DOI 10.29042/2019-4823-4831

Helix Vol. 9 (1): 4823- 4831

Computation of Parity Check Matrices for Binary EG-LDPC Codes used in Communication Systems

[1]J. Chinna Babu, [2]C.Chinnapu Reddy, [3]M.N.Giri Prasad

[1]Research Scholar, JNTUA Ananthapuramu & Assistant Professor, AITS, Rajampet.

[2]Head, Dept. of ECE, Govt. Polytechnic Adoni, India.

[3]Director of Admissions, JNTUA, Ananthapuramu, India.

Email: jchinnababu@gmail.com, ccreddyece@gmail.com, mahendragiri1960@gmail.com

Received: 05th January 2019, Accepted: 29th January 2019, Published: 28th February 2019

Abstract

The Low Density Parity Check (LDPC) are direct codes, which are true block and Shannon Limit codes. These codes are attained least error floors of data bits for data transfer applications used in communication systems. However, the proposed LDPC codes are more beneficial than Turbo codes because of reduction in the decoding complexity and detection of the errors in less cycle time. This results the reduction of decoding time, low decoding latency, complexity and as well as least error floorings in communication, when the transmitted data contains multiple error bits. This paper is proposed to represent the majority logic decoding/detecting of LDPC codes. This paper proposes the Generation of Originator and Parity estimated matrices for the Binary LDPC Codes. Here, the proposed techniques are hard decision decrypting and soft decision decrypting schemes. These schemes uses majority logic decoding based on the data transmission and reception in communication channel. This paper also elaborates the effective calculation of Euclidean distance and algorithm for constructing the LDPC codes.

Keywords

LDPC Codes; BF Algorithms; EG-LDPC; Turbo Codes; Coding Theory.

Introduction

Correcting Error Codes

The terminated data or parity bits is auxiliary to the novel coded data, therefore that the exact coded data can be recovered at the destination end without the requirement for the data re-transmission and similarly the faults can be discovered and rectified, if any faults are existing. For this, we are using sophisticated codes, which are called error-correcting odes.

Shannon's Theorem:

For the consistent broadcasting of data over a given communication medium, the data transmission rate should not exceed the channel capacity, which is proved by Shannon theorem.

LDPC Codes:

LDPC codes are established by R.G. Gallagher, hence these are also identified as Gallagher codes. Due to unbearable practical comprehension, these codes are abandoned, although these codes were designed in the early 1960's. These LDPC codes are direct error improving codes and methodologies near Shannon capability. These LDPC codes are the better blunder improving codes, used for encoding and decoding at present scenario [1]-[3]. Here are binary categories of LDPC codes are of two types.

LDPC Regular Codes: Regular type of codes take equivalent row masses (Wr) and equivalent column masses (Wc) (Equivalent number of ones in rows and columns).

Ex: $\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$

Here $W_r = 3$ for all of the rows and $W_c = 2$ for all of the columns.

LDPC Irregular Codes: In these, all of the rows might have dissimilar masses ($Wr \neq$ constant) and all of the columns should have dissimilar masses ($Wc \neq$ constant). For example let us assume the following equivalence matrix which is having four number of rows and four number of columns. Generally, LDPC codes are having higher number of zeros than the ones. It can be considered in equivalence check matrix is as shown below.

Ex: $\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$

If whole number of rows and columns in the above equivalence matrix comprise uniform amount of ones. Therefore, it can be named as Binary LDPC codes.

Various decoding algorithms have been established for decrypting of LDPC algorithms based on the following two schemes and they are

1) Hard Decision Decoding (HDD)

2) Soft Decision decoding schemes (SDD).

 Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

Decision Decoding:

Input Bit 1	Input Bit 2	Added Parity from Transmitter	Created Code Word
0	0	0	000
0	1	1	011
1	0	1	101
1	1	1	111

Table I: Code Word Generation

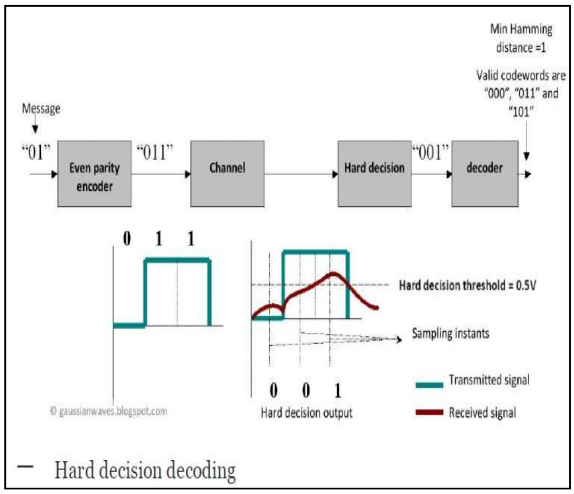

Fig.1: Hard Decision Decoding

All Possible Code Words	Decision Output	Hamming Distance
000	001	1
011	001	1
101	001	1
110	001	3

Table II: Hamming Distance Estimation

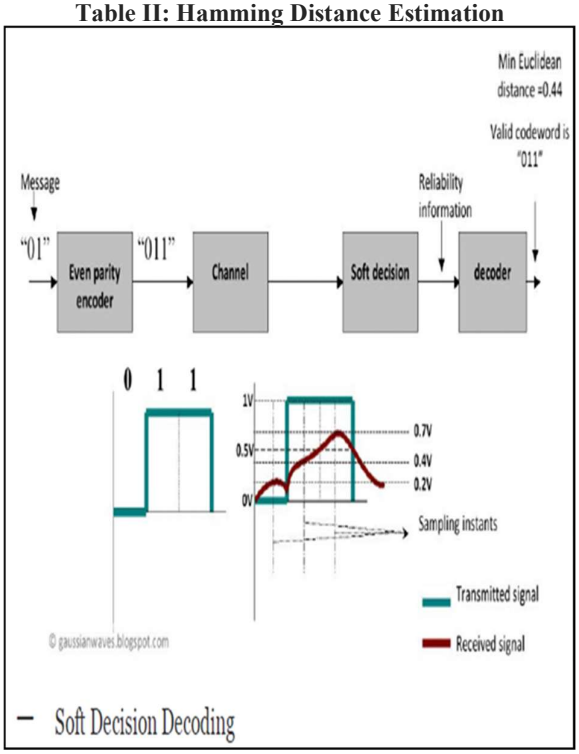

Fig.2. Soft Decision Decoding

Depends on the threshold value we may estimate the Euclidean Distance for all the code words, then the code word which may have least hamming distance that is the exact code word which was transmitted earlier [4]-[7].

Valid Code Words	Voltage Level at Each Sampling Instance of Traditional Waveform	Calculation of Euclidean Distance	Euclidean Distance
000 (0v, 0v, 0v)	0.2v, 0.4v, 0.7v	(0.2-0)^2+(0.4-0)^2+(0.7-0)^2	0.69
011 (0v,1v, 1v)	0.2v, 0.4v, 0.7v	(0.2-0)^2+(0.4- 1)^2+(0.7-1)^2	**0.49**
101 (1v, 0v, 1v)	0.2v, , 0.4v, 0.7v	(0.2-1)^2+(0.4-0)^2+(0.7-1)^2	0.89
110 (1v, 1v, 0v)	0.2v, 0.4v, 0.7v	(0.2-1)^2+(0.4-1)^2+(0.7-0)^2	1.49

Table III: Euclidean Distance Calculation

LDPC Codes - Basics

Undertake that the coded data to be encrypted is a k-bit data establishing a standard data, $m = (m_1, m_2 \dots m_k)$, and it is one of the vector of 2^k number of possible code words. The transmitter receipts this code pattern and creates a code word $c = (c_1, c_2 \dots c_n)$, where n is greater than k, that is there is an addition of redundancy. Moreover coding of blocks and coding of convolution types are also the tools for the redundancy addition in error improving coding methods.

Linear Block Codes Definition

A slab of data code word c is a direct code word, if the code words form a trajectory subset of the trajectory set Vn; it has k exactly self-governing vectors that are represented as code words, such that each probable code word is an exact grouping of subsets. This clarification represents that the set of 2^k possible code patterns establishes a trajectory subset of the set of words of n number of bits. An exact code is categorised by the datum that the summation of any two code patterns is also a code word or code pattern or code vector.

Originator (Generator) Matrix

Assume $c\ (n, k)$ can be an exact direct code word and the vectors $(g_1, g_2 \dots g_k)$ are k exactly self-governing trajectories. Each code word is an exact grouping of them:

$$C = m_1 g_1 + m_2 g_2 \dots m_k g_k$$

Where, all of the trajectory and matrix procedures are using XOR operation. These originally self-governing trajectories can be organised by using a matrix form said to be originator matrix G:

$$G = \begin{pmatrix} g_1 \\ g_2 \\ \vdots \\ \vdots \\ g_k \end{pmatrix} \begin{pmatrix} g_{1,1} & g_{1,2} & g_{1,3} & \cdots & \cdots & g_{1.n} \\ g_{2,1} & g_{2,2} & g_{2.,3} & \cdots & \cdots & g_{2,n} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ g_{k,1} & g_{k,2} & g_{k,3} & \cdots & \cdots & g_{k,n} \end{pmatrix}$$

Where, the given code vector is $m = (m_1, m_2 \dots m_k)$, the equivalent code vector is attained by using multiplication of the matrices is as shown below.

$$c = m.G = (m_1, m_2, \dots . m_k).\begin{pmatrix} g_1 \\ g_2 \\ g_3 \\ . \\ . \\ g_k \end{pmatrix} = m_1 g_1 + m_1 g_2 + \cdots \cdots + m_k g_k$$

Equivalence Check Matrix (H)

Here, the matrix having equivalence check equations can be of size $(n - k)\ x\ n$ matrix with $(n - k)$ self-governing rows and it is the code, which is having dual space c, i.e. $GH^T = 0$.

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

$$H = \begin{pmatrix} h_1 \\ h_2 \\ \vdots \\ \vdots \\ h_{n-k} \end{pmatrix} = \begin{pmatrix} h_{1,1} & h_{1,2} & h_{1,3} & \dots & \dots & h_{1.n} \\ h_{2,1} & h_{2,2} & h_{2.,3} & \dots & \dots & h_{2,n} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ h_{n-k,1} & h_{n-k,2} & h_{n-k,3} & \dots & \dots & h_{n-k,n} \end{pmatrix}$$

It can also established that the equivalence check estimations can be attained from the equivalence check matrix H, i.e. $CH^T = 0$. Where, this context also assumes entirely a block cypher.

Systematic Form of Block Codes

The construction of a code word in methodical form is as shown in Fig. 2.1. In this method, a code word contains of k message bits monitored by $(n-k)$ parity check bits.

$K -$ Data bits	$n - K$ Equivalence bits

Fig. 2.1: Code Vector Systematic Form of a Block Code

Thus, a systematic direct block code $c\ (n,k)$ might be stated by the succeeding creator matrix:

$$G = \left(\begin{array}{cccccc|ccccc} 1 & 0 & 0 & \dots & \dots & 0 & p_{1,k+1} & p_{1,k+2} & \dots & \dots & p_{1,n} \\ 0 & 1 & 0 & \dots & \dots & 0 & p_{2,k+1} & p_{2,k+2} & \dots & \dots & p_{2,n} \\ 0 & 0 & 1 & \dots & \dots & 0 & p_{3,k+1} & p_{3,k+2} & \dots & \dots & p_{3,n} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & \dots & 1 & p_{k,k+1} & p_{k,k+2} & \dots & \dots & p_{k,n} \end{array}\right)$$

$$Identity\ Matrix\ (k*k) \quad Parity\ Matrix\ (k*n-k)$$

Which, in a compact notation, is $G = [I_{k*k} \quad P_{k*(n-k)}]$. The comparable equivalence check matrix is given by

$$H = [P^T{}_{(n-k)*k} \quad I_{(n-k)*(n-k)}].$$

Interpreting the Rectilinear Block Codes

It can perceive from the communication scheme, that as a significance of its broadcast through a noisy medium, a code word might be acknowledged comprising some faults. The acknowledged vector can consequently be dissimilar from the conforming transferred code word, and it will be represented as $r = (r_1, r_2 \dots r_n)$. An error occasion can be demonstrated as a fault pattern or fault code word $e = (e_1, e_2 \dots e_n)$, where $e = r + c$.

To distinguish the faults, it can be used by the fact that any legal code word must satisfy the state $c.H^T = 0$. A fault finding tool is created from the above notation, which accepts the succeeding expression $s = r * H^T$, where $s = (s_1, s_2 \dots s_n)$ is named as the code word pattern. The perceiving process is accomplished over the traditional code pattern. If s is the entirely nil trajectory, the traditional trajectory is a legal code word or else, there are faults in the traditional code pattern. The pattern collection is tested to catch the conforming error pattern Ej for $j = 1, 2 \dots N$ and the decrypted code word is attained by using $m' = r + Ej$.

Definition of LDPC Codes

The LDPC codes are direct block codes, which are represented as (n,k) or (n, Wc, wr), here n can be span of the code word, k can be distance of the data bits, Wc can be mass of the column (i.e. the amount of non-zero elements in a column of the parity check matrix), and Wr become mass of the row (i.e. the amount of non-zero elements in a row of the equivalence check matrix). Here are two noticeable features of LDPC codes. They are is as follows.

Parity Check Notation

These codes are characterized by an equivalence check matrix H, here H is a dual matrix, contains ones and zeros. It must satisfy $c.H^T = 0$, here c is a code pattern.

Low-Density

Where H is a matrix, and it is a sparse matrix. It indicates that amount of ones are lower compared to zeros. It is the characteristic of H that assurances the lower computational complexity.

LDPC Construction Algorithm Construction of LDPC Codes: Algorithm

$$PROCEDURE : LDPC\ CONSTRUCTION\ (n, r, V, h)$$
$$n: required\ length;\ \ r: rate; V: Column; h: degrees$$

$$H \leftarrow all\ zeros\ n\ (1-r)\ X\ n\ matrix$$

▷ Matrix initialization

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

$\propto \leftarrow [\emptyset]$

$for\ i \leftarrow 1{:}\ max\ (v)\ do$

$for\ j \leftarrow 1{:}\ V_i x\ n\ \ do$

$\alpha\ \leftarrow \alpha(i,j)$

$end\ for$

$end\ for$

$\beta \leftarrow [\emptyset]$

$for\ i\ \leftarrow 1 : \max(h)\ do$

$for\ j\ \leftarrow 1{:}\ h_i xm\ do$

$\beta \leftarrow \beta(i,j)$

$end\ for$

$end\ for$

$for\ i \leftarrow 1{:}\ n\ do$ ▷ CONSTRUCTION

$C \leftarrow random\ variante\ (\beta, \alpha_i)$

$for\ j \leftarrow 1{:}\ \alpha_i\ do$

$H(C_j, i) \leftarrow 1$

$end\ for$

$\alpha \leftarrow (\alpha - C)$

$end\ for$

$end\ PROCEDURE$

The line 1 to 2 initializes the matrix H and α vector with $n(1-r)X\ n$ and null respectively. The lines 3 through 7 are initialization of α vector with size of H matrix. The line 8 is initialized the β vector with Null value. The lines 9 through 13 are computed β vector with size H matrix. The line 14 is construct code word with size n. In line 15, generate a random variant of C using vectors of β and α. The lines 16 through 19 to compute all 1's in a given matrix. This algorithm will execute till end of required length. From the above algorithm we can develop or construct the LDPC Codes of the required size [8]-[9].

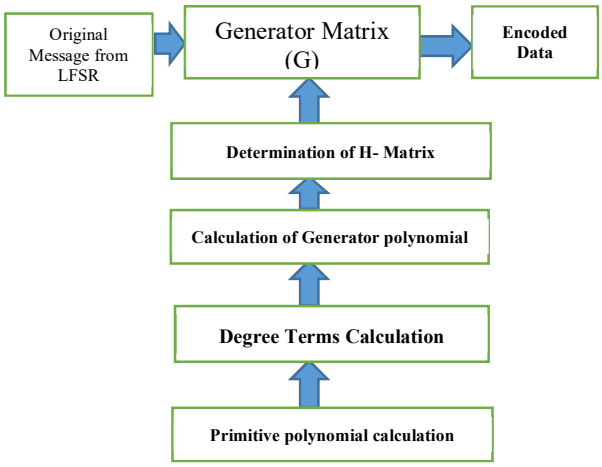

Fig. 3: Encoder Block Diagram

+	α^0 **1**	α^1 **2**	α^2 **4**	α^3 **3**	α^4 **6**	α^5 **7**	α^6 **5**
α^0 1	0	α^3	α^6	α^1	α^3	α^4	α^2
α^1 2	α^3	0	α^4	α^0	α^2	α^6	α^5
α^2 4	α^6	α^4	0	α^5	α^1	α^3	α^0
α^3 3	α^1	α^0	α^5	0	α^6	α^2	α^4
α^4 6	α^5	α^2	α^1	α^6	0	α^0	α^3
α^5 7	α^4	α^6	α^3	α^2	α^0	0	α^1
α^6 5	α^2	α^5	α^0	α^4	α^3	α^1	0

TABLE IV: Galois Addition

×	α^0**1**	α^1 **2**	α^2 **4**	α^3 **3**	α^4 **6**	α^5 **7**	α^6 **5**
α^0 1	α^0	α^1	α^2	α^3	α^4	α^5	α^6
α^1 2	α^1	α^2	α^3	α^4	α^5	α^6	α^0
α^2 4	α^2	α^3	α^4	α^5	α^6	α^0	α^1
α^3 3	α^3	α^4	α^5	α^6	α^0	α^1	α^2
α^4 6	α^4	α^5	α^6	α^0	α^1	α^2	α^3
α^5 7	α^5	α^6	α^0	α^1	α^2	α^3	α^4
α^6 5	α^6	α^0	α^1	α^2	α^3	α^4	α^5

TABLE V: Galois Multiplication

LDPC Check Matrix Construction

1. Calculation of Primitive polynomial
2. Calculation of Degree terms
3. Galois addition and Galois multiplication
4. Calculation of $G(x)$
5. Calculation of H-matrix from $G(x)$.

For the better understanding of the procedure, here let us consider a binary code of length $(16, 8)$.
To assume $(n, k) = (16, 8)$

Step 1: The Calculation of Primitive Polynomial

*Assume the odd numbers from 1to 16

$$1, 3,5,7,9,11,13,15$$

* Write the odd numbers in binary form

$$1 - 0001, 3 - 0011, 5 - 0101, 7 - 0111, 9 - 1001,$$
$$11 - 1011, 13 - 1101, 15 - 1111$$

* Concatenate 1 as MSB bit

$$1 - 10001;\ 3 - 11011;\ 5 - 11101;\ 7 - 11111,$$
$$9 - 11001, 11 - 11011, 13 - 11101, 15 - 11111$$

*Now reverse the bits which are obtained after concatenation

$$1 - 10001;\ 3 - 11011;\ 5 - 10111;\ 7 - 11111,$$
$$9 - 10011, 11 - 11011, 13 - 10111, 15 - 11111$$

* Now relate the appended bits with the reversed bits

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

$$1 - 10001 - 10001$$
$$3 - 11011 - 11011$$
$$5 - 11101 - 10111$$
$$7 - 11111 - 11111$$
$$9 - 11001 - 10011$$
$$11 - 11011 - 11011$$
$$13 - 11101 - 10111$$
$$15 - 11111 - 11111$$

*If both bit streams are equal ignore them otherwise note the numbers.

$$3 - 10011 - 11001$$
$$7 - 10111 - 11101$$
$$9 - 11001 - 10011$$
$$13 - 11101 - 10111$$

* Select the minimum value of remaining expression

For the primitive polynomial 11001➔ $x^4 + x^3 + x + 1$

*Primitive polynomial $P(x) = x^4 + x^3 + x + 1$

Step 2: Calculation Degree Terms

While calculating the degree term value, if the value exceeds the 16, then ex-or it with the primitive polynomial $x^4 + x^3 + x + 1$, $P(x) = 11001$.

$$\alpha^0 = 0001 = 1$$
$$\alpha^1 = 0010 = 2$$
$$\alpha^2 = 0100 = 4$$
$$\alpha^3 = 1000 = 8$$
$$\alpha^4 = 10000\text{^}11001 = 01001$$
$$\alpha^5 = 10010\text{^}11001 = 01011$$
$$\alpha^6 = 10110\text{^}11001 = 01111$$
$$\alpha^7 = 11110\text{^}11001 = 00111$$
$$\alpha^8 = 01110$$
$$\alpha^9 = 11100\text{^}11001 = 00101$$
$$\alpha^{10} = 01010$$
$$\alpha^{11} = 10100\text{^}11001 = 01101$$
$$\alpha^{12} = 11010\text{^}11001 = 00011$$
$$\alpha^{13} = 00110$$
$$\alpha^{14} = 01100$$
$$\alpha^{15} = 11000\text{^}11001 = 00001$$

Step 3: Calculation of Generator Polynomial

*The normalized form of originator polynomial G(x) is given by

$$G(x) = (x - \alpha^0)(x - \alpha^1) \ldots\ldots (x - \alpha^{n-k-1})$$

Where $(n, k) = (16, 8)$, therefore

$$G(x) = (x - \alpha^0)(x - \alpha^1)(x - \alpha^2)(x - \alpha^3)x - \alpha^4)$$
$$(x - \alpha^5)(x - \alpha^6)(x - \alpha^7)$$

*By solving above equation the result is

G(x) = $x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$

➔ (1 1 1 1 1 1 1 1)

Step 4: determination of parity check matrix

$$H = \begin{pmatrix} 1\,0\,0\,0\,0\,0\,0\,0\,1\,0\,1\,0\,0\,0\,0\,0 \\ 0\,1\,0\,0\,0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0\,0\,0\,0\,1\,0\,1\,0\,1\,1\,0\,0 \\ 0\,0\,0\,1\,0\,0\,0\,0\,1\,1\,1\,1\,0\,1\,0\,1 \\ 0\,0\,0\,0\,1\,0\,0\,0\,1\,1\,1\,1\,1\,1\,0\,1 \\ 0\,0\,0\,0\,0\,1\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0\,1\,0\,1\,1\,1\,1\,0\,1\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,1\,1\,0\,0\,0\,0 \end{pmatrix}$$

*The first column is obtained by writing the bits from LSB to MSB of $\alpha^0 = 1 = 0\,001$ and in the next step downshift the data in column 1 to obtain column 2 and so on.

*If the '1' appears as last bit in the column then for the next shift we should Ex-or the shifted data with G(x) to obtain that

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

column, otherwise data is shifted left/down.

*Here the last bit in column 3 is '1' hence it is shifted down and is then ex-or with $G(x) = [1\ 1\ 1\ 1\ 1\ 1\ 1\ 1]$ to obtain data in column 4. This is the procedure to obtain H-matrix for binary LDPC codes as shown in Fig.3.2.

The code word can be generated by making the product of message 'm' with originator matrix 'G', $\boldsymbol{c = m.G}$

1. Arranging the equivalence check matrix in systematic coded form

$$H_{sys} = \left[I_M \middle| P_{M\times K} \right]$$

2. Reorganising the regular equivalence check matrix

$$G = \left[P^T_{K\times M} \middle| I_K \right]$$

$$G = \begin{pmatrix} 1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0 \\ 0&1&1&1&1&1&0&0&1&0&0&0&0&0&0&0 \\ 1&0&1&1&1&0&1&1&0&0&1&0&0&0&0&0 \\ 0&1&0&1&1&1&1&1&0&0&0&1&0&0&0&0 \\ 0&0&1&0&1&0&0&0&0&0&0&0&1&0&0&0 \\ 0&0&1&1&1&0&1&0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0 \\ 0&0&0&1&1&1&0&0&0&0&0&0&0&0&0&1 \end{pmatrix}$$

Let us assume an 8-bit data message

$m = [1\ 0\ 0\ 1\ 0\ 1\ 0\ 0]$, This data vector can be encrypted by reproducing it with originator matrix i.e.Then the resultant code word for Encrypting is

$$C = [1\ 0\ 0\ 1\ 0\ 1\ 0\ 0]$$

$$\begin{pmatrix} 1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0 \\ 0&1&1&1&1&1&0&0&1&0&0&0&0&0&0&0 \\ 1&0&1&1&1&0&1&1&0&0&1&0&0&0&0&0 \\ 0&1&0&1&1&1&1&1&0&0&0&1&0&0&0&0 \\ 0&0&1&0&1&0&0&0&0&0&0&0&1&0&0&0 \\ 0&0&1&1&1&0&1&0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0 \\ 0&0&0&1&1&1&0&0&0&0&0&0&0&0&0&1 \end{pmatrix}$$

$$C = [1\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0]$$

The code pattern is supposed to be legal, if the code vector fulfils the conditional design $Z = C * H^T = 0$

Bit Flipping algorithms are of the most efficient LDPC decrypting algorithms which have decent error concert and have minimal hardware decrypting complication [10].

Steps for Decrypting of LDPC Codes Using Bit Flipping Algorithms:

Bit Flipping algorithms are of the greatest efficient LDPC decrypting algorithms which has decent error concert, and has minimal hardware computational complexity. The following steps reveals the process for decrypting of LDPC codes using BF algorithms. Brief explanation/ steps are offered below**.**

Step 1: Initially the received 32-bit code pattern and assigned to 32 variable nodes.

Step 2: Verify the Check node connections with all variable nodes and compute check node values.

Step 3: In step 3, majority check operations are performed for all variable nodes.

In this paper, Fig.4 shows the sequential hybrid LDPC decipherer which is used for Decrypting of LDPC codes. The decipherer consists of Input buffer (which grasps the code data), a buffer which is rollback (It grasps the outcome of complemented bits), an output buffer (It grasps the fault free data bits), check and variable nodes, a flipped unit (Which flips the bits of data, when limits are not met), a variable hub unit (VPU), and a check unit hub (CUH), and shift registers (these are used to alter the data bits as the reiteration starts) are used [11]-[16].

Conclusion

The LDPC codes have the near Shannon limit presentation and the creation of the equivalence check matrices is clarified in this paper for the binary LDPC codes. Where the methodology of creating the LDPC Codes and its corresponding general decoding block diagram of LDPC codes is also explained. General Decoding scheme for bit flipping algorithms is also elaborated. Hence, these LDPC codes have higher coding rates compared to other conventional codes available. From these construction, we conclude that these codes are used in Security and Signal processing applications which provides authentication between source and destination

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

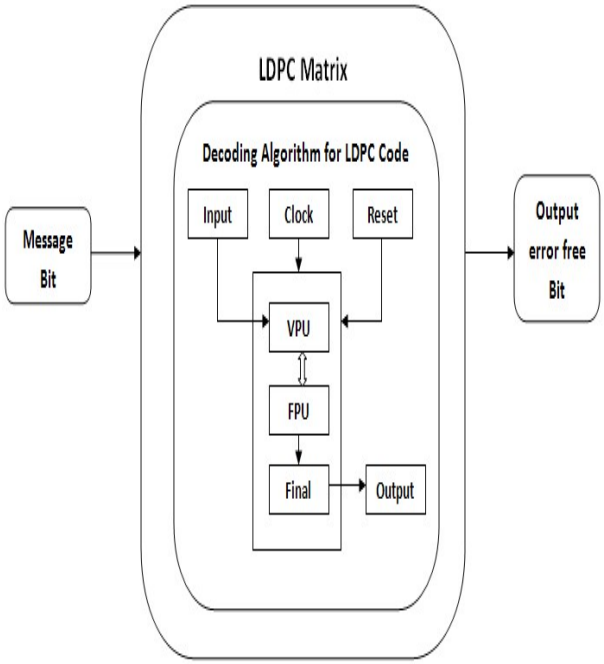

Fig.4. Block Diagram of BF Decoder for EG-LDPC Codes

References

[1] R. G. Gallagher, "Low density parity check codes", *IRE Trans. Info. Theory*, vol. IT-8, pp. 21-28, 1962.

[2] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes", *Electronics Letters*, vol. 32, no. 18, pp. 1645-1646, 1996.

[3] D. J. C. MacKay, "Good error correcting codes based on very sparse matrices", *IEEE transactions on Information Theory*, 1997.

[4] D. J. C. MacKay, "Gallagher codes that are better than turbo codes", *Proc. 36th Allerton Conf. Communication, Control, and Computing*, 1998.

[5] M. P. C. Fossorier, M. Mihaljevic, and H. Imai, "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," *IEEE Trans. Commun.*, vol. 47, no. 5, pp. 673–680, May 1999.

[6] J. Zhang and M. P. C. Fossorier, "A modified weighted bit-flipping decoding of low-density parity-check codes," *IEEE Commun. Lett*, vol. 8, no. 3, pp. 165–167, Mar. 2004.

[7] J. Cho and W. Sung, "High-performance and low-complexity decoding of high-weight LDPC codes," (in Korean) *J. Korea Inf. Commun. Soc.*, vol. 34, no. 5, pp. 498–504, May 2009.

[8] J. Chinna Babu, C. Chinnapu Reddy, M.N.Giri Prasad, "Comparison of technologies for the implementation of SBF decoder for geometric LDPC codes," *INDJST Journal.,* March 2016.

[9]. J. Chinna Babu, "VLSI Implementation of Decoding algorithms for EG-LDPC Codes," *Elsevier Procedia Computer Science.,* vol. 115, no. 3, pp. 143-150, 2017.

[10] M.N.Giri Prasad, C. Chinnapu Reddy, J. Chinna Babu "Generation and Decoding of Non-Binary LDPC Codes using MSA Decoding algorithm" Lecture Notes in Electrical Engineering., Vol: 434, PP. 583-591. Sept. 2017.

[11] JIN SHA, MINGLUN GAO, ZHONGJIN ZHANG,LI LI "Self-Reliability based Weighted Bit-Flipping Decoding for Low-density Parity-check Codes"5th WSEAS Int. Conf. on Instrumentation, April 2006.

[12]. J. Chen, A. Dholakia, E. Eleftheriou, M. P. C. Fossorier, and X.-Y. Hu, "Reduced- complexity decoding of LDPC codes," *IEEE Trans. Commun.*, vol. 53, no. 8, pp. 1288–1299, Aug. 2005.

[13] R. Palanki, M. P. C. Fossorier, and J. S. Yedidia, "Iterative decoding of multiple-step majority logic decodable codes," *IEEE Trans. Commun.*, vol. 55, no. 6, pp. 1099–1102, Jun. 2007.

[14] S. Kudekar, T. Richardson, and R. Urbanke, "Spatially coupled ensembles universally achieve capacity under belief propagation," *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 7761–7813, 2013.

[15] S. Kumar, A. J. Young, N. Macris, and H. D. Pfister, "Threshold saturation for spatially coupled LDPC and LDGM codes on BMS channels," *IEEE Trans. Inf. Theory*, vol. 60, no. 12, pp. 7389–7415, Dec. 2014.

[16] J. Chinna Babu, C.Chinnapu Reddy, M.N.Giri Prasad "Comparison of Various Decoding Algorithms for EG-Low Density Parity Check Codes" Lecture Notes in Electrical Engineering., Vol:442, PP. 605-613. Oct. 2017.

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495