HyperTransport™ CONSORTIUM

WHITE PAPER

Latency Comparison Between HyperTransport™ and PCI-Express™ In Communications Systems

Brian Holden
Vice President and Chair, Technical Working Group, HyperTransport Consortium

Executive Summary

Communication systems with real-time performance and ultra low latency have the potential to transform user experiences and create new business opportunities for network operators and service providers. From faster and more responsive games to converged voice, video and data services, communication systems will be the vehicle for new revenue streams and increased user satisfaction. In particular, compute-centric telecommunications standards, such as the IP Multimedia Subsystem (IMS) specification has placed new demands on communications systems.

Chip-to-chip interconnects directly impact overall communications system performance and can enable a new generation of IP-based services that deliver high quality of service (QoS) and flexible control. Communications system architects will need chip-to-chip interconnects that can handle not only the high bandwidth demands of these systems, but do so in real-time, thus requiring ultra-low latency. As the lowest latency interconnect technology on the market today, HyperTransport enables products that make converged networks, enhanced user experiences and new revenue streams a reality.

Introduction

This article discusses why latency is an important parameter of these interconnects; then contrasts and compares the latency experienced by transactions carried over two of the leading chip-to-chip interconnect standards, HyperTransport and PCI-Express (PCIe). It presents a couple of usage scenarios involving read accesses and derives the latency performance of each.

Why Latency Matters

Only a few years ago, the industry focused on the need to increase bandwidth – a bigger pipe to deliver all types of content including voice, video and data. While advancements have been made to ensure that systems can handle massive quantities of information, more

recently the industry has shifted its focus on the other critical aspect of communications system performance – latency.
Latency in chip-to-chip interconnects can be compared to the responsiveness of the steering of an automobile. The time between the driver's initiation of a turn and when the car is actually headed in the new direction is inversely correlated to the performance of a car in a slalom course. A full-sized luxury car typically has a larger engine than a sports sedan, but that sports sedan will consistently out-perform the luxury car on such a course.

By analogy, many of today's communications computing applications have the compute requirements of a slalom course. In such applications, a series of decisions are made based upon data that is being updated in real-time. Another slalom-like scenario occurs when the selection of the required data is unpredictable enough to defeat caching strategies. An example of such a scenario is an application that follows a large linked-list that is located across the link. In both of these cases, the latency of the chip-to-chip interconnect is a central determinant of overall application performance.

The Scenario We Will Be Analyzing

An important application of HyperTransport is in the CPU subsystem of a communication system. Individual CPUs within multiprocessor systems are connected together and have full access to each other's memory, and can execute programs stored on the memory connected to the other CPUs, rather than a hard disk. Large applications can be divided and portions can be stored on multiple devices, enabling easy, fast access to data. High-performance processor architectures including the AMD® Opteron™ and Broadcom® BCM1480 contain native HyperTransport interfaces that provide point-to-point interconnect capabilities. Both devices include symmetric multi-processing (SMP) extensions to HyperTransport that allow each CPU to have full, coherent access to the other's memory. These SMP extensions allow these CPUs to have full, coherent access to the other's memory. These devices can execute a program that resides in memory that is physically on the other CPUs. Figure 1 provides a detailed overview of a two CPUs connected together.

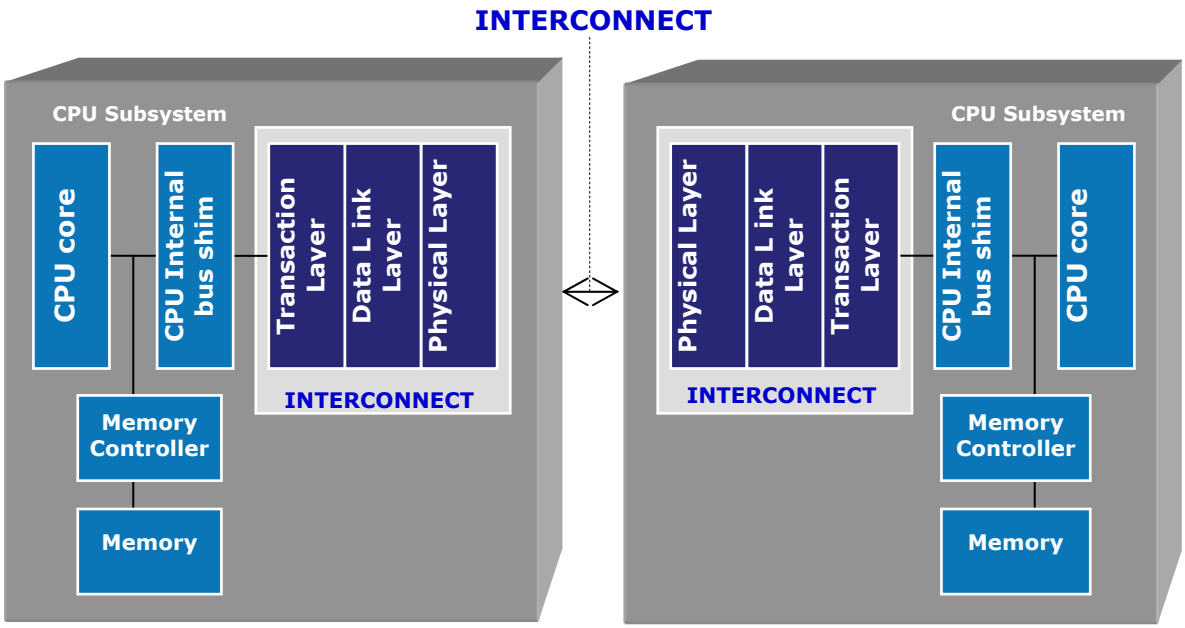

Figure 1: Point-to-Point Architecture Implementation

In Figure 1, the CPU on the left reads data from the CPU on the right via an address range that maps across the bus. Memory reads hit the processor's internal bus and are claimed by the interconnect based on an address match. The CPU interconnect logic appends headers and encodings at the transaction, data link and physical layers, and then transmits the packet across the wire. The packet then moves across the three layers in reverse order to the other CPU, completing the read transaction.

The Impact of Various Factors on Latency

The Impact of Packet Buffering on Latency

Interconnect technologies implement various packet buffering techniques which impact latency. Store-and-forward methods require full packet buffering at various layers within the CPU subsystem – usually at the transport and data link layers. This buffering technique increases latency and slows down system response times.

Low latency designs require transmission techniques that reduce delays between layers, limit the points at which packets are buffered, or eliminate the process of buffering entirely. Cut-through methods allow packets to continue transmission to the next communications layer after the first few bytes of the packet are read. This allows packets to "cut-through" layers without having to fully accumulate prior to being transmitted, as in the case of store-and-forward methods. Cut-through techniques therefore reduce latency and can be applied at both the data link and transaction layers.

Link Layer Cut-Through and Link Reliability

There are different types of cut-through techniques designed to improve performance and lower latency at the link layer. Designers can employ either "fully reversible" or "non-reversible" cut-through. Fully reversible cut-through ensures that all state changes made at the receiving end of the transmission can be fully and completely reversed if an error is detected. Non-reversible cut-through, on the other hand, resets the system or subsystem after error detection. Fully reversible cut-through is safer, but delivers very limited performance benefits, while non-reversible cut-through can dramatically lower latency. The degree to which it is safe to cut through depends on the application and on link reliability.

Clock-forwarded interfaces, such as those used on Synchronous SRAMs (SSRAM), SPI-4.2, and HyperTransport 1.x and 2.0 are often regarded as inherently reliable links. The chance of error with these interfaces is typically much smaller than that of other system error sources. Such designs routinely achieve bit-error-ratios of 10-15 or better. As a result, designers can rely on simple error detection and non-reversible cut-through techniques. If an error is detected, the link or the system is reset. Conversely, serializer/desererializer (SerDes) interfaces are not regarded as inherently reliable, and therefore employ fully reversible cut-through, which in turn introduces a significant latency penalty. It should be noted that HyperTransport 3.0 links will not be regarded as inherently reliable because of their use of data recovery techniques. As a result, reversible cut-through should be employed with HyperTransport 3.0.

Transaction Layer Cut-Through

Many techniques also exist for implementing cut-through at the transaction layer, among them are packet stomping and data poisoning. Both HyperTransport and PCIe support the classic strategies of packet stomping and data poisoning. PCIe has an optional end-to-end CRC check to help expose errors that are made in devices between the links. HyperTransport implements packet stomping by inverting the CRC of a packet that has been received in error. HyperTransport implements data poisoning by supporting a field indicating that a response has been received by an intermediary in error.

Low Latency Physical Layers

The HyperTransport interface has been designed to ensure very low latency physical, data link and transaction layers. A high-performance clock-forwarded architecture – like HyperTransport - delivers low latency at the physical layer. A single forwarded clock is used per set of 8 data path bits along with a skew-constrained PCB. This enables very low latency point-to-point data transfer. Conversely, other chip-to-chip interconnect technologies like PCI Express that use non-skew constrained PCBs and clock-encoded links require large 8B/10B overhead and force the addition of serializing/deserializing deskew logic. The combination of these increases the latency time well beyond ideal levels for high performance, chip-to-chip communication.

Low Latency Data Link and Transaction Layers

Adding to its low latency capability, HyperTransport interconnects also feature very efficient data link and transaction layers due to low packet overhead. In fact, HyperTransport requires just 8 bytes for a write operation and just 12 bytes for a read operation (for a write request, there is an 8-byte write request control packet followed by the data packet; for a read request, there is an 8-byte read request control packet followed by a 4-byte read response control packet and the data packet). This compares very favorably to PCI Express, for example, as shown Figure 2 below.

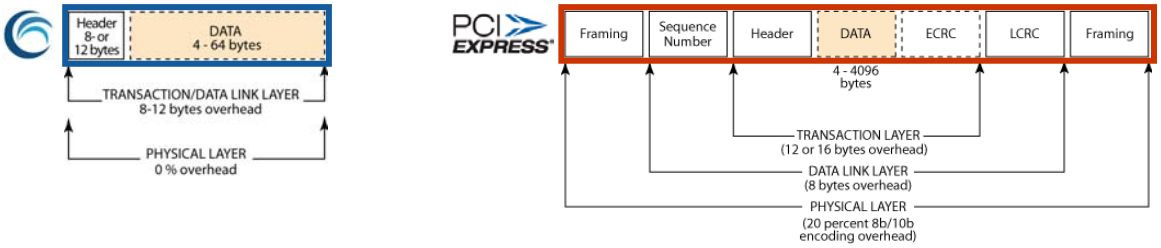

Figure 2: Hyper Transport Packet Format and Overhead

Innovative Features Enable Further Low Latency: Priority Request Interleaving

HyperTransport employs innovative low latency techniques, including its unique Priority Request Interleaving™ (PRI) feature illustrated in Figure 3. This feature enables data packets to be paused and control packets to be inserted on 4-byte boundaries. When the control packet has been inserted, the data packet resumes right where it left off. This is significant in that a Read Request packet is a control packet. Therefore, PRI can reduce latency considerably in certain situations like responding to a cache-miss.

While data transfer 1 is underway between peripheral B and the host, the need arises for peripheral A to start a data transfer from the host. Without PRI, transfer 2 would have to wait until transfer 1 completes and, should transfer 1 be the answer to a cache miss, for instance, latency for transfer 2 would become prohibitive. With PRI, a control packet is promptly inserted within transfer 1's data stream, instructing the link to initiate data transfer 2 on the other link channel concurrently with the completion of data transfer 1. This mechanism, unique to HyperTransport technology, greatly reduces latency of HyperTransport-based systems.

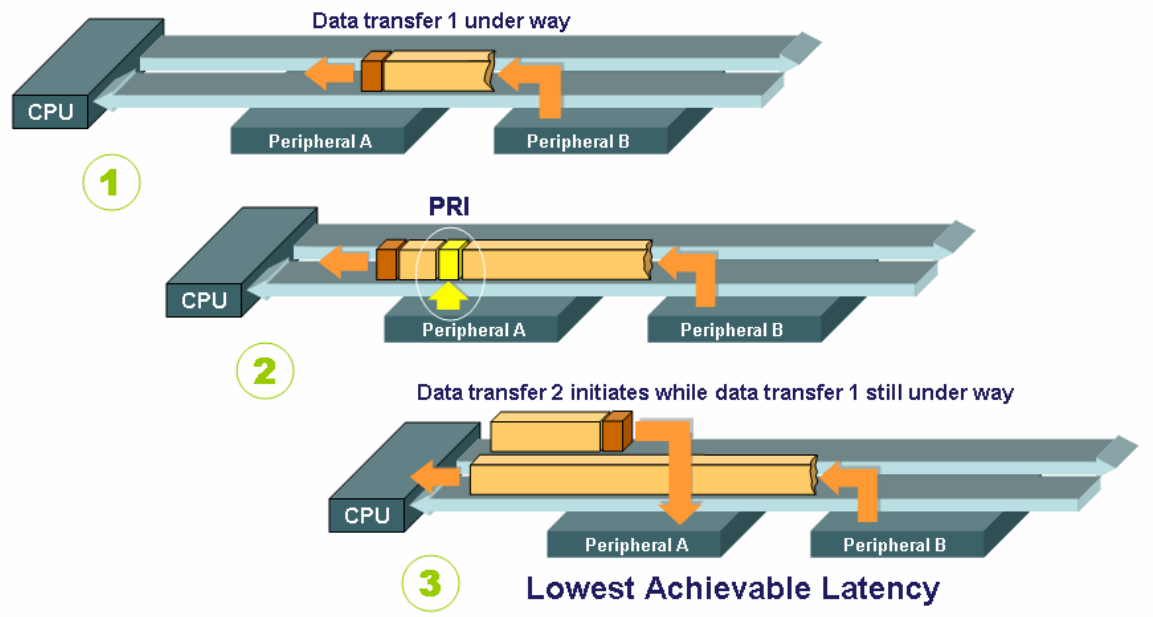

Figure 3: Priority Request Interleaving Feature of Hyper Transport Interface

Latency Effect of Priority Request Interleaving

The latency effect of PRI is probabilistic and highly application dependent. A simple, but useful analysis of that effect is as follows:

1. Assume that the CPU on the right is reading a block of memory from the memory of the CPU on the left.
2. Assume that the left-to-right link is 60 percent utilized with the data from those requests.
3. Assume that the CPU on the left wants to read 8 bytes from the memory of the CPU on the right.
4. 40 percent of the time it will get immediate access to the left-to-right link for the memory read request
5. 60 percent of the time it will conflict with a data packet

PRI in HyperTransport: In read responses, the HyperTransport control packets are 4 bytes long and the data packets are 64 bytes long. HyperTransport's native width is 4 bytes, and PRI allows the data packets to be paused on that native 4-byte boundary. Control packets cannot be paused. At the maximum rate of 2.8 Gb/s, a 16x HyperTransport link transmits 4 bytes every 714 ps.

With PRI, the read request will never have to wait because it will be prioritized ahead of the responses' control packet if they arrive simultaneously.

Without PRI, the added latency would have been equal to the average number of 4-byte intervals remaining in the packet. In this example, 60 percent of the time, the average wait will be (4+64)/(4*2) = 8.5 4-byte intervals. This results in an added latency of 714ps * 8.5 * 60% = 3.6ns.

PRI and PCIe: PCIe lacks a feature like PRI for reducing latency, and latency is further magnified because packet sizes are so much larger. A typical maximum payload setting is 256 Bytes. At the maximum rate of 2.5 Gb/s, a 16x PCIe link transmits 4 bytes every 1000ps.

Without PRI, the added latency in PCIe is equal to the average number of 4-byte intervals remaining in the packet. In this example, 60 percent of the time, the average wait is (22+256)/(4*2) = 34.75 4-byte intervals. This gives an added latency of 1000ps * 34.75 * 60% = 21ns

Latency Analysis of Chip-to-Chip Interconnects in Communications Systems Design

Methodology

Let's now look at an analysis of application latency using HyperTransport and PCIe. Our analysis assesses latency as one CPU reads data from another CPU's memory using the PCIe and HyperTransport interconnects. We'll compare cases with both short packets and long packets. Link speed determines the lag in transmitting the packet across the wire. For comparison, we use interconnects of similar link bandwidth, choosing the x16 link at the maximum available frequency in each case.

Link Speed – Link speed is important in determining the lag in transmitting the packet across the interconnect. For comparison, we used interconnects of similar link bandwidth, choosing the x16 link at the maximum available frequency in each case. For PCI Express, a link speed of 2.5 Gb/s was used, yielding 32.0 Gb/s throughput after line coding. For HyperTransport, the link speed of 2.8 Gb/s was used, yielding 44.8 Gb/s throughput.

Controller Speed - The speed of the link controller is important in determining the lag for the IP core to process data and header fields. For fair comparison, we operated each of the endpoints with a 64-bit width controller running at 333 MHz. Accesses were to open pages in a 333 MHz SDRAM.

Store-and-Forward vs. Cut-Through - Within the tables, the "Store-and-Forward" (S&F) columns show the cases where packets are buffered in each of the Tx Transaction Layer, Tx DataLink Layer, and Rx DataLink Layers. The "Cut-Through" (C-T) columns are optimized to have buffering only in the Rx DataLink Layer; other functions are performed on the fly. Retry-on-error algorithms were enabled in both HyperTransport and PCIe.

Payload Length - For each column, the value of "Max_Payload_Length" was chosen to minimize latency. This value has no effect on this short packet case, but is key for the long packet case below. For the S&F columns, a smaller length was best. For PCIe, the minimum length of 128 bytes was selected. For the C-T columns, a longer length was best, although values above 256 bytes increased the latency due to the buffering in the Rx DataLink Layer. HyperTransport has a fixed maximum packet size of 64 bytes.

PHY Delay - The SerDes+PMA+PCS+MAC rows capture the sum of the latencies experienced by those layers including encoding, scrambling and lane de-skew in each technology.

8-Byte Read Results

Table 1 shows the results in the case of short packets. The CPU reads 8 bytes of data from the other CPU's memory. The table illustrates the read-request to read-completion latency as measured at the initiating processor's internal bus. In this case, the size of the data to be read is small, but for complex applications with large linked-list, it is important to have the data returned quickly. This table compares typical latencies; the actual latencies are implementation dependent.

For short packets, Table 1 shows that the HyperTransport interface yields the best latencies under this set of assumptions. This is mainly due to the higher latency of the physical layer as well as the substantially longer packets associated with PCIe. The results also demonstrate the benefit of cut-through implementations.

The overall reduction in latencies is considerable for HyperTransport. In both the "Store-and-Forward" and "Cut-Through" cases HyperTransport latencies are 39 percent lower than PCIe.

	PCI Express		HyperTransport	
	S&F	C-T	S&F	C-T
	Latency (ns)			
Max Payload setting	**128**	**256**	**64**	**64**
Number of request packets	1	1	1	1
Read Request				
Tx Application	15	6	12	6
Data Link + Transaction Layers	15	15	12	12
SerDes + PMA + PCS + MAC	20	20	6	6
SerDes + PMA + PCS + MAC	30	30	8	8
Data Link + Transaction Layers	15	15	12	12
Rx Application	15	6	12	6
Fabric + DRAM cont. + DRAM (open)	51	51	51	51
Read Completion				
Tx Application (builds response packet)	18	6	12	6
Data Link + Transaction Layers	18	15	12	9
SerDes + PMA + PCS + MAC	20	20	6	6
SerDes + PMA + PCS + MAC	30	30	8	8
Data Link + Transaction Layers	18	18	12	12
Total to get 1st byte of 1st packet back	**265**	**232**	**165**	**144**
Rx Appl. (waits for all bytes @ link speed)	8	8	3	3
TOTAL: ***Source→Link→CPU→Link→Sink***	**273**	**240**	**168**	**147**

Table 1: Short Packet Reads

2K Byte Read Results

The next case compares the two technologies in long packet applications. It illustrates the read-request to read-completion latency as measured at the initiating processor's internal bus for a 2K byte read. This latency is important in the performance of bulk-transfer applications such as moving a media file.
As expected, the latency for the long packet case is higher than for just an 8-byte read. In this case, again, HyperTransport provides a lower latency than PCIe, and the value of cut-through is reiterated.

HyperTransport results yielded lower latencies in the long packet tests. In the "Store-and-Forward" case it is 41 percent lower than PCIe and in the "Cut-Through" case it's 43 percent lower.

	PCI Express		HyperTransport	
	S&F	C-T	S&F	C-T
	Latency (ns)			
Max Payload setting	**128**	**256**	**64**	**64**
Number of request packets	16	8	32	32
Read Request				
Tx Application	15	6	12	6
Data Link + Transaction Layers	15	15	12	12
SerDes + PMA + PCS + MAC	20	20	6	6
SerDes + PMA + PCS + MAC	30	30	8	8
Data Link + Transaction Layers	15	15	12	12
Rx Application	15	6	12	6
Fabric + DRAM cont. + DRAM (open)	51	51	51	51
Read Completion				
Tx Application (builds response packet)	63	6	33	6
Data Link + Transaction Layers	63	15	33	9
SerDes + PMA + PCS + MAC	20	20	6	6
SerDes + PMA + PCS + MAC	30	30	8	8
Data Link + Transaction Layers	63	111	33	33
Total to get 1st byte of 1st packet back	**400**	**325**	**228**	**165**
Rx Appl. (waits for all bytes @ link speed)	608	560	411	411
TOTAL: ***Source→Link→CPU→Link→Sink***	**1008**	**885**	**630**	**576**

Table 2: Long Packet Reads

Conclusion

In applications that have an interactive or “slalom” nature, latency is a major factor in application performance. By analyzing two typical usage scenarios for HyperTransport and PCI Express, we have shown the impact of various design choices on latency and overall application performance. Much like our automobile analogy, the appropriate interconnect design choice is critically tied to the end application: while any “vehicle” will complete the course, the highest performer is the one designed to suit the unique needs of a given application.

About the HyperTransport™ Technology Consortium

The HyperTransport Technology Consortium is a membership-based, non-profit organization that manages, promotes and licenses HyperTransport Technology. The HyperTransport Consortium consists of over 40 industry-leading member companies, including founding member Advanced Micro Devices, Alliance Semiconductor, Apple, Broadcom, Cisco Systems, NVIDIA, PMC-Sierra, Sun Microsystems and Transmeta. Membership is based on a yearly fee and it is open to any company interested in licensing the royalty-free use of HyperTransport technology and intellectual property. Consortium members have full access to HyperTransport technical documents database, they may attend Consortium meetings and events and may benefit from a variety of technical and marketing services, including the new, member-driven web portal, whose business benefits are part of a wide array of services offered by the Consortium free of charge to member companies. To learn more about member benefits and on how to become a Consortium member, please visit the HyperTransport Technology Consortium web site at http://www.hypertransport.org.