24 Lane, 6 Port PCI Express® Switch Performance Report

89PES24T6

Notes

Overview

This document presents performance measurements and benchmarking results for IDT's 89PES24T6 24-lane, 6-port peripheral chip, a member of IDT's PRECISE™ family of PCI Express Switching solutions. The PES24T6 has one upstream port and up to five downstream ports. Ports are nominally 4 lanes wide, but two 4-lane ports can be merged to create an 8-lane port. The switch is compliant with PCI Express (PCIe®) base specification revision 1.1.

The test vehicle for the PES24T6 is the evaluation board IDT89EBPES24T6 which hosts the PES24T6. Accompanying the throughput performance metrics are descriptions and methodologies outlining the test setup and procedures.

The nature of tests and the equipment used for these tests varies significantly across the spectrum of tests performed. In the interest of readability and searchability the document is divided into various sections. Each section represents a single test suite that employs a single test setup. A single test suite is capable of high-lighting several features of the switch device under test.

Section I provides some insight into issues that can affect the performance of a PCIe device. This includes overhead derived from the protocol, as well as the architectural decisions made while implementing the PCIe device.

Section II describes the performance of the PES24T6 with Gigabit Ethernet endpoints attached to its down-stream ports. Bidirectional performance comparisons with and without the PCIe switch in the traffic path are provided for both Windows and Linux environments. SmartBits™ SMB600 is used to generate controlled Ethernet traffic which is looped back between the GE NICs.

Section III provides a throughput report using PES24T6 with Fibre Channel (FC) traffic. 4 Gbps FC cards are used on three of the downstream ports. Reads and writes to an array of disk drives are done with the IOmeter software tool. IOmeter is also used to gather and analyze the performance data.

Appendix A gives a brief introduction to the SmartBits traffic generator and analyzer and the SmartFlow™ test software package used in conjunction with this test equipment.

Appendix B is an introduction to the software tool called IOmeter that is used in generating some of the storage and networking test results presented in this report.

Revision History

November 14, 2006: Initial version.

October 29, 2007: Section 1 completely rewritten.

.

SECTION I: PCIe Performance Basics

The PES24T6 primarily serves the purpose of high-performance I/O connectivity expansion in a typical system. Simply put, the PES24T6 uses one existing PCIe port in a system and offers up to five ports in its place. Given that nothing ever comes for free, it is presumed that the addition of a port has some "cost" associated with it in the form of real estate on the system board, power/heat, design complexity, support circuitry/devices (clocks, hot plug controllers, EEPROMs, power regulators, jumpers, etc.), signal integrity, or adverse effects on throughput/latency. All but the last item in this list are unavoidable to some extent. It is the impact on throughput and latency (system performance in general) that is the least intuitive to predict without a reasonable understanding of the system and switching device architecture, the usage model of the switching device, and some basic understanding of the PCIe protocol itself. In this section, some of these elements are introduced to the users of the PES24T6, specifically those users who are new to PCIe and switching. Advanced users of PCIe and switches may skip the reminder of this section.

What Does Performance Mean?

PCIe switch performance can mean different things to different users. The following introduction to some basic terminology may clarify what ought to be important when selecting a switch for your system design.

Throughput

"Raw throughput" refers to the total number of bits that pass through the switch in a given period of time, regardless of function, source, or destination. The PES24T6 is designed to handle 2.5 Gigabits per second (Gbps) of raw throughput in each direction on each of its lanes. This switch is designed for IO expansion (or fan-out) where the traffic flows to and from the root complex via the switch, and as such the maximum width of the upstream port indicates the maximum throughput that this switch can achieve. This results in (2.5 Gbps) x (2 directions) x 8 (lanes) = 40 Gbps of raw switching capacity. In reality, the switch is not required to "switch" this amount of data, as seen below.

PCI express data bytes undergo 8b/10b encoding. Discussion of the 8b/10b mechanism is beyond the scope of this document. It is sufficient to note that two out of every ten bits passing across a PCIe link do not contribute to any meaningful user data and are stripped off before the data enters the switch core. Therefore, this 20% overhead must be deducted from the raw throughput that the switch must support in terms of actual switching capacity. For the PES24T6, the ideal "switch throughput" now becomes 80% of 40 Gbps, i.e. 32 Gbps, assuming simultaneous bidirectional traffic on the upstream port.

However, there is more overhead at play. Every payload packet (actual user data) is preceded and followed by a variable number of bytes as required by the PCIe protocol. These bytes include the frame K-code, sequence number, TLP header, optional ECRC, and LCRC. This is the "framing" overhead (see Figure 1).

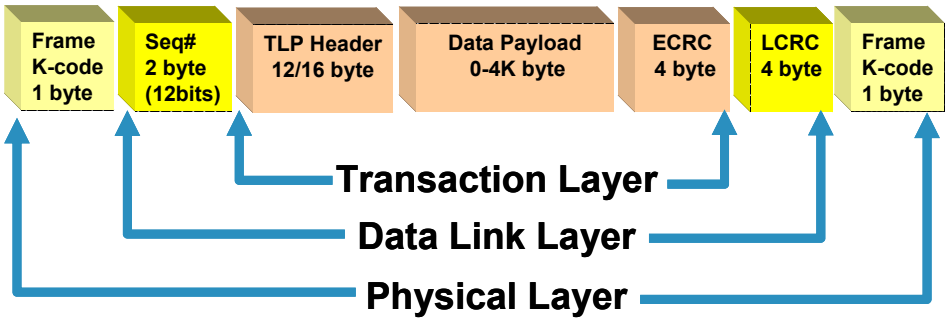

Figure 1 Framing Overhead in a Typical Transaction Packet

Figure 2 shows the effect of framing overhead on useful bandwidth for payloads at different PCIe link widths. A 20 byte overhead is assumed per payload packet for the purpose of this chart. This includes 1 byte of start of packet code, 2 bytes of sequence number, 12 bytes of TLP header, 4 bytes of LCRC, and 1 byte of end of packet code.

So, for example, on a x8 link, at 2.5 Gbps per lane per direction, raw bidirectional bandwidth is 40 Gbps. Upon removing the 8b/10b overhead, the useful theoretical maximum bandwidth available is 32 Gbps. Similarly, the theoretical maximum useful bandwidth for a x4 link is 16 Gbps, that for a x2 link is 8 Gbps and for a x1 link it is 4 Gbps.

To understand the calculations behind the chart shown in the figure, let us pick an example of a 64 byte payload packet on a x8 link to see how we come up with the corresponding data point on the chart. Total packet size with overhead becomes 84 bytes on account of the 20 byte overhead explained above. 32 Gbps (giga **bits** per second) of useful bandwidth is the same as 4 GBps (giga **bytes** per second). This is the same as 4000 MBps (mega bytes per second). In terms of packets, this means 4000/84 (i.e. 47.61) million packets. Payload bandwidth for 47.61 million packets is 47.61 multiplied by 64 bytes per packet, or a payload bandwidth of 24.38 Gbps. This is the 64 byte payload data point plotted on the x8 link chart in Figure 2). As seen in the chart, it is possible to achieve close to 32 Gbps (the theoretical maximum for a x8 link) under ideal conditions for payloads larger than 512 bytes.

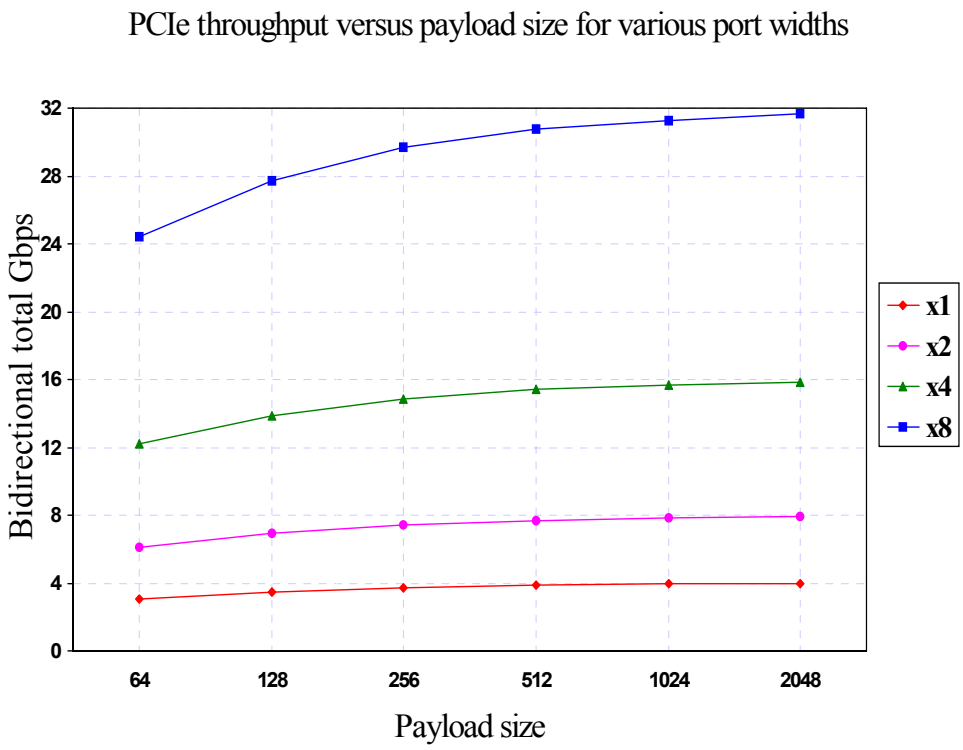

Figure 2 Effect of Framing Overhead on Link Efficiency

As calculated previously, at 64 byte payloads, the maximum throughput achievable on a x8 link is a bit over 24 Gbps. This means that out of the 32 Gbps useful bandwidth available, approximately 8 Gbps is spent on PCI Express framing overhead and approximately 24 Gbps on payload of 64 bytes payload per packet. This implies close to 75% efficiency on the “wire” (link). Since this framing overhead is constant irrespective of the link width, the wire efficiency is independent of link width in ideal conditions. For those who like to think in terms of wire efficiency as opposed to actual bytes or bits per second bandwidth, Figure 2 can help.

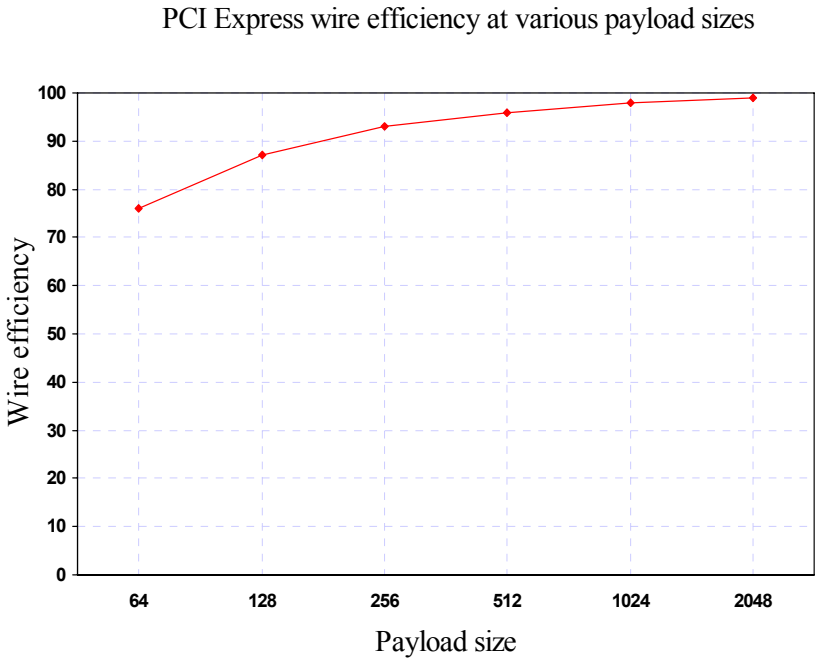

Figure 3 Data Path Efficiency of a PCI Express Link

There is more overhead to be considered in addition to the framing overhead. “Switch utilization” is the “switch throughput” described up until this point, less the overhead associated with the PCIe protocol infrastructure. Examples of this type of overhead traffic are TLPs containing no user data (messages related to interrupts, errors, hot plug, power management or vendor defined messages) and eight types of DLLPs (Ack/NAK, flow control, etc.). This overhead is variable in nature and can sometimes be fine tuned to meet system requirements by modifying the switch settings. Examples of such settings are, the ratio of ACK/NAKs to total packets, frequency of flow control updates, etc. In general, one can expect this overhead to be up to as much as 15% of switch throughput in several real life systems. So, for example, in a x8 link across the switch, for 64 byte payload size, starting from raw bits entering the switch as the base count, 20% is lost in 8b/10 encoding, 25% is lost in framing overhead and approximately 15% may be lost in other protocol overhead as described above.

A pictorial representation of the impact of this additional overhead is shown in Figure 2. This is similar to Figure 2 but also adds another line to the chart showing the effect of the additional DLLPs. The assumption here is that there are two DLLPs of 8 bytes each sent for every 4 TLPs. This equates to 16 bytes worth of DLLPs per 4 TLPs, or on average 4 bytes of DLLP overhead per TLP. This adds to the 20 bytes of framing overhead used previously as an example.

Clearly, the impact of fixed overheads such as these is minimized when the payloads are larger than 256 bytes.

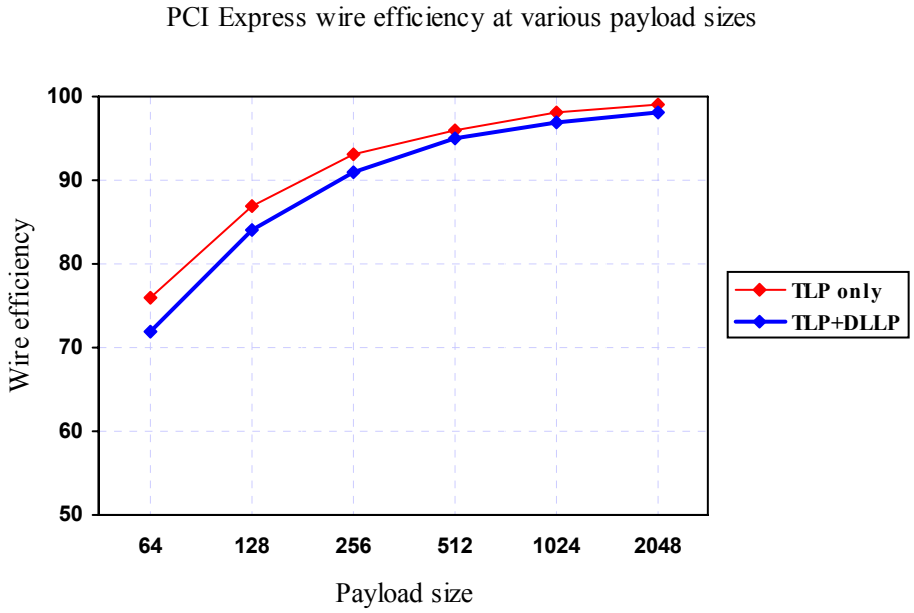

Figure 4 Effect of DLLP Overhead on Data Path Efficiency of a PCI Express Link

Latency

A different indicator of the performance of a switch is the switch "latency", which is defined as the time spent by a bit within the switch from the moment it enters the switch to the moment it exits. The latency number, typically low hundreds of nanoseconds, can be affected by several parameters including, but not limited to, switch architecture, traffic pattern, state of the switch in terms of loading, width of the ingress port, and width of the egress port.

It is crucially important to understand what matters and what does not matter when it comes to selecting a PCIe switch on the basis of latency. In general there is little correlation between the latency of a switch and the total throughput it can sustain across all its ports at the same time, which is the metric that truly matters for any system performance. An uninformed chase for a switch with the lowest latency number supplied by a switch vendor can inevitably lead to a wrong decision if no attention is paid to other performance metrics of a switch. Here is why...

Focus on the port width that matters to the application:

Some switch vendors tend to mislead customers by providing latency numbers which can only be realized when the switch is configured for the largest port width a switch can offer. In general, wider the port width, lower the latency. For a 24 lane switch, a vendor may offer a low latency number for data passing through the switch from an 8 lane ingress port to an 8 lane egress port. This information is worthless if your application requires data to move form a 8 or 4 lane port to a 4 lane port. The key is to focus on latency for the port widths actually required by your application.

Focus on simultaneous multi-port activity:

If your application requires data to flow simultaneously between several ports of the switch, what matters is the total latency experienced by the last packet within the set of packets attempting to pass through the switch at the same time. A 6 port switch may have up to 6 different packets trying to get through the switch at the same instance, one from each port. If a vendor provides the latency for one data transfer across an empty switch, that information is worthless in a scenario such as this. Insist on the total latency for the last bit of the data attempting to go across the switch in a fully loaded condition.

Impact of Architecture on Switch Performance

Now that the question of what performance means is understood, and what to expect from the PES24T6 is clear, some basic inquiry into how this performance is achieved is in order.

Two high-level architectural decisions which will have the biggest impact on switch performance are “how” the data is forwarded from one port to the other within a switch and “when” the data is forwarded. System designers must make these decisions at the very beginning of the design process.

The architectural choices available for the “how to forward” question are: Shared bus, Crossbar, and Shared memory, or a hybrid of some combination of the above. The PES24T6 is implemented in a shared bus style architecture. Explanation of these different types of switching architectures is beyond the scope of this document.

The architectural choices available for the “when to forward” question are: Cut-through (start forwarding a packet while it is being received) or Store and Forward (start forwarding only after an entire packet is received). The PES24T6 uses the Cut-through forwarding method and can fall back to store and forward mechanism when situations warrant such behavior. For example, when data travels from a narrow (x4) port to a wider (x8) port, the protocol requires that the transfer occur only in store-and-forward mode.

There are several other micro-architectural features or implementation details of a switch that can also have noticeable impact on the performance of a switch. Discussion of the relationship between a feature choice and its impact on performance are beyond the scope of this document. It is relevant to note that several implementation details, such as the transmit retry buffer sizes, ingress buffer sizes, flow control mechanism, allowable maximum payload size (MPS), and controllable frequency of DLLPs including flow control updates and ACK/NACK, have an impact on the performance of the switch. Specifications related to these implementation details for the PES24T6 are found in the 89HPES24T6 User Manual, available by contacting IDT.

SECTION II: GE Throughput Measurements

The goal of this set of tests is to demonstrate the behavior of the PES24T6 with Gigabit Ethernet endpoint devices. Test results are obtained both with and without the PES24T6 device in the data path, so as to measure the impact of the switch on data throughput.

Hardware Setup

Following is a list of system components used for this test:

- Tyan Thunder K8QE (S4885) motherboard
  - Four (quantity) - AMD Opteron 852 CPUs (64-bit)
  - 4GB of DDR-RAM
  - 4 available PCIe slots - two x16 and two x4
- Fedora Core 3 - Linux Kernel 2.6.9 - 1.667 SMP
  - Intel e1000 driver 7.2.7
- Windows 2003 Server
  - Intel Pro/1000 Drivers - 9.24
- IDT PES24T6 - x8 upstream, using three x4 downstream ports
  - Max Payload Setting 128 bytes
- Intel Pro/1000 PT Dual Port Server Adapter [x4 PCIe] Ethernet Controllers

Figure 5 is a logical representation of the hardware setup used for GE throughput measurements with the PES24T6. In this case, only one PCIe slot on the motherboard is used.

The NICs are plugged into the downstream port slots of the PES24T6 evaluation board (89EBPES24T6) hosting the PES24T6 switch. The upstream port of the PES24T6 is at the x8 edge connector of the PES24T6 evaluation board and is plugged into a x8 port slot of the motherboard. The PES24T6 switch uses one PCIe slot on the motherboard and creates a fan-out of three slots where three GE NIC endpoints can be used in this system.

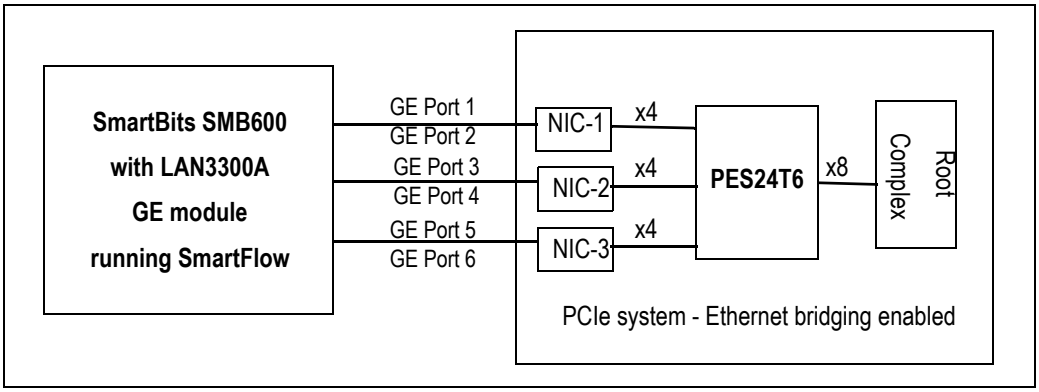

Figure 5 GE Throughput Measurement Setup with the PES24T6

Figure 6 is a logical representation of the hardware setup used for GE throughput measurements without the PES24T6 in the data path. In this setup, three PCIe slots on the motherboard are used by the endpoints since the fan-out provided by the PCIe switch is no longer available. The GE NIC cards are plugged directly into the PCIe slots on the motherboard.

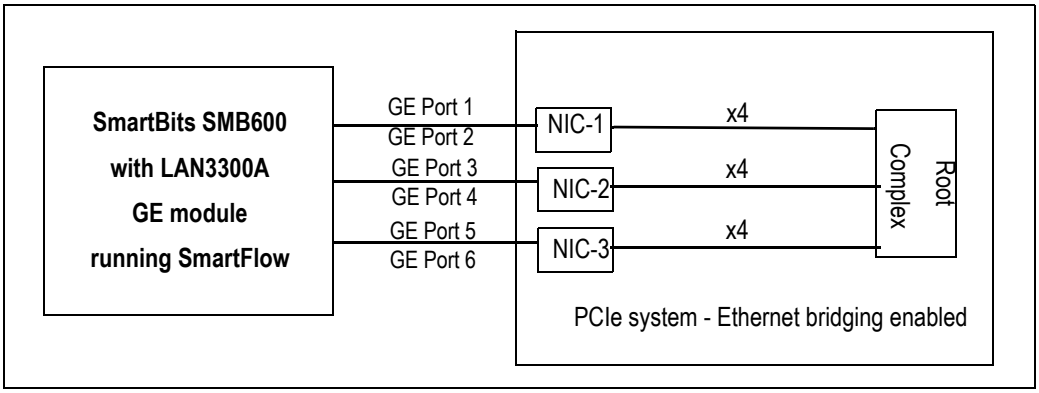

Figure 6 GE Throughput Measurement Setup without the PES24T6

Software Setup

The SmartBits 600 Gigabit Ethernet traffic generator is controlled by the SmartFlow software package to generate and sink Ethernet traffic in a loopback mode. Details related to SmartBits setup can be found in Appendix A. The PCI Express-enabled server system is controlled by the operating system and implements bridging of Ethernet traffic from one Ethernet port to another.

Test Procedure and Methodology

Each port of the SMB600 transmits Ethernet packets of predefined sizes targeted at another port. Each packet transmitted by Port 1 travels through the corresponding NIC in the PCIe system, through the PCIe switch, if present, through the memory in the PCIe system, gets bridged over to Port 2 via the PCIe switch, if present, and returns to Port 2 of the SMB600. Packets starting at Port 2 of the SMB600 traverse the exact opposite path described above. Ports 3 and 4 have the same relationship as do Ports 5 and 6. Combined throughput measurements of these six flows for each packet size, with and without the PCIe switch in the path, are recorded in Tables 1 and 2 below. No data loss is permitted along the entire data path in either direction.

Results

	Throughput in Megabits/Second						
Packet size (bytes)	**64**	**128**	**256**	**512**	**1024**	**1280**	**1518**
Mbits/S Without PES24T6	60	94	195	397	811	895	1191
Mbits/S With PES24T6	60	127	364	566	980	1075	1191

Table 1 Throughput versus Ethernet Packet Size With and Without PES24T6 — Windows

	Throughput in Megabits/Second						
Packet Size (bytes)	**64**	**128**	**256**	**512**	**1024**	**1280**	**1518**
Mbits/S Without PES24T6	467	727	1317	2541	4523	5789	6000
Mbits/S With PES24T6	431	727	1359	2456	4523	5789	6000

Table 2 Throughput versus Ethernet Packet Size With and Without PES24T6 — Linux

Analysis

The goal of this test is to show the effect of the PES24T6 PCIe switch on Ethernet traffic throughput. A quick review of the results reveals that the bridging performance of the operating system impacts how stressful the test will be for the PCIe switch under test. It is clear that Linux offers better Ethernet bridging performance and, therefore stresses the switch more than Windows.

Generally speaking, the presence of the PES24T6 has no adverse effect on performance. In some cases, the presence of the PES24T6 actually increases the throughput. The buffering capabilities of the PCIe switch allows the endpoints and root complex to send larger bursts of data than they otherwise would. Consequently, they are more frequently transmitting data and less frequently waiting to transmit. Another benefit is the coalescing of ACK messages, which frees the return path for data transmissions.

SECTION III: Fibre Channel Throughput Measurements

The goal of this set of tests is to demonstrate the behavior of the PES24T6 with FC storage controllers as endpoint devices connected to the PES24T6 downstream ports.

4 Gbps FC cards are used on three of the downstream ports. Reads and writes to an array of disk drives are done with the IOmeter software tool. IOmeter is also used to gather and analyze the performance data.

Hardware Setup

Following is a list of system components used for this test:

- SuperMicro X6DH8-G2 motherboard
  - Dual Intel Xeon 2.8 GHz
  - Intel E7520 (Lindenhurst) North Bridge
  - 1GB RAM
  - PCIe slots: Two x8 and one x4
- Windows 2003 Server
- IDT PES24T6 - x8 upstream, one x8 and two x4 downstream ports
- Two HPFC-6600A and one HPFC-6400 Fibre Channel HBAs, 4 Gbps FC ports
- JMR Marlin FC-to-SAS JBOD Chassis (8 SAS drives each)

The FC controller cards were plugged into the downstream port slots of the PES24T6 evaluation board hosting the PES24T6 switch. A total of 8 hard drives were connected to each JMR Marlin Chassis. The upstream port of the PES24T6 is at the upstream edge connector of the PES24T6 evaluation board and is plugged into a x8 port slot of the motherboard. In this way, the PES24T6 switch consumes one PCIe slot on the motherboard and creates a fan-out of three slots where three FC controller cards can be used. Figures 7 and 8 illustrate the system setups used in our testing.

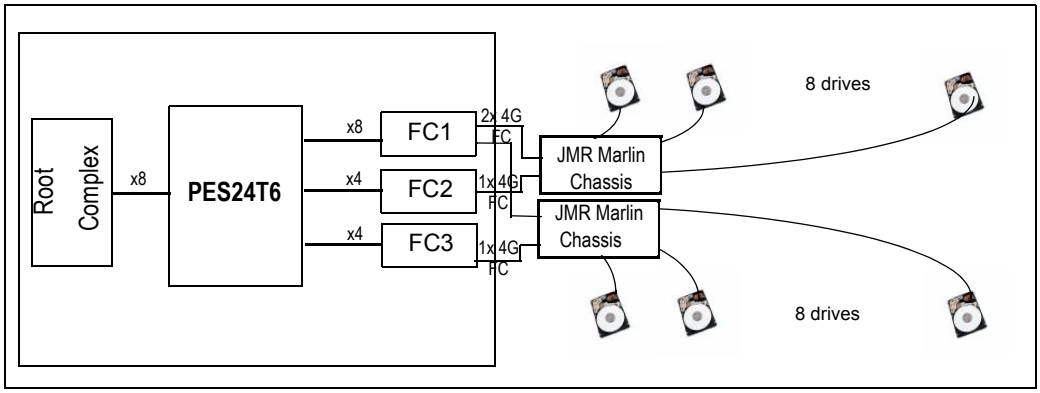

Figure 7 FC Throughput Measurement Setup with the PES24T6

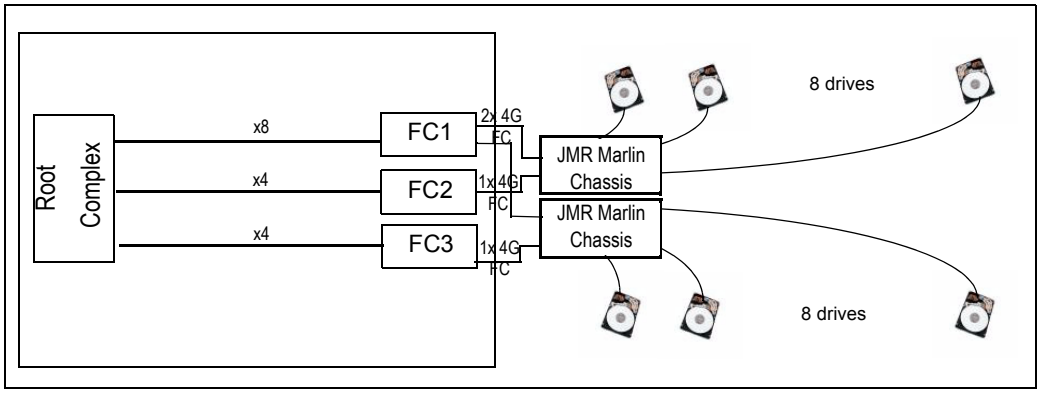

Figure 8 FC Throughput Measurement without the PES24T6

Software Setup

For each of the FC controller cards, the 8 drives connected to it were configured as a single 1280MB logical drive. Traffic was generated and measurements were taken using the IOmeter software package running on the PCIe host system under Windows. Details related to IOmeter software package can be found in Appendix B. IOmeter version 2004.07.30 was used.

Test procedure and Methodology

IOmeter settings were as follows:

- **100% sequential or random "reads"**

  Sequential 310KB transfers set as 100% Read & 0% Write; 1 Manager (Dynamo) with 4 Workers and 16 Logical Drives (1168GB). Run time was 15 seconds.

- **100% sequential or random "writes"**

  Sequential 2MB transfers set as 0% Read & 100% Write; 1 Manager (Dynamo) with 4 Workers and 16 Logical Drives (1168GB). Run time was 15 seconds.

- **67% "reads" and 33% "writes"**

  Sequential 310KB transfers set as 100% Read & 0% Write with 67% of Access and Sequential 64KB transfers set as 0% Read & 100% Write with 33% of Access; 1 Manager (Dynamo) with 4 Workers and 16 Logical Drives (1168GB). Run time was 15 seconds.

- **50/50% Random "reads" and "writes"**

  Equal amount of Read and Write random access; 1 Manager (Dynamo) with 4 Workers and 16 Logical Drives (1168GB). Run time was 15 seconds.

Results

Type of Workload	Throughput with PES24T6 (MB/sec)	Thereabout without PES24T6 (MB/sec)
100% sequential "reads"	1528	1529
100% sequential "writes"	1357	1566
67% "reads" and 33% "writes"	875	876
50/50% Random "reads" and "writes"	858	862

Table 3 FC Throughput with and without PES24T6

Analysis

The switch performs well under most common mix of real life data transfer mix of Reads and Writes.

Appendix A Introduction to SmartBits and SmartFlow

Note: Information contained in this section pertains to tools offered by a third party. The information is provided for the convenience of the reader and is not guaranteed to be complete or accurate.

The following document was used for reference while generating this text: Spirent Communications, Inc., 2005. “Introducing SmartFlow.” SmartFlow User Guide (5.0).

SmartFlow is a performance analysis tool to test Layers 2, 3, and 4 on Class of Service devices and networks built with Class of Service priority strategies. SmartFlow allows the setup of multiple flows of IP frames to simulate network traffic and measures latency, frame loss, and throughput. It presents results in charts and tables that include measurements for latency, frame loss, and standard deviation of flows. Results can be tracked by priority or by type of traffic to determine the effect a prioritizing Class of Service device has on the network.

Since our primary goal was to measure throughput through the PCI Express switch, we used the SmartFlow Group Wizard to simply generate flows, track them, and group them. SmartFlow is used in conjunction with a Spirent Communications SmartBits chassis and at least two SmartMetrics or TeraMetrics (or TeraMetrics-based) ports.

SmartFlow includes the following tests:

- Throughput
- Frame Loss
- Latency
- Latency Distribution
- Latency Snap Shot
- Smart Tracker

Below is a general description of the tests that were used for our measurements.

Throughput

Measures the maximum rate at which frames from flows and groups can be sent through a device without frame loss. A sequence of transmissions from one port on the SmartBits chassis to the other port on the chassis is setup. This traffic flows through the device under a test (PCI Express switch) which has Ethernet NICs connected to its downstream ports. An OS-based bridge is created between these two NIC, causing traffic entering one NIC to get forwarded to the other NIC. Bidirectional traffic is used, and each test consists of several sequential transmissions of Ethernet packets varying in size from 64 bytes to 1518 bytes with each type of packets getting transmitted in a single flow for several seconds at a time.

SmartFlow and SmartFlow Demos are available at **support.spirentcom.com**. Path: Self Service Tools –> Download Software Updates –> All Software –> SmartBits –> Applications or Demo. It is necessary to obtain a support account from Spirent to login to this site.

Appendix B: Introduction to IOmeter

Note: Information in this section pertains to tools offered by a third party. The information is provided for the convenience of the reader and is not guaranteed to be complete or accurate.

The following document was used for reference while generating this text: Intel Corporation: Iometer User's Guide December 16, 2003, which is available at:

http://cvs.sourceforge.net/viewcvs.py/*checkout*/iometer/iometer/Docs/Iometer.pdf

The latest version of Iometer, including the documentation, can be obtained from the Iometer project Web Site at the following URL: http://www.iometer.org/

An Iometer is an I/O subsystem measurement and characterization tool for systems. It is both a *workload generator* (that is, it performs I/O operations in order to stress the system) and a *measurement tool* (that is, it examines and records the performance of its I/O operations and their impact on the system). It can be configured to emulate the disk or network I/O load of any program or benchmark, or it can be used to generate entirely synthetic I/O loads. It can generate and measure loads on single or multiple (networked) systems.

An Iometer can be used for the measurement and characterization of:

- Performance of disk and network controllers.
- Bandwidth and latency capabilities of buses.
- Network throughput to attached drives.
- Shared bus performance.
- System-level hard drive performance.
- System-level network performance.

The Iometer tool consists of two programs, *Iometer* and *Dynamo*.

Iometer is the controlling program. Using the Iometer's graphical user interface, you configure the workload, set operating parameters, and start and stop tests. Iometer tells Dynamo what to do, collects the resulting data, and summarizes the results in output files. Only one copy of Iometer should be running at a time; it is typically run on the server machine in which the devices under test are plugged.

Dynamo is the workload generator. It has no user interface. At Iometer's command, Dynamo performs I/O operations and records performance information, then returns the data to Iometer. There can be more than one copy of Dynamo running at a time; typically one copy runs on the server machine and one additional copy runs on each client machine.

Dynamo is multithreaded; each copy can simulate the workload of multiple clients programs. Each running copy of Dynamo is called a *manager*; and each thread within a copy of Dynamo is called a *worker*. A system can simulate stress conditions by deploying several managers and workers. The worst case combination can be determined by experimenting and noting the results.

Once the IOmeter program has been started, a screen similar to that in Figure 9 is displayed. The "Results Display" tab displays performance statistics while a test is running. A user can choose which statistics are displayed, which managers or workers are included in a particular run of the test, and how often the display is updated in real time. A user can change the settings of all controls in the Results Display tab while the test is running. Changes take immediate effect.

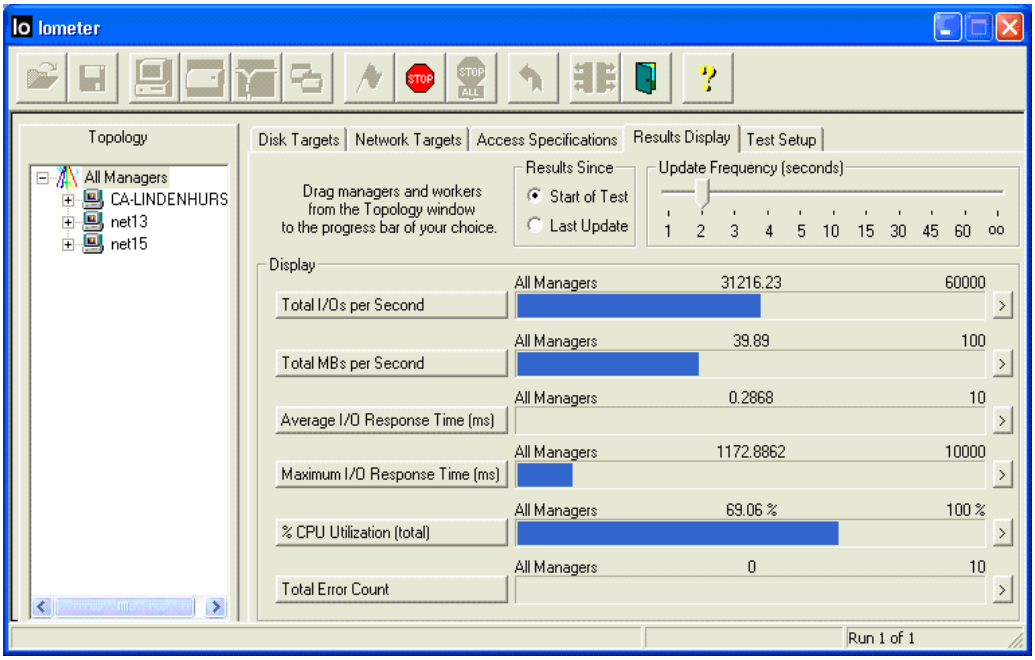

Figure 9 IOmeter Main Screen

As seen in Figure 9, six performance metrics can be displayed as bar charts on the program screen at one time. There are seven main categories of performance metrics. Within each main category, there are several sub-categories of more refined information. Any six out of these large number of metrics can be displayed and tracked as charts in real time. At the end of a test run, results of all sub-categories can be saved as tabulated text files in various formats.

The following is a list of the main metrics and sub-metrics within each main metric.

Operations per Second

- Total I/Os per Second
- Read I/Os per Second
- Write I/Os per Second
- Transactions per Second
- Connections per second

Megabytes per Second

- Total MBs per Second
- Read MBs per Second
- Write MBs per Second

Average Latency

- Average I/O response time (ms)
- Average Read response time (ms)
- Average Write response time (ms)
- Average Transaction time (ms)
- Average Connection time (ms)

Maximum Latency

- Maximum I/O response time (ms)
- Maximum Read response time (ms)
- Maximum Write response time (ms)
- Maximum Transaction time (ms)
- Maximum Connection time (ms)

CPU

- % CPU utilization (Total)
- % User time
- % Privileged time
- % DPC time
- % Interrupt time
- Interrupts per Second
- CPU effectiveness

Network

- Network packets per Second
- Packet Errors
- TCP segments retransmitted per Second

Errors

- Total Error Count
- Read Error Count
- Write Error Count