Design of SerDes Transceiver with fixed and high throughput implementation on FPGA

Charles Rajesh Kumar.J[1], Vanchinathan.T[2], Kharthik.K[3]

[1.] Lecturer, Department of Electrical & Computer Engineering, College of Engineering, Effat University, Kingdom of Saudi Arabia. charlesece@yahoo.com
[2.] Senior Design Engineer, AXIIP Semiconductor Pvt Ltd, Chennai. tvanchi@gmail.com
[3.] Design Engineer, AXIIP Semiconductor Pvt Ltd, Chennai. kharthik.k90@gmail.com

Abstract- In modern communications systems serial interconnects form the critical backbone, so the choice of serializer/deserializer (SerDes) can have a big impact on system cost and performance. Serial interfaces are commonly used for chip-to-chip and board-to-board data transfers. As system bandwidth continues to increase into multi-gigabit range, parallel interfaces have been replaced by high-speed serial links or SerDes (Serializer/Deserializer). SerDes plays a crucial role in multi-gigabit serial data communication links. A SerDes or Serializer/Deserializer that can take wide-bit width, single ended single buses and compress them to a few typically one is used to transmit and receive data over serial link. SerDes enables the movement of large amount of data point-to-point while reducing the complexity, cost, power and board space usage have to implement wide parallel data buses but most of the high-speed Serializer/Deserializer (SerDes) chips do not keep the same latency after a reset, loss of clock or a power cycle. This implementation choice is often made because fixed-latency operations require dedicated circuitry. The link architecture based on the high-speed SerDeses will show how we made it constant and predictable and GTP transceiver is used to achieve the Fixed-Latency with 8b/10b encoding/decoding.
[Charles Rajesh Kumar. J, Vanchinathan. T, Kharthik. K. **Design of SerDes Transceiver with fixed and high throughput implementation on FPGA.** *Life Sci J* 2013;10(2):2849-2857] (ISSN:1097-8135). http://www.lifesciencesite.com. 394

Index Terms- Fixed-Latency, FPGA, Serial link, 8b/10b encode/decode.

1. INTRODUCTION

Serial interconnects form the critical backbone of modern communications 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) devices, 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. The latency, clocking and timing requirements and the need for additional buffering as well as logic. These issues can have a big impact on system cost, performance, and efficiency distributed systems for data acquisition and control applications are increasingly being based on networks of multi-Gigabit serial links both on copper and optical fibers. Most of the deployed high-speed Serializers/Deserializer (SerDes) chips do keep the same latency neither in terms of Unit Intervals nor in terms of parallel clock cycles after a reset, a loss of lock or a power cycle. Such an implementation choice is often made because the fixed-latency operation requires dedicated circuitry and it is usually not needed for most telecom and data-com applications. Timing synchronization and clock distribution applications would benefit from high-speed fixed-latency SerDeses. The IEEE 1588 [3] standard defines the Precision Time Protocol (PTP), which is designed for the synchronization of the clocks of a distributed system to a high precision clock. The PTP allows a system to achieve microsecond and sub-microsecond time synchronization between the clocks depending on their intrinsic precision and on the timing performance of the underlying network. A higher precision is achieved with PTP when the latency of the up-link and down-link are constant and equal.

Fixed-latency links also find application in data-acquisition systems for High Energy Physics experiments, specifically in the trigger sub-systems, where it is crucial to preserve the timing information associated with the transferred signals. Trigger subsystems of experiments at the Large Hadron Collider rely on the Timing Trigger and Control [14] system developed at CERN, which distributes all the signals with predictable latency and phase. The GigaBit Transceiver (GBT) project [5], under development at CERN, is aiming at developing a unique link for data transfer, trigger and clock distribution. The GBT must feature deterministic latency and phase matched clock recovery. Fixed- latency serial links would also be needed for the Trigger and Data Acquisition system (TDAQ) where the links used to transport data also carry a global clock and timing information.

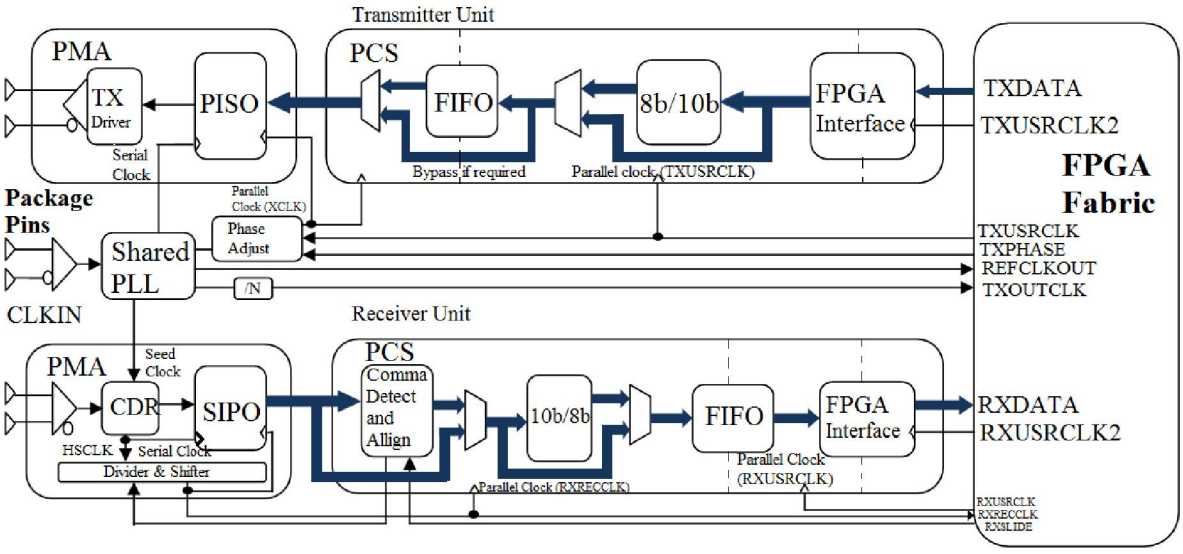

Fig.1. Simplified block diagram of the GTP transceiver.

2. GTP TRANSCEIVER

2.1 Introduction

The architecture presented in this project is based on the GTP transceiver [13] of FPGA family. This concept is also compatible with the GTX transceivers. Inside the FPGA, GTPs are available as configurable hard-macros (or "tiles"). Each tile includes a pair of transceivers, which share some basic components, like a Phase Locked Loop (PLL) and the reset logic. The device consists of a Physical Medium Attachment (PMA) sublayer, actually serializing and de-serializing the data, and a Physical Coding Sublayer (PCS), processing the data before serialization and after de-serialization. The shared PLL locks to a reference clock (CLKIN) and generates the high-speed serial clock for the transmitter, a seed clock for the Clock and Data Recovery (CDR) circuit and the parallel clock XCLK for the Parallel Input to Serial Output (PISO). The transmitter requires two input clocks TXUSRCLK and TXUSRCLK2, which, as far as it concerns the architecture we propose in this paper, are always driven with the same signal. Data is 8b to 10b encoded, if needed, and it is transferred to a First in First out (FIFO) buffer (also called elastic buffer), which allows safe data transfers between the TXUSRCLK domain and the XCLK domain. In some configuration XCLK and TXUSRCLK have the same frequency and a constant phase offset. The FIFO can be bypassed if the offset is sufficiently small and in fact the device offers a phase alignment circuit in order to minimize it. In the PMA, data in the XCLK domain is serialized by the PISO block, whose output is synchronous with the high-speed serial clock from the internal PLL. In addition to the already listed signals, the PLL also provides a parallel clock output (TXOUTCLK), which can be used to drive TXUSRCLK. In this implementation the TXUSRCLK input is clock synthesized in the fabric from the external reference clock. Analogously to the transmitter, the receiver requires two input clocks RXUSRCLK and RXUSRCLK2, which in this work are always driven with the same signal.

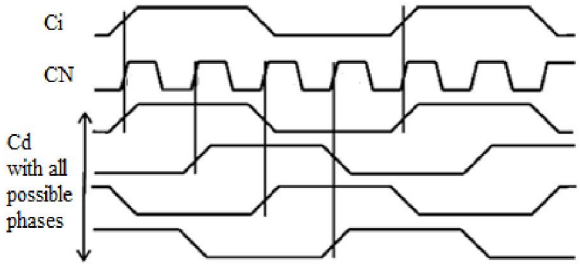

Fig. 2. Clock multiplication and subsequent division (case N=4)

On the GTP receiver unit, the serial stream from dedicated FPGA pins is received by the CDR, which extracts a clock (HSCLK) and uses it to sample the data. The extracted clock is divided to generate a parallel recovered clock (RXRECCLK) for the Serial In to Parallel Output (SIPO) and for the PCS. The recovered clock is also routed to the FPGA fabric to be used by the receiving logic. The Comma Detect and Align block following the SIPO can be programmed to search in the serial stream for a specific symbol (e.g., an 8b10b comma) and automatically use it to define a word boundary. Data is then 10b to 8b decoded, if needed and transferred to a FIFO (also called elastic buffer) in order to enter the RXUSRCLK domain. The GTP does not work with fixed latency in configurations based on its internal resources. We show that implementing in the

FPGA the adequate logic and suitably configuring the GTP permits to achieve fixed latency.

2.2 Latency Variation in Serial Links

This paragraph briefly presents sources of latency variations in a general SerDes architecture. It is not specifically related to the GTP, yet it is useful to better understand the discussion coming in the next two sections. Latency variations may come from both the serial and parallel sections of a SerDes device. We now focus on the serial section. Let us suppose to multiply the frequency of a clock signal Ci by a factor N and let CN be the resulting signal (Fig 2). Let us now divide the frequency of CN back by N, in order to obtain the same frequency. There are N possible phases for Cd each associated with an edge of CN (we are supposing the entire clock signals to be edge-aligned). If there is data traveling from the clock domain of Ci to the one of Cd, the phase variation of Cd leads also to a variation in latency of the data. If fixed-latency operation is required, a mechanism is needed to choose always the same phase for the divided signal and therefore also the same latency for the data.

In a serial link the frequency multiplication happens in the serializer and the division in the deserializer. In a serializer, the parallel transmit clock is multiplied to provide the high-speed clock to the serial side of the PISO. In the deserializer, the high-speed recovered clock from the CDR is divided to provide the low-speed recovered clock to the parallel side of the SIPO. We thus have a potential phase variation of the parallel recovered clock with respect to the transmit clock in terms of integer numbers of UIs. A phase variation of the recovered clock implies a latency variation of the data transferred on the link. It is important to keep in mind that this effect happens each time the frequency of a clock is multiplied and then divided, not only in SerDeses but also in PLLs. As far as it concerns the parallel section, latency variations may be induced by the presence of elastic buffers, which are usually implemented by means of FIFOs. Assuming an equal write and read rate from the buffer, after each reset the latency through the buffer is determined by the difference of the write and read pointers. This difference may vary depending on the behavior of the logic accessing the buffer and causes variations in terms of integer numbers of periods of the clocks for buffer read/write. So, a dedicated mechanism is needed to ensure that always the same number of words has been written in the buffer before they start being read (the receiver elastic buffer of the GTP implements this feature).

3. 8b10b ENCODING/DECODING

3.1 Introduction

The 8b/10b transmission code specifies the encoding of an 8-bit byte (256 unique data characters) and an additional 12 special characters into a 10-bit symbol; thus the 8b/10b designation. It also specifies the decoding of the 10-bit symbol back into an 8-bit word. The 8b/10b code is a well-established, industry standard and has been used in the physical layer (PHY) of a number of current and emerging standards, including Fibre Channel, Gigabit Ethernet, and Rapid IO. The delimiters. A subset of them, referred to as commas, are unique in that their bit pattern never occurs in a string of data symbols and hence can be used to determine symbol boundaries at the receiving end, in additional to this the code design allow the receiver to detect most transmission errors. Features of the 8b/10b Encoder reference design include: • Encoding of 8-bit bytes and an accompanying command bit KIN into 10-bit symbols.

• Optional control inputs: Clock Enable (CE), command input (KIN), Force Code (FORCE_CODE), Force Disparity (FORCE_DISP) and Disparity Input (DISP_IN).

• Encoder tracks running disparity to ensure that the disparity sequence of the transmitted symbols is valid. Optional status outputs: Running Disparity (DISP_OUT), command error (KERR), and new data (ND).

The principle of encoder is mapping transfer the input 8 bits into 10 bits data per the mapping principle, separate the 8-bits data into one group 3 bits data and another group 5 bits data, after encoding of the 8B/10B encoder, output one group 4 bits data and another group 6 bits data, to make a 10 bite data.

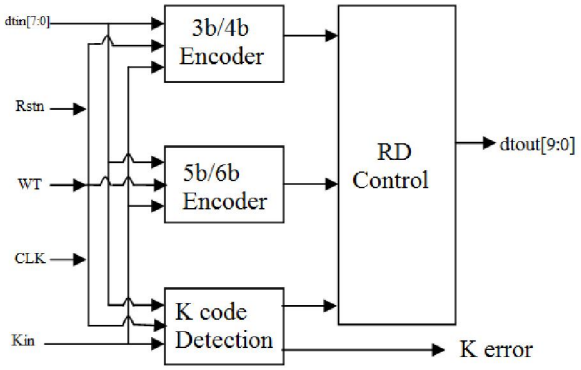

Fig.3 8b/10b code block diagram

3.2 8b/10b Encoder/Decoder Design

During the process of encoding, mark this unbalance with a parameter, Polarity deviation RD. This polarity deviation RD is unbalance with a parameter, polarity deviation RD composed with RD- and RD+. RD- means the times of number 1 is 2 times more than the times of number 0, RD+ means the times of number 0 is 2 times more than the times of number 1. The original 8B data was separated into two parts: front 3bits and back 5bits. For front 3bits, encoding it per 3B/4B, for the back 5bits, encoding it per 5B/6B.

Table 1. 5B/6B code

Input		RD=-1	RD=+1		Input		RD=-1	RD=+1
HGF		Fghj			HGF		Fghj	
D.x.0	000	1011	0100		k.x.0	000	1011	0100
D.x.1	001	1001			k.x.1	001	0110	1001
D.x.2	010	0101			k.x.2	010	1010	0101
D.x.3	011	1100	0011		k.x.3	011	1100	0011
D.x.4	100	1101	0010		k.x.4	100	1101	0010
D.x.5	101	1010			k.x.5	001	0101	1010
D.x.6	110	0110			k.x.6	001	1001	0110
D.x.P7	111	1110	0001					
D.x.A7	111	0111	1000		k.x.7	111	0111	1000

Table.2 3B/4B Code

Input		RD=-1	RD=+1		Input		RD=-1	RD=+1
EDCBA		abcdei			EDCBA		abcdei	
D.00	00000	100111	011000		D.16	10000	011011	100100
D.01	00001	011101	100010		D.17	10001	100011	
D.02	00010	101101	010010		D.18	10010	010011	
D.03	00011	11001			D.19	10011	110010	
D.04	00100	110101	001010		D.20	10100	001011	
D.05	00101	101001			D.21	10101	101010	
D.06	00110	011001			D.22	10110	011010	
D.07	00111	111000	000111		D.23	10111	111010	000101
D.08	01000	111001	000110		D.24	11000	110011	001100
D.09	01001	100101			D.25	11001	100110	
D.10	01010	010101			D.26	11010	010110	
D.11	01011	110100			D.27	11011	110110	001001
D.12	01000	001101			D.28	11100	001110	
D.13	01101	101100			D.29	11101	101110	010001
D.14	01110	011100			D.30	11110	011110	100001
D.15	01111	010111	101000		D.31	11111	101011	010100
					K.28	11100	001111	110000

As shown in the fig 3 the order of original 8B data is HGFEDCBA, HGF encoded into fghj, EDCBA encoded into abcdei. So, the transferred 10B is abcdeifghj. After the transformation, the data flow was output by 10B. The encoder assumes a negative RD- (-1) at start up. When a 8-bit data is encoding, the encoder will use the RD- column for encoding. If the 10-bit data been encoded is disparity neutral, the Running Otherwise, the Running Disparity will be changed and the RD+ column will be used instead. Similarly, if the current Running Disparity is positive (RD+) and a disparity neutral 10-bit data is encoded, the Running Disparity will still be RD+. Otherwise, it will be changed from RD+ to RD- and Otherwise, it will be changed from RD+ to RD- and not be changed and the RD- column will still be used.

As show in Table 1 & 2, the divided 8b is divided into 5b and 3b and the divided code is converted into 4b and 6b with the control signal based on its running disparity value. It will check the data flow, if the quantity of number 1 and number 0 is the same, the polarity of next 10B data will keep RD-, otherwise, it will turn to be RD+. If previous 10B polarity is RD+, and the quantity of number 1 and number 0 of previous 10B data keep same, the polarity of next 10B data will keep RD+, otherwise, it will turn to be RD-.

4. WORD ALIGNMENT MECHANISMS OF THE GTP TRANSCEIVER

In order to support custom alignment algorithms, the GTP also allows the alignment to be driven by the logic in the FPGA fabric. A dedicated signal (RXSLIDE), when asserted, causes the parallel word to be shifted by one more bit. There are two modes for the bit sliding to be achieved: the first one is realized in the PCS by shifting the parallel data (PCS mode) and the second one in the PMA with a shifting of the recovered clock phase combined with the logical shifting of the data (PMA mode). In PCS mode, the alignment is performed logically, with a barrel shifter, and there is no way to change the phase of the recovered clock. In order

to avoid latency variations, we have and clock from the CDR and a register re-captures the output from the shift-register on the parallel recovered clock (RXRECCLK) edge. Inside the "Clock Divider and Shifter", HSCLK is divided down by 5 and then fed in a 5-bit shift-register. The outputs of the shift-register are all separated by 1 high-speed clock cycle each (2 UIs). A modulo-10 counter receives the RXSLIDE signal and selects which output from the shift register to use as a to choose always the same phase of the recovered clock. It follows that the PMA mode is the best candidate for fixed latency implementations. The propose model of how the phase shift could be performed in PMA mode (Fig. 3). The CDR recovers a high-speed clock (HSCLK) running at half the bit-rate (UIs) and uses both edges to sample the incoming stream. A Double Data Rate (DDR) shift-register in the SIPO receives the serial data recovered clock. Each time a different phase from the multiplexer is selected, the recovered clock phase offset with respect to HSCLK changes. Inside the SIPO, this phase-offset determines which bit in the shift register is latched into the least significant bit (lsb) of the parallel register. Therefore, by selecting a different input in the multiplexer, the alignment of the parallel data with respect to the serial data is changed. With this model of the Clock divider and Shifter, it is possible to shift the recovered clock phase only with steps of 2 UIs with respect to the stream. The transceiver must perform a correct logical shift for any required number of bit slips. For this reason, a 1-bit barrel-shifter receives the parallel data from the SIPO and shifts the data by 1 bit, if needed. The select signal of the multiplexer inside the barrel-shifter is driven by the LSB of the modulo-10 counter. On the odd RXSLIDE assertions equals '1' and the barrel-shifter shifts the data, while on the even assertions the barrel shifter is deactivated and the data is shifted only by the shift of the recovered clock phase. This way the data is always correctly shifted but for each possible recovered clock phase there are two different alignments. of the parallel data, corresponding to an odd and an even number of bit shifts. There is 1 UI of latency difference between the two alignments. The alignment performed by shifting the recovered clock (PMA mode) can be driven only by logic external to the GTP, by means of the RXSLIDE signal. If we use the internal comma aligner and detector the alignment is achieved only by shifting the data (PCS mode). This is due to limitations of the device.

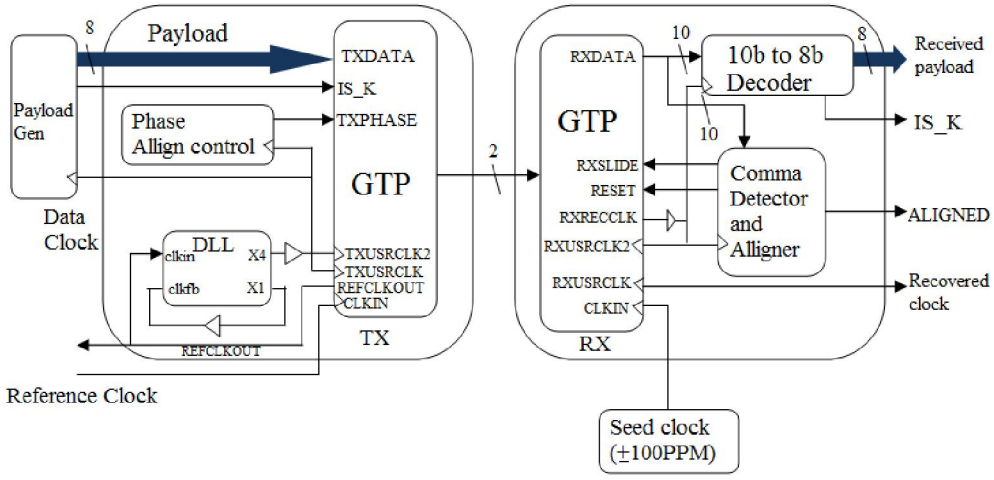

Fig.3 Conceptual block diagram of the shift architecture used in PMA mode.

5. FIXED-LATENCY DATA TRANSFERS WITH THE GTP

We will now present an implementation of a fixed-latency 8b10b-encoded serial link, running at 11.5 Gbps, based on the GTP SerDes (Fig.4). We have chosen a 8b10b coding because it is the most widespread coding for serial data (e.g., Gb-Ethernet, Fibre Channel). On the transmitter side, we exploit the internal 8b10b encoder and thus the GTP expects 8-bit input data words. The data generator can be programmed with a custom pattern of data and control symbols. If the IS_K input of the GTP is asserted, the incoming data is considered a control character and encoded accordingly. In this design we periodically send a control character on the link for the receiver to find the correct byte-boundary. The CLKIN input of the PLL of the GTP receives a reference clock. In order to provide the transmit clocks, a Delay Locked Loop (DLL) external to the GTP takes the reference clock (REFCLKOUT) from the GTP and multiplies its frequency by 4, thus providing TXUSRCLK. The DLL is there to ensure the same phase offset between TXUSRCLK and REFCLKOUT at each power-up. The reason for using the REFCLKOUT output of the GTP instead of the CLKIN signal directly is that CLKIN is connected to dedicated pins of the SerDes and it is not

available to the fabric. Since the parallel clock for the PISO (XCLK) is generated from the reference clock by multiplication and subsequent division, at each power-up its phase can be different. To work this issue around, we use the phase alignment circuit, which aligns the phase of XCLK to the one of TXUSRCLK. Since TXUSRCLK is generated with a deterministic phase from the reference clock, this leads to have XCLK, TXUSRCLK and the reference clock running with a deterministic fixed phase. In order to achieve this behaviour, at each power up or after a loss of lock, a controller in the fabric activates the phase align circuit by means of the TXPHASE signal. The procedure we follow for activating the alignment circuit is the one suggested in the GTP user's guide. Under this operating condition the transmit FIFO is unnecessary and therefore we bypassed it.

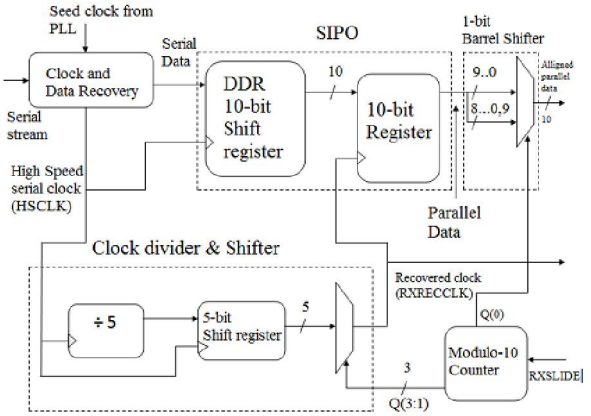

Fig 4. Implementation of a serial link with fixed latency and fixed clock phase, based on a GTP transceiver.

Table 3. Latency of the Internal Blocks of the Transmitter

	# of RXUSRCLK Cycles	Block Latency(ns)
Transmitter		
FPGA Interface	1	4
8b/10b Encoder	1	4
FIFO (bypassed)	1	4
Serial Section	2	8
Total Transmitter Latency	5	20

Table 4. Latency of the Internal Blocks of the Receiver

	# of RXUSRCLK Cycles	Block Latency (ns)
Receiver		
Serial Section	1.5	6
Comma Detector (bypassed)	3	12
FIFO	5	20
FPGA Interface	2	8
10b8b Decoder	1	4
Total Receiver Latency	12.5	50

On the receiver side, the CLKIN input of the GTP is driven by a clock generator with a frequency offset smaller than 100ppm with respect to the reference clock of the transmitter. The PLL uses this signal to generate a seed clock for the CDR to lock on the stream correctly recover the high-speed clock. The recovered clock (RXRECCLK) drives the receive clocks (RXUSRCLK and RXUSRCLK2). At each CDR lock-up, the recovered clock edge can be aligned with any bit inside the 10-bit symbol. In order to recover the clock always with the same phase with respect to the transmit clock, we had to deactivate the 8b10b comma detector and aligner internal to the GTP. We designed an external "Comma Detector and Aligner" in the fabric, which controls the bit sliding feature of the GTP in PMA mode. The comma detection in the fabric can be performed on the raw data (still encoded) from the transceiver. It is worth mentioning that the GTP gives access to a signal called RXBYTEIS ALIGNED which can be used to detect if a comma has been received on the current byte boundary, without accessing the raw data. However, this approach requires to pulse the RXSLIDE signal in a "trial-and-error" fashion until the correct alignment has been achieved. On the contrary, with an external comma detector, we exploit the property of comma symbols in such a way to find the correct alignment at the first received comma character. Also, being the decoder external to the GTP, our approach can be customized to work with any serial encoding. In fact, we have been able to implement

fixed-latency transfers between the GTP and some off-the-shelf devices, such as the Agilent G-Link chip-set [14] (supporting the conditional inversion master transition protocol) and the National Semiconductors DS92LV18 chip [15] (supporting a proprietary encoding). In order to establish a bi-unique relationship between the bit shifts performed and the corresponding recovered clock phase, it is necessary either to reject the CDR-locks leading to odd bit-shifts or to reject those leading to even bit- shifts. Our comma detector looks for commas, (e.g., the one included in the K28.5 symbol) and when it finds one it determines how many bit shifts, let it be n, are required in order to align the parallel data to the correct byte boundary. Once the comma is found, the aligner behaves according to the following algorithm: 1) If n is odd the logic resets the GTP and it waits for the CDR to re-lock. 2) If n is even the logic drives the RXSLIDE signal in such a way to perform an UI shift of the clock phase and thus a -bit logical shift of the data. The logic then asserts the ALIGNED output, flagging the correct alignment of the receiver.

Data from the GTP is decoded by means of a dedicated logic and payload bits are provided as outputs on the FPGA board for test purposes. The 10b to 8b decoder also provides a flag indicating if the received word is a control character (IS_K). Its architecture is similar to the one presented in [16]. The latencies of the transmitter and the receiver, estimated by means of the user guide, are given in Table 1 and in Table 2. The concept proven with our design can be used to achieve fixed latency on duplex-links. In the case of independent forward and return channels, one just simply needs to use two instances of the architecture we proposed. In the case of a synchronous re-transmission on the return channel, some care is needed. The clock recovered by the receiver cannot be directly used as a reference clock for the re-transmission. In fact, the internal PLL is shared among the transmitters and receivers in the same tile and at the receiver tile it is already locked to the seed clock. Moreover, the recovered clock might also require to be filtered in order to match the jitter specifications for the GTP reference clock. An example of such a work can be found in [18]. We notice that the value of the latency of our serializer and deserializer architecture is known and fixed. In some applications it might be necessary to measure in the field the latency of the link including the cables this could be performed in a duplex link based on our architecture.

6. FIXED-LATENCY ACHIEVMENT

On the transmitter, the parallel clock driving the PISO must have a predictable phase relationship with respect to the external parallel data clock. In many transmitters this might be supported by using features for serial channel bonding. In fact, channel bonding is used for multi-lane serial buses such as PCI Express, RapidIO and Infiniband, which require the same latency through each bonded data-path.

On the receiver, the recovered clock must have a predictable phase relationship with respect to the byte boundary in the incoming serial stream. The receiver should offer a direct method to determine this phase offset in order to determine it indirectly. The phase offset determination can be done externally if the device is able to output un-decoded and un-aligned parallel words. Since, at each power-up of the receiver, the phase offset changes, the designer might add an external logic, which only checks the offset (by finding the data alignment) and resets the receiver if it is not the desired one (*roulette* approach). When the desired hase offset occurs, the logic does not reset the device and the lock is achieved with the same latency as all the other successful locks. The *roulette* approach does not strictly require to perform any alignment of the data, since the receiver can simply reject the locks requiring to shift the parallel data and accept only the locks leading to have data already correctly aligned. The receive logic is then simpler, since it does not require a comma detector or a word aligner, but only a decoder to check that the received data is valid. An obvious drawback of such an approach is the increase in the average lock time, which is proportional to the number of bits in the parallel symbol. Therefore, there is here a trade-off between the average lock time and the simplicity of the receive logic.

7. TEST RESULTS

In order to test our architecture, we deployed two off-the-shelf boards (Xilinx ML-505 [19]). The boards route the serial I/O pins of one of the GTPs on the FPGA to Sub-Miniature version A (SMA) connectors. We connected the Tx and Rx GTPs with a pair of 6 ns, 50 impedance coaxial cables. Transmitted and received payloads were monitored by means of an oscilloscope. We checked that the transmission latency and the phase of the recovered clock remained always the same during transfers and between subsequent power-ups of the system. We performed a test, resetting the transmitter and the receiver every 3 seconds (i.e., simulating a power-cycle) and holding all the acquired waveforms on the oscilloscope screen, in order to record latency variations. The standard deviation of the latency is ~40 ps and it includes the contributions from all the subsequent power-ups of the system. The precision of the synchronization between the transmitted and recovered clocks is ~40 ps. We note that the latency in terms of integer number of parallel clock cycles and UIs is fixed at each power up. The distribution we observe is only due to the jitter of the received data edge with respect to the transmitted one. We measured the latency

of the transmitter and of the receiver with respect to the serial stream. In order to easily find the serial bit corresponding to a pulse on the parallel word, we sent a word sequence. We measured the latency of the transmitter (receiver) by probing the bit #0 of the payload on a test point and the serial output (input) of the transceiver. In our setup, the transmitter latency was 21ns. However, this measurement underestimates the latency by the propagation delay of the FPGA Input/ Output Block (IOB) and internal routing. This delay is estimated to be at most 7ns by the Xilinx timing analysis tool. This explains the difference between the measured and the estimated latency of Table 3 and 4. In our setup, the oscilloscope receives the stream at the same time of the GTP receiver but it receives the parallel payload bit with a delay equal to the sum of the propagation through the FPGA IOB (< 8ns) and the connection cable (5 ns). The measured latency after the subtraction of the cable delay is 55ns, but it still overestimates the actual latency by a quantity equal to the IOB delay. The rms jitter on the recovered clock has been measured to be of the order of 20 ps. We measured the jitter with respect to the reference clock at the transmitter, therefore the measured quantity can be taken as an estimate of the precision of the synchronization between the reference and recovered clock. For more information about the jitter performances of our architecture see [20].

8. CONCLUSION

High-speed SerDes chips are typically designed for variable latency transfers. In fact, the fixed-latency operation needs special design care and it is often not needed in most of telecom and data-com applications due to its latency variation. However, protocols for timing synchronization and clock distribution applications, which sometime are present in HEP experiments, would benefit from fixed-latency serial links. By suitably configuring two GTP transceivers embedded in Xilinx FPGAs and adding to them a control logic in the FPGA fabric, we implemented fixed-latency operation. Our link transfers data with fixed latency and recovers the clock from the serial stream with a predictable phase, even after a reset or a power-cycle of the system. As an example of implementation, we designed a 11.5 Gbps serial link based on 8b10b encoding. Our architecture is independent of the data encoding and can be customized to support any. We highlighted the features of the GTP architecture which are crucial for the fixed latency operation and we provided some guidelines to allow the reader to use our results also with other off-the-shelf high-speed SerDeses.

ACKNOWLEDGMENT

The authors are grateful to Charles Rajesh Kumar.J from Effat University, Vanchinathan.T and Kharthik.K from AXIIP semiconductor pvt ltd for their help in configuring the GTP transceiver.

REFERENCES

[1] TLK2711A—1.6 TO 2.7 GBPS TRANSCEIVER Texas Instruments, 2007 [Online]. Available: http://focus.ti.com/lit/ds/symlink/tlk2711a.pdf

[2] SCAN25100, National Semiconductors, 2008 [Online]. Available: http://www.national.com/ds/SC/SCAN25100.pdf

[3] *Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE Standard 1588, 2008, p. 34.

[4] B. G. Taylor, Timing, Trigger and Control (TTC) Systems for LHC Detectors, CERN/EP [Online]. Available: http://www.cern.ch/TTC/intro.html

[5] 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 Electronics for Particle Physics*, Prague, Czech Republic, Sep. 3–7, 2007.

[6] "SuperB a high-luminosity heavy flavour factory conceptual design report," The SuperB Collaboration, 2007 [Online]. Available: http://www.pi.infn.it/SuperB/files/active/4/paper.pdf

[7] 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.

[8] P. Moreira, J. Serrano, T. Wlostowski, P. Loschmidt, and G. Gaderer, in *Proc. Int. Symp. Precision Clock Synchronization for Measurement, Control and Communication*, Brescia, Italy, Oct. 12–16, 2009, pp. 1–5.

[9] R. Aliaga *et al.*, "PET system synchronization and timing resolution using high speed data links," in *Proc. Real Time Conf. Rec.*, Lisbon, Portugal, May 24–28, 2010.

[10] E. G. Cota, M. Lipinski, T. Włostowski, E. van der Bij, and J. Serrano, White Rabbit Specification: Draft for Comments, 2010 [Online]. Available: http://www.ohwr.org/attachments/306/WhiteRabbitSpec.pdf

[11] 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.

[12] A. Aloisio, F. Cevenini, R. Giordano, and V. Izzo, "Emulating the GLink chip set with FPGA serial transceivers in the ATLAS level-1 muon trigger," *IEEE Trans. Nucl. Sci.*, vol. 57, no. 2, pp. 467–471, Apr. 2010.

[13] Virtex-5 FPGA RocketIO GTP Transceiver User Guide, Xilinx, UG196, v1.7, 2008 [Online]. Available: http://www.xilinx.com/support/documentation/user_guides/ug196.pdf

[14] Agilent HDMP 1032-1034 Transmitter-Receiver Chip-Set Datasheet, 2001 [Online]. Available: http://www.physics.ohiostate.edu/~cms/cfeb/datasheets/hdmp1032.pdf

[15] DS92LV18 18-Bit Bus LVDS Serializer/Deserializer—15–66 MHz, National Semiconductor, 2006 [Online]. Available: http://www.national.com/ds/DS/DS92LV18.pdf

[16] A. X. Widmer and P. A. Franaszek, "A DC-balanced, partitioned-block, 8b/10b transmission code," *IBM J. Res. Develop.*, vol. 27, no. 5, p. 440, 1983.

[17] A. X.Widmer and P. A. Franaszek, "Byte Oriented DC Balanced (0,4) 8B/10B Partitioned Block Transmission Code," U.S. Patent 4 486 739, Dec. 4, 1984.

[18] P. P. M. Jansweijer and H. Z. Peek, Measuring Propagation Delay Over a 1.25 Gbps Bidirectional Data Link 2010 [Online]. Available: http://www.nikhef.nl/pub/services/biblio/technicalreports/ETR2010-01.pdf

[19] ML505/ML506/ML507 Evaluation Platform User Guide, UG347, v3.0.1, Xilinx , 2008 [Online]. Available: http://www.xilinx.com/support/documentation/boards_and_kits/ug347.pdf

[20] A. Aloisio, F. Cevenini, R. Giordano, and V. Izzo, "Characterizing jitter performance of multi gigabit FPGA-embedded serial transceivers, "*IEEE Trans. Nucl. Sci.*, vol. 57, no. 2, pp. 451–455, Apr.2010.

6/21/2013