International Journal of Scientific Research and Review ISSN No.: 2279-543X
Volume 07, Issue 03, March 2019 UGC Journal No.: 64650

Implementation Of Error Correcting Codes In Digital Communication System Using Raspberry Pi

Vasimakram Mulla, Tukaram Arkasali, Sanjay L P,

Vidya K P, Madan H T

School Of Electronics and Communication Engineering, REVA University, India

ABSTRACT

In the present Digital Communication systems, it is highly possible that the data or message get corrupted during transmission and reception through a noisy channel medium. To get the error free communication we need Error correction code. Over the years a number of error detection and correction methodologies have been devised to send and receive the data in a consistent and correct form. The best of these methodologies ensure that the data is received correctly by the receiver in minimum number of retransmissions. BCH (Bose, Chaudhuri and Hocquenghem) codes form a large class of powerful random error-correcting cyclic codes. The project deals with implementation of BCH Codes in digital communication systems using raspberry pi boards.
A mechanism has to be in place that detects these errors in the received data and corrects it to get the data back as it was meant to be sent by the sender. To demonstrate two way secured communication , two raspberry pi boards will be used, one at the transmitter end for encoding and other one at the receiver end for decoding.

Keywords*: *BCH codes, Binary symmetric channel,Cyclic codes,Galois field,Linear block codes,RS codes.

1. INTRODUCTION

Whenever communication takes place, data needs to be transmitted, be it human beings or computers. Data is of utmost importance for an effective communication to take place, so the transmission of data should be such that the receiver of the data should receive the data in the same condition as it was sent by the sender. If the data is prone to any disturbance, then the data received by the receiver will not be the same as it was meant and thus it will convey a different information than what it was meant to convey. Thus the goal of data transmission is to transmit the data over a communication channel without any errors. Various techniques have been developed over the past few years to secure and make the data transmission reliable. One of such techniques is Information Coding Theory. Coding theory the study of codes, including error detecting and error correcting codes, has been studied extensively for the past forty years. It has become increasingly important with the development of new technologies for data communications and data storage. Coding theory makes use of various codes to encode the data for transmission over a channel and then the data is decoded at the receivers end to get the required data bits. These codes used to encode the data bits show variations over different communication channel's, that is the behavior of these codes may be different in different communication channels. The performance of some codes may be better than other codes over the same channel. This paper discusses the performance

2 ERROR CORRECTING CODES

2.1 BCH CODE

BCH codes forms a class of random multiple error-correcting cyclic codes. For any positive integer $m \geq 3$ and $t < 2m - 1$, there exists a binary BCH code with the following parameters,
Block length: $n = 2m - 1$
Number of parity-check digits: $n - k \leq mt$
Minimum distance: $dmin \geq 2t + 1$.

BCH codes are subset of the Block codes. In block codes, the redundancy bits are added to the original message bits and the resultant longer information bits called "codeword" for error correction are transmitted. The block codes are implemented as (n, k) codes where n indicates the codeword and k the original information bits. The generator polynomial g(x) of the t-error-correcting BCH code of length 2m -1 is the lowestdegree polynomial over GF(2) which has α, α1, α2, . . , α2t as its roots. Let Φ(X) be the minimal polynomial of αi. g(X) is the least common multiple (LCM) of Φ1(X), Φ2(X),.., Φ2t(X). Let α be the primitive element of GF(26) for BCH (63, 51, t = 2) codes. Hence 1 + α + α6 = 0 is the primitive polynomial. The generator polynomial (GP) is given by

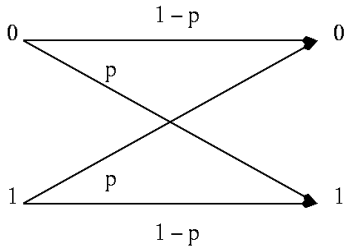
$$g(x)=LCM(\Phi 1(X), \Phi 2(X))$$

$$g(x) = (1 + x + x^6)(1 + x + x^2 + x^4 + x^6)$$

$$g(x) = (1 + x^3 + x^4 + x^5 + x^9 + x^{10} + x^{12})$$

2.2 Binary Symmetry Channel

A symmetric channel which has two inputs and two outputs is known as binary symmetric channel.



binary (0,1)
symmetric p(0→1) =p(1→0)

2.3 Cyclic Codes

A cyclic code is a linear code that has the property that a "cyclic shift" on any of its codewords produces another codeword.The cyclic shift may be leftwords or rightwords by any number of places.

C=(Cn-1,Cn-2_ _ _ _C1,C0)

Left shift
By one place
C1=(Cn-2,Cn-3_ _ _C1,C0,Cn-1)

Right shift
By four places
C2=(C2,C1,C0_ _ _C4,C8)

Hamming codes are cyclic codes.

2.3.1 Cyclic codes in polynomial.
For encoding and decoding process we are using cyclic codes.
P(x)=an-1x^n-1+an-2x^n-2+_ _ _ _ _+a1x+a0

2.4 Galois Field (Finite field)

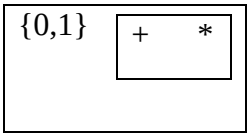
"Galois showed that for a field to be finite,the number of elements should be p^n,where p is prime and is a positive integer."The galois field ,Gf(p^n) is a finite field with p^n elements. When n=1 we have Gf(p) field, this field can be the set Zp,{0,1_ _ _ _p-1} with arithmetic operations.

Two arithmetic operations

*addition

*Multiplication

A very common field in this category is Gf(2) with the set {0,1} and operation addition and multiplication.



+	0	1
0	0	1
1	1	0

x	0	1
0	0	0
1	0	1

2.5 Linear block codes

A block code is said to be "linear code" if its codewords satisfy the condition that sum of any two codewords gives another codeword. i.e Ck=Ci+Cj

2.5.1 Properties

- The all_zero word (0 0 0_ _ _ 0) is always codeword(all_zero codeword is also a codeword.
- Given any three codewords Ci,Cj and Ck such that

  Ck=Ci+Cj then d(Ci,Cj)=W(k)

  C11=C1+C10

  d(C1,C10)=W(k)

  Where d(C1,C10)-distance

  W(k)-weight of C11

2.6 Reed-solomon codes

It is a subclass of non_binary BCH codes. The encoder for an RS code differs from a binary encoder in that it operates on the multiple bits rather than individual bits.

2.6.1 Properties

- Block length n=2^m-1 symbols
- Message size k symbols
- Parity check size (n-k)=2^t symbols
- Minimum distance dmin=2t+1

3. BCH ENCODER AND DECODER

3.1 BCH Encoder

The codewords are formed by adding the remainder after division of message polynomial with generator polynomial. All codewords are the multiples of generator polynomial. At the encoding side, the generator polynomials are not usually split as it will demand more hardware and control circuitry. The polynomial is used as such for encoding. The generator polynomial for BCH is given by

$$g(x) = (1 + x^3 + x^4 + x^5 + x^9 + x^{10} + x^{12})$$

The parity bits are obtained by computing the remainder polynomial r(x) as:

$$r(x) = \sum_{i=0} r_i x^i = x^{12} m(x) \bmod g(x)$$

where m(x) is the message polynomial.

$$m(x) = \sum_{i=0}^{50} m_i x^i$$

$r_i = 0,\ldots, 11$ and $m_i = 0,\ldots,50$ are elements of GF(64). m_{50} is the first message bit to be transmitted and m_0 is the last message bit, may be a shortened bit. The parity bits follow the order as follows: r11 first, followed by r10 and so on.

BCH codes are implemented as systematic cyclic codes. Hence can be easily implemented and the logic which implements encoder and decoder is controlled into shift register circuits. The remainder r(x) can be calculated in the (n-k) linear stage shift register with the feedback connection at the coefficient of generator polynomial. BCH codeword are encoded as follows:

During the 1st clock cycle m_{50} is given as input to the shift register. LFSR is initialized with seed value 0. From 1 to k clock cycles all the 51 messages bits are transmitted and the linear feedback shift register calculates the parity bits. The parity bits generated in the linear feedback shift register are taken in parallel. Thus the 12 parity bits are appended to the message bits. The serial in parallel out BCH encoder architecture is shown in the Figure 1.

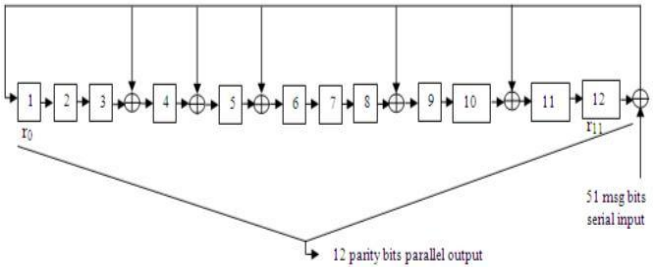


Figure 1. Block diagram of BCH Encoder

3.2. BCH Decoder

Determining where the errors are in a received codeword is a rather complicated process. Figure 2 shows the block diagram of BCH decoder. The decoding process for BCH codes consists of four major steps

1. Syndrome computation
2. Determine coefficients of error locator polynomial
3. Find the roots of error locator polynomial and it will indicate the erronous bits in the received codeword.4. Error correction

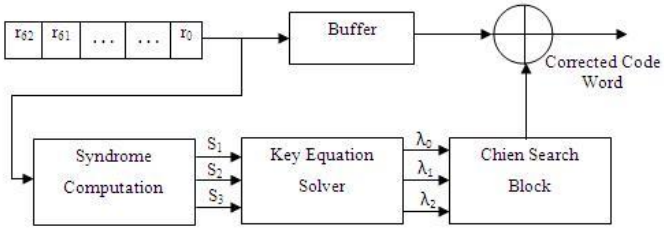


Figure 2. Block diagram of BCH Decoder

3.3. Syndrome Computation

The first step at the decoding process is to store the received codeword in a buffer and then calculate the syndromes. The input to the syndrome module is the received codeword. The received polynomial may be corrupted with error pattern e(x) as shown.

$$r(x) = c(x) + e(x)$$

Where the received codeword is

$$r(x) = r_0 + r_1 x + ... + r_{n-1} x^{n-1}$$

Transmitted codeword is given by

$$c(x) = c_0 + c_1 + ... + c_{n-1} x^{n-1}$$

The error pattern is

$$e(x) = e_0 + e_1 + ... + e_{n-1} x^{n-1}$$

Syndrome S_i can be computed by :

$$S_i = r(\alpha^i) = r_0 + r_1 \alpha^i + r_2 \alpha^{2i} + r_3 \alpha^{3i} + ... + r_{n-1} \alpha^{(n-1)i}$$

where $1 \le i \le 2t - 1$.

For BCH the three syndromes are:

$$S_1 = r(\alpha^1) = r_0 + r_1 \alpha^1 + r_2 \alpha^2 + r_3 \alpha^3 + ... + r_{n-1} \alpha^{62}$$

$$S_2 = r(\alpha^2) = r_0 + r_1 \alpha^2 + r_2 \alpha^4 + r_3 \alpha^6 + ... + r_{n-1} \alpha^{124}$$

$$S_3 = r(\alpha^2) = r_0 + r_1 \alpha^3 + r_2 \alpha^6 + r_3 \alpha^9 + ... + r_{n-1} \alpha^{186}$$

where α is the primitive element in GF(26)

If there is no error in the received codeword then syndromes generated will be zero. Since the syndromes only depends on the error polynomial, and if the syndromes are nonzero then next step is to find out the coefficients of error locator polynomial.

3.4. Error Locator Polynomial Coefficients

The error locator polynomial is defined as:

$$\lambda(\alpha^j) = \lambda_0 + \lambda_1 \alpha^j + \lambda_2 \alpha^{2j}$$

In this paper Inversion-less Berlekamp Massey Algorithm (BMA) is used for finding the coefficients λ0, λ1 and λ2.

$$\lambda_2 = S_3 + S_1 S_2$$
$$\lambda_2 = S_1 S_1$$
$$\lambda_0 = S_1$$

The hardware implementation of syndrome and BMA is shown in the Figure 3. The efficient implementation of Galois field polynomial multiplication is an important prerequisite in BCH decoding process. We are implementing a new method in which MSE (Most Significant element) approach is used for finding the partial products. Let a(α) and b(α) be two poynomials in GF(2^6):

$$a(\alpha) = a_5 \alpha^5 + a_4 \alpha^4 + a_3 \alpha^3 + a_2 \alpha^2 + a_1 \alpha^1 + a_0 \alpha^0$$

$$b(\alpha) = b_5 \alpha^5 + b_4 \alpha^4 + b_3 \alpha^3 + b_2 \alpha^2 + b_1 \alpha^1 + b_0 \alpha^0$$

$$y(\alpha) = a(\alpha) * b(\alpha) = y_5 \alpha^5 + y_4 \alpha^4 + y_3 \alpha^3 + y_2 \alpha^2 + y_1 \alpha^1 + y_0 \alpha^0$$

Using the primitive polynomial $1 + \alpha + \alpha^6$, the partial products will be as follows:

$$y_5 = a_5 (b_5 + b_0) + a_4 b_1 + a_3 b_2 + a_2 b_3 + a_1 b_4 + a_0 b_5$$

$$y_4 = a_5 (b_4 + b_5) + a_4 (b_5 + b_0) + a_3 b_1 + a_2 b_2 + a_1 b_3 + a_0 b_4$$

$$y_3 = a_5 (b_3 + b_4) + a_4 (b_5 + b_0) + a_3 b_1 + a_2 b_2 + a_1 b_3 + a_0 b_4$$

$$y_2 = a_5 (b_2 + b_3) + a_4 (b_3 + b_4) + a_3 (b_4 + b_5) + a_2 (b_5 + b_0) + a_1 b_1 + a_0 b_2$$

$$y_1 = a_5 (b_1 + b_2) + a_4 (b_2 + b_3) + a_3 (b_3 + b_4) + a_2 (b_4 + b_5) + a_1 (b_5 + b_0) + a_0 b_1 \; y_0 = a_5 b_1 + a_4 b_2 + a_3 b_3 + a_2 b_4 + a_1 b_5 + a_0 b_0$$

The proposed method provides a fast multiplication algorithm which uses lesser number of 'and' and

'xor' gates.

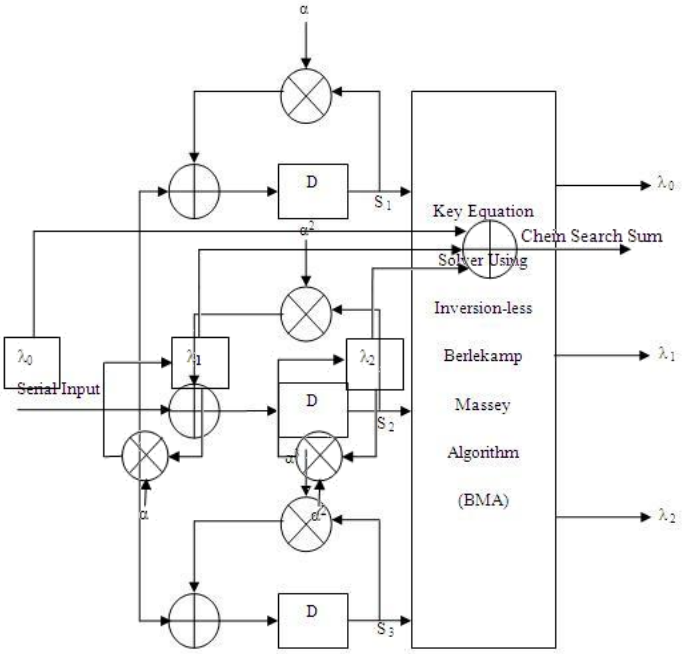


Figure 3. Implementation of Syndrome Calculator and Coefficient Solver

3.5. Finding Roots of Error Locator Polynomial

The next step in decoding process is to figure out the roots of the error locator polynomial. Chien search is an efficient algorithm for determining roots of polynomial defined over a finite field. In GF(2^6) all the 64 elements are examined to find whether it is a root or not. The hardware implementation of Chien search block is shown in Figure 4.

Figure 4. Implementation of Chien Search Block

The Chien search sum is given by:

$$\lambda_0 + \lambda_1 \alpha^{-j} + \lambda_2 \alpha^{2j}$$

where $1 \leq j \leq n$.

Let j be the clock cycle and if the Chien search sum equals to zero then α^j will be a root of the polynomial. Error location numbers are the reciprocal of the roots. In order to find the error positions in the received codeword a counter is initialized with the Chien search. Therefore, if Chien search sum is zero it will indicate a root and magnitude of the root will be the counter value. Let α^j be a root, and then error position can be obtained by finding the inverse in the finite field.

$$\alpha^{-j} = \alpha^{-j} * \alpha^{n}$$

3.6. Error Correction

The final step in the decoding process is error correction. The received codeword is stored in a buffer register until it is corrected. The erroneous bits can be corrected by simply flipping the bits at the error positions. Then output of the BCH decoder is the corrected codeword.

4. Methodology

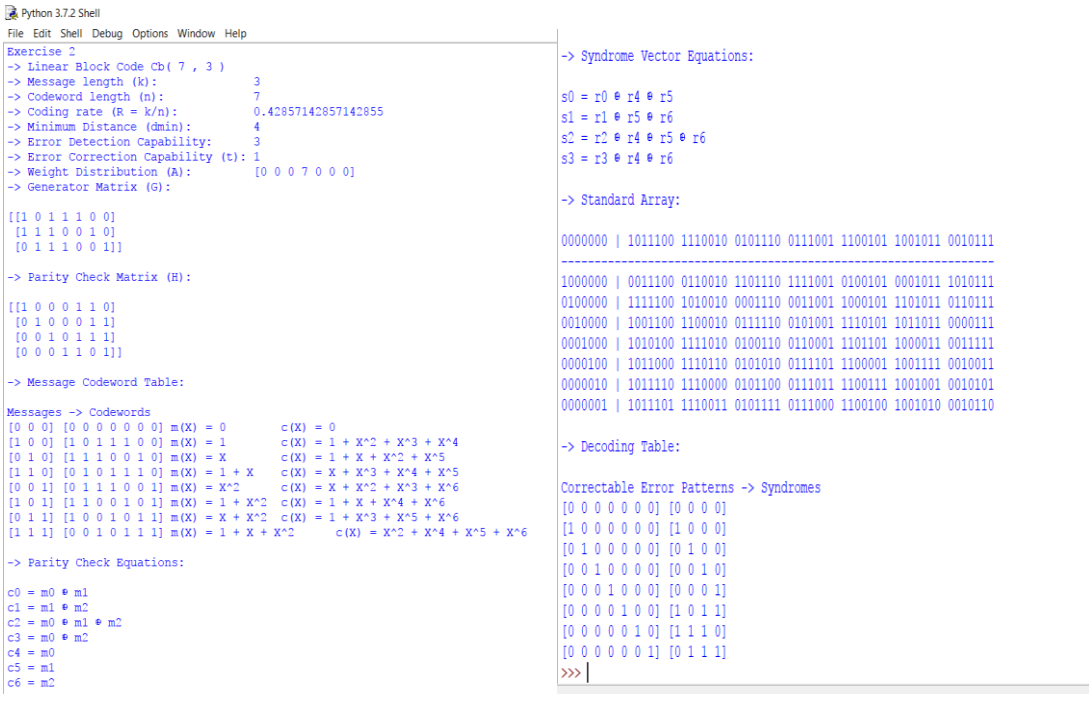
We are demonstrate the concept of BCH code, in this we are trying to implement these code in to practical world by using raspberry pi. So for this code we are extracting the encoding and decoding process mathematically. Completion of mathematical representation for this we converting all equations and properties of BCH codes in Python 3.7.2. After the coding part is done the simulation takes place both encoding and decoding by auto error detecting and correcting process. After successful simulation the python code is dump into the raspberry pi. A mechanism has to be in place that detects these errors in the received data and corrects it to get the data back as it was meant to be sent by the sender. To demonstrate two way secured communication , two raspberry pi boards will be used, one at the transmitter end for encoding and other one at the receiver end for decoding.

5. Hardware and Software Requirements:

Hardware	Software
Raspberry Pi 3	Python 3.7.2

6. Simulation Results

Below figure shows the simulation results of BCH encoder and decoder respectively. If the transmitted and the received codewords are the same then the syndromes will be zero. Here in this case the received codeword as erroneous is discussed. The received bit encoded data given as input to the syndrome calculation circuit. Due to the presence of error the syndrome value will be a non- zero. Once the error is detected, it is corrected using BMA algorithm and Chien search algorithm as discussed.



7. CONCLUSION

To ensure reliable transmission of information through a physical medium or wireless medium, error control coding are used in the digital information and communication systems. In this paper we have presented the implementation of (7,3 t = 2) BCH encoder and decoder it was successfully done using Python 3.7.2 and design implemented on Raspberry Pi.It can be detected and corrected.. The proposed Galois field polynomial multiplication is used for the syndrome calculation as well as for finding the error locating polynomial coefficients. It allows fast field multiplication. BCH code forms a large class of powerful random error-correcting cyclic codes. They are relatively simple to encode and decode. Further, the performance can be improved by adopting parallel approach methods.

REFERENCES

1) Michelle Effros, Andrea Goldsmith, Yifan Liang, "Generalizing Capacity: New Definitions and Capacity Theorems for Composite Channels", IEEE Transactions on Information Theory, vol. 56, no. 7, pp. 3069-3087, July 2010.
2) Wilfried Gappmair," Claude E. Shannon: The 50th Anniversary of Information Theory", IEEE Communications Magazine, pp. 102-105, April 1999.
3) Shyue-Win Wei, Che-Ho Wei, "High-speed hardware decoder for double-error-correcting binary BCH codes", IEEE Proceedings, vol 136, no. 3, pp. 227-231, June 1989.
4) Sana Ullah and Mohammed A. Alnuem, "A Review of IEEE 802.15.6 MAC, PHY, and Security Specifications", Hindawi Publishing Corporation International Journal of Distributed Sensor Networks, article ID 950704, pp:1-12, 2013.
5) Laurencin Mihai Ionesco, Constantin Anton, "Hardware Implementation of BCH Error-Correcting Codes on a FPGA", International Journal of Intelligent Computing Research, vol 1, Issue 3, June 2010.
6) P. Reviriego, C. Aggrades, J. A. Maestro, "Efficient error detection in Double Error Correction BCH codes for memory applications", Microelectronics Reliability, pp. 1528–1530,2012.
7) IEEE Computer Society, "IEEE Standard for Local and metropolitan area networks: Part 15.6 Wireless Body Area Networks," IEEE Standards Association, 29, Feb., 2012.
8) Kyung Sup Kwak, Sana Ullah, and Niamat Ullah, "An Overview of IEEE 802.15.6 Standard", IEEE Trans. 2012.
9) R. Lidl and H. Niederreiter, "Finite Fields", Cambridge University Press, Cambridge, 1966.
10) John Gill, "Finite Fields", Stanford University.