# A HyperTransport 3 Physical Layer Interface for FPGAs

Heiner Litz, Holger Froening, Ulrich Bruening

University of Heidelberg, Computer Architecture Group, ZITI,
B6 26, 68131 Mannheim, Germany
{heiner.litz, holger.froening, ulrich.bruening}@ziti.uni-heidelberg.de

**Abstract.** This paper presents the very first implementation of a HyperTransport 3 physical layer interface for Field Programmable Gate Arrays. HyperTransport is a low latency, high bandwidth point-to-point interconnect technology that can be used to directly connect hardware accelerators to AMD's Opteron CPUs. Providing support for HyperTransport 3 on FPGAs is highly relevant for increasing the performance of accelerators based on reconfigurable logic. This paper shows the challenges of such an implementation and novel ideas to solve them successfully. A new architecture is presented that uses Fast Serializer Logic to keep up with the increasing speeds of current host interface protocols. A solid evaluation is provided using a specially developed FPGA board as a verification platform.

## 1 Introduction

HyperTransport 2.x (HT2) [1][2] is an established technology for chip-to-chip connections offering a very low latency and a high bandwidth. It is used in all major AMD processors, including the Opteron [3]. The most recent development in the area of HyperTransport is the availability of the HyperTransport 3.1 (HT3) [4] and the HyperTransport Extension 3 (HTX3) [5] specification. The first allows for very high operating frequencies (3.2GBit/s per lane), the second defines a connector for add-in cards with direct HT3 connection to one of the CPUs.

HT3 will be mainly used by AMD processors. HT3 units can either be located on the mainboard, in the CPU socket or on an HTX3 add-in card. In all cases HT units are directly connected to the CPU(s) without any intermediate bridges or any kind of protocol conversion. The high performance of HT3 in terms of extremely high bandwidth of up to 25.6 GByte/s[1] and low latency due to the direct connection makes it highly suitable for high performance applications. Additionally, the direct connection allows to participate in the cache coherency protocol of the CPUs [6]. Typical applications are accelerated computing [7][8] and fine grain communication [9][10].

1. Maximum theoretical bandwidth for a 16 bit wide HT 3.1 link.

In order to leverage the advantages of Field Programmable Gate Array (FPGA) technologies some obstacles have to be overcome, because typical FPGA technology is not designed for the use in HT3 applications. In this paper we present an HT3 Physical Layer (PHY) as a major component of an HT3 unit implementation. The PHY together with the HT3 Core as an technology-independent module enables the use of HT3 in FPGAs. This allows to leverage the FPGA's advantages in terms of reconfigurability and high performance in a verification platform, rapid prototyping station or in applications like accelerated computing and fine grain communication. To the best of our knowledge there is currently no HyperTransport 3 implementation for FPGAs available. This paper presents a novel approach to use Fast Serializer Logic (FSL) for HyperTransport 3 interfaces leading to much higher supported bandwidths than any other existing implementations. Our proposed solution provides up to 12.8 GByte/s bidirectional host interface bandwidth, which is more than twice as much as the fastest PCI-Express 8x implementation for Xilinx FPGAs which offers 5 GByte/s.

The Physical Layer (PHY) is the first of seven layers in the Open Systems Interconnection (OSI) model of computer networking. Usually, it is the most complex one due to the large amount of different technologies. The PHY interfaces with the Medium Access Control (MAC) sublayer. Typical tasks for the PHY layer are media-dependent encoding and decoding and transmission and reception.

In the application here the PHY is responsible for: packet delivery, synchronization, data serialization and de-serialization, bit slip and clock/data recovery. It is not responsible for: flow control, error detection or correction and link configuration. These tasks are performed by the MAC sublayer, which is layered on top of the PHY layer. In this case, the MAC is the HT3 Core, which is based on an open source available HT2 Core [11]. An overview of the system in which the PHY is being used is provided in figure 1.

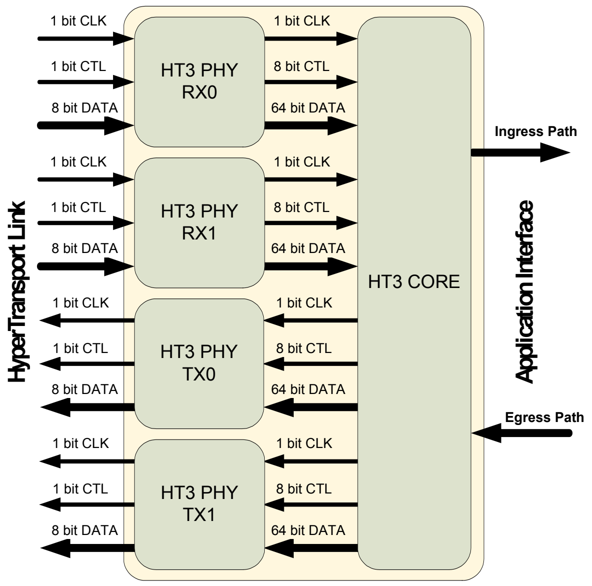

**Fig. 1** System Overview

### 1.1 Contributions

This paper presents some novel ideas that allow to implement a high performance architecture for next generation's host interface technologies. The results of our research are highly relevant for upcoming high performance applications in the area of accelerated computing, coprocessors and interconnection networks. Our major contributions include:

- First HyperTransport 3 implementation for FPGAs
- First HyperTransport implementation using FSL
- Highest host interface bandwidth for FPGAs (12.8 GB)
- Manual oversampling technique for FSL
- Real world FPGA implementation

The rest of this paper is organized as follows: We will provide background information in the following section. Section 3 describes the architecture which was developed to solve the various problems occurring within a HT3 PHY implementation for FPGAs. Section 4 is dedicated to a concrete implementation thereof, which is evaluated in Section 5. Section 6 provides the conclusion and outlook.

## 2 Background

Within the HyperTransport 3.1 specification several new features were introduced that pose a challenge on FPGA implementations. The high line rates of 2.4 - 6.4 Gbps together with the demanding electrical characteristics of HT3 make FPGA implementations difficult. To our best knowledge there is currently no FPGA vendor that naturally supports HT3.

HyperTransport defines two different operating modes. The first one is the Gen1 mode which refers to HT1 and HT2 implementations, the second one is called Gen3 and refers to HT3. HT3 is downwards compatible which means that Gen3 devices have to support Gen1 as well. The link initialization always begins in Gen1 mode at 200 MHz with an 8 bit link. Both endpoints then negotiate their link connection and switch to the highest transmission rate supported by both. If both devices support frequencies above 1 GHz the link switches into Gen3 mode. To further increase link bandwidth HT uses a Double Data Rate (DDR) mechanism resulting in a line rate of 400 Mbps in HT200 mode and a rate of 2.4 Gbps in HT1200 mode, for example.

Due to the high line rates of Gen3 the protocol significantly differs from Gen1 in terms of the electrical and link layer. Gen1 is a source synchronous protocol requiring the sender to provide a clock for each 8 bit link that can be used by the receiver to sample the data. To guarantee functionality at all line rates the specification poses strict skew requirements for the data and clock lanes. Gen3 on the other hand uses a Clock Data Recovery (CDR) mechanism on each individual lane to sample the data. To allow for reliable clock recovery the lanes have to have sufficient level transitions which requires the use of a bit scrambler on each lane. The link clock is still used in Gen3 to provide the Phase Locked Loops (PLLs) in the CDR circuits with a stable reference clock.

The HT3 specification defines many other attributes of the electrical interface like the common mode, the slew rate of the signal and the amplitude. In a summary, the minimum requirements of a HyperTransport3 physical layer interface for FPGAs are the following.

- Compliance to HT3 electrical specification
- Support of 200 MHz for Gen1 initialization
- Support of source synchronous sampling mechanism
- Support of 1.2 GHz and above for Gen3
- Support of CDR data sampling mechanism
- Capability to switch between Gen1 and Gen3

## 3 Architecture

The high line rates of HT3 prohibit the use of regular IO cells for the PHY. A basic PHY has been implemented with regular IO cells. Measurements have been performed at the pins with an oscilloscope, the results are shown in figure 2.

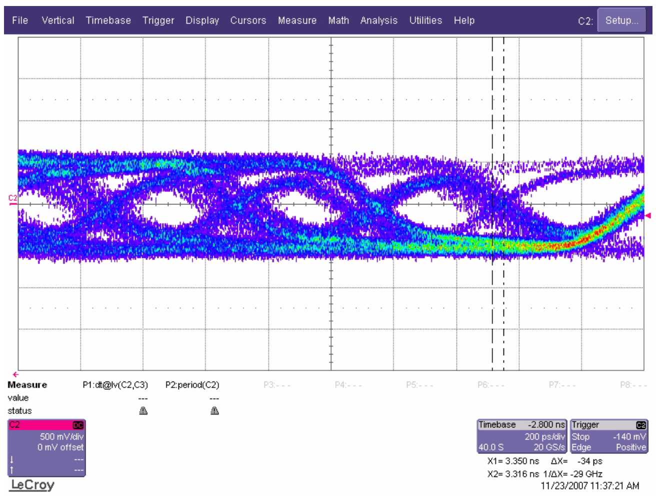

**Fig. 2** Two Gbps line rate with regular IO

It should be obvious that successful operation is not possible with the eye diagram shown above. The only solution, therefore, seems to be the use of Fast Serializer Logic (FSL) blocks provided by many FPGAs. They are capable of driving data rates of 3 Gbps and above however pose various limitations. At first, each FPGA only contains a very limited number of these FSL blocks. Low speed operation is a priori not supported, as a minimum line rate of 640 Mbps is required. Electrical parameters like the common mode are not flexibly adjustable but imposed by the FSL. This paper proposes a new architecture that solves these problems and that allows for successful HyperTransport 3 implementations in Xilinx FPGAs.

### 3.1 Electrical Compliance

The Xilinx FSL used in our implementation are not electrically compliant due to their supported common mode voltages. Various HSpice simulations have been performed to determine whether the common mode offset is critical. The simulations show successful operation, however, have to be approved in a real prototype.

### 3.2 Gen1 Operation

For the HyperTransport initialization phase support of Gen1 mode at HT200 speed is required. However, the FSL blocks in Xilinx devices operate at a minimum line rate of 640 Mbps. To overcome this problem a manual oversampling mode has been developed. A reference clock of 200 MHz is provided together with a PLL multiplier value of 4 to generate the external 800 MHz link frequency. Internally, instead of implementing a 1:8 serialization rate in the PHY as required, a 1:16 ratio is used. If the 16 bit parallel data is provided to the PHY at the same speed of the 8 bit interface, the resulting line rate is doubled to 800 Mbps. Mapping the 8 bit data to the 16 bit interface is done by assigning it to the even bit positions of the 16 bit interface and a copy to the odd bit positions. The result of this mechanism is that adjacent odd/even tuples like 0/1 or 2/3 carry the same data value. This mechanism is transparent to the link and therefore it appears as a 400 Mbps interface to the other link endpoint. Oversampling is used on both the receive (RX) and transmit (TX) path with the exception of the TX link clock. This clock has to be phase shifted by 90 degrees in regard to the data. To achieve this behaviour the oversampling is deactivated for the clock lane and a static bit pattern is assigned to the serializer. This pattern is a sequence of alternating pairs of zeros and ones. The sequence is then shifted by a phase offset of a single bit time.

In Gen1 mode the link is not scrambled which may result in idle phases of the link where no signal transitions occur for a long period in time. The PLLs in the CDR circuits therefore lose their lock rendering the clock recovery impossible. As a solution the GTPs are operated in lock to reference mode using the RX link clock.

### 3.3 Gen3 Operation

For Gen3 operation the interface has to be run at a line rate of 2.4 Gbps. This is achieved by providing a 300 MHz clock with a multiplier factor for the PLLs of 8. The multiplier factor of the PLL is normally set at compile time. However, as the line rate has to be switched during operation without the possibility of loading a new design, the Dynamic Reconfiguration Port (DRP) has to be used, which provides access to several parameters of the FSL at runtime. A Finite State Machine (FSM) takes care of the reconfiguration process. Furthermore, the oversampling mode has to be disabled by reconfiguring the FSL interface from 16 bit to 8 bit and by bypassing the oversampling logic. The lock to reference mode has to be disabled and the FSLs have to perform clock data recovery. Each lane is sampled with its own clock, so the lanes are inherently skewed. Elastic buffers are used to compensate for the phase mismatch and synchronize the different lanes into a unified clock domain. To remove multiple bit skew the lanes have to be put into relation to each other. The HyperTransport 3 specification defines a training phase during Gen3 initialization which can be utilized to adjust the delay of the lanes and to provide a fully synchronous interface.

## 4 Implementation

The proposed architecture has been implemented in Verilog Register Transfer Level (RTL) code and mapped to the Xilinx Virtex5 FPGA technology. The following paragraphs present a detailed overview of the implemented logic, clocking scheme and IO cells.

### 4.1 HT3 RX PHY

The HT3 PHY's purpose on the receive side is to deserialize the incoming link data, and to compensate for bit and byte skew by performing bitslip functionality. A toplevel block diagram with the different clock domains of the RX PHY is shown in figure 3.

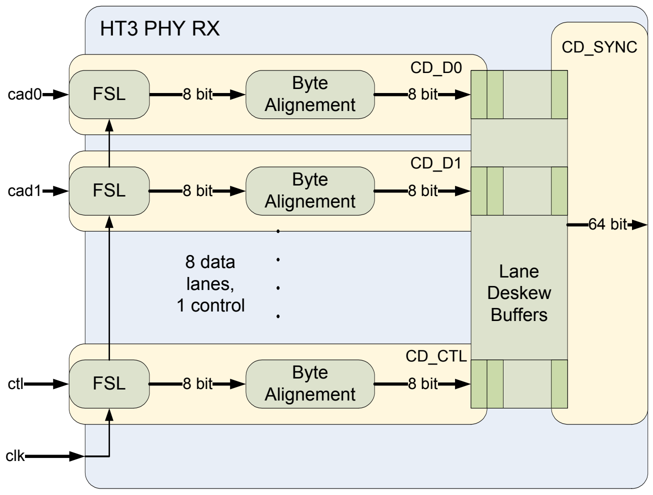

**Fig. 3** RX PHY Toplevel Diagram

The PHY comprises 10 different clock domains which are the 8 clock regions for control and data lanes, CD_D0 through CD_D7, another one for the CTL signal (CD_CTL) and the CD_SYNC domain. All domains run at the same clock frequency however have different phases. As in Gen1 mode a single reference clock is used to sample the data the CD_D domains can be regarded as synchronous. In Gen3 mode, however, the sample clock is provided by the FSL and, therefore, the lane deskew buffer is needed to synchronize the lanes so they can be accessed with single clock. The RX link clock is only needed in Gen1 mode as in Gen3 the FSL's are provided with a reference clock and perform CDR.

A detailed view of a per lane clock domain is given in figure 4. The functionality of the components again vary depending on Gen1 or Gen3 operation. In Gen1 mode the FSL provides a deserialized data stream of 16 bits width. Due to the manual oversampling technique only every second bit of this interface is extracted by the oversampling module. In both modes the bitslip module is provided with 8 bits of data each clock cycle. This module's purpose is to perform byte alignement. As the FSL is not aware of the position of the byte inside the serial data stream there are 8 possible ways of deserializing the same data stream. The bitslip module is responsible for compensating this misalignment and shift the bits accordingly. Therefore, it uses specific training patters which are send during initialization of both Gen1 and Gen3 and which determine the exact position of the byte in the serial stream.

Another task that has to be performed, while switching the operation mode, is to provide the correct clock. In Gen1 mode the 200 MHz RX link clock is used to sample the data. When switching to Gen3 the RX link clock changes to 1.2 GHz. The internal PLLs of the FSL, however, require a clock of 300 MHz in this mode which has to be provided through an external clock source, e.g. an oscillator. Concurrently with the switch of the input clock the user clock has to be adjusted correctly. In Gen1 mode the FSL provides a reference clock which is the feed through receive link clock. In Gen3 the CDR circuit provides a recovered clock that must be used to drive the clock domain and which is used to push the date into the lane deskew buffers.

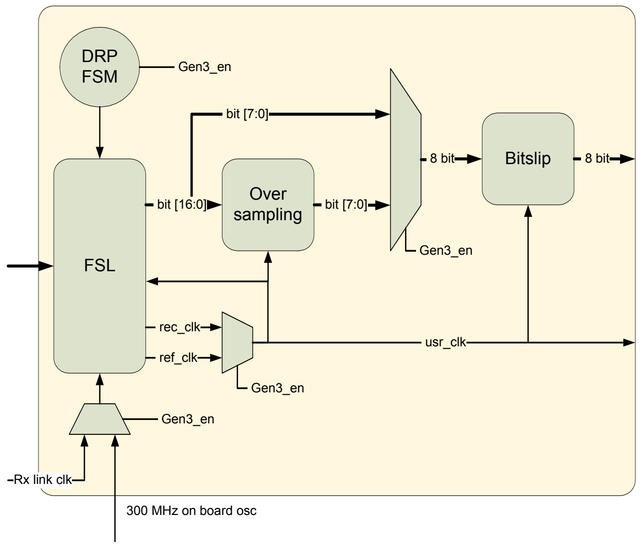

**Fig. 4** Per Lane Clock Domain

The final task is to switch between lock to reference mode and clock data recovery in between the Gen1 to Gen3 transition. This is realized through an FSM that writes the correct values into the configuration registers of the FSL using the dynamic reconfiguration port. The DRP is also used to adjust some analog parameters that control the PLLs. As soon as all modules are in Gen3 mode the FSL block and all its internal blocks are resetted. After reset, the CDR mechanism waits for valid scrambled data on the lanes to be able to lock and to recover a clock. If the complete initialization process including the training phase is successful the HT core connected to the PHY is notified and can start to process incoming data.

## 4.2 HT3 TX PHY

The transmit side of PHY is responsible for deserializing the data and driving it onto the lanes. The main functionality is already provided by the FSL, which allows for a straight forward implementation. An overview of the transmit side of the PHY is given in figure 5.

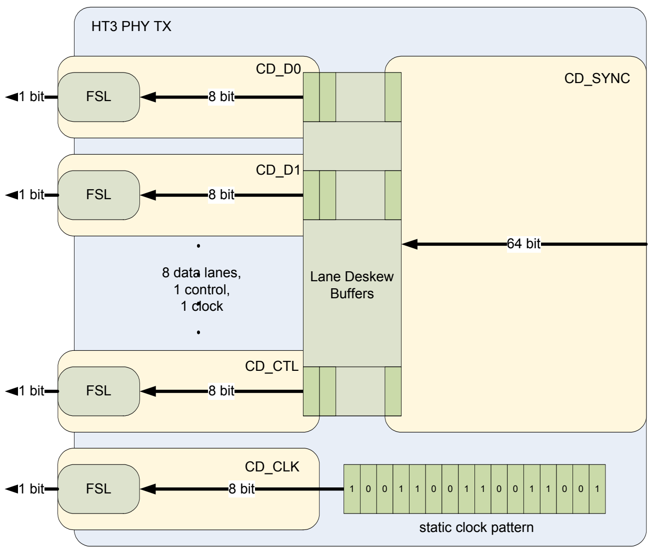

**Fig. 5** TX PHY Toplevel Diagram

It can be seen that the transmit path also uses several clock domains, one for each outgoing link and a synchronous source clock domain. In Gen1 mode all TX domains can be regarded as synchronous as they are provided with the same reference clock, in Gen3, however, the transmit PLLs provide their dedicated clocks per output lane. The transmit clock requires special attention. In Gen1 mode it has to possess a defined phase of exactly 90 degree relative to the CAD and CTL lanes. This behaviour can be achieved with the following technique. A static bit pattern of alternating ones and zeros is provided to the serializer with the reference clock frequency. As manual oversampling is used on the TX side, each value has to be duplicated. The 90 degree phase shift can now be achieved by starting with a single digit followed up by the duplicate values, as shown in figure 5. In Gen3 mode no oversampling is used and therefore the phase shifted clock cannot be generated in this way. As in Gen3 no skew requirements for the lanes exist, it is sufficient to provide a clock with the correct frequency of 1.2 GHz.

The transmit clock domains CD_D0 through CD_D7 implementations are similar to the ones of the receive side. An oversampling circuit is used in Gen1 which duplicates the tx data to produce the desired manual oversampling effect. It is bypassed in Gen3 mode. As the PLL of an FSL is shared between the rx and tx side no other modifications through the DRP are necessary. The parallel transmit clock for each specific FSL domain is already provided by the PLL.

## 5 Evaluation

The PHY implementation has been evaluated by simulations and in a real world test design. For the simulation, the PHY is connected to a HyperTransport bus functional model on the host interface side and to a HT3 core on the parallel side. The implementation of the HT3 core is out of scope of this paper. The complete system has been embedded into a System Verilog based verification environment to be able to provide the best possible test coverage. The PHY was tested under different operating conditions using randomly constrained and corner case test patterns. Especially the Gen1 to Gen3 switch has been tested precisely showing the desired behaviour for both modes.

Although the simulation models of the FSL are very accurate, the behaviour of simulated designs depending on analog and IO components tend to differ from real world implementations. Therefore, an FPGA based verification platform, in the form of a Printed Circuit Board (PCB), has been developed. It is shown in figure 6.

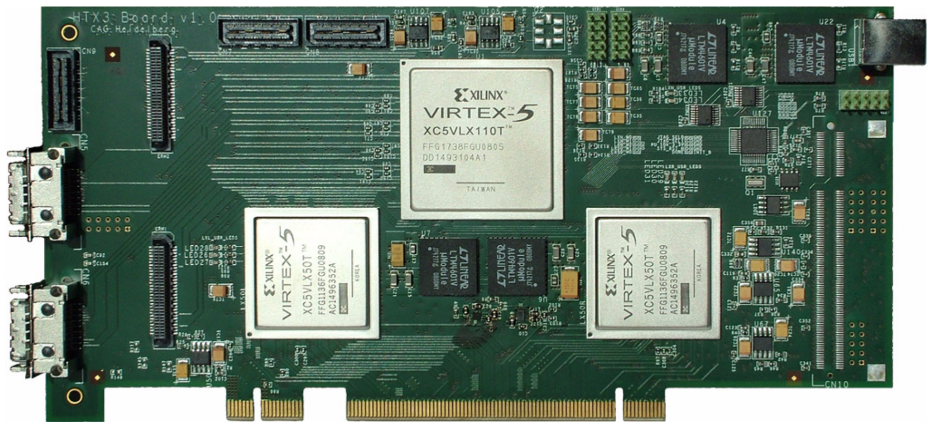

**Fig. 6** HT3 Verification Platform

The board's main components are three Xilinx Virtex5 FPGAs. Two Virtex5 LX50T devices are directly connected to the HyperTransport Extension link on the bottom side, acting as the PHYs. Both use their FSL blocks for the HTX interface and implement a bidirectional, 32 bit, double data rate (DDR) interface for communication with the Virtex5 LX110T residing on top. This large FPGA is used to hold the core logic of the HT3 link. The separation into three FPGAs was required as no FPGAs with sufficient FSL blocks for a HyperTransport 16 bit link were available at the time the board was designed. The board contains several other components which include an USB connector, SODIMM socket, two CX4 connectors for networking and several general purpose IO for extensions.

Configuring the LX50Ts with the PHY logic and loading the HT core logic into the LX110T, the card can be plugged into a commodity computer system. If everything works correctly the BIOS should detect the card and boot up the operating system. In the first basic test the card was configure to support only Gen1 operation at the Hyper-

Transport initialization frequency of 200 MHz. In this mode the PHYs operate in lock to reference mode as described above. The critical issues which are checked in this configuration are the compliance to the electrical specification, the functionality of the data sampling mechanism, the oversampling and the bitslip mechanism. Electrical compliance has been tested by measuring eye diagrams of the link interface, see figure 7, and by checking the internal registers of the Opteron CPU which is connected to the FPGA board. While proceeding the boot process in single step mode it can be seen that the HT3 PHY is correctly detected and that the link initialization process passes successfully. Long-run tests haven't shown any problems regarding signal integrity or the common mode. A Linux Operating System has been successfully booted on the system showing the HT device in its device tree. To perform further testing, a device driver and an application interface (API) has been developed which provides access to the verification platform from software. Various tests have been performed that write to and read data from the memory mapped device.

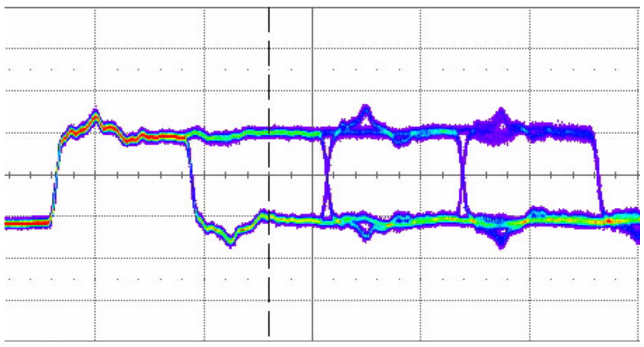
**Fig. 7** Eye diagram of the FSL based HT link

The successful tests for HyperTransport speeds of 200 MHz prove the correct functionality of the PHY at low clock rates that use the lock to reference mode and manual oversampling. This is an important result already, as in general, the fast serializer logic is built for transmitting data at frequencies of 640 Mbps and above.

The remaining task is to test the complete system at higher speeds and especially the in system switch from low speed to high speed operation. We have therefore configured the HT core to support higher frequencies like 800 and 1600 Mbps. In this case the system again boots in Gen1 mode at 200 MHz, then determines the capabilities of the HT device and then ramps up the link frequency. The PHY has to detect the switch, disable oversampling, change to clock data recovery mode and adjust the frequency of the PLLs through the DRP as explained above.

The card has been successfully booted in both high frequency modes and the same tests have been performed. This is an excellent proof of our HyperTransport 3 PHY architecture. It furthermore shows that fast serializer logic can be used for HyperTransport implementation despite their different electrical specification. The system already provides exceptional performance with 16 lanes at 1600 Mbps per lane. Further testing will show how far the frequency can be increased and which HyperTransport speeds can be supported at a maximum with our PHY.

## 6 Conclusion

To the best of our knowledge this paper presents the first fully functional HyperTransport 3 physical layer interface for FPGAs. It furthermore shows the first successful utilization of fast serializer logic for HyperTransport. The result is an architecture that offers high flexibility and great performance. To prove the quality of our work the design has been implemented in hardware description language code and mapped to a Xilinx FPGA. The verification of this design has been performed in a real world system. Therefore, an FPGA based rapid prototyping station has been designed which is comprised of three Xilinx Virtex5 devices. A device driver and API has been written to fully test the HyperTransport 3 PHY implementation. The current implementation has been tested with link speeds of up to 1600 Mbps. We believe that our implementation can scale up to 3200 Mbps, so future work will focus on increasing the frequency to the maximum of what our PHY can support.

## 7 References

[1] Anderson, D., and Trodden J. 2003. HyperTransport System Architecture. Addison-Wesley.

[2] Hypertransport Consortium. 2008. The Future of High-Performance Computing: Direct Low Latency CPU-to-Subsystem Interconnect. http://www.hypertransport.org.

[3] Keltcher, C.N., McGrath, K.J., Ahmed, A., and Conway, P. 2003. The AMD Opteron processor for multiprocessor servers. IEEE Micro, 23(2):66-67.

[4] Hypertransport Consortium. 2008. HyperTransport™ I/O Link Specification Revision 3.10. http://www.hypertransport.org.

[5] HyperTransport Consortium. 2008. HTX3™ Specification for HyperTransport™ 3.0 Daughtercards and ATX/EATX Motherboards. http://www.hypertransport.org.

[6] Conway, and P., Hughes, B. 2007. The AMD Opteron Northbridge Architecture. IEEE Micro, 27(2):10-21.

[7] Gothandaraman, A., Peterson, G. D., Warren, G. L., Hinde, R. J., and Harrison, R. J. 2008. FPGA acceleration of a quantum Monte Carlo application. Parallel Computing 34, 4-5 (May 2008), 278-291.

[8] Zhuo, L. and Prasanna, V. K. 2005. High Performance Linear Algebra Operations on Reconfigurable Systems. In Proceedings of the 2005 ACM/IEEE Conference on Supercomputing.

[9] Heiner Litz, Holger Froening, Mondrian Nuessle, Ulrich Bruening. VELO: A Novel Communication Engine for Ultra-low Latency Message Transfers. 37th International Conference on Parallel Processing (ICPP-08), Sept. 08 - 12, 2008, Portland, Oregon, USA

[10] El-Ghazawi, T., Carlson, W., Sterling, T., and K. Yelick. 2005. UPC: Distributed Sharend Memory Programming. Wiley Series on Parallel and Distributed Computing. John Wiley and Sons, Inc., Hoboken, NJ.

[11] David Slogsnat, Alexander Giese, Mondrian Nuessle, Ulrich Bruening. An Open-Source HyperTransport Core. ACM Transactions on Reconfigurable Technology and Systems (TRETS), Vol. 1, Issue 3, p. 1-21, Sept. 2008