Optimize I/O expansion in workstations

By Steve Moore
Senior Product Marketing Manager
PLX Technology Inc.

High-end workstations are constantly being pushed for increased performance—from gaming applications to modeling and simulating complex electronic and mechanical systems. It is crucial to get every bit of performance available from today's workstations. A number of pitfalls are often encountered as bridges and switches are added to increase the connectivity. These include increased latency, processor overloading, reduced bandwidth and diminished overall system performance.

PCIe switches and PCI-to-PCIe bridges can help get beyond many of the workstation I/O limitations mentioned. Let's look at how these devices can be used to optimize I/O expansion in high-performance workstations.

Optimizing lane count

How many lanes are sufficient to satisfy a workstation's thirst for I/O connectivity with enough bandwidth/throughput performance? **Figure 1** shows the relationship between lane count and maximum throughput. The maximum throughput shown here assumes ideal conditions; typically the throughput achieved in a real system is less than the maximum shown in this table. This is due to latencies in both the system and the PCIe switches and bridges, as well as in the bandwidth limitations of those devices.

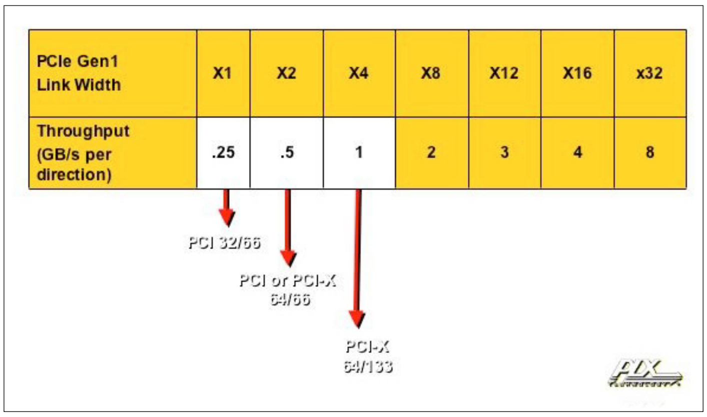
Every chipset and system board for a workstation comes with a limited number of I/O connections. **Figure 2** illustrates a typical workstation chipset with a 48-lane switch (PEX 8548 from PLX), adding three ports of high-performance I/O connectivity. In this application, the availability of 48 lanes is crucial to optimize the throughput in the workstation. Using a switch with 32 (or fewer) lanes will constrict the lane count on one or more of the ports thus creating a performance bottleneck. A 48-lane switch allows the creation of two x8 ports (for a Quad Gigabit Ethernet, or GigE, connection and a serial ATA (SATA) 2 storage interface) and a x16 port for a PCIe slot addressing high-performance add-in cards such as graphics adapters.

Once the port count and lane count are optimized, it is important to ensure that each I/O channel's latency is minimized. Overcoming latency isn't solely a matter of choosing the lowest-latency components suitable for your system, but it's a good place to start. It's also a matter of architecting operations to reduce or eliminate the sensitivity of performance to latency. It is impossible to totally eliminate the effects of latency, so the less there is to begin with, the better.

Figure 3 explains latency in a PCIe switch. Simply put, it is the time it takes for the beginning of a transfer layer packet (TLP) to move from the ingress port to the egress port of a switch. A PCIe device's latency consists of the sum of the "pipeline delay" and the "queuing delay." The pipeline delay is the length of time for a packet to traverse an otherwise empty switch and is dependent upon how the switch is architected. The queuing delay depends on the type of traffic patterns, in addition to the switch's flow-control credits, arbitration, and scheduling policies.

Eliminating bottlenecks

The early generations of PCIe switches use a "store and forward" architecture. While easy to implement, this architecture suffers from high latency because the entire packet must be received at the ingress port before being transmitted out through the egress port. A new architecture known as "cut-through" has been deployed in subsequent PCIe switch generations. Switches designed with cut-through engines start transmitting packets without waiting for the end of a packet to arrive, thus reducing switch latency. The cut-through architecture improves initial packet-to-packet latency, and is found to be most



PCIe Gen1 Link Width	X1	X2	X4	X8	X12	X16	x32
Throughput (GB/s per direction)	.25	.5	1	2	3	4	8

PCI 32/66
PCI or PCI-X 64/66
PCI-X 64/133

Figure 1: The throughput achieved in real systems is less than the throughput shown above, which is under ideal conditions.

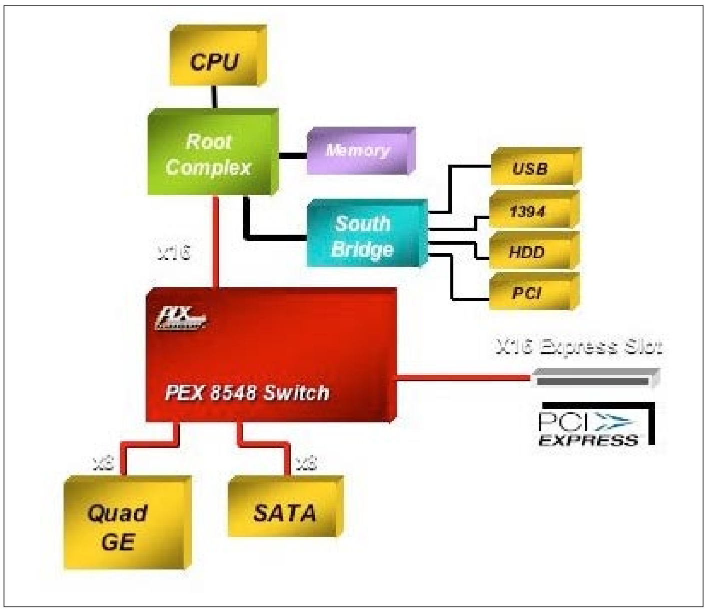


Figure 2: In this typical workstation chipset, the 48-lane switch (PEX 8548 from PLX) adds three ports of high-performance I/O connectivity which are crucial to optimize the throughput.

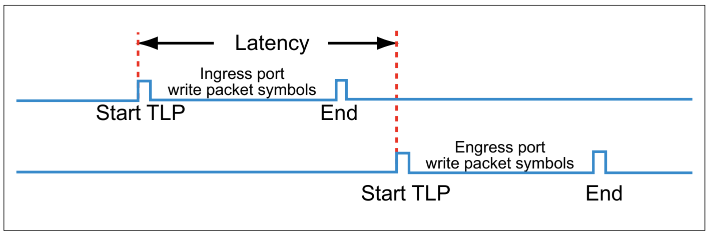

Latency
Ingress port write packet symbols
Start TLP
End
Engress port write packet symbols
Start TLP
End

Figure 3: The latency in a PCIe switch is the time it takes for the beginning of a TLP to move from the ingress port to the egress port of a switch.

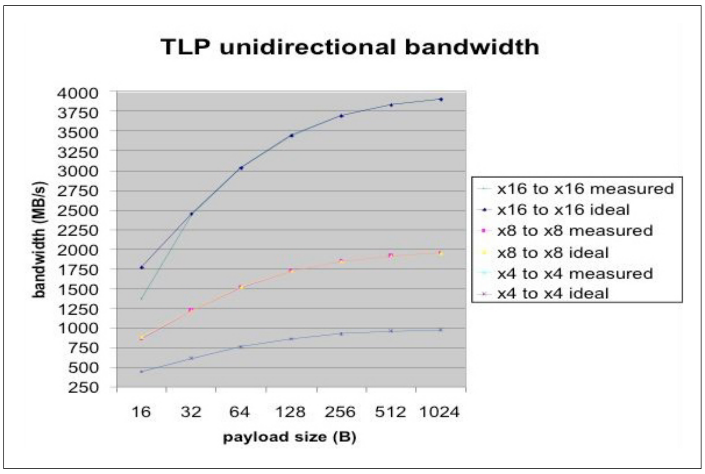


Figure 4: The PLX PEX 8548 PCIe switch performs close to ideal performance for payloads greater than 16bytes.

beneficial for bursty traffic patterns such as the traffic found in GbE systems and in control planes using PCIe.

Figure 4 plots the impact of payload size to the bandwidth achieved through a PLX PEX 8548 PCIe switch. The data indicates that this switch performs close to ideal performance for payloads greater than 16bytes. Clearly, as the payload size is decreased, the throughput is reduced. This is because the switch's latency, which is relatively constant, becomes a large percentage of the total system delay when payload size is reduced.

Port arbitration helps eliminate workstation I/O bottlenecks by providing a vehicle for prioritization of data flow in scenarios where there are oversubscribed ports on the switch. The most effective implementation of this is the weighted round-robin port arbitration. This scheme guarantees bandwidth allocation for critical traffic classes in applications such as high-throughput GbE I/Os, SATA storage controllers, and SCSI and Fibre Channel HBAs. Configuring port arbitration easily accomplished by loading an arbitration table and storing it in the device's configuration serial EEPROM.

It is necessary to consider the relationship between port width and latency as well. A switch will not forward a packet until it has seen enough of the packet's header to determine which egress port it will use. The wider the input link, the less time required to see the complete header. On x16 link for example, the entire header may be visible in a single clock, depending upon how the start-of-packet aligns on the link. On x8 link it will likely take twice as long to see the entire header, and twice as long again on a x4 port. Thus, using a wider port generally will result in lower latency.

Overcoming traps

Many I/O endpoints (like SCSI controllers) are only available with PCI and PCI-X interfaces, yet frequently high-performance workstations will need to connect them into the system board. Many workstation boards are shipping with little or no PCI or PCI-X connectivity, featuring primarily PCIe connections. This situation requires a PCIe-to-PCI (or PCIe-to-PCI-X) bridge, as shown in **Figure 5**. Since this scenario involves a parallel bus (PCI/PCI-X), loading effects need to be considered in order to optimize I/O performance.

Figure 5 also indicates how many slots can be included on a single PCI (or PCI-X) bus segment. The number of slots is dictated by the PCI specification, which was based primarily on the ability of a PCI-compliant I/O driver to meet the PCI timing requirements surrounding the capacitance of each slot's load. The timing becomes more difficult to meet at 66MHz, hence the number of slot loads tolerated by the specification is reduced. The PCI-X protocol provides for registered transactions, and thus the timing is easier to meet at 66MHz when compared with PCI. Therefore, more PCI-X slot loads are allowed at 66MHz For example, if one device is designed to operate at 66MHz, and another will only support 33MHz, the PCI bus, which would otherwise operate at 66MHz with the first device, will "fall-back" to the lowest frequency supported amongst the devices on the segment. In this case, the 66MHz device is constrained to operate at 33MHz, thereby reducing its throughput potential to one-half of what it could otherwise be.

The way around this is to create two separate PCI segments, each operating at its own rate. This is done by inserting an asynchronous PCI-to-PCI bridge, such as the PCI 6150 from PLX, as shown in **Figure 6**. This type of bridge allows different frequencies on the primary and secondary sides of the bridge.

Meanwhile, one of the drawbacks of an asynchronous bridge is the latency often introduced as the data traverses the FIFO required to support the asynchronous feature. For example, the

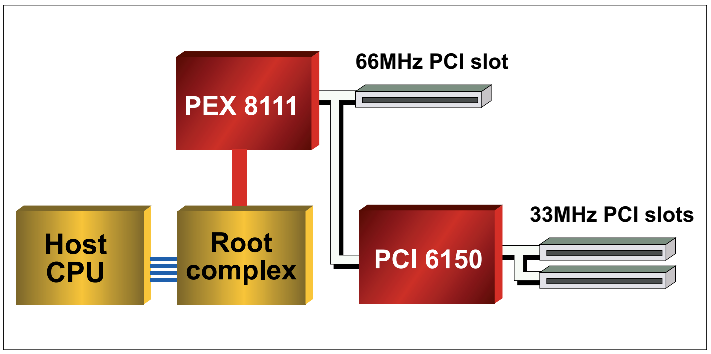


Figure 6: "Bus fall-back" can be avoided by creating two separate PCI segments using the PCI 6150 which allows different frequencies on the primary and secondary sides of the bridge.

versus conventional PCI at the same clock speed.

Figure 5 also calculates the maximum throughput available on a per-channel basis, assuming that all available slots are filled. The calculations factor in the maximum throughput divided by the maximum number of I/O channels on the segment; about 10 percent is allowed for arbitration overhead.

In the realm of parallel buses, one performance trap is known as "bus fall-back." This can occur when two or more devices are on the same PCI bus segment. PCI 6140 is a synchronous-only PCI-to-PCI bridge, which offers performance-optimized retry and programmable flow-through to minimize the latency and other delays typically associated with PCI bridges. If extremely low-latency bridging is required, a bridge like the PCI 6140 will work best.

Using prefetch

The PCIe-to-PCI bridge utilized in Figure 5 will also become a performance bottleneck if not properly selected and set up. One of the performance-optimizing

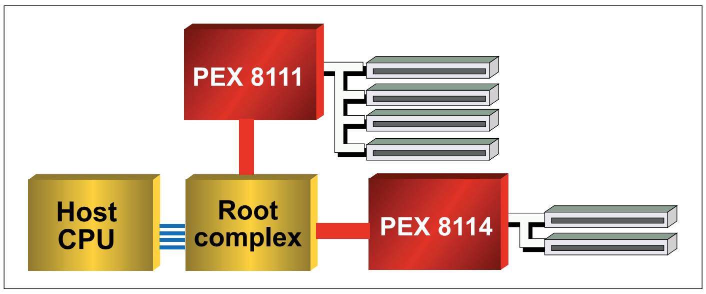


Figure 5: A PCIe-to-PCI bridge is useful for connecting I/O endpoints with PCI interfaces to a workstation board featuring PCIe connections.

techniques available on a PCIe-to-PCI bridge is known as "blind prefetching." Blind prefetching allows the bridge to read a pre-defined amount of data from PCIe memory. This data is buffered in the bridge whenever a device on the PCI side of the bridge reads one or more DWORDS from memory on the PCIe side of the bridge. The amount of data buffered on the bridge is typically more than what the PCI device is initially requesting. The extra data is buffered in anticipation of a second, sequential read from the PCI device. Bridges without the blind prefetch capability can only transfer a single DWORD at a time.

Each individual DWORD transfer requires some setup time to process the transaction, thus adding to the total latency through the bridge. This latency in the bridge can impact the throughput significantly. Blind prefetch, on the other hand, will allow the bridge to burst up to 4Kbyte of data in a single transaction. The bridge's burst transaction minimizes the setup time and, ultimately, the latency to only one transaction. When using a bridge with blind prefetch enabled, the initial latency penalty only occurs once for every 4Kbyte transferred. Thus, the blind prefetch capability allows for maximum read performance by minimizing the latency time required for devices reading large amounts of sequential data through the bridge.

Increasing performance levels demanded by gaming, modeling and simulating in workstations is challenging—and the I/O performance must be optimized, just as the CPU and other elements in the system are maximized. In deploying workstation I/O expansion to increase the connectivity of the workstation, you must overcome I/O switching and bridging latency, which can lead to processor overloading, reduced bandwidth and reduced overall system performance. Properly selecting and configuring the I/O devices will help in overcoming PCIe lane and port limitations, performance hits resulting from slot loading, and bridge and switch latency.