

PCI Express Design Considerations

Platform ASIC vs. FPGA Design Efficiency

by Greg Martin, RapidChip Technical Marketing, LSI Logic

Implementing a high-speed PCI-Express core is a complex task, even for the most seasoned engineers. To further complicate matters, the choice of implementation technology can play a significant role in the final design characteristics. When evaluating FPGA and Platform ASIC technologies, there are a number of key considerations.

Smaller Footprint

A typical 8-lane (32Gbps aggregate) PCI Express interface can be implemented with a 64-bit data path running at 250MHz or with a 128-bit data path running at 125MHz. It is extremely difficult to successfully implement any reasonably complicated digital design (with ~20 logic levels) at 150MHz in an FPGA. Reaching anywhere near 250MHz for such designs is not possible, even with the latest 90nm FPGAs. Therefore an x8 PCI Express core implemented in an FPGA will require a 128-bit datapath clocked at 125MHz. By contrast, when implemented in a Platform ASIC, the same core can easily achieve 250MHz allowing the smaller, more efficient 64-bit datapath implementation to be used.

In a Platform ASIC implementation, all of the data paths will be half the width of an FPGA implementation. Since these data paths comprise a large portion of the entire design it will have a major impact on overall gate count. A typical FPGA implementation uses approximately 60% more logic resources than a Platform ASIC implementation.

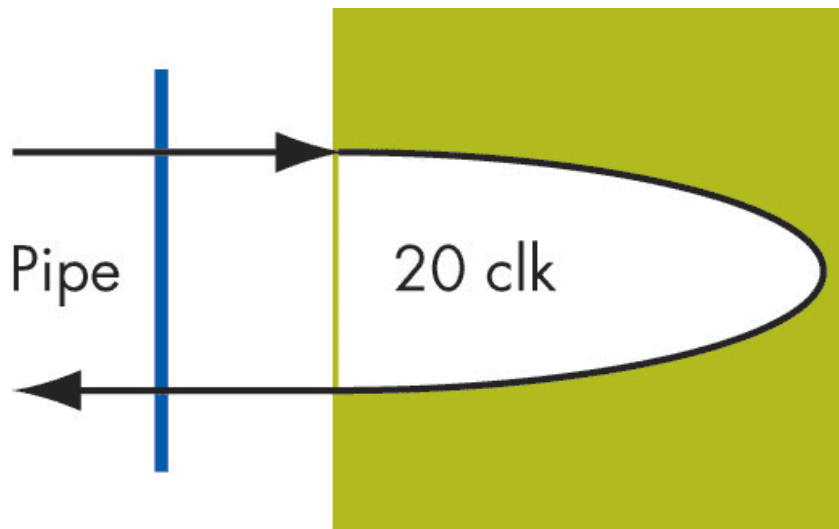
Additionally, buffer sizes in the FPGA implementation are often larger than a Platform ASIC implementation. Not only are the buffers wider, in many cases they must be deeper to cope with latency effects.

Reduced Latency

The latency of a controller greatly influences the overall performance of the

PCI Express interface and thus the entire system. The round trip latency of a design is a very important metric. It is measured from the PIPE Rx to the PIPE Tx, going across the physical, link and transaction layers. A typical PCI Express controller configuration will have ~15- 25 clock cycle round trip latency.

Consider the case of a controller with 20 clock cycles round trip latency. When implemented at 125MHz in an FPGA, the 20-clock cycle latency is $20 \times 8\text{ns} = 160\text{ns}$. The same core implemented at 250MHz in a Platform ASIC has only $20 \times 4\text{ns} = 80\text{ns}$ clock cycle latency. The 100% additional latency suffered by an FPGA implementation is a major reason for the superior performance of Platform ASICs.



Better Link Utilization

The reduced latency of Platform ASICs vs. FPGAs can also translate into superior link utilization. For example, consider the utilization of a PCI Express egress link with a standard-cell ASIC link-partner, in an Intel north bridge system. Figure 2a shows the PCI Express transmit path implemented in a Platform ASIC. Figure 2b shows the PCI Express transmit path implemented in an FPGA.

In the transmit datapath, the PCI Express core sends packets to the standard link partner buffer. When packets leave this buffer, credits are released back to the PCI Express core.

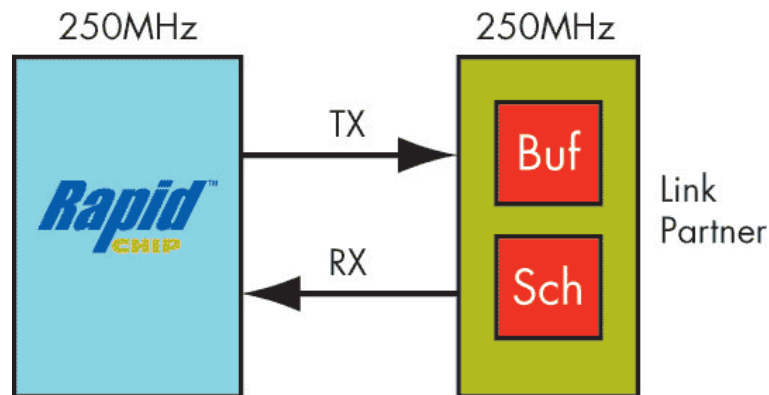


Figure 2a

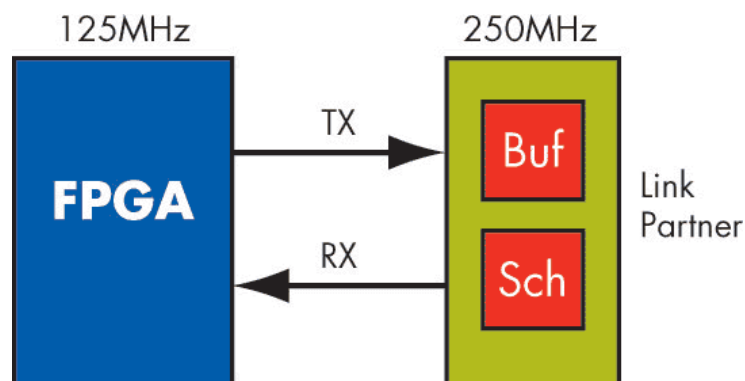


Figure 2b

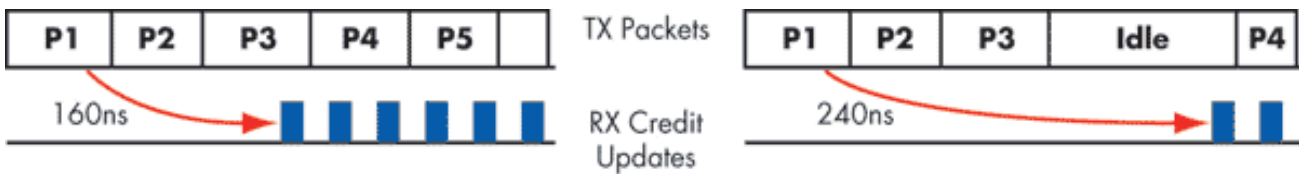
The size of the receive buffer in the link partner and the latency in receiving the credit back to the PCI Express core determines how efficiently the link is utilized.

The fixed size of the Virtual Channel (VC) buffer on the receiving standard-cell ASIC link partner will typically be optimized with the expectation of connection to a similar standard-cell ASIC like device. Thus it will work most efficiently when connected to something with corresponding latency similar to that of a standard-cell ASIC.

If the end-to-end latency involved in sending a packet from the PCI Express core and receiving the credit back is much more than the typical number assumed in the above buffer size estimation then the link will start idling due

to credit starvation. This starvation occurs when the receiving buffer is not large enough to absorb the additional end-to-end latency.

A simple comparison between the Platform ASIC and FPGA implementations is shown in Figures 2c and 2d. This analysis is simplified by excluding the effects of packet size, credit release policy etc. Figure 2c shows how the Platform ASIC implementation continuously sends packets. Its ASIC-like latency allows credits to be received back fast enough to avoid starvation. In contrast Figure 2d shows how an FPGA has to wait much longer for credit updates to occur causing the link to go idle.



This example only considers the case of posted-write packet types, although the effects also apply to other packet types and multi-VC cases. A major component of the credit latency path is the controller’s internal delay. Lets assume the round trip latency inside the link-partner is 20 cycles (at 250MHz). The most significant portion of the end-to-end credit return delay is the sum of the round trip latencies of the both controllers. I.e. 20 x 4 ns for the link-partner plus 20 x 4 ns for the Platform ASIC implemented PCI Express core. This gives a total of 160ns.

The same setup for an FPGA implementation of the PCI Express core will take 20 x 4 ns for the link-partner plus 20 x 8ns for the FPGA, giving a total of 240ns. If the buffer in the link-partner has been designed to cover only the first case latency of 160ns, then the link utilization for the FPGA implementation will be 33% lower.

Reduced Buffer Size

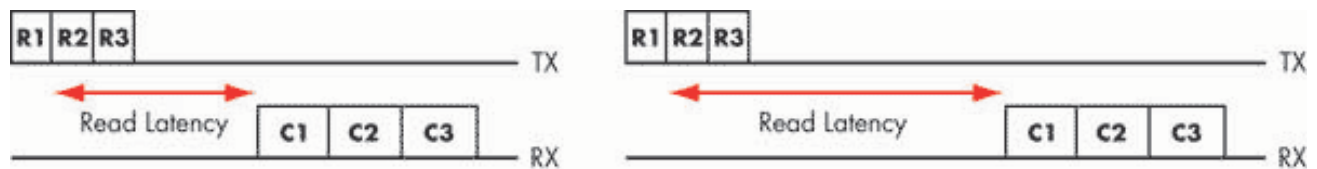
The receive path to a PCI Express core also has similar link utilization considerations.

In an FPGA implementation, the receive VC buffer size must be increased by 50% to absorb the increase in end-to-end latency (240ns instead of 160ns, for the above example). This means the Platform ASIC implementation

requires a reduced buffer size compared with an FPGA implementation. If the FPGA receive buffer size is not increased, the receive path into the PCI Express core will also suffer from utilization problems.

Increased Overall Performance

In addition to the local credit starvation and link utilization issues, the increased latency of an FPGA implementation affects other areas of system performance. Figures 3a and 3b highlight how latency affects the read performance in a system. For a given number of outstanding reads from a node, any increased latency in receiving a response adds significant waiting time for the read initiator. This reduces overall read bandwidth. If the read data contains assembly code to be executed or data-packets to be processed, then the efficiency of such processes will also be significantly reduced.



Conclusion

For complex, high-speed applications such as PCI Express, even the fastest 90nm FPGAs lack sufficient performance. The workarounds to compensate for this lower performance have wide reaching implications on the final system behavior. Both latency and link utilization are degraded in the slower FPGA-based implementation. In many cases this will ultimately slow down the entire system. Additional resources are also required for an FPGA implementation.

The ASIC-like characteristics of a Platform ASIC give it a clear performance advantage over FPGAs. The Latency and Link-utilization of a PCI Express core are similar to that of a standard cell ASIC and therefore optimal for this application. Having these characteristics is especially important when the link-partner is designed for connection to a similarly low-latency partner.

by Greg Martin, RapidChip Technical Marketing, LSI Logic

August 23, 2005