Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

High-speed, fixed-latency serial links with Xilinx FPGAs*

Xue LIU†1, Qing-xu DENG†‡1, Bo-ning HOU1, Ze-ke WANG2

(1Institute of Cyber-Physical System Engineering, Northeastern University, Shenyang 110004, China)

(2Institute of Digital Technology and Instruments, Zhejiang University, Hangzhou 310027, China)

†E-mail: liuxue0512@gmail.com; dengqx@mail.neu.edu.cn

Received Sept. 11, 2013; Revision accepted Dec. 18, 2013; Crosschecked Jan. 15, 2014

Abstract: High-speed, fixed-latency serial links find application in distributed data acquisition and control systems, such as the timing trigger and control (TTC) system for high energy physics experiments. However, most high-speed serial transceivers do not keep the same chip latency after each power-up or reset, as there is no deterministic phase relationship between the transmitted and received clocks after each power-up. In this paper, we propose a fixed-latency serial link based on high-speed transceivers embedded in Xilinx field programmable gate arrays (FPGAs). First, we modify the configuration and clock distribution of the transceiver to eliminate the phase difference between the clock domains in the transmitter/receiver. Second, we use the internal alignment circuit of the transceiver and a digital clock manager (DCM)/phase-locked loop (PLL) based clock generator to eliminate the phase difference between the clock domains in the transmitter and receiver. The test results of the link latency are shown. Compared with existing solutions, our design not only implements fixed chip latency, but also reduces the average system lock time.

Key words: Data acquisition circuit, Fixed-latency, Field programmable gate array (FPGA), Serial link, Trigger system
doi:10.1631/jzus.C1300249 **Document code:** A **CLC number:** TN79

1 Introduction

High-speed serial links are important components of distributed data acquisition and control systems. In these systems, the clock, reset, and trigger signals, along with the data, are transferred across the serial links, so it is crucial to keep the latency of the serial links constant after each power-up or reset. However, most commercial high-speed serial transceiver chips do not support this important feature, which needs additional hardware processing circuits and is usually not required for most communication applications. For example, the chip latency of the Texas Instruments SCAN25100 varies after each reset or power-up (TI, 2013). A typical distributed system is the timing, trigger and control (TTC) system, developed by CERN for high energy physics experiments. It distributes TTC signals with constant latency and predictable phase. It has successfully been used as the trigger subsystem of the Large Hadron Collider (LHC) experiments (Taylor, 1998). The GigaBit Transceiver project (Moreira *et al.*, 2007) and White Rabbit, launched by CERN, were used to address possible problems of data and timing information transfer in future super LHC experiments.

Up to now, several fixed-latency serial links, based on high-speed transceivers embedded in field programmable gate arrays (FPGAs), have been proposed to address system timing and synchronization problems in many applications, such as the data acquisition circuits of the Compressed Baryonic Matter experiment (Lemke *et al.*, 2010), the Positron Emission Tomography system (Aliaga *et al.*, 2011), and the future KM3NeT undersea network (Le-Provost *et al.*, 2011). Several fixed-latency serial links with Xilinx

‡ Corresponding author

* Project supported by the National Science and Technology Support Program of China (No. 2012BAK24B01), the Fundamental Research Funds for the Central Universities, China (No. N100204001), the Specialized Research Fund for the Doctoral Program of Higher Education, China (No. 20110042110021), and the National Science Foundation for Post-doctoral Scientists of China (No. 2013M541243)

FPGA have been published (Aloisio *et al.*, 2009; 2010). Other implementations (Giordano and Aloisio, 2011; 2012; Aloisio *et al.*, 2013) relieve the requirement for inputting the same clock both to the transmitter and to the receiver. They use the internal comma alignment and detection circuit of the GTP/GTX transceivers (Xilinx, 2009a; 2009b) to eliminate the clock phase offset between the transmitted clock and the received clock, which is one of the main reasons for latency variation in serial links. However, the internal alignment circuit can process only the clock phase offsets with even unit intervals (UIs, 1 UI is the duration of a serial symbol). For the odd-UI clock phase offsets, their solution is to reset the serial receiver and to wait for the receiver to relock. This process will proceed repetitively until the even-UI clock phase offsets are detected. This increases the average system lock time.

In this paper, we propose a fast-lock fixed-latency serial link based on both the internal alignment circuit of the serial transceiver and a digital clock manager (DCM)/phase-locked loop (PLL) based clock generator. Compared with Aloisio's designs (Giordano and Aloisio, 2011; 2012; Aloisio *et al.*, 2013), our serial link reduces the average system lock time. The reason is that it can process all possible clock phase offsets and eliminate the reset-relock process present in Aloisio's designs.

2 Latency variation in SerDes transceivers

Link latency may vary in both the serial section and parallel section of a SerDes transceiver. In the serial section, clock multiplication and subsequent division may cause phase differences between the transmitted clock and the received clock, leading to latency variation of the data. In the parallel section, the buffers in the transmitter and receiver may also change the link latency.

We first focus on the serial section. The reference clock CLK_IN of the transceiver (Fig. 1) is multiplied by a factor of four and the serial clock is the resulting clock. In the serializer, the serial clock is used by a parallel input to serial output (PISO) block for serializing the parallel transmitted data TXDATA. The transmitted clock TX_CLK can be considered as a copy of clock CLK_IN. In the deserializer, the clock data recovery (CDR) circuit extracts a recovered clock from the serial data stream. We omit the latency in the transmission medium, so the recovered clock can be considered as a copy of the serial clock. The recovered clock is used by a serial input to parallel output (SIPO) block for deserializing the serial data and presenting it as the parallel received data RXDATA. The received clock RX_CLK comes from the division of the recovered clock. From Fig. 1, we find that there are four possible phase differences

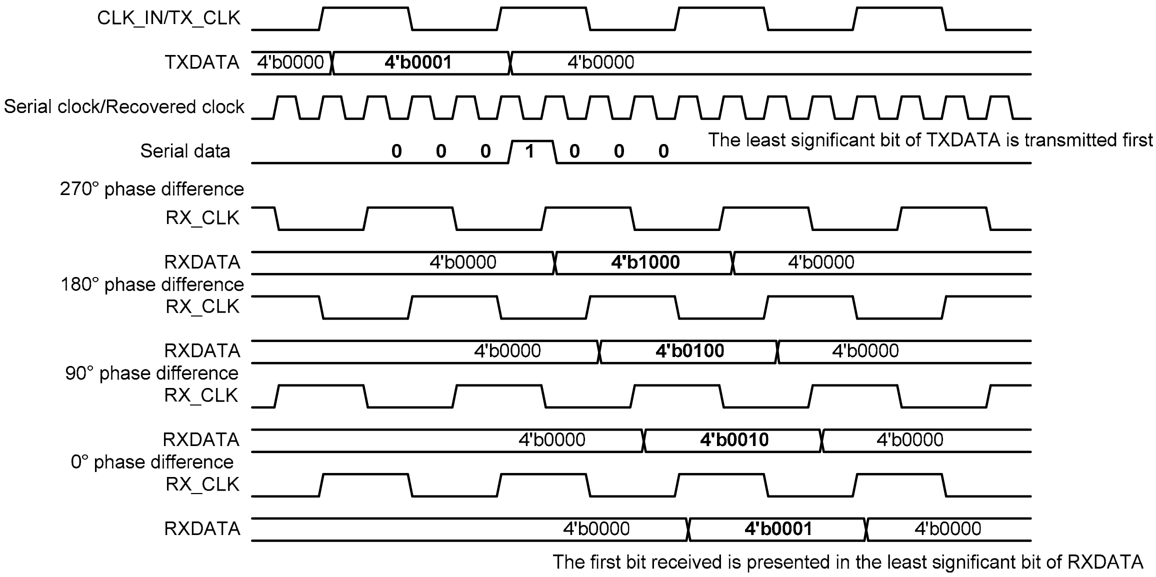


Fig. 1 Frequency multiplication and subsequent division of CLK_IN

between TX_CLK and RX_CLK. These phase differences lead to the latency variation of the data. This type of latency variation is expressed in UI. The latency variation in the parallel section is due to the buffers and is expressed in parallel clock period T. In summary, the link latency variation ΔL is defined by

$$\Delta L=N\cdot\mathrm{UI}+M\cdot T, \tag{1}$$

where N and M are integers decided by the transmission protocol and the implementation of the SerDes transceiver.

Giordano and Aloisio (2011) concluded that a straightforward solution to the fixed-latency serial link was to make the phase differences between TX_CLK and RX_CLK constant. Their experimental results indicate that the internal alignment circuit of the GTX/GTP transceiver can perform only the even-UI phase-shifts of RX_CLK. Based on this important feature, they proposed a fixed-latency serial link with lower hardware cost. For the odd-UI phase differences between TX_CLK and RX_CLK, their solution was to reset the transceiver. This reset process will repeat until the even-UI phase offsets are detected. This approach increases the average system lock time.

3 Proposed link architecture

In this section, we present a specific implementation of a fast-lock fixed-latency serial link based on the GTX transceiver embedded in Xilinx FPGA (Xilinx, 2009b). Our serial link can process all possible clock phase differences produced in the serial−parallel conversion process. Compared with the design of Giordano and Aloisio (2011), it reduces the average system lock time at the expense of relatively high hardware cost.

3.1 Overall architecture

Our fast-lock fixed-latency serial link based on the 8b/10b protocol (Widmer and Franaszek, 1983) is shown in Fig. 2. The whole design consists of a payload generator, two GTX transceivers, a TX phase alignment block, an RX phase alignment block, and a clock and data alignment (CDA) block. There is no deterministic phase relationship between the parallel clock for the SIPO block (XCLK) and the parallel clocks for the FPGA logic (RXUSRCLK and RXUSRCLK2) after each power-up and reset. To solve this problem, we first connect RXUSRCLK/RXUSRCLK2 with RX_CLK. This results in RXUSRCLK/RXUSRCLK2 and RX_CLK running with a deterministic phase relationship. Then we use the internal alignment circuit of the GTX transceiver to align XCLK with RX_CLK. The RX phase alignment block drives the RXPMA_SETPHASE signal to activate the phase alignment circuit. The procedure for activating the alignment circuit can be found in the GTX user guide (Xilinx, 2009b). Under this condition, we can bypass the elastic buffer in the receiver. Similarly, we connect TXUSRCLK/TXUSRCLK2 with TX_CLK, and then we use the internal phase alignment circuit to align TX_CLK with the parallel clock for the PISO block. Under this condition, we can bypass the phase adjustment FIFO in the transmitter. Therefore, we eliminate the effect of the buffers on latency variation. The major difference between our solution and Aloisio's design is that

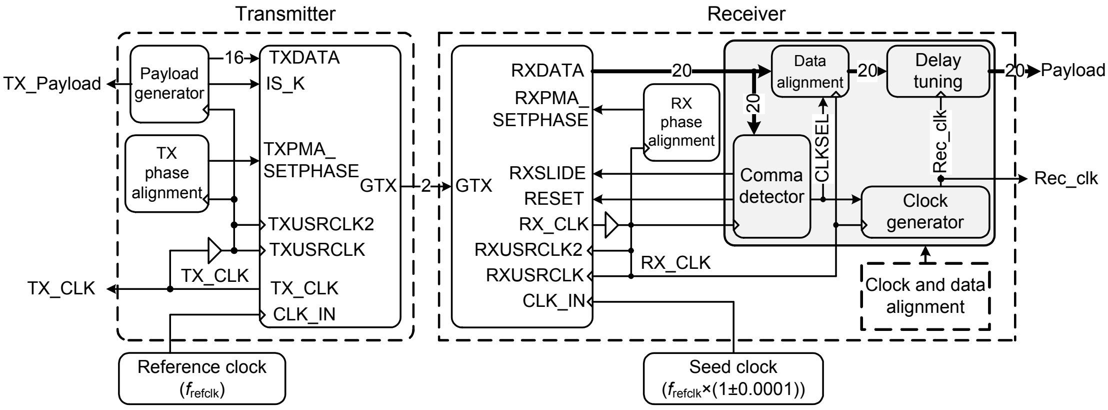


Fig. 2 Proposed fast-lock fixed-latency serial link

we use both the internal alignment circuit and a DCM/PLL-based clock generator to process all possible clock phase differences between TX_CLK and RX_CLK.

3.2 Clock and data alignment

The CDA block consists of a clock generator, a data alignment block, a delay tuning block, and a comma detector (Fig. 2). In the deserializer, a recovered clock is extracted from the serial data stream, and then divided to obtain the parallel received clock RX_CLK. The clock generator block provides a phase shift of the clock RX_CLK, and then outputs a clock Rec_clk to the delay tuning block. The clock Rec_clk has the same frequency as RX_CLK with a phase shift of 180° or (180°+360°/*N*), where *N* is the width of the internal data-path of the GTX transceiver. For the even-UI phase offsets between TX_CLK and RX_CLK, the phase-shift value is 180°. For the odd-UI phase offsets, the phase-shift value is (180°+360°/*N*). A DCM and a PLL embedded in Xilinx Virtex-5 FPGA (Xilinx, 2012) are used to construct the clock generator (Fig. 3a). The CLKSEL signal, driven by the comma detector, is used to decide the output phase-shift value of the Rec_clk. When CLKSEL is low, the phase-shift value is 180°. When CLKSEL is high, the phase-shift value is (180°+360°/*N*). The DCM allows fine-grained phase shifting, and the output phase can be dynamically and repetitively moved forwards and backwards. The phase shift type of DCM in this study was set to the fixed mode and the corresponding phase-shift value was set to 360°/*N*. The PLL allows coarse-grained phase shifting and the phase-shift value of the output clock was set to 180°. The excessive jitter of RX_CLK may cause the DCM and PLL to lose lock, so the CLK_IN input of the receiver should be driven by a seed clock with a frequency offset smaller than 0.0001 with respect to the transmitted clock.

In the deserializer, the SIPO block deserializes the serial data and presents it as parallel data. The data alignment block achieves the bit shifts for the parallel data. The delay tuning block transfers the bit-shifted parallel data from the RX_CLK clock domain to the Rec_clk clock domain. The latency produced in the data transfer process is constant and is not affected by the phase offsets. The clock Rec_clk is generated for two reasons. First, we select the proper phase for Rec_clk to meet the setup and hold time requirements for correct data transfer across two different clock domains. Second, the Rec_clk is used to ensure that the transfer latency is constant. The data alignment and delay tuning blocks consist of one multiplexer and two register groups, namely FF_A and FF_B (Fig. 3b). The CLKSEL signal, driven by the comma detector, is used to decide the output value of the data alignment block. When CLKSEL is low, FF_A[*N*−1:0] is chosen as the MUX output. When CLKSEL is high, {RXDATA[0], FF_A[*N*−1:1]} is chosen.

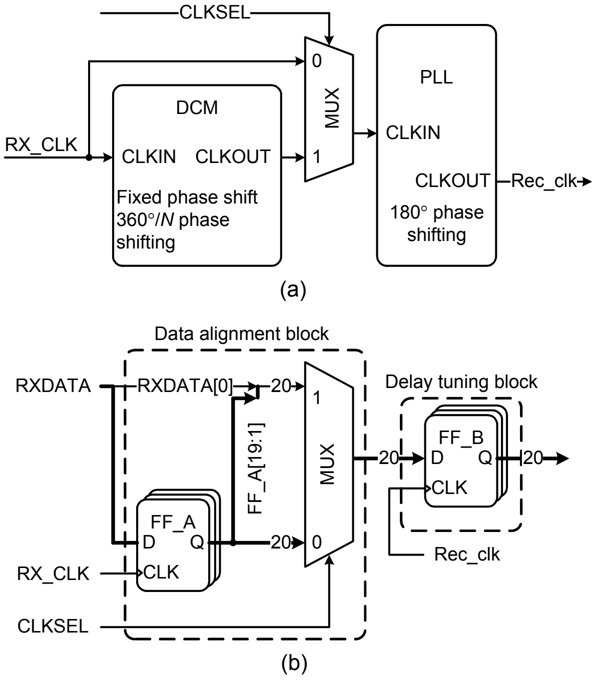


Fig. 3 Structures of the clock generator (a) and data alignment and delay tuning blocks (b)

From the hardware test results, we found that the bit-shift value *n* of the parallel data and the phase difference Δ*P* between TX_CLK and RX_CLK satisfy the following equation:

$$\Delta P = n\times 360°/N,\quad n=0, 1, \ldots, N-1, \tag{2}$$

where *N* is the width of the internal data-path of the GTX transceiver. The comma detector searches for commas in the parallel data stream and determines its bit-shift value *n*. Once it finds one, it then obtains the clock phase differences Δ*P*, which may vary after each power-up or reset. By driving the RXSLIDE signal of the GTX transceiver, the comma detector enables the internal alignment circuit to perform the clock phase-shift and the parallel data bit-shift. By driving the CLKSEL signal, the comma detector enables the clock generator and data alignment blocks to perform the clock phase-shift and the parallel data bit-shift, according to the following algorithm:

1. If n is even, it drives the RXSLIDE to perform an n-UI phase-shift of the clock and an n-bit bit-shift of the data. The CLKSEL signal is driven to low level.

2. If n is odd, it drives the RXSLIDE to perform an $(n-1)$-UI phase-shift of the clock and an $(n-1)$-bit bit-shift of the data. The CLKSEL signal is driven to high level.

Compared with the design of Giordano and Aloisio (2011), our solution can process all possible clock phase offsets and eliminate the reset-relock process, thereby reducing the average system lock time. However, this improvement comes at the expense of increased FPGA resources, including DCM, PLL, and logic resources.

3.3 Operational details of the CDA

The internal data-path width N for the GTX transceiver is set to 20, so n is in the range of 0–19. For clarity, the transmitted data after the 8b/10b encoder is replaced with a sequence containing several zero words and one word $(0\_0001)_{16}$. The word $(0\_0001)_{16}$ presents the comma, and the zero words present the data. The comma detector obtains the bit-shift n from the location of the '1' in the parallel received data. By Eq. (2), it computes the phase offset ΔP between RX_CLK and TX_CLK. According to ΔP, it activates the internal alignment circuit to perform the clock phase-shift and the parallel data bit-shift. By driving the CLKSEL signal, it enables the clock generator and data alignment blocks to perform the clock phase-shift and the parallel data bit-shift. The data alignment and delay tuning blocks achieve the bit-shift and delay adjustment for the raw parallel data. The clock generator generates a clock Rec_clk for the delay tuning and FPGA logic, so our ultimate goal is to make the phase difference between TX_CLK and Rec_clk constant.

We provide several representative phase offsets to illustrate the operational details (Fig. 4). The chosen values for the phase offset ΔP are 288°, 216°, 270°, and 234°. The corresponding values for the bit-shift n are 16, 12, 15, and 13, respectively. Here, we omit the link latency in the transmission medium.

When ΔP is 288°, namely the even-UI phase offset, the comma detector enables the internal alignment circuit to perform a 16-UI phase-shift and 16-bit bit-shift. After internal phase shifts and bit shifts, the phase difference between RX_CLK and TX_CLK is 0°. For the even-UI phase offsets, FF_A[19:0] is chosen as the output (input of FF_B) of the data alignment block (the delay tuning block). The clock generator generates the clock Rec_clk with the same frequency as RX_CLK phase-shifted by 180°, so the phase difference between TX_CLK and Rec_clk is 180°. The setup and hold times relative to Rec_clk are $T/2$, where T is the period of Rec_clk (Fig. 4a). When ΔP is 270°, namely the odd-UI phase offset, the comma detector enables the internal alignment circuit to perform a 14-UI phase-shift and 14-bit bit-shift. After internal phase shifts and bit shifts, the phase difference between RX_CLK and TX_CLK is 18°. For the odd-UI phase offsets, {RXDATA[0], FF_A[19:1]} is chosen as the output (input of FF_B) of the data alignment block (the delay tuning block). The clock generator generates the clock Rec_clk with the same frequency as RX_CLK phase-shifted by 198°, so the phase difference between TX_CLK and Rec_clk is still 180°. The setup and hold times relative to Rec_clk are close to $11T/20$ and $9T/20$, respectively (Fig. 4b).

Utilizing the phase-shifted clock Rec_clk, we not only meet the setup and hold time requirements for correct data transfer across two different clock domains, but also adjust the delay value in the data transfer process. The link latency ΔL is always the same, no matter how many phase offsets are produced (Fig. 4).

4 Experimental results

Table 1 presents the hardware resources of Aloisio's design (Giordano and Aloisio, 2011; 2012; Aloisio *et al.*, 2013) and our serial link based on the GTX transceiver in Xilinx FPGA. The link rate is 2.5 Gb/s and the transmission standard is 8b/10b encoding (Widmer and Franaszek, 1983). Our design uses more hardware resources. The reason is that it uses both the internal circuit and a DCM/PLL-based clock generator to implement the data bit-shifting and the clock phase-shifting.

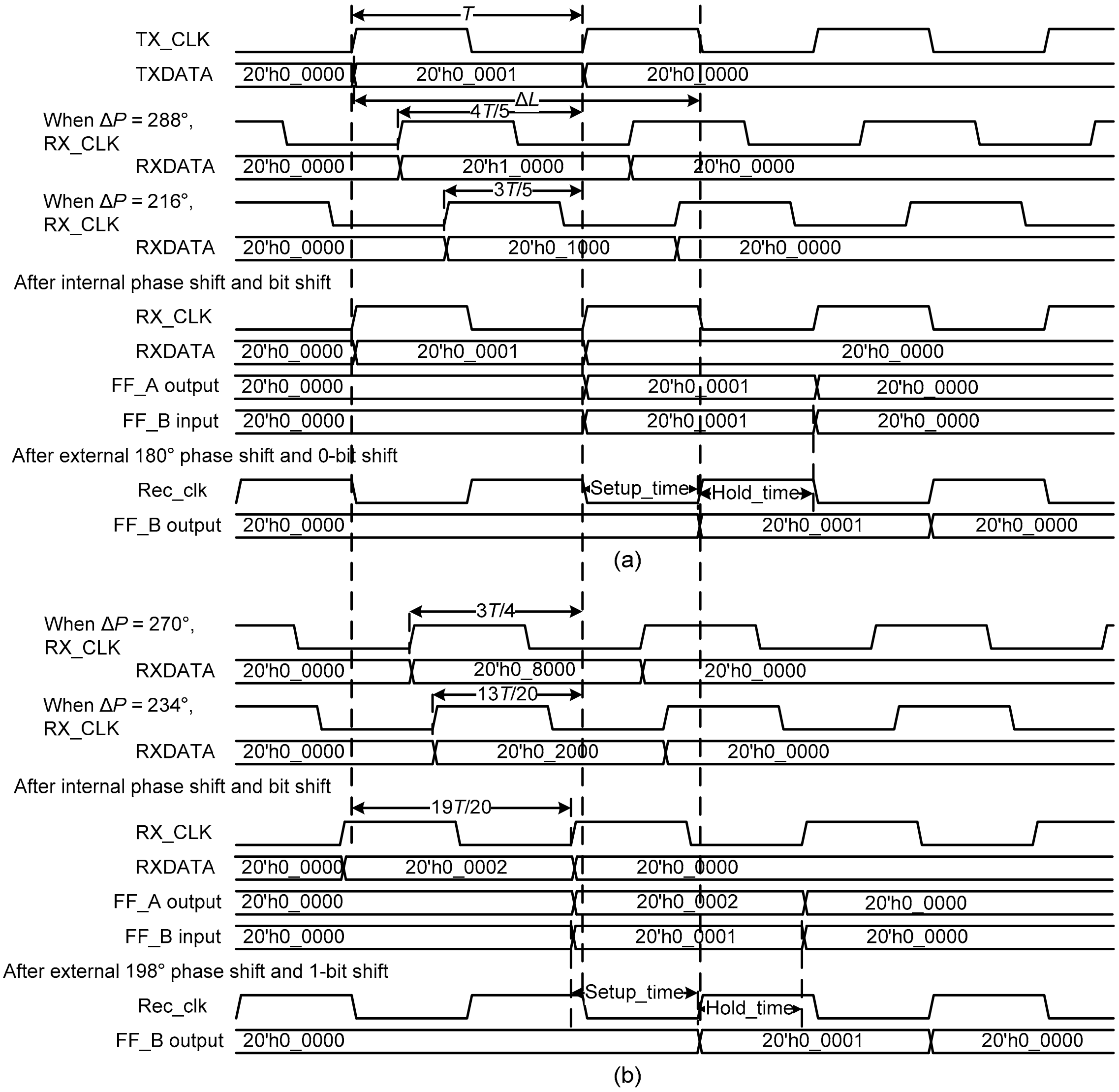


Fig. 4 Operational details of clock and data alignment (CDA)
(a) Even-UI phase offsets, ΔP=288°, 216°; (b) Odd-UI phase offsets, ΔP=270°, 234°

Table 1 Hardware resources of Aloisio's design and our serial link

Resource	Amount	
	Aloisio's design*	Our serial link
Slice register	130	140
Slice LUT	201	230
Block RAM	1	1
BUFG	4	7
DCM_ADV	0	1
PLL_ADV	0	1
GTX_DUAL	1	1

* Giordano and Aloisio (2011)

We used two Xilinx development boards (Xilinx, 2011) to test our serial link, and a pair of 50 Ω impedance coaxial cables to connect the GTX transmitter and receiver. The test points for the transmitted and received payloads were available in the expansion interface and were monitored by a digital oscilloscope (Fig. 5). We found that the latency of our serial link was always the same during data transfers and between power cycles of the system. We performed 5000 tests. In each test, we reset the GTX transceiver to simulate a power-up process and recorded the latency results on the oscilloscope screen. We also tested the link latency of Aloisio's design for comparison. The histogram in Fig. 6 shows the distribution of link latency of Aloisio's design. The corresponding standard deviation is ~44 ps and the synchronization precision between the transmitted clock and the recovered clock RX_CLK, namely the range of the histogram in Fig. 6, is ~516 ps. The histogram in Fig. 7 shows the distribution of our link latency. The corresponding standard deviation is ~43 ps and the synchronization precision between the transmitted clock and the phase-shifted clock Rec_clk is ~493 ps. Thus, our link architecture has almost the

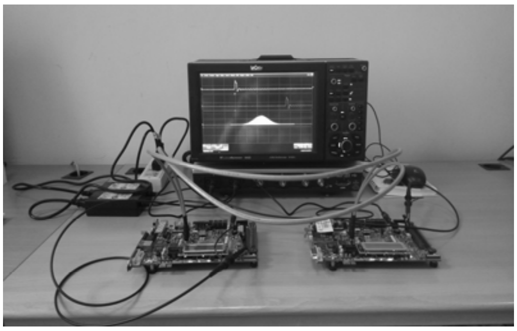
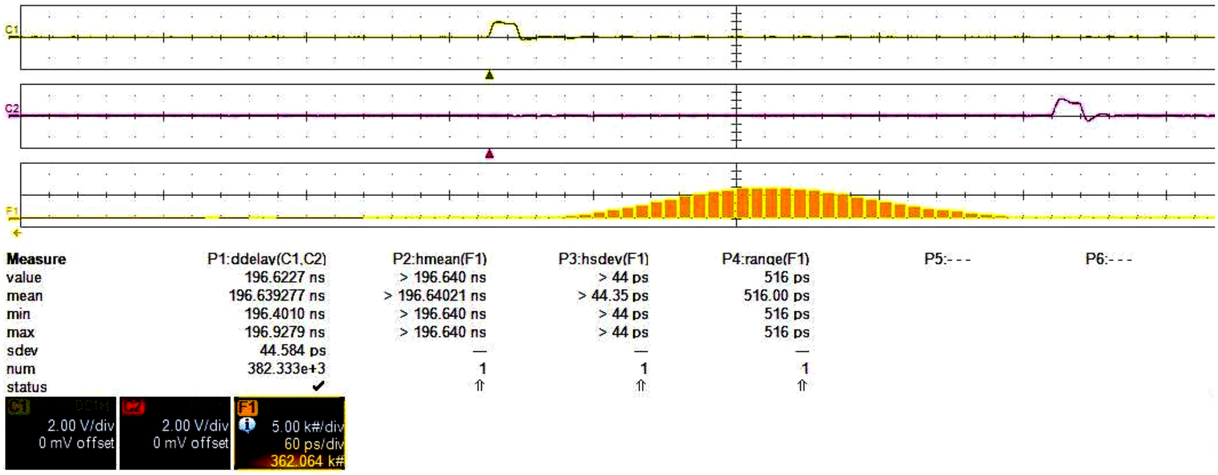


Fig. 5 Test platform for our serial link



Measure	P1:ddelay(C1,C2)	P2:hmean(F1)	P3:hsdev(F1)	P4:range(F1)	P5:- - -	P6:- - -
value	196.6227 ns	> 196.640 ns	> 44 ps	516 ps		
mean	196.639277 ns	> 196.64021 ns	> 44.35 ps	516.00 ps		
min	196.4010 ns	> 196.640 ns	> 44 ps	516 ps		
max	196.9279 ns	> 196.640 ns	> 44 ps	516 ps		
sdev	44.584 ps	—	—	—		
num	382.333e+3	1	1	1		
status	✔	⇑	⇑	⇑		

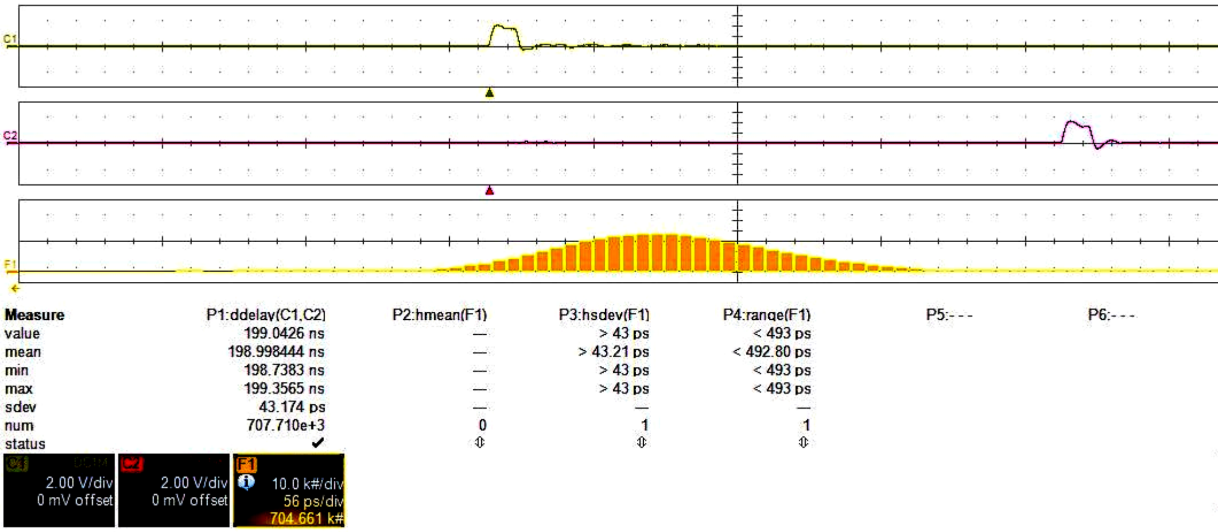
C1 2.00 V/div 0 mV offset

C2 2.00 V/div 0 mV offset

F1 5.00 k#/div 60 ps/div 362.064 k#

Fig. 6 Histogram of the latency of Aloisio's design (Giordano and Aloisio, 2011)

Upper trace: transmitted trace; middle trace: corresponding received trace; bottom trace: histogram of the latency. P1 is the measurement of delay between channels 1 and 2. F1 is the histogram distribution of delay. Three histogram parameters hmean, hsdev, and range are the mean, standard deviation, and range of the histogram distribution, respectively



Measure	P1:ddelay(C1,C2)	P2:hmean(F1)	P3:hsdev(F1)	P4:range(F1)	P5:- - -	P6:- - -
value	199.0426 ns	—	> 43 ps	< 493 ps		
mean	198.998444 ns	—	> 43.21 ps	< 492.80 ps		
min	198.7383 ns	—	> 43 ps	< 493 ps		
max	199.3565 ns	—	> 43 ps	< 493 ps		
sdev	43.174 ps	—	—	—		
num	707.710e+3	0	1	1		
status	✔	⇕	⇕	⇕		

C1 2.00 V/div 0 mV offset

C2 2.00 V/div 0 mV offset

F1 10.0 k#/div 56 ps/div 704.661 k#

Fig. 7 Histogram of the latency of our solution

Upper trace: transmitted trace; middle trace: corresponding received trace; bottom trace: histogram of the latency. P1 is the measurement of delay between channels 1 and 2. F1 is the histogram distribution of delay. Three histogram parameters hmean, hsdev, and range are the mean, standard deviation, and range of the histogram distribution, respectively same performance as Aloisio's design (Figs. 6 and 7). The distributions are due only to the jitter between the transmitted clock and the received clock.

Let T_{GTX} be the lock time when only the initial reset of the GTX transceiver is required for Aloisio's design. When the odd-UI phase offsets are detected, there is a 50% chance of achieving lock from the first attempt, 25% of cases will require one extra reset, 12.5% will need two, etc. In total, the average lock time of Aloisio's design will be $2T_{\mathrm{GTX}}$. In our design, the system lock time will be the sum of T_{GTX}, T_{DCM}, and T_{PLL}, namely ($T_{\mathrm{GTX}}+T_{\mathrm{DCM}}+T_{\mathrm{PLL}}$). T_{DCM} and T_{PLL} are the lock times of the DCM and PLL, respectively. Real hardware tests indicate that T_{GTX}, T_{DCM}, and T_{PLL} are about 240, 35, and 2 μs, respectively. Compared with Aloisio's design, our design reduces the average system lock time.

5 Conclusions

Distributed data acquisition and control systems based on a network of high-speed serial links require the latency of the serial links to be constant after each power-up or reset. However, most of the commercial serial transceiver chips do not support this feature, which is usually not required for communication applications. In this paper, we present a fast-lock fixed-latency serial link based on a transceiver embedded in Xilinx FPGA. Compared with existing serial links, our solution not only implements fixed-latency transfer operations, but also reduces the average system lock time. The reason is that our solution can process all possible clock phase offsets produced in the serializing and deserializing conversion.

References

Aliaga, R.J., Monzo, J.M., Spaggiari, M., *et al.*, 2011. PET system synchronization and timing resolution using high-speed data links. *IEEE Trans. Nucl. Sci.*, **58**(4):1596-1605. [doi:10.1109/TNS.2011.2140130]

Aloisio, A., Cevenini, F., Giordano, R., *et al.*, 2009. High-speed, fixed-latency serial links with FPGAs for synchronous transfers. *IEEE Trans. Nucl. Sci.*, **56**(5): 2864-2873. [doi:10.1109/TNS.2009.2027236]

Aloisio, A., Cevenini, F., Giordano, R., *et al.*, 2010. Emulating the GLink chip set with FPGA serial transceivers in the ATLAS Level-1 Muon trigger. *IEEE Trans. Nucl. Sci.*, **57**(2):467-471. [doi:10.1109/TNS.2009.2036175]

Aloisio, A., Ameli, F., Bocci, V., *et al.*, 2013. Design, implementation and test of the timing trigger and control receiver for the LHC. *J. Instrum.*, **8**(2):T02003. [doi:10.1088/1748-0221/8/02/T02003]

Giordano, R., Aloisio, A., 2011. Fixed-latency, multi-gigabit serial links with Xilinx FPGAs. *IEEE Trans. Nucl. Sci.*, **58**(1):194-201. [doi:10.1109/TNS.2010.2101083]

Giordano, R., Aloisio, A., 2012. Protocol-independent, fixed-latency links with FPGA-embedded SerDeses. *J. Instrum.*, **7**(5):P05004. [doi:10.1088/1748-0221/7/05/P05004]

Lemke, F., Slogsnat, D., Burkhardt, N., *et al.*, 2010. A unified DAQ interconnection network with precise time synchronization. *IEEE Trans. Nucl. Sci.*, **57**(2):412-418. [doi:10.1109/TNS.2010.2042176]

Le-Provost, H., Moudden, Y., Anvar, S., *et al.*, 2011. A readout system-on-chip for a cubic kilometer submarine neutrino telescope. *J. Instrum.*, **6**(12):C12044. [doi:10.1088/1748-0221/6/12/C12044]

Moreira, P., Marchioro, A., Kloukinas, K., 2007. The GBT: a proposed architecture for multi-Gb/s data transmission in high energy physics. Proc. Topical Workshop on Electronics for Particle Physics, p.332-336.

Taylor, B.G., 1998. TTC distribution for LHC detectors. *IEEE Trans. Nucl. Sci.*, **45**(3):821-828. [doi:10.1109/23.682644]

Texas Instruments (TI), 2013. SCAN25100 2457.6, 1228.8, and 614.4 Mbps CPRI SerDes with Auto RE Sync and Precision Delay Calibration Measurement. Available from http://www.ti.com.cn/cn/lit/ds/symlink/scan25100.pdf

Widmer, A.X., Franaszek, P.A., 1983. A DC-balanced, partitioned-block, 8b/10b transmission code. *IBM J. Res. Devel.*, **27**(5):440-451. [doi:10.1147/rd.275.0440]

Xilinx, 2009a. UG196 Virtex-5 FPGA RocketIO GTP Transceiver User Guide v2.1. Available from http://www.xilinx.com/support/documentation/user_guides/ug196.pdf

Xilinx, 2009b. UG198 Virtex-5 FPGA RocketIO GTX Transceiver User Guide v3.0. Available from http://www.xilinx.com/support/documentation/user_guides/ug198.pdf

Xilinx, 2011. UG347 ML505/ML506/ML507 Evaluation Platform User Guide v3.1.2. Available from http://www.xilinx.com/support/documentation/boards_and_kits/ug347.pdf

Xilinx, 2012. UG190 Virtex-5 FPGA User Guide v5.4. Available from http://www.xilinx.com/support/documentation/user_guides/ug190.pdf