International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 9, September - 2013

Design and Implementation of Encoder for (15, k) Binary BCH Code Using VHDL

K. Rajani *, C. Raju **

*M.Tech, Department of ECE, G. Pullaiah College of Engineering and Technology, Kurnool
**Assistant Professor, Department of ECE, G. Pullaiah College of Engineering and Technology, Kurnool

Abstract

In this project we have designed and implemented a (15,k)BCH code using VHDL for reliable data transfer in channel with multiple error correction control. The digital logic implementation of binary encoding of multiple error correcting BCH code (15, k) of length n=15 over GF (2^4) with irreducible primitive polynomial x^4+x+1 is organised into shift register circuits. Using the cyclic codes, the reminder b(x) can be obtained in a linear (15-k) stage shift register with feedback connections corresponding to the coefficients of the generated polynomial. Three encoder are designed using VHDL to encode the single, double and triple error correcting BCH code (15, k) corresponding to the coefficient of generated polynomial. Information bit is transmitted in unchanged form upto k clock cycles and during this period parity bits are calculated in the LFSR then the parity bits are transmitted from k+1 to 15 clock cycles. Total 15-k numbers of parity bits with k information bits are transmitted in 15 codeword. Here we have implemented (15, 5, 3), (15, 7, 2) and (15, 11, 1) BCH code encoder on Xilinx Spartan 3 FPGA using VHDL and the simulation & synthesis are done using Xilinx ISE 10.1. Also a comparative performance based on synthesis & simulation on FPGA is presented.

1. Introduction

In a noisy channel when the data is transmitted, at the receiver side it is very difficult to retrieve actual data. It is frequently the case that a digital system must be fully reliable, as a single error may shutdown the whole system, or cause unacceptable corruption of data, e.g. in a bank account [5], [6]. There are so many error correcting methods, one of them is liner block code and the simplest block codes are Hamming codes [1]-[4]. They are capable of correcting only one random error and therefore are not practically useful, unless a simple error control circuit is required. More sophisticated error correcting codes are the Bose, Chaudhuri and Hocquenghem (BCH) codes that are a generalization of the Hamming codes for multiple-error correction. The (Bose-Chaudhuri-Hocquenghem) BCH codes form a large class of powerful random error correcting cyclic codes [7]-[9] having capable of multiple error correction [8]. BCH codes operate over finite or Galois fields [7]. The mathematical background concerning finite fields is well specified and in recent years the hardware implementation of finite fields has been extensively studied. In recent years there has been an increasing demand for digital transmission and storage system and it has been accelerated by the rapid development and availability of VLSI technology and digital processing. Programmable Logic Device (PLD) and Field Programmable Gate Arrays (FPGAs) [14], [15] has revolutionized hardware design and its implementation advantages provides various solution like FPGA is fully reprogrammable and reconfigurable. A design can be automatically converted from the gate level into the layout structure by the place and route software. Xilinx Inc. offers a wide range of components [12] which offers millions gate complexity and flip-flops, so even a relatively complex design can be implemented.

Here implementation of encoder for (15, k) BCH code organized by LFSR for single, double and triple error correction control using VHDL [16], [17] on FPGA presented and also performance compared based on synthesis and simulation result to understand the device utilization and timing simulation by targeting on Xilinx Spartan 3S 1000 FPGA and XSA 3S1000 Board of Xess Corporation [13]. For simulation and synthesis Xilinx ISE 10.1 is used.

The structure of this paper is as follows. Section II contains a brief description of the BCH code and generated polynomial. Section III contains Encoder Design for multiple error correction. Section IV contains simulation shows FPGA implementation results.

2. Generated Polynomial of Binary BCH Code Over GF (2^4)

As the BCH code operate in Galois Field [7], it can be defined by two parameters that are length of codewords (n) and the number of error to be corrected t.

A t-error-correcting binary BCH code is capable of correcting any combination of t or fewer errors in a block of $n = 2^m - 1$ digits. For any positive integer $m \geq 3$ and $t < 2^{m-1}$, there exists a binary BCH code with the following parameters:

Block length: $n = 2^m - 1$
Number of information bits: $k \geq n - m*t$
Minimum distance: $d_{min} \geq 2t + 1$.

The generator polynomial of the code is specified in terms of its roots over the Galois field GF (2^m) which is explained in [7]. Let _ be a primitive element in GF (2^m). The generator polynomial g(x) of

the code is the lowest degree polynomial over GF(2), which has α, α^2 , α^3,…… α^{2t} as its roots. [$g(\alpha^i)$= 0 for $1 \leq i \leq 2t$].

Let $\Phi_i(x)$ be the minimum polynomials of α_i then g(x) must be the,

$$g(x)= LCM\{\Phi_1(x),\Phi_2(x),\ldots\ldots\ldots,\Phi_{2t}(x)\} \quad (1)$$

As the minimal polynomial for conjugate roots are same i.e. as $\alpha^i = (\alpha^{i'})^{2l}$, $\Phi_i(x) = \Phi_{i'}(x)$, where $i = i' * 2^l$ for $l \geq 1$, thus generated polynomial g(x) of binary t-error correcting
BCH code of length given by eqn.(1) can be reduced to

$$g(x) = LCM\{\Phi_1(x),\Phi_2(x),\ldots\ldots\ldots,\Phi_{2t-1}(x)\} \quad (2)$$

BCH code generated by primitive elements is given in [8]. An irreducible polynomial *g(x)* of degree m is said to be primitive if only if it divides polynomial form of degree *n*, $x^n + 1$ for $n = 2^m-1$. In fact, every binary primitive polynomial *g(x)* of degree *m* is a factor of $x^{2^m-1} + 1$. A list of primitive polynomial for degree m and for finding irreducible polynomial is given in [7].

For (15, k) BCH code, let _ be a primitive element of the GF (2^4) given in [7] such that $1 + \alpha + \alpha^4$ is a primitive polynomial. From [7], [8] we find that minimal polynomials of α, α^3, α^5 are

$$\varphi_1(x) = 1+ x + x^4$$
$$\varphi_3(x) = 1+ x + x^2+ x^3+ x^4$$
$$\varphi_5(x) = 1+ x + x^2$$

For single error correcting, BCH code of length $n = 2^4 -1 = 15$ is generated by

$$g(x)= \varphi_1 (x)= 1+ x + x^4 \quad (3)$$

Here highest degree is 4 i.e (n-k = 4), thus the code is a (15, 11) cyclic code with $d_{min} \geq 3$ Since the generator polynomial is code polynomial of weighted 5, the minimum distance of this code is exactly 3.
For double error correcting, BCH code of length n = 15 is generated by

$$g(x) = LCM\{\Phi_1(x),\Phi_3(x)\}$$
$$= 1+ x^4 + x^6 + x^7 + x^8 \quad (3)$$

Here highest degree is 8 i.e (n-k = 8), thus the code is a (15, 7) cyclic code with $d_{min} \geq 5$.

For triple error correcting, BCH code of length n = 15 is generated by

$$g(x) = LCM\{\Phi_1(x),\Phi_3(x),\Phi(x)\}$$

$$= 1+ x+ x^2+ x^4+ x^5+ x^{88}+ x^{10} \quad (4)$$

Here highest degree is 10 i.e (n-k = 10), thus the code is a (15, 5) cyclic code with $d_{min} \geq 7$.

3. Design of BCH Encoder on FPGA

BCH encoder is usually implemented with a serial linear feedback shift register (LFSR) architecture [10], [11].

BCH codeword are encoded as

$$C(x) = x^{n-k} * i(x)+b(x) \quad (6)$$

Where $C(x) = c_0 + c_1 x + \ldots\ldots + c_{n-1} x^{n-1}$

$$i(x) = i_0 + i_1 x + \ldots\ldots+i_{k-1} x^{k-1}$$

$$b(x) = b_0 + b_1 x + \ldots\ldots + b_{m-1} x^{m-1}$$

and c_j , i_j , b_j Є GF(2) .

Then if *b(x)* is taken to be the polynomial such that

$$x^{n-k} i(x) = q(x) * q(x) – b(x) \quad (7)$$

The *k* data bits will be present in the codeword. Using the properties of cyclic codes [7], the remainder *b(x)* can be obtained in a linear *(n-k)*-stage shift register with feedback connections corresponding to the coefficients of the generator polynomial

$$g(x) = 1+ g_1 x+\ldots\ldots+ g_{n-k-1} x^{n-k-1} + x^{n-k} \quad (8)$$

Such a circuit is shown on Fig. 2.
On the encoder side, systematic encoding has been used, which makes easier implementation of encoder which is shown in Fig. 1.

Code Block	
Information	**Error Control**
$I_1I_2I_3\ldots\ldots I_{k-1}I_k$	$P_1\ldots\ldots p_{n-k-1}p_{n-k}$
k Data bits	n-k parity check bits

Figure 1: Systematic Encoding Diagram for (n, k) BCH Code

It is not useful to split the generator polynomial at the encoding side because it will demand more hardware and control circuitry. Therefore, the polynomial (1) is used as it is for encoding procedure. The digital logic implementing the encoding algorithms is organized into linear feedback shift-register circuits (LFSR) that mimic the cyclic shifts and polynomial arithmetic required in the description of cyclic codes.

The LFSR block diagram for (n, k) BCH encoder is shown in Fig. 2.

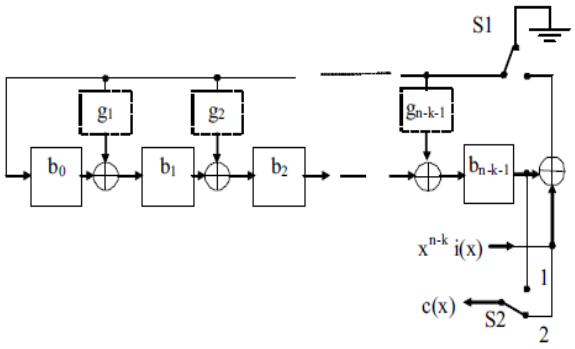

Figure 2: LFSR Encoding circuit for a (n, k) BCH codes

The encoder which is shown in Fig. 2 operates as follows

- For clock cycles *1* to *k*, the information bits are transmitted in unchanged form (switch *S2* in position *2*) and the parity bits are calculated in the Linear Feedback Shift Register (LFSR) (switch *S1* is on).
- For clock cycles *k+1* to *n*, the parity bits in the LFSR are transmitted (switch *S2* in position *1*) and the feedback in the LFSR is switch off (*S1* - off).

To observe the speed and resource utilization, RTL is generated verified and synthesized. The proposed BCH encoder has been implemented on Spartan3 XC3S1000 target device by using Xilinx ISE 10.1

A. Design of Encoder for (15, 11, 1) BCH Code

Encoder for (15, 11, 1) single error correcting BCH code is designed by organizing LFSR with generated polynomial $1+x+x^4$ and implemented on Spartan 3S1000 FPGA of Xilinx. The RTL view and Schematic is generated by synthesis with Xilinx ISE 10.1, shown in Fig 3 and Fig. 4.

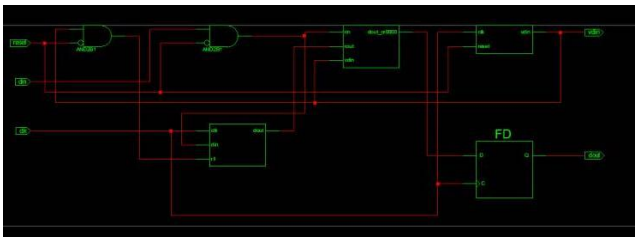

Figure 3: RTL for (15, 11, 1) BCH Encoder

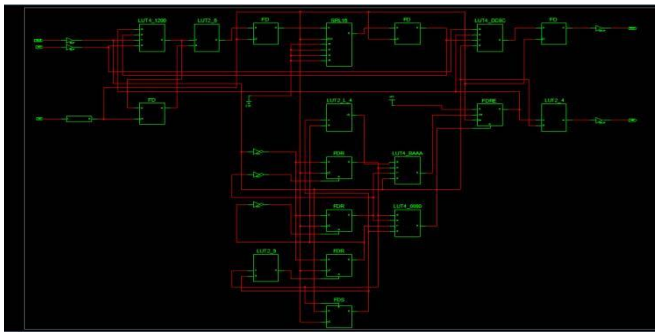

Figure 4: Schematic for (15, 11, 1) BCH Encoder

B. Design of Encoder for (15, 7, 2) BCH Code

Encoder for (15, 7, 2) double error correcting BCH code is designed by organizing LFSR with generated polynomial $1+x^4+x^6+x^7+x^8$ and implemented on Spartan 3S1000 FPGA of Xilinx. The RTL view and Schematic is generated by synthesis with Xilinx ISE 10.1, shown in Fig. 5and Fig. 6.

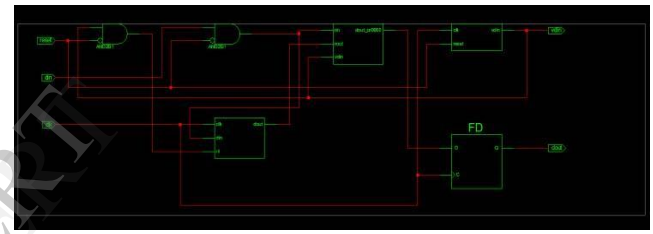

Figure 5: RTL for (15, 7, 2) BCH Coder

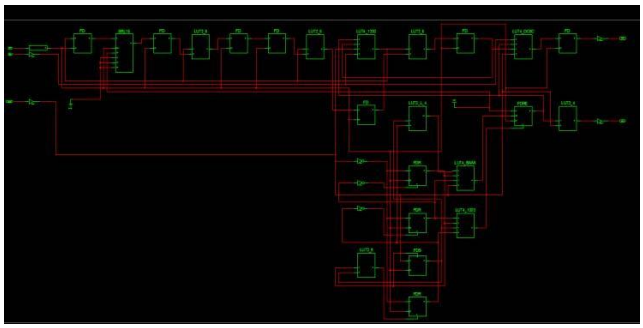

Figure 6: Schematic for (15, 7, 2) BCH Encoder

C. Design of Encoder for (15, 5, 3) BCH Code

Encoder for (15, 5, 3) triple error correcting BCH code is designed by organizing LFSR with generated polynomial $1+x+x^2+x^4+x^5+x^8+x^{10}$ and implemented on Spartan 3S1000 FPGA of Xilinx. The RTL view and Schematic is generated by synthesis with Xilinx ISE 10.1, which are shown in Fig.7 and Fig. 8.

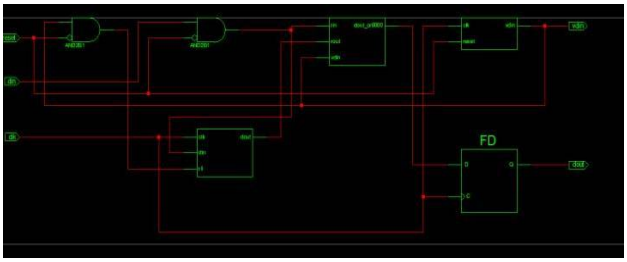

Figure 7: RTL for (15, 5, 3) BCH Encoder

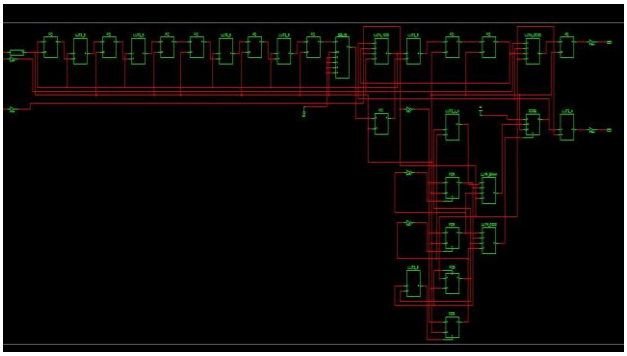

Figure 8: Schematic for (15, 5, 3) BCH Encoder

4. Result and Discussion

Input the netlist file generated from synthesizing, placing and routing on the Xilinx ISE 10.1 software. The simulation waveform for (15, k) BCH encoder is shown in Fig. 9, Fig. 10 & Fig. 11 under the simulation clock is 217.533 MHz.

The waveform simulation takes place with 20 ns clock period.

A. Simulation Waveform Result of (15, 11, 1) BCH Encoder

The timing simulation of (15, 11, 1) BCH encoder is shown in Fig. 9. Two data sequence is shown from 380 ns - 680 ns and 680 ns -980 ns. Total of 15 clock cycle is taking to complete transmitting of 15 codeword, 11-bits are information bit and 4- bits are parity bit. 11 Information bits "01011001001" are transmitting as it is where as other 4 bits "1010" are transmitting as parity bit "1100".

Figure 9: Simulated Waveform for (15, 11, 1) BCH Encoder

B. Simulation Waveform Result of (15, 7, 2) BCH Encoder

The timing simulation of (15, 7, 2) BCH encoder is shown in Fig. 10. Two data sequence is shown from 580 ns - 880 ns and 880 ns - 1180 ns. Total of 15 clock cycle is taking to complete transmitting of 15 codeword, 7-bits are information bit and 8- bits are parity bit. 7 Information bits "1011001" are transmitting as it is where as other 8 bits "00110100" are transmitting as parity bit "00011110".

Figure 10: Simulated Waveform for (15, 7, 2) BCH Encoder

C. Simulation Waveform Result of (15, 5, 3) BCH Encoder

The timing simulation of (15, 5, 3) BCH encoder is shown in Fig. 11. Two data sequence is shown from 640 ns -940 ns and 940 ns - 1240 ns. Total of 15 clock cycle is taking to complete transmitting of 15 codeword, 5-bits are information bit and 10- bits are parity bit. 5 Information bits "00110" are transmitting as it is where as other 10 bits "0110010100" are transmitting as parity bit "010001110".

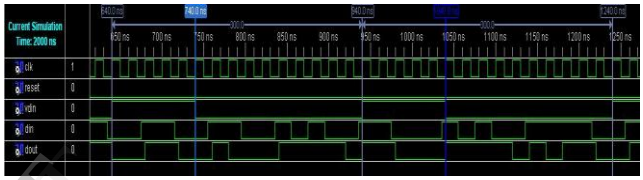

Figure 11: Simulated Waveform for (15, 5, 3) BCH Encoder

D. Comparison of performance between single, double and tripple error correcting (15, k) BCH code

We study and compare the behavior of multiple error correcting (15, k) BCH encoder by implementing on FPGA using VHDL. The device utilization and timing summary is given on table 1.

Table 1: Device Utilization and Timing Summary

Component Utilization/ Time	(15, 11, 1) BCH Encoder	(15, 7, 2) BCH Encoder	(15, 5, 3) BCH Encoder
No. of Slices	6	8	9
No. of Slice FF	9	12	15
4 input LUTs	12	14	16
Number of IOs	5	5	5
Simulation Clock	20 ns	20 ns	20 ns
Max. Combinational path delay	9.159 ns	9.159 ns	9.159 ns
Max. output required	8.81 ns	8.73 ns	8.57 ns
Total CPU time to Xst completion	6.13 sec	5.9 sec	5.7 sec

5. Conclusion

The result presented from the synthesis and timing simulation, shows the (15, 5, 3) BCH Encoder is more advantageous over the other two, according to speed requirement It can correct 3 error at the receiver side when the original data corrupt by the noise. But when considering area then (15, 11, 1) is better which can correct only 1 bit error. Also redundancy is less and data rate is more in it.

BCH codes have been shown to be excellent error-correcting codes among codes of short lengths. They are simple to encode and relatively simple to decode. Due to these qualities, there is much interest in the exact capabilities of these codes. The speed and device utilization can be improved by adopting parallel approach methods.

6. References

[1] M.Y. Rhee - "*Error Correcting Coding Theory*", McGraw-Hill, Singapore, 1989.
[2] S. Lin, and D.J. Costello Jr. - "*Error Control Coding*", Prentice-Hall, New Jersey, 1983.
[3] E. R. Berlekamp, *"Algebraic coding theory",* McGraw-Hill, New York, 1968.
[4] R.E. Blahut, "*Theory and practice of error-control codes*", Addison-Wesley, Reading, MA, 1983
[5] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*. Upper Saddle River, New Jersey 075458: Prentice Hall, Inc,1995.
[6] Berlekamp, E.R., Peile, R.E. and Pope, S.P. (1987), "*The application of error control to communications*", IEEE Communication Magazine, 25, no.4, pp 40-57.
[7] Shu Lin, Daniel J. Castello, "*Error control coding, Fundamentals and applications*", Premtice-Hall, New Jersey, 1983, Pages 15-50.
[8] Shu Lin, Daniel J. Castello, "*Error control coding, Fundamentals and applications*", Premtice-Hall, New Jersey, 1983, Page 141-182.
[9] W.W. Peterson, *"Encoding and error-correction procedures for the Bose-Chaudhuri Codes"*, IRE Trans.Inf. Theory, IT-6, pp. 459-470, September 1960.
[10] Goresky, M. and Klapper, A.M. Fibonacci and Galois representations of feedback-with-carry shift registers, *IEEE Transactions on Information Theory*, Nov 2002, Volume**:** 48, On page(s): 2826 – 2836.
[11] Panda Amit K, Rajput P, Shukla B, "Design of Multi Bit LFSR PNRG and Performance comparison on FPGA using VHDL", *International Journal of Advances in Engineering & Technology (IJAET), Mar 2012*, Vol. 3, Issue 1, pp. 566-571
[12] Xilinx, Inc. Xilinx Libraries Guide, 2011.
[13] Xess Corp.. XSA-3S1000 Board V1.1 User Manual. Available:http://xess.com/manuals/xsa-3S-manual-v1_1.pdf. Sept 2007.
[14] J J.Rose S.D. *Brown, R.J. Francis – "Field Programmable Gate Arrays*", Kluwer Academic Publishers, 1992
[15] Brown S., Vranesic Z "Fundamental of Digital Logic Design with VHDL" McGraw Hill, 2nd Edition.
[16] P. J. Ashenden, *The VHDL Cookbook*, 1st ed. Dept. Computer Science, University of Adelaide, South Australia: University ofAdelaide, 1990.
[17] Bhasker J, "*A VHDL Primer*", P T R Prentice Hall, Pages 1-2, 4-13, 28-30