FPGA Implementation of 3 bits BCH Error Correcting Codes

Samir Jasam Mohammed
College of Engineering, University of Babylon.
Babylon, Iraq

Hayder Fadhil Abdulsada
College of Engineering, University of Babylon.
Babylon, Iraq

ABSTRACT
This paper describes the prototyping of a BCH (Bose, Chaudhuri, and Hocquenghem) code using a Field Programmable Gate Array (FPGA) reconfigurable chip. BCH code is one of the most important cyclic block codes. Designing on FPGA leads to a high calculation rate using parallelization (implementation is very fast), and it is easy to modify. BCH encoder and decoder have been designed and simulated using MATLAB, Xilinx-ISE 10.1 Web PACK and implemented in a xc3s700a-4fg484 FPGA. In this implementation we used 15 bit-size code word and 5 bits data, any 3 bits error in any position of 15 bits has been corrected. The results show that the system works quite well.

Keywords
Error correcting codes, BCH codes, encoding, decoding, FPGA

1. INTRODUCTION
The error control coding is one of the basic requirements of digital information and communication systems. It is important to ensure reliable transmission of information over noisy channels. Error correction coding is the means whereby errors which may be introduced into digital data as a result of transmission through a communication channel can be corrected based upon received data [1,2]. Error correcting codes have a wide range of applications in different fields like digital data communications, memory system design, and fault tolerant computer design among others [3].

BCH codes are one of the most powerful random-error correcting cyclic codes. BCH codes can be defined by two parameters that are code size n and the number of errors to be corrected t. BCH codes are being widely used in mobile communications, computer networks, satellite communication, as well as storage systems such as computer memories or the compact disc [1,4].

BCH codes are polynomial codes that operate over Galois fields (or finite fields). The generator polynomial of this code is specified in terms of its roots from the Galois field GF(2^m). The generator polynomial g(x) of the t error-correcting BCH code of length $2^m - 1$ is the lowest-degree polynomial over GF(2) which has $\alpha, \alpha^2, \alpha^3, \dots, \alpha^{2t}$ as its roots [i.e., g(α^i) = 0 for $1 \le i \le 2t$]. Let $\phi_i(x)$ be the minimal polynomial of α^i.Then g(x) must be the least common multiple of $\phi_1(x), \phi_2(x), \dots, \phi_{2t}(x)$, that is,

$$g(x) = \text{LCM}\ \{\phi_1(x), \phi_2(x), \dots, \phi_{2t}(x)\} \quad (1)$$

For any positive integers m ($m \ge 3$) and t ($t < 2^{m-1}$), there exists a binary BCH code with parameters of code words length $n = 2^m - 1$, number of parity check bits $n - k \le mt$, and minimum distance ($d_{min} \ge 2t + 1$) [5].

2. BCH CODES
BCH codes are polynomial codes that capable of correcting any combination of t or fewer errors in a block of $n = 2^m - 1$ digits. To know the encoding and decoding of the BCH code, the knowledge of finite fields is necessary [5].

2.1 BCH Encoder
An (n,k) binary BCH code encodes k-bits message into n-bits code word. The message vector can be expressed in a polynomial form as follows [6]:

$$m(x) = m_0 + m_1 x + m_2 x^2 + \cdots + m_{k-1} x^{k-1} \quad (2)$$

The message digits are utilized as a part of the codeword. The systematic encoding can be implemented by:

$$c(x) = p(x) + x^{n-k}\, m(x) \quad (3)$$

Where p(X) is the reminder and can be expressed as

$$p(X) = x^{n-k}\, m(x) \bmod g(x) \quad (4)$$

It follows from the definition of a t-error-correcting BCH code of length $n = 2^m - 1$ that-each code polynomial has $\alpha, \alpha^2, \dots, \alpha^{2t}$ as roots, $c(\alpha^i) = 0$, for $(1 \le i \le 2t)$ [7].

2.2 BCH Decoder
The biggest advantage of BCH codes is the existence of efficient decoding methods due to the special algebraic structure introduced in the codes [8].

Suppose that a code word $c(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{n-1} x^{n-1}$ is transmitted and the transmission errors result in the following received vector

$$r(x) = c(x) + e(x) \quad (5)$$

where e(x) is the error pattern [9].

Suppose that the error pattern e(x) has v errors at locations $X^{j_1}, X^{j_2}, \dots, X^{j_v}$, that is,

$$e(x) = x^{j1} + x^{j2} + \dots + x^{jv} \quad (6)$$

Where $0 \le j_1 < j_2 < \cdots < j_v < n$. Since $v(\alpha^i) = 0$, then $r(\alpha^i) = e(\alpha^i)$ [7].

The algebraic decoding BCH codes has the following general steps [2]:

1- Computation of the syndrome.
2- Determination of an error location polynomial σ(x), whose roots provide an indication of where the error are. There are several different ways of finding the error

polynomial such as Peterson's algorithm, and the Berlekamp-Massey algorithm.

3- Finding the roots of the error locator polynomial. This is usually done using the Chien search, which is an exhaustive search over all the elements in the field.

The first step of decoding a code is to compute the 2t syndrome components from the received vector r(x). These syndrome components may be obtained by substituting the field elements $\alpha, \alpha^2, \ldots, \alpha^{2t}$ into the received polynomial r(x).

$$S_i = r(\alpha^i) = e(\alpha^i) \quad (7)$$

From equation (7) we see that the syndrome S depends on the error pattern e(x) only.

To determine the error-location polynomial σ(x), we use Peterson's algorithm or Berlekamp algorithm. For small number of errors, Peterson's algorithm is more efficient than Berlekamp algorithm [2,5].

The error location polynomial σ(x) can be :

$$\sigma(x) = \sigma_0 + \sigma_1 x + \sigma_2 x^2 + \cdots + \sigma_v x^v \quad (8)$$

Where $v \leq t$. Which is equivalent to :

$$\sigma(x) = (1 + \beta_1 x)(1 + \beta_2 x) \ldots (1 + \beta_v x) \quad (9)$$

The error location polynomial coefficients $\sigma_1, \sigma_2, \ldots, \sigma_v$ are obtained for triple error correction

$$\sigma_1 = S_1$$

$$\sigma_2 = \frac{S_1^2 S_3 + S_5}{S_1^3 + S_3}$$

$$\sigma_3 = (S_1^3 + S_3) + S_1 \sigma_2 \quad (10)$$

The last step in decoding a BCH code is to find the error-location numbers that are the reciprocals of the roots of σ(x). The roots of σ (x) can be found simply by substituting $1, \alpha, \alpha^2, \ldots, \alpha^{(n-1)}$ into σ (x). Since $\alpha^n = 1$, $\alpha^{-l} = \alpha^{n-l}$. Therefore if α^l is a root of σ(X), α^{n-l} is an error-location number and the received digit r_{n-1} is an erroneous digit. The decoding of the code is completed by adding (modulo-2) e(x) to the received vector r(x) [8,10].The decoding steps are shown in the following block diagram in fig.(1)

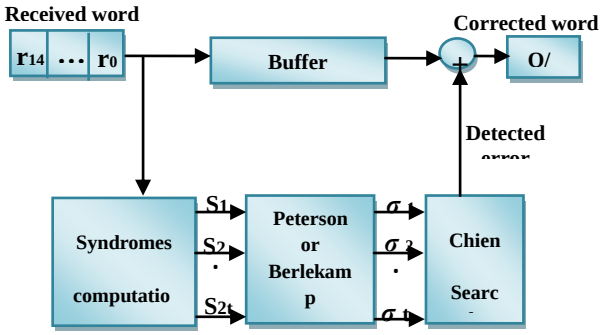

Fig 1: Block diagram for Decoding of BCH code

3. FIELD PROGRAMMABLE GATE ARRAY (FPGA)

Field-Programmable Gate Arrays (FPGAs) are pre-fabricated silicon devices that can be electrically programmed to become almost any kind of digital circuit or system [11].

FPGAs provide a number of advantages over fixed-function Application Specific Integrated Circuit (ASIC) technologies such as standard cells. ASICs are designed for specific application , and once manufactured, they cannot be modified, while FPGAs are configured in a relatively short amount of time, and often be reconfigured if a mistake is made [11,12].

An FPGA consists of an array of uncommitted configurable logic blocks (CLBs), programmable interconnects and Input Output blocks (IOBs). The basic architecture of an FPGA is shown in Fig.(2). FPGA architecture is dominated by programmable interconnects, and the configurable logic blocks which are relatively simple. This feature makes these devices far more flexible in terms of the range of designs that can be implemented with these devices [13].

FPGA can be configured anytime is needed, having a structure based on RAM technology that allowing the interconnectivity of the components to be changed as required. On the other hand, they allow parallel structures implementation, with response time less than a system with microprocessor [14].

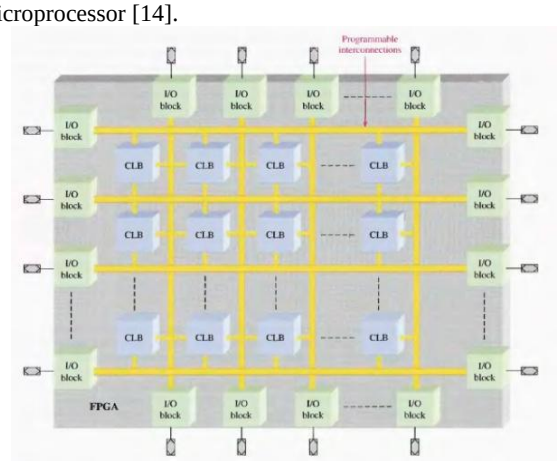

Fig 2: FPGA architecture

4. PROPOSED BCH CODEC DESIGN

The system proposed in this paper is based on the use of reconfigurable FPGA circuit for hardware implementation of encoder and decoder. Fig.(3) shows that one FPGA is implemented as encoder and the other as decoder. Each FPGA is connected with a computer in order to download the software of each system into an FPGA chip.

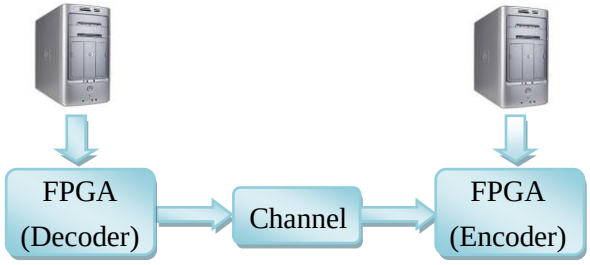

Fig 3: Communication system with FPGA.

4.1 Encoder Design

The encoding circuit calculates the parity bits using the LFSR (Linear Feedback Shift Register). The feedback connections of the LFSR are formed in a way that depends on the

generator polynomial of the code. The generator polynomial of the (15,5) BCH code is:
$g(x) = 1 + x + x^2 + x^4 + x^5 + x^8 + x^{10}$.

So the feedback connections of the LFSR are formed as shown in fig.(4).

The message digits are utilized as a part of the codeword. We shift the message digits into the rightmost k stages of a codeword register, and then appending the parity digits by placing them in the leftmost n-k stages. The input data of the encoding circuit is 5 bits and the output is a serial of 15 bits.

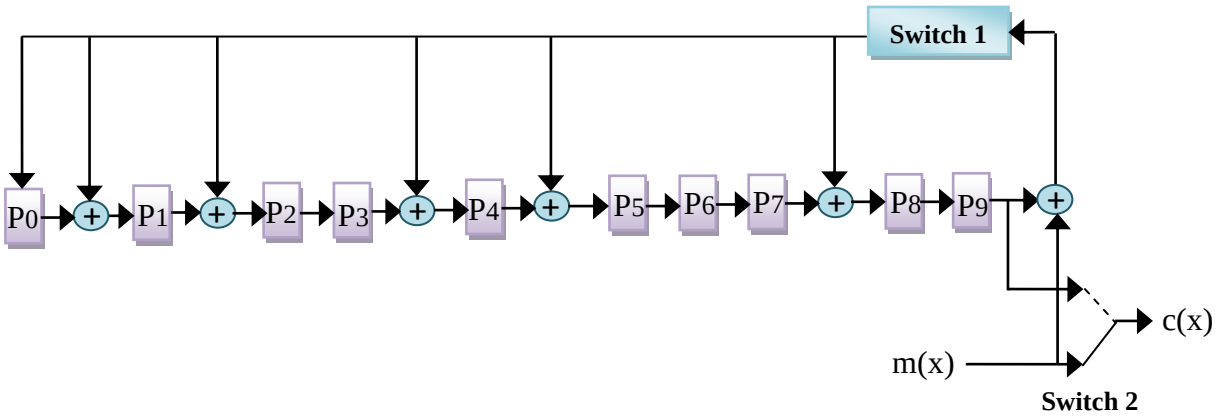

Fig 4: Encoding circuit for (15,5) BCH code.

Based on fig.(4), the proposed BCH encoder has been implemented on FPGA target device as shown in fig.(5).

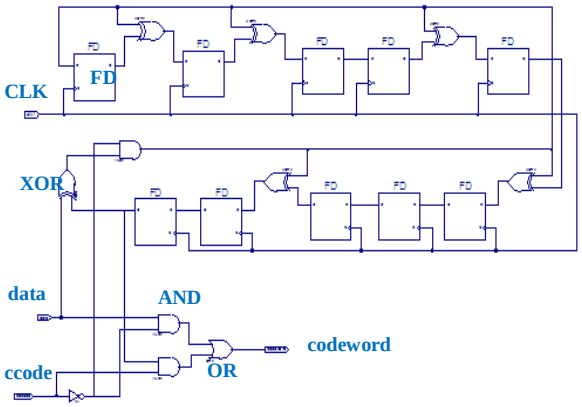

Fig 5: BCH encoding logic circuit implemented on FPGA.

For the BCH encoding circuit, a control signal is needed to allow the data signal to enter the encoding circuit and pass to the output at the same time. Also control signal gives a delay in order to make the encoding circuit able to prepare the parity bits. As a result the control signal is necessary to control the operation of switch 1 and switch 2 shown in fig.(4).

4.2 Decoder Design

The decoding process includes three steps as described in section 2. We have implemented the syndrome computation circuit for (15,5) BCH code. When the received word has entered the decoder, six syndrome components ($s_1, s_2, s_3, s_4, s_5,$ and s_6) are computed by substituting the field elements $\alpha, \alpha^2, \ldots, \alpha^6$ into the received polynomial r(x). All the computations depend on Galois fields GF(2^4). Fig.(6a), (6b), and (6c) show syndromes computation circuits.

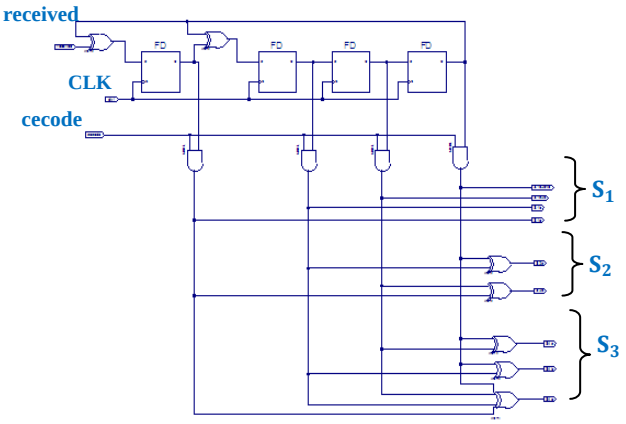

Fig 6a: Syndrome components (s_1, s_2 and s_4) logic circuit implemented on FPGA.

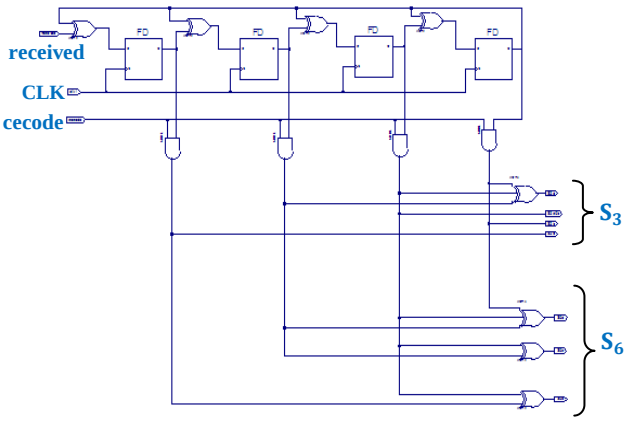

Fig 6b: Syndrome component (s_3 , s_6) logic circuit implemented on FPGA.

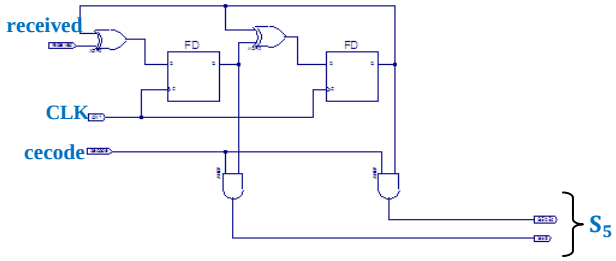

Fig 6c: Syndrome component (s_5) logic circuit implemented on FPGA.

When all the syndrome components are zero, this means that there is no error introduced for the transmitted code word during the transmission through a communication channel and hence the received data is the same as the transmitted data.

A BCH code is a polynomial code that operates over Galois fields (or finite field) so that the computation processes of a BCH error correcting code such as addition, subtraction, multiplication, and division are designed to operate over finite fields.

The error location polynomial coefficients of (15,5) BCH code are computed by using equation (10). The multiplication circuit which multiplies two elements over Galois fields $GF(2^m)$ for $m = 4$ is needed in computations of error location polynomial coefficients. This circuit is shown in fig.(7) below. The result of the multiplication is stored in shift registers.

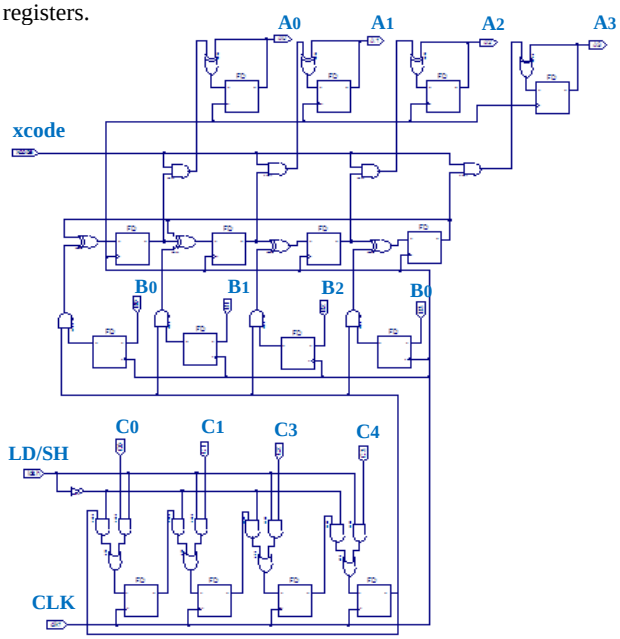

Fig 7: Circuit for multiplying two elements of $GF(2^4)$ implemented on FPGA.

σ(x). The roots of σ(x) can be found simply by substituting $1, \alpha, \alpha^2, \ldots, \alpha^{(n-1)}$ into σ (x). Then the error-location numbers are computed. This is called the Chien's search.

In the computation process of the decoder, we need to determine the inverse of an element in the field. The circuit that compute the inverse of any element over Galois fields $GF(2^m)$ for $m = 4$ is shown in fig.(8).

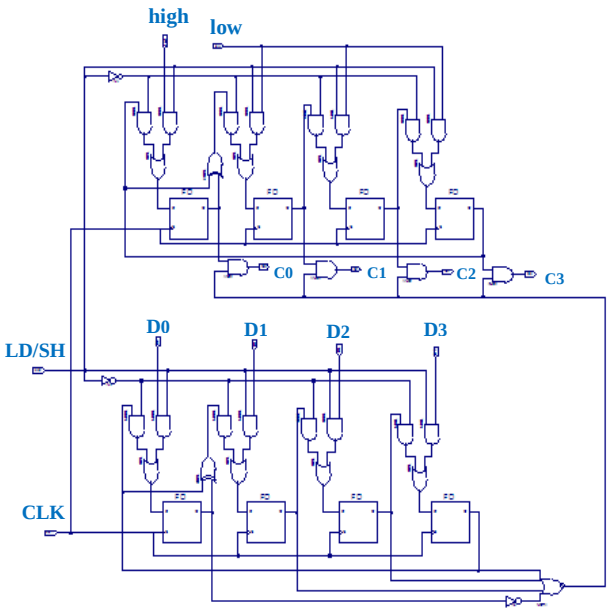

Fig 8: Circuit for computing the inverse of any element over $GF(2^4)$ implemented on FPGA

When the error location polynomial σ(x) is computed, then the error-location numbers represent the reciprocals of the roots of

After that we add (modulo-2) the detected error to the received word to get the corrected code word. The Chien's search circuit implemented on FPGA is shown in fig.(9).

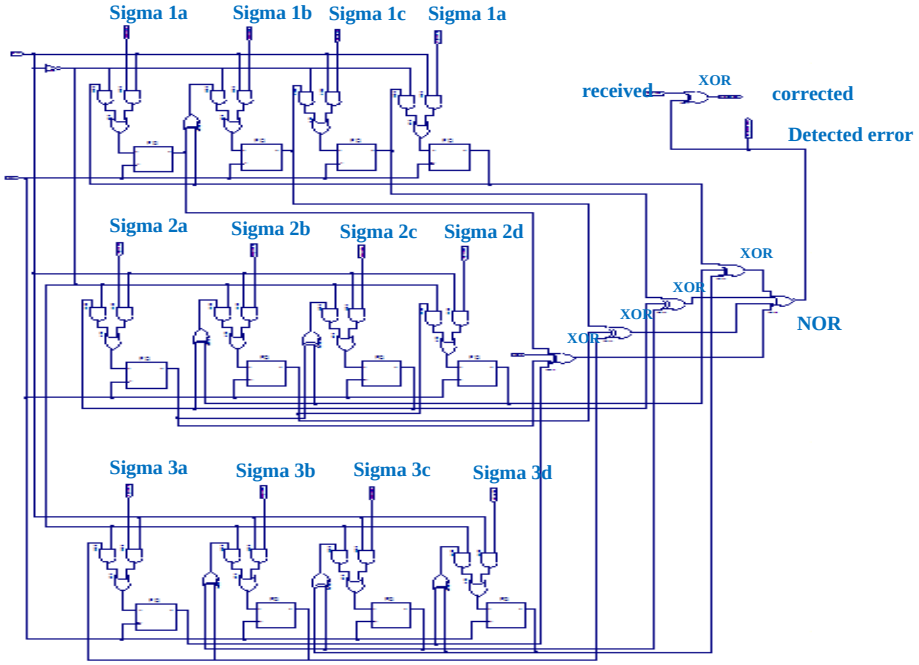

Fig 9: Chien's searching circuit for (15,5) BCH code implemented on FPGA

The complete decoder circuit which includes syndromes computation, error location polynomial coefficients and the Chien's search is shown in fig.(10) below. The decoder system also contains processing circuits such as shifting and enabling circuits which are needed to manage and control the operation of the decoder.

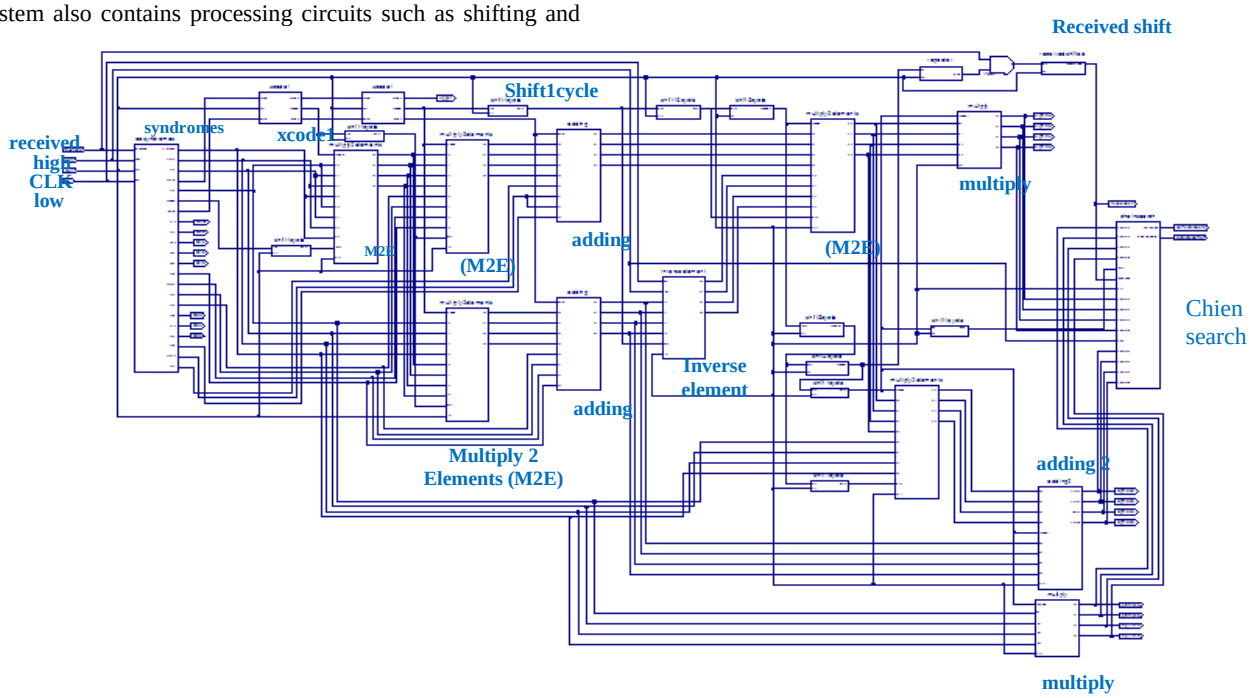

Fig 10: Complete decoder circuit for (15,5) BCH code implemented on FPGA.

At this point, it would be appropriate to map the decoding procedure in the following flowchart

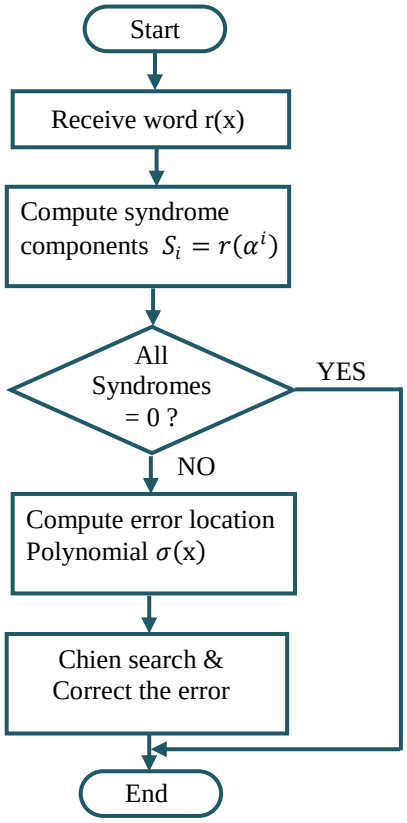

Fig 11: Flowchart for the decoding algorithm

5. RESULTS AND DISCUSSION

The proposed BCH encoder and decoder have been implemented on Spartan 3a-xc3s700a FPGA. Fig.(12) shows the experimental system of the block diagram shown in fig.(3). One FPGA is used as encoder and the other as decoder, and a wire channel is used between the two FPGAs. Each FPGA is connected with a computer in order to download the software of each system into an FPGA chip.

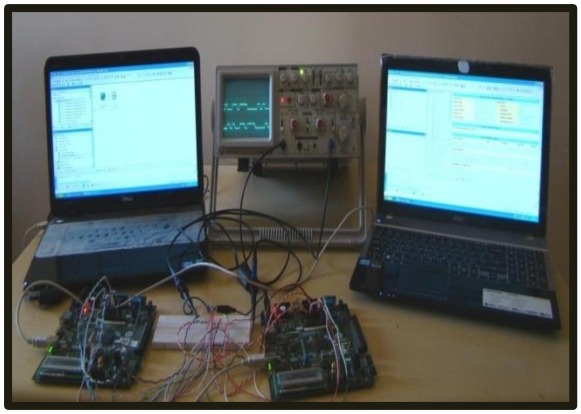

Fig 12: Picture of experimental system

The system is implemented in 50 MHZ clock frequency and the simulation results show that the circuits work quite well. In order to make the results of the system appear purely in the oscilloscope. We used 1.5625 MHZ clock frequency such as in [10]. So (1.5625 MHZ) clock frequency is used in our simulation and hardware implementation.

The proposed BCH encoder and decoder have been designed and simulated using MATLAB and Xilinx-ISE 10.1 Web PACK. Fig.(13a) shows the simulation results of BCH encoder using Xilinx-ISE 10.1 simulator and illustrates the date word (10011) which entered to the FPGA encoder to be encoded. After adding the parity bits to the data, the codeword are performed to be (100110111000010) as shown in the figure.

The simulation results of the BCH decoder are shown in fig.(13b). This figure shows the received word at the input of the decoder. When we make a comparison between the received word and the codeword, it appears that 3 bits error have been introduced into the codeword.

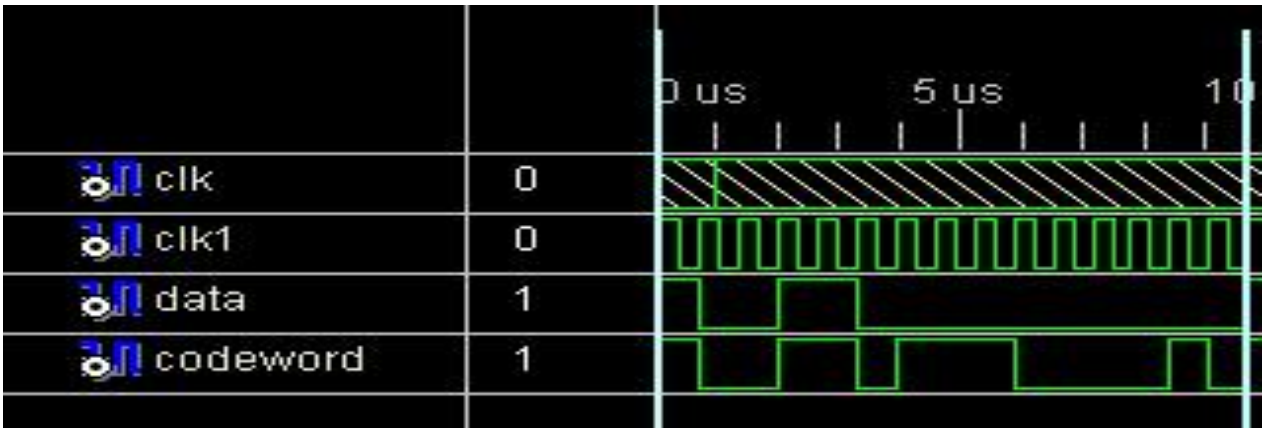

Fig 13a: Simulation results of BCH encoder

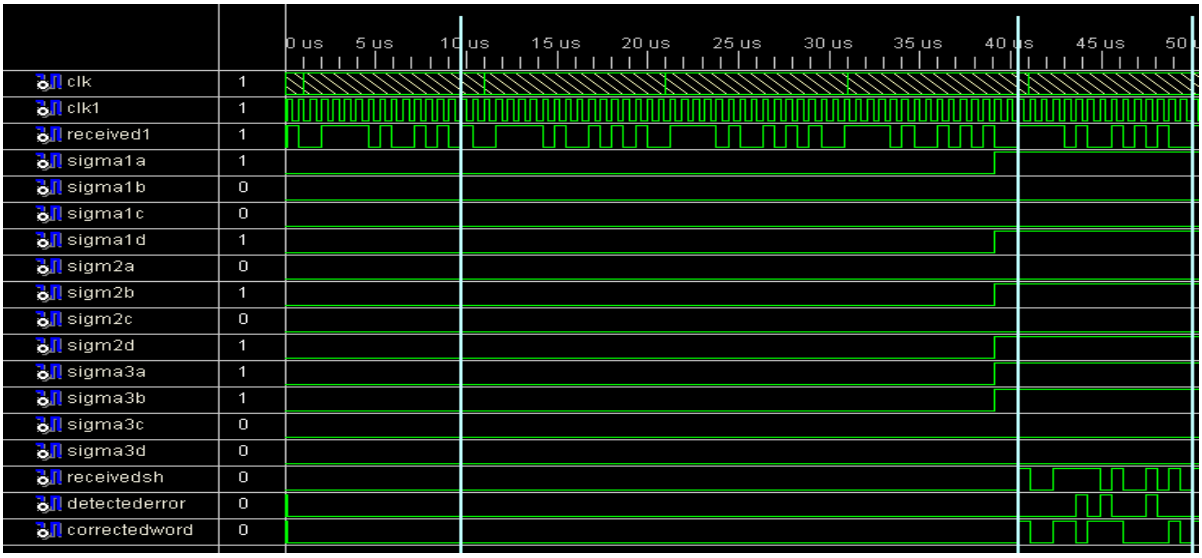

Fig 13b: Simulation results of BCH decoder

The values of the error location polynomial coefficients $\sigma_1, \sigma_2,$ and σ_3 are also computed to be $\sigma_1 = 1001, \sigma_2 = 0101,$ and $\sigma_3 = 1100$. The error that may be introduced into the codeword as a result of transmission through a communication channel is determined from the Chien's search circuit. By adding (modulo-2) the detected error to the received vector, the codeword can be recovered. The experimental results of BCH encoder and decoder are shown in fig.(14). The results of encoder decoder on the FPGA are displayed on oscilloscope.

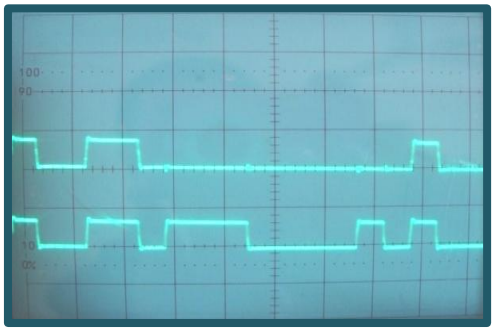

Fig 14a: Data word (5 bits) and codeword(15 bits).
Voltage: 5V/DIV, Time: 1µsec/DIV

The upper signal is the data word (5 bits) while the lower signal is the code word (15 bits). The codeword is transmitted through the channel. Fig (14.b) shows the transmitted codeword and the error presented by the channel.

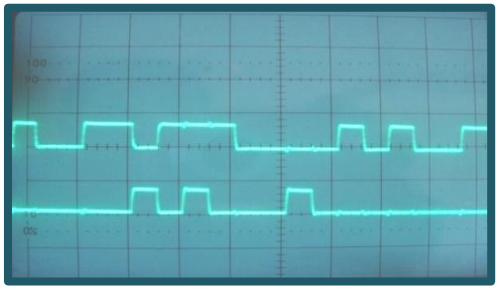

Fig 14b: Codeword and error presented by the channel.

Voltage: 5V/DIV, Time: 1µsec/DIV

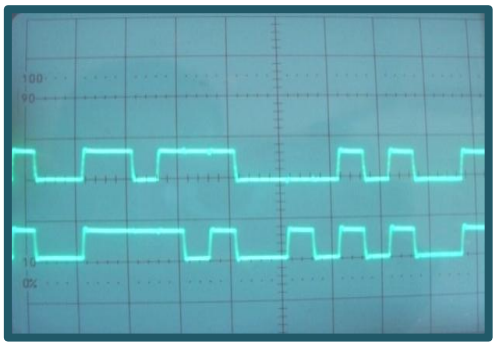

Fig 14c: Codeword and received word.
Voltage: 5V/DIV, Time: 1µsec/DIV

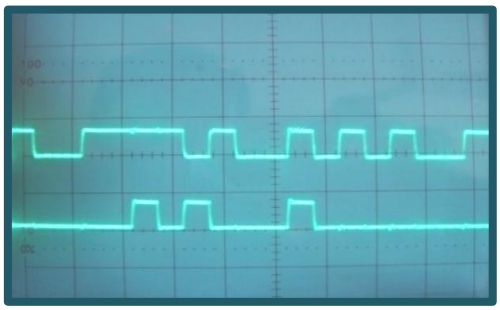

Fig 14d: Received word and the error detected by the decoder.
Voltage: 5V/DIV, Time: 1µsec/DIV

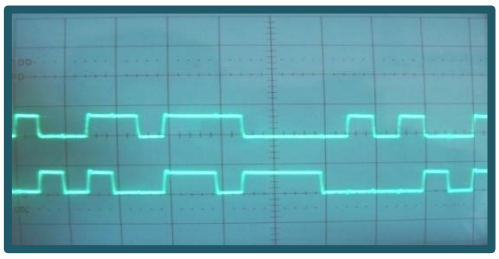

Fig 14e: Codeword and the corrected word.
Voltage: 5V/DIV, Time: 1µsec/DIV

After the hardware synthesis of the encoder and the decoder, the following device utilization summary was obtained:

Table 1. Encoder and decoder area summary.

	Encoder		Decoder	
Number of slices	16 out of 5888	1 %	147 out of 5888	2 %
Number of slices flip flops	26 out of 11776	1 %	235 out of 11776	1 %
Number of 4 input LUTs	22 out of 11776	1 %	154 out of 11776	1 %
Number of bonded IOBs	5 out of 372	1 %	28 out of 372	7 %
Number of BUFGMUXs	1 out of 24	4 %	1 out of 24	4 %

Table (1) shown above illustrates that the chip area occupied in a xc3s700a-4fg484 FPGA is very small. Therefore our systems can be integrated on one FPGA chip with others modules.

Also implementation of the system with FPGA has less computation time as compared with software solution. Table (2) shows a comparison between the computation time of the hardware system and the estimated response time, when the detection and correction processes run on a computer with a processor which works 2.4 GHZ clock frequency.

Table 2. Comparison between hardware and software processing at encoder and decoder for (15,5) BCH code

	$T_{hardware}$	$T_{software}$	n	k
Encoder	9.6106 µsec	1245.1 µsec	15	5
Decoder	40.22 µsec	13678.2 µsec		

6. CONCLUSION

The reliable transmission of information over noisy channels is one of the basic requirements of digital information and communication systems. Because of this requirement, modern communication systems rely heavily on error control coding.

In this work we have presented the prototyping of a BCH encoder and decoder using a Field Programmable Gate Array (FPGA). We used 15 bit-size codeword, any 3 bits error in any of 15 bits has been corrected.

The proposed BCH encoder and decoder have been designed and simulated using MATLAB and Xilinx-ISE 10.1 Web PACK and implemented in a xc3s700a-4fg484 FPGA and the results show that the system works quite well. Implementation of the system with FPGA has less computation time as compared with software solution.

7. REFERENCES

[1] Neubauer, J. Freudenberger and V. Kuhn "Coding Theory Algorithms, Architectures and Applications" John Wiley & Sons, 2007.

[2] T. K. Moon, "Error Correction Coding", John Wiley & Sons, 2005.

[3] A. S. Das, S. Das, and J. Bhaumik "Design of RS (255,251) Encoder and Decoder in FPGA", international journal of soft computing and engineering, Volume- 2, Issue-6, January 2013.

[4] J. G. Proakis, "Digital Communications", Prentice-Hall, 4th edition, 2005.

[5] S. Lin, and D.J. Costello Jr. "Error Control Coding Fundamentals and Applications", Prentice-Hall, New Jersey, 1983.

[6] R. Merha, G. Saini, and S. Singh, "FPGA Based High Speed BCH Encode for Wireless communication Applications", International Conference on Communication System and Network Technologies, 2011.

[7] B. Sklar, "Digital Communications Fundamentals and Applications Mathematical Methods and Algorithms", Prentice Hall, 2nd edition, 2001.

[8] Y. Jiang, "A Practical Guide to Error-Control Coding Using MATLAB", 2010.

[9] L. M. Ionescu, C. Anton, and I. Tutanescu, "Hardware Implementation of BCH Error-Correcting Codes on FPGA", Intrnational Journal of Intelligent Computing Research, Volume 1, Issue 3, June 2010.

[10] S. J. Mohammed, H. F. Abdulsada, "Design and Implementation of 2 BCH Error Correcting Codes using FPGA", Journal of Telecommunicatios, Volume 19, ISSUE 2, APRIL 2013.

[11] I. Kuon, R. Tessier and J. Rose. " FPGA Architecture: Survey and Challenges", 2008.

[12] J. P. Deschamps, G. J. A. Bioul and G. D. Sutter, "Synthesis of Arithmetic Circuits FPGA, ASIC and Embadded Systems", John Wiley & Sons, 2006.

[13] A.K. Maini, " Digital Electronics Principles, Devices and Applications ", John Wiley & Sons, 2007.

[14] R. Woods, J. McAllister, G. Lightbody and Y. Yi, "FPGA-based Implementation of Signal Processing Systems", John Wiley & Sons, 2008.