



Test Method for Encoder and Decoder Circuits used in Communication Networks

Rita Jain* and Mahendra Kumar Gupta**

Department of Electronics and Communication Engineering,

*Lakshmi Narayan Collage of Technology, (LNCT), Bhopal, (MP), India.

**MANIT, Bhopal, (MP), India

(Received 25 March 2012 Accepted 10 April 2012)

ABSTRACT : Transistor density on integrated circuit doubles every two year. For decades, Intel has met this challenge and has made Moore's Law a reality. As transistor counts climb so does the ability to increase device complexity and integrate many capabilities onto a chip. With increase in the functional complexity on the chip, accessing of internal sub-circuits of chip for testing purposes is becoming very difficult, as they are not directly accessible through primary inputs. So, the testing of chip is also becoming difficult, very time consuming and costly process with increasing cost. To reduce the cost of testing of chips by costly Automatic Test Equipment (ATE), Built-In-Self-Test (BIST) technique has emerged as a cheap alternative.

Keywords: Testing, Built-in Self-Test, Encoder, Decoder, Block Code, Linear Feedback Shift Register (LFSR), Pseudo Random Bit Sequence (PRBS) Generator.

I. INTRODUCTION

We use various electronics gadgets in our day to day life. At times it has been observed that these gadgets do not perform as desired. There can be many reasons for malfunctioning.

Technological development is enabling the production of increasingly complex electronic systems. All such systems must be verified and tested to guarantee their correct behavior. As the complexity grows, testing becomes one of the most significant factors that contribute to the total development cost. Testing takes up to 35% of all costs in the Integrated Circuit (IC) design and manufacturing process. Test, diagnosis and repair costs of complex electronic systems reach often 40–50% of the total product realization cost and it is projected that the test of a transistor will be more expensive than manufacturing it [1].

Thus, if a fault can be detected at chip or board level, then significantly larger costs per fault can be avoided. This is the reason why the focus is now on providing testability at chip, module or even at board level. The fault takes on 10X Cost rule, the need of an early detection of fault has forced most designers to implement BIST technique to their designs.

II. INCREASING PROBLEM OF TEST

Ever Expanding Chip : IC technology has shown an Compounded Annual Growth Rate of over 55% over past 50 years. This incredible growth has come from steady miniaturization of transistors and improvements in manufacturing processes. [2]

The Shrinking Boards: Yesterday there were few components on chip and pin count used to be very less. Most ICs were based on Plated Through Hole (PTH) technology and the dual-in line package provided more access to each and every pin.

Problem with Multi-layer Printed Circuit Board (PCB) : With SMT and BGA Packages, PCB has to be multilayered. The move from PTH to SMT packaging limits the physical access to every point on the target land [2].

Long Test Application Time : To test a combinational logic circuit with n inputs the truth table contains 2^n rows. Each of the n inputs can be independently assigned a value of 0 or 1 as test vectors to give a total of 2^n different test vectors. For larger circuits, the amount of time required to apply all of the test vectors becomes astronomical [2].

The situation for sequential circuits is even worse because the output of a sequential circuit depends, not only on the current inputs, but also on the internal state of all the memory elements.

III. DESIGN FOR TESTABILITY

Improvement in circuit testability can be achieved by decomposing the design into smaller circuit modules that can be tested more easily. Circuit modules are easy to test when only a few test vectors are needed to ensure that the module is fault free [3]. Thus, there must be a testing strategy that works directly on the nodes, i.e. the test vector generator and response analyser must be available on the chip. Design for testability is accomplished when circuit designers use techniques that provide good observability and controllability for all nodes in the circuit.

IV. BUILT IN SELF TEST

BIST is a testing technique in which unit (circuit/chip/board/system) under test is capable of testing itself. BIST significantly reduces off-chip communication by accommodating test generation and response evaluation hardware on the chip. Therefore, the limited I/O access constraint is eased. It eliminates much of the test pattern generation and simulation process. Well-organized BIST also partitions the circuit into pieces of moderate size to reduce the complexity of test generation and fault simulation.

Storing all of the test vectors internally would take extra circuitry. Instead, a Pseudo Random Bit Sequence (PRBS) generator is used as an Test Pattern Generator (TPG) to generate input vectors and a Signature Analyzer (SA) is used as a Test Response Analyser (TRA) on the outputs. The PRBS Generator and Signature Analyzer (SA) can be implemented using Linear Feedback Shift Registers (LFSR) which requires only a small number of logic gates and shift registers. This BIST Architecture is given in Fig. 1. A generic approach to implement BIST in a circuit is discussed in Fig. 2.

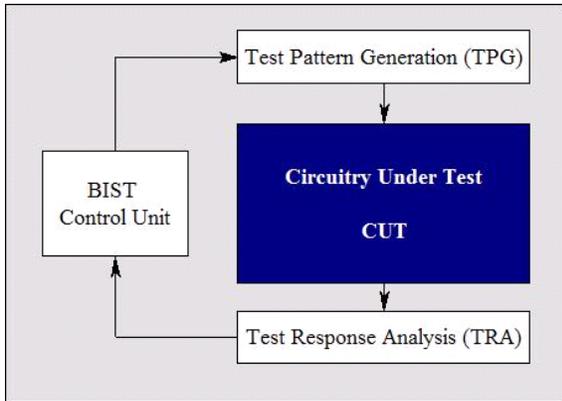


Fig. 1. BIST Architecture. BIST is a method of design, whereby the mission circuit tests itself. The test patterns are generated and applied to the unit under test (UUT) and a response analyzer declares the result.

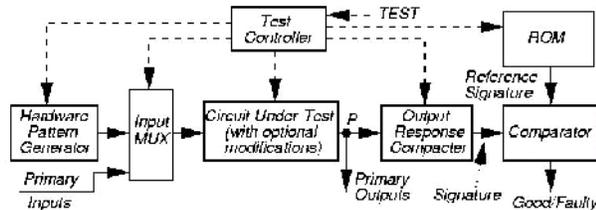


Fig. 2. A generic approach to BIST implementation in a circuit. Input MUX is used to switch between primary inputs in the normal mode and test patterns from TPG in the test mode. The response of a BIST-driven circuit is compared to its expected response to determine whether the circuit is operating correctly.

Different methods are used for implementing BIST. Each method has its own set of trade-offs and design considerations. If the BIST design is not appropriate for the

IC it is testing, then it can actually be a detriment to the design. Fig. 2 describes a generic approach to BIST implementation in a circuit. In normal mode the Circuit Under Test (CUT) is driven by external primary inputs, but in test mode the patterns are generated by built in TPG and applied to the circuit. The response of the circuit is observed by the test controller to the applied input test pattern.

V. LFSR FUNDAMENTALS

A LFSR is the heart of any digital system that generates PRBS with applications ranging from cryptography and bit-error-rate measurements, to wireless communication systems employing spread spectrum or CDMA techniques.

A. LFSR Generator Implementations

LFSR as given in Fig. 3 and Fig. 4 is a shift register which use a feedback to modify itself on each rising edge of the clock. The feedback causes the value in the shift register to cycle through a set of unique values. The length of the pseudo-random sequence is dependent on the length of the shift register and the number and the position of the feedback taps. The number and the position of the taps are represented by a polynomial. The choice of LFSR length, gate type, LFSR type, maximum length logic, and tap positions allows the user to control the implementation. And the feedback of the LFSR, controls the sequence of repeating values the LFSR will iterate through.

LFSR can be implemented using either the Galois or Fibonacci configuration of gates and registers. The order of the Galois weights is opposite that of the Fibonacci weights. The two LFSR implementations produce the same sequence when identical feedback weights are given.

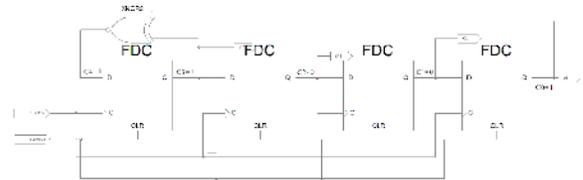


Fig. 3. Fibonacci Implementation of LFSR. In Fibonacci Implementation of LFSR the outputs from some of the registers are Exclusive-ORed/NORed with each other and fed back to the input of the shift register. In this figure these outputs are X_{12} , X_{22} , X_{32} and X_{42} .

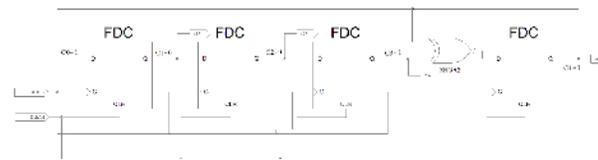


Fig. 4. Galois Implementation of LFSR. In Galois Implementation of LFSR, the XOR/XNOR gates are placed between the registers. In this figure these outputs are X_{11} , X_{21} , X_{31} and X_{41} .

The simulated waveforms of the LFSR as shown in fig. 3 and Fig. 4 are placed in Fig. 5. If characteristic polynomial is same the output of both implementations is same. .

The Galois form is generally faster than the Fibonacci in hardware due to the reduced number of logic gates in the feedback loop, thus making it the favoured form. For LFSR with only a few taps, the Fibonacci implementation will generally achieve a faster clock speed than its Galois counterpart. Although faster for a small number of taps, the Fibonacci implementation's performance degrades as the number of taps increases. The Galois implementation, however, sees hardly any performance loss with an increase in the number of taps. So, for a basic LFSR, the Galois implementation is recommended to achieve desired speed results for more than 5 taps [4].

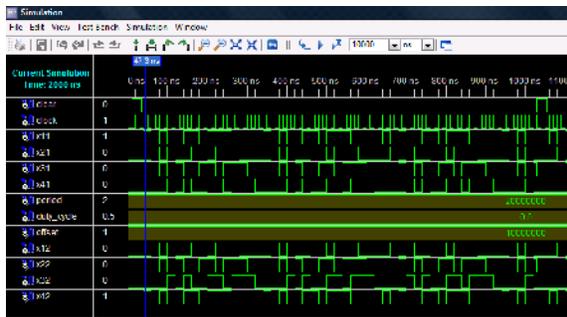


Fig. 5. Simulated results of both implementations. The outputs X_{12} , X_{22} , X_{32} and X_{42} are of Fibonacci Implementation and X_{11} , X_{21} , X_{31} and X_{41} are of Galois Implementation.

Due to static and dynamic randomness property LFSR finds its place in the basic BIST architecture. LFSR not only generates the pseudo random bit sequences (PRBS) but also act as polynomial multiplier and divider. This property is used for correlation and detection of faults if any.

VI. THE IMPORTANCE OF LINE CODING IN TELECOMMUNICATION SYSTEMS

Line Coding is needed to transmit digital information; more specifically, binary data over a digital repeated line. In a digital communication system, there exists a known set of symbols to be transmitted. At the receiver the binary data is recreated into a set of symbols by decoding the digital signal. The detected valid symbols are transmitted where as invalid symbols are discarded after declaring an error signal. A variety of line codes are available for different channel characteristics, different applications and performance requirements.

The dominant considerations effecting the choice of a line code are: 1) Timing, 2) DC content/component, 3) Power Spectrum, 4) Performance Monitoring, 5) Baseline Wandering 6) Probability of Error, and 7) Transparency. [5]

Commonly used line encoding techniques are Return-to-Zero (RZ), Non-Return-to-Zero (NRZ), bipolar, unipolar,

Manchester Encoding, Differential Manchester Encoding, Alternate Mark Inversion (AMI), multi-level, block codes etc.

VII. BLOCK CODES

A block code generates a block of n coded bits from m information bits and the coded word is known as $mBnB$ block code. The n -bit codeword can take on 2^n possible values corresponding to all possible combinations of the n binary bits. 2^m code words are selected from these 2^n possible code words, such that each m bit information block is uniquely mapped to one of these 2^m code words. Rest 2^{n-m} code words are discarded as they represented invalid code words and are declared as errors [1, 5, 9].

The rate of the code is $R_c = n/m$ information bits per codeword. If it is assume that code words are transmitted across the channel at a rate of R_s symbols/second, Then the information rate associated with an (m, n) block code is $R_b = R_c \times R_s = n/m R_s$ bits/second. [6], [7].

VIII. IMPLEMENTATION OF THE LOGIC DESIGN

Xilinx ISE 10.1 is used for implementing LFSR, Encoder, Decoder and Signature Analyser.

A. Concept of BIST Implementation on Encoder Decoder

This section describes the concept of BIST implementation on an encoder decoder circuit used in telecommunication. This encoder is designed to achieve all the advantages of line encoding and also acts as an multiplexer to combine two bit streams operating at bit rates of 2.048 Mbps (2B) and 1.024 Mbps (1B). This 2B bit stream carries the main information data (M_Data) whereas the 1B bit stream can be used to transmit the supervisory data (SV_Data) of the transmitting station. The designed encoder is termed as 2B4B Encoder because it encodes 2 bits of M_Data and 1 bit of SV_Data into 4 bits of coded data (C_Data). At the receiver end, a 4B2B Decoder circuit is used to decode and separate the two bit streams of 2B and 1B. Fig. 6 depicts this concept.

BIST is implemented to test 2B4B Encoder and 4B2B Decoder link. The status of Normal/Test signal on the control input of 2:1 Multiplexer (MUX) decides which of the two bit streams i.e. either M_Data or PRBS must be input to 2B4B Encoder. This Normal/Test signal acts as an enable signal to the LFSR. Similary another 2:1 MUX inputs either a PRBS or SV_Data to 2B4B Encoder at the second input. The 2B4B Encoder after encoding, outputs C_Data which acts as an input to 4B2B Decoder.

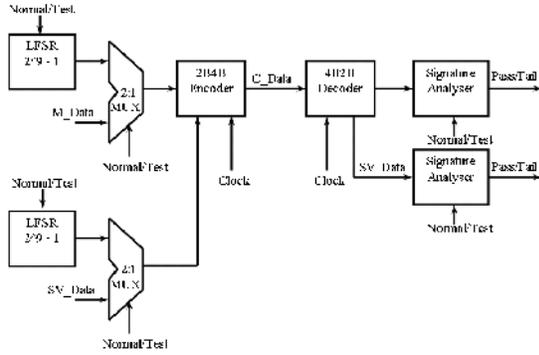


Fig. 6. A concept used to implement BIST on an Encoder Decoder Circuits.

The Normal/Test signal at the Signature Analyser switches ON the on-board LFSR and compares the received PRBS with the locally generated PRBS. This tests the circuit thoroughly. If the two streams are same, then the output of comparator is bit zero, it means the encoder – decoder circuits on the chip are functioning logically. If the output of comparator is bit one than any of the three circuits i.e encoder or decoder or the link is malfunctioning. Two LFSRs are used as a TPG to generate PRBS, one for M_Data and other for SV_Data. The bit rate of C_Data is 4.096 Mbps.

B. Implementation of 2^9-1 PRBS Generator using LFSR

According to ITU-T PRBS generator is used to test the communication system. 2^9-1 PRBS is constructed using

a LFSR having nine 1-bit D-flip-flops connected in a serial fashion as shown in fig. 7. The outputs of flip-flops 5th and 9th are XNORed and feedback to the input of LFSR. The control input clear (CLR) is used to clear the LFSR outputs and allows the LFSR to sequence pseudo randomly with the clock (CLK_MAIN). The LFSR performs its proper operation when its control input Enable (CHIP_EN) is high. Otherwise, it holds the previous value. All the control inputs are asynchronous. The outputs from all the flip-flops are taken and named as PRBS1, PRBS2, PRBS3 to PRBS9.

Xilinx ISE 10.1 is used to design the LFSR. The simulation results are shown in the fig. 9. To use this circuit in other circuits, it is converted into a symbol as depicted in fig. 8 and stored in the library for further use. The simulation results show that the sequence so generated repeats after 511 bits.

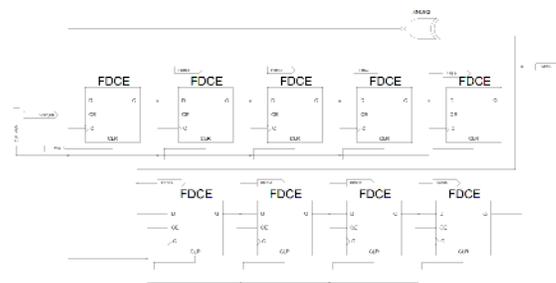


Fig. 7. Schematic Diagram of LFSR for generating 2^9-1 PRBS. Nine outputs are obtained. One from each D – Flip flop and are named as PRBS1, PRBS2 to PRBS9.

Table 1.

Sl. No.	1-bit of SV_Data		2-bits of M_Data		4-bit C_Data		
	A4	A3	A2	Z3	Z2	Z1	Z0
1.	0	0	1	1	0	0	1
2.	0	1	0	1	0	1	0
3.	0	1	1	0	0	1	1
4.	1	0	0	1	1	0	0
5.	1	0	1	0	1	0	1
6.	1	1	0	0	1	1	0

PRBS9_MANY_EN

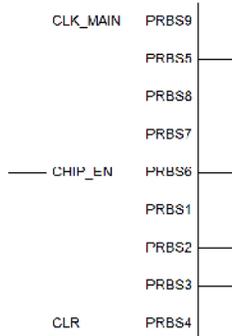


Fig. 8. Schematic Symbol of 2^9-1 PRBS Generator with nine outputs.



Fig. 9. Simulation results of 2^9-1 PRBS Generator with nine outputs.

C. Implementation of 2B4B Encoder [6], [7], [9], [10]

The 2B4B encoder module provides 3-bit to 4-bit encoding. The technique is shown in Fig. 10. A serial-in-parallel-out block is used to convert the serial bits of M_Data into two parallel bits. One bit is taken from SV_Data. These three parallel bits can have eight possible values that are 000, 001, 010, 011, 100, 101, 110 and 111 and are called datawords.

The bits of data word are identified as A4, A3, A2 (MSB to LSB). The output of the 2B4B encoder is a 4-bit C_Data and the bits are identified as Z3, Z2, Z1, Z0 (MSB to LSB). Analysis shows that by adding a bit one or bit zero to the data words, they get converted into 4-bit codeword. Since block codes are balanced codes, i.e. each 3-bit sequence after conversion must have equal numbers of ones and zeros. It has been found that out of six only eight data words will have two 1s and two 0s after conversion. These data words are 001, 010, 011, 100, 101 and 110 and are arranged according to the specific code translation map given in Table I.

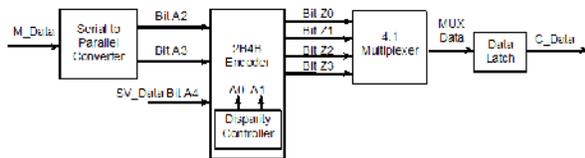


Fig. 10. Block Diagram for 2B4B Encoding Technique.

A 4-bit sequence can produce sixteen code words. Out of sixteen code words only six possess the DC balance property of line encoding. The other ten code words have either one 1s and three 0s, or, three 1s and one 0s. Thus

the two left out data words 000 and 111 could not be converted to a balanced codeword. Such cases are handled by a disparity controller.

D. Disparity Controller

Disparity is the difference between the number of 1s and 0s in the codeword.

- No/Neutral disparity indicates the number of 1s and 0s are equal in the codeword.
- Positive disparity indicates more 1s than 0s in the data word. A disparity bit A1 is set (if reset) when the data word is 111. This A1 bit is reset (if set) with next run of 111 after inverting the mapped codeword. This maintains DC Balance on the line.
- Negative disparity indicates more 0s than 1s in the data word. A disparity bit A0 is set (if reset) when the codeword is 000. This A0 bit is reset (if set) with next run of 000 after inverting the mapped codeword. This maintains DC Balance on the line.

To remove the disparity, the data word 111 is mapped to code words 1011 and 0100 alternatively. Thus in two runs of three ones balances the DC content on the line. Similarly the disparity generated by data word 000 can be balanced by mapping it into code words 0010 & 1101. Disparity controller thus provides two outputs A1 and A0 which keep the record of mapping and in turn disparity. Hence now input to the 2B4B encoder is of 5 bits and output is of 4 bits. Table II is the 2B4B mapping table.

In order to maintain the synchronisation property of block code, the code words like 0001, 1000, 1110 and 0111 has been avoided as they may lead to streams of four 1s or 0s or even greater.

Table 2: 2b4b Encoding Translation Map Having Unequal Numbers Of 1s And 0s.

S.No.	1-bit SV_Data & 2-bits of M_Data			Disparity Bits		4-bit C_Data (Codeword)				Comments
	A4	A3	A2	A1	A0	Z3	Z2	Z1	Z0	
1	0	0	0	0	0	0	0	1	0	-ve disparity
2	0	0	0	0	1	1	1	0	1	+ve disparity
3	0	0	0	1	0	0	0	1	0	-ve disparity
4	0	0	0	1	1	1	1	0	1	+ve disparity
5	0	0	1	0	0	1	0	0	1	No disparity
6	0	0	1	0	1	1	0	0	1	No disparity
7	0	0	1	1	0	1	0	0	1	No disparity
8	0	0	1	1	1	1	0	0	1	No disparity
9	0	1	0	0	0	1	0	1	0	No disparity
10	0	1	0	0	1	1	0	1	0	No disparity

11	0	1	0	1	0	1	0	1	0	No disparity
12	0	1	0	1	1	1	0	1	0	No disparity
13	0	1	1	0	0	0	0	1	1	No disparity
14	0	1	1	0	1	0	0	1	1	No disparity
15	0	1	1	1	0	0	0	1	1	No disparity
16	0	1	1	1	1	0	0	1	1	No disparity
17	1	0	0	0	0	1	1	0	0	No disparity
18	1	0	0	0	1	1	1	0	0	No disparity
19	1	0	0	1	0	1	1	0	0	No disparity
20	1	0	0	1	1	1	1	0	0	No disparity
21	1	0	1	0	0	0	1	0	1	No disparity
22	1	0	1	0	1	0	1	0	1	No disparity
23	1	0	1	1	0	0	1	0	1	No disparity
24	1	0	1	1	1	0	1	0	1	No disparity
25	1	1	0	0	0	0	1	1	0	No disparity
26	1	1	0	0	1	0	1	1	0	No disparity
27	1	1	0	1	0	0	1	1	0	No disparity
28	1	1	0	1	1	0	1	1	0	No disparity
29	1	1	1	0	0	1	0	1	1	+ve disparity
30	1	1	1	0	1	0	1	0	0	-ve disparity
31	1	1	1	1	0	1	0	1	1	+ve disparity
32	1	1	1	1	1	0	1	0	0	-ve disparity

The five input bits to the encoder are defined as bits A4, A3, A2, A1 and A0. A4 is SV_Data at bit rate of 1024 Kbps and A3 and A2 are at main data bits at bit rate of 2048 Kbps and A0 and A1 are the disparity bits. Simulated results are placed in Fig. 11. The RESET input, resets the logic circuit. The next rising CLK_4096 after RESET is deasserted and latches valid input data.

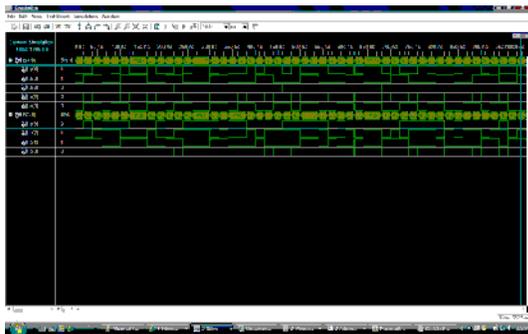


Fig. 11. Simulated results for 2B4B Encoder. a[0], a[1], a[2], a[3] and a[4] are inputs to the encoder whereas b[0], b[1], b[2] and b[3] are the encoded outputs.

E. Implementation of 2B4B Encoder Circuit

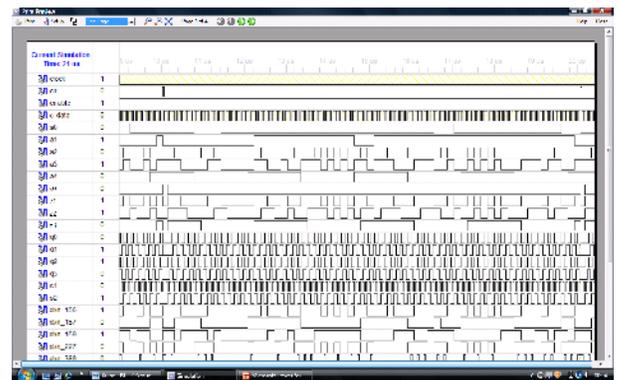


Fig. 12. Simulated Results for 2B4B Encoder Circuit. The M_Data is a PRBS data and C_Data is multiplexed data of Z0, Z1, Z2 and Z3.

Fig 13 gives the circuit schematics of the 2B4B Encoder. To obtain two parallel bits of M-Data, a serial to parallel converter is made using two D-Flip Flops. To detect all ones and all zeroes condition in the input bits, two triple input AND gates are used. If all the three bits are ones,

then there is generation of odd parity and output of AND gate is high otherwise it is low. Similarly if all the three bits are zero then the other AND gate is used. All the inputs are inverted and the output of the AND gate is set to one else it is low. Thus the outputs of the AND gates acts like a clock signal and these signals are used to drive the two counters.

The two AND gates and the two counters acts are a disparity controller and the least significant bits (LSB) of these counters are termed as A1 and A0. All the five bits A4, A3, A2, A1 and A0 are given to 2B4B Encoder which is designed using VHDL. The output of the encoder is parallel data bits Z3, Z2, Z1 and Z0. These four bits are converted into serial stream by using a 4:1 multiplexer. Thus the output of multiplexer is C_Data.

F. Implementation of 4B2B Decoder [6], [7], [9], [10]

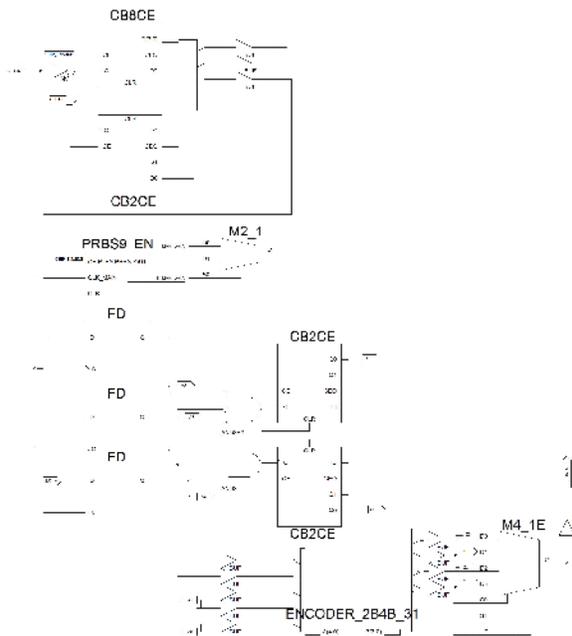


Fig. 13. The circuit schematics of 2B4B Encoder Circuit. PRBS9_EN is used as TPG. The serial bit stream of M_Data is converted into parallel as A2, A3. The SV_Data bit is A4 and the disparity generator give A1 and A0. The outputs of 2B4B Encoder are Z0, Z1, Z2 and Z3.

At the receiver end 4B2B decoder does the inverse function by first converting the serial bit stream into parallel by using a serial to parallel converter which is 1:4 demultiplexer. The input bits to the decoder are Y3, Y2, Y1 and Y0. The concept of 4B2B decoding is shown by a block diagram of Fig. 14.

The decoded bits B3, B2 and B1 are information bits A4, A3 and A2. In case of valid code words received, the bits B2 and B1 are decoded as the information signal bits A3 and A2, bit B3 is decoded as the supervisory information bit A4 and is handled by microcontroller of the system whereas bit B0 is decoded as error signal. Ten valid code words will be transmitted but there is a probability that the receiver receives any of the 16 code words. When the decoder receives an unrecognized codeword, such as an illegal codeword, it asserts the "Error Signal".

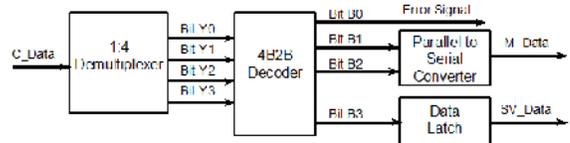


Fig. 14. Block Diagram for 4B2B Decoding Technique.

By asserting the error signal, the decoder indicates that an invalid code word has been received. The 4B2B decoding translation map is given in table III. If several codes in a row are received with errors, then the 4B2B decoder assumes that synchronisation between the transceiver has been lost and transmits all ones signal to the next networked equipment as recommended in ITU-T standard G 708. The circuit schematic of 4B2B decoder circuit is given in fig. 15.

The serial C_Data is converted into parallel data using 1:4 demultiplexer (DEMUX4). In order to remove the shifts between bits and to provide 50% duty cycle, the output of DEMUX4 is applied to a 4-bit shift register. The outputs of shift register are Y3, Y2, Y1 and Y0. These four bits are applied to 4B2B decoder, which decodes them into 1 bit of SV_Data, 2 bits of M_Data and one bit of Error Signal. If the received signal C_Data is error free the status of Error Signal is low. A high on Error Signal indicates the received data is in error. The simulated results are given in Fig. 16.

Table 3: 4b2b Decoding Translation Map.

S.No.	4-bit Codeword				3-bit Datarword			Error	Comments
	Y3	Y2	Y1	Y0	B3	B2	B1	B0	
0	0	0	0	0	0	0	0	1	Bits in Error
1	0	0	0	1	0	0	1	1	Bits in Error
2	0	0	1	0	0	0	0	0	Error Free Bits
3	0	0	1	1	0	1	1	0	Error Free Bits

4	0	1	0	0	1	1	1	0	Error Free Bits
5	0	1	0	1	1	0	1	0	Error Free Bits
6	0	1	1	0	1	1	0	0	Error Free Bits
7	0	1	1	1	1	1	1	1	Bits in Error
8	1	0	0	0	0	0	0	1	Bits in Error
9	1	0	0	1	0	0	1	0	Error Free Bits
10	1	0	1	0	0	1	0	0	Error Free Bits
11	1	0	1	1	1	1	1	0	Error Free Bits
12	1	1	0	0	1	0	0	0	Error Free Bits
13	1	1	0	1	0	0	0	0	Error Free Bits
14	1	1	1	0	1	1	0	1	Bits in Error
15	1	1	1	1	1	1	1	1	Bits in Error

G. Implementation of 4B2B Decoder Circuit

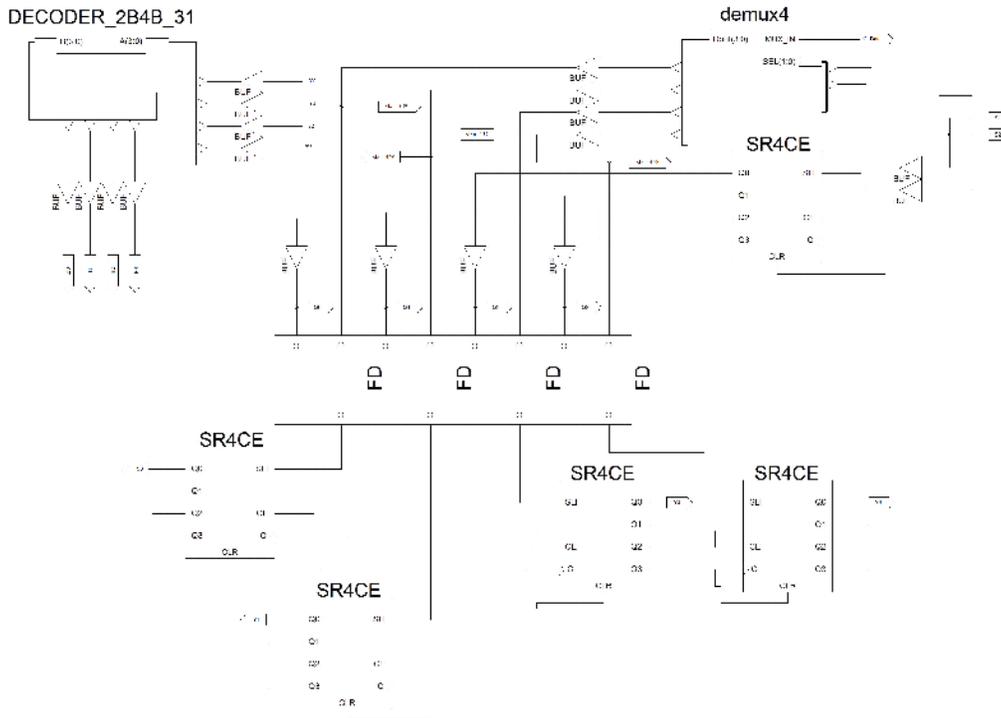


Fig. 15. Circuit Schematics of the 4B2B Decoder Circuit. The C_Data is applied at MUX_IN of DEMUX4. The output of DEMUX4 is xlnx_389, xlnx_456, xlnx_457 and xlnx_458. Y0, Y1, Y2 and Y3 are the parallel data without any shift and applied to 4B2B decoder, which after decoding provide the outputs as B0, B1, B2 and B3.

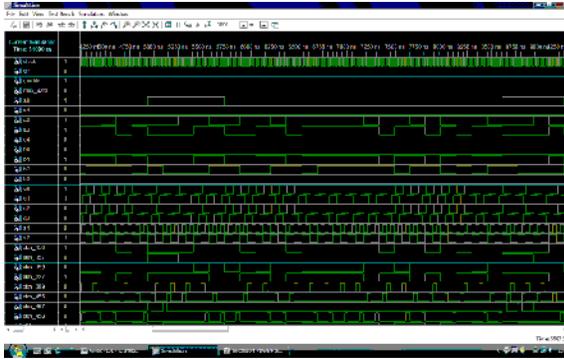


Fig. 16. Simulated results of the 4B2B Decoder Circuit. The waveforms of the xlxn_389, xlxn_456, xlxn_457, xlxn_458, B0, B1, B2 and B3 are shown.

IX. DESIGN IMPLEMENTATION

To implement the concept given in the fig. 6, the C_Data i.e output of the 2B4B Encoder Circuit is connected to C_Data of the 4B2B Decoder Circuit. The status on the normal/test signal decides the mode of operation of the system. The PRBS9_EN generates test patterns if the mode is TEST. The multiplexer passes these test patterns to the encoder. The output C_Data is applied to the decoder circuit. A local TPG is enabled at the decoder end. The received bit stream is compared with the locally generated bit stream. The result of comparison verifies the total link. If both the bit streams are same, it means that the link is healthy and systems are also in good health, else need attention for recovery. A test result for a healthy system is presented in the Fig. 17. The input test pattern is available on A2 and A3; output of decoder for these two signals is available on B2 and B1.

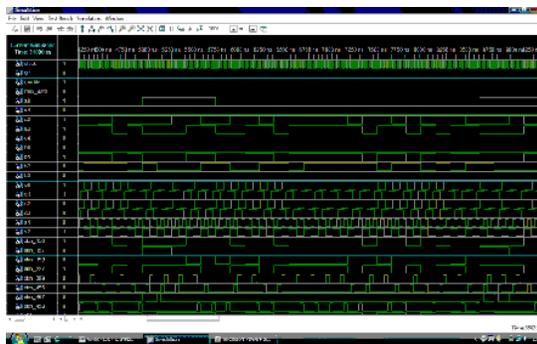


Fig. 17. Simulated results for the concept given in block diagram of fig. 6. The inputs to the 2B4B encoder are A2, A3 and A4 and the corresponding outputs of 4B2B Decoder are B1, B2 and B3. The output signal B0 is continuously low which indicates there is no error in the circuit. The above results depict that output signals are same as input.

X. CONCLUSION

This paper presents a design of block code encoder and decoder circuit – a useful component for telecommunications secondly it presents a new concept of implementing BIST for encoder decoder circuit which can be used for transmitting Network Management Signals over telecommunication networks using block codes. The circuit has been successfully designed on a breadboard and simulated using VHDL. The outputs of both the designs are as per concept and same.

REFERENCES

- [1] Afaq Ahmad, "A Simulation Experiment on a Built-In Self Test Equipped with Pseudorandom Test Pattern Generator and Multi-Input Shift Register (MISR)," *International Journal of VLSI design & Communication Systems (VLSICS)*, December 2010, Volume 1, Number 4, ISSN : 0976 – 1357 (Online : <http://aircse.org/journal/vlsi/currissue.html>) ; 0976 – 1527 (print).
- [2] JTAG (IEEE 1149.1/P1149.4) Tutorial – Introductory AL 10 Sept.–97 1149.1(JTAG)–Tut.I–1 1997 TI Test Symposium.
- [3] Indradeep Ghosh and Niraj K. Jha. (December 1998). High-level test synthesis: a survey. *Integration, the VLSI Journal. Volume 26*(1–2): 1, Pages 79–99.
- [4] Xilinx, Logi core, "Linear Feedback Shift Register," v3.0 DS257 (v1.0) March 28, 2003.
- [5] Behrouz A Forouzan, "Data Communication and Networking, IV Edition", The McGraw–Hill Companies, 2007pp 101 – 118.
- [6] Lattice, "8b/10b Encoder/Decoder," Reference Design RD1012, Nov. 2002.
- [7] Actel, "Implementing an 8b/10b Encoder/Decoder for Gigabit Ethernet in the Actel SX FPGA Family," *Application Note AC135*, Oct. 1998.
- [8] Xilinx, FPGA Basic Flow [online]. Available : http://www.xilinx.com/itp/xilinx8/help/iseguide/html/ise_basic_flow.htm.
- [9] Atef Abou El_Azm, "Circuits of Coder/Decoder and Error Detection in 5B6B Transmission Code," *Computers and Communications, IEEE Symposium on*, pp. 690, 2nd IEEE Symposium on Computers and Communications (ISCC '97), 1997.
- [10] Yong-Woo Kim, Beomseok Shin and Jin-Ku Kang, "High-speed 8B/10B encoder design using a simplified coding table" *IEICE Electronics Express*, Vol. 5(16): 581–585.