



PCI Express and FPGAs

Why FPGAs are the best platform for building PCI Express endpoint devices.

by Alex Goldhammer

Technical Marketing Manager, Platform Solutions
Xilinx, Inc.

alex.goldhammer@xilinx.com

PCI Express is a high-speed serial I/O interconnect scheme that employs a clock data recovery (CDR) technique. The PCI Express Gen1 specification defines a line rate of 2.5 Gbps per lane, allowing you to build applications that have a throughput of 2 Gbps (after 8B/10B encoding) for a single-lane (x1) link to 64 Gbps for 32 lanes. This allows a significant reduction in pin count while maintaining or improving throughput. It also reduces the size of the PCB, the number of traces and layers, and simplifies layout and design. Fewer pins also translate to reduced noise and electromagnetic interference (EMI). CDR eliminates the clock-to-data skew problem prevalent in wide parallel buses, making interconnect implementations easier.

The PCI Express interconnect architecture is primarily specified for PC-based (desktop/laptop) systems. But just like PCI, PCI Express is also quickly moving into other system types, such as embedded systems. It defines three types of devices: root complex, switch, and endpoint (Figure 1). The CPU, system memory, and graphics controller connect to a root complex, which is roughly equivalent to a PCI host. Because of PCI Express' point-to-point nature, switch devices are necessary to expand the number of system functions. PCI Express switch devices connect a root complex device on the upstream side to endpoints on the downstream side.

Endpoint functionality is similar to a PCI/PCI-X device. Some of the most common endpoint devices are Ethernet controllers or storage HBAs (host-bus adapters). Because FPGAs are most frequently used for data processing and bridging functions, the largest target function for FPGAs is endpoints. FPGA implementations are ideally suited for video, medical imaging, industrial, test and measurement, data acquisition, and storage applications.

The PCI Express specification maintained by the PCI-SIG mandates that every PCI Express device use three distinct proto-

col layers: physical, data-link, and transaction. You can build a PCI Express endpoint using a single- or two-chip solution. For example, you can use a low-cost FPGA such as a Xilinx® Spartan™-3 device for building data-link and transaction layers with a commercially available discrete PCI Express PHY (Figure 2). This option is best suited for x1 lane applications such as bus controllers, data-acquisition cards, and performance-boosting PCI 32/33 devices. Or you can use a single-chip solution such as Virtex™-5 LXT or SXT FPGAs, which have an integrated PCI Express PHY. This option is best for communications or high-definition audio/video endpoint devices (Figure 3) that require higher performance of x4 (8-Gbps throughput) or x8 (16-Gbps throughput) links.

Before selecting a technology for implementing a PCI Express design, you must carefully consider the choice of IP, link efficiency, compliance testing, and availability of resources for the application. In this article, I'll review the factors for building single-chip x4- and x8-lane PCI Express designs with the latest FPGA technology.

Choice of IP

As a designer, you can choose to build your own soft IP or buy IP from either a third party or an FPGA vendor. The challenge of building your own IP is that not only do you have to create the design from scratch, you also have to worry about verification, validation, compliance, and hardware evaluation. IP purchased from a third party or FPGA vendor will have gone through all of the rigors of compliance testing and hardware evaluation, making it plug and play. When working with a commercially available, proven, compliant PCI Express interface, you can focus on the most value-added part of the design: the user application. The challenge of using soft IP is the availability of resources for the application. As the PCI Express MAC, data-link, and transaction layers in soft IP cores are implemented using programmable fabric, you must pay special attention to the number of remaining block RAMs, look-up tables, and fabric resources.

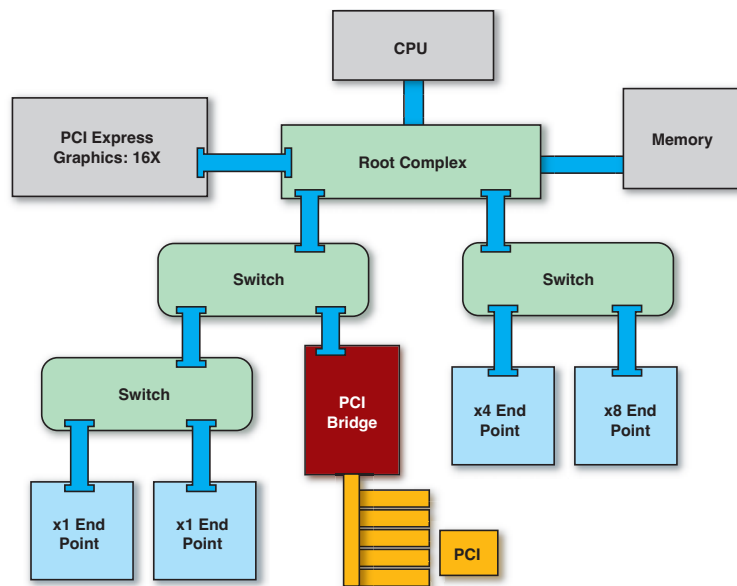


Figure 1 – PCI Express topology

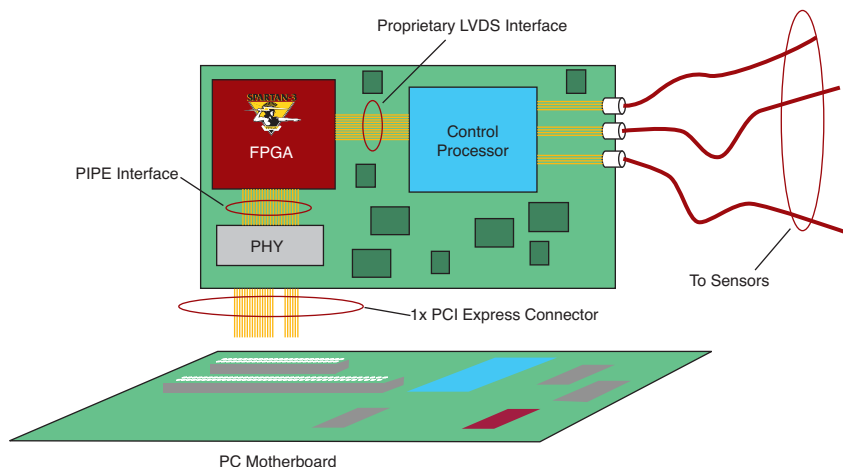


Figure 2 – Spartan-3 FPGA-based data-acquisition card

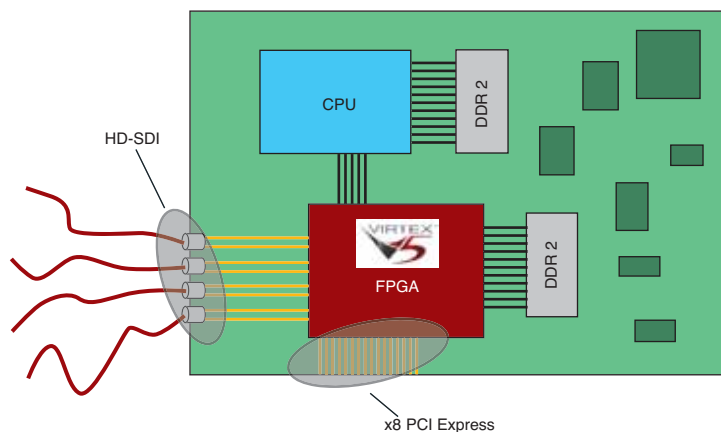


Figure 3 – Virtex-5 LXT FPGA-based video application

Another option is to use an FPGA with the latest technology. The Virtex-5 LXT and SXT have an integrated x8-lane PCI Express controller implemented in dedicated gates (Figure 4). This type of implementation is very advantageous, as it requires a minimal number of FPGA logic resources because the design is implemented in hard silicon. For example, in the Virtex-5 LXT FPGA, an x8-lane soft IP core can consume up to 10,000 logic cells, while the hard implementation will need about 500 logic cells, mostly for interfacing. This resource savings sometimes allows you to choose a smaller device, which is generally cheaper. Integrated implementations also have typically higher performance, wider data paths, and are software-configurable.

Another challenge with soft IP implementations is the number of features. Typically, such cores only implement the minimum features required by the specification to meet performance or compliance goals. Hard IP, on the other hand, can support a comprehensive feature list based on customer demand and full compliance (Table 1). There are no major performance or resource-related issues.

Latency

Although the latency of a PCI Express controller will not have a huge impact on overall system latency, it does affect the performance of the interface. Using a narrower data path helps latency.

For PCI Express, latency is the number of cycles it takes to transmit a packet and receive that packet across the physical, logical, and transaction layers. A typical x8-lane PCI Express endpoint will have a latency of 20-25 cycles. At 250 MHz, that translates into 80-100 ns. If the interface is implemented with a 128-bit data path to make timing easier (such as 125 MHz), the latency doubles to 160-200 ns. Both the soft and hard IP implementations in the latest Virtex-5 LXT and SXT devices implement a 64-bit data path at 250 MHz for x8 implementations.

Link Efficiency

Link efficiency is a function of latency, user application design, payload size, and overhead. As payload size (commonly referred to as maximum payload size) increases, the

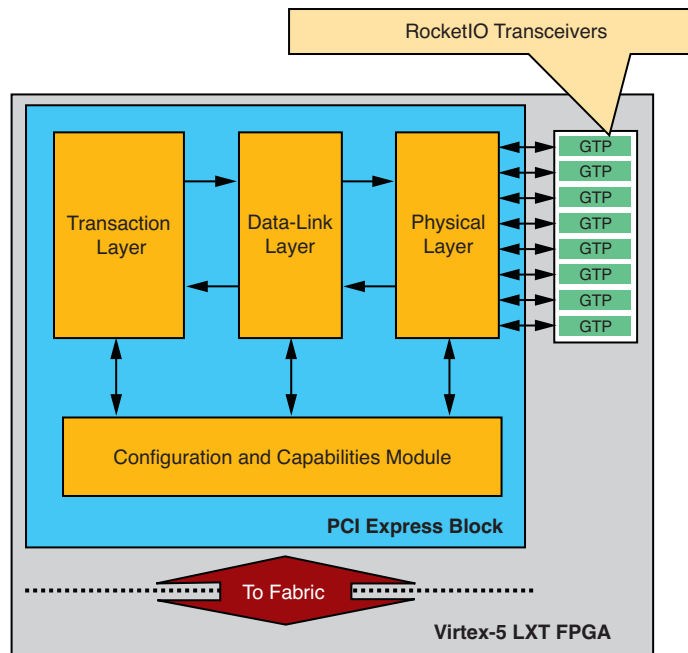


Figure 4 – Virtex-5 LXT FPGA PCI Express endpoint block diagram

Performance			
Lane Width	Interface Data Width	Interface Speed	Bandwidth (each direction)
x1	64	62.5/125/250 MHz	2 Gbps
x2	64	62.5/125/250 MHz	4 Gbps
x4	64	125/250 MHz	8 Gbps
x8	64	250 MHz	16 Gbps

PCI Express Specification v1.1 Compliance	
Requirement	Support (Y/N)
Clock Tolerance (300 ppm)	Y
Spread-Spectrum Clocking	Y
Electrical Idle Generate and Detect	Y
Hot Plug	Y
De-Emphasis	Y
Jitter Specifications	Y
CRC	Y
Automatic Retry	Y
QOS	2 VC/round robin, weighted round robin, or strict priority
MPS	128-4096 bytes
BARs	Configurable 6 x 32 bit or 3 x 64 bit for memory or I/O
Required Power Management States	Y

Table 1 – Virtex-5 LXT FPGA PCI Express capabilities

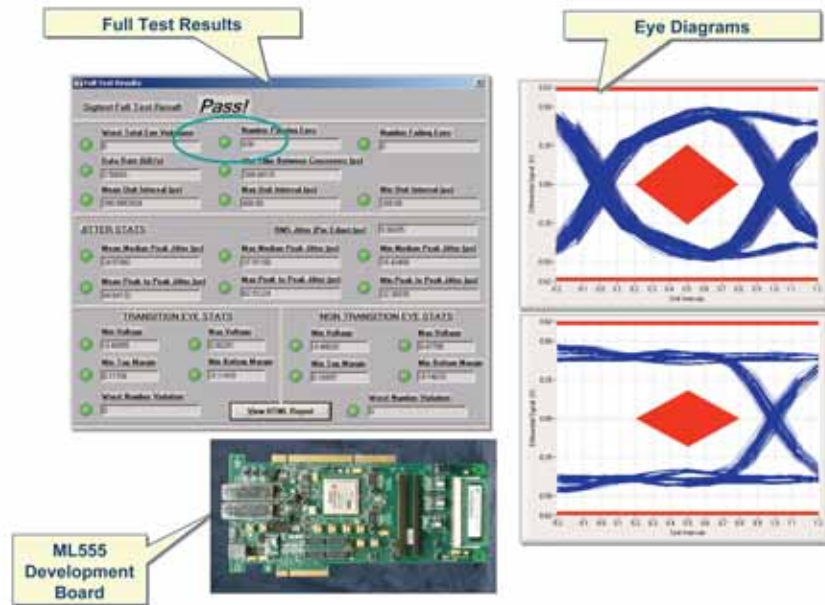


Figure 5 – Virtex-5 LXT FPGA PCI Express compliance workshop results

effective link efficiency also increases. This is caused by the fact that the packet overhead is fixed; if the payload is large, the efficiency goes up. Normally, a payload of 256 bytes can get you a theoretical efficiency of 93% (256 payload bytes + 12 header bytes + 8 framing bytes). Although PCI Express allows packet sizes up to 4 KB, most systems will not see improved performance with payload sizes larger than 256 or 512 bytes. A x4 or x8 PCI Express implementation in the Virtex-5 LXT FPGA will have a link efficiency of 88-89% because of link protocol overhead (ACK/NAK, re-transmitted packets) and flow control protocol (credit reporting).

Using FPGAs for implementation gives you better control over link efficiency because it allows you to choose the receive buffer size that corresponds to the endpoint implementation. If both link partners do not implement the data path in a similar way, the internal latencies on both will be different. For example, if link partner #1 uses the 64-bit, 250-MHz implementation with a latency of 80 ns and link partner #2 uses the 128-bit, 125-MHz implementation with a latency of 160 ns, the combined latency for the link will be 240 ns. Now, if link partner #1's receive buffer was designed for a latency of 160 ns – expecting that the link partner would also be a 64-bit, 250-MHz implementation – then the link efficiency would

go down. With an ASIC implementation, it would be impossible to change the size of the receive buffer, and the loss of efficiency would be real and permanent.

User application design will also have an impact on link efficiency. The user application must be designed so that it drains the receive buffer of the PCI Express interface regularly and keeps the transmit buffer full all the time. If the user application does not use packets received right away (or does not respond to transmit requests immediately), the overall link efficiency will be affected regardless of the performance of the interface.

When designing with some processors, you will need to implement a DMA controller if the processors cannot perform bursts longer than 1 DWORD. This translates into poor link utilization and efficiency. Most embedded CPUs can transmit bursts

longer than 1 DWORD, so the link efficiency for such designs can be effectively managed with a good FIFO design.

PCI Express Compliance

Compliance is important detail that is frequently missed and often undervalued. If you are building PCI Express applications that must work with other devices and applications, ensuring that your design is compliant is a must.

The compliance is not just for the IP but for the entire solution, including the IP, user application, silicon device, and hardware board (Figure 5). If the entire solution has been validated at a PCI-SIG PCI Express compliance workshop (also known as a “plug fest”), it is pretty much guaranteed that the PCI Express portion of your design will always work.

Conclusion

Replacing PCI, PCI Express has become the de facto system interconnect standard and has jumped from the PC into other system markets including embedded system design. FPGAs are ideally suited to build PCI Express endpoint devices, as they allow you to create compliant PCI Express devices with the added customization that embedded users desire.

New 65-nm FPGAs like the Virtex-5 LXT and SXT families are fully compliant to the PCI Express specification v1.1 and offer an abundance of logic and device resources to the user application. The Spartan-3 family of FPGAs with external PHY offer a low-cost solution. These factors, combined with the inherent programmable logic advantages of flexibility, reprogrammability, and risk reduction, make FPGAs the best platforms for PCI Express. ●●

TAKE THE NEXT STEP (Digital Edition: www.xcellpublications.com/subscribe/)

- Buy the Development Kit for PCI Express:
 - Virtex-5 FPGA edition
 - Spartan-3 FPGA edition
- Download the Protocol Pack for PCI Express
- Watch our live demo from the Embedded Systems Conference
- Register for a PCIe class