Implementation of Designed Encoder and Decoder for Golay Code

vaibhav chandrakar[1],Mr. sanjeev shrivastava[2], Dr. Mohit gangwar[3], Mr. Suresh Gawande[4]

Abstract— *In the wireless communication system the receiver has the ability to detect and correct the error from the received information. Receiving the error is an important issue, so it provides the processor for correcting the information of the data. There are some different methods for the implementation of the hardware and software. The communication distance between transmitter and receiver play an important role because the transmission of data between transmitter and receiver depends on length of communication. Multiple bits of information transmitted from transmitter changed due to the effect of noise in transmitted signal. This causes intense loss in many of the cases. This paper presents the brief explanation of FPGA (Field Programmable Gate Array) based simulation and design of Golay Code (G23) and extended Golay code (G24) Encoding scheme. For encoding the data packet this paper use the Golay Encoder for the optimization of the time delay of the operational circuit design to encode the data packet.*

I. INTRODUCTION

In digital transmission, bit errors were begin due to the change of bits at the time of transmission of data over communication channels and the bit has been changed due to noise, distortions, interference, or bits synchronization error. This can be explaining by taking an example of transmitted bits sequence as follows:

0	1	1	0	1	0	1	0	1	1

And the following received bit sequence:

0	0	1	1	1	1	1	0	0	1

Figure 1.2- Transmitted and Received bit sequence

The number of bit error (the colored bit) in this case, 4. The Bit Errors Rate is 4 incorrect bits divided by 10 reassigns the bits, resultant in a BER of 0.4 or 40%. Here the description of bits error rate (BER) is the number of bits error per unit time. The bits error ratio (also BER) is defined as the number of bit errors divided by the total numbers of transferred bits during a considered time interval. BER is expressed in percentage.

In digital communication systems, BER at the receiver side may be affected by transmission channel noise, distortion, interference, attenuation, bit synchronization problems, wireless multipath fading, etc. By choosing strong signal strength the BER may be improved (unless this causes more bit errors and cross-talk), by choosing a robust modulation scheme and slow or line coding scheme, and by applying channel coding schemes such as redundant forward error correction codes. An efficient hardware implementation of encoding the algorithm in field FPGA prototype for both binary Golay code ($g23$) and extended binary golay code ($g24$). The high speeds architecture with low latency have been designed and implemented in a virtex-4 which are explained in reference [1]. This hardware's module for both the encoder and decoder may be good candidates for different application in high speed communication link, photo spectroscopy, and ultrasonography.

II. GOLAY CODE

A binary Golay code is represented by (23, 12, 7) shows length of the codeword is 23 bit while message is shown by 12 bits and the minimum distance between two binary Golay code is7. Build a binary codes in Galois field denoted by GF(2) which support the different binary operation. The coding sequence is generated by Generator polynomial [13] over GF(2) for Golay code (23, 12, 7) code are x11 + x10 + x6 + x5 + x4 + x2 + x1 and x11+ x9+ x7+ x6+ x5+ x1+ 1

This paper shows AE3h as characteristic polynomial which is of 12 bit binary number can be encoded into a 23-bit Golay code by using the long division method to generate check bits (11-bit) information and 12 bit data together make the Golay code of 12-bit information data. A extended Golay code (24, 12, 8) produced by adding a parity bit with the binary Golay code or using as a generator matrix G. Matrix G is defined by [I, B] or [I, B] where I is shown as Identity matrix and B is shown as Matrix. The matrix B is shown in figure below-

$$B = \begin{vmatrix} 110111000101 \\ 101110001011 \\ 011100010111 \\ 111000010111 \\ 110001011011 \\ 100010110111 \\ 000101101111 \\ 001011011101 \\ 010110111001 \\ 101101110001 \\ 011011100011 \\ 111111111110 \end{vmatrix}$$

Figure 3.1-Matrix-B

The steps of algorithm required to achieve the encoding procedure as follows-

A characteristic polynomial is preferred for check bits generation. 'M' bit of data contribute in long division method with the characteristic polynomial. So 11 zero are appended to the right of the m bit of data. Checks bit for G(23) are achieved by MSB result at the end of the long division method. By appending the check bits with the message the encoded Golay code (23, 12, 7) codeword are obtained. A parity bit is added for conversion of binary Golay code into extended binary Golay code (24, 12, 8). If the weight of binary Golay codes are odd then parity bits 1 is appended, otherwise 0 is appended.

III. ALGORITHM, PROPOSED ARCHITECTURE, AND COMPARISON OF RESULTS FOR GOLAY ENCODER

A. Proposed Algorithm for Encoder

The steps required to accomplish the encoding procedure are enlisted as follows.

1) A characteristic polynomial $G(x)$ is chosen for check bits generation.

2) 11 zeros are appended to the right of message $M(x)$, such that resultant polynomial $P(x)$ participates in long division process with $G(x)$.

3) The remainder bits except the most significant bit (MSB) resulted at the end of the division operation are the check bits for G_{23}. Appending check bits with the message gives us the encoded Golay (23, 12, 7) codeword.

4) A parity bit is added to convert the binary Golay code into extended binary Golay code (24, 12, 8). If the weight of binary Golay code is even, then parity bit 0 is appended, otherwise 1 is appended.

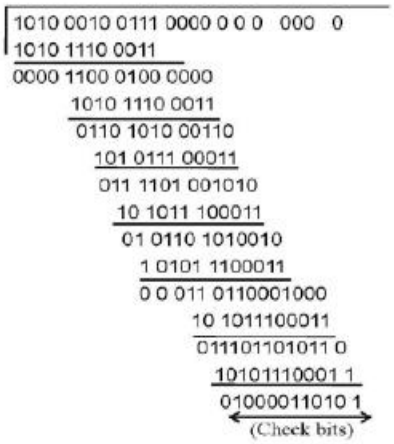

Fig. 3.1. Example of check bits generation.

The proposed encoder algorithm clearly follows the basic CRC generation process and includes a method for converting binary Golay code to extended Golay code before proceeding for designing architecture. An example of Golay code word generation based on the above mentioned algorithm is shown in Fig. 1. Let us say, the message to be encoded is A27h. Hence, $M(x)$ = A27h and $P(x)$ in binary format is represented as 1010 0010 0111 0000 0000 000. Finally, the generated check bits in hexadecimal format are 435h. Hence, the encoded codeword for the message bits (A27h) is A27435h. This is a binary Golay code word. To convert it into an extended Golay code, a parity bit 1 is appended, as weight of A27435h is 11 (odd). Finally, the generated Golay (24, 12, 8) codeword is (1010 0010 0111 10000110 101 1). The validity of the generated Golay code can be tested by measuring the weight of the code. The weight of every $G24$ code should be a multiple of four and greater than equal to eight. The generated code word shown in the example has a weight of 12, which is a multiple of four and greater than eight. Hence, the generated code is valid and thus the algorithm.

B. Proposed Architecture for Binary Golay Code and Extended Golay Code

In this section, optimized architectures for encoder are shown in Figs. 2 and 3. The whole architecture is partitioned into two parts, one for $G23$ generation and the other showing conversion of $G23$ to $G24$. In each step during polynomial division, simply binary XOR operation occurs for modulo-2 subtraction. The residual result obtained at each step during the division process is circularly left shifted by number of leading zeros present in the result. A 12:4 priority encoder is used to detect efficiently the number of leading zeros before first 1 bit in the residual result in each step. A circular shift register is used to shift the intermediate result by the output of priority encoder. A 2:1 multiplexer is used to select the initial message or the circularly shifted intermediate result. The control signal used for the multiplexer and the controlled subtractor is denoted as p, which is bit wise OR operation of priority encoder output. A controlled subtractor is used for loop control mechanism. Initially, one input of subtractor is initialized with 11, which is the number of zeros appended in the first step of the long division process and it gets updated with the content of $R7$ register due to multiplexer selection after each iteration. The output of the priority encoder is the other input to the subtractor. After the final iteration, the result of subtractor is zero, which is stored in register $R7$. The register $R6$ is loaded when the content of register $R7$ becomes zero, which depicts the end of the division process and hence the check bits generation process. The Ld control signal controls the loading of the $R3$ [10:0]

Data (12-bit)	Appended zeros	Operation
101000100111	00000000000	
101011100011		… xor
000011000100	0000	… shift
10101110	0011	… xor
01101010	00110	… shift
1010111	00011	… xor
0111101	001010	… shift
101011	100011	… xor
010110	1010010	… shift
10101	1100011	… xor
00011	0110001000	… shift
10	1011100011	… xor
01	11011010110	… shift
1	01011100011	… xor
0	10000110101	
<Check-bits>		

Figure 3.2- Long –Division of data for check bit Generation

As shown in the above figure 10, the characteristic polynomial 101011100011 and 12-bit data is 101000100111. The 11- checks bit sequences are generated by long division method is 10000110101. The 23-bits encoded Golay codeword (G23) is 101000100111-10000110101. A parity bit is added for the conversion of binary Golay code into the extended binary Golay code. In the G(23) word the weight is 11, i.e., the encoded word has 11 1‟s, so a 1 will be appended in it. This will generate extended codeword G(24) as (101000100111-10000110101-1). Generation of parity bits is implemented by XOR operation of the bits of G(23) codeword.

In verified G(24) codeword the weight is multiple of 4 and greater than equal to 8. The weight of the G(24) codeword is 12, so it is a valid codeword.

S[11] = w[23] xor w[11] xor w[10] xor w[8] xor w[7] xor w[6] xor w[2] xor w[0]

S[10] = w[22] xor w[11] xor w[9] xor w[8] xor w[7] xor w[3] xor w[1] xor w[0]

S[9] = w[21] xor w[10] xor w[9] xor w[8] xor w[4] xor w[2] xor w[1] xor w[0]

S[8] = w[20] xor w[11] xor w[10] xor w[9] xor w[5] xor w[3] xor w[2] xor w[0]

S[7] = w[19] xor w[11] xor w[10] xor w[6] xor w[4] xor w[3] xor w[1] xor w[0]

S[6] = w[18] xor w[11] xor w[7] xor w[5] xor w[4] xor w[2] xor w[1] xor w[0]

S[5] = w[17] xor w[8] xor w[6] xor w[5] xor w[3] xor w[2] xor w[1] xor w[0]

S[4] = w[16] xor w[9] xor w[7] xor w[6] xor w[4] xor w[3] xor w[2] xor w[0]

S[3] = w[15] xor w[10] xor w[8] xor w[7] xor w[5] xor w[4] xor w[3] xor w[0]

S[2] = w[14] xor w[11] xor w[9] xor w[8] xor w[6] xor w[5] xor w[4] xor w[0]

S[1] = w[13] xor w[10] xor w[9] xor w[7] xor w[6] xor w[5] xor w[1] xor w[0]

S[0] = w[12] xor w[11] xor w[10] xor w[9] xor w[8] xor w[7] xor w[6] xor w[5] xor w[4] xor w[3] xor w[2] xor w[1]

Figure 3.3- Syndrome ‘S’ Bit-Generation Logic Equations

3.2 CHARACTERISTIC OF ENCODER THAT WORK WITH POLYNOMIAL

In digital codes, the error detection and correction abilities of the polynomial code are determined by using Hamming distance of the codes techniques. The polynomial codes are generally linear codes. The minimum Hamming distance is equal to the minimum weight of non-zero codeword’s. As example, the minimum Hamming distance is 2, 01001 are codeword and there are no nonzero codeword’s with only one bit set.

Some specific properties of polynomial codes are often depend on particular algebraic properties of its generator polynomial. Some examples of such properties:

- A polynomial codes is a cyclic code if and only if the generator polynomials divides.

- If generator polynomial is a primitive, then the resultant code of Hamming distance is at least '3'.
- In BCH code, the generator polynomial is selected to have exact root in an additional field, in this way it achieves the high Hamming distance.

The algebraic natures of polynomial code, by chosen the generator polynomial, may also frequently be exploited to find the efficient errors correction algorithm. This is the case for BCH codes.

3.3 ENCODED BINARY SEQUENCE GENERATION

The parity encoding the binary sequences is basically the inverse operation of differentiation of binary sequences; partition the sequence into the cycle of sequences. Then the spectrum of these cycle is shown as a fractal-like semi-infinite sequences of the powers of 2, the properties that truncating it at the $(n+1)$st term yields the cyclic spectrum for the parity encoding sequence generation of the n-bit binary sequences. An easy and complete characterization of both cycles and of the cyclic spectrum is given.

3.4 GOLAY CODE DECODER

To calculate the error in the data the error decoding process is used for the calculation of weight of (S + Bi) and (SB + Bi). To calculate (S + Bi), for $1 \le i \le 12$, the implemented hardware use bit inversion of „S" as per the logic as shown in figure below-

The steps of algorithm are required to achieve the decoding process are enlisted as follows:

1) For the received codeword 'W' and matrix 'H', where H = [I / B] Compute the Syndrome 'S'.
2) Error vector, E = [S, 0], If weight of 'S' is less than or equal to 3, i.e., wt(S) ≤ 3.
3) If wt(S+Bi) ≤ 2, then E = [S+Bi, Ii]. Where Ii represents ith row of the identity matrix I.
4) The second syndrome SB can be computed.
5) If wt(SB) ≤ 3, then E = [0, SB].
6) If wt(SB+Bi) ≤ 2, then E = [Ii, S+Bi].

If E is still not determined then received data is required to be retransmitted.

S + b1 = { ~S[11], ~S[10], S[9], ~S[8], ~S[7], ~S[6], S[5], S[4], S[3], ~S[2], S[1], ~S[0] }

S + b2 = { ~S[11], S[10], ~S[9], ~S[8], ~S[7], S[6], S[5], S[4], ~S[3], S[2], ~S[1], ~S[0] }

S + b3 ={ S[11], ~S[10], ~S[9], ~S[8], S[7], S[6], S[5], ~S[4], S[3], ~S[2], ~S[1], ~S[0] }

S + b4 ={ ~S[11], ~S[10], ~S[9], S[8], S[7], S[6], ~S[5], S[4], ~S[3], ~S[2], S[1], ~S[0] }

S + b5 ={ ~S[11], ~S[10], S[9], S[8], S[7], ~S[6], S[5], ~S[4], ~S[3], S[2], ~S[1], ~S[0] }

S + b6 ={ ~S[11], S[10], S[9], S[8], ~S[7], S[6], ~S[5], ~S[4], S[3], ~S[2], ~S[1], ~S[0] }

S + b7 ={ S[11], S[10], S[9], ~S[8], S[7], ~S[6], ~S[5], S[4], ~S[3], ~S[2], ~S[1], ~S[0] }

S + b8 ={ S[11], S[10], ~S[9], S[8], ~S[7], ~S[6], S[5], ~S[4], ~S[3], ~S[2], S[1], ~S[0] }

S + b9 ={ S[11], ~S[10], S[9], ~S[8], ~S[7], S[6], ~S[5], ~S[4], ~S[3], S[2], S[1], ~S[0] }

S + b10 ={ ~S[11], S[10], ~S[9], ~S[8], S[7], ~S[6], ~S[5], ~S[4], S[3], S[2], S[1], ~S[0] }

S + b11 ={ S[11], ~S[10], ~S[9], S[8], ~S[7], ~S[6], ~S[5], S[4], S[3], S[2], ~S[1], ~S[0] }

S + b12 ={ ~S[11], ~S[10], ~S[9], ~S[8], ~S[7], ~S[6], ~S[5], ~S[4], ~S[3], ~S[2], ~S[1], S[0] }

Figure 3.4- Calculation of (S + Bi)

A simplified adder based architecture is implemented to calculate the weight of a 12-bit (S + Bi) and (SB + Bi), for $1 \le i \le 12$. Architecture of Weight Calculation Unit is shown in figure below-

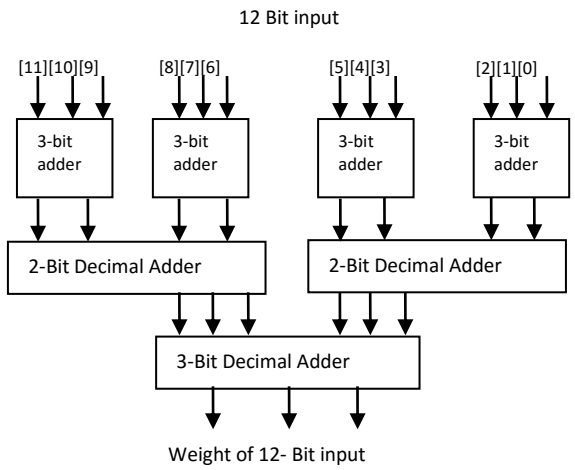

Figure 3.5- Architecture of Weight Calculation Unit

3.5 SYNDROME CALCULATION

Suppose the code word (x0, x1, x2 …vx-1) is transmitted over a noisy channel resulting in the received word (y_0, y_1…y_{n-i}), we recall that the first step in decoding of a linear block code is to calculate the syndrome for the received word. If the syndrome is zero, there are no transmission error is received in word. On other side if the syndrome is non zero, then the word received is non-zero words which contain transmission errors that required correction in the words.

In systematic form of cyclic code the syndrome may be calculated easily. The received words be given in the polynomial of degree n-1 or less, as shown by

$$y(D)= y_0 + y_1D +....+ y_{n-1}D^{n-1} \qquad (1)$$

let a(D) denote the quotient and s(D) denote the remainder, which are result of dividing y(D)by generator polynomial g(D). y(D) is expressed as-

$$y(D)= a(D)g(D) + s(D) \qquad (2)$$

The remainder s(D) is a polynomial of degree n-k-1 or less. Is also known as polynomial syndrome in that (n-k)-by-1 syndrome s. If the syndrome polynomial s(D) is nonzero, then the presence of transmission error in the received word is detected.

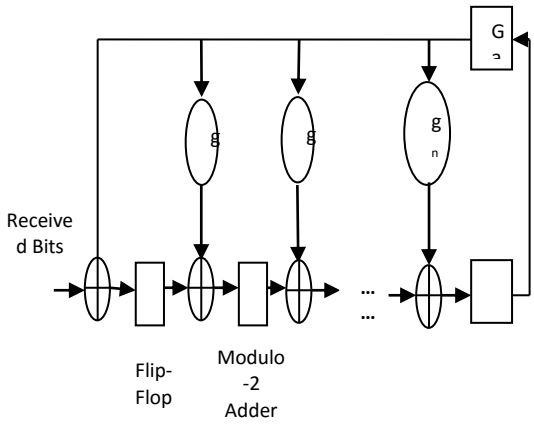

Figure 3.6- Syndrome Calculation

The above figure shows syndrome calculation that is identical to the encoder except the fact of received bits and fed to the (n-k) stages of the feedback shift register from left so the all received bits have been shifted to the shift register its contents desired state syndrome s. We know s, we can determine the corresponding error pattern e and thereby make the appropriate correction.

3.6 ERROR DETECTION

When we transmit a message it may get affected by noise or we can say that may get corrupted. We use error – detecting code in which some additional bits of data can be added to the message which helps to detect the error in the transmission of the message. This method is very simple for detecting or correcting the error. The MSB of an 8-bits word is used as parity bit and the other 7 bits are used for data or message bits. The transmitted 8 bits word can be either even or odd parity bits. The addition of parity bit in a 7-bit data is shown in figure.15.

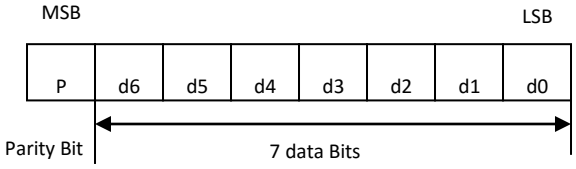

Figure 3.7- 8 Bit Parity

Even parity -- Even parity means the number of 1's in the given word including the parity bit should be even (2,4,6,....).

Odd parity -- Odd parity means the number of 1's in the given word including the parity bit should be odd (1,3,5,....).

Depending on the type of parity it can be set to either 0 or 1 as it required.

- For the even parity the bits are set to either 1 or 0 so that the number of "1" in the word is even. Shown in figure. 16(a).
- For the odd parity, the bits are set to either 1 or 0 so that the number of "1" in the word is odd. Shown in figure. 16(b).
-

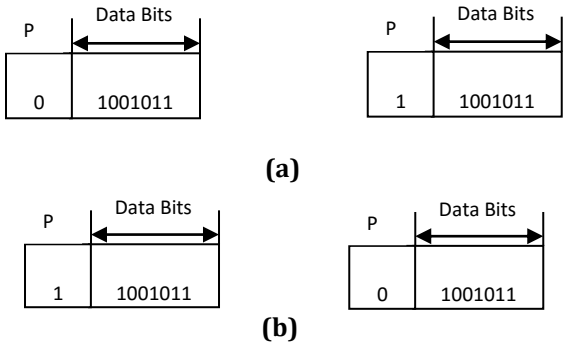

Figure 3.8- (a) Even Parity (b) odd parity

At receiver we may detect the rate of an error if parity of received signal is different from the expected parity. That means, if the parity of transmitted signals is "even" and if the received signals has an odd parity, then the receiver terminates the received signal is not correct. Then if an error is detected, the receiver ignores the received byte. Then it requested for retransmission of the same byte to the transmitter.

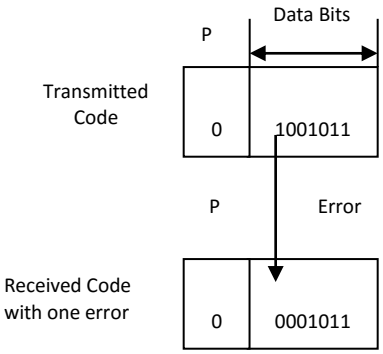

Figure 3.9- Transmitted and received code

3.7 ERROR CORRECTION

For the error correction of the data we use some error-detecting codes. That code may add some data bits to find out the original messages from the corrupt messages which we are received. These types of codes are known as error-correcting codes. Error-correcting codes may employ same methods as the error-detecting codes but additionally, these codes also detect the exact location of the corrupted bits. In error-correcting codes technique, the parity check has simple ways to detect the errors along with a complicated method to determine the corrupt bits location. When the corrupt bits are located, then its value is reverted from 0 to 1 or 1 to 0 to get the original data as in input.

An error-correcting code is a process of adding the redundant data bits, or parity data bits, to a message, so it can be recovered by a receiver even when a number of errors were introduced in that message during the process of transmission. Then the receiver asks to the sender for retransmission of the data. Error-correcting codes are mainly used in lower-layer communication of the signal, as well as for reliable storage in media such as CDs, DVDs, hard disks, and RAM Conventional design.

The FPGA based hardware utilization summary of the proposed Encoder and Decoder designs is presented in Table-5.1 and Table-5.2 respectively.

TABLE 5.1

HARDWARE UTILIZATION SUMMARY OF ENCODER

Vertex-IV XC4VLX160-12FF1148	Total	12-bit Golay Encoder	
		Used	%
Slices	67584	45	0
Flipflops	135168	44	0
LUTs 4-Inputs	135168	65	0
Bonded IOBs	768	39	6

TABLE 5.2

HARDWARE UTILIZATION SUMMARY OF DECODER

Vertex-IV XC4VLX160-12FF1148	Total	12-bit Golay Decoder	
		Used	%
Slices	67584	275	0
Flipflops	135168	291	0
LUTs 4-Inputs	135168	520	0
Bonded IOBs	768	43	5

Table-5.3 represents a comparative analysis of the delay based results of the proposed work with some existing works. Table 5.4 presents Latency (clock cycles) and LUT utilization based comparison of encoder design in this work and some previous works.

TABLE 5.3

COMPARISON OF OPERATIONAL FREQUENCY

Work	Operational Frequency (MHz)	
	Encoder	Decoder
Proposed	360.828	329.76
[1]	238.575	195.082
[5]	-	100

TABLE 5.4

COMPARISON OF THE PROPOSED ENCODER ARCHITECTURE CONSIDERING LATENCY AND LUT UTILIZATION

Reference	LTU Utilization(%)	Latency (clock cycle)
Proposed	0.048	12
[1]	0.14	12
[19]	1.33	12
[20]	1.72	12

6.1 CONCLUSION

In the proposed work optimization of hardware architecture of extended Golay encoder and decoder based designed and simulation. The consequences obtained from the design the combination for encoder and decoder in term of operational frequency. This design used high rated application based configurable circuits.

The future scope is to further optimizing the performance of proposed algorithm. The scholars may assume the challenges to reduce the ratio overhead bits versus data bits in encoded codeword and researchers may increase the length of data word that may be encoded the algorithm with error detection and correction capacity.

6.2 FUTURE SCOPE

The design can be used in integration with the existing communication systems for data communication. The golay decoder is among the most secured method to recover the data from errors in a noisy wireless communication channel. With the increasing applications of wireless devices this design can be utilized in future for communication among hand-held wireless devices.

There is always a scope of modification in all the designs with respect to the future emerging technologies. So the present work can also be improved in future for its parametric performances.

REFERENCE

[1] Satyabrata Sarangi and Swapna Banerjee, "Efficient Hardware Implementation of Encoder and Decoder for Golay Code", IEEE Transaction on very large scale Integration (VLSI) system, Vol.23 Issue No.9, pg.1965-1968, September 2015.

[2] Marcel J.E.GOLAY "Notes on Digital Coding", Reprinted from proc. IRE, Vol.37, pg-657 June 1949.

[3] Mario de Boer and Ruud Pellikaan, "The Golay codes" Springer, pg.338-347, September 1995.

[4] Dr. Ravi Shankar Mishra, Prof Puran Gour and Mohd Abdullah, "Design and Implementation of 4 bits galois Encoder and Decoder in FPGA", International Journal of Engineering Science and Technology (IJEST), Vol.3 No.7, pg.5724-5732, July 2011.

[5] Dongfu Xie, "Simplified algorithm and hardware implementation for the (24, 12, 8) Extended golay soft Decoder up to 4 Errors", The International Arab Journal of Information Technology, Vol.11 No.2, pg.111-115, March 2014.

[6] Xiao-Hong Peng and Paddy G. Farrell" On Construction of the (24, 12, 8) Golay Codes", December 2005.

[7] Matthew G. Parker, Kenneth G. Paterson and Chintha Tellambura, "Golay Complementary Sequences", January 2004.

[8] Yan-Haw Chen, Chih-Hua Chine, Chine-Hsiang Huang, Trieu- Kien Truong And Ming-Haw Jing, "Efficient Decoding of schematic (24,12,7) and (41,21,9) Quadric Residue codes",Journal of Information science And Engineering Vol.26, pg.1831-1843, December 2010.

[9] Li Ping and Kwan L. Yeung, "Symbol-by-Symbol APP Decoding of the Golay Code and Iterative Decoding of Concatenated Golay Codes", IEEE Transaction on Informationtheory, Vol.45, No.7, pg.2558-2562, November 1999.

[10] Yihua Chen, Juehsuan Hsiao, PangFu Liu and Kunfeng Lin, "Simulation and Implementation of BPSK BPTC of MSK golay code in DSP chip", Communications in Information Science and Management Engineering, Vol.1 No.4, pp.46-54, Nov 2011.

[11] Eyas El-Qawasmeh, Maytham Safar and Talal Kanan, "Investigation of golay code (24, 12, 8) Structure in improving search techniques", The

International Arab Journal of Information Technology, Vol.8, No.3, pg.265-271, July 2011.

[12] Ali Pezeshki, A. Robert Calderbank, William Moran andStephen D. Howard, "Doppler Resilient Golay Complementary Waveforms", IEEE Transaction on Information Theory Vol.54, No.9 , pg.4254-4266, September 2008.

[13] Faisal Alsaby, Kholood Alnoowaiser and simon Berkovich, "Golay code Transformation for ensemble clustering in application of medical Diagnostics", International Journal of Advanced Computer Science and Applications (IJACSA), Vol.6 No.1, pg.49-53, 2015.

[14] B. Honary, G. Markarian, and M. Darnel, "Low-complexity trellis decoding of linear block codes", IEE Proc.-Commun., Vol. 142, No. 4, pg. 201-209, August 1995.

[15] Arun kumar Balasundaram, Angelo Pereira, Jun Cheol Park and Vincent Mooney, Wavelet Error Control Codes in VLSI" IEEE Press Piscataway, pg. 563-564, January 2004.

[16] Wen-Ku Su, Pei-Yu Shih, Tsung-Ching Lin, and Trieu-Kien Truong, "Soft-decoding of the (23, 12, 7) Binary Golay Code", International MultiConference of Engineers and Computer Scientists IMECS, Vol II, pg.1191, March2008.

[17] Jonathan Jedwab and Matthew G. Parker, "construction of binary Golay sequence pairs from odd-length Barker sequences", Journal of Combinatorial Designs, Vol. 17, pg. 478- 491, September 2008.

[18] Satish kumar Buddha, "Hamming and Golay Codes", December 2011.