UNIT - 5

LINEAR BLOCK CODES

Introduction

When a digital signal is transmitted between two systems such as computers the signal gets distorted because of the addition of noise to the transmitted signal. The noise can introduce an error in the binary train of bit travelling from one system to the other. This means that a '0' may change to '1' or a '1' may change to 0. These errors can become a serious problem to the accuracy and performance of the digital system. Therefore, it is essential to detect and correct the errors.

The rediability of data transmission gets severely affected because of these errors. To improve the reliability of data transmission, the designer has to increase the signal power or reduce the noise spectral density N_0 so as to maximize the ratio (S_0/N_0).

But, in practice, there is a limitation on the maximum value of the ratio (S_0/N_0). We cannot increase the ratio beyond this limit. Hence for a fixed value of (S_0/N_0), we have to use some kind of coding in order to improve the quality of the transmitted signal.

Another advantage of using coding is that we can reduce the required value of (S_0/N_0) for a fixed Bit Error Rate (BER). Thus in turn reduces the transmitted power and size of the antenna.

Detection and correction of Errors

For the detection and correction of transmission errors, one or more than one extra bits are added to the data bits at the time of the transmitting the signal. These extra bits are known as "Parity Bits" The data bits along with the parity bits form a "code word"

Classification of codes

The codes are basically classified as under:

1. Error detecting codes: They are capable of only detecting the errors. They cannot correct the errors.
2. Error correcting codes: They are capable of detecting as well as correcting the errors.

Classification of Errors

The errors introduced in the transmitted data during their transmission may be categorized as under:

(i) Content Errors

(ii) Flow integrity Errors.

Content Errors: Errors present in the content of a message i.e a '0' may be received as 1 or vice versa. Such errors are introduced due to noise added into the data signal during its transmission.

Flow integrity Errors: Missing blocks of data. It is possible that a data block may be lost in the network as it has been delivered to a wrong destination.

Generation and Detection of coded signals.

Error control for data integrity can be achieved by using a technique called Forward Error Correction (FEC). Fig 1 shows the model of a digital communication system using FEC approach.

The discrete source generates information in the form of binary symbols. The channel encoder accepts these message bits and add redundancy according to a prescribed rule.

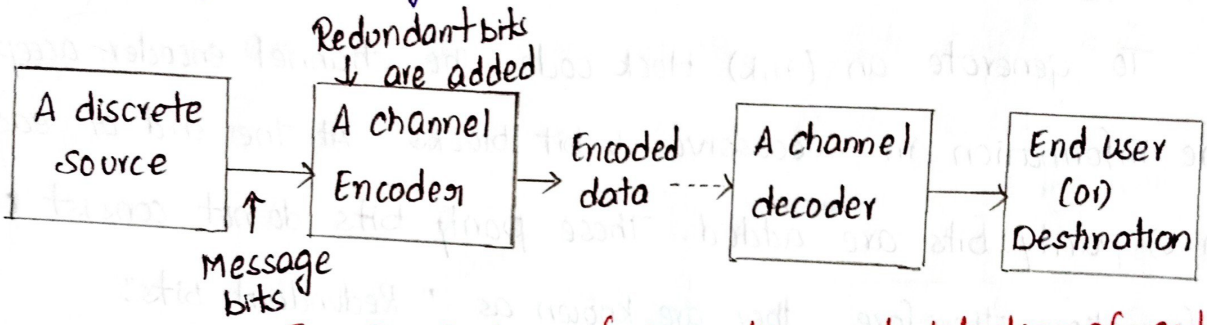


Fig: Illustration of generation and detection of coded signals

The channel decoder takes the help of the additional bits (i.e redundancy) to decide whether the transmitted signal is a 1 or a 0. Thus, the combined objective of channel encoder and decoder is to minimize the effect of channel noise.

→ **Classification of Error-Correcting Codes**

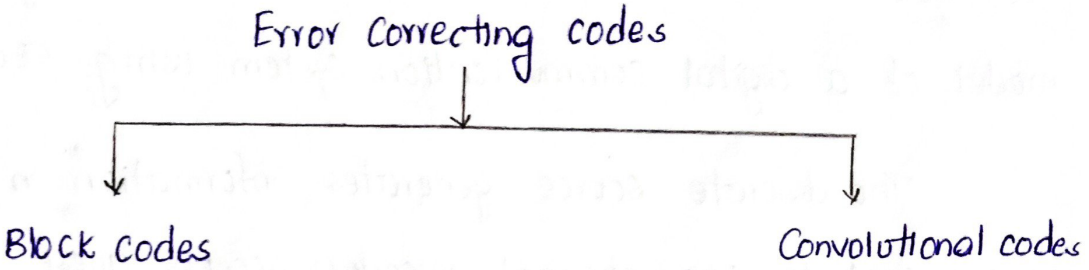
These codes can be classified into two categories as under:

1. Block codes
2. Convolutional codes.

The major difference between these codes is that the block codes do not need memory whereas the convolutional codes needs memory.

→ Classification of Codes

Error control code falls under one of the two basic types as under:



Block codes:

To generate an (n,k) block code, the channel encoder accepts the information in successive k-bit blocks. At the end of each block (n-k) parity bits are added. These parity bits do not consist of any information therefore, they are known as "Redundant bits".

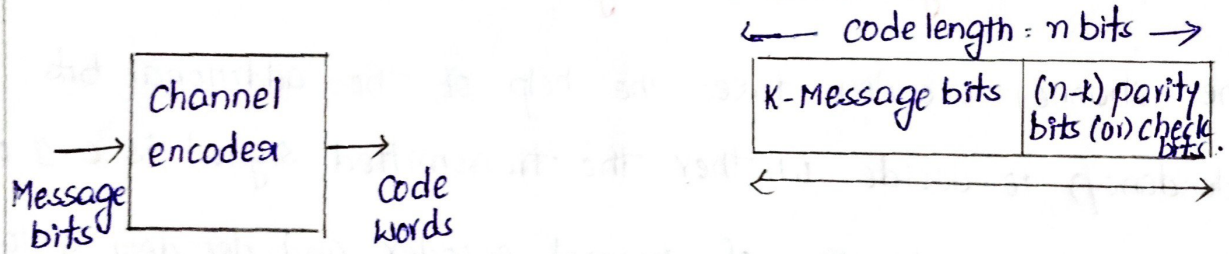


Fig2: Generation of n-bit block code.

Convolutional Code.

In Convolutional codes, the code words are generated by discrete-time convolution of the input sequence with the impulse response of the encoder. Convolutional codes needs memory for their generation.

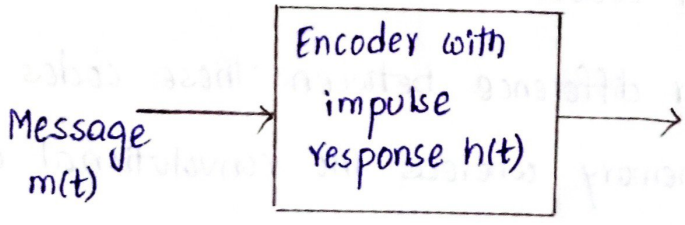


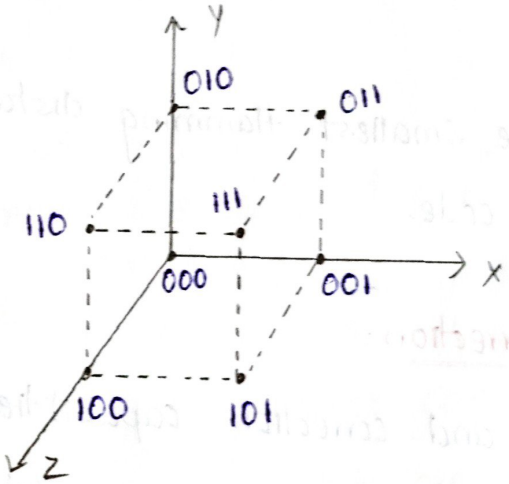
Fig: Convolutional codes.

Code Word: The code word is the n-bit encoded block of bits. It contains message bits and parity (or) redundant bits.

Code Rate: It is defined as the ratio of the no. of message bits (k) to the total no. of bits (n) in a code word.

$$r = \frac{k}{n}$$

Code vectors: We can visualize an n-bit code word to be present in an n-dimensional space. We can consider 3-bit code vectors. It has 8-possible combinations. We can assume bits x_0 to be on X-axis, bits x_1 on the Y-axis and bits x_2 on the Z-axis.



S.No	Codeword $x_2 : Z$	$x_1 : Y$	$x_0 : X$
0	0	0	0
1	0	0	1
2	0	1	0
⋮			
7	1	1	1

Fig: Code vectors for 3-bit code words.

Hamming Distance: Distance between the two code words is defined as the no. of locations in which their respective elements differ. For example, let us consider the two code words given below:

Code Word 1 — 1 1 0 1 0 1 0 0

Code Word 2 — 0 1 0 1 1 1 1 0

Hamming distance is 3

Hamming Weight of a codeWord [W(x)]

It is defined as the no. of non-zero elements in the codeword. Hamming weight of a code vector (code word) is the distance between that code word and an all-zero code vector. (A code having all elements equal to zero).

Code Efficiency:

It is defined as the ratio of message bits to the no. of transmitted bits per block.

$$\text{code rate} = \text{code efficiency} = \frac{k}{n}$$

Minimum distance dmin:

It is defined as the smallest Hamming distance between any pair of code vectors in the code.

Error detection and correction:

The error detection and correction capabilities of a coding technique depend on the minimum distance d_{min} as in table 1.

S.No	Description	Expression
1.	Detect upto 's' errors per word	$d_{min} \geq (s+1)$
2.	Correct upto 't' errors per word	$d_{min} \geq (2t+1)$
3.	Correct upto 't' errors and detect 's>t' errors per word	$d_{min} \geq (t+s+1)$

Table 1: Error detection and correction capabilities.

P
→ For a Hamming distance of 5, how many errors can be detected? How many errors can be corrected?

Sol. Given Minimum distance $d_{min} = 5$

(i) No. of errors that can be detected (s) can be obtained from

$$(s+1) \leq d_{min}$$

$$s+1 \leq 5$$

$$s \leq 4$$

Thus, at the most 4 errors can be detected.

(ii) No. of errors that can be corrected, (t) can be obtained from

$$(2t+1) \leq d_{min}$$

$$2t+1 \leq 5$$

$$t \leq 2$$

Thus, at the most 2 errors can be corrected.

→ Linear Block codes:

1. We can consider (n,k) linear block code in which 'k' no. of message bits and (n-k) parity bits are added to the message bits.
2. In codeword 'k' message bits are denoted by $m_0, m_1, \ldots m_{k-1}$ and (n-k) parity bits are denoted by $c_0, c_1, \ldots c_{n-k-1}$.
3. The sequence of message bits is applied to a linear block encoder to produce an n-bit code word. The elements of this code word are $x_0, x_1 \ldots x_{n-1}$

4. The first k-bits of the codeword are identical to message bits and (n-k) parity bits ($c_0, c_1 \ldots$) are linear sum of the k-message bits.

Mathematically it can be expressed as

$$x_i = \begin{cases} m_i & \text{for } i = 0, 1, \ldots k-1 \\ c_{i-k} & \text{for } i = k, k+1, \ldots n-1 \end{cases}$$

5. The code vector mathematically represented as

$$X = [M : C] \quad \text{where} \quad M = k\text{- message bits}$$

$C = (n-k)$ parity bits.

6. The code vector 'X' can also be represented as

$$X = MG$$

where X = codevector of $1 \times n$ size

M = Message vector of $1 \times k$ size

G = Generator Matrix of $k \times n$ size.

$$[X]_{1 \times n} = [M]_{1 \times k} \cdot [G]_{k \times n}$$

7. The generator matrix is dependent on the type of linear block code used.

The generator matrix is generally represented as under:

$$[G] = [I_k \mid P]_{k \times (n-k)}$$

where I_k = $k \times k$ Identity matrix

P = $k \times (n-k)$ co-efficient matrix (or) Parity Matrix.

$$I_k = \begin{bmatrix} 1 & 0 & \ldots & 0 \\ 0 & 1 & \ldots & 0 \\ \vdots & & & \\ 0 & 0 & \ldots & 1 \end{bmatrix}_{k \times k} \qquad P = \begin{bmatrix} P_{00} & P_{01} & \ldots & P_{n-k,0} \\ P_{01} & P_{11} & \ldots & P_{n-k,1} \\ \vdots & & & \\ P_{0,k-1} & P_{1,k-1} & \ldots & P_{n-k,k-1} \end{bmatrix}_{k \times (n-k)}$$

8. The parity vector can be obtained as under:

$$C = MP$$

Substituting the matrix form, we obtain

$$[C_0\ C_1 \ldots C_{n-k}]_{1\times(n-k)} = [m_0, m_1 \ldots m_k]_{1\times k} \begin{bmatrix} P_{00} & P_{10} & \ldots & P_{n-k,0} \\ P_{01} & P_{11} & \ldots & P_{n-k,1} \\ \vdots & & & \\ P_{0,k-1} & P_{1,k-1} & \ldots & P_{n-k,k-1} \end{bmatrix}_{k\times(n-k)}$$

By solving the above matrix equation, we get the parity vectors as under:

$$C_0 = m_0 P_{00} \oplus m_1 P_{01} \oplus m_2 P_{02} \oplus \ldots \oplus m_k P_{0,k-1}$$

$$C_1 = m_0 P_{10} \oplus m_1 P_{11} \oplus m_2 P_{12} \oplus \ldots \oplus m_k P_{1,k-1}$$

$$C_2 = m_0 P_{20} \oplus m_1 P_{21} \oplus m_2 P_{22} \oplus \ldots \oplus m_k P_{2,k-1}$$

→ **Hamming Codes:**

Hamming codes are linear block codes. The family of (n, k) Hamming codes for $q \geq 3$ is defined by the following expressions:

1. Block diagram: $n = 2^q - 1$
2. No. of message bits: $k = 2^q - q - 1$
3. No. of parity bits: $(n-k) = q$

where $q \geq 3$ i.e minimum no. of parity bits is 3.

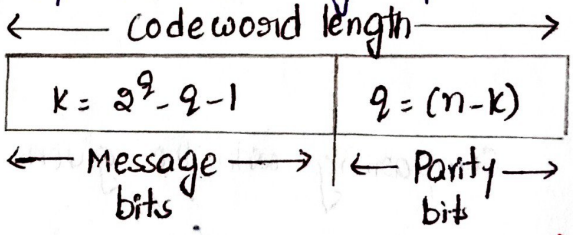


Fig: Codeword structure of Hamming code.

4. Minimum distance $d_{min} = 3$
5. Code rate or code efficiency $= \frac{k}{n}$

$$= \frac{2^q - q - 1}{2^q - 1} = 1 - \frac{q}{2^q - 1}$$

Another way of expressing the relationship between the message bits and the parity check bits, of a linear block code is "Parity check Matrix" It can be represented by

$$H = [P^T \mid I_{n-k}]$$

→ The generator matrix for a (6,3) block code is given below. Find all the code vectors of this code.

$$G = \left[\begin{array}{ccc:ccc} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{array}\right]$$

Sol:

Given Generator Matrix is $\left[\begin{array}{ccc:ccc} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{array}\right]$

$$G = [I_k : P].$$

Block code of size (6,3) i.e $n = 6$

$k = 3$

$q = n - k = 6 - 3 = 3$ bits

Comparing with the given Generator Matrix, we obtain

$$I_k = I_{3\times3} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad ; \quad P_{3\times3} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

The size of message block is $k = 3$, there are eight possible message block: (0,0,0), (0,0,1), (0 1 0), (0 1 1), (1 0 0), (1 0 1), (1 1 0), (1,1,1)

Now, let us obtain the relation between parity vectors [C], Message vectors M and the co-efficient (or) Parity Matrix P as under.

$$[C_0\ C_1\ C_2] = [m_0\ m_1\ m_2]\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

$C_0 = m_1 \oplus m_2$

$C_1 = m_0 \oplus m_2$

$C_2 = m_0 \oplus m_1$

1. For message word $(m_0\ m_1\ m_2) = (0\ 0\ 0)$, we have

$C_0 = 0 \oplus 0 = 0$; $C_1 = 0 \oplus 0 = 0$; $C_2 = 0 \oplus 0 = 0$.

2. For $(0\ 0\ 1) \Rightarrow C_0 = 1$; $C_1 = 1$, $C_2 = 0$

3. For $(0\ 1\ 0) \Rightarrow C_0 = 1$; $C_1 = 0$; $C_2 = 1$

4. For $(0\ 1\ 1) \Rightarrow C_0 = 0$; $C_1 = 1$; $C_2 = 1$

5. For $(1\ 0\ 0) \Rightarrow C_0 = 0$; $C_1 = 1$; $C_2 = 1$

S.No	Message vector			Parity Bits			Complete Code Vectors						W(x)
	m_0	m_1	m_2	C_0	C_1	C_2	m_0	m_1	m_2	C_0	C_1	C_2	
1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	1	1	1	0	0	0	1	1	1	0	3
3	0	1	0	1	0	1	0	1	0	1	0	1	3
4	0	1	1	0	1	1	0	1	1	0	1	1	4
5	1	0	0	0	1	1	1	0	0	0	1	1	3
6	1	0	1	1	0	1	1	0	1	1	0	1	4
7	1	1	0	1	1	0	1	1	0	1	1	0	4
8	1	1	1	0	0	0	1	1	1	0	0	0	3

→ The parity check matrix of a particular (7,4) linear block code is given by,

$$[H] = \left[\begin{array}{cccc:ccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

(i) Find the generator matrix (G)

(ii) List all the code vectors

(iii) What is the minimum distance between the code vectors?

(iv) How many errors can be detected? How many errors can be corrected?

Sol. First, let us obtain the P^T matrix.

P^T is the transpose of the co-efficient matrix P. The given parity check matrix is in the form of $(n-k) \times n$ (or) $2 \times n$ size.

It is given that the code is (7,4) Hamming code. Therefore we have

$$n = 7 \; ; \; k = 4$$

The parity check Matrix is in the form of $H = [P^T : I_{n-k}]_{2 \times n}$.

$$[H]_{3\times 7} = \left[\begin{array}{cccc:ccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

P^T (3×4) | I (3×3)

The transpose matrix P^T is given by

$$P^T = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}_{3\times 4}$$

The P matrix can be obtained by interchanging the rows and columns of the transpose matrix P^T.

$$P = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}_{4\times 3}$$

The Generator matrix G is in the form

$$G = [P : I_k]_{k\times n} \qquad G = [I_k : P]_{k\times n}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}_{4\times 7}$$

(ii) The parity bits can be obtained by using the following expression

$$C = MP$$

$$[C_0\ C_1\ C_2] = [m_0\ m_1\ m_2\ m_3] \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

$$C_0 = m_0 \oplus m_1 \oplus m_2$$

$$C_1 = m_0 \oplus m_1 \oplus m_3$$

$$C_2 = m_0 \oplus m_2 \oplus m_3$$

For message 0001 $\Rightarrow C_0 = 0,\ C_1 = 1,\ C_2 = 1$

The complete codeword for the give message is

0001011

Similarly the codevectors for the remaining messages as shown in the table.

S.NO	Message Vector M				Parity Bits C			code Words X							Weight of the code
	m_0	m_1	m_2	m_3	c_0	c_1	c_2	x_0	x_1	x_2	x_3	x_4	x_5	x_6	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	1	1	0	0	0	1	0	1	1	3
2	0	0	1	0	1	0	1	0	0	1	0	1	0	1	3
3	0	0	1	1	1	1	0	0	0	1	1	1	1	0	4
4	0	1	0	0	1	1	0	0	1	0	0	1	1	0	3
5	0	1	0	1	1	0	1	0	1	0	1	1	0	1	4
6	0	1	1	0	0	1	1	0	1	1	0	0	1	1	4
7	0	1	1	1	0	0	0	0	1	1	1	0	0	0	3
8	1	0	0	0	1	1	1	1	0	0	0	1	1	1	4
9	1	0	0	1	1	0	0	1	0	0	1	1	0	0	3
10	1	0	1	0	0	1	0	1	0	1	0	0	1	0	3
11	1	0	1	1	0	0	1	1	0	1	1	0	0	1	4
12	1	1	0	0	0	0	1	1	1	0	0	0	0	1	3
13	1	1	0	1	0	1	0	1	1	0	1	0	1	0	4
14	1	1	1	0	1	0	0	1	1	1	0	1	0	0	4
15	1	1	1	1	1	1	1	1	1	1	1	1	1	1	7

Table: Codevectors for the (7,4) Hamming code.

(iii) Minimum distance d_{min} is equal to the minimum weight of any non-zero code vector.

$$d_{min} = 3$$

(iv) No. of errors can be detected is given by $d_{min} \geq s+1$

$$3 \geq s+1 \Rightarrow s \leq 2$$

No. of errors can be corrected is given by $d_{min} \geq 2t+1$

$$3 \geq 2t+1 \Rightarrow t \leq 1$$

Thus, for the (7,4) Linear block code, at the most two errors can be detected and atmost only one error can be corrected.

→ For a systematic linear block code, the three parity check digits C_4, C_5 & C_6 are given by

$$C_4 = m_1 \oplus m_2 \oplus m_3$$

$$C_5 = m_1 \oplus m_2$$

$$C_6 = m_1 \oplus m_3$$

(i) Construct generator matrix

(ii) Construct code generated by this matrix

(iii) Determine error correcting capability.

Sol (i) First, let us obtain the parity matrix 'P' & Generator matrix 'G'.

We know the relation between the parity matrix 'P' and parity vector 'C'

$$C = MP \quad -(1)$$

$$[C_4\ C_5\ C_6]_{1\times3} = [m_1\ m_2\ m_3]_{1\times3} \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix}_{3\times3} \quad -(2)$$

Therefore,

$$C_4 = m_1P_{11} + m_2P_{21} + m_3P_{31} \quad -(3)$$

$$C_5 = m_1P_{12} + m_2P_{22} + m_3P_{32} \quad -(4)$$

$$C_6 = m_1P_{13} + m_2P_{23} + m_3P_{33} \quad -(5)$$

Comparing eq (3), (4), (5) with the given equations, then we get

$P_{11} = 1$ $\quad P_{12} = 1$ $\quad P_{13} = 1$

$P_{21} = 1$ $\quad P_{22} = 1$ $\quad P_{23} = 0$

$P_{31} = 1$ $\quad P_{32} = 0$ $\quad P_{33} = 1$

Hence, Parity check matrix $P = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}_{3\times3}$

The Generator matrix 'G' is in the form

$$G = [I_k : P_{k\times k}]_{k\times n}$$

$$= [I_{3\times 3} : P_{3\times 3}]_{3\times 6}$$

$$G = \left[\begin{array}{ccc:ccc} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right]$$

(ii) Given $C_4 = m_1 \oplus m_2 \oplus m_3$

$C_5 = m_1 \oplus m_2$

$C_6 = m_1 \oplus m_3$

for message $(m_1\ m_2\ m_3) = (0\ 0\ 1)$, we have

$C_4 = 0 \oplus 0 \oplus 1 = 1$

$C_5 = 0 \oplus 0 = 0$

$C_6 = 0 \oplus 1 = 1$

The codeword for the message 001 is 001101

S.No	Message vector			check Bits			Code vectors						code Weight W(x)
	m_1	m_2	m_3	C_4	C_5	C_6	m_1	m_2	m_3	C_4	C_5	C_6	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	1	1	0	1	0	0	1	1	0	1	3
2	0	1	0	1	1	0	0	1	0	1	1	0	3
3	0	1	1	0	1	1	0	1	1	0	1	1	4
4	1	0	0	1	1	1	1	0	0	1	1	1	4
5	1	0	1	0	1	0	1	0	1	0	1	0	3
6	1	1	0	0	0	1	1	1	0	0	0	1	3
7	1	1	1	1	0	0	1	1	1	1	0	0	4

Table : Codewords

iii) Error detecting and correcting capability

For detection $d_{min} \geq s+1$

From the codevector table $d_{min} = 3$

$$3 \geq s+1 \implies s = 2$$

So, Atmost two errors can be detected.

For correction $d_{min} \geq 2t+1$

$$3 \geq 2t+1$$

$$t \leq 1$$

So, Atmost one error can be corrected.

→ **Encoder of (7,4) Hamming code**

The encoder of (7,4) Hamming code is given below. This encoder is implemented for generator matrix $G = \left[\begin{array}{cccc:ccc} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right]$.

The parity bits can be obtained by

$$[C_1\ C_2\ C_3] = [m_1\ m_2\ m_3\ m_4] \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

$C_1 = m_1 \oplus m_2 \oplus m_3$

$C_2 = m_1 \oplus m_2 \oplus m_4$

$C_3 = m_1 \oplus m_3 \oplus m_4$

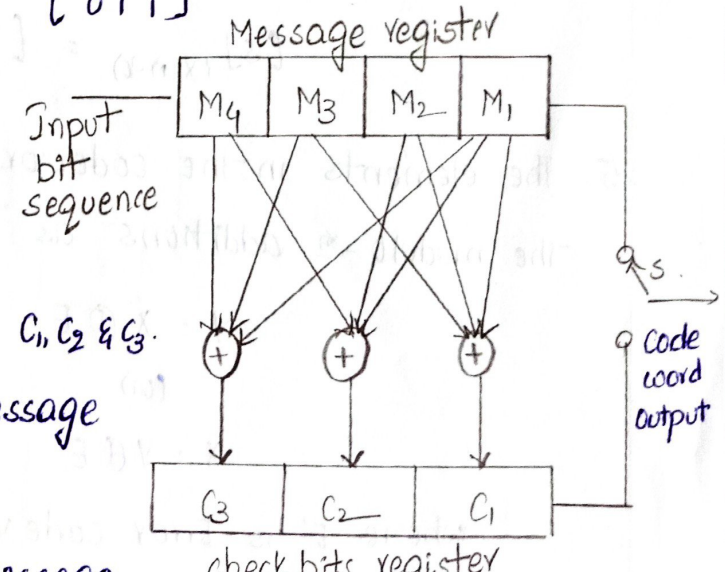


Fig: Encoder for (7,4) hamming code (or) (7,4) Linear block code.

The lower register contains check bits C_1, C_2 & C_3. These bits are obtained from the message bits by mod-2 additions.

The switch 's' is connected to message register first and all message bits are

transmitted. The switch is then connected to the check bit register and check bits are transmitted. This forms a block of '7' bits.

→ Syndrome Decoding

1. Let 'X' represent the transmitted code word and Y represent the received codeword.

2. If $X = Y$, no errors in the received signal and

   If $X \neq Y$, then some errors are present.

3. If $YH^T = (0, 0, \ldots 0)$ then $Y = X$ i.e there is no error.

   $YH^T \neq (0, 0, \ldots 0)$ then $Y \neq X$ i.e errors exists in the received code word.

4. Syndrome and its use for Error Detection

   The syndrome is defined as the Non-zero output of the product YH^T. Thus, the non-zero syndrome represents the presence of errors in the received codeword. The syndrome is represented by 'S' and is mathematically given as

$$S = YH^T$$

$$[S]_{1\times(n-k)} = [Y]_{1\times n}\,[H^T]_{n\times(n-k)}$$

5. The elements in the codeword vector 'Y' can be obtained by using the modulo-2 additions as under

$$Y = X \oplus E$$

(or)

$$X = Y \oplus E$$

where 'E' is Error code vector.

Relation between Syndrome and Error Vector:

The syndrome is given by

$$S = YH^T$$

Substituting $Y = (X \oplus E)H^T = XH^T \oplus EH^T$

Now $S = (X \oplus E)H^T = XH^T \oplus EH^T$

$$\boxed{S = EH^T} \quad [\because XH^T = 0]$$

→ To clear above concept of error correction using syndrome vector, Let us consider one particular example. For this, let us use the following parity check matrix.

$$[H] = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}_{3\times 7}$$

Sol. Assuming $X = [0\ 1\ 0\ 0\ 1\ 1\ 0)$ to be the transmitted code vector. Let the received code vector be obtained by assuming that the third bit is in error.

Thus $Y = (0\ 1\ \textcircled{1}\ 0\ 1\ 1\ 0)$, here encircled bit represent error.

We know that $S = YH^T$

$$= [0\ 1\ 1\ 0\ 1\ 1\ 0] \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$S = [1 \oplus 1 \oplus 1 \oplus 1 \oplus 1 \oplus 1 \oplus 1 \quad 1 \oplus 1 \oplus 1 \quad 1 \oplus 1 \quad 1]$$

$$S = [1\ 0\ 1]$$

The syndrome vector corresponds to the 3rd row of the transpose matrix H^T

The error vector corresponding to the syndrome (1,0,1) is given by

$$E = (0\,0\,1\,0\,0\,0\,0)$$

The correct vector X can be obtained as under

$$X = Y \oplus E$$

$$= [0\,1\,1\,0\,1\,1\,0] \oplus [0\,0\,1\,0\,0\,0\,0]$$

$$X = [0\,1\,0\,0\,1\,1\,0]$$

∴ This is same as the transmitted code vector.

→ CYCLIC CODES

A Binary code is said to be a cyclic code if it exhibits the following properties:

1. Linearity property
2. Cyclic property.

(i) Linearity Property : A code is said to be linear if sum of any two code words also a codeword. This property states that the cyclic codes are linear block codes.

(ii) Cyclic property: A code is said to be cyclic if any cyclic shift of a codeword results in the formation of another codeword.

Consider an n-bit codevector as shown below

$$X = \{x_{n-1}\ x_{n-2} \ldots x_1\ x_0\}$$

One cyclic shift of X gives

$$X = \{x_0\ x_{n-1}\ x_{n-2} \ldots x_1\}$$

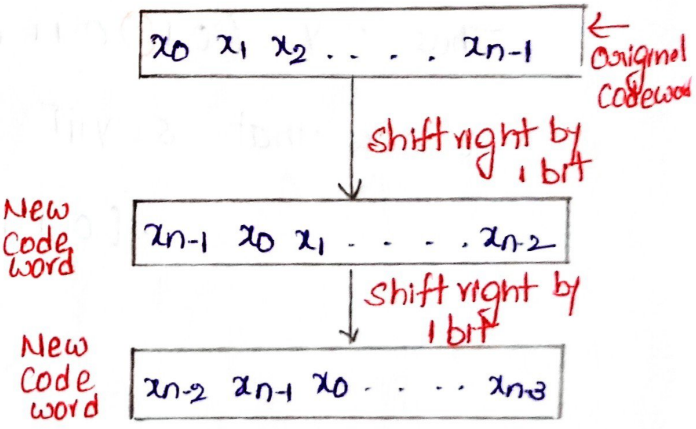


Fig: illustration of cyclic property of the cyclic codes.

Algebraic structures of cyclic codes

The codewords can be represented by a polynomial.

For example, consider the n-bit codeword,

$$X = (x_{n-1}, x_{n-2}, \ldots x_1 x_0)$$

The codeword can be represented by a polynomial of degree less than or equal to (n-1) i.e

$$X(p) = x_{n-1} p^{n-1} + x_{n-2} p^{n-2} + \ldots + xp + x_0 p^0$$

where X(p) is the polynomial of degree (n-1)

p is the arbitary variable of the polynomial.

Generation of code vectors in Non-systematic form:

Let $M = \{ m_{k-1}, m_{k-2}, \ldots m_1, m_0 \}$ be 'k' bits of message vector.

Then it can be represented by the polynomial as

$$M(p) = m_{k-1} p^{k-1} + m_{k-2} p^{k-2} + \ldots + mp' + m_0$$

Let X(p) is the codeword polynomial, It is given as

$$X(p) = M(p) \cdot G(p)$$

where G(p) is the generating polynomial of degree 'q'.

For an (n,k) cyclic code $q = n-k$ represent the no. of parity bits

$$G(p) = p^q + g_{q-1} p^{q-1} + \ldots + g_1 p + 1$$

where $g_1, g_2 \ldots g_{q-1}$ are the parity bits.

If $M_1, M_2, M_3 \ldots$ etc are the other message vectors. Then the corresponding code vectors can be calculated as

$$X_1(p) = M_1(p) \cdot G(p) \; ; \; X_2(p) = M_2(p) G(p) \; ; \; X_3(p) = M_3(p) \cdot G(p)$$

G(p) is same for all codevectors.

→ The generator polynomial of a (7,4) cyclic code is $G(p) = p^3+p+1$. Find all the codevectors for the code in Non-systematic form.

Sol. Given (7,4) cyclic code i.e $n = 7$

$k = 4$; $q = n-k$

$= 3$

$G(p) = p^3+p+1$

$k = 4$ i.e $2^4 = 16$ possible message words are occured

For Message (0 0 0 0) the codevector can be calculated as

$X_0(p) = M(p) \cdot G(p) = 0$

For $M = (0\ 0\ 0\ 1) \Rightarrow X_1(p) = 1 \cdot G(p)$

$= 1 \cdot (p^3+p+1) = p^3+p+1 = (0\ 0\ 0\ 1\ 0\ 1\ 1)$

For $M = (0\ 0\ 1\ 0) \Rightarrow X_2(p) = p(p^3+p+1) = p^4+p^2+p = (0\ 0\ 1\ 0\ 1\ 1\ 0)$

For $M = (0\ 0\ 1\ 1) \Rightarrow X_3(p) = (p+1)(p^3+p+1)$

$= p^4 \oplus p^2 \oplus p \oplus p^3 \oplus p \oplus 1$

$= p^4+p^3+p^2+p(1 \oplus 1)+1$

$= p^4+p^3+p^2+1$

$= (0\ 0\ 1\ 1\ 1\ 0\ 1)$

For $M = (0\ 1\ 0\ 0) \Rightarrow X_4(p) = p^2(p^3+p+1)$

$= p^5+p^3+p^2 = (0\ 1\ 0\ 1\ 1\ 0\ 0)$

For $M = (0\ 1\ 0\ 1) \Rightarrow X_5(p) = (p^2+1)(p^3+p+1)$

$= p^5+p^3+p^2+p^3+p+1$

$= p^5+p^2+p+1 = (0\ 1\ 0\ 0\ 1\ 1\ 1)$

For $M = (0110) \Rightarrow X_6(p) = (p^2+p)(p^3+p+1)$

$= p^5+p^3+p^2+p^4+p^2+p$

$= p^5+p^4+p^3+p = (0111010)$

For $M = (0111) \Rightarrow X_7(p) = (p^2+p+1)(p^3+p+1)$

$= p^5+p^3+p^2+p^4+p^2+p+p^3+p+1$

$= p^5+p^4+1 = (0110001)$

For $M = (1000) \Rightarrow X_8(p) = p^3(p^3+p+1)$

$= p^6+p^4+p^3 = (1011000)$

For $M = (1001) \Rightarrow X_9(p) = (p^3+1)(p^3+p+1)$

$= p^6+p^4+p^3+p^3+p+1$ = (

$= p^6+p^4+p+1 = (1010011)$

For $M = (1010) \Rightarrow X_{10}(p) = (p^3+p)(p^3+p+1)$

$= p^6+p^4+p^3+p^4+p^2+p$

$= p^6+p^3+p^2+p = (1001110)$

For $M = (1011) \Rightarrow X_{11}(p) = (p^3+p+1)(p^3+p+1)$

$= p^6+p^4+p^3+p^4+p^2+p+p^3+p+1$

$= p^6+p^2+1 = (1000101)$

For $M = (1100) \Rightarrow X_{12}(p) = (p^3+p^2)(p^3+p+1)$

$= p^6+p^4+p^3+p^5+p^3+p^2$

$= p^6+p^5+p^4+p^2 = (1110100)$

For $M = (1101) \Rightarrow X_{13}(p) = (p^3+p^2+1)(p^3+p+1)$

$= p^6+p^4+p^3+p^5+p^3+p^2+p^3+p+1$

$= p^6+p^5+p^4+p^3+p^2+p+1 = (1111111)$

For $M = (1110) \Rightarrow X_{14}(p) = (p^3+p^2+p)(p^3+p+1)$

$= p^6+p^4+p^3+p^5+p^3+p^2+p^4+p^2+p$

$= p^6+p^5+p = (1100010)$

For $M = (1111) \Rightarrow X_{15}(p) = (p^3+p^2+p+1)(p^3+p+1)$

$= p^6+p^4+p^3+p^5+p^3+p^2+p^4+p^2+p+p^3+p+1$

$= p^6+p^5+p^3+1 = (1101001)$

S.No	Message Bits (M)				Non-systematic code vectors (X)						
	m_3	m_2	m_1	m_0	x_6	x_5	x_4	x_3	x_2	x_1	x_0
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1	0	1	1
2	0	0	1	0	0	0	1	0	1	1	0
3	0	0	1	1	0	0	1	1	1	0	1
4	0	1	0	0	0	1	0	1	1	0	0
5	0	1	0	1	0	1	0	0	1	1	1
6	0	1	1	0	0	1	1	1	0	1	0
7	0	1	1	1	0	1	1	0	0	0	1
8	1	0	0	0	1	0	1	1	0	0	0
9	1	0	0	1	1	0	1	0	0	1	1
10	1	0	1	0	1	0	0	1	1	1	0
11	1	0	1	1	1	0	0	0	1	0	1
12	1	1	0	0	1	1	1	0	1	0	0
13	1	1	0	1	1	1	1	1	1	1	1
14	1	1	1	0	1	1	0	0	0	1	0
15	1	1	1	1	1	1	0	1	0	0	1

Table: code vectors of (7,4) cyclic code in Non-systematic form.

→ Generation of code vectors in systematic form

In systematic form of block code contain 'k' message bits followed by (n-k) parity / check bits. $q = n-k$

X = ('k' message bits : (n-k) check bits)

$= (m_{k-1}\ m_{k-2} \ldots m_1\ m_0 : c_{q-1}\ c_{q-2} \ldots c_1\ c_0)$

The checksum form a polynomial as

$$C(p) = c_{q-1}p^{q-1} + c_{q-2}p^{q-2} + \ldots + c_1 p + c_0$$

The checkbit polynomial can be obtained by

$$C(p) = \text{rem}\left[\frac{p^q M(p)}{G(p)}\right]$$

→ The generator polynomial of a (7,4) cyclic code is $G(p) = p^3+p+1$. Find all the code vectors for the code in systematic form.

Sol. Given cyclic code size is (7,4) i.e $n=7$; $k=4$; $q = n-k = 7-4 = 3$

Message vector = $(m_3\ m_2\ m_1\ m_0)$

$$M(p) = m_3p^3 + m_2p^2 + m_1p + m_0$$

There are $2^k = 2^4 = 16$ possible message vectors of 7 bits each.

Consider any message vector as

$$M = (m_3\ m_2\ m_1\ m_0) = (0\ 1\ 0\ 1)$$

$$M(p) = p^2+1$$

Given generator polynomial as $G(p) = p^3+p+1$

Then checkbit polynomial can be obtained by

$$C(p) = \text{rem}\left[\frac{p^q M(p)}{G(p)}\right] = \text{rem}\left[\frac{p^3(p^2+1)}{p^3+p+1}\right] = \text{rem}\left[\frac{p^5+p^3}{p^3+p+1}\right]$$

$$P^3+P+1 \;)\; P^5+0P^4+0P^3 \quad (P^2$$
$$\underline{P^5+0+P^3+P^2}$$
$$(P^2)$$

$$\therefore C(p) = \text{rem}\left[\frac{P^5+P^3}{P^3+P+1}\right] = P^2$$

From $q=3$ the checkbit polynomial as

$$C(p) = c_2P^2 + c_1P + c_0P^0$$
$$= 1\cdot P^2 + 0P + 0P$$

By comparing, the checkbit vector for given message vector is

$$[c_2\ c_1\ c_0] = [1\ 0\ 0]$$

∴ The codevector for message [0101] is [0101100]

→ Generator and Parity check Matrices of Non-systematic cyclic code.

The Generator polynomial can be represented as

$$G(p) = P^q + g_{q-1}P^{q-1} + \ldots\ g_1P + 1$$

Multiply both sides of this polynomial by P^i i.e

$$P^iG(p) = P^{i+q} + g_{q-1}P^{i+q-1} + \ldots + g_1P^{i+1} + P^i$$

where $i = (k-1), (k-2) \ldots 0$

where k = message bits.

→ Obtain the generator matrix corresponding to $G(p) = P^3+P^2+1$ for a (7,4) cyclic code.

Sol. Given cyclic code size is (7,4) i.e $n=7$, $k=4$, $q = 7-4 = 3$

$i = (k-1) \ldots 0$

$3, 2, 1, 0$

For Row 1 : $i=3 \Rightarrow P^3 G(p) = P^3(P^3+P^2+1) = P^6+P^5+P^3 \cdot (1101000)$

Row 2 : $i=2 \Rightarrow P^2 G(p) = P^2(P^3+P^2+1) = P^5+P^4+P^2 \cdot (0110100)$

Row 3 : $i=1 \Rightarrow PG(p) = P(P^3+P^2+1) = P^4+P^3+P = (0011010)$

Row 4 : $i=0 \Rightarrow P^0 G(p) = 1 \cdot (P^3+P^2+1) = P^3+P^2+1 = (0001101)$

Therefore the Generator Matrix for the given Generator polynomial is

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

→ Find out the possible generator polynomials (7,4) cyclic code. Find out the code vectors corresponding to these generator polynomials.

Sol. Given (7,4) cyclic code i.e $n=7$; $k=4$; $q=3$

We know the generator polynomial is the factor of P^n+1

For this example, the generator polynomial is the factor of P^7+1

The factors of P^7+1 is

$$P^7 \oplus 1 = (P \oplus 1)(P^3 \oplus P^2 \oplus 1)(P^3 \oplus P \oplus 1)$$

The valid generating polynomial is given by

$$G(p) = P^q + g_{q-1} P^{q-1} + \ldots + g_1 P + 1$$

The degree of generating polynomial should be 'q'

For this example $q=3$

Therefore the valid polynomials for (P^7+1) will be $(P^3 \oplus P^2 \oplus 1)$ & $(P^3 \oplus P \oplus 1)$

The generator polynomials for (7,4) cyclic code are

$$G_1(p) = P^3 \oplus P^2 \oplus 1$$

$$G_2(p) = P^3 \oplus P \oplus 1$$

→ Find a generator polynomial $g(x)$ for a (7,4) cyclic code and find code vectors for the data 0001

Sol. Given cyclic code size is (7, 4)

The factors of P^7+1 is $(P \oplus 1)(P^3 \oplus P^2 \oplus 1)(P^3 \oplus P \oplus 1)$

Therefore the generator polynomial is $P^3 \oplus P \oplus 1$

The codeword polynomial can be obtained as

$$X(P) = M(P) \cdot G(P)$$
$$= 1 \cdot (P^3 + P + 1)$$
$$= P^3 + P + 1$$

∴ The codevector is (0 0 0 1 0 1 1)

→ Systematic form of Generator Matrix

The systematic form of generator matrix is in the form of

$$G = [\, I_k \,:\, P_{k \times q} \,]_{k \times n}$$

The t^{th} row of this matrix will be represented in the polynomial form as

$$t^{th} \text{ row of } G = P^{n-t} + R_t(P)$$

where $t = 1, 2, 3 \ldots k$

Let's divide P^{n-t} by a generator matrix $G(P)$. Then we can express the result of this division in terms of quotient and remainder

$$\frac{P^{n-t}}{G(P)} = \text{Quotient} + \frac{\text{Remainder}}{G(P)}$$

$$\frac{P^{n-t}}{G(P)} = Q_t(P) + \frac{R_t(P)}{G(P)}$$

$$P^{n-t} = Q_t(P)\, G(P) \oplus R_t(P) \implies \boxed{P^{n-t} \oplus R_t(P) = Q_t(P) G(P)}$$

→ Findout the generator matrix for a systematic (7,4) cyclic code if $G(p) = p^3+p+1$ Also find the parity check matrix.

Given $G(p) = p^3+p+1$ i.e

Cyclic code size is (7,4) i.e $n=7$, $k=4$, $q=3$

We know that t^{th} row of generator polynomial can be represented by

$$p^{n-t} + R_t(p) = Q_t(p) \cdot G(p) \quad -① \text{ where } t = 1,2,3,4$$

(or)

$$\frac{p^{n-t}}{G(p)} = Q_t(p) + \frac{R_t(p)}{G(p)}$$

For $t=1$

$$\frac{p^6}{p^3+p+1}$$

$p^3+p+1)\ p^6\ (\ p^3+p+1$

$p^6+p^4+p^3$

p^4+p^3

p^4+p^2+p

p^3+p^2+p

$p^3+0+p+1$

p^2+1

$\therefore R_t(p) = p^2+1$

$Q_t(p) = p^3+p+1$

Putting $Q_t(p)$ & $R_t(p)$ in eq ①, we get

$$p^6+p^2+1 = (p^3+p+1)(p^3+p+1)$$

$\therefore$ 1st row polynomial is $p^6+p^2+1 = [1000101]$

2nd row polynomial

$t = 2$

$$\frac{p^5}{G(p)} = Q(t) + \frac{R(t)}{G(p)}$$

$p^3+p+1)\ p^5\ (p^2+1$

$p^5+p^3+p^2$

p^3+p^2

$p^3+0+p+1$

p^2+p+1

$R(t) = p^2+p+1$

$Q(t) = p^2+1$

$p^{n-t} + R_t(p) = Q(p) \cdot G(p)$

$p^5+p^2+p+1 = (p^2+1)(p^3+p+1)$

$[0\ 1\ 0\ 0\ 1\ 1\ 1]$

3rd row polynomial

$t = 3$

$$\frac{p^{n-t}}{G(p)} = Q(t) + \frac{R(t)}{G(p)}$$

$$\frac{p^4}{p^3+p+1} =$$

$p^3+p+1)\ p^4+0+0\ (p$

p^4+p^2+p

p^2+p

$p^{n-t} + R_t(p) = Q_t(p) * G(p)$

$p^4+p^2+p = p(p^3+p+1)$

The code vector is

$[0\ 0\ 1\ 0\ 1\ 1\ 0]$

For $t = 4$ 4th row polynomial

$$\frac{p^3}{p^3+p+1}$$

$p^3+p+1)\ p^3+0+0\ (1$

p^3+p+1

$p+1$

$p^{n-t} + R_t(p) = Q(p) \cdot G(p)$

$p^3+p+1 = p^3+p+1$

Codevector is

$[0\ 0\ 0\ 1\ 0\ 1\ 1]$

Therefore the Generator matrix is

$$G = \begin{bmatrix} 1\ 0\ 0\ 0 & 1\ 0\ 1 \\ 0\ 1\ 0\ 0 & 1\ 1\ 1 \\ 0\ 0\ 1\ 0 & 1\ 1\ 0 \\ 0\ 0\ 0\ 1 & 0\ 1\ 1 \end{bmatrix}_{4\times 7}$$

$$= [\, I_{4\times 4} : P_{4\times 3} \,]_{4\times 7}$$

The codevector can be obtained as

$$X = MG$$

Let us consider any 4 Bit message vector [1100] then codevector is

$$X = [1\ 1\ 0\ 0] \begin{bmatrix} 1000 & 101 \\ 0100 & 111 \\ 0010 & 110 \\ 0001 & 011 \end{bmatrix}$$

$$= [1100010]$$

All the message vectors from [0000 to 1111] and calculate codevectors like above.

(ii) To obtain parity check matrix

We know $G = [I_k \mid P_{kxq}]_{kxn}$

From the Generator matrix, We get the co-efficient (or) Parity matrix of size kxq

$$P = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}_{4x3}$$

Parity check matrix 'H' is in the form of $H = [P^T : I_{qxq}]_{qxn}$

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}_{3x7}$$

Cyclic Redundancy Check (CRC)

A codeword can be generated for a given dataword (message) polynomial M(p) with the help of long division.

A sequence of redundant bits called CRC or CRC remainder is appended at the end of a data unit such as byte.

CRC Generator

The procedure for CRC generation is as under:

1. We append a string of 'n' 0s to the data unit where 'n' is '1' less than the no. of bits in the predecided divisor (n+1) bit long.
2. We divide the newly generated data unit in step 1 by the divisor.
3. The remainder obtained after the division in step 2 is the n-bit CRC.
4. This CRC will replace the 'n' 0s appended to the data unit in step 1, to get the codeword to be transmitted as shown in below figure.

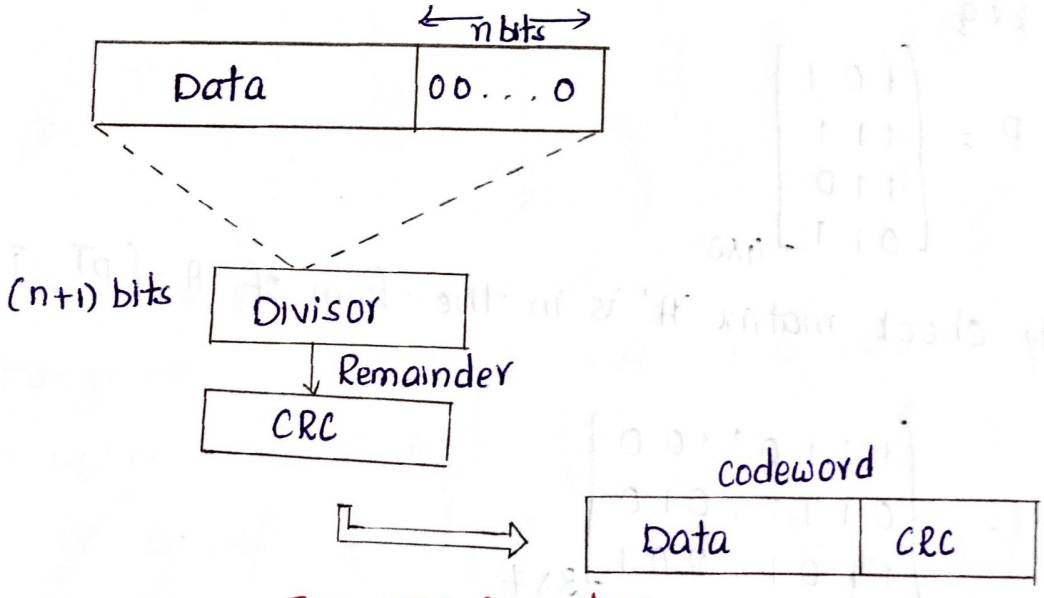
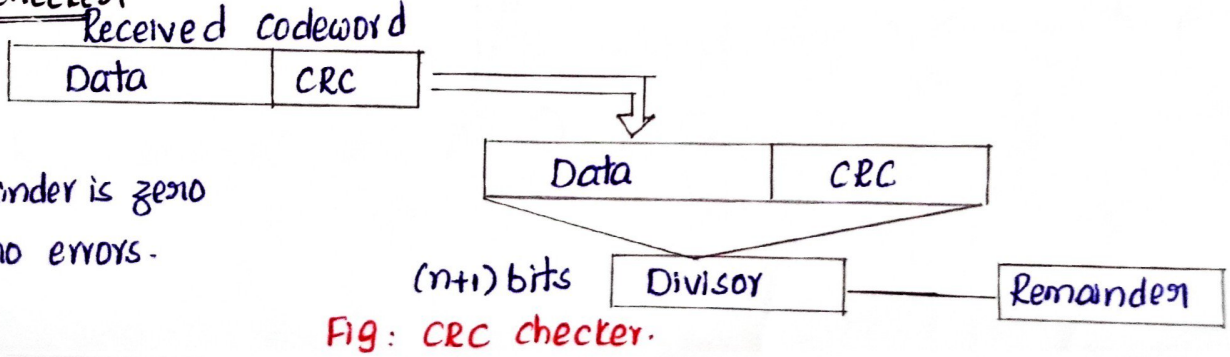


Fig: CRC Generator

CRC checker



If Remainder is zero then no errors.

Fig: CRC checker.

→ A bit stream 10011101 is transmitted using the standard CRC method. The generator polynomial is x^3+1. Show the actual bit string transmitted. Suppose the third bit from left is inverted during transmission. Show that this error is detected at the receiver's end.

Sol) Given that, data word (Bit string) : 10011101

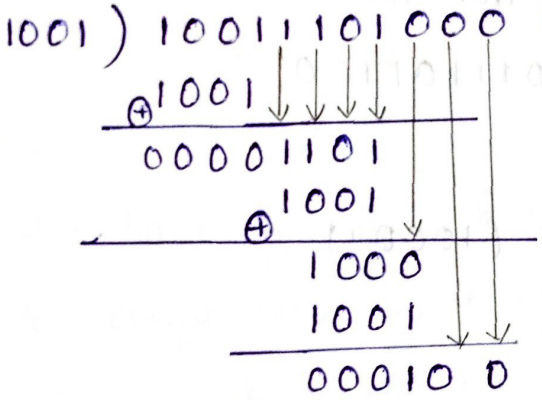
Generator polynomial : $x^3+1 = 1001 = n = 4$

We know that

Dividend = Dataword + (n-1) zeros
3(n-1)

= 10011101 000

Now, let us carryout the division as under



∴ Remainder is 100

The transmitted word may be obtained by writing the data word followed by the remainder as under :

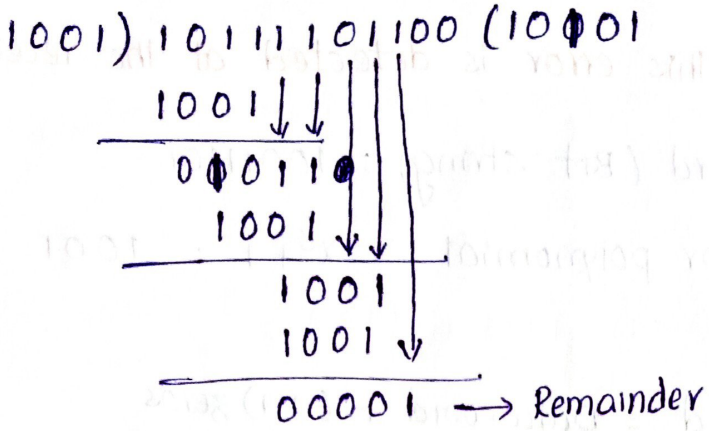
∴ Transmitted word = 10011101 100
Data word Remainder

Now, let us write the erroneous received word.

Given the third bit from the left is inverted during transmission

The received word is 10①11101100
↳ Error

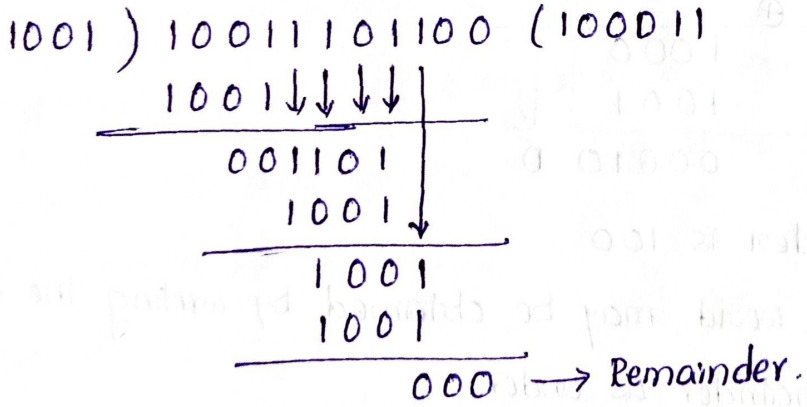
At the receiver, this word is divided by the same divisor used at the transmitter i.e 1001



A Non-zero remainder shows that there is an error in the received codeword.

— o —

Let us consider there is no error for the received codeword The received codeword is 10011101100



Remainder zero indicates there is no error for received codeword.

→ Encoding using an (n-k) Bit shift register

The below figure shows the block diagram of a generalized (n, k) cyclic code. The symbols used to draw encoders are —[r_1]→ flip flops

(S) if $g=1$ closed path and if $g=0$ is open path (no connection).

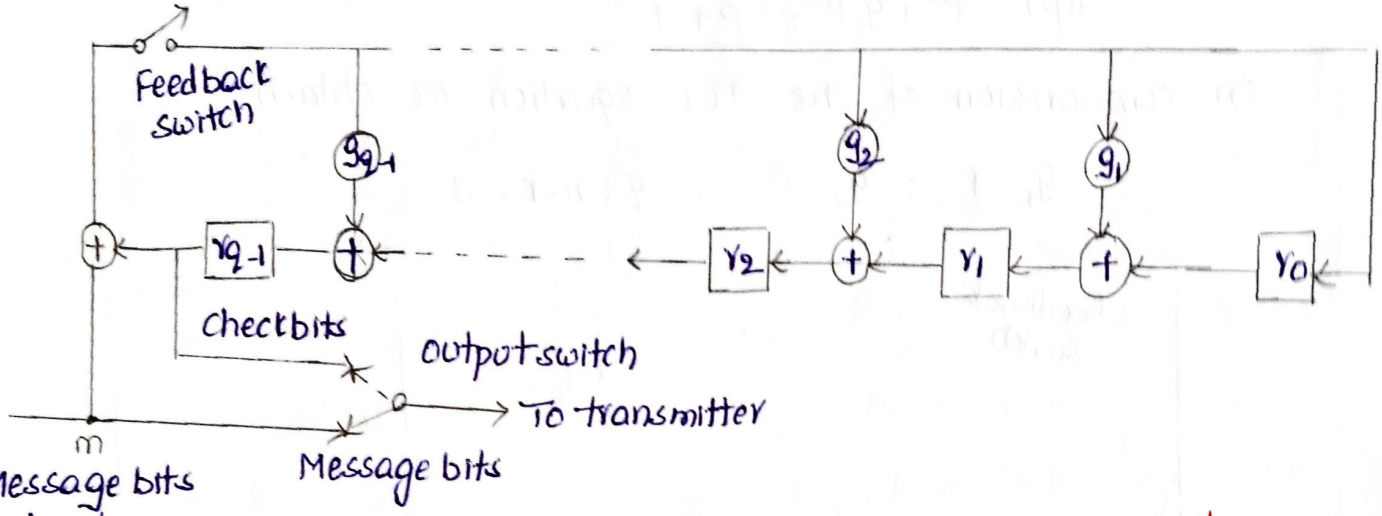


Fig: Encoder for systematic (n, k) cyclic code

Operation:

The feedback switch is first closed. The output switch is connected to message input. All the shift registers are initialized to all zero state. The 'k' message bits are shifted to the transmitter as well as shifted to the registers.

After the shift of 'k' message bits the registers contain 'q' check bits. The feedback switch is now opened and output switch is connected to check bits position. With the every shift, the check bits are then shifted to the transmitter.

→ Design the encoder for the (7,4) cyclic code generated by $G(p) = p^3 + p + 1$ and verify its operation for any message vector.

Sol. The generator polynomial is

$$G(p) = p^3 + 0p^2 + p + 1$$

$$G(p) = p^3 + g_2 p^2 + g_1 p + 1$$

On comparision of the two equation we obtain

$$g_1 = 1 \; ; \; g_2 = 0 \; ; \; q = n - k = 3$$

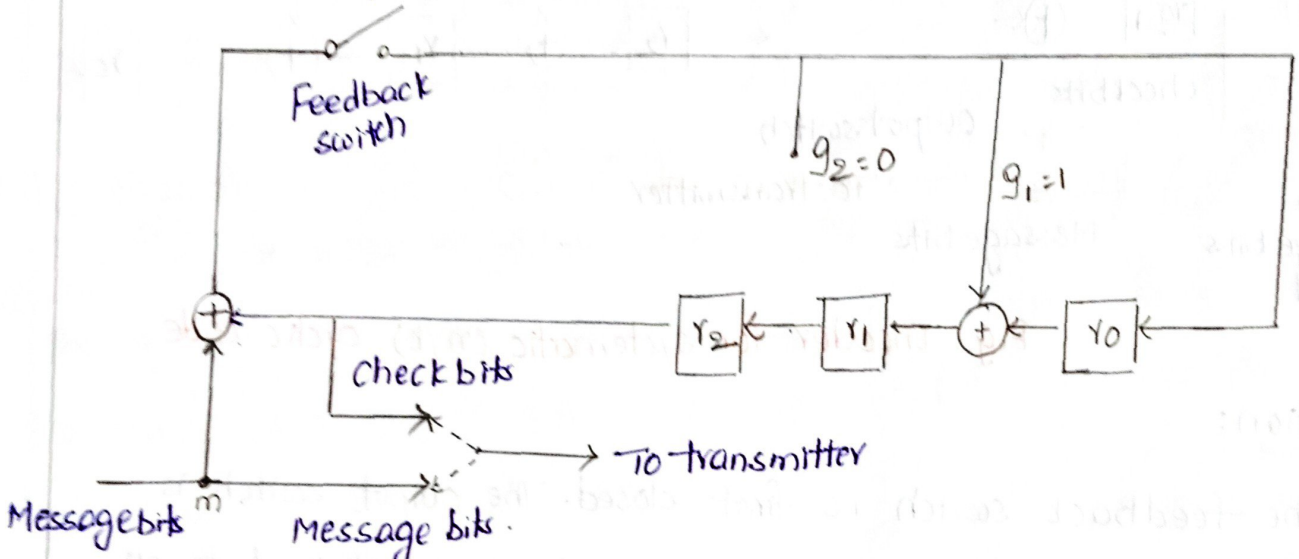


Fig: Encoder for (7,4) cyclic code for $G(p) = p^3 + p + 1$

Since $q = 3$, there are '3' flip-flops in shift register to hold check bits c_1, c_2 and c_3. Since $g_0 = 0$, the link is not connected. $g_1 = 1$, hence its link is connected.

Let verify the operation of this encoder for message vector $M = (m_3\ m_2\ m_1\ m_0) = (1\ 1\ 0\ 0)$. The below table shows the contents of shift register before and after shifts.

I/p message bit	Register bit i/ps before shift			Register bit outputs after shift		
m	$r_2 = r_2'$	$r_1 = r_1'$	$r_0 = r_0'$	$r_2' = r_1$	$r_1' = r_0 \oplus r_2 \oplus m$	$r_0' = r_2 \oplus m$
-	0	0	0	0	0	0
1	0	0	0	0	$0 \oplus 0 \oplus 1 = 1$	$0 \oplus 1 = 1$
1	0	1	1	1	$1 \oplus 0 \oplus 1 = 0$	$0 \oplus 1 = 1$
0	1	0	1	0	$1 \oplus 1 \oplus 0 = 0$	$1 \oplus 0 = 1$
0	0	0	1	0	$1 \oplus 0 \oplus 0 = 1$	$0 \oplus 0 = 0$

Table: shift register bits positions for input message M = (1100)

The above table shows that at the end of last message bit the register bit outputs are $r_2' = 0$, $r_1' = 1$ and $r_0' = 0$. The feedback switch is opened and output switch is closed to check bits position. The check bits are then shifted to the transmitter.

$\therefore X = (1100010)$

Shift clock	Message bit m	Shift register outputs r_2'	r_1'	r_0'	Feedback switch ON/OFF	Output switch position	Transmitted bits
1	1	0	1	1	ON	Message	1
2	1	1	0	1	ON	Message	1
3	0	0	0	1	ON	Message	0
4	0	0	1	0	ON	Message	0
5	-	0	1	0	OFF	check bits	0 (r_2')
6	-	1	0	0	OFF	check bits	1 (r_2')
		$r_2' = r_1'$	$r_1' = r_0'$	$r_2' = 0$			
7	-	0	0	0	OFF	check bits	0 (r_2')

Table: Operation of (7,4) cyclic code encoder

Block diagram of Syndrome calculator

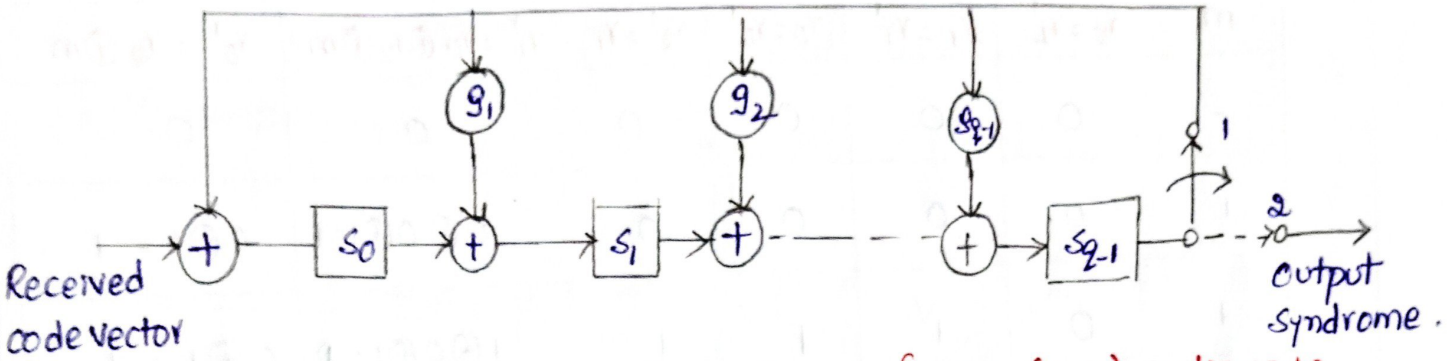


Fig: Computation of syndrome for an (n, k) cyclic code.

The above figure shows the generalized block diagram of a syndrome calculator. There are 'q' stage shift register to generate 'q' bit syndrome vector. The operation as follows -

Initially all the shift registers contents are zero and the switch is closed in position 1. The received vector 'y' is shifted bit by bit into the shift register. The contents of flip-flops keep on changing according to input bits 'y' and values of g_1, g_2 etc

After all the bits of 'y' are shifted, the 'q' flip-flops of shift register contains the q-bit syndrome vector.

The switch is then closed to position 2 and clocks are applied to the shift register. The output is a syndrome vector

$S = (s_{q-1}, s_{q-2}, \ldots s_1 s_0)$

→ Design a syndrome calculator for a (7,4) cyclic hamming code generated by the polynomial $G(p) = p^3+p+1$. Calculate the syndrome for $Y = (1\,0\,0\,1\,1\,0\,1)$.

Sol. Given (7,4) cyclic Hamming code i.e $n=7$, $k=4$, $q = n-k = 3$.

The given generator polynomial is

$$G(p) = p^3 + 0p^2 + p + 1$$

$$: p^3 + g_2 p^2 + g_1 p + 1$$

By comparing $g_2 = 0$; $g_1 = 1$

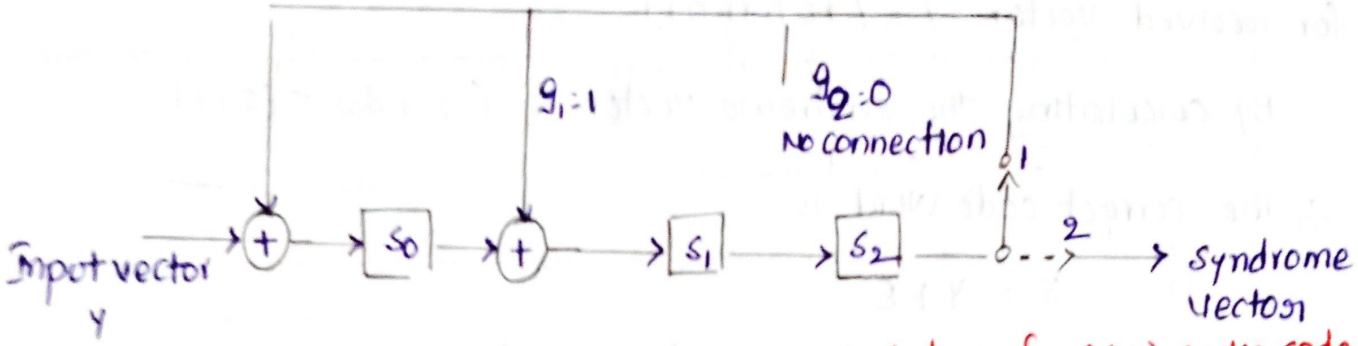


Fig: Block diagram of a syndrome calculator for (7,4) cyclic code with $G(p) = p^3 + p + 1$

Operation and Explanation

Shift	Received vector i.e bits of Y	Contents of flipflops in shift register		
		$S_0 = Y \oplus S_2$	$S_1 = S_0 \oplus S_2$	$S_2 = S_1$
-	-	0	0	0
1	1 →	$1 \oplus 0 = 1$	0	0
2	0	0	$1 \oplus 0 = 1$	0
3	0	0	0	1
4	1	0	1	0
5	1	1	0	1
6	0	1	0	0
7	1 syndrome →	1	1	0

Table: Calculation of syndrome for Y = (1001101)

The table shows that at the end of last shift the register contents are $(S_0\ S_1\ S_2) = (1\ 1\ 0)$

The switch is kept in position 1 until all the '7' bits of received vector 'Y' are shifted into the shift register. The flip-flops of the shift register contain syndrome vector when all bits 'Y' are shifted. This gives syndrome vector at the output.

The above table illustrates the operation of this syndrome calculator for received vector $Y = (1001101)$.

By calculating the syndrome vector is $(S_2\, S_1\, S_0) = (011)$

∴ The correct code word is

$$X = Y + E$$
$$= (1001101) \oplus (0001000)$$
$$= (1000101)$$

→ Bose - Chaudhri - Hocquenghem (BCH) codes

BCH codes are one of the most important and powerful linear block codes. The BCH codes are cyclic codes with a wide variety of parameters. The most common BCH codes are characterized as under:

For any positive integer m (with $m \geq 3$) and $t < \frac{(2^m - 1)}{2}$, there exists a BCH code with the following parameters:

1. Block length : $n = 2^m - 1$
2. No. of message bits : $K \geq n - mt$
3. Minimum distance : $d_{min} \geq 2t + 1$

Each BCH code can correct 't' no. of errors. This means that it can detect and correct upto 't' random errors per codeword.

The block lengths and code rates of BCH codes are variable. In that sense they provide a high flexibility. For block lengths of a few hundred, the BCH codes are among the best known codes of the same block length and code rate.

The below table shows the code parameters and generator polynomials for the binary BCH codes of length upto 2^5-1

m	n	K	t	Generator Polynomial
3	7	4	1	1 011
4	15	11	1	1 10 011
4	15	7	2	111 010 001
5	31	26	1	100 101
5	31	21	2	11 101 101 001
5	31	16	3	1 000 111 110 101 111
5	31	11	5	101 100 010 011 011 010 101
5	31	6	7	11 001 011 011 110 101 000 100 111

Table: Binary BCH codes of length upto 2^5-1

where n = block length = 2^m-1

k = no. of message bits = $k \geq n-mt$

t = Maximum no. of detectable errors $\left(\frac{2^m-1}{2}\right)$

Let us consider that we want to construct the generator polynomial for (15,7) BCH code, then we refer to third row of the above table. The generator polynomial from the third row is given by

111 010 001

Hence, the generator polynomial is

$$G(p) = p^8 + p^7 + p^6 + 0p^5 + p^4 + 0p^3 + 0p^2 + 0p + 1$$

$$G(p) = p^8 + p^7 + p^6 + p^4 + 1 \quad \text{Ans}$$

CONVOLUTIONAL CODES

The main difference between the block codes and the convolutional codes are listed below:

1. **Block codes:** In Block codes, 'n' bits generated by the encoder in a particular time slot depends only on the block of 'k' message bits with in that time slot.

2. **Convolutional code:** In convolutional codes, 'n' bits generated by the encoder in a time slot depends not only on the 'k' message bits within that time slot, but also on the preceeding 'L' blocks of the message bits ($L > 1$).

Applications of Convolutional code

Block codes are more suitable for error detection and the convolutional codes are more suitable for error correction.

3 Encoding and Decoding

Encoding of the convolutional codes can be accomplished using simple shift register. Several practical methods have been developed for decoding.

Convolutional codes perform better than the block codes in many error control applications.

→ Few definitions Related to Convolutional codes

1. **Code Rate (r):** It is defined as the ratio of no. of message bits to the no. of encoded bits per message.

$$r = \frac{k}{n}$$

$$= \frac{1}{2}$$

k = No. of message bits = 1

n = No. of encoded bits per message = 2

→ **Constraint Length (K)**

It is defined as the no. of shifts over which a single message bit can influence the encoder output. It is expressed in terms of message i.e K=3

→ **Code Dimension:**

The code dimension of a convolutional code depends on n, k and L.

Here 'k' represents the no. of message bits taken at a time by the encoder.

'n' is the no. of encoded bits per message bit.

L is encoders memory.

Therefore, the dimension of the code is represented by (n, k, L)

→ **Encoder for convolutional coding**

The fundamental hardware unit for convolutional encoding is a taped shift register with (L+1) stages as shown in below figure. Here $g_0, g_1 \ldots$ etc are the tap gains which are binary digits 0's and 1's.

A tap gain of '0' represents an open circuit whereas tap gain of 1 represents a short circuit.

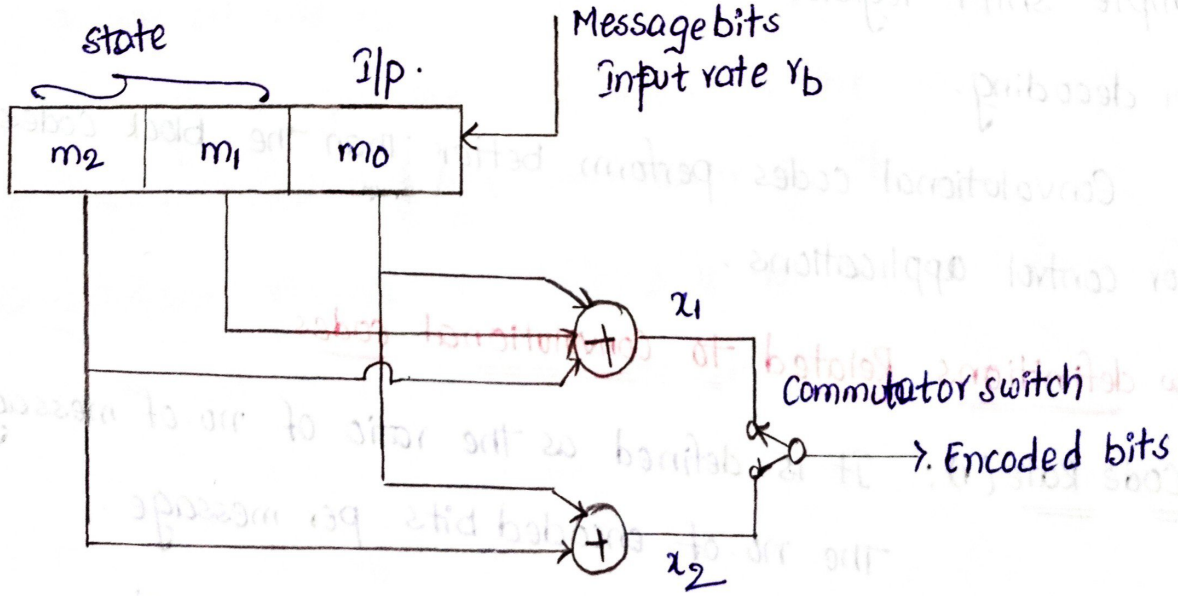


Fig: Convolutional Encoder with n=2, k=1 and L=2

The message bits enter one by one into the tapped shift register, which are then combined by mod-2 addition to form the encoded bit x. Therefore, we have

$$X = m_L g_L \oplus \ldots \oplus m_1 g_1 \oplus m_0 g_0 \quad -(1)$$

$$= \sum_{i=0}^{L} m_i g_i$$

The name convolutional encoding comes from the fact that equation (1) has a form of binary convolution which analogous to the convolutional integral. The message bit m_0 represents the current input where as the bits m_1 m_2 represent the past input or state of the shift register.

The encoder output depends on the current message bit m_0 and the state of the shift register defined by the previous 'L' message bits.

To provide an extra bit needed for error control, a complete convolutional encoder must generate output bits at a rate greater than the message bit rate r_b. This is achieved by using two or more modulo-2 adders to the registers as shown in the figure and interleaving the encoded bits via commutator switch.

The convolutional encoder of figure 1 is for $n=2$, $k=1$ and $L=2$. It therefore generates $n=2$ encoded bits as under:

$$x_1 = m_0 \oplus m_1 \oplus m_2$$

$$x_2 = m_0 \oplus m_2$$

The commutator switch selects these encoded bit alternately to produce the stream of encoded bits as under.

$$X = x_1 x_2 x_1' x_2' x_1'' x_2''$$

→ Time Domain Approach

It is defined in terms of n-impulse responses. Let the impulse response of the adder generating x_1 in fig 1 is given by, the sequence $\{g_0^{(1)}, g_1^{(1)}, \ldots g_L^{(1)}\}$. similarily, let the sequence $\{g_0^{(2)}, g_1^{(2)}, \ldots g_L^{(2)}\}$ denote the impulse response of the adder generating x_2. These impulse responses are also called as generator sequences of the code.

Let $(m_0, m_1, \ldots m_l)$ denote the message sequence entering the encoder of figure 1. one bit at a time (starting from m_0). The encoder generates two output sequences by performing convolutions on the message sequences and impulse responses.

The bit sequence x_1 is given by

$$x_1 = x_i^{(1)} = \sum_{l=0}^{L} g_l^{(1)} m_{i-l}, \quad i = 0, 1$$

Similarily, the other bit sequence x_2 is given by

$$x_2 = x_i^{(2)} = \sum_{l=0}^{L} g_l^{(2)} m_{i-l}, \quad i = 0, 1$$

The output is

$$X = \{x_0^{(1)} x_0^{(2)} x_1^{(1)} x_1^{(2)} x_2^{(1)} x_2^{(2)} \ldots\}$$

where $x_1 = x_0^{(1)} x_1^{(1)} \ldots$.

$x_2 = x_0^{(2)} x_1^{(2)} x_2^{(2)} \ldots$

Note: Each message bit influences a span of $n(L+1) = 6$ successive output bits

$n = 2$, $L = 2$ $\quad 2(2+1) = 6$

→ The convolutional encoder of figure 2 has the following two generator sequences each of length 3.

$(g_0^{(1)}\ g_1^{(1)}\ g_2^{(1)}) = (1,1,1)$; $(g_0^{(2)}\ g_1^{(2)}\ g_2^{(2)}) = (1\ 0\ 1)$

Determine the encoded sequence for the following input message:

$(m_0\ m_1\ m_2\ m_3\ m_4) = (1\ 0\ 0\ 1\ 1)$

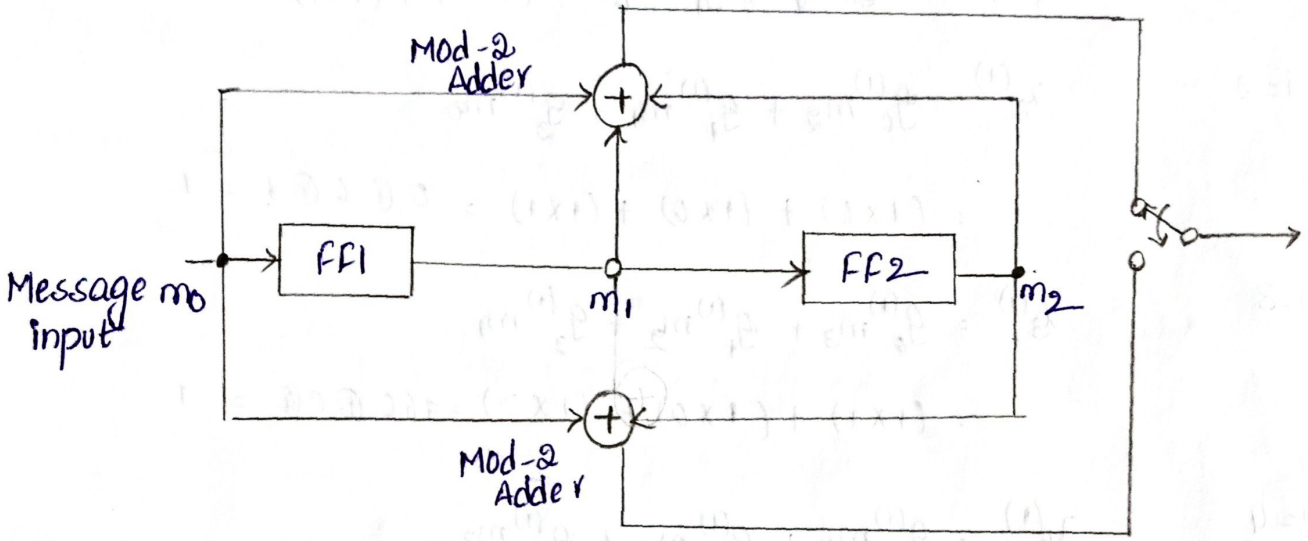
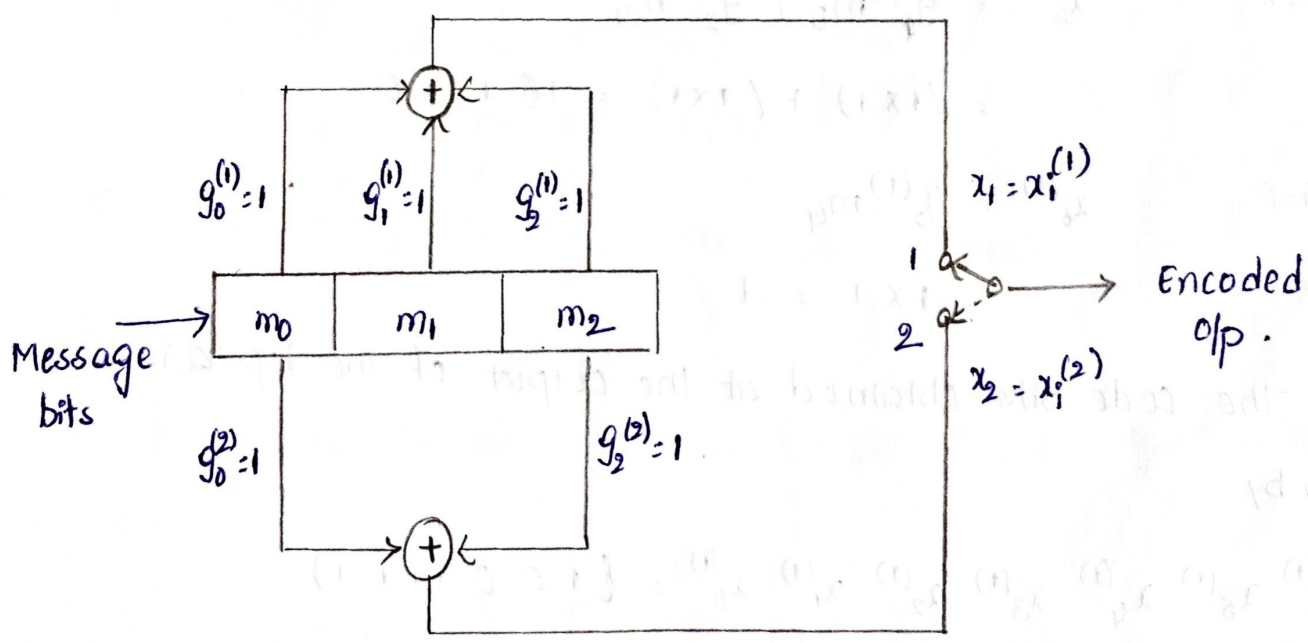


Fig 2: Convolutional Encoder

Sol] The given encoder can be drawn in the standard form:



1. To obtain the Bit stream $x_i^{(1)}$

We have

$$x_1 = x_i^{(0)} = \sum_{l=0}^{L} g_l^{(1)} m_{i-l} \quad i = 0, 1, 2 \ldots$$

$i = 0$

$$= x_0^{(1)} = g_0 m_0 = 1 \times 1 = 1$$

For $i = 1$

$$= x_1^{(1)} = g_0^{(1)} m_1 + g_1^{(1)} m_0 = (1 \times 0) + (1 \times 1) = 1$$

For $i = 2$

$$x_2^{(1)} = g_0^{(1)} m_2 + g_1^{(1)} m_1 + g_2^{(1)} m_0$$

$$= (1 \times 0) + (1 \times 0) + (1 \times 1) = 0 \oplus 0 \oplus 1 = 1$$

For $i = 3$

$$x_3^{(1)} = g_0^{(1)} m_3 + g_1^{(1)} m_2 + g_2^{(1)} m_1$$

$$= (1 \times 1) + (1 \times 0) + (1 \times 0) = 1 \oplus 0 \oplus 0 = 1$$

For $i = 4$

$$x_4^{(1)} = g_0^{(1)} m_4 + g_1^{(1)} m_3 + g_2^{(1)} m_2$$

$$= (1 \times 1) + (1 \times 1) + (1 \times 0) = 1 \oplus 1 \oplus 0 = 0$$

For $i = 5$

$$x_5^{(1)} = g_0^{(1)} m_4 + g_2^{(1)} m_3$$

$$= (1 \times 1) + (1 \times 1) = 1 \oplus 1 = 0$$

For $i = 6$

$$x_6^{(1)} = g_2^{(1)} m_4$$

$$= 1 \times 1 = 1$$

Thus the code bits obtained at the output of the top adder is given by

$$x_6^{(1)}\ x_5^{(1)}\ x_4^{(1)}\ x_3^{(1)}\ x_2^{(1)}\ x_1^{(1)}\ x_0^{(1)} = (1\ 0\ 0\ 1\ 1\ 1\ 1)$$

2. To obtain the Bit stream $x_i^{(2)}$:

The bit stream $x_i^{(2)}$ can be obtained at the bottom adder. It is given by

$$x_i^{(2)} = x_2 = \sum_{l=0}^{L} g_l^{(2)} m_{i-l}, \quad i = 0, 1, 2, 3 \ldots$$

Substituting values of i in equ $x_i^{(2)}$ we get,

$g_i = \{1\ 0\ 1\}$ (0 1 2)

$m = \{1\ 0\ 0\ 1\ 1\}$ (0 1 2 3 4)

For $i=0 \Rightarrow x_0^{(2)} = g_0^{(2)} m_0 = 1 \times 1 = 1$

For $i=1 \Rightarrow x_1^{(2)} = g_0^{(2)} m_1 + g_1^{(2)} m_0$

$= (1 \times 0) + (0 \times 1) = 0 \oplus 0 = 0$

For $i=2 \Rightarrow x_2^{(2)} = g_0^{(2)} m_2 + g_1^{(2)} m_1 + g_2^{(2)} m_0$

$= (1 \times 0) + (0 \times 0) + (1 \times 1) = 1$

For $i=3 \Rightarrow x_3^{(2)} = g_0^{(2)} m_3 + g_1^{(2)} m_2 + g_2^{(2)} m_1$

$= (1 \times 1) + (0 \times 0) + (1 \times 0) = 1$

For $i=4 \Rightarrow x_4^{(2)} = g_0^{(2)} m_4 + g_1^{(2)} m_3 + g_2^{(2)} m_2$

$= (1 \times 1) + (0 \times 1) + (1 \times 0) = 1$

For $i=5 \Rightarrow x_5^{(2)} = g_1^{(2)} m_4 + g_2^{(2)} m_3 = (0 \times 1) + (1 \times 1) = 1$

For $i=6 \Rightarrow x_6^{(2)} = g_2^{(2)} m_4 = 1 \times 1 = 1$

No of iterations = L + memory of encoder

$5 + 2$

$= 7$

o/p bits $n(L + \text{memory state})$

$2(5+2)$

14 bits

The code bits obtained at the output of the bottom adder is given by

$x_6^{(2)} x_5^{(2)} x_4^{(2)} x_3^{(2)} x_2^{(2)} x_1^{(2)} x_0^{(2)} = (1111101)$

Therefore the output encoded sequence is

codeword : 11 11 11 11 01 01 11

→ Transform domain Approach:

We know that the convolution in time domain is transformmed into multiplication in frequency domain

The impulse response of top adder in figure 2 is

$$\{g_0^{(1)} \; g_1^{(1)} \; g_2^{(1)}\} = \{1 \; 1 \; 1\}$$

The impulse response of bottom adder is

$$\{g_0^{(2)} \; g_1^{(2)} \; g_2^{(2)}\} = \{1 \; 1 \; 1\}$$

The Generator polynomial is

$$G^{(1)}(p) = g_0^{(1)} + g_1^{(1)} p + g_2^{(1)} p^2$$

By substituting the values of $\{g_0^{(1)} \; g_1^{(1)} \; g_2^{(1)}\}$ we get

$$G^{(1)}(p) = 1 + p + p^2$$

The Generator polynomial for bottom adder is

$$G^2(p) = g_0^{(2)} + g_1^{(2)} p + g_2^{(2)} p^2$$

$$= 1 + p^2$$

The codeword polynomial to top adder is given by

$$x^{(1)}(p) = G^{(1)}(p) \cdot m(p)$$

The codeword polynomial for bottom adder is given by

$$x^{(2)}(p) = G^{(2)}(p) \cdot m(p)$$

where $m(p)$ is the message polynomial.

→ Determine the codeword for the cyclic encoder of figure 2 for the message signal (10011), using the transform domain approach The impulse response of the input top adder output path is (1 1 1) and that of the input bottom adder output path is (1,0,1).

Sol. Given data message signal $(m_0\ m_1\ m_2\ m_3\ m_4)$ is (1 0 0 1 1)

$$m(p) = 1 + 0.p + 0.p^2 + 1.p^3 + 1.p^4$$

$$= 1 + p^3 + p^4$$

Generator polynomial for top adder for given impulse sequence is

$$\{g_0^{(1)}, g_1^{(1)}, g_2^{(1)}\} = \{1, 1, 1\}$$

$$G^{(1)}(p) = g_0^{(1)} + g_1^{(1)} p + g_2^{(1)} p^2$$

$$= 1 + p + p^2$$

codeword polynomial for top adder is

$$x^{(1)}(p) = m(p) \cdot G^{(1)}(p)$$

$$= (1 + p^3 + p^4)(1 + p + p^2)$$

$$= 1 + p + p^2 + p^3 + p^4 + p^5 + p^4 + p^5 + p^6$$

$$= p^6 + p^5(1 \oplus 1) + p^4(1 \oplus 1) + p^3 + p^2 + p + 1$$

$$= p^6 + p^3 + p^2 + p + 1 \quad (\text{or}) \quad 1 + p + p^2 + p^3 + p^6$$

∴ The codeword is (1 1 1 1 0 0 1)

Generator polynomial for bottom adder is given by

$$x^2(p) = G^2(p)\, m(p)$$

$$= (1 + p^2)(1 + p^3 + p^4)$$

$$= 1 + p^3 + p^4 + p^2 + p^5 + p^6$$

$$= 1 + p^2 + p^3 + p^4 + p^5 + p^6$$

The code sequence is (1 0 1 1 1 1 1)

Hence, the final codeword at the output of the encoder is obtained by multiplexing (interleaving) the two code sequences

$\therefore$ codeword = 11 10 11 11 01 01 11

Advantage of Transform Domain Approach

The codeword obtained using the time domain approach and the frequency (i.e transform) domain approach get the same result. But the computation using transform domain approach demands less efforts than the time-domain approach.

→ Graphical Representation for Convolutional Encoding

For the convolutional encoding, there are 3 different Graphical representations are widely used. They are related to each other. They are:

1. Code Tree
2. code Trellis
3. state diagram

Code Tree:

Let us draw the code tree for the (2,1,2) cyclic encoder of figure 3. We assume that the register has been cleared so that it contains all zeros, when the first message bit m_0 arrives. Therefore, the initial state is $m_1 m_2 = 00$. Now, if $m_0 = 0$ then, the encoder output as under

If $m_0 = 0$ $\quad x_1 = m_0 \oplus m_1 \oplus m_2$

$= 0 \oplus 0 \oplus 0 = 0$

$x_2 = m_0 \oplus m_2 = 0 \oplus 0 = 00$

$\therefore x_1 x_2 = 00$

If $m_0 = 1$, then we have

$$x_1 = 1 \oplus 0 \oplus 0 = 1$$

$$x_2 = 1 \oplus 0 = 1$$

$$\therefore x_1 x_2 = 11$$

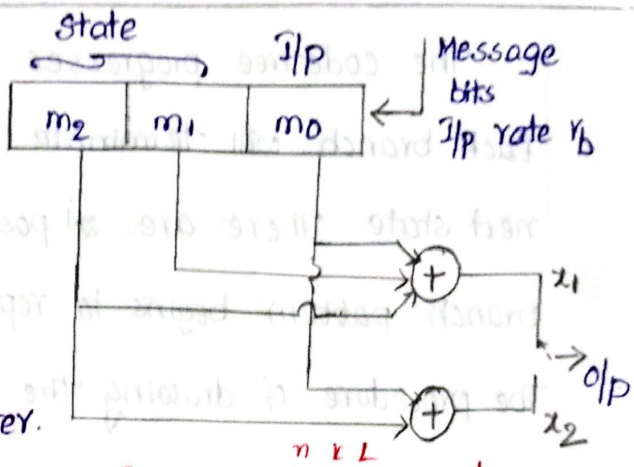


Figure 3 : (2,1,2) Encoder

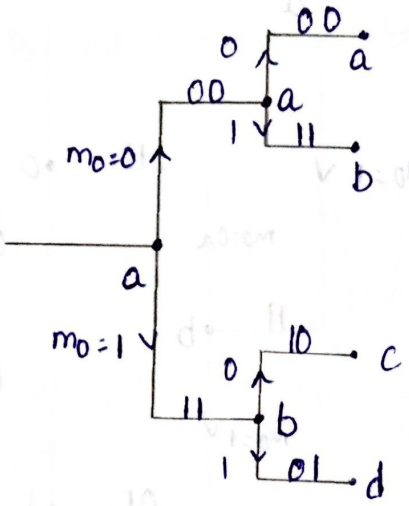
In the given encoder two flipflops are used for memory. It gives the state of the register. Therefore 4 possible states are occured. They are named as follows:

state	m_2	m_1
a	0	0
b	0	1
c	1	0
d	1	1

Table : states of shift register

The code tree has been drawn in figure 4 It begins at a branch point on node 'a' which represents the initial state. Hence, if $m_0 = 0$ we should take the upper branch from 'a' to obtain the output 00 and the next state after shifting one bit is $m_2 m_1 = 00$ i.e state 'a'.

If $m_0 = 1$ we should take the lower branch from 'a' to obtain the output '11' and the next state after shifting one bit is $m_2 m_1 = 01$ i.e state state 'b'



Now at state b

$m_2 m_1 = 01$

If $m_0 = 0 \rightarrow x_1 = 0 \oplus 1 \oplus 0 = 1$

$x_2 = 0 \oplus 0 = 0$

$m_0 = 1 \rightarrow x_1 = 1 \oplus 1 \oplus 0 = 0$

$x_2 = 1 \oplus 0 = 1$

Output	Nextstate	Present state
10	$m_2 m_1$ = 10 (c)	m_2 m_1 m_0 = 0 1 0
01	$m_2 m_1$ = 11 (d)	m_2 m_1 m_0 = 0 1 1

The code tree progresses in this manner for each new message bit. Each branch will terminate at another node which is labelled with the next state. There are 2^j possible branches for the j^{th} message bit, but the branch pattern begins to repeat at $j=3$ since the register length is $L+1=3$. The procedure of drawing the code tree is illustrated in Fig 5.

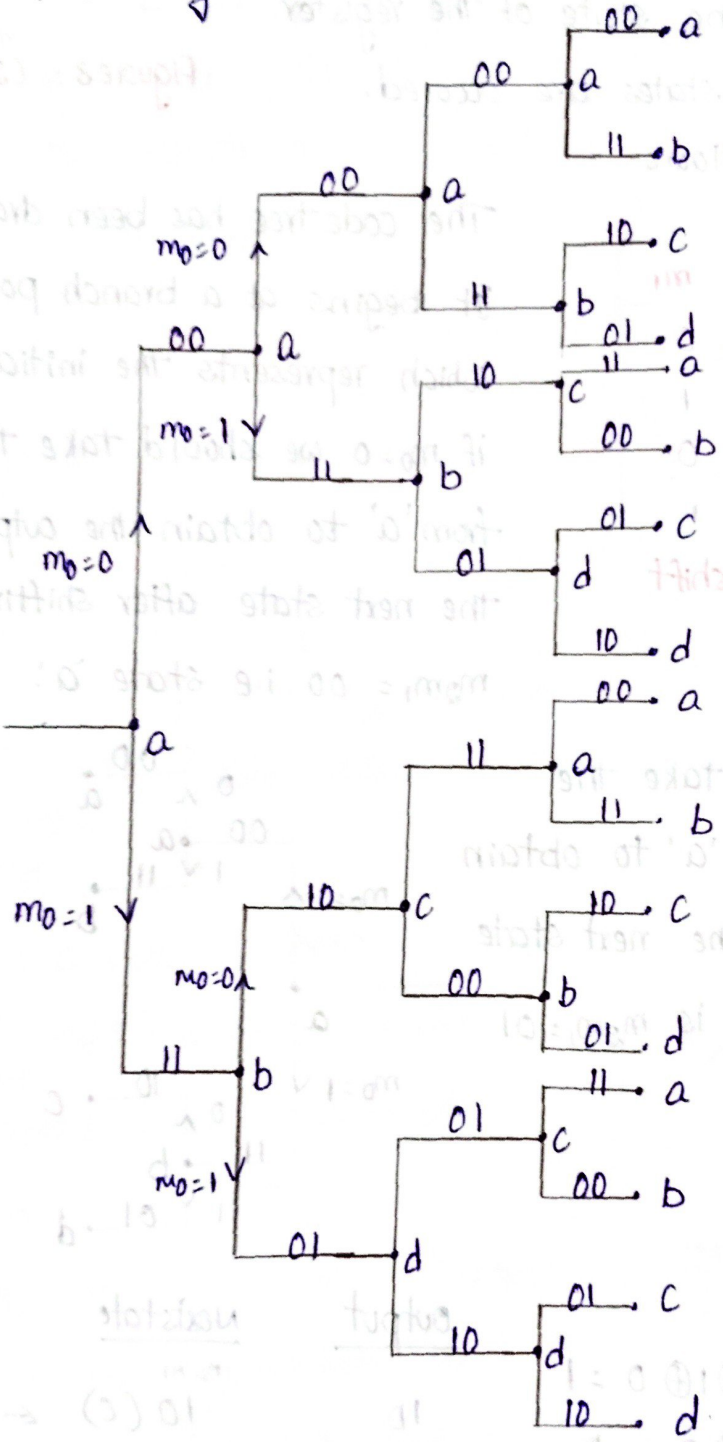


Fig 5 : code tree for (2,1,2) encoder

Code Trellis

Figure 6 shows a more compact graphical representation which is popularly known as code trellis. The nodes on the left denote the four possible current states and the nodes on the right are the resulting next state.

A solid line represents the state transition for $m_0 = 0$ and the dotted line represents the branch for $m_0 = 1$. Each branch is labelled with the resulting output bits $x_1 x_2$

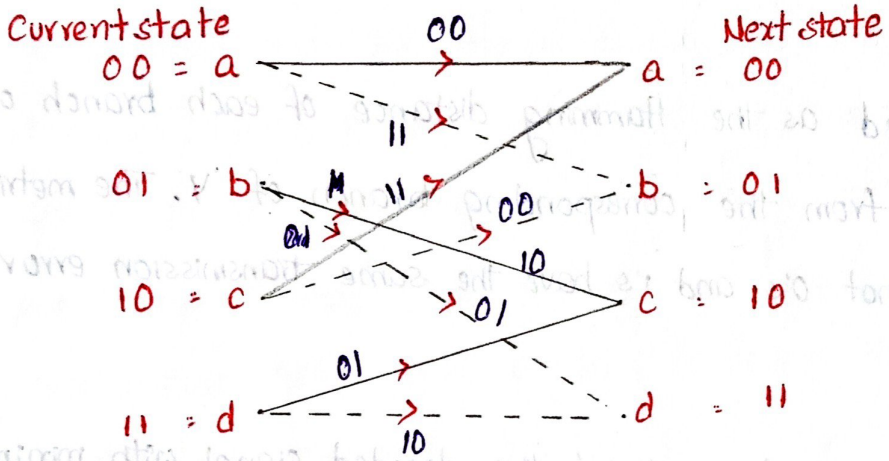


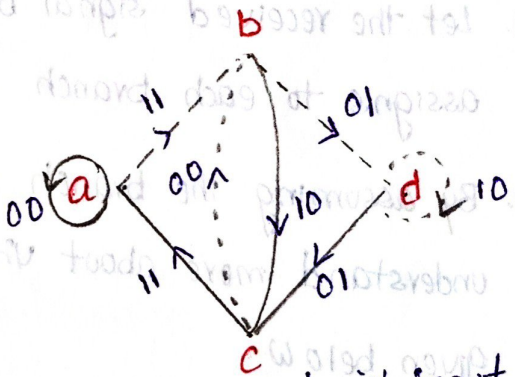
Fig 6 : Code trellis for the (2,1,2) convolutional encoder.

State diagram :

Fig 6 shows a state diagram for the encoder of fig 3 i.e (2,1,2) encoder.

We can obtain state diagram from the code trellis, by connecting the left and right sides of trellis. The self loop at the nodes 'a' and 'd' represent the state transition a-a and d-d.

Each branch is labelled with the resulting output bits $x_1 x_2$.



→ represents '0' input
-›- represents '1' input

Fig 7: State diagram for the (2,1,2) convolutional encoder

→ Decoding Method of convolutional codes (Viterbi Algorithm)

The Viterbi algorithm operates on the principle of "maximum likelihood decoding and achieves optimum performance".

The maximum likelihood decoder has to examine the entire received sequence 'Y' and find a valid path which has the smallest Hamming distance from 'Y'. But there are 2^N possible paths for a message sequence of N-bits.

Metric

It is defined as the Hamming distance of each branch of each surviving path from the corresponding branch of 'Y'. The metric is defined by assuming that 0's and 1's have the same transmission error probability.

Surviving Path.

It is defined as the path of the decoded signal with minimum metric.

1. Let the received signal be represented by 'Y'. The Viterbi decoder assigns to each branch of each surviving path of a metric.
2. By assuming the branch metrices we get the path metric. To understand more about Viterbi algorithm, let us solve the example given below:

→ Given the convolutional encoder of figure 3 and for a received signal Y = 11 01 11. Show that the first three branches of the valid paths emerging from the initial node a_0 in the code trellis.

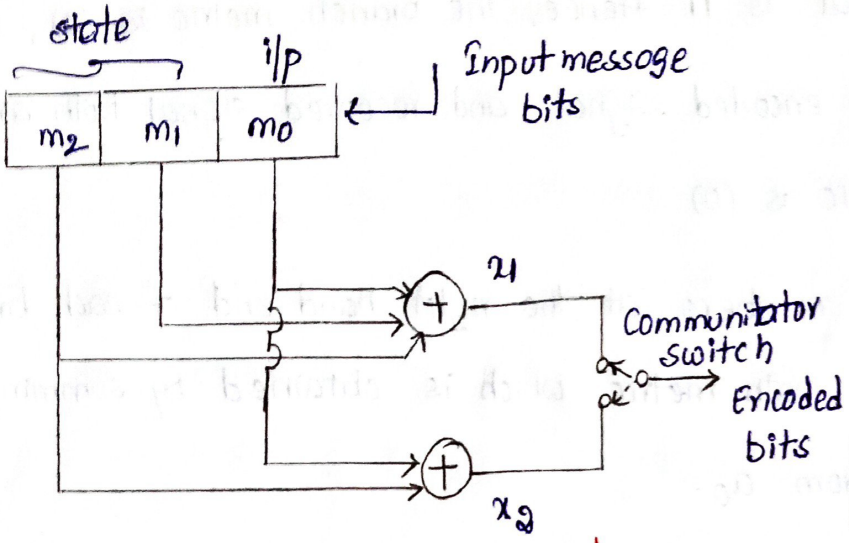


Fig 8: (2,1,2) convolutional encoder

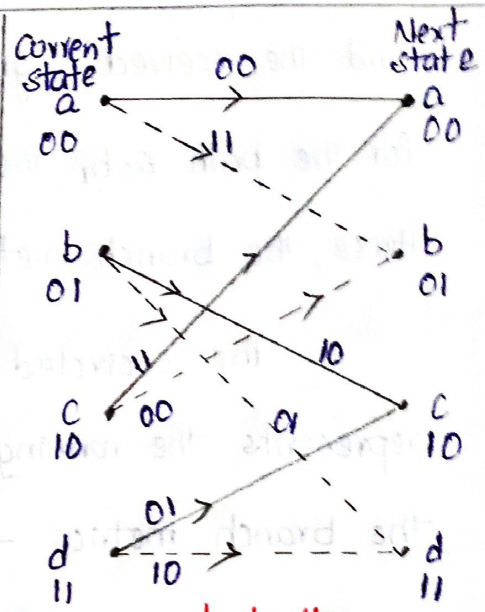


Fig 9: code trellis

Let us consider the code trellis diagram of fig 9 for the (2,1,2) encoder. It shows that for the current state 'a', the next state can be either 'a' or 'b' depending on the message bit $m_0 = 0$ or 1.

We have redrawn these two branches in Fig 10: where a_0 represents the initial state and a_1 & b_1 represent the next possible states. The solid line represents the branch for $m_0 = 0$ and the dotted line represent the branch for $m_0 = 1$

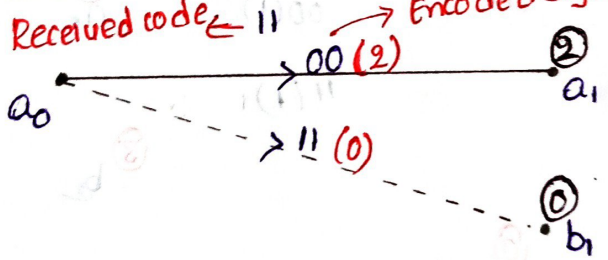


Fig 10: First step in Viterbi's algorithm

In Fig(10): The ~~written~~ below number in the brackets, written below the branches represent the branch metric which is obtained by taking the difference between the encoded signal and corresponding received signal 'Y'.

For example, for the branch a_0 to a_1, encoded signal is 00

and the received signal is 11. Hence, the branch metric is (2), whereas for the path a_0b_1 the encoded signal and received signal both are 11. Hence, the branch metric is (0).

The encircled numbers at the right hand end of each branch represents the running path metric which is obtained by summing the branch metrics from a_0.

For example, the running path metric for the branch $a_0a_1 = 2$ and that for the branch a_0b_1 is 0.

When the next part of input bits i.e Y = 01 are received at the nodes a_1 and b_1, then four possible branches emerge from these two nodes as shown in Fig11. The next four possible states are a_2, b_2, c_2 and d_2

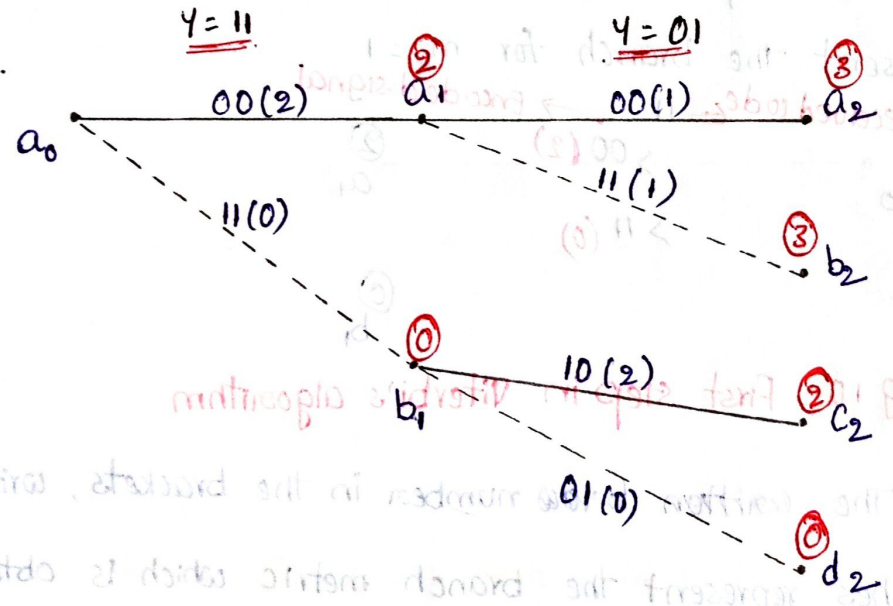


Fig11: Second step in viterbi algorithm.

The numbers in the brackets written below each branch represent the branch metric. For example, for the branch a_1a_2, the branch metric is (1) which is obtained by taking the difference between the encoded signall 00 and received signal 01. The running path metric for the same branch is (3) which is obtained by adding the branch metrics from a_0 [(2) + (1) = 3 from a_0 to a_1 and a_1 to a_2].

The viterbi's algorithm for all the input bits is shown in below figure:

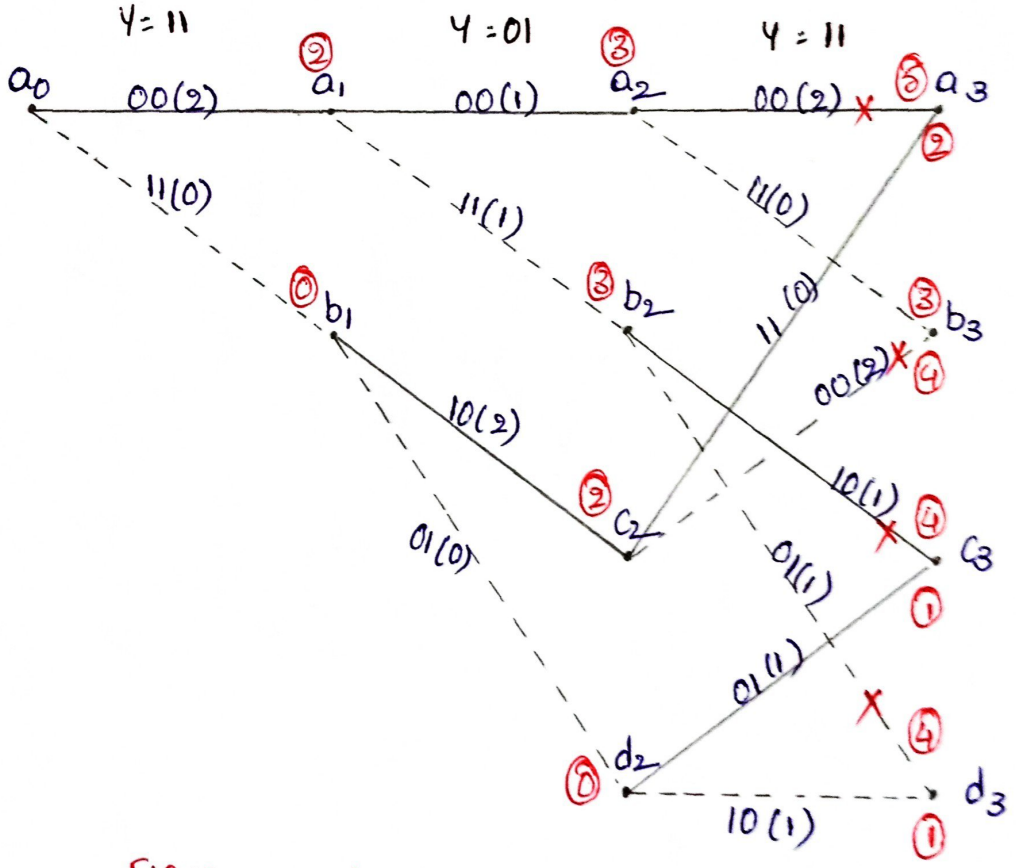


Fig 12 : Third step in Viterbi Algorithm

The large metric paths are discarded and the paths with smaller metric paths are declared as survivor at that node. The surviving paths are $a_0a_1a_2b_3$, $a_0b_1c_2a_3$, $a_0b_1d_2c_3$ and $a_0b_1d_2d_3$. The smaller metric path is $a_0b_1d_2c_3$ & $a_0b_1d_2d_3$.