DEVELOPMENT OF A FIELD-DEPLOYABLE BIT ERROR RATE TESTER FOR 10 Gbps FIBER OPTIC TRANSMISSION UTILIZING FIELD PROGRAMMABLE GATE ARRAY TECHNOLOGY

A thesis presented to the faculty of the Graduate School of Western Carolina University in partial fulfillment of the requirements for the degree of Master of Science in Technology.

by

Jeremy Brandon Howell

Director:
Dr. Paul Yanik
Associate Professor
School of Engineering and Technology

Committee Members:
Dr. Robert D. Adams
School of Engineering and Technology
Dr. Weiguo Yang
School of Engineering and Technology

April 2020

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

DEVELOPMENT OF A FIELD DEPLOY ABLE BIT ERROR RATE TESTER FOR 10 GIGABIT FIBER OPTIC BY UTILIZING FIELD PROGRAMMABLE GATE ARRAY TECHNOLOGY

Jeremy Brandon Howell, Master of Science in Technology.
Western Carolina University (April 2014)
Director: Dr. Paul Yanik

The bit error rate of a digital signal is the number of bits received incorrectly divided by the total number of bits received. A Bit Error Rate Tester (BERT) is a piece of equipment designed to calculate the bit error rate of a communication medium. The typical method of bit error rate testing is to produce a pseudorandom-binary sequence that can be transmitted over a fiber optic line and verified on the receiving end. The current equipment for bit error rate testing of 10 Gbps fiber optic data equipment is generally large, lab bench-based equipment which is not feasible for use in the field. Utilizing current FPGA technology, the design, and the manufacture of BERT equipment for 10 Gbps fiber optic data lines that is compact enough for field use should be feasible. This thesis investigates prevailing methods for bit error testing and compares them for relative strengths and weaknesses. Based on this analysis, a proposed process will allow BERT implementation in a compact package. Successful development and implementation of this design will facilitate the productization of a portable and cheaper alternative to the more expensive and stationary desktop BERTs in current use.

INTRODUCTION

According to IBM, 2.5 quintillion bytes of data are created every day, with that number increasing every year. The expectation that data will be received correctly at the destination is not always justified. There are numerous reasons for data to be lost or corrupted in transmission. Sometimes the reason that data can get lost in transit is obvious hardware failures, such as damaged or unplugged communication lines. Other times the reason is that communication lines and connections are bottlenecks that cause lost data. Each of the different types of data transmission technology has an expected error rate at its designated bandwidth. The bit error rate (BER) is the number of bits received in error divided by the number of bits received in total. A Bit Error Rate Tester (BERT) is a device used to measure the bit error rate (BER) and in the case of fiber optics lines, it is usually in an electronics laboratory or a desktop device [12].

High-speed communication processing requires devices that are capable of processing large amounts of data at a very fast rate. This high-speed processing is more than most computers can handle. The two devices commonly used for this operation are the Application Specific Integrated Circuit (ASIC) and the Field Programmable Gate Array (FPGA). Both the ASIC and FPGA are used in designs to perform these operations; however, the FPGA has the added advantage of being programmable, while the ASIC is not programmable and has one set of capabilities hardcoded at the factory that produces the device. An FPGA is a device that contains many logic blocks, each having a large number of logic gates. These logic gates mimic the output of any simple computer logic. When multiple blocks are employed, the FPGA is capable of processing and directing a large amount of data, in a very quick and efficient manner.

This research explores the feasibility of creating an FPGA based design that can utilize a 10 Gigabit transceiver to function as a BERT. This Mini BERT design will be a smaller, mobile alternative to the laboratory bench based non-mobile BERT.

1 LITERATURE REVIEW

1.1 The Basics of a BERT

To provide the best communication service possible, testing communication devices before putting these devices into service is needed to determine if the devices will perform as required. Testing allows communication technicians to fix any faulty line sections or connections before affecting end-user satisfaction. A Bit Error Rate Tester (BERT) is a device that can send and receive data over communication hardware and lines and is used to estimate the percentage of bits that would be received incorrectly in a real-world situation. The BERT will calculate the bit error rate by dividing the total bit errors by the number of bits received and then display the error rate on a screen display or in a data log. Since the purpose of a BERT is to determine the limits of a communication device or line, the data sent needs to represent the most random and complex data that would be transmitted and received while in regular service. Because this data also needs to be verified as it is received, genuinely random data is not an option. A Pseudorandom Binary Sequence (PRBS) emulates what an in-service binary data stream would look like in the real world but can be predictable when decoded with the proper algorithm or decoding table. Therefore, one of the many PRBSs should be the chosen data sequence used for BERTs [1].

1.2 Bit Error Rate Confidence Level and Duration of Test

Tests are run to achieve confidence that the device is going to perform as expected in service. These tests also need to be run long enough to get a high level of confidence without using an excessive amount of time. The object is to get the highest confidence in your equipment and hardware in the shortest amount of time. The confidence level of the bit error rate is the percentage of certainty that the actual bit error rate is below a specific value. The confidence level is found by (1):

$$CL = 1 - e^{-N \cdot BER} \cdot \sum_{k=0}^{E} \frac{(N \cdot BER)^k}{k!} \tag{1}$$

where E is the total number of errors detected while the test was performed, and BER is a specified bit error rate. N is found by (2):

$$N = BPS \cdot T \tag{2}$$

where BPS is the data rate in bits per second, and T is the duration of the test to be run in seconds.

The industry-standard bit error rate confidence level required is 95% or higher, according to Xilinx and Keysight Technologies [38] [39]. According to Viavi Solutions, an acceptable bit error rate for fiber optic data transmission is 1×10^{-12} [44]. For example, given the data rate of 10 Gbps, a specified BER of 1×10^{-12}, running a five-minute BER test, assuming 0 errors, will give a 95% confidence level. If the same test were repeated, but with a specified BER of 1×10^{-13} and assuming 0 errors, that test would take 50 minutes to get a 95% confidence level. Each error received increases the number of bits required to reach a given confidence level, as illustrated by the Figure 1 [40] .

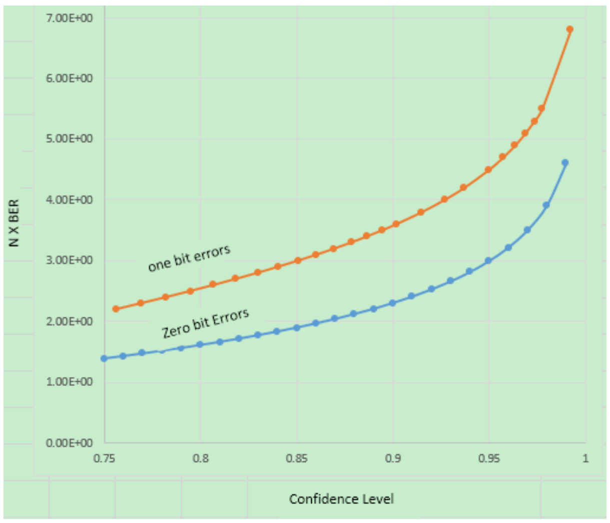


Figure 1: Confidence Level Graph [38].

The confidence level will control the duration, and the pass/fail aspects of a bit error rate test completed on the Mini BERT. A 95% confidence level at 10 Gbps can be achieved in 300 seconds or 5 minutes with zero bit errors. With one additional, bit error this time is raised to approximately 450 seconds or 7.5 minutes, according to Figure 1. After a sufficient amount of time, if 95% confidence is not achieved, then the test should be considered a failure.

1.3 A Retail BERT

Standard BERTs are designed to be used in an electronics laboratory and are generally about the size of a medium desktop printer. Because of their size, weight and cost standard BERTs, are not suitable for use in the field. The serial BERT sold by Keysight Technologies, model number N4962A serial BERT, 12.5 Gbps, costs $37,996.00. The N4962A is shown in Figure 2.

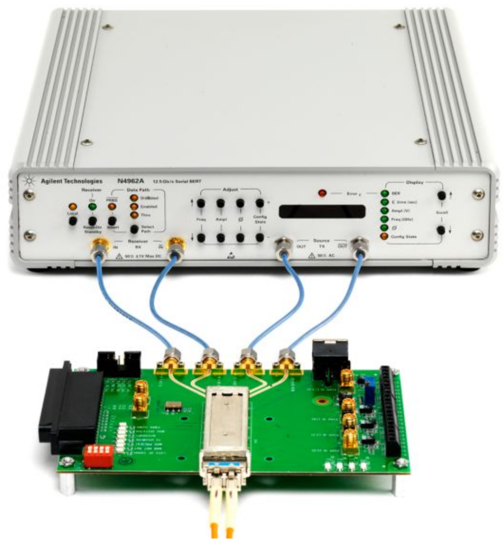


Figure 2: N4962A Serial BERT 12.5 Gbps [41].

This model can operate with signals ranging from 0.5 to 12.5 Gbps. It has multiple clocks and pattern triggers, and differential or single-ended inputs and outputs. Available options are calibration plans, extended warranties, and software bundles. These additions can cause the BERT to have a final price that is much greater than the standard price. The proposed Mini BERT design will have limited capability, with no additional options installed beyond bit error rate testing. The removal of other options is necessary to create a BERT that is as affordable as possible. The Mini BERT design will be able to create the data to be sent, send that data, receive additional different data, and generate comparison data to determine if received data are accurate. The Mini BERT design would need to keep a running tally of the number of bits received in error, the total bits received, and display this data [?].

1.4 FPGA

A BERT is required to perform several operations simultaneously: create data, send data, receive additional data, and generate comparison statistics. A CPU based computer system generally executes code one operation at a time and would not be able to keep up with the number of computations required by the Mini BERT design. Field Programmable Gate Arrays (FPGAs) are devices that implement hardware designed to handle thousands to millions of logic operations every clock cycle. An FPGA contains arrays of programmable logic blocks in a two-dimensional matrix. Each FPGA may have millions of these logic gates. FPGAs can have new functionality by reprogramming and reordering these logic gates. A large number of gates give the FPGA the ability to do a large number of relatively simple tasks at the same time, making the FPGA the practical device to use as the core of the Mini BERT design [4, 5].

1.5 Communication Protocol Layers

The Open Systems Interconnection model (OSI model) is the standard theoretical model used to describe the possible functions of a telecommunication system. The OSI model breaks down the operation of a telecommunication system into seven layers from the top layer to the bottom layer: Application Layer, Presentation Layer, Session Layer, Transport Layer, Network Layer, Data Link Layer, and the Physical layer. Most end users only interact with the application and Presentation Layers [18] [27]. The OSI model stack is shown in Figure 3.

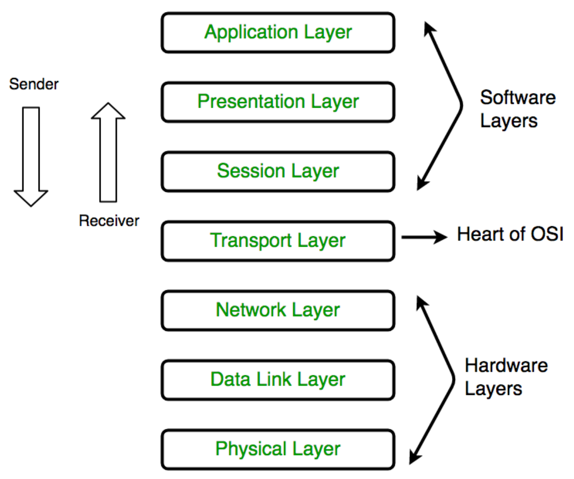


Figure 3: OSI Model Layers [18].

1.6 Communication Protocols

While most bit errors occur in transmission, or at the Physical Layer, communication protocols handle errors in the Network or Data Link Layers of the OSI model. Modern communication protocols have developed ways to detect or reduce the effects of bit errors, to the point that the average user does not realize when a bit error occurs. There are two main ways communication protocols handle bit errors. When two devices communicate using the TCP/IP (Transmission Control Protocol / Internet Protocol), a checksum system is utilized to ensure the data is transmitted and received correctly. If the receiving side checksum is correct, the receiver will transmit an acknowledgment (ACK) back to the original transmitter. This signals that the data was received correctly and that the receiver is ready for more data to be transmitted. However, if the checksum is not valid, a non-acknowledgment (NACK) will be sent instead of an ACK. The receiver will inform the sender that the data was not correct and needs to be resent. In this method, a 1-bit error can cause up to 12,176 bits to be resent, which, in the case of multiple bit errors, can combine to cause a serious strain to the communications line and appear to the user as poor service. The second method of handling bit errors, which is used by User Datagram Protocol (UDP), is to allow the application requesting the data to determine how to handle errors. Some applications using UDP create a buffer of correctly received data so that data received with errors can be deleted. If the number of errors is below a threshold, the correct data can be averaged together to decrease the importance of data lost to errors. If the number of errors is above an error threshold, the application can halt operation and request the data be resent, or halt until enough valid data is present to continue operation. In this way, bit errors are usually manifested as lag when seen in a video game, or a sudden drop in picture quality when seen in streaming video [19, 27, 28].

1.7 Ethernet

IEEE 802.3, or as it is better-known, Ethernet, is a set of protocols and standards describing the Physical Layer Hardware (PHY) and the Data Link Layer in most internet-enabled devices. The PHY is the hardware used to connect devices in a Local Area Network (LAN). The Ethernet Physical Layer includes, but is not limited to, the following sublayers: Physical Coding Sublayer (PCS), Physical Medium Attachment (PMA), and Physical Medium Dependent (PMD). An overview of the Ethernet architecture is shown in Figure 4.

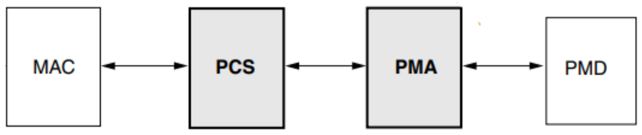


Figure 4: Overview of Ethernet Architecture [7].

The Ethernet PHY connects to the parents device Data Link Layer using a media access control (MAC). The Ethernet PHY connects to other LAN devices using a medium such as copper twisted pair cables, wireless, and fiber optic cables [7–9, 25, 35].

1.7.1 MAC

The Media Access Control (MAC) acts as the connection port for the physical aspects of the Ethernet protocol and the FPGA. The MAC has a 12-digit hexadecimal address that acts as a unique identifier for the specific device [7, 8, 25, 29].

1.7.2 PCS

When the Physical Coding Sublayer (PCS) receives data from the MAC Sublayer, if the design has enabled the encoders, then the data is encoded with the selected encoding and sent to the Physical Medium Attachment (PMA) Layer. If the design did not enable the encoders, the data would pass through the PCS unchanged. When the PCS receives data from the PMA Sublayer, if enabled, the encoders decode the data and send it to the MAC Sublayer. If the encoders are not enabled, the data will pass through unchanged to the MAC Sublayer [7, 8, 25, 30, 35].

1.7.3 PMA

The Physical Medium Attachment (PMA) Sublayer is responsible for serializing the data it transmits, known as Parallel In Serial Out (PISO). The PMA Sublayer is responsible for using received data to frame the data, deserialize the data, and perform clock data recovery. Deserializing the data is part of the Serial In Parallel Out (SIPO). The PISO and SIPO are known collectively as Serializer/Deserializer, (SERDES). After receiving the serialized data, the deserializer looks for a flag that indicates where the first bit of data is to begin in the frame. The flag should occur every specified number of bits, determined by the encoding scheme. The second flag found where it is supposed to be indicates that the data is in a frame. A block diagram of the PCS/PMA sublayer operations is shown in Figure 5.

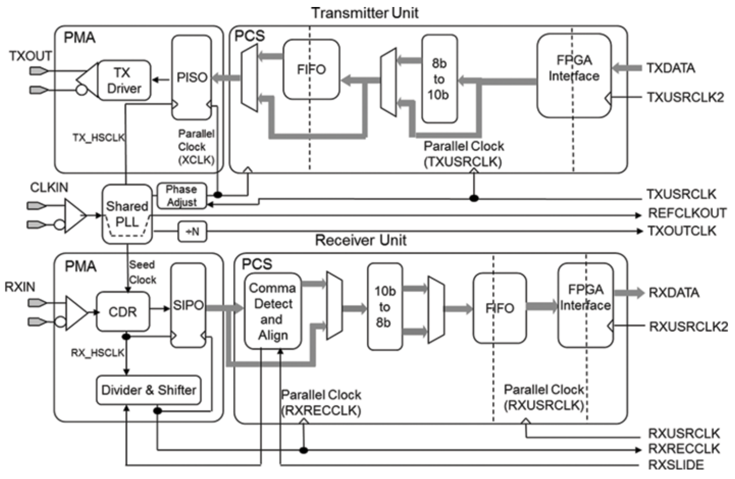


Figure 5: Overview of PCS/PMA [35].

The received data must have consistent transitions from zero to one and one to zero. Clock data recovery not only reproduces the frequency of the clock but also the phase of the clock using a Phased Locked Loop (PLL). The basic idea is that it uses the transitions from a received signal to recreate a clock with the same frequency and phase as the received data. If there are not enough transitions in the data, then this circuitry will not produce a clock of the right frequency. An encoding that has the right number of transitions in its data is known as a self-clocking signal [7, 8, 25, 35, 36]. An overview of the Clock Data Recovery operation is shown in Figure 6.

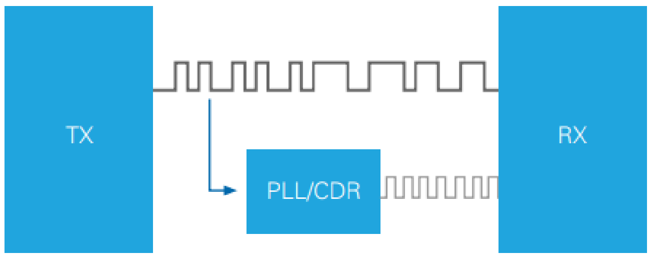


Figure 6: Overview of Clock Data Recovery [10].

1.7.4 PMD

The Physical Medium Dependent (PMD) is where the medium connects to the actual device. In the 1-Gigabit and above range, this can be a Small Formfactor Pluggable (SFP) connector.

A Small Formfactor Pluggable (SFP) connector is hot-swappable, meaning it can be unplugged and plugged back in without having to shut down the device. It has an industry-standard form factor used by many corporations. When transmitting, it is a component that will change an electrical signal into a light signal to be sent over fiber-optic lines. On the receiver side, it can convert a light signal into an electrical signal that will be sent to the PMA layer. When PMA connects to the SFP, it is connected by two sets of differential signals. Differential signals consist of a positive signal line and a negative signal line. There is one set of signals for transmitting and one set of signals for receiving. All four of these lines will have a capacitor in series. These four capacitors are designed to block any DC noise that may be present. These capacitors are referred to as

DC blocks or DC blocking capacitors. They act as a high pass filter blocking any of the lower frequencies. To avoid the signal strength being filtered, there needs to be a voltage balance on any signal transmitted. This voltage balance is known as DC balance. The DC balance is the property of having an equal number of ones and zeros. The mathematical definition of DC balance is the number of ones and zeroes, averaged over any sequence of N bits, differs by no more than M. If the absolute value of the DC balance is too high, then there are no guarantees that the data filtered and received is the correct polarity [5, 22–25, 31, 32]. A simplified block diagram of the optical SFP operation is shown in Figure 7.

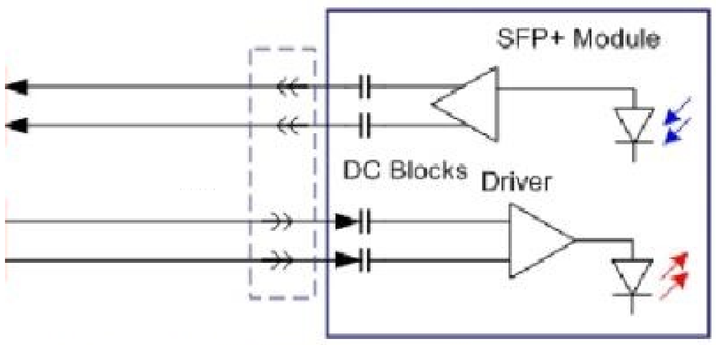


Figure 7: Simplified Block Diagram of Optical SFP [32].

1.7.5 Fiber Optic Medium

Fiber optic cables transmit data by sending light waves through a glass core that is 9 μm, 50 μm, or 62.5 μm, which measures thinner than a piece of hair at 60-80 μm. This glass core has a surrounding cladding or covering that is designed to keep the light waves contained in the center to avoid data loss. Outside the cladding are a buffer material and a jacket material that protect the cable from damage. Fiber optic cables come in two types called single-mode and multi-mode. The multi-mode has a larger diameter core, which has a lower cost and a lower maximum range. The single-mode has a smaller diameter core, which gives tighter control on the light wave, allowing for up to 50 times the signal range. Single-mode fiber optics can have up to 5 times the cost of multi-mode cables [33, 34]. Optical fiber types and inner diameter sizes shown in Figure 8.

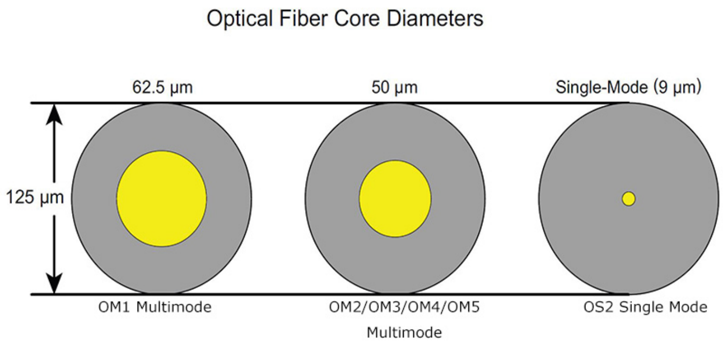


Figure 8: Optical Fiber Types [35].

1.7.6 Encoding Options

A Pseudorandom Binary Sequence (PRBS) emulates what a random binary data stream would look like in the real world but can be predictable when decoded with the proper algorithm. Two common ways to produce a PRBS are line encoding and Linear Feedback Shift Registers (LFSR).

Line encoding is when a code that helps facilitate transmission over a communication channel replaces a binary value. Common line encodings include 4b5b and 8b10b. 4b5b encoding is a line code that replaces a 4-bit value with a 5-bit value. 4b5b is a run-length limited code, meaning that the number of 0s or 1s has a limit to the consecutive 0s or 1s that can appear in the line code. 5-bit values are selected so that no more than eight 1s appear consecutively, and no more than three 0s are in a consecutive row. 4b5b is also considered a self-clocking signal. A self-clocking signal means that enough bit transitions are present for (Clock Data Recovery) CDR circuitry to produce a clock from the received data. 4b5b is considered an inefficient line coding scheme, since it adds an additional 25 percent more data to that transmitted by the user. 4b5b is not a DC balanced line code. 4b5b does not keep track of running disparity. Running disparity is when the number of 1s or 0s exceeds the opposite value. A positive running disparity is when the number of 1s exceeds the number of 0s. A negative running disparity is when the number of 0s exceeds the number of 1s. The 4 bit data, 5 bit symbol and symbol disparity for the 4b5b encoding is shown in Table 1.

Table 1: 4b5b Encoding with Disparity Values [42].

4b5b Encoding And Disparity Values					
4b Data	5b symbol	Disparity	4b Data	5b symbol	Disparity
0000	11110	+3	1000	10010	-1
0001	01001	-1	1001	10011	+1
0010	10100	-1	1010	10110	+1
0011	10101	+1	1011	10111	+3
0100	01010	-1	1100	11010	+1
0101	01011	+1	1101	11011	+3
0110	01110	+1	1110	11100	+1
0111	01111	+3	1111	11101	+3

8b10b encoding is a combination of a 5b6b and 3b4b line codes that can have a disparity of -2, 0, or +2. 8b10b is a run-length limited code with a limit of consecutive 1s or 0s being 5. It is also a self-clocking signal. Since 8b10b adds 25 percent more bits that are transmitted, 8b10b is considered an inefficient line coding scheme. When an 8-bit data word is to be encoded, the encoder breaks the data word into the five least significant bits and the three most significant bits. If the three most significant bits found in the 3b/4b table have a disparity of zero, the data encoded is from the symbol choice listed below. However, if the running disparity has a value of -2 or +2, then the symbol selected is determined on the current running disparity value. If that running disparity value is -2, then the +2 variant of the symbol will be used. The inverse is true if the running disparity value is +2, then the -2 variant of the symbol will be used. If there is a running disparity of zero, then it can be chosen at random or predetermined if -2 or +2 symbol is picked. The 3 bit data, 4 bit symbol and symbol disparity for the 3b4b encoding is shown in Table 2.

Table 2: 3b4b Encoding with Disparity Values [10].

3b4b Encoding And Disparity Values					
3b Data	4b symbol	Disparity	3b Data	4b symbol	Disparity
000	0100 or 1011	-2 or +2	100	0010 or 1101	-2 or +2
001	1001	0	101	1010	0
010	0101	0	110	0110	0
011	0011 or 1100	0	111	0001 or 1110	-2 or +2

If the five least significant bits found in the 5b6b table have a disparity of zero, the data encoded is from the symbol choice listed below. However, if the running disparity has a value of -2 or +2, then the symbol selected is determined on the current running disparity value. If that running disparity value is -2, then the +2 variant of the symbol will be used. The inverse is true if the running disparity value is +2, then the -2 variant of the symbol will be used. If there is a running disparity of zero, then it can be chosen at random or predetermined if -2 or +2 symbol is picked. The 5 bit data, 6 bit symbol and symbol disparity for the 5b6b encoding is shown in Table 3.

Table 3: 5b6b Encoding with Disparity Values [10].

5b6b Encoding And Disparity Values					
5b Data	6b symbol	Disparity	5b Data	6b symbol	Disparity
00000	100111 or 011000	+2 or -2	10000	011011 or 100100	+2 or -2
00001	011101 or 100010	+2 or -2	10001	100011	0
00010	101101 or 010010	+2 or -2	10010	010011	0
00011	110001	0	10011	110010	0
00100	110101 or 001010	+2 or -2	10100	001011	0
00101	101001	0	10101	101010	0
00110	011001	0	10110	011010	0
00111	111000 or 000111	0	10111	111010 or 000101	+2 or -2
01000	111001 or 000101	+2 or -2	11000	110011 or 001100	+2 or -2
01001	100101	0	11001	100110	0
01010	010101	0	11010	010110	0
01011	110100	0	11011	110110 or 001001	+2 or -2
01100	001101	0	11100	001110	0
01101	101100	0	11101	101110 or 010001	+2 or -2
01110	011100	0	11110	011110 or 100001	+2 or -2
01111	010111 or 101000	+2 or -2	11111	101011 or 010100	+2 or -2

The LFSR utilized in creating data scramblers is used to create the 64/66, 128/132, and 128/130 encoding. The LFSR uses what is known as a seed or starting value, XOR logic, and data shifts to create a PRBS. The PRBS created by an LFSR has a repeating pattern that is proportional to the length of the seed. The LFSR has the advantage over 8b10b encoding in that it is more efficient since using 64/66 and 128/132 adds 3.125% extra data. 128/130 encoding adds 1.5625% extra data. However, LFSR has the downside of not having strict DC balancing, and it is not predictable, like that of 8b10b [8, 10, 17, 42].

2 METHODS

The starting point for the creation of a Mini BERT is to determine what communication mediums and bandwidths need to be tested with this hardware. For this project, the focus was on creating a 10 Gigabits Per Second (Gbps), fiber optic BERT. Design aspects included operating parameters, device features, design environment, and design architectures.

2.1 Required Operating Parameters

The Mini BERT is designed based on several required operating parameters, including communication line medium, bandwidth, data aligning, pseudorandom binary sequence payload, limited test durations, and a stable power source. Data statistics that are required to be tracked include data sent, loss of signal, and error types produced.

The medium required for testing by Optical Cable Corporation is a 1310 nm SFP+ and a single-mode fiber optic cable that was 1.5m in length. The bandwidth of 10 Gbps is an Optical Cable Corporation requirement as well.

The creation of a data packet frame is required to create a pseudorandom binary sequence (PRBS) that constitutes the bulk of data to be transmitted and to align data. This PRBS was also required by Optical Cable Corporation to have a minimum test duration without repeating this pattern. The pattern duration requested was three weeks or more. In order to meet this requirement, an encoded count value was used for the creation of the PRBS. The 4b5b encoding was used for simplicity of design and to avoid any possible copyright issues with the 8b10b encoding.

At 10 Gbps the number of bits transmitted per day is found by (3):

$$\frac{10 \times 10^9\ bits}{1\ sec} \cdot \frac{60\ sec}{1\ minute} \cdot \frac{60\ minutes}{1\ hour} \cdot \frac{24\ hours}{1\ day} = 8.64 \times 10^{14}\ bits/day \tag{3}$$

The number of bits transmitted over the 21 days is found by (4):

$$\frac{8.64 \times 10^{14}\ bits}{1\ day} \cdot \frac{21\ days}{1\ period} = 1.8144 \times 10^{16}\ bits/period. \tag{4}$$

The maximum decimal value for a number of bits is found by (5):

$$M = 2^{N} - 1 \tag{5}$$

where M is the maximum decimal value, and N is the number of bits. The maximum decimal value for 54 bits is found by (6):

$$2^{54} - 1 = 1.8014 \times 10^{16} \tag{6}$$

The minimum number of bits to accommodate a 21-day count is over the max value of 54 bits. The maximum decimal value for 54 bits is found by (7):

$$2^{55} - 1 = 3.6028 \times 10^{16} \tag{7}$$

The minimum number of bits to accommodate a 21-day count is under the max value of 55 bits. So 55 bits are required for the count. Since this count will be encoded using 4b5b, the number of bits in the count value must be divisible by 4. The minimum number of bits that meet all requirements is a 56-bit count value. The 56-bit value is encoded to become a 70-bit 4b5b value. To align the data, a value that cannot be created by 4b5b encoded data, referred to as a flag, is used to signify the beginning of a frame. This flag value will repeat after a set number of encoded count values. A Frame Number Identifier will appear after the flag to signal how many frames have been transmitted and help identify the current count value. Table 4 shows the data frame structure.

Table 4: Data Frame Structure.

Data Frame Structure					
Flag for Start of Frame	Frame Number Identifier	Value#1	Value#2	Value #3	Value# 4
...	...	...	...	...	...
Value#507	Value#508	Value#509	Value#510	Value#511	Value#512

Test duration is a variable that should have a maximum worst-case scenario test duration of 50 minutes with a minimum test duration of 5 minutes, depending on the number of errors detected. A power source capable of supplying a minimum of 5 volts

DC is required for operation. The Mini BERT is intended to detect two error types: generic bit errors and a bit slip or bit add. The device is required to detect and count bit errors received. The design is required to detect a signal loss and track the number of occurrences. Signal losses include how many times the flow of data is interrupted or halted as well as sudden misalignments due to an additional or too few bits known as bit slip/add in the expected data.

2.2 Device Features

Device features of the Mini BERT include a rugged housing, and a user interface consisting of buttons to control operation, and an LCD display for displaying messages to the user. The Mini BERT is encased in a hard, plastic housing unit, smaller than a standard laptop. The Mini BERT device will include operations such as tracking data received, detecting bit errors, and logging when the data stream has been interrupted. The device will use this information to calculate and track the bit error rate. The LCD posts information regarding tracked data in a format that is easy to understand. Five buttons are included in the interface functions. The button options include start, reset, display select, inject error, and inject signal loss. Error injections and signal loss injection are added to the design to assist with testing operations.

2.3 Design Environment

The design environment includes the FPGA selection, and (Hardware Description Language) HDL choice. A Xilinx, Kintex 7 development board, was supplied by Optical Cable Corporation and used to support 10 Gbps prototyping with an FPGA. The Kintex 7 Development Board used in the design is shown in Figure 9.

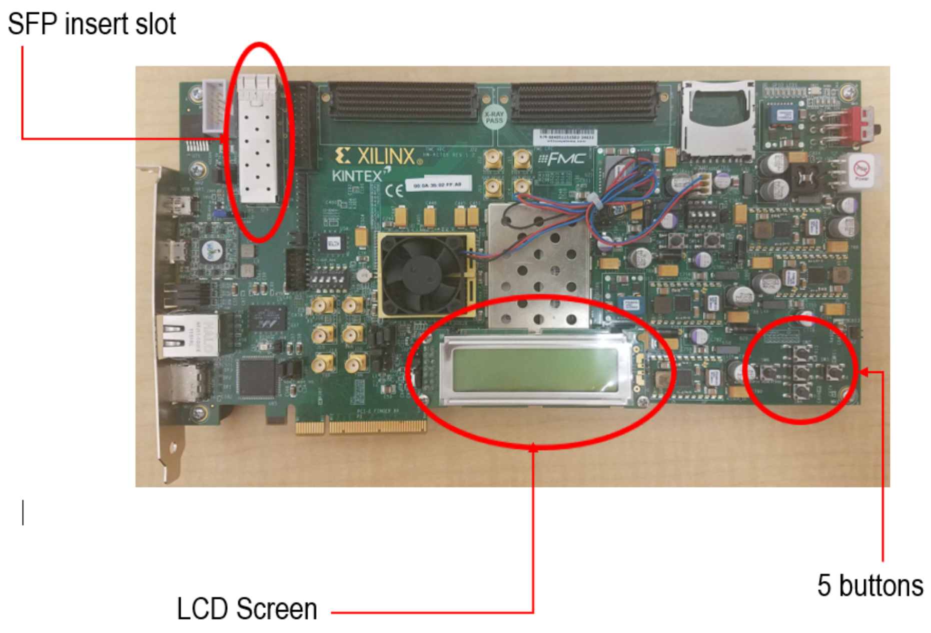


Figure 9: Kintex 7 Development Board used in design.

The software package used by Xilinx for HDL development is the Vivado Software Suite. The version of Vivado used is Vivado HLx 2018.3 [3]. The license for Vivado is a per-computer license and a per-user license. Therefore, a computer running a purchased version of Vivado was required for the project.

Vivado can support VHDL and Verilog. Either language would function in the project creation; however, the Verilog language was chosen because it is more similar in syntax to C, making Verilog more familiar and more comfortable to learn for users of the C programming language.

Dr. Paul Yanik and the Optical Cable Corporation will be the future custodians of all HDL code used in the design of the Mini BERT. HDL code is not included to maintain the Optical Cable Corporations requested confidentiality.

2.4 Design Architectures

The design architecture is shown as block diagrams to illustrate the logic of the device. The block diagram in Figure 10 shows the logic flow of the architecture of the Mini BERT design. Figure 10 shows the functional split between the transmission and reception of data. As with other programming languages, a project often needs to be broken down into smaller and more easily managed pieces called modules. Each block in the block diagram may be composed of one or more modules. The final design of the Mini Bert includes a module for generating 64-bits of encoded data each clock cycle, at 156.25 MHz, that is 10 Gbps. It includes a module with the capability of intentionally injecting bit errors (one at a time), has an optical transmitter and receiver, and analysis tools to recreate expected data to compare with received data, as well as an analysis tool to display bit error statistics.

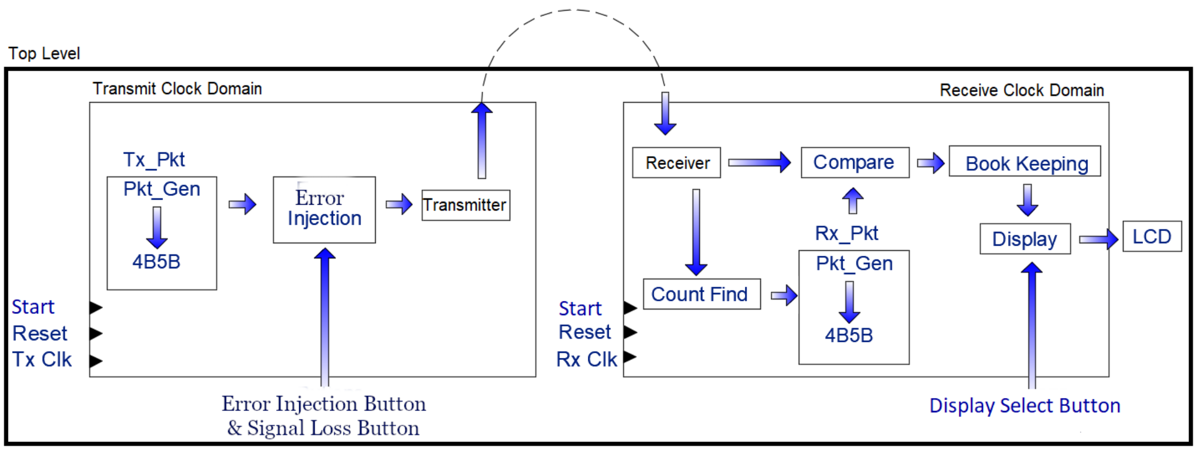


Figure 10: Block Diagram for Final Design.

To make the development on the Mini BERT easier to verify and test the design was split into two separate designs: the Fiber-Optic Transceiver Design and the Fiberless design. The logic flow minus the transceiver comprises the Fiberless Design. The Fiberless Design is purely logic-based. It does not have any exterior hardware added to it other than the LCD and the control buttons. The Fiberless design does not use the SFP or the fiber optic cables to transmit data. No data is transmitted over fiber with the Fiberless Design. The Fiberless design was created as a way to test the FPGA logic that would be needed by the Mini BERT. The block diagram in Figure 11 shows the logic flow of the Fiberless Design.

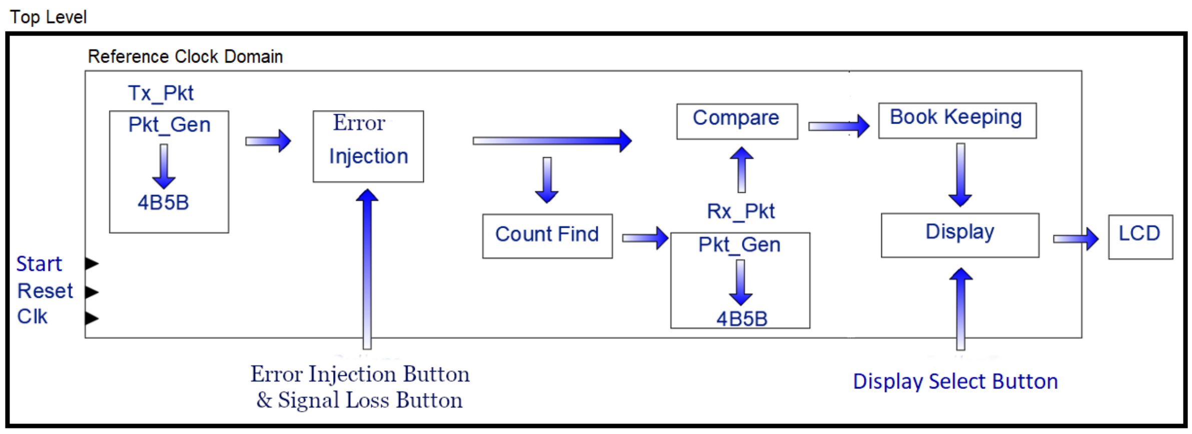


Figure 11: Block Diagram for Fiberless Design.

The Fiber-Optic Transceiver Design was generated using the Vivado transceiver wizard. The transceiver consists of a pattern generator/checker and the fiber-optic transmitter and receiver. The block diagram in Figure 12 shows the logic flow of the fiber-optic transceiver design.

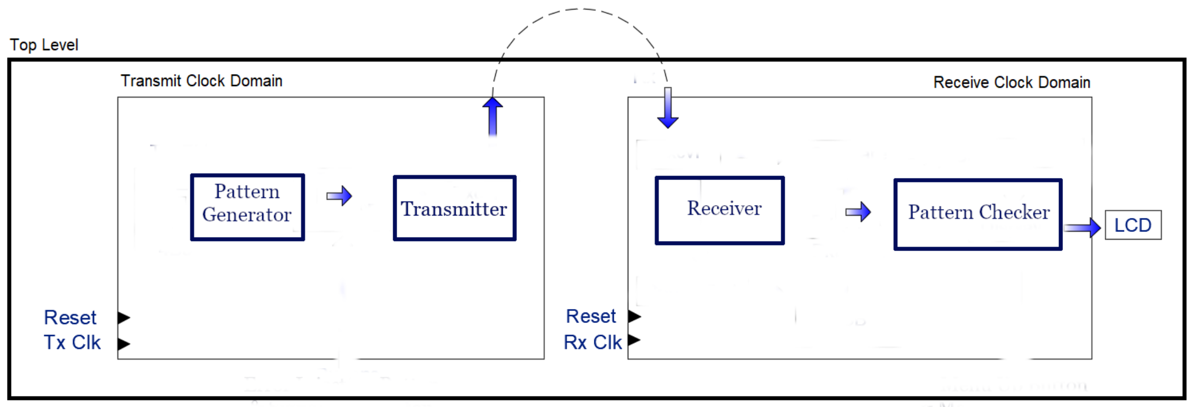


Figure 12: Block Diagram of Transceiver Design.

The fiberless design along with the fiberoptic transceiver comprise the Mini BERT design.

2.5 Fiberless Design

This section describes the HDL modules that comprise the Fiberless Design. The HDL modules include Top Level, Packet Design, Convert 4b5b, Mass Convert 4b5b, Alignment, Count Find, Invert 4b5b, Mass Invert 4b5b, Book Keeping, Error Injection

Buttons Control, Display, Binary Code Decimal, Bit Division, Fraction to Binary Coded Data, LCD Control, and LCD Button Control. Each module is crucial to the successful inner workings of the Mini BERT.

2.5.1 Top Level

The Top Level instantiates all modules and their corresponding inputs and outputs. The Top Level module is also where the hardware that the buttons control is instantiated and assigned logic. These buttons are for the user to control the outputs and reset of the Mini BERT. The Mini BERT buttons and their functions are outlined in Table 5.

Table 5: Mini BERT Buttons.

Mini BERT Buttons	
Reset button	Reset all functions and values to the starting values.
Start button	Releases the idle operation state to begin operation.
Display select button	Controls the information displayed on the LCD by scrolling the possible information values.
Error injection button	Manually inserts a single bit error.
Signal loss button	Simulates a disruption to the data stream and induces a signal loss.

The reference clock originates from a clocking buffer instantiated in the Top Level. The reference clock supplies clocking information to all the modules in the Fiberless Design.

2.5.2 Packet Design

The Packet Design Module creates data that is transmitted over the fiber optic line. It also creates the data for comparison against the data received. The 70-bit encoded value created uses a 56-bit count value as the base. However, the Mini BERT sends 64-bits at a time. For the transmitted data, the 56-bit value is a count that starts at zero, when released from the idle operation state with the start button. The 70-bit data is manipulated in a way that can be sent 64-bits continuously. The data created is accomplished by using a frame packet. The frame starts with a 64-bit flag, which acts as a unique identifier for the other modules to use later. This flag will consist of the 14-bit value 10001100011000, which is a value that cannot occur naturally in 4b5b encoding. This 14-bit value assists with frame alignment. The next 50-bits are the first 50-bits, of the current 70-bit encoded count value. The following data sent will be the first 64-bits of the first 70-bit encoded count value of the frame. The next data transmitted is the last 6-bits of the first value and the first 58-bits of the second 70-bit count value. Table 6 shows how a data frame begins.

Table 6: Data Frame Start.

Data Frame Start				
Start flag + Value1[0:49]	Value1[0:63]	Value1[64:69]+ Value2[0:57]	Value2[58:69]+ Value3[0:51]	Value3[52:69]+ Value4[0:45]

This pattern repeats for a total of eleven iterations, at which point there are 66-bits leftover. A transmission will occur that does not increase the count but will reduce that number of leftover bits. This count pattern will repeat eleven times until 68-bits are leftover. Another send will occur without a count. These steps will repeat, but this time only ten times with a leftover being 64-bits. This creates a pattern that, for every thirty-two counts, data transmits thirty-five times. Table 7 shows this data rotation for the bulk of the data in the Frame Structure.

Table 7: Rotation of Data Frame Structure.

Rotation of Data Frame Structure			
Value1[0:63]	Value1[64:69]+ Value2[0:57]	Value2[58:69]+ Value3[0:51]	Value3[52:69]+ Value4[0:45]
Value4[46:69]+ Value5[0:39]	Value5[40:69]+ Value6[0:33]	Value6[34:69]+ Value7[0:27]	Value7[28:69]+ Value8[0:21]
Value8[22:69]+ Value9[0:15]	Value9[16:69]+ Value10[0:09]	Value10[10:69]+ Value11[0:3]	Value11[4:67]
Value11[68:69]+ Value12[0:61]	Value12[62:69]+ Value13[0:55]	Value13[56:69]+ Value14[0:49]	Value14[50:69]+ Value15[0:43]
Value15[44:69]+ Value16[0:37]	Value16[38:69]+ Value17[0:31]	Value17[32:69]+ Value18[0:25]	Value18[26:69]+ Value19[0:19]
Value19[20:69]+ Value20[0:13]	Value20[14:69]+ Value21[0:7]	Value21[8:69]+ Value22[0:1]	Value22[2:65]
Value22[66:69]+ Value23[0:59]	Value23[60:69]+ Value24[0:53]	Value24[54:69]+ Value25[0:47]	Value25[48:69]+ Value26[0:41]
Value26[42:69]+ Value27[0:35]	Value27[36:69]+ Value28[0:29]	Value28[30:69]+ Value29[0:23]	Value29[24:69]+ Value30[0:17]
Value30[18:69]+ Value31[0:11]	Value31[12:69]+ Value32[0:5]	Value32[6:69]	...

This count pattern repeats multiple times without the flag portion being resent, creating a customizable packet size of multiples of thirty-two 70-bit values.

2.5.3 Convert 4b5b

The Convert 4b5b Module is used to covert a 4-bit value into a 5-bit encoded value. The Convert 4b5b module has a 4-bit value as an input. Convert 4b5b assigns the 5-bit encoded value output from the 4-bit value input. Convert 4b5b is called multiple times by the Mass Convert 4b5b module (see Section 2.5.4).

2.5.4 Mass Convert 4b5b

The Mass Convert 4b5b Module is used to covert a 56-bit value into a 70-bit encoded value. The Mass Convert 4b5b module receives an inputted 56-bit value. The 56-bit value divided into fourteen, 4-bit values, is sent to a copy of the Convert 4b5b module see Section 2.5.3. The fourteen returned 5-bit values are then appended together in the consecutive order, and the output is transmitted to the Packet Design module as a 70-bit value. Mass Convert 4b5b is called by the Packet Design module (see Section 2.5.2).

2.5.5 Alignment

When data is received from the transceiver, the alignment of that data cannot be guaranteed. Therefore, a method of aligning the data is needed for the data to be of use to the system. The current 64-bit data is appended to the previous data, to create a 128-bit value. The first if statement is created, so that from each bit location in the first 64 bits, that bit and the next thirteen bits will be compared against the 14-bit flag value of 10001100011000. If the values are equal, then a value corresponding to the bit location, where the flag resides, will be assigned to a selection variable used in the next nested if statement of alignment. If none of the values are equal, then the value of the selection remains at its previous value. The second nested if statement uses the 128-bit value from earlier. However, the selection bit assigned above plus the next 63-bits will now be the starting bit as an output for the module.

2.5.6 Count Find

The method for comparing expected received data to actual received data is a required first step. The next step in recreating this data is to determine what is the current 56-bit count value. The Count Find module does this by first looking for the flag value 10001100011000, from the Alignment module. When the flag value is located, a triple-check begins. Looking in the 64 bits that the flag value resides, if the value of the last 50-bits is equal to the first 50-bits, of the following data and the next data after that, then the count should be without errors. When this triple check passes, a send count flag triggers the count value to be updated; otherwise, the triple check fails and waits for the next flag value to appear. If the count flag was enabled the 64-bit data of the next cycle after the flag data is appended to the first 6-bits of the next data. This recreates the expected 70-bit encoded data and sends it as input to the Mass Invert4b5b module. If the 56-bit output value from Mass Invert 4b5b is valid, then 4b5b encoding had no errors. The 56-bit value will transmit to a receive-side version of the Packet Design module. The 56-bit value will act as a seed value to recreate the expected data produced for comparison against the received data.

2.5.7 Invert 4b5b

The Invert 4b5b Module is used to covert a 5-bit encoded value into a 4-bit value. The Invert 4b5b module has a 5-bit encoded value as an input. The 5-bit encoded value assigns the 4-bit value that will be outputted. If the 5-bit encoded value does not match a valid 4b5b encoding, the value returned will be equal to zero, and the data flagged as invalid. The Invert 4b5b module is called multiple times by the Mass Invert 4b5b module (see section 2.5.8).

2.5.8 Mass Invert 4b5b

The Mass Invert 4b5b Module is used to covert a 70-bit encoded value into a 56-bit value. The Mass Invert 4b5b module receives an inputted 70-bit encoded value. The 70-bit encoded value divides into fourteen, 5-bit encoded values, with each of these values sent to a copy of the Invert 4b5b module see Section 2.5.7. The returned 4-bit values are then aligned together, in the same order, and the output is sent to the Packet Design module as a 56-bit value, along with a valid flag to indicate if no error was present in the data. The Mass Invert 4b5b module is called one time by the Count Find module (see Section 2.5.6).

2.5.9 Book Keeping

The Bookkeeping module has two inputs: the data received after alignment and the output from the receiving side version of the Packet Design Module. These two 64-bit values are compared, bit to bit, by XORing the values of each corresponding bit location. If the values match, then the value of the XOR should be zero, and if they do not match, there should be a 1, indicating an error has occurred. To get a total count, each bit is added together. Since all sixty-four values cannot be added all at once, without causing timing issues, a rolling count ensures that only two calculations occur at a time. For example, bit 1 plus bit 33 and bit 2 plus bit 34 continues until all 32 additions are calculated. When only one value remains, the cycle described above completes. Following the completion of the second cycle, there will be sixteen values calculated. The third cycle continues with eight values calculated. The fourth cycle completion reveals four values calculated. In the fifth cycle, there will be two values calculate. And finally, on the sixth cycle, there will be one value calculated. Once the final value representing the current errors is determined, it is used to track data. The first check created gauges stability. If we have received data with no errors, a stability count value increases by one. If the device has a stability count value of at least 10, then stability is achieved, and total bits sent are tracked and counted. If the errors received in one cycle are less than 10, then the errors are added to a total error count. However, if the error count is greater than 10, then the stability has been lost. A signal loss should be added to the total signal losses count. The stability count returns to zero, and the total bit count is stopped until stability is regained. The Bookkeeping Module will have three outputs: signal losses, error count, and total bits sent.

2.5.10 Error Injection Buttons Control

There are two buttons used in Error Injection Buttons Control: the signal loss button and the bit error injection button. With each button release, it produces only one error of the corresponding type.

2.5.11 Display

The display consists of four modules designed to facilitate displaying the numerical values in a form easier for a human user to interpret the information. The four modules consist of Binary Coded Decimal, Bit Division, Fraction to Binary Coded Decimal, and LCD Control.

2.5.12 Binary Coded Decimal

The Binary Coded Decimal (BCD) module converts the 64-bit input Value into a binary encoded decimal value that will need to be displayed to the LCD screen, BCD is simpler for the LCD to display. This module has output data to display signal losses, total bit errors, and total bits received. The Binary Coded Decimal module is a modified version of the Double Dabble algorithm. This method uses a mixture of left bit shifting and adding 3 to the value, to convert pure binary into a code in which the first 4 bits represent the one's place, and the next 4 bits represent the tens place and so on. Once the conversion is complete, the original 64-bit binary value transforms into an 80-bit BCD value. A series of if statements convert the 80-bit BCD value into a 60-bit output value that is formatted to display the most significant digits on the LCD. This formatting removes all but the six most significant digits and adds exponent data for the LCD [43]. Figure 13 shows an example value converted to BCD.

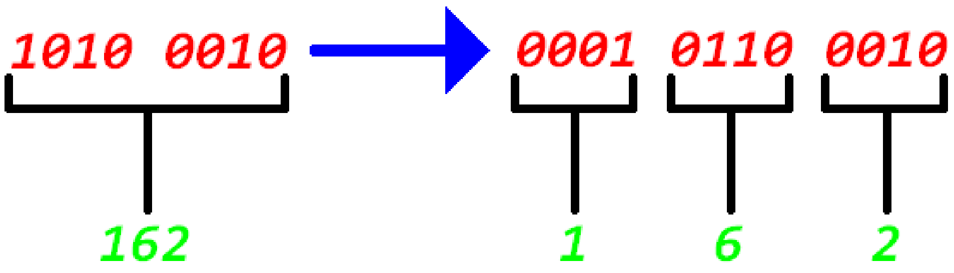


Figure 13: Example value converted to BCD [43].

2.5.13 Bit Division

The division function that comes standard with the Vivado Suite is limited to a maximum of 32-bit values. The standard division functions found in Vivado would not give the accuracy desired when calculating bit error rates. A method of dividing a 64-bit value was created for this design. It was possible to get values as low as 1×10^{-21} by using this method.

2.5.14 Fraction to Binary Coded Data

The output from the Bit Division module is always a value of less than one. Since this value is less than one, a modified way to convert the Binary Coded Data is needed. While this module is similar to the BCD module, it works as an inverse of the BCD. The original 64-bit binary value is transformed into an 80-bit BCD value. A series of if statements convert the 80-bit BCD value into a 60-bit output value that is formatted to display the most significant digits on the LCD. This formatting removes all but the six most significant digits and adds the negative exponent data for the LCD. This module will output data used to display the bit error rate.

2.5.15 LCD Control

The LCD Control module inputs data for signal losses, bit errors, and total bits received from the Binary Coded Decimal modules, as well as the bit error rate from the Fraction to BCD module. The LCD exhibits one of these values at a time, depending on the current selection. The LCD uses a slower clock than the clock of an FPGA. Therefore, a way to slow the clock was needed. A 25-bit value is created to be used as a count. This count controls a case statement that will dictate when letters or digits are shown on the LCD. This case statement will only act when the bits above bit 17 change on the count or every 262,144 cycles. This should cause that logic to trigger only 596 times per second. Each time the case statement is incremented, a command is given to send the data to the LCD one time.

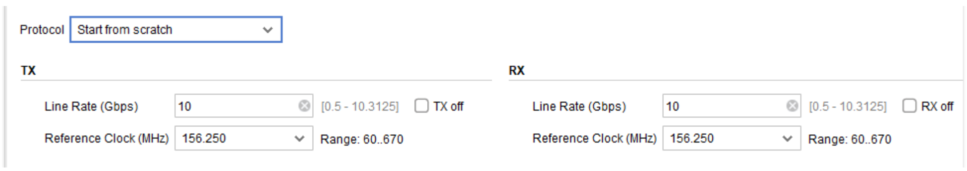
2.5.16 LCD Button Control

This module helps the user control the output displayed on the LCD via a control button. This button has four possible states that can be displayed. With each button release, the next state is selected and displayed.

This concludes the modules in the fiberless Mini BERT design. The results of this design will be reviewed in the results section.

2.6 Fiber Optic Transceiver Design

Transceiver Design for the Vivado Suite is handled by a software wizard that creates the HDL code for controlling the transceiver. The transceiver wizard has several different screens where you input the settings that correspond to the transceiver design desired. Figure 14 shows line rate is set at 10 Gbps, the reference clock is set at 156.25 MHz.



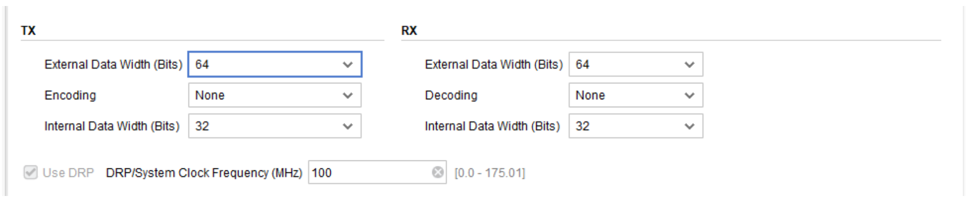
Protocol: Start from scratch

TX
Line Rate (Gbps): 10 [0.5 - 10.3125] ☐ TX off
Reference Clock (MHz): 156.250 Range: 60..670

RX
Line Rate (Gbps): 10 [0.5 - 10.3125] ☐ RX off
Reference Clock (MHz): 156.250 Range: 60..670

Figure 14: Example Line Rate and Reference Clock Values [8].

Figure 15 shows the data path width is set to 64-bits and the DRP clock is set to 100MHz.



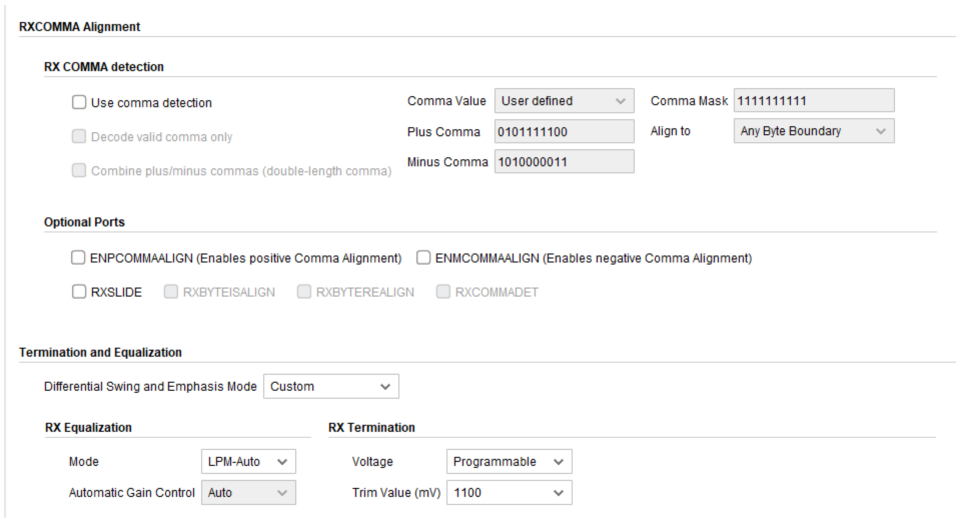
TX
External Data Width (Bits): 64
Encoding: None
Internal Data Width (Bits): 32

RX
External Data Width (Bits): 64
Decoding: None
Internal Data Width (Bits): 32

✓ Use DRP DRP/System Clock Frequency (MHz): 100 [0.0 - 175.01]

Figure 15: Example External Data Width [8].

Figure 16 shows the use of comma detection is not checked, and RX Equalization has LPM-Auto mode selected.


RXCOMMA Alignment

RX COMMA detection

☐ Use comma detection
☐ Decode valid comma only
☐ Combine plus/minus commas (double-length comma)

Comma Value: User defined
Plus Comma: 0101111100
Minus Comma: 1010000011
Comma Mask: 1111111111
Align to: Any Byte Boundary

Optional Ports

☐ ENPCOMMAALIGN (Enables positive Comma Alignment)
☐ ENMCOMMAALIGN (Enables negative Comma Alignment)
☐ RXSLIDE ☐ RXBYTEISALIGN ☐ RXBYTEREALIGN ☐ RXCOMMADET

Termination and Equalization

Differential Swing and Emphasis Mode: Custom

RX Equalization

Mode: LPM-Auto
Automatic Gain Control: Auto

RX Termination

Voltage: Programmable
Trim Value (mV): 1100

Figure 16: Example Commas and LPM Settings [8].

Once the wizard is completed, an option to open an IP configuration of the transceiver is available. From the IP configuration, an example transceiver design is opened in a new window. This design will have a Top-Level automatically created, a pattern generator created, and a transceiver design created using the settings input into the wizard. This design will instantiate two clocks of importance: the transmit clock and receive clock. In addition to these clocks, a clocking wizard needs to be used to instantiate another clock with the same frequency as the transmit clock. This new clock drives the SMA differential pins needed by design. There is a Data Valid logic signal from the transceiver design that outputs a high when data is being received and a low when data is not being received. This signal was connected to an LED.

2.7 Initial Mini BERT Design

The Initial Mini BERT Design was created by taking the Fiber-Optic Transceiver Design and removing the Pattern Generator and Pattern Checker. Then all the modules from the Fiberless design were brought into the remaining Fiber-Optic Transceiver Design. The output of the Error Injection Module was connected with Transmit Port of the Transceiver, and the Receiver Port of the Transceiver was connected with the input of the Alignment Module. Also, the Transmit Clock of the Transceiver was used as the System Clock for Packet Generation on the transmit side and for Error Injection. All other modules used the receive side clock as the system clock.

2.8 Modified Fiber-Optic Transceiver Design

A way to verify the integrity of the Fiber-Optic Transceiver Design was needed that was more consistent and reliable than an LED. The Vivado transceiver design has a built-in Pattern Generator that is directly connected to the Transmit Port. The pattern generator repeatedly reads from gt_rom_init_tx.dat a text file and transmits the sixteen 64-bit values are shown in Table 8.

Table 8: Built-in Transceiver Pattern [8].

64-bit Pattern Value
0706050403020100
0e0d0c0b0a09087c
161514131211100f
1e1d1c1b1a191817
262524232221201f
2e2d2c2b2a292827
363534333231302f
3e3d3c3b3a393837
464544434241403f
4e4d4c4b4a494847
565554535251504f
5e5d5c5b5a595857
666564636261605f
6e6d6c6b6a696867
767574737271706f
7e7d7c7b7a797877

To verify that the value transmitted is the same as the value that is received, a 16 value if statement compares the value received to all the possible 16 values sent. If the received value is not equal to any of these 16 possible values transmitted, then the error count would increase by one. If the received value is equal to one of the 16 values transmitted, then the error count would not increase at all.

The modules from BCD and LCD Controls were also copied from the Fiberless Design and the error count value inputted through the BCD Module. Doing this allows the error count value to display on the LCD for easy monitoring.

2.9 Modified Mini BERT Design

The Modified Mini BERT Design was created by taking the Initial Mini BERT design and sending an alternation of 64-bits of 4b5b encoded data and 64-bits of hexadecimal 5s. With the inherent dc imbalance with the 4b5b encoding. This change was implemented to reduce the average dc imbalance. The Packet Creation Module was only enabled for half the clock cycles. This change slowed the operation of the Packet Design Modules in half so that for two cycles, the same data would be represented. Since Packet Creation is used for data being transmitted and data being compared, this change would affect both the received data and the compared data. A function was created on the top level to override the output from the Packet Design, every other cycle. This override created a data stream that was encoded 4b5b for half the signal and alternating 1's and 0's for the other half. This function had a transmit side version and a received side version.

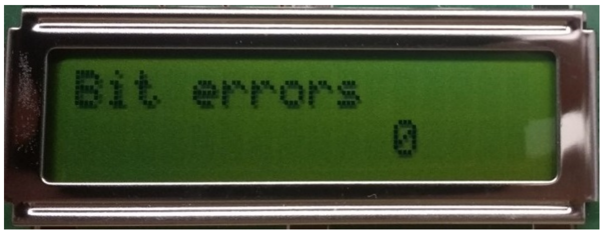
3 RESULTS

The following results of design validation are for the Fiberless Design, Fiber Optic Transceiver Design, Initial Mini BERT Design, Modified Fiber Optic Transceiver Design, and the Modified Mini BERT Design.

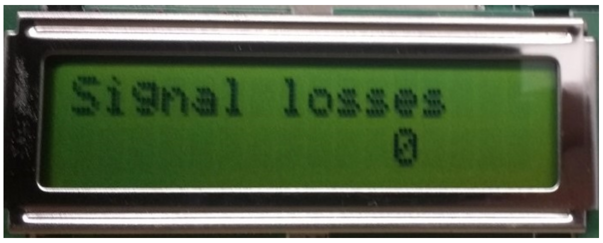
3.1 Fiberless Design Testing

While the Fiberless design was in operation at a simulated 10 Gbps, it had a 0-bit error rate unless an error was directly injected. The Fiberless design operated with a 0-bit error rate and 0 signal losses. The results shown are for the Fiberless Design operating for 5 minutes (300 seconds) when no errors or signal losses are induced manually. Figure 17a shows that the display is reading 0 BER after the Fiberless Design has begun service, Figure 17b shows the display is reading 0-bit errors. Figure 17c shows the display is reading 0 signal losses during service.

(a) Display of 0 BER.



(b) Display of 0 Errors.



(c) Display of 0 Signal Losses.

Figure 17: The results for the Fiberless Design testing.

The tests for the Fiberless Design were three identical error injection button tests and three identical signal loss button tests. A passing result for these tests was when the LCD screen displayed an increase in the expected value, and the failing result was when no increase occurred. The results for three identical tests of the Fiberless Design are shown in Table 9

Table 9: Tests performed on Fiberless Design.

Tests performed on Fiberless Design.			
Tests performed	Pass	Fail	Complication
Error injection Test 1	✓		None
Error injection Test 2	✓		None
Error injection Test 3	✓		None
Signal loss injection Test 1	✓		None
Signal loss injection Test 2	✓		None
Signal loss injection Test 3	✓		None

The error rate and signal loss numbers are a bit misleading. With no fiber optic cable to physically disconnect to show signal losses and errors, the error injection button and signal loss button had to be used to show errors manually. Each press of the error injection button or the signal loss button would increase the current value for each value, respectively. So as long as an error injection button was not pressed, the Fiberless design could operate indefinitely with 0 errors.

3.2 Fiber Optic Transceiver Design Testing

The primary purpose of the Fiber Optic Transceiver Design was to achieve a functioning Transceiver Design that operated at 10 Gbps. While the design had limited feedback, the LED light stayed on consistently. However, the biggest problem faced was that there was no visual confirmation if the LED was steadily on or if the LED was losing signal multiple times a second faster than the human eye could detect. The tests performed for the Fiber Optic Transceiver Design consisted of unplugging the fiber-optic cable for 5 seconds and verifying if the LED light turned off, then reconnecting the cable and verifying the LED turned back on. A passing result for this test was considered when the LED turns off during disconnection and illuminates while connected. The failing result was considered when the LED stayed illuminated while disconnected or failing to illuminate while connected. The results for three identical Fiber Optic Transceiver Design tests are shown in Table 10.

Table 10: Tests performed on Fiber Optic Transceiver Design.

Tests performed on Fiber Optic Transceiver Design.			
Tests performed	Pass	Fail	Complication
Fiber Optic Disconnection Test 1	✓		None
Fiber Optic Disconnection Test 2	✓		None
Fiber Optic Disconnection Test 3	✓		None

The limited but promising results supported the decision to move forward with the Initial Mini BERT Design.

3.3 Initial Mini BERT Design Testing

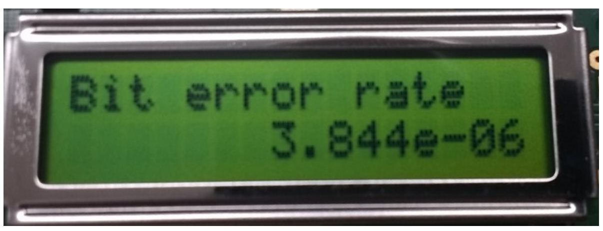
The Fiberless Design was merged into the Fiber Optic Transceiver Design to create the Initial Mini BERT Design. The tests performed for the Initial Mini BERT Design were the error injection button test, the signal loss button test, and unplugging the fiber-optic cable. A passing result for the error injection button test and the signal loss button test was when the LCD screen displayed an increase in the expected value, and the failing result was when no increase occurred. A passing result for unplugging the fiber-optic cable is if a noticeable increase in both bit errors and signal losses occur. Failure for this test is if no noticeable increase happens to the bit error count or signal loss count. These tests were performed three times each and the results for are shown in Table 11.

Table 11: Tests performed on Initial Mini BERT Design.

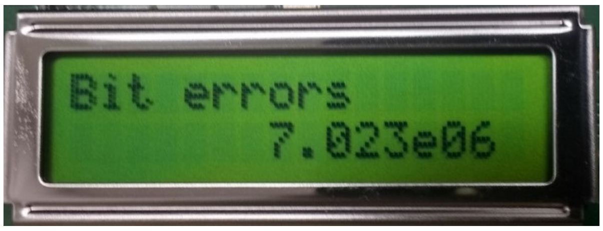
Tests performed on Initial Mini BERT Design.			
Tests performed	Pass	Fail	Complication
Error injection Test 1		✓	Not verifiable
Error injection Test 2		✓	Not verifiable
Error injection Test 3		✓	Not verifiable
Signal loss injection Test 1		✓	Not verifiable
Signal loss injection Test 2		✓	Not verifiable
Signal loss injection Test 3		✓	Not verifiable
Fiber Optic Disconnection Test 1		✓	Not verifiable
Fiber Optic Disconnection Test 2		✓	Not verifiable
Fiber Optic Disconnection Test 2		✓	Not verifiable

Because of the high amount of fundamental errors with the Initial Mini BERT Design, it was not possible to verify if errors occurred naturally or were induced by pressing the error injection button. The Initial Mini BERT design consistently took approximately thirty seconds to a minute to establish a signal. It would not go over 10 seconds without having a signal loss. The bit error rate for this design was never better than 1×10^{-7}. Bit errors and signal losses came in bursts and not at any patterned rate or expected rate. The bit error count often exceeded 10,000 per minute with hundreds of signal losses. A high BER, shown on the display in Figure 18a, was attained without the error injection button being pressed. The bit error count is high as well in Figure 18b, without the error injection button being used. Without the signal loss button being utilized, a high signal loss count is displayed in Figure 18c.

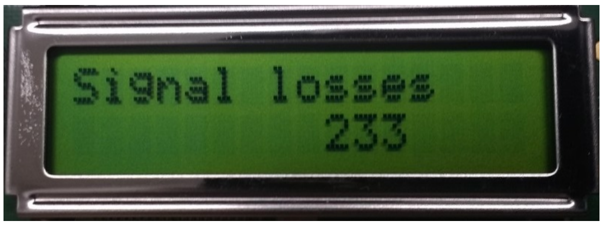
Given these poor results, a long-term test was not attempted. Instead, it was decided that creating a design that would test if the Fiber Optic Transceiver Design was receiving the transmitted data without errors would be a better test.



(a) High BER for Initial Mini BERT Design.



(b) High Error Count for Initial Mini BERT Design.



(c) High Signal Losses for Initial Mini BERT Design.

Figure 18: The results for the Initial Mini BERT Design.

3.4 Modified Fiber Optic Transceiver Design Testing

The primary purpose of the Modified Fiber Optic Transceiver Design was to test to see if the Transceiver worked correctly. The Transceiver transmitted the built-in Transceiver Pattern from Table 8 in a repeating pattern. The Modified Fiber Optic Transceiver Design only had one variable that was tracked for data purposes. The variable was an error count for when 1 of the 16 possible values sent did not equal the value received.

Whenever this design started, it would read 18 errors and then stay on 18 errors. It is believed that the 18 errors were the difference between when the device startup occurred, and when FPGA internal logic received the first data. The Transceiver took 18 cycles to send and receive data that was able to make a comparison that did not result in an error. Figure 19 shows the 18 errors created by the Modified Fiber Optic Transceiver Design.

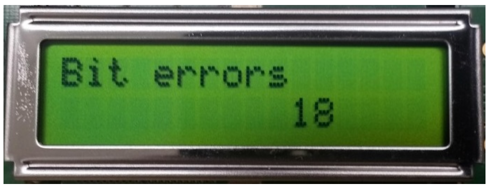


Figure 19: Display of 18 Errors on Modified Fiber Optic Transceiver Design.

The only way to inject errors with this design was to disconnect the fiber optic cable. The tests performed for the Modified Fiber Optic Transceiver Design were unplugging the fiber-optic cable. A passing result for this test is if a noticeable and consistent increase in bit errors occurs. Failure for this test is if no noticeable increase happens to the bit error count. The results for three of these tests are shown in Table 12.

Table 12: Tests performed on Modified Fiber Optic Transceiver Design.

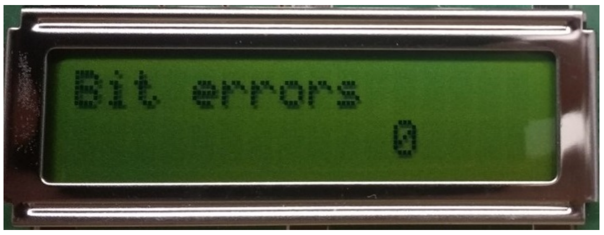
Tests performed on Modified Fiber Optic Transceiver Design.			
Tests performed	Pass	Fail	Complication
Fiber Optic Disconnection Test 1	✓		None
Fiber Optic Disconnection Test 2	✓		None
Fiber Optic Disconnection Test 2	✓		None

For each second the cable was disconnected, the error value would jump by approximately 160×10^6. Since the operating clock was 156.25 MHz, this was within expectation. This led to the belief that the Fiber Optic Transceiver Design was working correctly without errors.

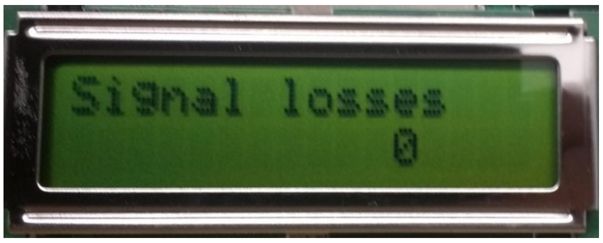
3.5 Modified Mini BERT Design Testing

While the Modified Mini BERT Design was in operation, the Modified Mini BERT Design had a 0-bit error rate. It operated with a 0-bit error rate and 0 signal losses. At 10 Gbps, 8.64×10^{14} bits are transmitted across the design daily. The Modified Fiber Design was able to receive 31.768×10^{15} bits with 0-bit errors, 0 signal losses, and 0-bit error rate. This is equivalent to 36.77 days or 5.25 weeks of error-free operation at a data rate of 10Gbps. Figure 20a shows the display reading of 31.768×10^{15} bits received. Figure 20b shows the display reading of 0 bit errors after over five weeks of service. Figure 20c shows the display reading of 0 signal losses after the same period of time.

(a) Display of 31.768×10^{15} bits received.



(b) Display of 0 Errors.



(c) Display of 0 Signal Losses.

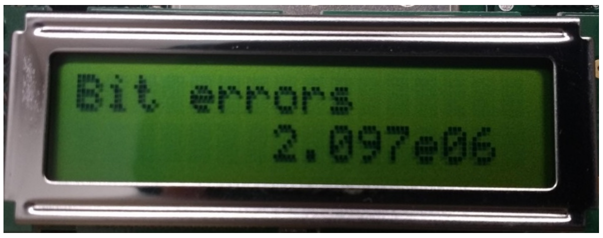
Figure 20: The results for the Modified Mini BERT Design testing.

The tests performed for the Modified Mini BERT Design were the error injection button test, the signal loss button test, and unplugging the fiber-optic cable. A passing result for the error injection button test and the signal loss button test was when the LCD screen displayed an increase in the expected value, and the failing result was when no increase occurred. A passing result for unplugging the fiber-optic cable is if a noticeable increase in both bit errors and signal losses occur. Failure for this test is if no noticeable increase happens to the bit error count or signal loss count. These tests were performed three times with the results shown in Table 13.

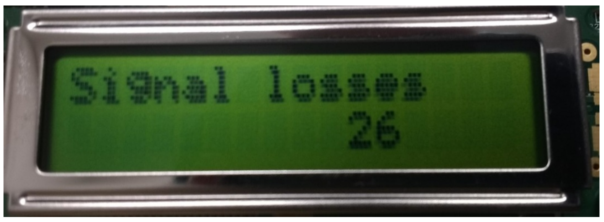
Table 13: Tests performed on Modified Mini BERT Design.

Tests performed on Modified Mini BERT Design.			
Tests performed	Pass	Fail	Complication
Error injection Test 1	✓		None
Error injection Test 2	✓		None
Error injection Test 3	✓		None
Signal loss injection Test 1	✓		None
Signal loss injection Test 2	✓		None
Signal loss injection Test 3	✓		None
Fiber Optic Disconnection Test 1	✓		None
Fiber Optic Disconnection Test 2	✓		None
Fiber Optic Disconnection Test 2	✓		None

If the error injection button was pressed, it injected a single bit error. If the signal loss button was pressed, it injected a single signal loss. If the fiber optic cable was quickly unplugged and then later reconnected, this produced millions of errors and dozens of signal losses. Since no data was being received during the disconnection, the duration of the disconnection was less critical to the accrued error count than the speed of disconnection and reconnection. While the disconnection may be quick by human standards in the digital domain, as the cable is rocked back and forth and pulled free, this instability would allow for expected errors and signal losses. Figure 21a shows the display of errors caused by disconnection. Figure 21b shows the display of signal losses caused by disconnection.



(a) Display of Errors Caused by Disconnection.



(b) Display of 0 Errors.

Figure 21: Display of Signal Losses Caused by Disconnection.

The Modified Mini BERT Design was able to operate for extended periods of time and did not exhibit unexpected behaviors.

4 DISCUSSION

The purpose of this thesis is to research the viability of creating a bit error rate tester that will be used for field applications. The current bit error rate testers on the market have a starting cost of \$37,986.00 and can become substantially higher in cost depending on the additional options desired by the consumer. The Mini BERT Prototype that was designed would retail for less than \$5000.00 with a total cost of less than \$2340.85 [2]. The retail price of the Mini BERT would be a desirable option compared to the more expensive BERTs, especially to those customers who have a limited budget.

As a first edition prototype, the Mini BERT had only two considerations desired by Optical Cable Corporation. Optical Cable Corporation specified a need for 10 Gbps line speed, which is a core part of the design and fiber optic cables. However, further into the design, the required fiber optic cables became more of an option that could have been replaced with RJ45 based SFPs. The fiber optic cables were the medium the SFPs used for transmission. If the optical SFPs were replaced with RJ45 based SFPs, the device functionality would not change. The proposed design is a specialized device with a much cheaper cost of production. A prototype design for the Mini BERT case was also created an undergraduate capstone team working in parallel. The team utilized the FPGA-based design described in this research. The team was assigned to take the Mini BERT design and create a device that was compact and had all the functionality of the Mini BERT design. Part of the Capstone team worked on the PCB layout, while another part of the team worked on the case design. The Mini BERT would be a small handheld device, unlike the large and bulky desktop BERTs. The dimensions of the case the Capstone team designed are 119 mm by 119 mm by 45 mm. The prototype Mini BERT case is shown in Figure 22.

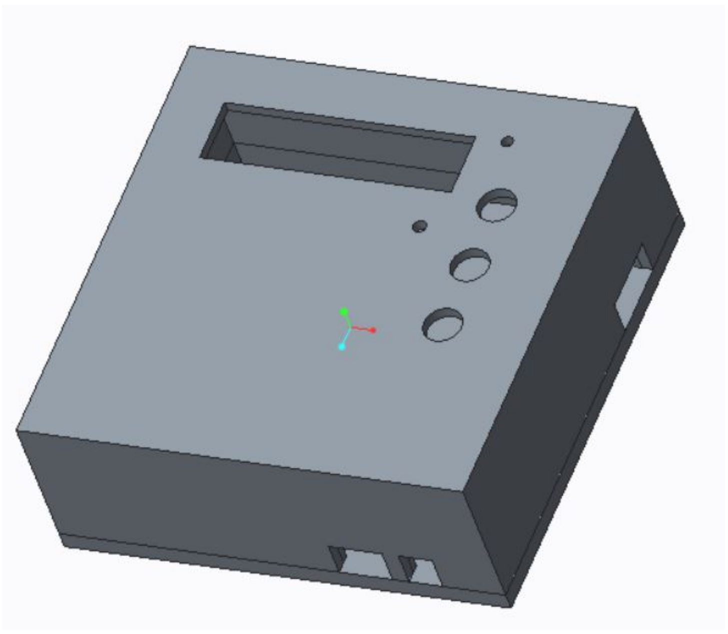


Figure 22: Mini BERT case [2].

The work on the PCB layout began with a schematic of the Kintex 7 development board, the team divided the development board into three different categories: components kept, components removed, and components of unknown importance. Components of unknown importance were researched to determine whether the components should be kept or removed. If, after researching the unknown components, the team was still uncertain which category the components belonged in, the team erred on the side of caution and left the components in the design. The Capstone team removed unused LEDs, unused voltage regulators, and RJ45 components. With strict requirements on wiring and the number of pins in such a confined space, it was not possible to make a functioning breadboard; therefore, a layout had to be designed to test functionality. There was no way to determine if the layout would work properly without creating the device. However, the price for creating the prototype of the Mini BERT design was more than was allowed in the budget for this project, so no testing was able to be performed.

With this project being a sponsored Capstone project for Optical Cable Corporation, we needed an encoding scheme that would be reasonably easy to implement. The decision was to be made between an LFSR and a line code. It was decided that an LFSR would be too complicated; therefore, the line code was chosen. The most common line code encoding platform was 8b10b, but since this was both a WCU project and a corporate-sponsored project, the legal copyright descriptions were not clearly understood at the time. However, 4b5b had no known copyright issues; it was an open-sourced encoding platform and would be straight forward to implement.

The expected bit error rate of fiber optic data lines is a 1-bit error for every 1 trillion bits of data sent. With that level of integrity, every bit that is transmitted over the line should be accounted for and counted. For example, in the case of 8b10b, a single bit error that occurs in the transmission line could result in an invalid decoded value. Depending on how the decoder handles the value, it could either receive it or perform error correction, which will make it appear as though an error never occurred. Or it could receive it incorrectly for as many as eight possible errors. Either one of these outcomes is an undesired result. To avoid this situation and make sure all bits are accounted for, whether in error or not, using logic to produce the encoding makes testing for errors more consistent and reliable.

Also, if the transceiver hardware handles encoding, no matter what encoding is used, there will be extra bits added to the data stream that will never be known. 8b10b encoding introduces 25% more overhead or extra bits introduced to the data stream that the logic will never know existed. 64/66 encoding adds 3.125% additional bits, while 128/130 encoding has 1.5625% additional bits added. Even if using the most efficient coding, 128/130, for every trillion bits sent, it has transmitted 15.38 billion bits that are unaccounted for in the data stream. 8b10b would add 200 billion unaccounted for bits. Since the bit error rate equals bit errors divided by received bits. The unaccounted bits cause the received bits to be inaccurate, which in turn causes the bit error rate to be incorrect. Using transceiver hardware would make the recorded bit errors to be unreliable, as well as the bits received uncertain; therefore, making any bit error rate calculated with these values questionable. The transceiver encoding hardware should never be used for a bit error rate, testing design.

When the team working on the project at the time decided what encoding arrangement to use, the importance of DC balance was not well understood, and not taken into consideration. The extent to which dc balance can corrupt a signal was surprising and a lot of time was spent determining the reason for the instability in 4b5b. If 8b10b had been used in the Mini BERT, the importance of dc balance would have never been discovered because 8b10b has a strict disparity boundary. 8b10b is not going to stress out the receiver with a strict dc balance. It would verify operations under the best data scenario, but it would not be near as representative of the data it would see while in service. 8b10b will never have more than a +2 and -2 disparity. 8b10b is being phased out by 64/66 in many 10 Gbps applications. However, 64/66 does not have a strict dc balance like 8b10b. Therefore if the current trend is to use an encoding without a strict dc boundary, then the encoding used to verify transceiver operation should have a broader dc boundary as well. The current encoding of 64-bits of 4b5b and 64-bits of hexadecimal 5s is a better option to implement in code for the prototype than 64/66. 4b5b was more easily implemented than 8b10b and LFSR. LFSR has a more complicated implementation than line encoding. In the future, encoding is moving more in the direction of LFSR scramblers. Scramblers have adequate dc balancing, but their efficiency is far superior to 4b5b or 8b10b. Since an essential goal of this project was to give a high confidence level in the transmission medium being tested. 8b10b would have been under nearly perfect dc balance and would not have stressed the SFP, making the test invalid. Therefore 4b5b encoding was the encoding that was chosen.

In table 1, there are 16 possible 4b5b values. Of those values, only four have a negative disparity, and all of those were a -1 disparity. Seven of the remaining values have a +1 disparity and five have a +3 disparity. The five +3 values create a line code that has a substantial positive dc imbalance. Also, the PRBS for the Mini BERT has a count value that started at zero and counted up. The 4b5b encoded value for 0000, which would be the most common value to occur close to a startup, has a disparity of +3. When the device is initiated in this state, it will cause the transceiver to be very imbalanced. In the first fiber design, this would explain the enormous bit error rate, errors, signal losses, and inability to get a stable signal. A worst-case scenario for the 4b5b would have been a +3 disparity for every 5 bits sent, averaged over 128-bits would have given a worst-case disparity of +76.8. In the modified Mini BERT design, by making half of every 128-bits transmitted the unencoded 64-bits of 5s or 0101, this produces half the worst-case disparity of +38.4. It is not known how high the worst-case disparity has to be to cause instability. However, it was demonstrated that the modified Mini BERT design was able to transmit data for five weeks with zero errors indicating the worst-case disparity value that is required to cause instability must be higher than +38.4. For ease of implementation and the ability to test and stress the SFP, the 4b5b implementation with the addition of 50 percent hexadecimal 5s works well with the Mini BERT Design.

The limitations of the device come from the fact that it has a single reference clock and a single transceiver design line rate that are both fixed. The most important item missing from this design that a retail BERT would have is multiple encoding options and multiple line rates that can be tested. If another line rate were to be tested, both the clock and the transceiver design would have to be changed. If the customer wanted to test a line that was 5 Gbps or 20 Gbps, etc., it would not be possible with this design. The Mini BERT is only designed for 10 Gbps. With a few minor changes, the Mini BERT would have more functionality; however, if the design were changed to ASIC silicon, the design would always be locked down to one line rate. FPGAs are programmed from flash memory. Multiple designs would require multiple flashed programs to be used. FPGAs could be equipped with flash memory with multiple flashed programs that would determine at which line rate the customer wants the device to work.

5 CONCLUSION

The Mini BERT is a success in that this device is capable of detecting bit errors and recording those errors. It can estimate what kind of error rate to expect for fiber optic communications lines. The proposed design is a functioning device to satisfy the minimum requirements for a bit error rate tester to be made in the simplest cheapest way possible.

There are two possible routes to navigate to a future design. The current design is a prototype, and although it does satisfy the minimum requirements, it is a design concept whose functionality and performance could be improved with a few additions. Being able to select multiple line rates would be beneficial to customers who want to use differing fiber optic line rates. One route would be to take a single simplified device that could be programmed with any line speed and encoding a customer chooses upon device manufacturing. A second route would be adding as much customized memory, and device components as required to a single device, knowing this will increase the production cost significantly. The future design will walk a very tight line because it could be designed to add many customized options, but the goal of keeping it cost-efficient would keep this to a minimum. Future designs could include multiple versions of the transceiver designs for multiple line rates with a flash memory module. Another possible improvement to the design would be to make this hardware possible to integrate into a computer system for transmission back to the office from the field.

Future development of the Mini BERT is not tied to the Kintex 7 chip. The Kintex 7 chip is limited to a 12 Gbps transceiver design. If a newer, more powerful chip is used, even higher data rates could be utilized. Since we are using the Kintex 7, the Vivado Suite is required for the modules and code for describing the ethernet PHY. All other modules can be taken to any other software and maintain its functionality. Different FPGA chips could be used, as long as the brand-specific HDL software is used to create the ethernet PHY for that chip. The bulk of the design is not Kintex 7 dependent. With LFSRs comprising much of the future of data communications, there is still room for use in the Mini BERT design. There is not much gained by encompassing an LFSR into the

Mini BERT design, and the Mini BERT is a great inexpensive tool that could be used for testing the bit error rates of fiber optic lines in all companies that use a 10 Gbps line rate.

REFERENCES

[1] Optical_Cable_Corporation, "The importance of bit error rate testing to fiber optic channels." https://www.occfiber.com/x_upload/news/files/1469115240_05052016_White_Paper_-_(BERT)_Bit_Error_Rate_Testing.pdf, May 2016. Accessed on 2019-03-01.

[2] C. Allard, J. Campisi, S. Chue, and D. Winters, "WCU OCC Mini BERT 2020," tech. rep., Western Carolina University, School of Engineering and Technology, Cullowhee, NC, USA, May 2020. Accessed on 2020-04-11.

[3] Xilinx_inc, "Vivado hlx 2018.3 update 1." https://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/vivado-design-tools/2018-3.html, December 2018. Accessed on 2019-03-03.

[4] Xilinx_inc, "Kc705 evaluation board for the kintex-7 fpga." https://www.xilinx.com/support/documentation/boards_and_kits/kc705/...ug810_KC705_Eval_Bd.pdf. Accessed on 2018-07-10.

[5] Xilinx_inc, "Xilinx xtp132 kc705 schematics (rev 1.1)." https://www.xilinx.com/-support/documentation/boards_and_kits/kc705_Schematic_xtp132_rev1_1.pdf, April 2012. Accessed on 2020-03-14.

[6] Xilinx_inc, "Bitslip in logic." https://www.xilinx.com/support/documentation/application_notes/xapp1208-bitslip-logic.pdf, May 2014. Accessed on 2019-03-29.

[7] Xilinx_inc, "1g/2.5g ethernet pcs/pma or sgmii v16.0." https://www.xilinx.com/-support/documentation/ip_documentation/gig_ethernet_pcs_pma/v16_0/pg047-gig-eth-pcs-pma.pdf, April 2017. Accessed on 2020-03-14.

[8] Xilinx_inc, "7 series fpgas gtx/gth transceivers." https://www.xilinx.com/support-/documentation/user_guides/ug476_7Series_Transceivers.pdf, August 2018. Accessed on 2020-03-14.

[9] Phoenix_Contact_Inc., "Ethernet basics rev. 02." https://www.mouser.com/pdf-docs/Ethernet_Basics_rev2_en.pdf. Accessed on 2010-03-14.

[10] National_Instruments_Inc., "High-speed serial explained." file:///C:/Users/Downloads/HighSpeedSerial_WP_Final.pdf, 2016. Accessed on 2020-03-13.

[11] D. Behera, S. Varshney, S. Srivastava, S. Tiwari, and Freescale_Semiconductor, "Eye diagram basics: Reading and applying eye diagrams." https://www.edn.com/design/test-and-measurement/4389368/Eye-Diagram-Basics-Reading-and-applying-eye-diagrams, December 2011. Accessed on 2018-01-27.

[12] R. Jacobson, "2.5 quintillion bytes of data created every day. how does cpg & retail manage it?." https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/. Accessed on 2019-02-17.

[13] R. Killen, S. Appleton, A. Plankenhorn, P. Calvert, and S. Creg, "Multichannel -capable bit error rate test system." https://patents.google.com/patent /US6628621B1/en, Novermber 1999. Accessed on 2019-01-28.

[14] Nguyen, D.-L. Chen, R. Waldron, and C. Khanh, "Integral bit error rate test system for serial data communication links." https://patents.google.com/patent/US5726991A/en, March 1998. Accessed on 2019-01-28.

[15] Onsemi_inc, "Understanding data eye diagram methodology for analyzing high speed digital signals." https://www.onsemi.com/pub/Collateral/AND9075-D.PDF, June 2015. Accessed on 2019-01-18.

[16] V. Pataska and V. Puranik, "Fpga-based bit error rate performance measuring of wireless systems," International Journal of Innovative Research in Science,Engineering and Technology, p. 6, 2015. Accessed on 2019-01-18.

[17] GaussianWaves, “Coherent detection of differentially encoded bpsk (debpsk).” https://www.gaussianwaves.com/tag/bpsk/, November 2017. Accessed on 2019-03-18.

[18] GeeksforGeeks.com, “Computer network layers of osi model.” https://www.geeksforgeeks.org/layers-osi-model. Accessed on 2019-02-17.

[19] searchnetworking.techtarget.com, “Udp (user datagram protocol).” https://searchnetworking.techtarget.com/definition/UDP-User-Datagram-Protocol. Accessed on 2019-02-17.

[20] maximintegrated.com, “Pseudo random number generation using linear feedback shift registers.” https://www.maximintegrated.com/en/app-notes/index.mvp/id/4400. Accessed on 2019-02-17.

[21] B. Ley, “Diameter of a human hair.” https://hypertextbook.com/facts/1999/BrianLey.shtml, 1999. Accessed on 2020-03-14.

[22] Analog_Devices_Inc., “Sfp reference design kit preliminary data sheet.” https://www.analog.com/media/en/reference-design-documentation/reference-designs/56930225203490155448678519O5SFP_RDK_pra.pdf. Accessed on 2020-03-12.

[23] Maxim_Integrated_Inc., “Hfrd-05.0 reference design.” https://www.gaussianwaves.com/tag/bpsk/, November 2008. Accessed on 2020-03-13.

[24] R. Lavoie, “Understanding blocking capacitor effects.” https://www.embrionix.com/storage/app/media/presentation/UnderstandingBlockingCapacitorEffects_BEAugust2011.pdf, August 2011. Accessed on 2020-03-13.

[25] Synopsys.com, “Understanding the ethernet nomenclature data rates, interconnect mediums and physical layer.” https://www.synopsys.com/designware-ip/technical-bulletin/ethernet-dwtb-q117.html, 2020. Accessed on 2020-03-12.

[26] IEEE_802.3bp_1000BASE-T1_PHY_Task_Force, "Draft amendment to ieee std 802.3-2012." https://www.gaussianwaves.com/tag/bpsk/, March 2015. Accessed on 2020-03-12.

[27] premji schoolofcisconetworking.blogspot.com/, "tcp-versus-udp." https://premji-schoolofcisconetworking.blogspot.com/2011/09/tcp-versus-udp.html, September 2011. Accessed on 2020-03-12.

[28] thebroadcastbridge.com, "Understanding ip networks - forward error correction." https://www.thebroadcastbridge.com/content/entry/7502/understanding-ip-networks-forward-error-correction, January 2017. Accessed on 2020-03-12.

[29] juniper.net, "Introduction to the media access control (mac) layer 2 sublayer." https://www.juniper.net/documentation/en_US/junos/topics/concept/mac-qfx-series-understanding.html, December 2019. Accessed on 2020-03-12.

[30] revolvy.com, "Physical coding sublayer." https://www.revolvy.com/page/Physical-Coding-Sublayer?cr=1. Accessed on 2020-03-12.

[31] D. H. Johnson, "When to use ac coupling." http://www.sigcon.com/Pubs/news/4_15.htm, 2001. Accessed on 2020-03-12.

[32] ethernetopticaltransceiver.com, "Single mode 10 gigabit ethernet sfp module 8g fiber channel low power consumption." http://www.ethernetopticaltransceiver.com/sale-10710401-single-mode-10-gigabit-ethernet-sfp-module-8g-fiber-channel-low-power-consumption.html. Accessed on 2020-03-12.

[33] ad net.com, "Difference between single mode fiber and multi mode fiber." https://www.ad-net.com.tw/difference-between-single-mode-fiber-and-multi-mode-fiber/, June 2006. Accessed on 2020-03-12.

[34] community.fs.com, "Single mode vs multimode fiber:what's the difference." https://-community.fs.com/blog/single-mode-cabling-cost-vs-multimode-cabling-cost.html, March 2017. Accessed on 2020-03-12.

[35] V. I. Raffaele Giordano and A. Aloisio, "Highspeed deterministiclatency serial io." https://www.intechopen.com/books/field-programmable-gate-array/high-speed-deterministic-latency-serial-io, May 2017. Accessed on 2020-03-12.

[36] Arrow_Devices_Inc., "Behavioral modeling of clock/data recovery." https://www.slideshare.net/arrowdevices/behavioral-modeling-ofclockdatarecovery, April 2014. Accessed on 2020-03-12.

[37] geeksforgeeks.org, "Block coding in digital electronics." https://www.geeks-forgeeks.org/block-coding-in-digital-electronics. Accessed on 2020-03-12.

[38] Xilinx_Inc., "Ber measure and test time." https://www.xilinx.com/support/answers/66799.html, April 2016. Accessed on 2020-03-12.

[39] Keysight_Technologies_Inc., "How do i measure the bit error rate (ber) to a given confidence level on the j-bert m8020a and the m8040a high-performance bert?." https://www.keysight.com/main/editorial.jspx?ckey=1481106&id=1481106&nid=-11143.0.00&lc=eng&cc=US. Accessed on 2020-03-12.

[40] Jitterlabs_Inc., "Ber confidence-level calculator." https://www.jitter-labs.com/support/calculators/ber-confidence-level-calculator. Accessed on 2020-03-12.

[41] Keysight_Technologies_Inc., "N4962a serial bert 12.5 gb/s." https://www.keysight.com/en/pd-2166193-pn-N4962A/serial-bert-125-gb-s?cc=US&lc=eng. Accessed on 2020-03-12.

[42] techbed, "encoding concepts." https://www.slideshare.net/shashank03/1432encoding-concepts, September 2010. Accessed on 2020-03-12.

[43] N. McDonald, "Binary to bcd conversion algorithm." https://my.eng.utah.edu/ nm-cdonal/Tutorials/BCDTutorial/BCDConversion.html, May 2010. Accessed on 2020-03-12.

[44] viavisolutions.com, “Bit error rate test (bert).” https://www.viavisolutions.com/en-us/product-category/test-measurement/network-test-certification/bit-error-rate-test-bert. Accessed on 2020-03-12.