Intel adapters with Intel I/OAT capability.

Intel QuickData Technology provides the close link with Windows Server 2008 through NetDMA. When a TOE is not present and Windows Server 2008 detects the Intel QuickData DMA engine, it invokes NetDMA. Together, Intel QuickData Technology and NetDMA provide memory-to-memory data transfers. NetDMA uses the interfaces provided by Intel QuickData Technology to provide data-copy functionality with the chipset instead of the CPUs. This frees the CPUs from memory copy overhead and streamlines the I/O process.

The Intel QuickData Technology memory transfer process is sometimes referred to as asynchronous low-cost copy. This is because payload data copies from the NIC buffer in system memory to the application buffer are made at a much lower cost in CPU cycles. This allows returning of the saved CPU cycles to productive application workloads.

Direct Cache Access (DCA). The objective of DCA is to take full advantage of high-speed cache and eliminate the CPU cycles needed to read packet headers and descriptors from system memory. To do this, DCA tags the data packets and sends the information to the CPUs. This allows the CPU to pre-fetch the data packet from memory, thus avoiding cache misses and the associated waste of CPU cycles. The result is streamlined I/O and improved application response and service levels.

Server LAN Technologies

Various network acceleration features of Intel I/OAT are present on PCI Express Intel Server Adapters. Intel I/OAT features are optimized for use with multi-core processors in a platform-wide Intel I/OAT environment such as supported by Microsoft Windows Server 2008. In such an environment, significant network throughput and performance improvements can be expected.

It is also important to note that the Intel I/OAT features discussed below do not preclude use of PCI Express Intel Server Adapters in other environments that do not support Intel QuickData Technology. In such environments, the server adapter still retains all of its Ethernet functionality and still applies acceleration features that do not rely on Intel QuickData Technology support on the server platform or from the operating system.

Receive Side Scaling (RSS). As mentioned previously, RSS is an optimization for multi-processor or multi-core processor-based servers. Within the server adapter, it maps queues to specific cores, allowing multi-core, parallel processing of receive traffic. This balances traffic across multiple cores for faster, more efficient processing of receive traffic. The process is sometimes referred to as affinitized data flows, since different flows are given affinities for certain processor cores by queue mapping.

Intel Server Adapters also support extended message-signaled interrupts (MSI-X) in conjunction with RSS for ideal scalability. With both RSS and MSI-X, the adapters scale interrupt processing as well as the actual packet processing.

The RSS support provided by Intel I/OAT on Intel Server Adapters uses the RSS feature of Microsoft Windows Server 2008 to accomplish dynamic load balancing of receive traffic. This is just one of many examples of tight integration with the OS for providing a system-wide solution for networking acceleration.

Receive Side Coalescing (RSC). RSC identifies packets belonging to the same TCP/IP flow and coalesces them into a single large packet. This reduces the number of packets for TCP/IP stack processing. Reducing the number of packets also reduces the number of interrupts and memory fetches. The overall result is significant increases in TCP/IP processing efficiencies and significant decreases in CPU utilization for TCP/IP processing. Currently, Microsoft Windows Server 2008 does not support this feature.

Extended Message-Signaled Interrupts (MSI-X). PCI Express Intel Server Adapters provide multiple hardware queues for use by RSS in balancing traffic across multiple cores. With MSI-X, the server adapter also has the ability to program each hardware queue to interrupt the CPU core that will be processing the received data. This allows MSI-X to provide efficient communication between multiple queues and specific processor cores. It does this by providing multiple interrupt vectors and giving each queue its own set of MSI-X controllable interrupt vectors. This allows efficient packet management and fine-tuning of the processor load.

In essence, by providing an interrupt vector for each queue, MSI-X provides simultaneous handling of multiple interrupts and load balancing across multiple CPU cores. The benefits are more efficient CPU utilization and greater scalability.

MSI-X and multiple queues work together with RSS to distribute Ethernet traffic across CPUs in a multi-core system. Distribution is controlled by the memory controller hub (MCH) on the chipset.

Low-Latency Interrupts (LLI). LLI is an interrupt moderation tool. It tunes the interrupt time intervals between data packets of different applications or workloads. LLI automatically tunes or adjusts the time interval based on the latency sensitivity of the applications or the workloads. As a result, latency sensitive data can move faster through the system.

Header Splitting. Header splitting is a technique in which the header and packet payload are separated to allow processing of each part of the network packet on a separate, parallel path. Header splitting reduces latency in packet processing because some time is wasted when the CPU looks at both the header and the payload as a single unit. In truth, the CPU only needs to look at the header in order to start the delivery process, and header splitting allows this to occur without wasting CPU resources on the payload portion.

TCP Checksum Offload. Various stateless offloads are provided on Intel Server Adapters to provide more efficient processing of network traffic. TCP Checksum Offload moves the burden of checksum computations from the CPU to the network adapter. This includes TCP, User Datagram Protocol (UDP), and Internet Protocol (IP) checksum computations. Since these offloads are stateless, they are OS- and environment-agnostic.

Benefits of Intel® I/OAT

At the highest level, the benefit of Intel I/OAT is that it is a fully integrated, system-wide solution that enhances network data flow across the system. Additionally, it is a solution that is further optimized for multi-core servers, for data center virtualization, and for emerging unified networking architectures. Consequently, Intel I/OAT provides a network acceleration solution that not only addresses the issues of networking today but also provides a migration path to the networking solutions of the future.

Fully Integrated, System-Wide Solution

As previously discussed in detail, I/O bottlenecks are a system-wide issue. They occur in the NIC, the server, and the operating system–throughout the entire TCP/IP packet-processing path, from received packet to the transmitted packet.

It makes sense then, that resolving I/O bottlenecks requires a system-wide solution spanning the NIC, the server, and the operating system. Intel I/OAT with Intel QuickData Technology offers just such a solution.

PCI Express Intel Server Adapters combined with Multi-Core Intel Xeon processor-based servers provide the Intel I/OAT-enabled platform. NetDMA and RSS in Microsoft Windows Server 2008 integrate with Intel QuickData Technology (DMA engine) on the server and RSS support on the server adapter to complete the solution. NetDMA and RSS work along with all of the other Intel I/OAT and server adapter features – DCA, MSI-X, LLI, header splitting, and stateless offloads – to provide substantially enhanced flow of network data.

Intel I/OAT and its tight integration with Windows Server 2008 provide enhanced network data flow through the following actions:

- Optimized TCP/IP computation
- Streamlined memory access
- Minimized processing required by CPU
- Reduced overall system overhead

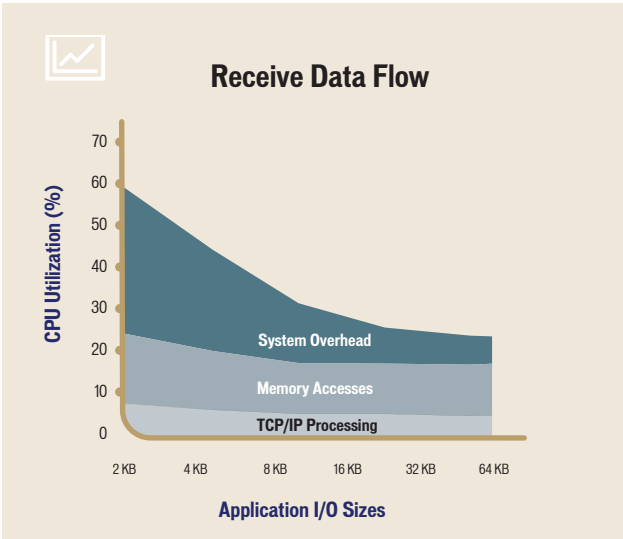

Figure 3. CPU utilization varies with I/O packet size. TCP/IP processing remains fairly constant, while memory accesses and system overhead increase at smaller packet sizes.

The benefit of these system-wide enhancements is that they apply to a wide range of network environments. The importance of this can be deduced from Figure 3, which illustrates GbE traffic bottlenecks in terms of CPU utilization percentage versus application I/O packet size. A NIC-only solution that addresses TCP/IP processing only, for example, fails to address the more significant CPU usages caused by system overhead and memory accesses at smaller I/O packet sizes. In contrast, Intel I/OAT address all major issues – TCP/IP processing, memory accesses, and system overhead – for all packet sizes and connection times. The result is a comprehensive solution for a wider range of applications.

Intel I/OAT is a solution that moves data faster and frees CPU usage for critical compute tasks. It is scalable across multiple GbE ports and can scale up to 10GbE. And it is reliable because Intel I/OAT is tightly integrated into most popular operating systems, such as Windows Server 2008. This tight integration avoids the support risk associated with relying on third-party hardware vendors for network stack updates. Also, Intel I/OAT preserves critical network capabilities, such as teaming and failover, by maintaining control of network stack processing in the CPU. This results in less support risk for IT departments.

Optimized for Now and the Future

Current networking trends include increased use of 10GbE server adapters on multi-core processor-based servers, as well as in a data center consolidation and virtualization environment. All of this is being done with the goal of increasing bandwidth and processing power for faster application response times while increasing the manageability and efficient usage of data center resources. Intel I/OAT is optimized to serve all of these needs as well as newly emerging networking architectures and technologies.

Specifically, the data throughput enhancements offered by Intel I/OAT allow IT managers to realize the bandwidth expectations offered by use of adapter teaming techniques as well as migration to 10GbE. Even greater enhancements can be expected and realized on multi-core processor-based servers where the RSS feature of Intel I/OAT and Windows Server 2008 can use multiple queues to implement the efficiencies of flow affinity and load balancing across multiple cores.

The network acceleration benefits offered by Intel I/OAT and Windows Server 2008 also offer an entry into the new arena of unified networking. An example of this is the availability of Internet Small Computer System Interface (iSCSI) initiators in Windows Server 2008 and iSCSI Remote Boot capability for Intel Server Adapters. These iSCSI features, along with the accelerated data flow provided by Intel I/OAT, make Ethernet-based storage-area networks (SANs) practical. The advantage, of course, is that using Ethernet as a SAN fabric is less costly than the specialized fabrics and hardware traditionally used for SAN implementation. An Ethernet SAN fabric is also easier to maintain since it does not require the specialized skills necessary for Fibre Channel* or other proprietary SAN fabrics. Moreover, an iSCSI SAN can coexist with Fibre Channel SANs, making iSCSI an ideal and a very economical expansion path for existing SANs.

Ratification of the Fibre Channel over Ethernet (FCoE) standard in 2008 offers yet another option for SANs. FCoE allows LAN and SAN traffic to be converged onto a single 10 Gigabit Ethernet (10GbE) link. This reduces the number of ports, adapters, and switches necessary for LAN and SAN implementation. To support this course in the near future, PCI Express Intel® 10 Gigabit Ethernet Server Adapters can be upgraded to be FCoE capable by loading an Intel-supplied FCoE initiator onto the server operating system. This makes the 10 Gigabit server adapter a multi-functional device in that it can be both a 10GbE adapter and an FCoE adapter, each of which enjoys the full benefits of Intel I/OAT.

Conclusion

Providing faster network traffic flow is not a matter of a single solution. It is a matter of a well-integrated suite of solutions that spans the server adapter, the chipset, and the operating system. Intel I /OAT and Microsoft Windows Server 2008 provide such a solution.

The primary result is enhanced network traffic flow across the system. This provides more efficient server usage and allows servicing more client connections more efficiently with the same resources. However, the benefits of I/O acceleration go well beyond that. With enhanced network traffic flow and optimization for multi-processor and multi-core systems, the benefits of Intel I/OAT also extend to more efficient and effective server virtualization. Intel I/OAT also has the ability to make iSCSI SANs a practical and economic reality. Such are the far-reaching benefits of the system-wide network acceleration solution offered by the integration of Intel I/OAT with Microsoft Windows Server 2008.

For More Information

Windows Server 2008 at **www.microsoft.com/windowsserver2008**

Intel I/O Acceleration Technology at **www.intel.com/go/ioat**

For the latest information about Intel® networking products, see the Intel® Network Connectivity site at **www.intel.com/network**

[1] Intel® I/O Acceleration Technology (Intel® I/OAT) requires an operating system that supports Intel I/OAT.

[2] Intel® QuickData Technology requires an operating system that supports the DMA Engine.

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, visit www.intel.com/performance/resources/limits.htm or call (U.S.) 1-800-628-8686 or 1-916-356-3104.

Printed in USA 0508/JAL/OCG/XX/PDF ♻ Please Recycle 317106-002US