IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834,p- ISSN: 2278-8735. Volume 7, Issue 6 (Sep. - Oct. 2013), PP 01-06
www.iosrjournals.org

Design and Implementation of Encoder for (15, k) Binary BCH Code Using VHDL and eliminating the fan-out

M. Divya, K. Vinodalakshmi, G. Sumalatha, Shahbazsarik, Abhishek Awasthi
Dept. of ECE, Avanthieng college Hayathnagar, India

Abstract: *In this paper we have designed and implemented(15, k) a BCH Encoder on FPGA using VHDL for reliable data transfers in AWGN 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 organized 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 up to 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 code word. 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 13.3. BCH encoders are conventionally implemented by linear feedback shift register architecture. Encoders of long BCH codes may suffer from the effect of large fan out, which may reduce the achievable clock speed. The data rate requirement of optical applications require parallel implementations of the BCH encoders. Also a comparative performance based on synthesis & simulation on FPGA is presented.*
Keywords: *BCH, BCH Encoder, FPGA, VHDL, Error Correction, AWGN, LFSR cyclic redundancy checking, fan out .*

I. 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 generalisation 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 fields 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.

II. 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 code words (*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 B BCH code with the followingparameters:

Block length: $n = 2^m$ _ 1
Number of information bits: k _ n – m*t
Minimum distance: d_{min} _ 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

$\alpha, \alpha^2, \alpha^3, \ldots, \alpha^{2t}$ as its roots. [g(α^i)= 0 for 1α i α2t]

Let $\_{i}$(x) be the minimum polynomials of $\_{i}$ then g(x) must be the,

$$g(x) = LCM\{\varphi_1(x), \varphi_2(x), \ldots\ldots, \varphi_{2t}(x)\} \quad (1)$$

As the minimal polynomial for conjugate roots are same i.e. as$\alpha^i = (\alpha^{i'})^{2l}$,$\alpha_i$(x) =$\alpha_{i'}$ (x), where i = i' * 2^l for l α1, thus generated polynomial g(x) of binary t-error correcting BCH code of length given by eqn.(1) can be reduced to

$$g(x) = LCM\{\varphi_1(x), \varphi_3(x), \ldots\ldots, \varphi_{2t-1}(x)\} \quad (2)$$

BCH code generated by primitive elements is given in [8]. An irreducible polynomial *g(x)* of degree m is said to beprimitive if only if it divides polynomial form of degree n,x^n+ 1 for $n = 2^m$_1. In fact, every binary primitivepolynomial *g(x)* of degree *m* is a factor of x^{2m_1}+ 1. A listof 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 $\alpha, \alpha^3, \alpha^5$ 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}= 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

$$g(x) = LCM\{\varphi_1(x), \varphi_3(x)\}$$

$$= 1 + x^4 + x^6 + x^7 + x^8 \qquad (3)$$

Here highest degree is 8 i.e. (n-k = 8), thus the code is a (15, 7) cyclic code with d_{min} _ 5.
For triple error correcting, BCH code of length n = 15 is generated by

$$g(x) = LCM\{\varphi_1(x), \varphi_3(x), \varphi_5(x)\}$$
$$= 1 + x + x^2 + x^4 + x^5 + x^8 + x^{10} \quad (4)$$

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

III. Design Of Bch Encoder On Fpga

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

BCH codeword are encoded as

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

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

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

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

and c_j, i_j, $b_j \in GF(2)$.

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

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

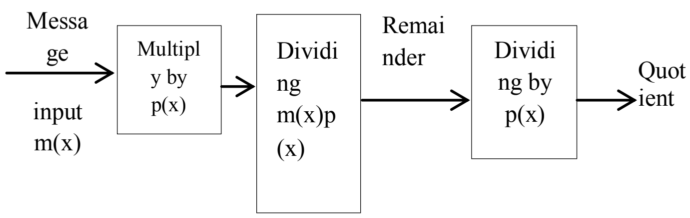
The *k* data bits will be present in the code word. Using theproperties of cyclic codes [7], the remainder *b(x)* can beobtained in a linear *(n-k)*-stage shift register with feedbackconnections corresponding to the coefficients of the generator polynomial

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

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 blocks →	
Information	Error control
I1 I2 I3 .... Ik-1 Ik	P1 .... Pn-k-1 Pn-k
k Data bits	n-k parity check bits

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 (I) 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.



IV BCH ENCODERArchitecture

An (n, k) binary BCH code encodes a k-bit message into an n-bit code word. A k bit message $(m_{k-1}, m_{k-2}, \ldots, m_0)$ can be considered as the coefficients of a degree $k-1$ polynomial $m(x) = m_{k-1}x^{k-1} + m_{k-2}x^{k-2} + \cdots + m_0$ where $m_{k-1}, m_{k-2}, \ldots, m_0 \in GF(2)$ Meanwhile, the corresponding n-bit code word $(c_{n-1}, c_{n-2}, \ldots, c_0)$ can be considered as the coefficients of a degree $n-1$ polynomial $c(x) = c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \cdots + c_0$, where $c_{n-1}, c_{n-2}, \ldots, c_0 \in GF(2)$ encoding of BCH codes can be simply expressed by

$$c(x) = m(x)g(x)$$

where the degree $n-k$ polynomial $g(x) = g_{n-k}x^{n-k} + g_{n-k-1}x^{n-k-1} + \cdots + g_0 (g_{n-k}, g_{n-k-1}, \ldots, g_0 \in GF(2))$ is the generator polynomial of the BCH code. Usually, $g_{n-k} = g_0 =' 1'$. However, systematic encoding is generally desired, since message bits are just part of the code word. The systematic encoding can be implemented by

$$c(x) = m(x) \cdot x^{n-k} + \text{Rem}(m(x) \cdot x^{n-k})_{g(x)} \quad (1)$$

where $\text{Rem}(f(x))_{g(x)}$ denotes the remainder polynomial of dividing $f(x)$ by $g(x)$. The architecture of a systematic BCH encoder is shown in Fig. 1. During the first k clock cycles, the two switches are connected to the "a" port, and the k-bit message is input to the LFSR serially with most significant bit (MSB) first.

Meanwhile, the message bits are also sent to the output to form the systematic part of the code word. After k clock cycles, the switches are moved to the "b" port. At this point, the $n-k$ registers contain the coefficients of $\text{Rem}(m(x)x^{n-k})$. The remainder bits are then shifted out of the registers to the code word output bit by bit to form the remaining systematic code word bits. For binary BCH, the multipliers in Fig. 1 can be re-placed by connection or no connection when $g_i (0 \le i < n-k)$ is "1" or "0," respectively. The critical path of this architecture consists of two XOR gates, and the output of the right-most XOR gate is input to all the other XOR gates. In the case of long BCH codes, this architecture may suffer from the long delay of the right-most XOR gate caused by the large fanout. Although the serial architecture of BCH encoder is quite straight forward, in the case when it cannot run as fast as the application requirements, parallel architectures must be employed. Fanout bottle-neck will also exist in parallel architectures.

Steps of Algorithim

set $a = t_0$; $b = t_1$;
set $\tilde{p}(x) = 1$; $\tilde{g}(x) = g(x)$;

loop: while $a - b < E$
{
$\tilde{g}(x) = \tilde{g}(x) + g(x)x^{b-a}$;
$\tilde{p}(x) = \tilde{p}(x) + x^{b-a}$;

$num = a - b$;
$b =$ highest power in $(\tilde{g}(x) - x^a)$;
}
final step:
$p(x) = \tilde{p}(x)x^{num}$;
$g'(x) = \tilde{g}(x)x^{num}$;

Simulation Results

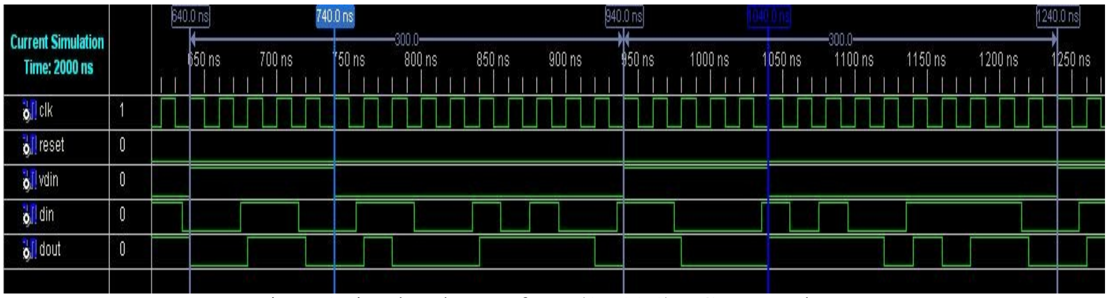


Figure1 simulated wave form (15,11,1) BCH Encoder

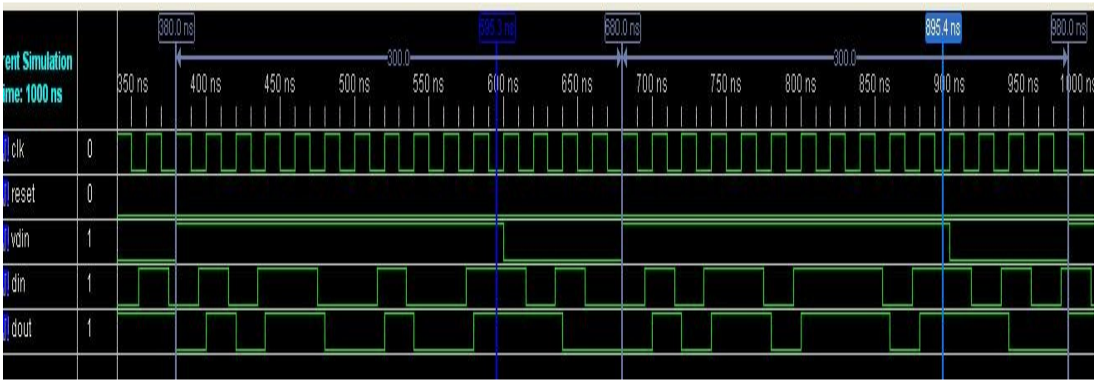


Figure 2 simulated wave form (15,7,2) BCH Encoder

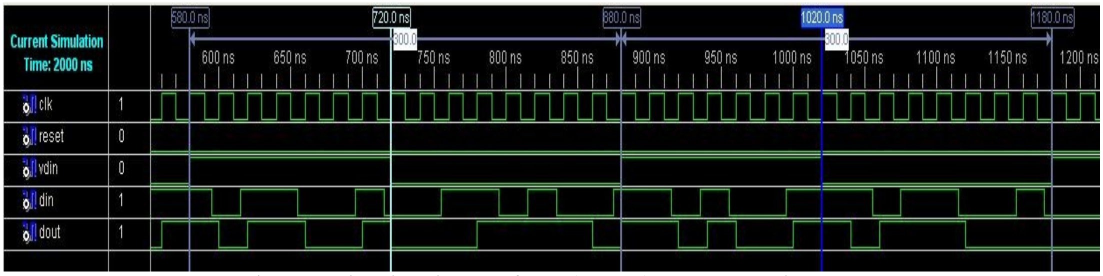


Figure 3 simulated wave form (15,5,3) BCH Encoder

Synthesis Report

Cell:in->out	(15,11,1)BCH	(15,7,2)BCH	(15,5,3)BCH
LUT3:I0->O	1	1	1
IBUF:I->O	3	5	7
FDC:D	5	3	2

V. 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.

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 andStorage. Upper Saddle River, New Jersey 075458: Prentice Hall, Inc,1995.
[6]. J. H. Derby, “High-speed CRC computation using state-space transformation” in Proc. Global Telecommunications Conf. 2001, GLOBECOM.
[7]. R. J. Glaise, “A two-step computation of cyclic redundancy code CRC-32 for ATM networks,” IBM J. Res. Dev., vol. 41, pp. 705–709
[8]. K. K. Parhi, VLSI Digital Signal Processing Systems-Design and Imple mentation. New York: Wiley, 1999.
[9]. 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.
[10]. Panda Amit K, Rajput P, Shukla B, “Design of Multi Bit LFSRPNRG and Performance comparison on FPGA using VHDL”, International Journal of Advances in Engineering & Technology (IJAET), Mar 2012, Vol. 3, Issue 1, pp. 566-571
[11]. Xilinx, Inc. Xilinx Libraries Guide, 2011. XessCorp.. XSA-3S1000 Board V1.1 User Manual. Available: http://xess.com/manuals/xsa-3S-manual-v1_1.pdf. Sept 2007.
[12]. J J.Rose S.D. Brown, R.J. Francis – “Field Programmable GateArrays”, Kluwer Academic Publishers, 1992