High-Performance and Energy-Efficient Fault Tolerance FPGA-to-FPGA Communication

Naresh Kumar Reddy (✉ naresh.klu@gmail.com)

ICFAI Foundation for Higher Education Faculty of Science and Technology <https://orcid.org/0000-0002-9865-3856>

Swamy Cherukuru

NITW

Veena Vani

IFHE: ICFAI Foundation for Higher Education

Vishal Reddy

IFHE: ICFAI Foundation for Higher Education

Research Article

Keywords: communication, error detection, error correction, FPGA, data transmission

Posted Date: August 9th, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-741169/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

High-Performance and Energy-Efficient Fault Tolerance FPGA-to-FPGA Communication

¹B. Naresh Kumar Reddy, ²Ramalingaswamy Cheruku, ³B. Veena Vani and ³G. Vishal Reddy

¹Department of Electrical Engineering, Indian Institute of Technology Delhi, India.

²Department of Computer Science and Engineering, NIT Warangal, India.

³Faculty of Science and Technology, IcfaiTech, ICFAI Foundation for Higher Education, Hyderabad

Abstract. These days, due to the increasing demand for high speed and parallel computation, several real world applications and systems include multiple FPGAs in them. Due to this, FPGAs often need to communicate among them. So, communication between the FPGAs is one of the key factors that determines the accuracy, performance and correctness of the entire multiple FPGAs systems or applications. This paper presents the design of an efficient multi-bit fault tolerant communication system for FPGA-to-FPGA communication. The proposed design is synthesized and also simulated through Vivado design suit 2018.3 and was communicated with two Kintex-7 FPGA boards. When compared with the existing FPGA-to-FPGA communication and inter FPGA communication designs, the proposed design have higher performance, error detection and correction capability.

Keywords: communication, error detection, error correction, FPGA, data transmission

1. Introduction

The universal serial bus (USB) interface is used as a standard type of interface in many different kinds of devices for the transmission of data. The data transfer rate, low power and ease to use are the key features that made USB as an industry standard data transmission interface. Hot plugging can be achieved automatically in USB interface. Without shutting down the system, USB can connect the computer with electronic devices. A RS232 dependent system can be easily portable to USB interface by using embedded RS232 to USB converter. To bridge the difference in data rates among the 2 interfaces first-in first-out (FIFO) logic is used. The study of USB protocol with an FPGA development board is important. The USB standard interface broadly has 2 units, Parallel interface engine (PIE) and USB transceiver macro cell interface (UTMI). PIE is accountable for packet extraction or construction and also responsible for the communication with the peripherals. UTMI is connected to USB cables and are used for transmission of serial data and for synchronization of time frames. The serial interface engine (SIE) has two sub-blocks, Endpoint Logic sub-block and SIE control sub-block. The endpoint logic block has, FIFO, FIFO logic and endpoint number recognition. The SIE block has sequencing logic for USB packets and transactions managing, logic of address recognition and USB product identification logic (PID). SIE blocks lies in between the processing unit of computer and UTMI. It receives the data from the computer processor and transmits to the UTMI. The UTMI unit does clock synchronization, clock recovery, bit stuffing, data serialization and deserialization. UTMI uses differential signals to transmit data from one USB to other USB compactable device.

To meet the needs of the exponentially growing demand for high speed computations, we need to use multiple FPGAs. Applications that employ multiple FPGAs are the target for several fields that require high speed parallel computations. This gives the desired performance with high speed data processing.

Multiple FPGAs are been widely used in the areas like multi-processor system on chips (MPSoC), hardware emulation, hardware acceleration and so on. In all the above mentioned applications, to get the desired functionality of the systems, multiple FPGAs must communicate with high speed, with low power, without any errors during the data transmission. Error less communication play a crucial role in determining the performance and correctness of the system. So, this paper presents a fault tolerant communication system between the FPGAs. The proposed system uses Bose, Ray- Chaudhuri, Hocquenghem (BCH) codes, Universal Serial Bus (USB) transceiver with serial interface engine (SIE) and an asynchronous FIFO. BCH codes are the class of cyclic codes that are powerful and have the capability of multiple error detection and correction. The main aim of the study is to guarantee errorless and fault tolerant communication between multiple FPGA development boards.

The rest of the paper is organized as follows. Literature review is presented in Section II. The basics are presented in Section III. The proposed design for FPGA-FPGA communication is reported in Section IV. Comparisons to the other FPGA-FPGA communication techniques results and simulation results are presented in Section V. Conclusion of the paper is reported in Section VI.

2. Literature Review

FPGA devices play a crucial role in product prototyping of any advanced and complex designs. Estimating performance and power of a complex FPGA system is always a major challenge. An efficient and new methodology is proposed in [6], which precisely and quickly evaluates the performance and power consumption of a wireless base-band communication systems that are implemented on FPGAs. To make the real world systems robust, FPGAs need to communicate with other computing devices that makes the entire performance of the system to be boosted. In [7], a wireless communication system is implemented in between a microcontroller and an FPGA using UART protocol that allows the transfer of data between them.

Orthogonal Frequency Division Multiplexing (OFDM) is viewed as an important constituent in the modern day wireless communication because of its ability to render high spectral efficiency, higher data rates and is powerful against intra-symbol as well as inter-symbol interference. Design and implementation of various components of OFDM system is presented in [8]. Limited number of input-output pins (I/O pins) create problems in a multiple-FPGA systems. To address this issue, utilization of configurable logic blocks (CLBs) must be high enough, which is possible by efficient scheduling of communication among FPGAs. Different FPGA communication architectures are surveyed in [9] and qualitative factors, quantitative performance metrics are calculated. Multiple FPGA communication based on DDR interface is proposed in [10]. This approach substitutes the traditional LVDS pins with a stable communication among the devices. A very high data exchange rate is achieved between the multiple FPGAs with the stable communication channel proposed in [10].

Communication scheduling algorithms is proposed in [11] which provides the efficient usage of CLBs in multi FPGA system. In a high performance multi FPGA communication systems, to ensure high data transfer bandwidth, every node uses a very high speed bus to communicate with other computing node. As each compute node uses different bus communication protocols it is tough for the direct communication. In [12], a node-to-node communication interface Unified PHY interface (UPI) is proposed that can transmit various bus communication protocols like Ethernet protocol and PCIe protocols simultaneously, for reliable and efficient inter-processor communication in Field Programmable System-in-Chip or Platform-FPGA architectures. An FPGA-to-FPGA communication system is designed and implemented in [13] using USB transceiver with an asynchronous FIFO queue and SIE block.

3. Basics

BCH codes are the set of powerful and effective classes of cyclic codes which are strong base for several advanced multiple error correction and detection codes with an innate capability of multiple error detection and correction. BCH codes are constructed using polynomials over a finite field called Galois Field. In BCH codes, there is a flexible control against the correctable number of symbols. As implementation of BCH codes are not critical, so they are widely used in information coding. These codes are the basic foundation for several error detection as well as correction algorithms in present day. BCH provide wide range of controllability in terms of number of errors to be corrected and the length of the code word that is to be transmitted by the encoder.

In BCH codes,

Length of Block (c) = $(2^m) - 1$

Number of Bits (message) (k) $\geq c - (m \cdot p)$

Distance (Minimum) ($D_{\min}$) $\geq (2 \cdot p) + 1$

We obtain the length of the block, number of message bits, and minimum distance by using the above equations. From the above equations, 'p' number of errors or less than 'p' number of errors can be corrected. Generator polynomial in BCH codes is defined from the roots of Galois field $GF(2^m)$. Arithmetic operations such as addition, subtraction, multiplication and division can be performed by Galois field by satisfying some basic rules. A field that comprises the components of a finite number is called Galois field. $GF(xy)$ is the representation of the Galois field, where 'y' refers to a positive integer and 'x' refers to a prime number.

TABLE I. GF (2^4) REPRESENTATION BY $P(Y) = 1+Y+Y^4$

Roots of conjugate	Minimal Polynomial
0	y^1
1	y^1+1
$\beta^1, \beta^2, \beta^4, \beta^8$	$y^4+ y^1+1$
$\beta^3, \beta^6, \beta^9, \beta^{12}$	$y^4+ y^3+ y^2+y^1+1$
β^5, β^{10}	$y^2+ y^1+1$
$\beta^7, \beta^{11}, \beta^{13}, \beta^{14}$	$y^4+ y^3+1$

TABLE II. GF (2^4) POWER REPRESENTATION $P(Y) = 1+Y+Y^4$

Power representation	Polynomial representation	Power representation	Polynomial representation
0	0	β^7	$\beta^3+ \beta+1$
β^0	1	β^8	β^2+1
β^1	β	β^9	$\beta^3+ \beta$
β^2	β^2	β^{10}	$\beta^2+ \beta+1$
β^3	β^3	β^{11}	$\beta^3+ \beta^2+ \beta$
β^4	$\beta+1$	β^{12}	$\beta^3+ \beta^2+ \beta+1$
β^5	$\beta^2+ \beta$	β^{13}	$\beta^3+ \beta^2+1$
β^6	$\beta^3+ \beta^2$	β^{14}	$\beta^3+ 1$

The USB interface is also known as data interface, which enables communication between computers and peripherals. It can also provide power supply for some peripherals like flash memory sticks, disk drives and so on. It is easy to use, low cost and available in different sizes and provides a powerful connection system. USB standard has several versions and in each version data transfer speed varies. The speeds can vary from Megabits per second (Mbps) to Gigabits per second (Gbps).

Algorithm	
{	
Input:	Binary signal
Output:	Transmission signal (Code word)
Intermediate Results:	Transmitter polynomial
01:	The message signal in binary form need to be converted into polynomial form $F(x)$
02:	Multiply $F(x)$ with $M(x)$ // $B(x)= F(x) * M(x)$
03:	$B(x)$ which is in polynomial form need to be converted into binary form // i.e. $B[0111010110110110]$
}	

Fig. 1. Algorithm for BCH encoder

SIE block plays a vital role in transmitting as well as receiving the data with USB interface. SIE block acts as an intermediate between the central processing unit of a computer and the universal transceiver macro cell interface (UTMI). In Fig. 3, the block diagram of SIE block is presented. The SIE block handles all the receiving and sending of data in the form of transactions. It typically detects all the incoming packets, sends data, handshake and token packets, detects and generates reset, start-of-packet, end-of packet, resume signaling information, decodes and encodes the data on the bus in the required form, generates as the packet identifiers (PIDs), converts serial data of USB to parallel data on registers or memory and vice versa. UTMI block takes care about low level USB signaling and protocol. The main aim of UTMI block is to make the data compatible with the USB protocol and also to shift clock domain of the input or data signal from USB rate so that it is compatible with central processing unit (CPU)

Algorithm	
{	
Input:	Received signal
Output:	Corrected code word
Intermediate Results:	Number of errors and Error polynomial
01:	Received signal is to be converted into polynomial form $J(x)$ // $J(x)$ is polynomial form of received signal
02:	Find the number of syndromes C and R // $C=2*p$; $R=p$;
03:	Get syndrome value N_i // N_i ; $i=1,2,...,2*p$ from $J(X)$
04:	Get determinant of matrix L $//L = \begin{pmatrix} N_1 & N_2 & \dots & N_R \\ \vdots & \vdots & & \vdots \\ N_R & N_{R+1} & \dots & N_{2R-1} \end{pmatrix}$
05:	Calculate the number of errors in $J(x)$ if ($\det L \neq 0$) Then: number of errors in $J(x)$ is R if ($\det L = 0$) Then: number of errors in $J(x)$ is less than R
06:	Find error locator polynomial $\alpha(Z)$ // i.e. $\alpha(Z) = \alpha_R Z^R + \alpha_{R-1} Z^{R-1} + \dots + \alpha_1 Z + 1$ Inverse of the roots of $\alpha(Z)$ gives the location of errors.
07:	Get the error polynomial $T(x)$ //Where $T(x) = \sum$ Inverse of roots of $\alpha(Z)$
08:	To get the corrected code word $H(x)$ // $H(x) = J(x) + T(x)$
}	

Fig. 2. Algorithm for BCH decoder

Fig. 3. Block diagram of SIE block

4. Proposed Design

For efficient communication between multiple FPGAs, the architecture is shown in Fig. 4. FIFO queue and SIE block with USB transceiver is used in the architecture. By USB physical layer FPGAs communicate with each other using general purpose input or output (GPIO) ports present on FPGAs.

Fig. 4. Architecture of the system

In the proposed design, we introduce error detection and correction blocks of BCH codes before the SIE blocks which ensures fault tolerant communication among multiple FPGAs. Fig. 5 represents the system level architecture of the transmitting unit of the proposed high performance fault tolerant FPGA-to-FPGA communication system. In the transmitter (first FPGA) board, the data that is to be transmitted is stored in memory (flash) and then the data is sent to FIFO (asynchronous) queue. Before sending the data to SIE block, the data passes through BCH encoder block, which encodes the data that is to be transmitted to other FPGA through USB interface. The BCH encoder block inputs the binary data and encodes the signal with Galois field. The encoded signal is sent to SIE block to transmit it using USB interface.

Fig. 5. Transmitter architecture

For example, consider [00111] as the data that enters BCH encoder block with $p=3$ and $c=15$.

From the Galois field $GF(2^m)$

$$M(y) = \text{LCM} [f_1(y), f_2(y), \dots, f_{p-1}(y)]$$

Here $p=3$

$$M(y) = \text{LCM} [f_1(y), f_2(y), f_3(y), f_4(y), f_5(y)]$$

From Table I,

$$M(y) = \text{LCM}[(y^4+y+1)(y^4+y+1)(y^4+y^3+y^2+y+1)(y^4+y+1)(y^2+y+1)]$$

$$= (y^4+y+1)(y^4+y^3+y^2+y+1)(y^2+y+1)$$

$$= (y^{10}+y^9+y^8+y^5+y^4+y^2+y+1)$$

$$M(y) = (y^{10}+y^9+y^8+y^5+y^4+y^2+y+1)$$

Using encoder algorithm from Fig. 1,

$$\text{Step 1: } F(y) = (y^2+y+1)$$

$$\text{Step 2: } B(y) = F(y) * M(y)$$

$$B(y) = (y^2+y+1)(y^{10}+y^9+y^8+y^5+y^4+y^2+y+1)$$

$$B(y) = (y^{10}+y^9+y^8+y^5+y^4+y^2+y+1)$$

$$\text{Step 3: } B [000010100110111]$$

The digital signal B [000010100110111] is the output of the BCH encoder block. This output is sent to SIE block which converts B signal into USB supported format and sends to the other FPGA (receiver) using USB physical cable. Let us suppose that there are three errors introduced into transmitted signal B while transmission. So, B signal is changes to [010010100010011]. We need to detect the error bits in the received signal and need to correct the effectedbits due to transmission errors. Otherwise error data will be received by other FPGA which leads to wrong data processing which is not desired. To overcome this problem, we introduce BCH decoder block in the receiver FPGA after the SIE block which is represented in Fig.6. As we receive the transmitted data using USB cable, UTMI interface collects the data and sends to SIE block.

Fig. 5.Receiverarchitecture

Using encoder algorithm from Fig.2,

Step 1: $J(y) = [010010100010011]$
 $J(y) = (y^{13} + y^{10} + y^8 + y^4 + y + 1)$

Step 2: Here $p=3$

Total no of syndromes $(C) = 2 \times 3 = 6$

$R=2$

Step 3: We get the syndrome values N_1, N_2, N_3, N_4, N_5 and N_6 by using Table II.

$$\begin{array}{ll} N_1 = (\beta^{12}) & N_4 = (\beta^3) \\ N_2 = (\beta^9) & N_5 = (\beta^5) \\ N_3 = (\beta^{10}) & N_6 = (\beta^5) \end{array}$$

Step 4: $L = \begin{pmatrix} N_1 & N_2 & N_3 \\ N_2 & N_3 & N_4 \\ N_3 & N_4 & N_5 \end{pmatrix}$

Step 5: $\det(M) = \beta^4$

As $|M| \neq 0$, so number of errors in $J(y)$ are 3.

Step 6:
$$\begin{pmatrix} L \end{pmatrix} \begin{pmatrix} \alpha_R \\ \alpha_{R-1} \\ \vdots \\ \alpha_1 \end{pmatrix} = \begin{pmatrix} N_{R+1} \\ N_{R+2} \\ \vdots \\ N_{2R} \end{pmatrix}$$

By solving the above equation, we get

$$\begin{pmatrix} \alpha_3 \\ \alpha_2 \\ \alpha_1 \end{pmatrix} = \begin{pmatrix} \beta^5 \\ \beta^1 \\ \beta^{12} \end{pmatrix}$$

Error locator polynomial $\alpha(Z) = \alpha_3 Z^3 + \alpha_2 Z^2 + \alpha_1 Z + 1$
 $= \beta^5 Z^3 + \beta^1 Z^2 + \beta^{12} Z + 1$

In step 7, the multiple error locations are detected and in step 8 all the detected errors are corrected. If any errors occurs during the data transmission, we can correct the errors by including BCH encoder block in the transmitting FPGA and BCH decoder block in the receiving FPGA.

5. Simulation Results and Analysis

The proposed Fault Tolerance FPGA-to-FPGA Communication architecture is synthesized and also simulated through Vivado design suit 2018.3 and was communicated with two Kintex-7 FPGA boards.

This paper mainly concentrates on performance and energy. Performance is calculated in terms of throughput, frequency, bit length and area. Throughput is one of the most significant design metrics in an FPGA, which is the measure of information transmission finished per unit of time. The throughput of the processor is influenced by the activity recurrence and transmission capacity of the correspondence channels. On the other hand, speed can be described as dormancy, which can be detailed as the entirety of sender overhead, transport idleness and beneficiary overhead. Area is a significant standard for architecture. It tends to be characterized as number of semiconductors, CLB's, memory and wire length. Particularly, for FPGA platform, number of CLB's are generally utilized as a area estimation. Interconnect usage is the sum or level of time that the wire is conveying data. Power is the estimation of energy utilization in interconnect wires.

In view of the distributed outcomes, we have summed up the performance estimations for various FPGA to FPGA communication methods in Table III. The proposed method gives an enormous scope of throughput and its most extreme throughput essentially beats different FPGA to FPGA communication methods shown in Fig. 6.

TABLE III. SUMMARY OF THE PERFORMANCE ESTIMATIONS FOR VARIOUS FPGA TO FPGA COMMUNICATION METHODS

	Throughput (MByte/s)	Frequency (MHz)	Bit length (Bit)	Area (CLB's)
[11]	1450	48	16	3852
[12]	1568	46	16	3564
[13]	2500	56	16	3238
Proposed Architecture	3200	62	16	2864

Fig. 6. Comparison of Throughput

To compare our proposed architecture with the state-of-the-art methods, we have collected energy results from four various FPGA to FPGA implementations tabulated in Table IV and shown in Fig. 7.

TABLE IV. SUMMARY OF THE ENERGY EFFICIENCY FOR VARIOUS FPGA TO FPGA COMMUNICATION METHODS

	Power (W)	Time (ms)	Energy(J)
[11]	84.16	29.16	1892.49
[12]	72.33	24.98	1681.78
[13]	63.48	23.16	1576.84
Proposed Architecture	56.19	16.9	1262.71

Energy efficiency results show that our proposed FPGA to FPGA communication method is on average 1.12 times, 1.3 times and 1.5 times higher than the state-of-the-art FPGA to FPGA communication methods, respectively.

Fig. 7. Comparison of Energy

6. Conclusion

FPGA to FPGA communication architectures play a crucial role in determining the performance and energy consumption of platform- FPGAs containing embedded coarse-grain modules. In the Paper, an efficient Fault Tolerance FPGA-to-FPGA Communication architecture studied. The design of efficient fault tolerance communication architecture is a challenging multi objective optimization problem. We have efficiently implemented the FPGA to FPGA data transferring. Energy efficiency results show that our proposed method is on average .12 times, 1.3 times and 1.5 times higher than the state-of-the-art FPGA to FPGA communication methods, respectively. In future work, we will look into ways to further improve the search process, and explore how data transfer through wireless among FPGA boards.

Declaration:

Conflict of interest: The authors declare that they have no conflict of interest

References

- [1]. L. A. Plana et al., "spiNNlink: FPGA-Based Interconnect for the Million-Core SpiNNaker System," in IEEE Access, vol. 8, pp. 84918-84928, 2020.
- [2]. P. Liu, S. Li and Q. Ding, "An Energy-Efficient Accelerator Based on Hybrid CPU-FPGA Devices for Password Recovery," in IEEE Transactions on Computers, vol. 68, no. 2, pp. 170-181, 1 Feb. 2019,
- [3]. J. Lorandel, J. Prévotet and M. Hlard, "Fast Power and Performance Evaluation of FPGA-Based Wireless Communication Systems," in IEEE Access, Vol 4, 2016.
- [4]. R. F. Molanes, J. J. Rodrguez-Andina and J. Faria, "Performance Characterization and Design Guidelines for Efficient Processor–FPGA Communication in Cyclone V FPSoCs," in IEEE Transactions on Industrial Electronics, vol. 65, no. 5, pp. 4368-4377, May 2018.
- [5]. Z. Zheng and G. Li, "An Efficient FPGA Design and Performance Testing of the ACE Algorithm for PAPR Reduction in DVB-T2 Systems," in IEEE Transactions on Broadcasting, vol. 63, no. 1, pp. 134-143, March 2017.
- [6]. JordaneLorandel, Jean-Christophe Prevotet, and MarylineHlard, "Fast Power and Performance Evaluation of FPGA-Based Wireless Communication Systems" IEEE Access, Vol. 4, 2016.
- [7]. Pratik Shrestha, BinayakSubedi, Megan Girard, Dr. Chirag Parikh and Dr. NabeehKandalaft, "Wireless Communication Between FPGA and Microcontroller" IEEE Annual Computing and Communication Workshop and Conference (CCWC), 2020.
- [8]. S.W.A. Shah, H.S. Okleh, A. S. Hussaini, R.A. Abd-Alhameed, J. M. Noras, and J. Rodriguez, "FPGA Implementation of Modern Wireless Communication System" 2012 IEEE 17th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD).
- [9]. Terrence S. T. Mak*, Pete Sedcole, Peter Y. K. Cheung and Wayne Luk, "ON-FPGA COMMUNICATION ARCHITECTURES AND DESIGN FACTORS" International Conference on Field Programmable Logic and Applications, 2006.
- [10]. LT Melo, SHC Santana, AG Silva-Filho, ME Lima, VWC Medeiros and MLM Marinho, "An inter-FPGA communication bus with error detection and dynamic clock phase adjustment" Journal of the Brazilian Computer Society Vol. 21, 2015.
- [11]. JinawooSuh, Dong-In Kang, and Stephen P. Crag, "A communication Scheduling Algorithm For Multi-FPGA Systems", IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM), 2000.
- [12]. An WU, Jin XI, XueLiang Du, ShuaizhiGuo, "A Flexible FPGA-to-FPGA Communication System", ICACT Transactions on Advanced Communications Technology (TACT) Vol. 5, Issue 3, May 2016.
- [13]. Guo-Ming Sung, Li-Fen Tung, Hsin-Kwang Wang, and Jhih-Hao Lin, "USB Transceiver With a Serial Interface Engine and FIFO Queue for Efficient FPGA-to-FPGA Communication", IEEE Access, Vol 8, 2020.