International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Design of Fixed Latency Serial Transceiver on FPGA

Mallika Patil

PG Student, ECE Department, VTU RC Gulbarga, Karnataka, India

Abstract: ***Fixed-latency serial links are essential components of the distributed measurement and control systems. Serial interfaces are generally used for chip-to-chip and board-to-board data transfers. However, largely high-speed Serializer-Deserializer (SerDes) chips do not keep the similar link 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 project, a fixed-latency serial link based on high-speed transceivers embedded in Xilinx field programmable gate arrays (FPGAs) has been designed. This implementation choice is often made because fixed-latency operations require dedicated circuitry. A fixed-latency serial link is established based on techniques such as dynamic clock phase shifting (DCPS) and changeable delay tuning (CDT). Where a DCPS block utilizes a digital clock manager (DCM)-phase-locked loop (PLL) based clock generator to remove the phase difference between the clock domains in the transmitter and receiver. Our solution can process all possible phase offsets between the transmitted and received clocks, so it relaxes the necessity of fanning in the same reference clock both to the transmitter and to the receiver.We present a specific example of implementation based on the serial transceiver in FPGA.***

Keywords: Changeable delay tuning, dynamic clock phase shifting, fixed-latency, FPGA, SerDes transceiver.

1. Introduction

Serial interconnects form the critical backbone of modern communication systems, so the choice of serializer/deserializer (SerDes) can have a big impact on system cost and performance. When most system designers look at serializer/deserializer (SerDes) device, they often compare speed and power without considering how the SerDes works and what it actually does with their data. Internal SerDes architecture may seem to be irrelevant, but this overlooked item can dictate many important system parameters like system topology, protocol overhead, data formatting and flow.

A Multi-Gigabit Transceiver (MGT) is a SerDes capable of operating at serial bit rates above 1 Gigabit/second. MGTs are used increasingly for data communications because they can run over longer distances, use fewer wires, and thus have lower costs than parallel interfaces with equivalent data throughput.

Like other SerDes, the primary function of the MGT is to transmit parallel data as stream of serial bits, and convert the serial bits it receives to parallel data. The most basic performance metric of an MGT is its serial bit rate, or line rate, which is the number of serial bits it can transmit or receive per second. Although there is no strict rule, MGTs can typically run at line rates of 1 Gigabit/second or more. MGTs have become the 'data highways' for data processing systems that demand a high in/out raw data input and output (e.g. video processing applications). They are becoming very common on FPGA - such programmable logic devices being especially well fitted for parallel data processing algorithms. We briefly summarize some representative works in the field of serial links for application to distributed systems. A typical example is the Timing, Trigger, and Control (TTC) system, which has successfully been used as the trigger systems of

2. Latency Variations of Transceiver

Link latency may vary in both the serial section and parallel section of a SERDES transceiver. The serial section and parallel section have been differentiated based on the nature of data flow. In the blocks of transceiver, if the data is in the form of serial bit stream it is referred as serial section and if the data is in the form of parallel bits, it is referred as parallel section.

In serial section, the reference clock CLK_In of the transceiver 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. in our implementation, the clock and data recovery (CDR) circuit is included in the SIPO block. 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. We find that there are four possible phase differences 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 unit interval (UI).

The latency variation in the parallel section is due to the buffers and is expressed in parallel clock period t. Hence, the link latency variation ΔL is defined by

$$\Delta L = N * UI + M * T \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

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

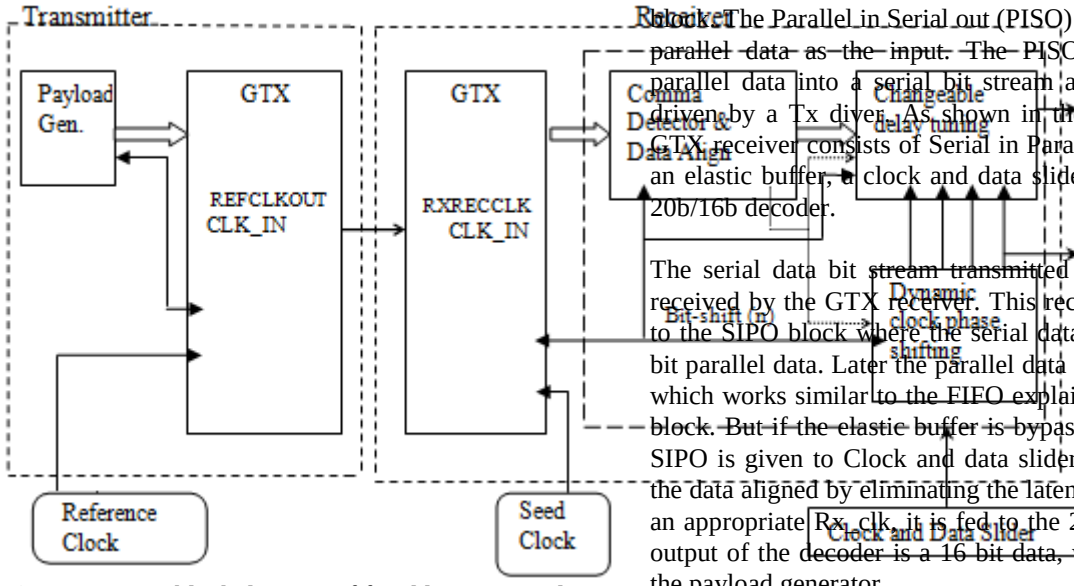

Figure 1: Main block diagram of fixed latency serial transceiver.

3. Proposed Architecture

From the above figure.1 we can infer that both the transmitter and receiver are fanned in with different clocks. The complete design consists of a Payload Generator, two GTX transceivers and one Clock and Data Slider (CDS). By modifying the configuration and clock distribution of the GTx transceiver, they can bypass the internal data buffers in the transmitter or receiver (internal data buffers are explained in next section), so the phase difference between the clock domains within the transmitter or receiver can be eliminated. The internal circuit of the GTX transceiver is used to align the transmitter clock with the receiver clock, so the phase difference between and can be eliminated. Instead of using the internal alignment circuit of the GTX transceiver, we use the external logic circuit, called Clock and Data Slider (CDS), to perform the dynamic phase-shift for RXRECCLK and the changeable delay-tune for received RXDATA. As shown in Fig. 1, the CDS consists of one Comma Detector and Data Alignment (CDDA) block, one Dynamic Clock Phase Shifting (DCPS) block, and one Changeable Delay Tuning (CDT) block.

3.1 GTX Transmitter

The figure.2 gives the block diagram of GTX transmitter. Both the GTX transmitter and receiver consists two layers Physical medium attachment layer (PMA) and physical coding sublayer (PCS). The PMA is responsible for serializing the parallel data and de-serializing the serial data, while the PCS is used to process the data before serialization and after de-serialization As shown in the fig.2, the GTX transmitter consists of an Encoder, a First in first out (FIFO) and a parallel in serial out (PISO) block. The input data Txdata_in is a 16 bit parallel data, which is encoded by a 16b/20b encoder to get a 20bit data as the output of the encoder. Further, the 20bit data is buffered in a FIFO. FIFO writes the data when the 'write_en' pin is high and reads the data when the 'read_en' pin is high. But when the FIFO is bypassed the output of encoder is directly fed to the PISO block. The Parallel in Serial out (PISO) block takes the 20 bit parallel data as the input. The PISO block converts the parallel data into a serial bit stream and this serial data is driven by a Tx diver. As shown in the above figure 3, the GTX receiver consists of Serial in Parallel out block (SIPO), an elastic buffer, a clock and data slider block (CDS) and a 20b/16b decoder.

The serial data bit stream transmitted by the transceiver is received by the GTX receiver. This received data is first fed to the SIPO block where the serial data is converted to a 20 bit parallel data. Later the parallel data is fed to elastic buffer which works similar to the FIFO explained in the transmitter block. But if the elastic buffer is bypassed the data out from SIPO is given to Clock and data slider block. Further when the data aligned by eliminating the latency and is received for an appropriate Rx_clk, it is fed to the 20b/16b decoder. The output of the decoder is a 16 bit data, which is given out by the payload generator.

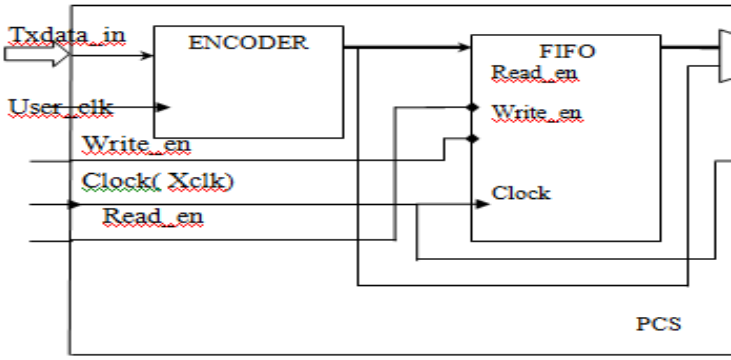

Fig 2 Simplified Architecture of the G

3.2 GTX Receiver

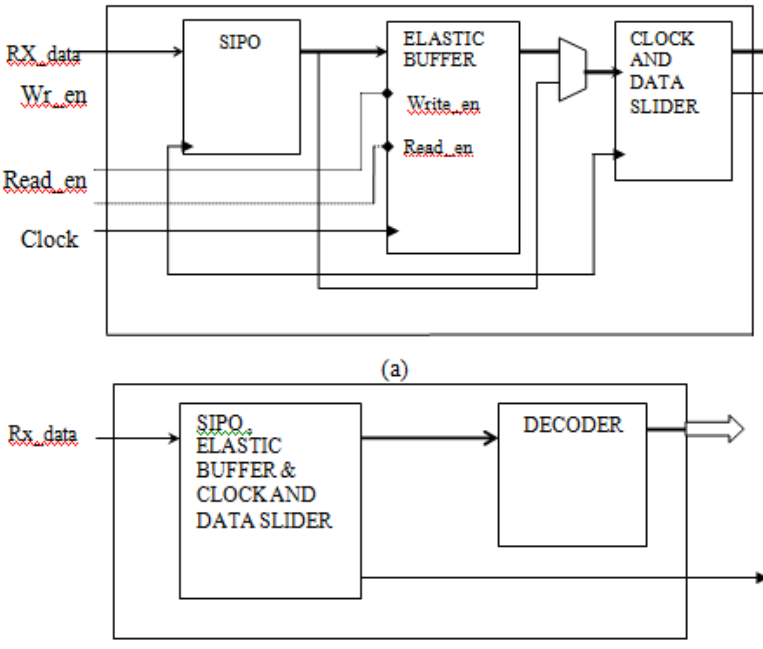

Figure 3: Simplified block diagram of GTX receiver. a) data processing blocks such as SIPO, elastic buffer and CDS and b) complete receiver block

3.3 Clock and Data Slider

The CDS consists of one Comma Detector and Data Alignment (CDDA) block, one Dynamic Clock Phase Shifting (DCPS) block, and one Changeable Delay Tuning (CDT) block. As shown in the figure 4. Bit-shift value of RX_data which is given by 'n' and the phase offset ΔP between RXRECCLK and the transmitted clock, satisfy the following equation:

$$\Delta P = n * 360° / N \text{ where } n = 0,1,2.....N-1.....(2)$$

Where, N is the internal data-path width of the GTX transceiver. In our implementation the value of 'N' is 20 bits. The Comma detector and data alignment (CDDA) block is responsible for finding commas, such as the K28.5 symbol (one of 8 b/10 b control characters) , in the parallel received data Data_in. In our implementation, we have included an undefined symbol 'U' as the required comma bit. Once it finds such a comma bit, it can determine the bit-shift value n.

By equation (2), we attain the phase offset ΔP, which might change after each power-up or reset. Based on the phase offset ΔP, the Dynamic clock phase shifting (DCPS) block provides four multi-phase clocks to the Changeable delay tuning (CDT) block. The four clocks, namely Rec_clk, Rec_clk90, Rec_clk180, and Rec_clk270, have the same frequency as RXRECCLK. Their output phases relative to RXRECCLK are ΔP, ΔP+90°, ΔP+180° and ΔP+270° respectively. Utilizing the four multi-phase clocks, the CDT block transfers the received parallel data Data_in from the RXRECCLK clock domain to the Rec_clk clock domain.

3.3.1 Comma Detector and Data Alignment

The simplified structure of CDDA is as shown in Figure 5. The bit-shift value can be extracted from Data_in. The output of CDDA consists of partial bits of Data_in and partial bits of Data_in's delay register, namely RXDATA_DLY. The bit_shift value decides how to combine the two data sources.

The CDDA block is responsible for finding commas as explained above. In the parallel received data Data_in, it locates a comma bit, later from this we find out the bit-shift value n. By using equation (2), we can obtain the phase offset ΔP , which might change after each power-up or reset. Four possible bit-shift values between the received data Data_in and TXDATA correspond to four possible phase offsets between RXRECCLK and the transmitted clock. The CDDA block also achieves the realignment for received Data_in. From now onwards we mention Data_in as RXDATA in further sections.

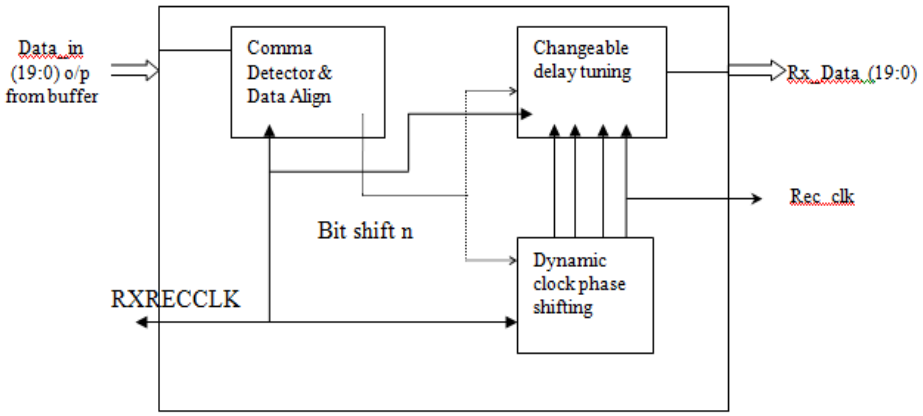

Figure 4 Block diagram of Clock and data slider block at the receiver end.

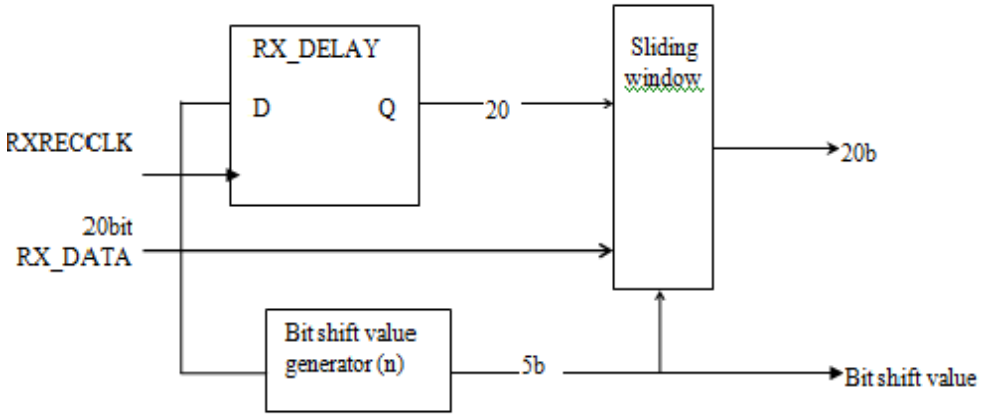

Figure.5 Structure of the Comma Detector and Data Alignment block

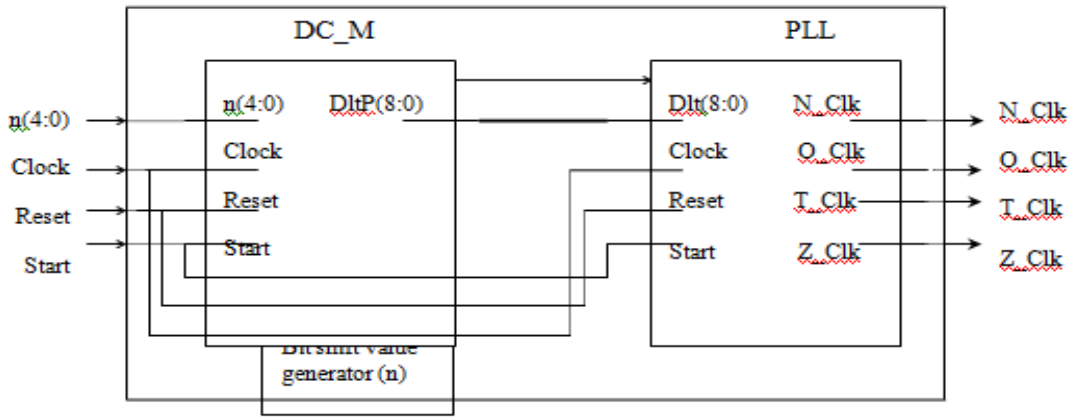

Figure. 6 Structure of the Dynamic Clock Phase Shifting block

3.3.2 Dynamic Clock Phase Shifting Block

A Digital Clock Manager (DCM), a Phase-Locked Loop (PLL), and a Phase Shift Control Unit are used to construct the DCPS block, as shown in Fig. 5. The DCM provides coarse and fine-grained phase shifting. When fine-grained phase shifting is activated, all the output clocks of the DCM are adjusted, so we can further extend the phase-shift range by using the CLK180 output of the DCM. The PLL provides coarse-grained phase shifting. The CLK0, CLK90, CLK180, and CLK270 output clocks of the PLL are each phase-shifted by a quarter of the input clock period relative to each other.

The Phase Shift Control Unit enables the DCM to execute the phase-shifting function. It operates based on the following two steps:

1) It first resets the DCM and waits for the DCM to relock. So, the phase-shift value P is 0 ns.
2) If ΔP <= 180° , the phase of the CLK0 output clock of the DCM is incremented from 0° to ΔP and it is chosen as the input clock of PLL. If ΔP > 180° , the phase of the CLK0 output clock of the DCM is incremented from 0 to (180°-ΔP), and the CLK180 output clock of the DCM is chosen as the input clock of PLL. The CLK_SEL signal, driven by the Phase-Shift Control Unit, performs a selection between the two input clocks of PLL

3.3.3 Changeable Delay Tuning Block

The CDT block consists of a Comma Comparator, a Delay Tuning Unit, and a asynchronous FIFO, as shown in Fig. 6. In Comma Comparator, the parallel input data realigned by the CDDA block is compared to a constant predefined comma character. The Delay Tuning unit consists of four parallel independent delay units and one multiplexer. Each delay unit is composed of two or three cascaded registers. The bit-shift value is used as the select signal for the multiplexer, which performs a selection among the four delay units. The asynchronous FIFO bridges the RXRECCLK clock domain and the Rec_clk clock domain

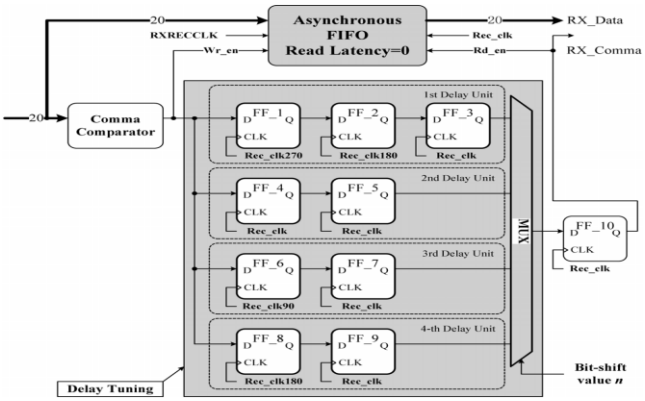

Figure 7: Structure of the Changeable delay tuning block

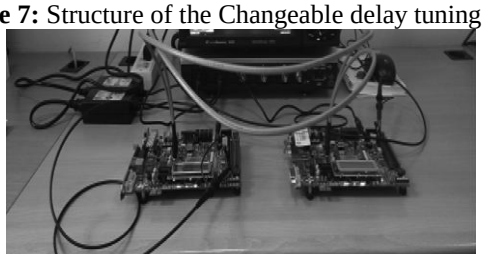

Figure 8: Test platform for our serial transceiver.

4. Experimental Results

The simulation result in figure 9 shows the various latencies in received data at different phase offsets. Here, we have generated four different phase clocks they are 0°, 90°, 180° and 270°. The received data at phase shifted clocks is bit shifted. As we can observe from the figure that at 270° the data is shifted by 3 bits, at 180° data is shifted by 2 bits similarly at 90° it is shifted by single bit but we receive unchanged data at 0° clock.The simulation result in figure 10 shows the complete fixed latency serial transceiver output de-serialization. After this it is fed to the CDS block. In CDS block the clock shifting and data alignment is achieved hence the output at the receiver after decoding is Rx_out = 1100110011001100 and appropriate Rec_clk as output.

5. Conclusion

Our solution can process all the possible clock phase offsets, so the transmitter and receiver need not use the same reference clock. Utilizing the external circuit to perform the clock phase-shift and data bit-shift, our solution reduces the dependence on the transceiver architecture.

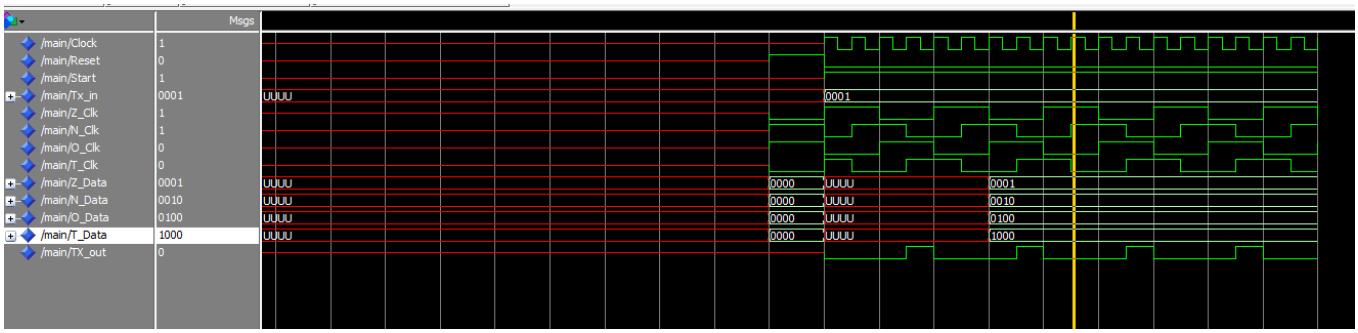

Figure 9: The above simulation result shows that phase offset results in data latency.

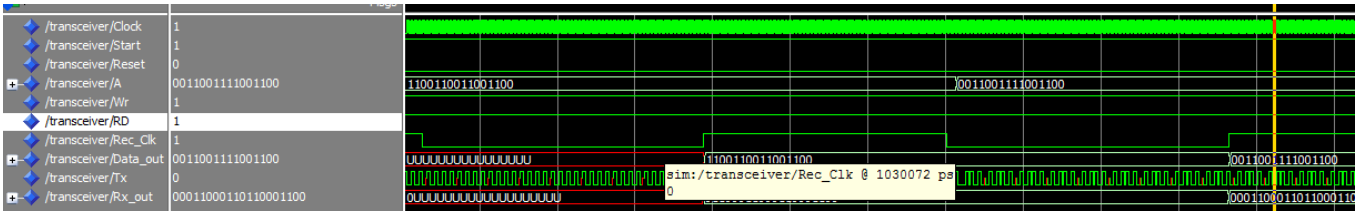

Figure 10: The above simulation result shows the complete transceiver out.

References

[1] SCAN25100 24576, 1228.8, and 614.4 Mbps CPRI SerDes with Auto RE Sync and Precision Delay Calibration Measurement, Texas Instruments, 2013.

[2] TLK2711 A 1.6 TO 2.7 GBPS TRANSCEIVER, Texas Instruments, 2012 [Online].

[3] B. G. Taylor, "TTC Distribution for LHC Detectors," *IEEE Trans. Nucl. Sci.*, vol. 45, no. 3, pp. 821–828, Jun. 1998.

[4] P. Moreira, A. Marchioro, and K. Kloukinas, "The GBT, a Proposed Architecture for Multi-Gb/s Data Transmission in High Energy Physics," in *Proc. Topical Workshop Electron. Particle Physics* Prague, Czech Republic, Sep. 3–7, 2007.

[5] White Rabbit [Online]. Available: http://ohwr.org/projects/whiterabbit/documents

[6] F. Lemke, D. Slogsnat, N. Burkhardt, and U. Bruening, "A Unified DAQ Interconnection network with precise time synchronization," *IEEE Trans. Nucl. Sci.*, vol. 57, no. 2, pp. 412–418, Apr. 2010.

[7] R. J. Aliaga, J. M. Monzo, M. Spaggiari, N. Ferrando, R. Gadea, and R. J. Colom, "PET system synchronization and timing resolution using high-speed data links," *IEEE Trans. Nucl. Sci.*, vol. 58, no. 4,pp. 1596–1605, Aug. 2011.

[8] H. Le Provost, Y. Moudden, and S. Anvar *et al.*, "A readout system-on chip for a cubic kilometer submarine neutrino telescope," *J. Inst.*, vol.6, no. 12, p. C12044, Dec. 2011

[9] P. P. M. Jansweijer and H. Z. Peek, Measure Propagation Delay Over a 1.25 Gbps Bidirectional Data Link, Tech. Rep. ETR 2010-01, 2010 [Online]. Available:http://www.nikhef.nl/pub/services/biblio/technical reports/ETR2010-01.pdf

[10] A. Aloisio, F. Cevenini, R. Giordano, and V. Izzo, "High-speed, fixed-latency serial links with FPGAs for synchronous transfers," *IEEE Trans. Nucl. Sci.*, vol. 56, no. 5, pp. 2864–2873, Oct. 2009.