Theory and Simulation of High-speed Data Transmission Based on Virtex-7 GTH

Xin Jiang
Beijing Institute of Technology
Beijing China
e-mail:1070935984@qq.com

Dongrui Jia
Beijing Institute of Technology
Beijing China
e-mail:804098418@qq.com

Yang Zhou
Beijing Institute of Technology
Beijing China
e-mail: zhouyang850611@163.com

Leichen Zhou
Beijing Institute of Technology
Beijing China
e-mail: zhouleichen@163.com

Ning Wang
Beijing Institute of Technology
Beijing China
e-mail: wangningqin@126.com

Abstract—Along with the development of information communication, the amount of data to be obtained also increases. However, the requirement for high-speed data transmission is far more than the development of data transmission and put forward to a higher demand on high-speed data transmission. So, researches on high-speed data transmission are with great importance. This paper makes a research of the GTH, and the GTH transceiver which is embedded in Xilinx Virtex-7 FPGA chips is an advanced high-speed transceiver. The GTH can reach the highest data rate up to 13.1 Gbps and is more powerful than GTP. The GTH module has flexible user-defined features and parameters, and costs lower power. The internal structure and functional module of the GTH is introduced. Furthermore, gets simulation and verification in the ISE environment, the results of experiment show that this module can realize the function of high-speed, high-reliability and high-stability data communication when the serial speed is 10Gbps, and is with great application.

Keywords-GTH; FPGA; Virtex-7; 10Gbps; data transmission

I. INTRODUCTION

With the development of information transmission, the necessity for speed is infinite, especially for the communication industry. The requirement of higher data rate is explosively increasing. For example, the rate of optical communication doubles every six months, which is much faster than the speed defined by the famous MOORE LAW. At the same time, the capacity of transmitted data has become the core issue and a revolution of bandwidth must be inevitable [1-3].

Xilinx have already promoted the low power Gigabit Transceiver, and the high speed serial transmitter is widely applied for interfaces between ICs on the PCB, backplane, and longer distances, ensuring the quality and integrity of signal[4][5].

The research of this paper is based on Virtex-7 FPGA. The transceiver GTH, as one IO interface, supports up to 13.1Gbps [6]. Each transceiver has separate transmitter and receiver circuit. There are a large number of user-defined features and parameters, which determine the working mode, signal routing, clock choice and so on. All these parameters can be defined through configuration or reconfiguration using DRP (Dynamic Reconfiguration Port).

The GTH module adapts to the CML (Current Mode Logic), CDR (Clock & Data Recovery), Equalizer, Line Encoding Schemes and Pre-Emphasis technologies. This kind of transceiver is critical to the future of high-speed applications, and supports industrial protocols, such as Fiber Channel, PCI Express, Rapid IO Serial, Advanced Switching Interface, Serial ATA, 10-Gb Ethernet (XAUI) and so on.

II. THEORY OF GTH

Fig. 1 is the GTH Transceiver Block Diagram. On the RX side, the input high-speed differential signal is transformed into single-ended signal firstly. Then, the single-ended signal goes through SIPO and Polarity parts. The following modules Comma detect, 8B/10B Decoder, Elastic buffer and Gearbox are also available. We can bypass these modules if they are not necessary in our design. Finally, the signal will be transferred into FPGA Interface. On the TX side, the transmitted data from FPGA Interface goes through Gearbox or 8B/10B Encoder at the beginning. Then, the data can be written to Polarity and PISO after Phase Adjust FIFO. The single-ended signal is transformed into differential signal and then sent out. Similarity, the Gearbox, the 8B/10B Encoder and the Phase Adjust FIFO can be bypassed if not necessary. All operating clocks are derived from the Clock Management. The following part briefly describes the function of major components [7].

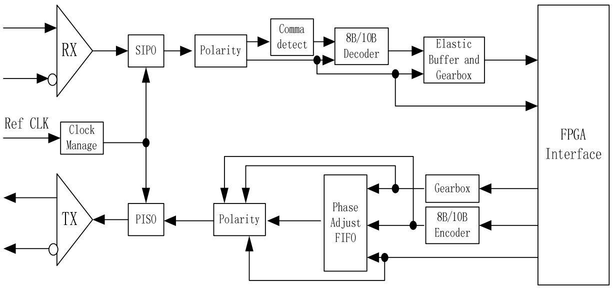

Figure 1. GTH transceiver block diagram

A. *Analog Front End*

The RX analog front end（AFE）is a high-speed current-mode input differential buffer. It can configure RX termination voltage and calibrate termination resistors.

B. *RX Equalizer*

The DFE mode is available for equalizing lossier channels. The DFE allows compensation of transmission channel losses by providing a closer adjustment of filter parameters than when using a linear equalizer. It compensates for the post cursors. A linear equalizer allows pre-cursor and post-cursor gain. The GTH RX DFE mode is a discrete-time adaptive high-pass filter. The TAP values of the DFE are the coefficients of this filter that are set by the adaptive algorithm.

C. *SIPO/PISO*

The SIPO transfers serial data stream to parallel data stream that the width is selectable. The PISO transfers parallel data stream to serial data stream.

D. *Polarity Control*

If RXP and RXN differential traces are accidentally swapped on the PCB, the differential data received by the GTH transceiver RX are reversed. The GTH transceiver RX allows inversion to be done on parallel bytes in the PCS after the SIPO to offset reversed polarity on differential pair. Polarity control function uses the RXPOLARITY input, which is driven high from the fabric user interface to invert the polarity.

E. *CDR*

The RX clock data recovery (CDR) circuit in each GTH channel transceiver extracts the recovered clock and data from an incoming data stream. The transceiver employs phase rotator CDR architecture. Incoming data first goes through receiver equalization stages. The equalized data is captured by an edge and a data sampler. The data captured by the data sampler is fed to the CDR state machine and the downstream transceiver blocks. The CDR state machine uses the data from both the edge and data samplers to determine the phase of the incoming data stream and to control the phase interpolators (PIs). The phase for the edge sampler is locked to the transition region of the data stream while the phase of the data sampler is positioned in the middle of the data eye.

F. *Comma Detect*

Serial data must be aligned to symbol boundaries before it can be used as parallel data. To make alignment possible, transmitters send a recognizable sequence, usually called a comma. The receiver searches for the comma in the incoming data. When it finds a comma, it moves the comma to a byte boundary so the received parallel words match the transmitted parallel words.

G. *8B/10B Decoder and Encoder*

Many protocols use 8B/10B encoding on outgoing data. 8B/10B is an industry standard encoding scheme that trades two bits overhead per byte for achieved DC-balance and bounded disparity to allow reasonable clock recovery. The GTH transceiver has a built-in 8B/10B TX/RX path to encode or decode TX/RX data without consuming FPGA resources. Enabling the 8B/10B encoder increases latency through the TX path. The 8B/10B encoder can be disabled or bypassed to minimize latency, if not needed [8].

H. *Elastic Buffer*

The GTH transceiver includes an RX elastic buffer to resolve differences between the XCLK and RXUSRCLK domains. The phase of the two domains can also be matched by using the RX recovered clock from the transceiver to drive RXUSRCLK and adjusting its phase to match XCLK when the RX buffer is bypassed. The RX elastic buffer is also used for clock correction and channel bonding.

III. Simulation

As the design is aimed at achieving 10Gbps line rate using GTH. In order to lower the frequency that the FPGA logic operates, we choose to set the parameter TX_DATA_WIDTH to 64 bits. This means that both the RX and TX module of the GTH have a 64 bits width interface to the FPGA logic. So the parallel frequency processed by the FPGA logic is 156.25MHz, which is easily managed by FPGA logic.

A. *RX design*

As the parameter RX_DATA_WIDTH is 64 bits and the RX_INT_DATAWIDTH is 1, which means that the RX part processes the 32 bits width data.

Due to the situation, the RXUSRCLK2 is half of the RXUSRCLK. So, the MMCM module must be used to divide the clock.

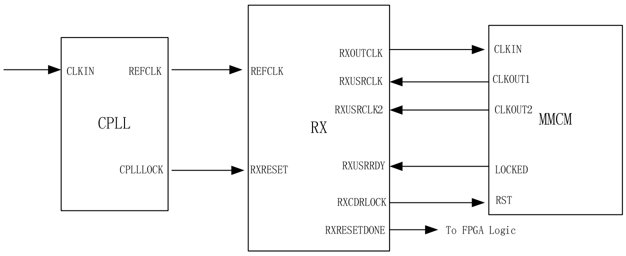

Figure 2. RX clock design

Clock provided outside first goes into CPLL, the channel PLL will lock and track the input clock. When it finishes locking the input clock, the signal CPLLLOCK will be pulled high, which indicates the clock provided by the CPLL is locked successfully. We use the CPLLLOCK signal as the reset signal to RX. This ensures that RX begins to reset itself after achieving a stable clock input [9][10].

As the RXOUTCLK is sourced by the RXCDR module, MMCM shouldn't reset until RXOUTCLK is stable. So we use the output RXCDRLOCK to reset MMCM. RXCDRLOCK high means that RXOUTCLK is locked. At this time, reset the MMCM. However, as the reset symbol to MMCM is active-high, so reverse the RXCDRLOCK before using it to reset MMCM.

MMCM has one clock input, which should be 312.5MHz and two clock outputs, which should be 312.5MHz and 156.25MHz, respectively. The locked signal of the MMCM is used as the RXUSRRDY signal. This means that the RX has finished resetting operation after the clocks generated from the MMCM are stable.

B. TX design

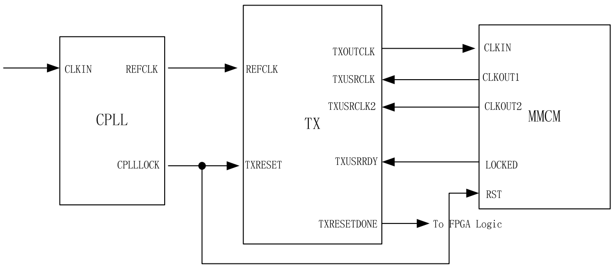

Figure 3. TX clock design

The design of the TX is similar to the RX part. The only difference is that CPLLLOCK is also used for the MMCM reset signal. Unlike the RX, TXOUTCLK is stable as soon as the CPLLLOCK becomes high.

The locked signal provided by the MMCM makes sure that the TX finishes resetting until it achieves two stable clock inputs (TXUSRCLK and TXUSRCLK2).

The TXRESETDONE signal becoming high indicates that the TX has finished resetting, and then the output of the TX is reliable. So, this signal can be used in the FPGA logic.

IV. Simulation Result

During the simulation, we use RX to receive 10Gbps signal, and then transmit the parallel data to TX. Eventually, TX outputs 10Gbps signal which is exactly the same as the input port except for a certain time delay.

State signals shown as below. CPLLLOCK signal goes high means CPLL has finished locking the clock input. RXCDRLOCK goes high and be used as the reset signal to the MMCM. After MMCM locking its two output clocks, its locked signal goes high, and this signal serves as the RXUSRRDY signal to the RX, which ensures that the RX can finish its reset sequentially. Also, MMCM uses the signal CPLLLOCK as its reset at the TX channel.

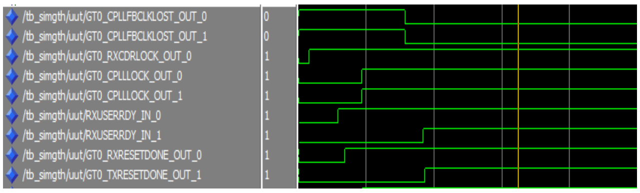
/tb_simgth/uut/GT0_CPLLFBCLKLOST_OUT_0 0
/tb_simgth/uut/GT0_CPLLFBCLKLOST_OUT_1 0
/tb_simgth/uut/GT0_RXCDRLOCK_OUT_0 1
/tb_simgth/uut/GT0_CPLLLOCK_OUT_0 1
/tb_simgth/uut/GT0_CPLLLOCK_OUT_1 1
/tb_simgth/uut/RXUSERRDY_IN_0 1
/tb_simgth/uut/RXUSERRDY_IN_1 1
/tb_simgth/uut/GT0_RXRESETDONE_OUT_0 1
/tb_simgth/uut/GT0_TXRESETDONE_OUT_1 1

Figure 4. Reset succeed and clock locked

The input data is a sequence of 010010010010010...., which can be seen from the picture below, TX really outputs the data sequence as same as the input port. By the way, RX and TX are all set to be LSB first.

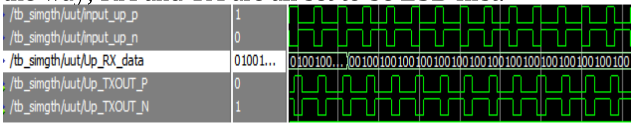
/tb_simgth/uut/input_up_p 1
/tb_simgth/uut/input_up_n 0
/tb_simgth/uut/Up_RX_data 01001...
0100100...
)00100100100100100100100100100100100100100100100100
/tb_simgth/uut/Up_TXOUT_P 0
/tb_simgth/uut/Up_TXOUT_N 1

Figure 5. TX code vs. RX code

As can be seen from the picture below, the period of a single input code is 100ps, which means the frequency is 10GHz.

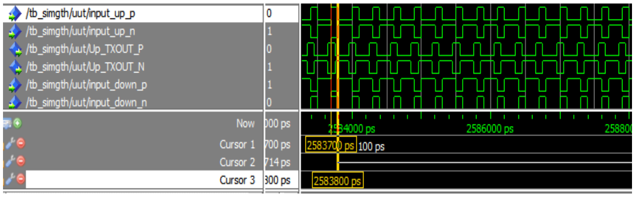
/tb_simgth/uut/input_up_p 0
/tb_simgth/uut/input_up_n 1
/tb_simgth/uut/Up_TXOUT_P 0
/tb_simgth/uut/Up_TXOUT_N 1
/tb_simgth/uut/input_down_p 1
/tb_simgth/uut/input_down_n 0
Now 000 ps
Cursor 1 700 ps
Cursor 2 714 ps
Cursor 3 300 ps
2584000 ps
2586000 ps
25880
2583700 ps
100 ps
2583800 ps

Figure 6. Input code period

As can be seen from the picture below, the period of a single output code is 100ps, which means the frequency is 10GHz.

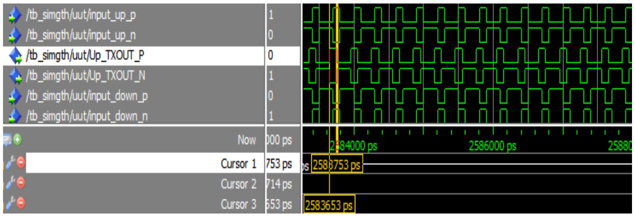
/tb_simgth/uut/input_up_p 1
/tb_simgth/uut/input_up_n 0
/tb_simgth/uut/Up_TXOUT_P 0
/tb_simgth/uut/Up_TXOUT_N 1
/tb_simgth/uut/input_down_p 0
/tb_simgth/uut/input_down_n 1
Now 000 ps
Cursor 1 753 ps
Cursor 2 714 ps
Cursor 3 653 ps
2584000 ps
2586000 ps
25880
2583753 ps
2583653 ps

Figure 7. Output code period

V. Conclusions

This paper finishes a research on the theory and simulation of GTH transceiver, and then proves that the GTH supports high-speed data transmission that the rate up to 10Gbps. It's really useful for high-speed data communication.

References

[1] Altera Inc. The Evolution of High-Speed Transceiver Technology [Z]. 2002. 11.

[2] WU Rongwei, SU Tao, LIANG Zhongying. Application of RocketIO in High-speed Data Communication [J].Communications Technology,2010,43(11):9-11.

[3] He Bin,Liu Fengxin.The design interface chip in the signal processing system [J].Computer Engineering and Application, 2004,40(33):118-120,197.

[4] Abhijit Athavale,Carl Christensen,High-speed Serial I/O Made Simple[Z].Xilinx inc.,2005.4.

[5] ZHAO Zhengrong, LAN Julong.Solution of several key problems about RocketIO [J].Application of Electronic Technique,2005, 31 (12):51-53.

[6] Xilinx.7 Series FPGAs GTX/GTH Transceivers User Guide [Z] .May 7 2012.

[7] SAVAGE S.Implementing high-speed serial and custom digital protocols thru FPGA technology and graphical progrannimg techniques[C].Baltimore, 2007 IEEE Autotestcon,2007:214-223.

[8] Xilinx.LogiCORE IP Aurora 8B/10B V7.1 user guide UG766 [Z].2011.

[9] SUN Hang.The advanced application and design techniques of Xilinx PLD[M].Beijing:Publishing House of Electronics Industry, 2004.

[10] Mentor Graphics.ModelSim SE User’s Manual[EB/OL]. [2015-3] WWW.modelsim.com.