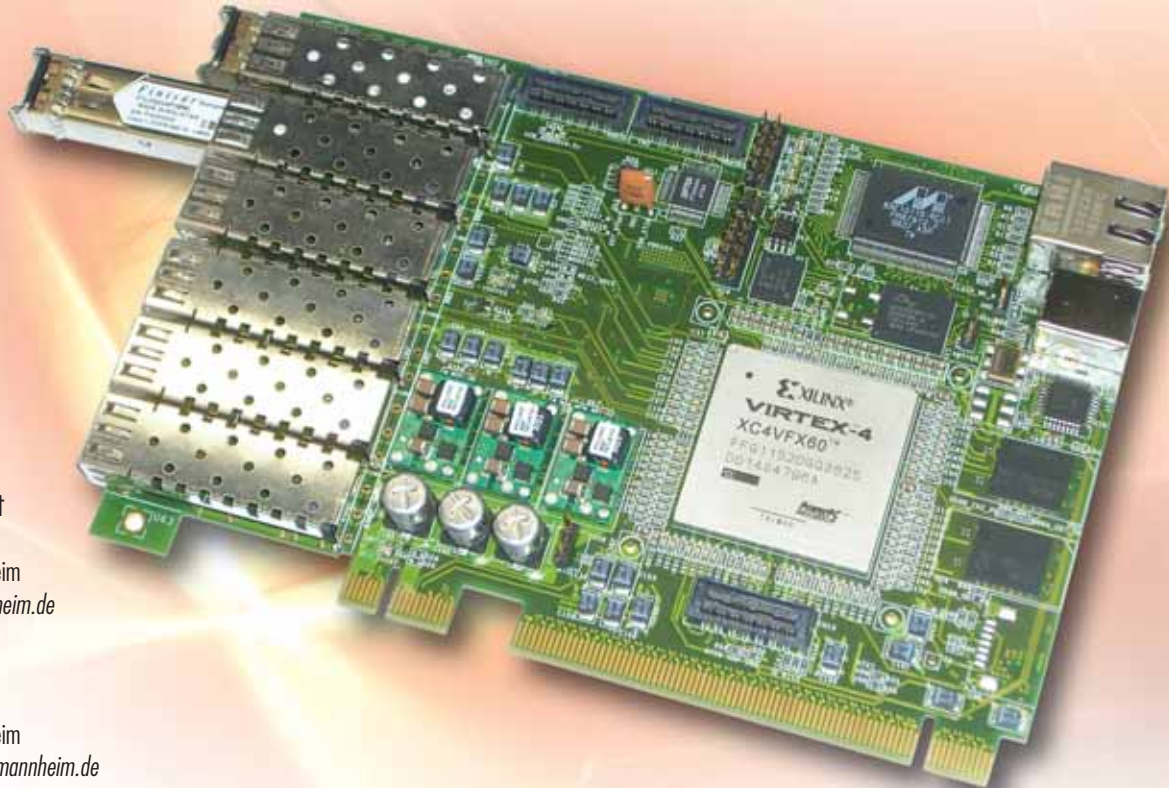


Leveraging HyperTransport on Xilinx FPGAs

An open-source HyperTransport IP core allows FPGAs to directly connect to AMD Opteron processors.



by David Slogsnat
Research Associate
University of Mannheim
slogsnat@uni-mannheim.de

Alexander Giese
Research Associate
University of Mannheim
agiese@rumms.uni-mannheim.de

Ulrich Bruening
Professor
University of Mannheim
bruening@uni-mannheim.de

FPGA-based devices are used widely in standard computing systems, either as reprogrammable coprocessors or as prototyping devices. Usually, they are connected to the system using peripheral buses like PCI Express.

AMD's Opteron processors offer a unique opportunity to improve how devices are connected to the system. Rather than using a proprietary front-side bus, there are three HyperTransport (HT) interfaces per processor. The adoption of the HyperTransport Expansion Connector (HTX) specification and mainboards

equipped with this connector now make it possible to directly connect devices to an Opteron processor.

The Computer Architecture Group at the University of Mannheim researches networks and network interface controllers (NICs) for high-performance computing. For the construction of high-speed prototypes, we sought to implement an efficient HyperTransport IP on an FPGA. Implemented on a Virtex-4 FX device, the IP is capable of running in HT400 mode, with a link clock of 400 MHz. Of course, not only NICs benefit from a direct HyperTransport connection. Any device with high bandwidth or low latency requirements, like FPGA coprocessors, will benefit from the increased performance.

Advantages of HyperTransport

Both HyperTransport and its closest competitor, PCI Express, are very capable of delivering bandwidth. For example, the 3.6-Gbps bi-directional bandwidth of our HT400 core is also possible using a PCI Express x8 device.

Besides bandwidth, there is a second important criterion to specify the performance of an interconnect: latency. Compared to PCI Express, HyperTransport adds significantly less latency to the transfer of data between a device and a host system comprising processors and memory.

One reason for this is the protocol itself, which does not require steps like 8B/10B coding and high-speed serialization. (A detailed latency analysis of both protocols

can be found on www.hypertransport.org.) The main reduction in latency, however, stems from the fact that the HTX slot is directly connected to Opteron processors instead of having to go through one or more I/O bridges, as depicted in Figure 1. This avoids time-consuming protocol conversions and usually saves one chip-to-chip hop.

Overview of HyperTransport

HyperTransport is a packet-based communication protocol for data transfer. There are three versions: HT 1.05 was developed in 2001 and updated to HT 2.0 in 2004. In April 2006, HT 3.0 was defined as the next successor. Current Opteron processors adhere to the HT 2.0b specification; no HT 3.0 devices or systems are currently available. Therefore, this article focuses on the implementation of an HT 2.0b device.

A HyperTransport link comprises two sets of unidirectional signals. Each set can be distinguished into three signal types: CAD (command, address, and data), CTL (control), and CLK (clock signals). The CAD lines are used to transport command and data packets, while the CTL line distinguishes between command and data packets on the CAD lines. The HyperTransport protocol supports CAD buses with a width of 2, 4, 8, 16, or 32 bit, as depicted in Table 1. The width of the CAD bus is usually known as the width of the HyperTransport link. If more than eight CAD lines are used per link and direction, every group of eight signals has its own CLK signal. These groups of signals are synchronously transmitted with the source-associated CLK signal.

The data transferred on the CAD bus is 32-bit aligned independent of the bus width. All transferred packets are at least the size of one doubleword (32 bit). HyperTransport allows clock frequencies from 200 MHz to 1.4 GHz in HT 2.0 and up to 2.6 GHz in HT 3.0. On the CAD bus, double-data-rate signaling is used. Current Opteron processors use 16-bit link widths and frequencies up to 1 GHz. In Opteron systems, all devices start at power up of the system with 200-MHz and 8-bit-

wide links. The BIOS checks the capabilities of all devices by accessing the device's HyperTransport register space and sets new values for frequency and width for every link according to the capabilities of the two devices that share the link. After that, it forces a re-initialization of all HyperTransport devices to establish the new parameters.

All transfers in HyperTransport are packet-based. To decouple the transmission

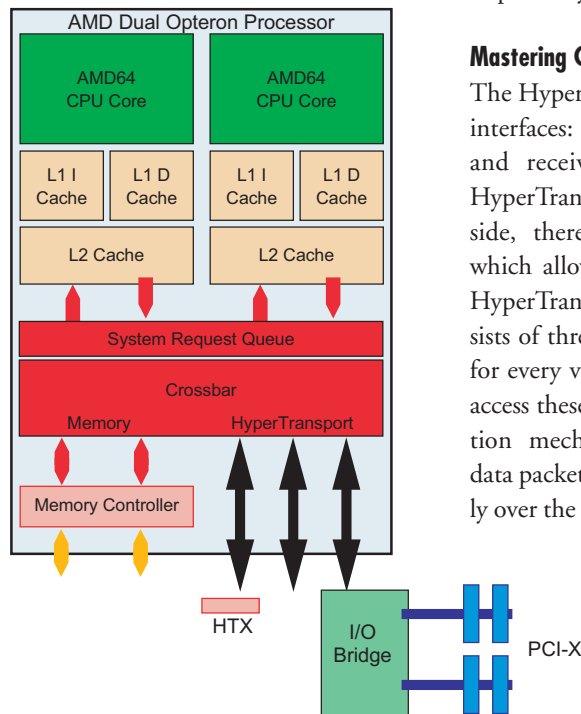


Figure 1 – Block diagram of an Opteron dual-core processor in a system with HTX and PCI-X slots

response from the request, the packets are transferred in a split phase transaction. A transfer always starts with one of three kinds of control packets: information, request, or response. Information packets are used for flow control and synchronization. Request packets are sent to write data to a receiver and are also used to initiate reads. Response packets contain the answer to a corresponding request.

Control packets have a size of 4 or 8 bytes or, if they use addresses of 64 bits instead of 40-bit addresses, the extended format has a size of 12 bytes. If a transfer contains payload data, the next data packet

that is sent on the link belongs to this packet. A data packet can have a maximum size of up to 64 bytes.

Sending other control packets during a stream of data packets at every 32-bit boundary is allowed, but only if this control packet would not be followed by data. Otherwise it could not be possible to determine which control packet the data belongs to. This mechanism makes it possible to send urgent control packets with high priority. Flow control on the link is imposed by a credit-based protocol.

Mastering Challenges in FPGA Design

The HyperTransport core has two different interfaces: one is the HyperTransport send and receive links, as specified in the HyperTransport protocol. On the other side, there is the application interface, which allows FPGA designs to access the HyperTransport core. This interface consists of three queues in each direction, one for every virtual channel. Applications can access these using a valid-stop synchronization mechanism. Control and attached data packets can be delivered simultaneously over the 160-bit-wide interface (96 bit to handle extended control packets, 64 bit for data packets).

Of course, our aim was to build a core that is as fast as possible. One limiting factor is the speed of the serial I/Os. In the Xilinx® Virtex™-4 FX

devices we used, the speed is limited to 400-MHz DDR, thus HT400. Xilinx SERDES blocks can parallelize/serialize the link by a factor of four so that the clock frequency of the core is 200 MHz. The SERDES blocks are also controlled by a bit-slip module to generate proper 32-bit boundary alignment.

Our second challenge was to process this data stream with the lowest number of pipeline stages and reasonable resource requirements. The decode unit, responsible for decoding the incoming packet stream and putting packets into the corresponding application interface queues or

Resource	8-Bit Link, HT200		16-Bit Link, HT400	
	Count	Usage	Count	Usage
Logic Slices	2,699	6.5%	6,700	15.8%
FIFO16/RAMB16s	30	7.5%	30	7.5%
DCM_ADVs	3	25%	4	33.4%
ISERDESs	10	1%	19	2%
OSERDES	9	1%	19	2%

Table 1 – Resource requirements in a Virtex-4 XC4VFX100 FPGA

Direction	Clock Cycles	Delay@ HT200	Delay@ HT400
In	11	55 ns	27.5 ns
Out	7	35 ns	17.5 ns

Table 2 – Hardware latency of the HyperTransport core

HyperTransport core units, is the most complex unit in the design. There may be two incoming 32-bit control packets at the same time, so there are two decode sub-units in parallel. On the other hand, they may be parts of one large 64-bit or 96-bit control packet.

Our third challenge was to reach an internal clock frequency of 200 MHz. We are currently using the core with only 100 MHz internally, and thus in HT200 for more than half a year. The HT400 version still requires some optimizations to run reliably at speed.

Results

Table 1 shows the resource requirements of the HT400 core (which we did not yet optimize for resource utilization) in com-

parison with an HT200 version that supports only 8-bit-wide links. The pure internal hardware latency of the core, shown in Table 2, is very low: 11 clock cycles for the input path from the HyperTransport link to the application interface and seven clock cycles outbound.

You can download the HT200 core from our website, as well as up-to-date information about the core and its performance. The HT400 core is still in the verification phase and will be available soon.

Implementing a Low-Latency NIC

As a university research group, our main focus is in high-performance computing. We used the HyperTransport core to build up a prototype NIC for the EXTOLL network. We developed this network to

research and implement new methods in high-performance networks and network interfaces. Figure 2 is a block diagram of such a NIC. The current prototype is implemented using the HT200 8-bit core with an internal clock frequency of 100 MHz. Nevertheless, the Netpipe (www.scl.ameslab.gov/netpipe/) benchmark shows a latency of <1.5 μs on the software API level for sending small messages, which is an excellent result.

The hardware of our prototyping platform comprises the Iwill DK8-HTX dual Opteron mainboard and the HTX board. The Xilinx Virtex-4 FPGA-based board that we developed in 2006 can be plugged directly into the HTX slot of the Iwill mainboard. This is also the only environment in which we have verified both the core and the board in depth thus far.

Conclusion

The HT IP core successfully exploits the potential of the FPGA in terms of bandwidth, latency, and resource utilization. Offering an HT400 connection and thus 3.6 GB of bi-directional bandwidth, the HyperTransport core can be used for more than just prototyping – its performance is sufficiently good enough to serve as a production coprocessor as well.

The HyperTransport core can also be mapped to an ASIC implementation to reach higher link speeds. For FPGAs, faster links may be possible with faster FPGA families such as the Virtex-5 device family.

We are also working on a coherent HyperTransport core with HT400 speed that allows devices to participate in the cache-coherence protocol of Opteron processors. In addition to the functionality of a peripheral device, this will, for example, allow a device or coprocessor to act like an Opteron system memory controller or a coherent cache. In contrast to the non-coherent HyperTransport, the use of the coherent version requires a license from AMD.

For more information about the open-source HyperTransport core or the HTX board, visit our Center of Excellence for HyperTransport Technology: www.ra.informatik.uni-mannheim.delcoehb.

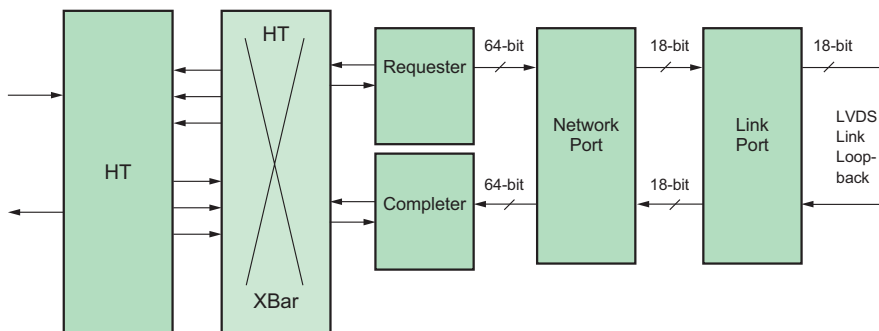


Figure 2 – Prototype NIC device