BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 73(6), 2025, Article number: e155044
DOI: 10.24425/bpasts.2025.155044

Fast data transmission with FPGAs for implementation in ALICE FIT detector in CERN

Radosław FEIGLEWICZ*, Andrzej KOS, Paweł RUSSEK, and Sebastian KORYCIAK

AGH University of Krakow, al. Mickiewicza 30, 30-059 Krakow, Poland

Abstract. In high-energy physics experiments, massive amounts of data need to be processed. The limitations of peripherals in field-programmable gate arrays (FPGAs), such as analog-to-digital converters (ADCs), make it impractical to design setups based solely on a single FPGA chip, creating a need for high-speed, low-latency methods for inter-chip data transmission, where multi-gigabit transceivers (MGTs) are essential. This paper aims to assess the feasibility of using MGTs in the readout electronics of the fast interaction trigger (FIT) detector within the A Large Ion Collider Experiment (ALICE) at CERN. It examines two established protocols, i.e. Aurora 8b/10b and Aurora 64b/66b, focusing on their throughput and latency. Additionally, it presents a custom protocol based on 64b/66b encoding, specifically designed to reduce latency, and compares its performance with standard solutions. Transmission tests were conducted between two Xilinx boards, ZCU102 and ZCU106, using both the Aurora protocols and the custom protocol. Furthermore, transmission quality was evaluated based on the type and length of the transmission medium as well as on the lengths of the pseudo-random binary sequence (PRBS).

Keywords: Aurora; high-speed serial transmission; low-latency; FPGA; custom protocol.

1. INTRODUCTION

Numerous studies focusing on high-speed data transmission have been conducted for experiments at CERN. Three main research areas for high-speed serial transmission can be identified:

- Radiation-hardened: various methods for preventing and correcting potential errors are developed to enable data transmission in environments that are heavily exposed to radiation [1–3].
- Fixed latency: many measurement and control systems require transmission where it is critical for the latency to remain constant, regardless of external factors such as temperature and differences in phase between clock domains [4–6].
- High throughput and low latency: in many applications, large volumes of data must be transmitted within a short time frame so that decision-making modules can respond in a timely manner [7–9].

This article focuses on achieving the required throughput along with sufficiently low latency to enable the transmission of signals to other, slower radiation detectors for their activation. While fixed latency is not required, transmission delay must not exceed a maximum established value.

The fast interaction trigger is one of the key subdetectors in the ALICE experiment. FIT is responsible for recording and monitoring beam collision outcomes as well as for background radiation. Primarily, however, FIT is tasked with triggering the readout for ALICE's slower subdetectors, serving as a so-called "trigger detector" [10]. Figure 1 presents the current architecture of the system responsible for generating the trigger signal. Data from two scintillator arrays, FT0-A and FT0-C, are used to generate this signal. Analog signals from 208 Cherenkov photomultipliers are transmitted via coaxial cables to analog-to-digital converters located on boards called processing modules (PM). In the PM, intermediate calculations are performed in the FPGA, after which the data are transmitted to the trigger & controller module (TCM). High-definition multimedia interface (HDMI) cables are used for data transmission from PM to TCM. Each cable contains four differential pairs, with data transmitted at a maximum frequency of 320 MHz per pair.

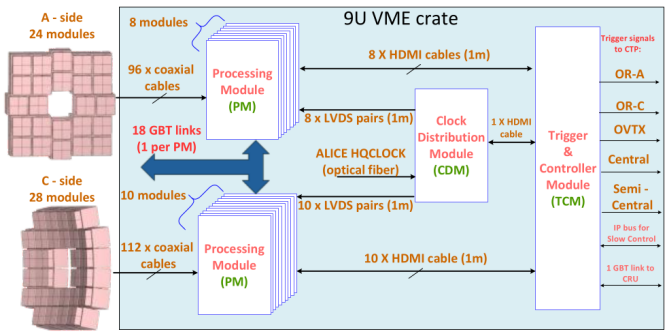

Fig. 1. Architecture of FIT trigger electronics [10]. Analog signals from the scintillators are sent to the processing modules (PM), where they are digitized and processed. The processed data are then transmitted to the crigger & Controller module (TCM), where the trigger decision is generated

This work aims to develop solutions that could be used to increase bandwidth and reduce transmission latency between the PM modules and the TCM module. It is assumed that in a future

*e-mail: feiglewicz@agh.edu.pl

Manuscript submitted 2024-11-27, revised 2025-04-11, initially accepted for publication 2025-08-11, published in November 2025.

upgrade of the FIT design, FPGAs with high-performance GTH or better multigigabit transceivers could be utilized. Therefore, it was necessary to examine their potential to replace the existing HDMI and input/output serializer/deserializer (IOSERDES)-based transmission with solutions based on MGT. Given that the new electronics for the FIT detector will be enhanced, the transmission must exhibit the following characteristics:

- the delay in transmitting eight 32-bit values cannot exceed 200 ns
- the distance between the PM and TCM can be of maximum 5 meters
- unidirectional transmission is required.

2. SERIAL TRANSMISSION

Modern inter-chip transmission methods are dominated by serial communication. This is due to the fact that, unlike parallel transmission, serial communication is characterized by lower energy consumption, the ability to operate over greater distances, and easier design implementation on printed circuit boards (PCBs). Unlike parallel buses, where it is essential to ensure that all signal lines are of the same length, serial communication simplifies design considerations. To achieve gigabit speeds in serial transmission, both the transmitter and receiver must be impedance matched. The fundamental topology for the physical layer of serial transmission is point-to-point, as seen in standards such as the peripheral component interconnect express (PCI Express) [11].

2.1. Clock distribution methods for serial transmission

Even minor differences in the clock frequency of the transmitter and receiver can result in data not being correctly received at the receiver. Factors such as temperature and technology process variations can cause clock generation circuits to deviate from their nominal frequencies by even several percent [12]. Therefore, the clock signal used for transmission must be explicitly delivered to the receiver (synchronous transmission), as seen in system-synchronous and source-synchronous topologies, or recovered from the data signal, as in self-synchronous topology. The self-synchronous method involves using a phase-locked loop (PLL) circuit to retrieve the transmission frequency, as shown in Fig. 2. The transmitted data must contain multiple transitions from 0 to 1 and from 1 to 0 to prevent the PLL from losing synchronization. Both system-synchronous and source-synchronous methods are used for communication between devices in close proximity, such as between a standalone ADC and an FPGA, as seen in the JESD204 standard [13, 14]. For high-speed communication between separate boards, the self-synchronous method proves essential. Therefore, multi-gigabit transceivers (MGTs) utilize this method [15, 16].

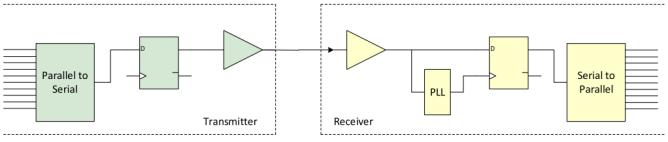

Fig. 2. Self-synchronous timing model [17]

2.2. MGT architecture

The primary difference between MGT and I/OSERDES circuits is their transmission speed. The Zynq UltraScale+ system on chip (SoC) modules used in this article are equipped with GTH transceivers, which support transmission speeds of up to 16.375 Gb/s. Xilinx also offers GTY and GTM transceivers, capable of achieving speeds of 32.75 Gb/s and 112 Gb/s, respectively [18]. To reach these high speeds, current mode logic (CML) is used instead of low-voltage differential signaling (LVDS), as in I/OSERDES. Figure 3 shows the transmitter and receiver diagram of the GTH circuit. Parallel data from the FPGA can be encoded in 8b/10b format or, using a gearbox circuit, in 64b/66b format. In the physical medium attachment (PMA) section, polarity inversion is available. Finally, the bit vector is converted to a serial format by a parallel-in serial-out (PISO) circuit and transmitted using current mode logic (CML). The receiver, using a serial-in parallel-out (SIPO) circuit, converts the serial bit stream into a parallel format. For proper operation, SIPO requires clock recovery, which is achieved through a clock data recovery (CDR) circuit. The receiver can then decode the data from the 8b/10b or 64b/66b format, after which the parallel data are passed to the programmable logic. By integrating certain operations, such as encoding and decoding, within the MGT, transmission can be achieved with lower latency than in solutions where these operations are performed in programmable logic. To improve transmission quality, the transmitter can apply pre-emphasis or post-emphasis, while the receiver can use an equalizer for this purpose.

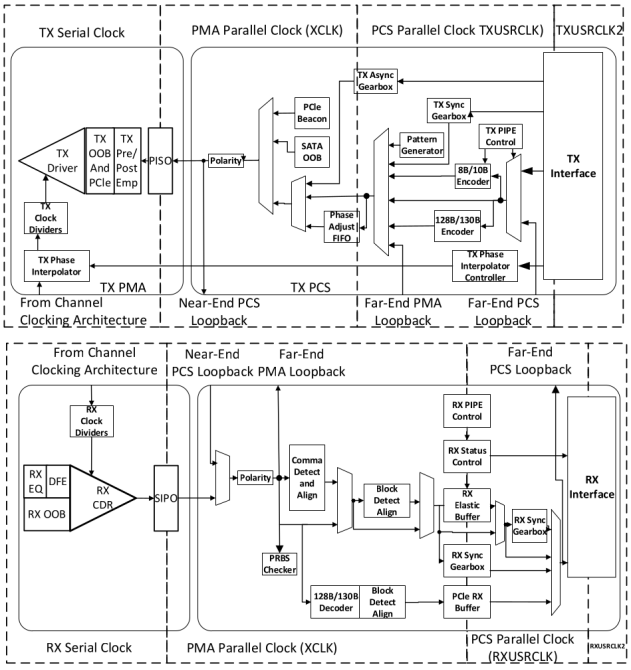

Fig. 3. Diagram of the GTH transmitter and receiver circuit [19]

2.3. Data encoding

In the context of high-speed data transmission, encoding primarily serves to prepare data for reliable transmission and proper decoding by the receiver. Modern transmission systems typically

use three types of encoding: 8b/10b, 64b/66b and 128b/130b. In 8b/10b encoding each 8-bit segment of data is converted into a 10-bit code, where the lower 5 bits are mapped to a 6-bit code (5b/6b) and the upper 3 bits are converted to a 4-bit code (3b/4b). Combining these two groups produces a 10-bit code, which is then transmitted serially by the serializer. Each 10-bit code has an alternate version to balance the average number of zeros and ones sent through the transmission channel. In addition to encoded data, control symbols are also transmitted to indicate the start and end of a frame, align multi-channel transmissions, or signal an idle state in the transmission channel. The undeniable advantages of 8b/10b encoding include excellent DC balance and the guarantee that no more than five consecutive zeros or ones will appear in the data stream, preventing the receiver's CDR circuit from desynchronizing. However, this comes at the cost of relatively low transmission efficiency, which is capped at 80 percent [20]. As the demand for higher data throughput grows, 8b/10b encoding is gradually being replaced by more efficient methods, such as 64b/66b encoding.

64b/66b encoding differs significantly from 8b/10b encoding. In 64b/66b, each 64-bit data word is accompanied by an additional 2-bit header. This header can either be "10" or "01". If the receiver detects a header value of "11" or "00", it indicates that the receiver is out of sync and requires resynchronization. Resynchronization is achieved through a process known as bitslip, which involves shifting the bit reception window by one bit at a time until the receiver identifies the correct synchronization bits (01 or 10), enabling proper data frame interpretation [21].

Figure 4 illustrates the types of frames in 64b/66b encoding.

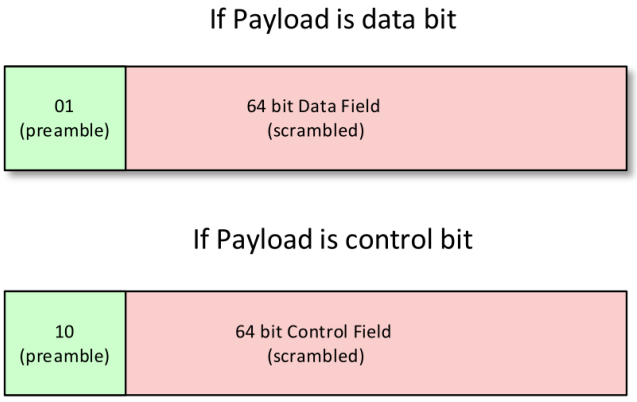

Fig. 4. Types of data frames in 64b/66b encoding. Preamble 01 indicates frames carrying only user data, while 10 marks frames containing control information or a mix of control and user data

If the header is "01", all 64 bits carry data. In the case of a header value of "10", the first byte indicates which bytes will contain control symbols, such as start, idle, or error, and which should be interpreted as data. The 64b/66b encoding does not inherently provide a balance between the number of transmitted zeros and ones, nor does it protect against long runs of the same bit. To address this, bit scrambling must be applied. The greatest advantage of 64b/66b encoding is its transmission efficiency, which can reach up to 97 percent.

2.4. Scrambling

In wireless communication, scrambling is widely used for pseudo-randomization of data and for encryption. This randomization results in a smooth signal spectrum without any prominent frequencies. Furthermore, this energy dispersion reduces the issue of inter-carrier interference (ICI). In the context of wired transmission, scrambling aims to eliminate long sequences of identical bit values and to balance the number of zeros and ones present in the transmission [22]. An example implementation of a scrambler using the polynomial $p(x) = x^7 + x^4 + 1$ is shown in Fig. 5. The operation of the scrambler is quite simple: the data intended for transmission undergo an XOR operation with the current value of a linear feedback shift register (LFSR). As a result, we obtain a pseudorandom sequence that can be descrambled at the receiver. There are two main types of scramblers: synchronous and self-synchronous ones. A detailed discussion of these types can be found in [23].

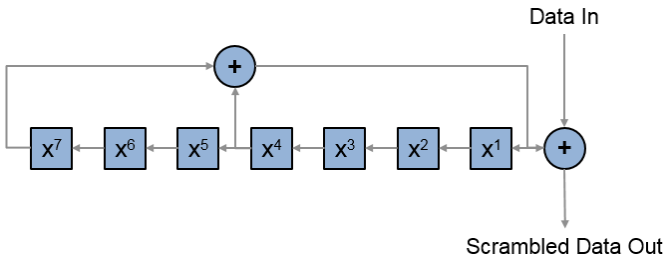

Fig. 5. Additive (self-synchronizing) scrambler based on the polynomial $1 + x^4 + x^7$. The input data are XORed with the feedback from the shift register taps (positions 4 and 7), producing scrambled output. The structure ensures that long runs of identical bits are broken, enhancing data integrity and synchronization [24]. In highspeed serial protocols such as Aurora 64b/66b, as well as in our custom protocol, a longer scrambling polynomial such as $1 + x^{39} + x^{58}$ is used to achieve better randomness and signal quality

2.5. Aurora 8b/10b

To facilitate the design of systems based on MGTs, Xilinx provides ready-to-use solutions known as Aurora. There are two commonly used versions of Aurora: Aurora 8b/10b and Aurora 64b/66b, catering to different data encoding requirements. Aurora 8b/10b is a scalable, lightweight, link-layer protocol for high-speed serial communication [25]. By utilizing the Advanced eXtensible Interface (AXI-4) stream based on the valid-ready protocol for user data transmission, Aurora can be easily integrated into various FPGA-based projects. Additionally, Aurora incorporates an additive scrambler, CRC code generation, and supports multi-channel transmission. The entire configuration can be performed using the Aurora 8b/10b wizard in Vivado software. In addition to the configuration, it is only necessary to connect the clock signal and the signal responsible for resetting the module. To verify whether the Aurora 8b/10b protocol meets the previously established requirements, simulations were conducted using Vivado software. The protocol was tested in loopback mode, meaning that the transmitter's output was connected to the receiver's input.

A counter written in SystemVerilog was used for data generation. Assuming a theoretical cable delay of 5 ns/m and that 8 values of 32 bits each need to be transmitted, the total delay is calculated as follows:

$$\begin{aligned}\text{Aurora 8b/10b delay} &\approx 29 \cdot \frac{1}{165\ \text{MHz}} \\ &\quad + 8 \cdot \frac{1}{165\ \text{Mhz}} + 5\ \text{m} \cdot 5\ \frac{\text{ns}}{\text{m}} \\ &\approx 250\ \text{ns}. \end{aligned} \tag{1}$$

Processing a 32-bit data vector through both the transmitter and receiver takes 29 clock cycles.

Unfortunately, the delay calculated significantly exceeds the specified latency requirement of a maximum of 200 ns, which disqualifies the above protocol from being used in the FIT project.

2.6. Aurora 64b/66b

To meet modern demands for high data transmission speeds, Xilinx released a version of the Aurora protocol based on 64b/66b encoding. This approach improves transmission efficiency, allowing for more user data to be transmitted at the same sender speed. By utilizing the Aurora 64b/66b protocol, the maximum transmission speed of the tested GTH module, 16.375 Gbit/s, can be fully utilized, as compared to Aurora 8b/10b, which is limited to a maximum speed of 6.6 Gbit/s. Assuming a theoretical cable delay of 5 ns/m and that 8 values of 32 bits each need to be transmitted (4 transactions, each 64 bit), the total delay is calculated as follows:

$$\begin{aligned}\text{Aurora 64b/66b delay} &\approx 45 \cdot \frac{1}{255.86\ \text{Mhz}} \\ &\quad + 4 \cdot \frac{1}{255.86\ \text{Mhz}} + 5\ \text{m} \cdot 5\ \frac{\text{ns}}{\text{m}} \\ &\approx 217\ \text{ns}. \end{aligned} \tag{2}$$

Processing a 64-bit data vector through both the transmitter and receiver takes 45 clock cycles.

Transmission latency has been reduced as compared to the Aurora 8b/10b protocol; however, it still exceeds the established maximum threshold.

3. CUSTOM PROTOCOL

Since the existing transmission protocols do not meet the requirements specified, the authors of this article undertook to develop a new protocol aimed at reducing latency as compared to currently available protocols. This protocol is based on 64b/66b encoding but includes certain modifications. If the header value is "01", 64 bits of user data are transmitted. When no data are pending at the transmitter input, synchronization data are sent instead, with a header value of "10". Headers "00" and "11" indicate, as in the Aurora protocol, that the receiver is desynchronized or that transmission errors are present.

The custom protocol design prioritized minimizing latency, which led to the decision to forego multi-channel transmission support, as this would require additional first-in, first-out (FIFO) buffers to align data across lanes [26]. Additionally, the scrambling and descrambling modules were optimized to operate on a 64-bit vector within a single clock cycle, further reducing transmission latency.

3.1. Custom protocol architecture

Figure 6 illustrates the architecture of the custom protocol. User data are transferred through the TX_USER_INTERFACE, which operates on a valid/ready handshaking mechanism. The data are then passed to the TX_scrambler module, where they undergo scrambling via an additive scrambler. Finally, the data are sent to the transceiver, where they are converted from parallel to serial format and transmitted. When there is no user data to transmit, the transmission is maintained by sending an 8-bit counter value, repeated eight times and then scrambled. Such a frame is marked with a header set to "10".

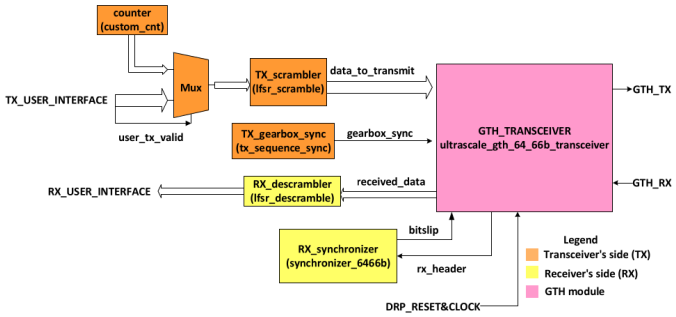

Fig. 6. Architecture of custom protocol. All blocks, except for GTH_TRANSCEIVER, were written in SystemVerilog and then integrated using Vivado Block Design

On the receiver side, the GTH_TRANSCEIVER module converts the serial bitstream into a parallel format. The data then undergo a descrambling process, and finally, the descrambled data are output on the RX_USER_INTERFACE. The user_rx_valid signal indicates whether the data on the interface are valid; this signal is high only when the receiver is synchronized and the received data have a header equal to "01". The RX_synchronizer module is responsible for synchronizing the receiver, ensuring that the receiver accurately identifies the start of each frame encoded in the 64b/66b format.

To prevent metastability issues, all transmitter-side modules (indicated in orange) are clocked using the tx_usr_clk2 signal, which is generated within the GTH_TRANSCEIVER module. The frequency of this clock signal is set to the transmission rate divided by 64, allowing only 64 bits to be sent within one clock cycle. To facilitate data transmission in the 64b/66b format, a gearbox unit is employed. This unit enables the addition of data headers and buffering of pending bits. The operational principle of the gearbox is illustrated in Fig. 7. Every 32 clock cycles of tx_usr_clk2, the user_tx_ready signal goes low for one cycle. This is due to the accumulation of 64 bits in the gearbox. As a result, this protocol behavior causes the transmission delay to be variable rather than constant.

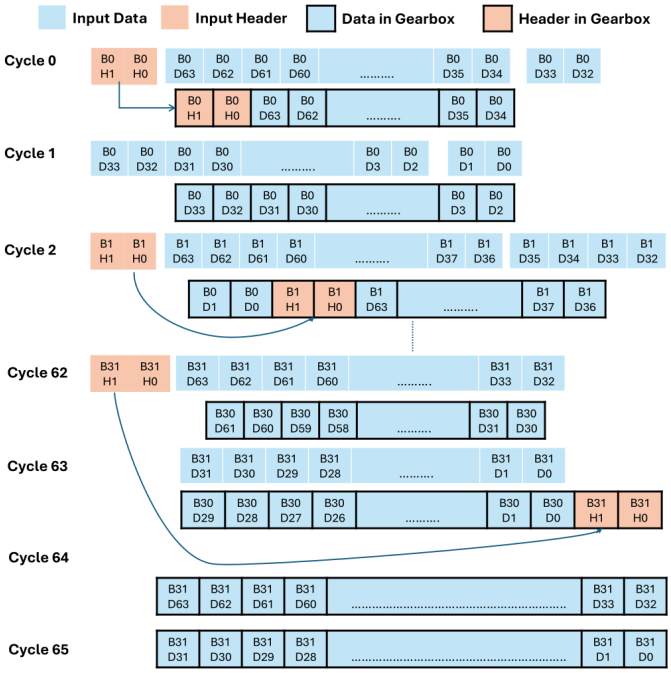

Fig. 7. Data transfer process in TX synchronous gearbox [27]

3.2. Receiver synchronization

As mentioned earlier, receiver synchronization is achieved using the bitslip technique. A Mealy state machine, shown in Fig. 8, was used to manage this process. Once the reset sequence in the GTH module is completed, the state machine transitions from RESET_ST to UNSYNCHRONIZED_ST. In this state, if a valid header ("10" or "01") is received, the machine enters the SYNCHRONISATION_ST state, where it must receive 128 consecutive valid headers to confirm synchronization. Upon successfully receiving this sequence, the machine transitions to SYNCHRONIZED_ST, indicating that the receiver is correctly synchronized. If, in any state other than RESET_ST, an incorrect header ("11" or "00") is detected, the state machine returns to UNSYNCHRONIZED_ST and sets the bitslip signal to 1 until a valid header is received.

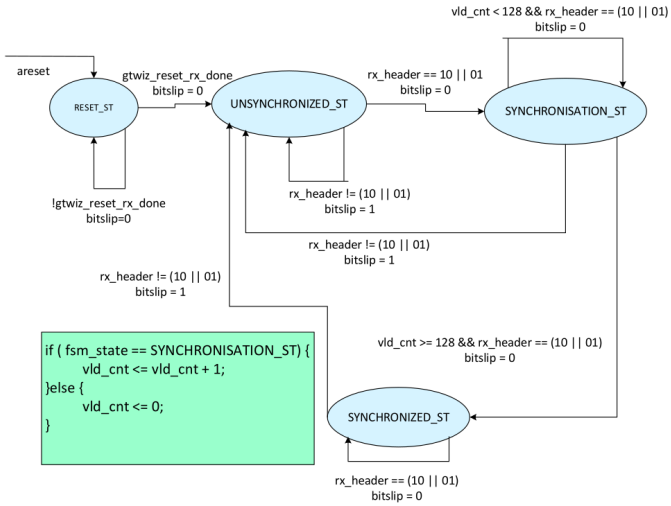

Fig. 8. State machine responsible for receiver synchronization

3.3. Transceiver configuration

The GTH_TRANSCEIVER module was generated using the UltraScale FPGAs Transceivers Wizard, which is available within the Vivado suite. This tool facilitates the configuration of all essential components required for the proper operation of GTH transceivers. The same parameters were configured for both the transmitter and receiver, as shown in Table 1.

Table 1
Transceiver configuration

Parameter	Value
Line rate (Gb/s)	16.375
Actual reference clock (MHz)	163.75
Encoding	Sync. gearbox for 64B/66B
User data width	64
Internal data width	32
Buffer	Enable
TXOUTCLK source	TXOUTCLKPMA
RXOUTCLK source	RXOUTCLKPMA
Link coupling	AC
DRP clock frequency (MHz)	50

A detailed description of the parameters listed above, as well as additional information regarding GTH configuration, can be found in the documentation provided by Xilinx [28].

3.4. Simulation

Similarly to the Aurora protocol, simulations of the custom protocol were conducted in loopback mode. Assuming a theoretical cable delay of 5 ns/m and that 8 values of 32 bits each need to be transmitted (4 transactions, each 64 bit), the total delay is calculated as follows:

$$\begin{aligned}\text{Custom protocol delay} &\approx 14 \cdot \frac{1}{255.86\ \text{Mhz}} \\ &\quad + 4 \cdot \frac{1}{255.86\ \text{Mhz}} + 5\ \text{m} \cdot 5\ \frac{\text{ns}}{\text{m}} \\ &\approx 95.38\ \text{ns}. \end{aligned} \tag{3}$$

Processing a 64-bit data vector through both the transmitter and receiver takes 14 clock cycles, including one clock cycle for scrambling and one for descrambling. The delay value obtained fully meets the design requirements and is significantly lower than that of the ready-made solutions available in the Xilinx's portfolio.

4. PROTOCOLS' COMPARISON

A comparison of the transmission parameters of the protocols discussed in this article is presented in Table 2.

The only protocol that met the design requirements was the custom protocol. When compared to Aurora 64b/66b, the custom implementation proved to be more than twice as fast in

Table 2
Comparison of protocol transmission parameters

	Aurora 8b/10b	Aurora 64b/66b	Custom protocol
Maximum latency for transmitting the smallest data packet, measured in user clock cycles on the transmitter side (excluding transmission medium delay)	29	45	14
Maximum latency in ns for transmitting eight 32-bit values over a 5-meter distance (assuming a cable delay of 5 ns/m)	250	217	95.38
Coding efficiency [%]	80	96.97	96.97
Maximum transmission speed [Gb/s]	6.6	16.375	16.375
DC balance	Excellent	Very good	Very good

transmitting the entire data packet meant to simulate data transfer from all ADC channels of radiation detectors. In the cases where latency is less critical, Aurora 64b/66b can be a suitable choice due to its high data throughput and support for multi-lane transmission. As for the Aurora 8b/10b protocol, it is reasonable to use it only in the cases where 8b/10b encoding provides better transmission quality than more efficient encoding types. Unfortunately, the main limitation of the Aurora 8b/10b protocol is the maximum transmission speed achievable in the GTH setup, which is only 6.6 Gb/s.

It is essential to note that the actual delay may differ by a few nanoseconds from the values presented in the table. The first reason for this variation is the deviation in the reference clock frequency of the transmitter circuit. The second reason is the phase difference between the transmitter clock and the recovered clock in the receiver circuit, which can shift with each transceiver reset [29].

5. HARDWARE SETUP

To verify the proper functionality of the custom protocol developed by our team, tests were conducted using two Xilinx evaluation boards: ZCU102 and ZCU106. Vivado 2022.1 was used for the design and integration of the project. Transmission was performed via small form-factor pluggable transceiver (SFP) connectors, utilizing both multimode fiber optic cables and direct attach copper cables (DAC). The tests covered three distances: 1 m, 3 m, and 5 m. Figure 9 illustrates the tested transmission setup. The ZCU106 board served as the transmitter, while the ZCU102 board acted as the receiver. In the CPU part of the Zynq UltraScale+ SoC, test data were generated and sent via direct memory access (DMA) to the programmable logic. DMA was chosen due to its seamless integration with the Python productivity for Zynq (PYNQ) framework, which simplifies data transfers between software and hardware components. From there, the data were transmitted to the ZCU102 board using the custom protocol. On the receiver side, correctly received data were transferred from the programmable logic to the processor's memory using DMA. By leveraging Ubuntu and the PYNQ framework on both evaluation boards, we were able to compare and verify whether the transmitted data were received correctly.

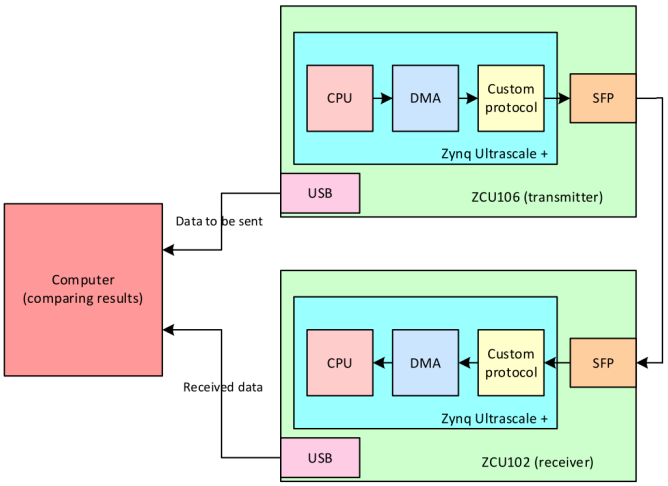

Fig. 9. Test transmission diagram

The actual test setup, along with the correct connections, is shown in Fig. 10. For the ZCU106 board, the X0Y15 channel located in Quad X0Y3 was used to carry out the transmission. On the ZCU102 board, the X1Y14 channel corresponding to Quad X1Y3 was utilized. The reference clock source for the GTH transmitter originated from the SI5328 module, while the clock signal for the receiver was provided by the SI570 module. Each module was located on the same evaluation board as the transmitter and receiver, respectively. The experiments were conducted in an environment not exposed to direct ionizing radiation.

Fig. 10. Test setup. On the left is a ZCU106 board, serving as a transmitter, and on the right is a ZCU102 board, serving as a receiver. An additional SFP+ FPGA Mezzanine Card was used for testing loopback mode with different clock signal generators

6. TESTING

To test the transmission using the custom protocol, data packets of lengths 10, 100, and 1000 times 64 bits were transmitted. For optical fiber transmission, the process was successful with no errors or packet losses across all tested transmission medium lengths. In the case of DAC cables, transmission was seamless for cables of 1 m and 3 m in length. However, with a 5 m cable, frequent data errors and receiver synchronization losses began to occur. This behavior indicated potential signal integrity issues.

To confirm these assumptions, the authors conducted transmission quality measurements using the integrated bit error ratio tester (IBERT) tool included in the Vivado suite. This tool allows the determination of bit error rate (BER) for various lengths of pseudorandom bit sequences (PRBS). Additionally, it provides a visual representation of transmission quality through an eye diagram. More information about the IBERT tool can be found in [30].

Figure 11 shows the eye diagram obtained during the transmission of a PRBS-31 sequence over a 5 m distance with default transmission settings. The larger the blue area in the diagram, the lower the bit error rate (BER) of the transmission. A particularly critical region is the sector where the unit interval equals 0, as this is the point where the receiver determines whether the received signal represents a logical 1 or 0.

To enhance transmission quality, channel equalization was applied on the transmitter side using post-cursor equalization. This effectively compensated for the effects of inter-symbol interference. The eye diagram after applying post-cursor equalization is shown in Fig. 12. Compared to Fig. 11, the "blue eye" is wider and more defined.

Following this equalization, quality analysis of the transmission was performed by continuously transmitting a PRBS-31 sequence over a 5m distance at a speed of 16.375 Gbit/s using a DAC cable. After 48 hours of testing, no transmission errors were observed, achieving a BER of 3.39E-16. Subsequently, tests with the custom protocol were conducted, and after applying equalization, issues with erroneous transmissions and receiver desynchronization were fully resolved.

7. COMPARISON OF MGT-BASED AND IOSERDES-BASED DATA TRANSMISSION

A significant advantage of MGT-based data transmission lies in its much higher transfer speed. For example, GTH transceivers offer data rates of up to 16.375 Gb/s per channel, whereas IOSERDES modules provide approximately 320 Mb/s. As a result, the current solution, which is based on IOSERDES, requires the use of four parallel channels to achieve comparable throughput.

The existing IOSERDES-based system, utilizing HDMI cables, follows a system-synchronous transmission architecture. This requires careful PCB design to minimize clock signal trace delays, which could otherwise lead to metastability issues. In contrast, MGT-based transmission is self-synchronous, thereby eliminating such timing concerns.

Another notable advantage of MGTs is their scalability. Upgrading to faster transceivers such as GTY would require only minimal design modifications – limited to configuration updates in the MGT setup and a resynthesis of the Vivado project. However, due to the multi-gigabit nature of the transmission, PCB traces must be precisely engineered to maintain impedance matching. This may necessitate the use of higher-grade laminate materials, curved trace routing, or buried vias [31], all of which can increase the overall cost of the hardware.

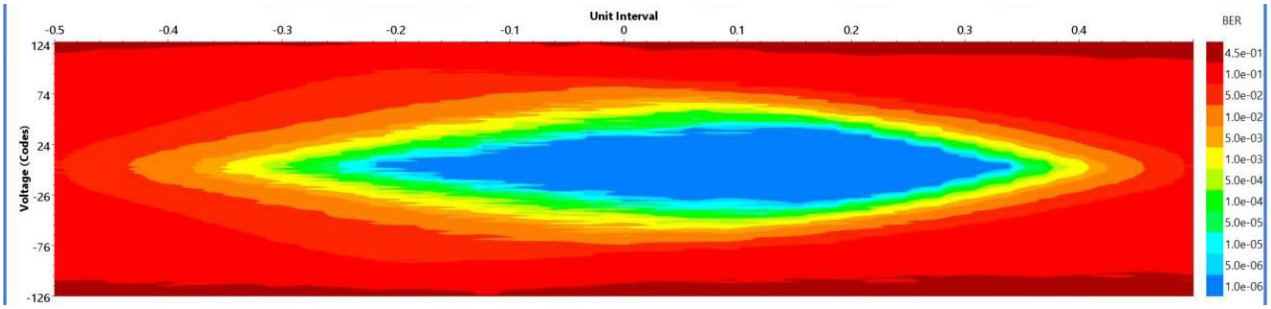

Fig. 11. Eye diagram for 5m transmission, data rate 16.375 Gb/s, DAC cable, PRBS-31, default settings

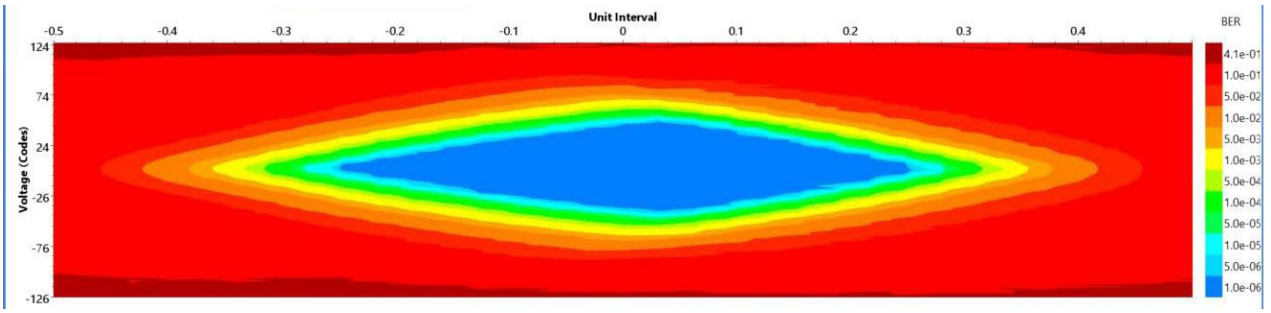

Fig. 12. Eye diagram for 5m transmission, data rate 16.375 Gb/s, DAC cable, PRBS-31, TX Post-Cursor 9.12 dB

Moreover, designing a PCB for high-speed serial links typically requires signal integrity simulations using IBIS (input/output buffer information specification) models, which may prolong the development timeline. To ensure error-free transmission over distances of up to 5 meters, the transmission channel may need fine-tuning or the use of optical transceiver modules. While optical modules improve signal integrity, they also introduce additional active components into the transmission path. This, in turn, increases power consumption and introduces potential points of failure, also potentially reducing overall system reliability.

Power estimations performed using Xilinx Power Design Manager indicate a significant difference in energy consumption between the two transmission approaches. A single connection between the PM (processing module) and the TCM (timing and control module) implemented using four IOSERDES channels consumes approximately 0.0063 W, whereas a comparable connection using a single GTH transceiver consumes approximately 0.504 W. This represents a substantial increase in power requirements for the MGT-based solution.

However, due to increasing computational demands imposed by the integration of faster and higher-resolution ADCs, the use of more advanced FPGAs – specifically, the Kintex UltraScale series – is currently under consideration as a replacement for the existing XC7K160T-2FBG676C (Kintex-7) devices. All Kintex UltraScale devices are equipped with GTH or GTY transceivers, meaning that the adoption of MGT-based transmission would not incur additional FPGA-related costs, as the upgraded devices inherently support this capability.

Given the superior bandwidth offered by MGTs, it is also worth exploring alternative interconnect topologies beyond the traditional point-to-point architecture. For instance, multiple FPGAs (e.g. 4 or 8) could be integrated onto a single PM board, with one FPGA designated to aggregate and forward data from the others to the TCM. Such a design could capitalize on the high-speed nature of MGT links to simplify cabling and reduce the number of direct PM-TCM connections, potentially enhancing system scalability and modularity.

8. CONCLUSIONS

Based on the simulations and tests conducted, it can be concluded that data transmission between FPGAs equipped with GTH-series multigigabit transceivers can be effectively used in the FIT experiment at ALICE CERN. Ready-made solutions such as Aurora 8b/10b and Aurora 64b/66b fail to meet the critical requirement of a maximum transmission latency of 200 ns, significantly exceeding this threshold.

In contrast, the custom protocol developed by our team, based on 64b/66b encoding, successfully fulfills all project requirements, significantly outperforming the ready-made solutions. The protocol is not only suitable for high-energy physics experiments but can also find applications in autonomous vehicles, where large amounts of data must be transmitted between systems in the shortest possible time.

Using DAC cables for transmission can be a cost-effective alternative to solutions based on fiber-optic SFP modules and optical cables. However, it is important to note that employing DAC cables for transmissions over several meters may require additional effort to better match the transmitter with the receiver. This can involve techniques such as using an equalizer or applying pre/post-emphasis.

An undeniable advantage of fiber-optic cables is their light weight and compact size, which can be crucial in scenarios where numerous transmission connections are required. However, it is important to consider that fiber-optic transmission relies on devices to convert electrical signals into light and vice versa. This increases the likelihood of system failure as compared to systems based solely on electrical transmission.

REFERENCES

[1] M. Cannon, M. Wirthlin, A. Camplani, M. Citterio, and C. Meroni, "Evaluating Xilinx 7 Series GTX Transceivers for Use in High Energy Physics Experiments Through Proton Irradiation," *IEEE Trans. Nucl. Sci.*, vol. 62, no. 6, pp. 2695–2702, Dec. 2015, doi: 10.1109/TNS.2015.2497216.

[2] A. Harding, K. Ellsworth, B. Nelson, and M. Wirthlin, "Characterization and Mitigation of the MGT-Based Aurora Protocol in a Radiation Environment," *2013 IEEE Radiation Effects Data Workshop (REDW)*, USA, 2013, pp. 1–4, doi: 10.1109/REDW.2013.6658186.

[3] R. Giordano, S. Perrella, D. Barbieri, and V. Izzo, "A Radiation-Tolerant, Multigigabit Serial Link Based on FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 67, no. 8, pp. 1852–1860, Aug. 2020, doi: 10.1109/TNS.2020.2998612.

[4] X. Liu, Q. Deng, and Z. Wang, "Design and FPGA Implementation of High-Speed, Fixed-Latency Serial Transceivers," *IEEE Trans. Nucl. Sci.*, vol. 61, no. 1, pp. 561–567, Feb. 2014, doi: 10.1109/TNS.2013.2296301.

[5] J. Wang *et al.*, "Fixed-Latency Gigabit Serial Links in a Xilinx FPGA for the Upgrade of the Muon Spectrometer at the ATLAS Experiment," *IEEE Trans. Nucl. Sci.*, vol. 65, no. 1, pp. 656–664, Jan. 2018, doi: 10.1109/TNS.2017.2784411.

[6] K. Chen, H. Chen, W. Wu, H. Xu, and L. Yao, "Optimization on Fixed Low Latency Implementation of the GBT Core in FPGA," *J. Instrum.*, vol. 12, no. 7, p. 07011, Jul. 2017, doi: 10.1088/1748-0221/12/07/P07011.

[7] J. Hu, J. Wang, and R. Li, "Low-Latency Ultra-Wideband High-Speed Transmission Protocol Based on FPGA," *J. Phys.-Conf. Ser.*, vol. 1621, no. 1, 2020, doi: 10.1088/1742-6596/1621/1/012066.

[8] T. Kurosawa, Y. Yamaguchi, R. Tsugami, T. Fujiwara, T. Fukui, and S. Narikawa, "Low-latency and High-bandwidth Communication Using HSS Transceivers on a Versal FPGA," *IEEE International Conference on Consumer Electronics (ICCE 2024)*, USA, Jan. 2024.

[9] B.A. Reese, L. Ruckman, R. Herbst, D. Doering, and M. Kwiatkowski, "PGP4: A Pretty Good Protocol for 10+ Gigabit FPGA-to-FPGA Communication," *IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC)*, Nov. 2022.

[10] M. Słupecki, "The Fast Interaction Trigger for the ALICE Upgrade," Ph.D. dissertation, JYU Dissertations 235, University of Jyväskylä, 2020, CERN-THESIS-2020-143.

[11] F. Zhang, *High-speed Serial Buses in Embedded Systems*, 1st ed., Springer, 2020, doi: 10.1007/978-981-15-1868-3.

[12] L. Li, Y. Wang, Y. Hu, and X. Zhang, "Study of Delay Instabilities in Xilinx FPGA-embedded Multi-Gigabit Transceivers for Clock Distribution and Synchronization," *IEEE Trans. Nucl. Sci.*, vol. 71, no. 11, pp. 2457–2468, Nov. 2024, doi: 10.1109/TNS.2024.3469166.

[13] "JESD204B Survival Guide," Analog Devices. [Online]. Available: https://www.analog.com/en/lp/001/glp2-jesd204b-survival guide-article.html [Accessed: Nov. 24, 2024].

[14] S. You, F. Li, C. Zhang, and Z. Wang, "High-Speed Serial Interface Transmitter Controller Based on JESD204B for 1GSPS ADCs," *IEEE International Conference on Electron Devices and Solid-State Circuits (EDSSC)*, Jun. 2015, doi: 10.1109/EDSSC.2015.7285056.

[15] M. Assaad and A. Harb, "A Synthesizable Serial Link for Point-to-Point Communication in SoC/NoC," *Proc. 29th IEEE International Conference Microelectronics (IEEE-ICM17)*, Lebanon, Dec. 2017, doi: 10.1109/ICM.2017.8268824.

[16] J. Zhang, Q. Lin, Y. Zhang, and Z. Chen, "Design and Implementation of High-Speed Data Transmission Scheme between FPGA Boards Based on Virtex-7 Series," *Proc. 2nd IEEE Advanced Information Management, Communications, Electronic and Automation Control Conference (IMCEC)*, China, 2018, doi: 10.1109/IMCEC.2018.8469284.

[17] A. Athavale and C. Christensen, "High-Speed Serial I/O Made Simple: A Designers' Guide, with FPGA Applications," Xilinx, 2005. [Online]. Available: https://www.xilinx.com/publications/archives/books/serialio.pdf [Accessed: Nov. 24, 2024].

[18] "High-Speed Serial Technologies for Adaptive SoCs and FPGAs." AMD. [Online]. Available: https://www.amd.com/en/products/adaptive-socs-and-fpgas/technologies/high-speed-serial.html [Accessed: Nov. 24, 2024].

[19] "UltraScale Architecture GTH Transceivers User Guide – UG576 (v1.6)," Xilinx, Aug. 26, 2019. [Online]. Available: https://www.xilinx.com/support/documentation/user_guides/ug576-ultrascale-gth-transceivers.pdf [Accessed: Nov. 24, 2024].

[20] S. Liu, H. Lin, and C. Wang, "The Implementation of High-speed Data Transmission for 8B/10B Protocol on FPGAs," *2017 International Conference on Applied System Innovation (ICASI)*, pp. 810–813, May 2017, doi: 10.1109/ICASI.2017.7988557.

[21] R. Giordano and A. Aloisio, "Fixed-Latency, Multi-Gigabit Serial Links With Xilinx FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 58, no. 1, pp. 194–201, Feb. 2011, doi: 10.1109/TNS.2010.2101083.

[22] A. Salvador and V. Corso, "100 Gbit/s scrambler architectures for OTN protocol: FPGA implementation and result comparison," *2012 IEEE 55th International Midwest Symposium on Circuits and Systems (MWSCAS)*, USA, 2012, pp. 904–907, doi: 10.1109/MWSCAS.2012. 6292167.

[23] "Difference between synchronous scrambler and self-synchronizing scrambler." RF Wireless World. [Online]. Available: https://www.rfwireless-world.com/Terminology/Difference-between-synchronous-scrambler-and-self-synchronizing-scrambler.html [Accessed: Nov. 24, 2024].

[24] "wlanNonHTData – MATLAB Function." MathWorks. [Online]. Available: https://in.mathworks.com/help/wlan/ref/wlannonhtdata.html [Accessed: Nov. 24, 2024].

[25] "Aurora 8B/10B v11.1 LogiCORE IP Product Guide," Xilinx, 2021. [Online]. Available: https://docs.amd.com/r/en-US/pg046-aurora-8b10b [Accessed: Nov. 24, 2024].

[26] S. Shin, S. Choi, E. Lee, S. Lee, and H. Yoo, "Implementation of Aurora Interface using SFP+ Transceiver," *19th International SoC Design Conference (ISOCC)*, pp. 350–351, 2022.

[27] J. Zhang, Y. Wang, "FPGA Implementation of Fixed-Latency Command Distribution Based on Aurora 64B/66B," *IEEE Trans. Nucl. Sci.*, vol. 71, no. 6, pp. 1348–1356, June 2024, doi: 10.1109/TNS.2024.3400378.

[28] "UltraScale FPGAs Transceivers Wizard v1.7," Xilinx Documentation, 2020. [Online]. Available: https://www.xilinx.com/support/documents/ip_documentation/gtwizard_ultrascale/v1_7/pg182-gtwizard-ultrascale.pdf [Accessed: Nov. 24, 2024].

[29] M. Cannon, M. Wirthlin, A. Camplani, M. Citterio, and C. Meroni, "Evaluating Xilinx 7 Series GTX Transceivers for Use in High Energy Physics Experiments Through Proton Irradiation," *IEEE Trans. Nucl. Sci.*, vol. 62, no. 6, pp. 2695–2702, Dec. 2015, doi: 10.1109/TNS.2015.2497216.

[30] "IBERT for UltraScale/UltraScale+ GTH Transceivers v1.4 LogiCORE IP Product Guide," Xilinx, 2020. [Online]. Available: https://docs.amd.com/v/u/en-US/pg173-ibert-ultrascale-gth [Accessed: Nov. 24, 2024].

[31] D. Telian, "Signal Integrity," in *Practice: A Practical Handbook for Hardware*, SI, FPGA & Layout Engineers, 2023.