UNIT –III ERROR CONTROL CODING

UNIT III ERROR CONTROL CODING

Linear block codes

Syndrome decoding

Minimum distance consideration

Cyclic codes

Generator polynomial

Parity check polynomial

Encoder for cyclic codes

Calculation of syndrome

Convolutional codes.

1 Introduction to linear block codes

Recall the example from the last time of a code: we had some generator matrix, some parity check matrix, and a means of doing some decoding. We will generalize these ideas.

A block code is a code in which k bits (or, more generally, symbols) are input and n bits (or, more generally symbols) are output. We designate the code as an (n, k) code. We will start with bits, elements from the field $GF(2)$; later we will consider elements from a field $GF(q)$ (after we know what this means).

If we input k bits, then there are 2^k distinct messages (or, more generally q^k). Each message of n symbols associated with a with each input block is called a *codeword*. We could, in general, simply have a lookup table with k inputs and n outputs. However, as k gets large, this quickly becomes infeasible. (Try $k = 255$, for example.) We therefore restrict our attention to linear codes.

Definition 1 A block code C of length n with 2^k codewords is called a **linear** (n, k) code if and only if its 2^k code words form a k-dimensional subspace of the vector space of all n-tuples over the field $GF(2)$.

More generally, with a bigger field, a block code C of length n with q^k is called a **linear** (n, k) code if and only if its q^k code words form a k-dimensional subspace of the vector space of all n-tuples over the field $GF(q)$. □

We remind ourselves of what a vector space is: we have an addition defined that is commutative and closed; we have scalar multiplication that is closed, distributive, and associative. We will formalize these properties a little further, but this suffices for the present purposes. We will see (later) that we have a *group structure* on the addition operation.

So what does this mean for codewords: the sum of any two codewords is a codeword. Being a linear vector space, there is some *basis*, and all codewords can be obtained as linear combinations of the basis. We can designate $\{\mathbf{g}_0, \mathbf{g}_1, \ldots, \mathbf{g}_{k-1}\}$ as the basis vectors. In a nutshell, it means that we can represent the coding operation as matrix multiplication, as we have already seen. We can formulate a *generator matrix* as

$$G = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix}$$

G is a $k \times n$ matrix. If $\mathbf{m} = (m_0, m_1, \ldots, m_{k-1})$ is an input sequence, then the output is the codeword

$$\mathbf{m}G = m_0\mathbf{g}_0 + m_1\mathbf{g}_1 + \cdots + \mathbf{m}_{k-1}\mathbf{g}_{k-1}.$$

We observe that the all-zero sequence must be a codeword. Therefore, *the minimum distance of the code C is the codeword of smallest weight.*

Comment on circuits to implement encoding.

We have a vector space of dimension k embedded in a vector space of dimension n, the set of all n-tuples. Associated with every linear block code generator G is a matrix H called the parity check matrix whose rows span the nullspace of G. Then if $\mathbf{c}$ is a codeword, then

$$\mathbf{c}H^T = 0.$$

That is, a codeword is orthogonal to each row of H. From this we observe that

$$GH^T = \mathbf{0}.$$

There is also associated with each code a **dual code** that has H as its generator matrix. The dual code is denoted as $C^{\perp}$. If G is the generator for an (n, k) code, then H is the generator for an $(n, n-k)$ code.

Example 1 A $(7, 4)$ Hamming code can be generated by

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

The 16 codewords are

0 0 0 0 0 0 0
0 0 0 1 1 1 1
0 0 1 0 1 1 0
0 0 1 1 0 0 1
0 1 0 0 1 0 1
0 1 0 1 0 1 0
0 1 1 0 0 1 1
0 1 1 1 1 0 0
1 0 0 0 0 1 1
1 0 0 1 1 0 0
1 0 1 0 1 0 1
1 0 1 1 0 1 0
1 1 0 0 1 1 0
1 1 0 1 0 0 1
1 1 1 0 0 0 0
1 1 1 1 1 1 1

The parity check matrix is

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

When regarded as a generator of an $(7, 3)$ code, the codewords of this code, the dual code has the codewords

0 0 0 0 0 0 0
1 1 0 1 0 0 1
1 0 1 1 0 1 0
0 1 1 0 0 1 1
0 1 1 1 1 0 0
1 0 1 0 1 0 1
1 1 0 0 1 1 0
0 0 0 1 1 1 1

It may be verified that every codeword in C is orthogonal to every codeword in $C^{\perp}$ □

When we want to do the encoding, it is often convenient to have the original data explicitly evident in the codeword. Coding of this sort is called *systematic encoding.* For the codes that we are to talk about, it will always be possible to determine a generator matrix in such a way the encoding is systematic: simply perform row reductions and column reordering on G until an identity matrix is revealed. We can thus write G as

$$G = [P|I_k]$$

where I_k is the $k \times k$ identity matrix and P is $k \times n-k$. . Then

$$\mathbf{c} = \mathbf{m}G = \mathbf{m}[P|I_k] = [c_0\ c_1\ \ldots\ c_{n-k-1}|m_0\ m_1 \ldots\ m_k]$$

When G is systematic, it is easy to determine the parity check matrix H. It is simply

$$H = [I_{n-k}| - P^T].$$

Note: in $GF(2)$ (binary operations) the negative of a number is simply the number. We could write (for binary codes)

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

The parity check matrix (whether systematic or not) can be used to get some useful information about the code.

Theorem 1 *Let a linear block code C have a parity check matrix H. The minimum distance of C is equal to the smallest positive number of columns of H which are linearly dependent.*

This concept should be distinguished from that of *rank*, which is the *largest* number of columns of H which are linearly independent.

Proof Let the columns of H be designated as $\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_{n-1}$. Then since $\mathbf{c}H^T = 0$ for any codeword $\mathbf{c}$, we have

$$0 = c_0\mathbf{d}_0 + c_1\mathbf{d}_1 + \cdots + c_{n-1}\mathbf{d}_{n-1}$$

Let $\mathbf{c}$ be the codeword of smallest weight, $w = w(\mathbf{c}) = d_{\min}$. Then the columns of H corresponding to the elements of $\mathbf{c}$ are linearly dependent. □

Based on this, we can determine a bound on the distance of a code:

$$d_{\min} \leq n - k + 1. \qquad \text{The Singleton bound}$$

This follows since H has $n-k$ linearly independent rows. (The row rank = the column rank.) So *any* combination of $n-k+1$ columns of H must be linearly dependent.

For a received vector $\mathbf{r}$, the **syndrome** is

$$\mathbf{s} = \mathbf{r}H^T.$$

Obviously, for a codeword the syndrome is equal to zero. We can determine if a received vector is a codeword. Furthermore, the syndrome is independent of the transmitted codeword. If $\mathbf{r} = \mathbf{c} + \mathbf{e}$,

$$\mathbf{s} = (\mathbf{c} + \mathbf{e})H^T = \mathbf{e}H^T.$$

Furthermore, if two error vectors $\mathbf{e}$ and $\mathbf{e}'$ have the same syndrome, then the error vectors must differ by a nonzero codeword. That is, if

$$\mathbf{e}H^T = \mathbf{e}'H^T$$

then

$$(\mathbf{e} - \mathbf{e}')H^T = 0$$

which means they must be a codeword.

2 Maximum likelihood detection

Before talking about decoding, we should introduce a probabilistic criterion for decoding, and show that it is equivalent to finding the closest codeword. Given a received vector $\mathbf{r}$, the decision rule that minimizes the probability of error is to find that codeword $\mathbf{c}_i$ which maximizes $P(\mathbf{c} = \mathbf{c}_i|\mathbf{r})$. This is called the *maximum a posteriori* decision rule. (Proof that this minimizes probability of error is shown in the communications class.) We note by Bayes rule that

$$P(\mathbf{c}|\mathbf{r}) = \frac{P(\mathbf{c})P(\mathbf{r}|\mathbf{c})}{P(\mathbf{r})},$$

where, for example, $P(\mathbf{r})$ is the probability of observing the vector $\mathbf{r}$. Now, since $P(\mathbf{r})$ is independent of $\mathbf{c}$, maximizing $P(\mathbf{c}|\mathbf{r})$ is equivalent to maximizing

$$P(\mathbf{c})P(\mathbf{r}|\mathbf{c}).$$

If we now assume that *each codeword is chosen with equal probability*, then maximizing $P(\mathbf{c})P(\mathbf{r}|\mathbf{c})$ is equivalent to maximizing

$$P(\mathbf{r}|\mathbf{c}).$$

A codeword which is selected on the basis of maximizing $P(\mathbf{r}|\mathbf{c})$ is said to be selected according to the *maximum likelihood* criterion. We shall assume throughout the text a maximum likelihood criterion.

Let us see what this means for us.

$$P(\mathbf{r}|\mathbf{c}) = \prod_{i=1}^{n} P(r_i|c_i)$$

Assuming a BSC channel with crossover probability p, we have

$$P(r_i|c_i) = \begin{cases} 1-p & \text{if} c_i = r_i \\ p & \text{if} c_i \neq r_i \end{cases}$$

Then

$$\begin{aligned} P(\mathbf{r}|\mathbf{c}) = \prod_{i=1}^{n} P(r_i|c_i) &= (1-p)^{\#(p_i = c_i)} p^{\#(p_i \neq c_i)} \\ &= (1-p)^{n - \#(p_i \neq c_i)} p^{\#(p_i \neq c_i)} = (1-p)^n \left(\frac{p}{1-p}\right)^{d(\mathbf{c},\mathbf{r})}. \end{aligned}$$

Then if we want to maximize $P(\mathbf{r}|\mathbf{c})$, we should choose that $\mathbf{c}$ which is **closest** to $\mathbf{r}$, since $0 \leq (p/(1-p)) \leq 1$. Thus, under our assumptions, the ML criterion is the minimum distance criterion. In every case, we should choose the error vector of lowest weight.

3 The standard array and syndrome table decoding

Suppose we send $\mathbf{c}$ and we receive

$$\mathbf{r} = \mathbf{c} + \mathbf{e}.$$

Assuming that error sequences with lower weight are more probable than error sequences with higher weight (the maximum likelihood criterion), we want to determine our decoded word $\mathbf{c}'$ such that the error sequence $\mathbf{e}'$ satisfying

$$\mathbf{r} = \mathbf{c}' + \mathbf{e}'$$

has minimum weight.

One way to do this is to create a *standard array*. We form it the following way:

1. Write down a list of all possible codewords in a row, with the all-zero codeword first.
2. From the remaining n-tuples which have not already been used in the standard array, select one of smallest weight. Write this down as the *coset leader* under the all-zero codeword. On this row, add the coset leader to each codeword at the top of the column.
3. Repeat step 2 until all the n-tuples have been listed.

An example standard array for a $(7, 3)$ code is shown here, where

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

1. There are 2^k codewords (columns) and 2^n possible vectors, so there are 2^{n-k} rows in the standard array.

2. The sum of any two vectors in the same row of the standard array is a code vector.

3. No two vectors in the same row of a standard array are identical. Because otherwise we have

$$\mathbf{e}_i + \mathbf{c}_i = \mathbf{e}_i + \mathbf{c}_j, \text{ with } i \neq j$$

which means $\mathbf{c}_i = \mathbf{c}_j$, which is impossible.

4. Every vector appears exactly once in the standard array. We know every vector must appear at least once, by the construction. If a vector appears in both the lth row and the mth row we must have

$$\mathbf{e}_l + \mathbf{c}_i = \mathbf{e}_m + \mathbf{c}_j$$

for some i and j. Let us take $l < m$. We have

$$\mathbf{e}_m = \mathbf{e}_l + \mathbf{c}_i - \mathbf{c}_j = \mathbf{e}_l + \mathbf{c}_k$$

for some k. This means that $\mathbf{e}_m$ is on the lth row of the array, which is a contradiction.

Each row of the standard array is called a *coset*; we will encounter the term coset in a more formal setting soon.

To decode with the standard array, we locate the received vector $\mathbf{r}$ in the standard array. Then the error sequence is the coset leader; the best guess of the transmitted word is the codeword at the top of the column. For example, if

$$\mathbf{r} = 0011011$$

then

$$\mathbf{c}' = 0011101.$$

Since we designed the standard array with the smallest error patterns as coset leaders, this is the ML decision.

As observed before, there are 2^{n-k} coset leaders. These are called the *correctable error patterns*. Fact: an (n, k) code is capable of correcting 2^{n-k} error patterns.

The standard array can be used to decode linear codes, but suffers from a major problem: the memory requirements quickly become excessive. We want to look for easier approaches.

A first step (which doesn't go far enough), is to use the syndrome in the decoding. Based on the properties of the syndrome above, all elements in a row of the standard array have the same syndrome. We therefore only need to store syndromes and their associated error patterns.

For the code whose standard array was given before, we have

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

GENERATOR AND PARITY CHECK MATRICE 0F CYCLIC CODES

To construct the 4 by 7 generator generator matrix **G** , we start with four polynomials represented by **g**(X) and three cyclic shifted versions of it as shown by:-

$$\mathbf{g}(X) = 1 + X + X^3 \quad \text{(zero shift)}$$

$X \bullet \mathbf{g}(X) = X + X^2 + X^4$ (1 – cyclic shift).
$X^2 \bullet \mathbf{g}(X) = X^2 + X^3 + X^5$ (2 – cyclic shift).
$X^3 \bullet \mathbf{g}(X) = X^3 + X^4 + X^6$ (3 – cyclic shift).

If the coefficients of these polynomials are used as elements of the rows of a 4 by 7 matrix, we got:-

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

TABLE 4.2 A (7, 4) CYCLIC CODE GENERATED BY $g(X) = 1 + X + X^3$

Message	Code word	
(0 0 0 0)	(0 0 0 0 0 0 0)	$0 = 0 \cdot g(X)$
(1 0 0 0)	(1 1 0 1 0 0 0)	$1 + X + X^3 = g(X)$
(0 1 0 0)	(0 1 1 0 1 0 0)	$X + X^2 + X^4 = Xg(X)$
(1 1 0 0)	(1 0 1 1 1 0 0)	$1 + X^2 + X^3 + X^4 = (1 + X)g(X)$
(0 0 1 0)	(1 1 1 0 0 1 0)	$1 + X + X^2 + X^5 = (1 + X^2)g(X)$
(1 0 1 0)	(0 0 1 1 0 1 0)	$X^2 + X^3 + X^5 = X^2g(X)$
(0 1 1 0)	(1 0 0 0 1 1 0)	$1 + X^4 + X^5 = (1 + X + X^2)g(X)$
(1 1 1 0)	(0 1 0 1 1 1 0)	$X + X^3 + X^4 + X^5 = (X + X^2)g(X)$
(0 0 0 1)	(1 0 1 0 0 0 1)	$1 + X^2 + X^6 = (1 + X + X^3)g(X)$
(1 0 0 1)	(0 1 1 1 0 0 1)	$X + X^2 + X^3 + X^6 = (X + X^3)g(X)$
(0 1 0 1)	(1 1 0 0 1 0 1)	$1 + X + X^4 + X^6 = (1 + X^3)g(X)$
(1 1 0 1)	(0 0 0 1 1 0 1)	$X^3 + X^4 + X^6 = X^3g(X)$
(0 0 1 1)	(0 1 0 0 0 1 1)	$X + X^5 + X^6 = (X + X^2 + X^3)g(X)$
(1 0 1 1)	(1 0 0 1 0 1 1)	$1 + X^3 + X^5 + X^6 = (1 + X + X^2 + X^3)g(X)$
(0 1 1 1)	(0 0 1 0 1 1 1)	$X^2 + X^4 + X^5 + X^6 = (X^2 + X^3)g(X)$
(1 1 1 1)	(1 1 1 1 1 1 1)	$1 + X + X^2 + X^3 + X^4 + X^5 + X^6 = (1 + X^2 + X^5)g(X)$

If the first row is added to the third row and the sum of the first two rows is added to the fourth row, we obtain the following matrix:

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

Which is in systematic form. This matrix generates the same code as **G**.

'The generator matrix in systematic form can also be formed easily. Dividing X^{n-k+i} by the generator polynomial $\mathbf{g}(X)$ for $i = 0, 1, \ldots, k-1$, we obtain

$$X^{n-k+i} = \mathbf{a}_i(X)\mathbf{g}(X) + \mathbf{b}_i(X),$$

where $\mathbf{b}_i(X)$ is the remainder with the following form:

$$\mathbf{b}_i(X) = b_{i0} + b_{i1}X + \cdots + b_{i,n-k-1}X^{n-k-1}.$$

Since $\mathbf{b}_i(X) + X^{n-k+i}$ for $i = 0, 1, \ldots, k-1$ are multiples of $\mathbf{g}(X)$, they are code polynomials. Arranging these k code polynomials as rows of a $k \times n$ matrix, we obtain

$$\mathbf{G} = \begin{bmatrix} b_{00} & b_{01} & b_{02} & \cdots & b_{0,n-k-1} & 1 & 0 & 0 & \cdots & 0 \\ b_{10} & b_{11} & b_{12} & \cdots & b_{1,n-k-1} & 0 & 1 & 0 & \cdots & 0 \\ b_{20} & b_{21} & b_{22} & \cdots & b_{2,n-k-1} & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots & & \vdots \\ b_{k-1,0} & b_{k-1,1} & b_{k-1,2} & \cdots & b_{k-1,n-k-1} & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}, \quad \text{....(5)}$$

which is the generator matrix of C in systematic form. The corresponding parity-check matrix for C is

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & b_{00} & b_{10} & b_{20} & \cdots & b_{k-1,0} \\ 0 & 1 & 0 & \cdots & 0 & b_{01} & b_{11} & b_{21} & \cdots & b_{k-1,1} \\ 0 & 0 & 1 & \cdots & 0 & b_{02} & b_{12} & b_{22} & \cdots & b_{k-1,2} \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 & b_{0,n-k-1} & b_{1,n-k-1} & b_{2,n-k-1} & \cdots & b_{k-1,n-k-1} \end{bmatrix}. \quad \text{...(6)}$$

Example 3.

Again, let us consider the (7, 4) cyclic code generated by $g(X) = 1 + X + X^3$. Dividing X^3, X^4, X^5, and X^6 by $g(X)$, we have

$$X^3 = g(X) + (1 + X),$$

$$X^4 = Xg(X) + (X + X^2),$$

$$X^5 = (X^2 + 1)g(X) + (1 + X + X^2),$$

$$X^6 = (X^3 + X + 1)g(X) + (1 + X^2).$$

Rearranging the equations above, we obtain the following four code polynomials:

$$v_0(X) = 1 + X + X^3,$$

$$v_1(X) = X + X^2 + X^4,$$

$$v_2(X) = 1 + X + X^2 + X^5,$$

$$v_3(X) = 1 + X^2 + X^6.$$

Taking these four code polynomials as rows of a 4×7 matrix, we obtain the following generator matrix in systematic form for the (7, 4) cyclic code:

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

which is identical to the matrix G' obtain earlier in this section.

EXAMPLE 4 : Construct Parity Check Matrix **H** of example 2?
We simply find the party polynomial **H**(X) as follow:

$$\mathbf{h}(X) = \frac{X^7 + 1}{\mathbf{g}(X)}$$

$$= 1 + X + X^2 + X^4.$$

The reciprocal of $\mathbf{h}(X)$ is

$$X^4\mathbf{h}(X^{-1}) = X^4(1 + X^{-1} + X^{-2} + X^{-4}).$$

$$= 1 + X^2 + X^3 + X^4.$$

Also $X^5 \bullet \mathbf{h}(X^{-1}) = X + X^3 + X^4 + X^5$,
And $X^6 \bullet \mathbf{h}(X^{-1}) = X^2 + X^4 + X^5 + X^6$.

Then using the coefficients of these three equations as the elements of the rows of the 3 by 7 parity check matrix, we got

$$\mathbf{H}' = \begin{matrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{matrix}$$

Here $\mathbf{H}'$ is not in systematic form therefore we must put it into a systematic by add 3rd row with the 1st row to obtain :-

$$\mathbf{H} = \begin{matrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{matrix}$$

Convolution Codes

Convolution codes were first introduced by Elias [1] in 1955 as an alternative to block codes.

Convolution codes differ from block codes in that the encoder contains memory and the n encoder outputs at any given time unit depend not only on the k inputs at that time unit but also on m previous input blocks. An (n, k, m) convolutional code can be implemented with a k-input, n-output linear sequential circuit with input memory m. Typically, n and k are small integers with $k < n$, but the memory order m must be made large to achieve low error probabilities.

The encoder of convolution code for a binary (2, l, 3) code is shown in Figure (1). Note that the encoder consists of an m = 3-stage shift register together with n = 2 modulo-2 adders and a multiplexer for serializing the encoder outputs. The mod-2 adders can be implemented as EXCLUSIVE-OR gates. Since mod-2 addition is a linear operation, the encoder is a linear feedforward shift register All convolutional encoders can be implemented using a linear feedforward shift register of this type.

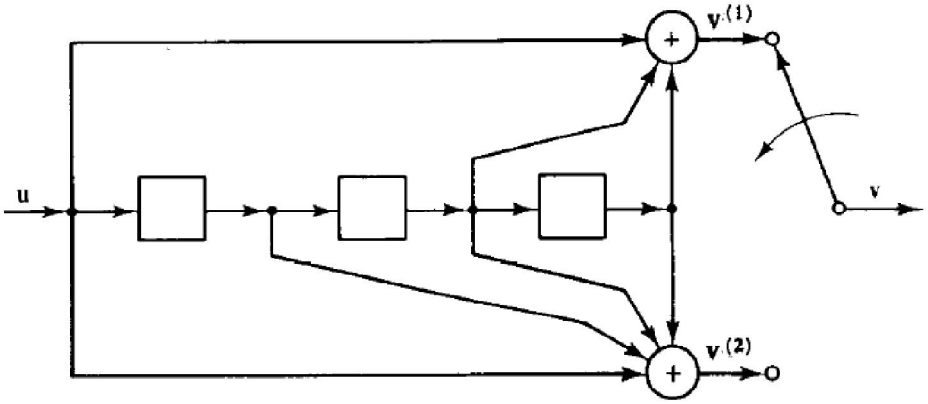

Fig. 1

The *information sequence* $\mathbf{u} = (u_0, u_1, u_2, \ldots)$ enters the encoder one bit at a time. Since the encoder is a linear system, the two encoder *output sequences* $\mathbf{v}^{(1)} = (v_0^{(1)}, v_1^{(1)}, v_2^{(1)}, \ldots)$ and $\mathbf{v}^{(2)} = (v_0^{(2)}, v_1^{(2)}, v_2^{(2)}, \ldots)$ can be obtained as the convolution of the input sequence $\mathbf{u}$ with the two encoder "impulse responses." The impulse responses are obtained by letting $\mathbf{u} = (1\ 0\ 0 \bullet \bullet \bullet)$ and observing the two output sequences. Since the encoder has an w-time unit memory, the impulse responses can last at most $m + 1$ time units, and are written $\mathbf{g}^{(1)} = (g_0^{(1)}, g_1^{(1)}, g_2^{(1)}, \ldots, g_m^{(1)})$ and $\mathbf{g}^{(2)} = (g_0^{(2)}, g_1^{(2)}, g_2^{(2)}, \ldots, g_m^{(2)})$ For the encoder of Figure 1

$$\mathbf{g}^{(1)} = (1 \quad 0 \quad 1 \quad 1)$$

$$\mathbf{g}^{(2)} = (1 \quad 1 \quad 1 \quad 1).$$

The impulse responses $\mathbf{g}^{(1)}$ and $\mathbf{g}^{(2)}$ are called the *generator sequences* of the code. The *encoding equations* can now be written as

$$\begin{aligned} \mathbf{v}^{(1)} &= \mathbf{u} * \mathbf{g}^{(1)} \\ \mathbf{v}^{(2)} &= \mathbf{u} * \mathbf{g}^{(2)}, \end{aligned} \qquad \ldots\ldots\ldots\ldots\ldots(1)$$

Where * denotes discrete convolution and all operations are modulo-2. The convolution operation implies that for all $l \geq 0$,

If the generator sequences $\mathbf{g}^{(1)}$ and $\mathbf{g}^{(2)}$ are interlaced and then arranged in the matrix

$$\mathbf{G} = \begin{bmatrix} g_0^{(1)}g_0^{(2)} & g_1^{(1)}g_1^{(2)} & g_2^{(1)}g_2^{(2)} & \cdots & g_m^{(1)}g_m^{(2)} & & \\ & g_0^{(1)}g_0^{(2)} & g_1^{(1)}g_1^{(2)} & \cdots & g_{m-1}^{(1)}g_{m-1}^{(2)} & g_m^{(1)}g_m^{(2)} & \\ & & g_0^{(1)}g_0^{(2)} & \cdots & g_{m-2}^{(1)}g_{m-2}^{(2)} & g_{m-1}^{(1)}g_{m-1}^{(2)} & g_m^{(1)}g_m^{(2)} \\ & & \cdot & & \cdot & \cdot & \cdot \\ & & \cdot & & \cdot & \cdot & \cdot \\ & & \cdot & & \cdot & \cdot & \cdot \end{bmatrix},$$

....(3)

Where the blank areas are all zeros, the encoding equations can be rewritten in matrix form as

$$\mathbf{v} = \mathbf{uG},$$

.....(4)

Where all operations are modulo-2. **G** is called the *generator matrix* of the code. Note that each row of **G** is identical to the preceding row but shifted $n = 2$ places to the right, and that **G** is a semi-infinite matrix, corresponding to the fact that the information sequence **u** is of arbitrary length. If **u** has finite length L, then **G** has L rows and $2(m + L)$ columns, and **v** has length $2(m + L)$.

Example 2: If **u** (10 111), then

$$\mathbf{v} = \mathbf{uG}$$

$$= (1 \quad 0 \quad 1 \quad 1 \quad 1) \begin{bmatrix} 1\,1 & 0\,1 & 1\,1 & 1\,1 & & & & \\ & 1\,1 & 0\,1 & 1\,1 & 1\,1 & & & \\ & & 1\,1 & 0\,1 & 1\,1 & 1\,1 & & \\ & & & 1\,1 & 0\,1 & 1\,1 & 1\,1 & \\ & & & & 1\,1 & 0\,1 & 1\,1 & 1\,1 \end{bmatrix}$$

$$= (1 \quad 1, \quad 0 \quad 1, \quad 0 \quad 0, \quad 0 \quad 1, \quad 0 \quad 1, \quad 0 \quad 1, \quad 0 \quad 0, \quad 1 \quad 1), \checkmark$$

As a second example of a convolutional encoder, consider the (3, 2, 1) code shown in Figure 2.

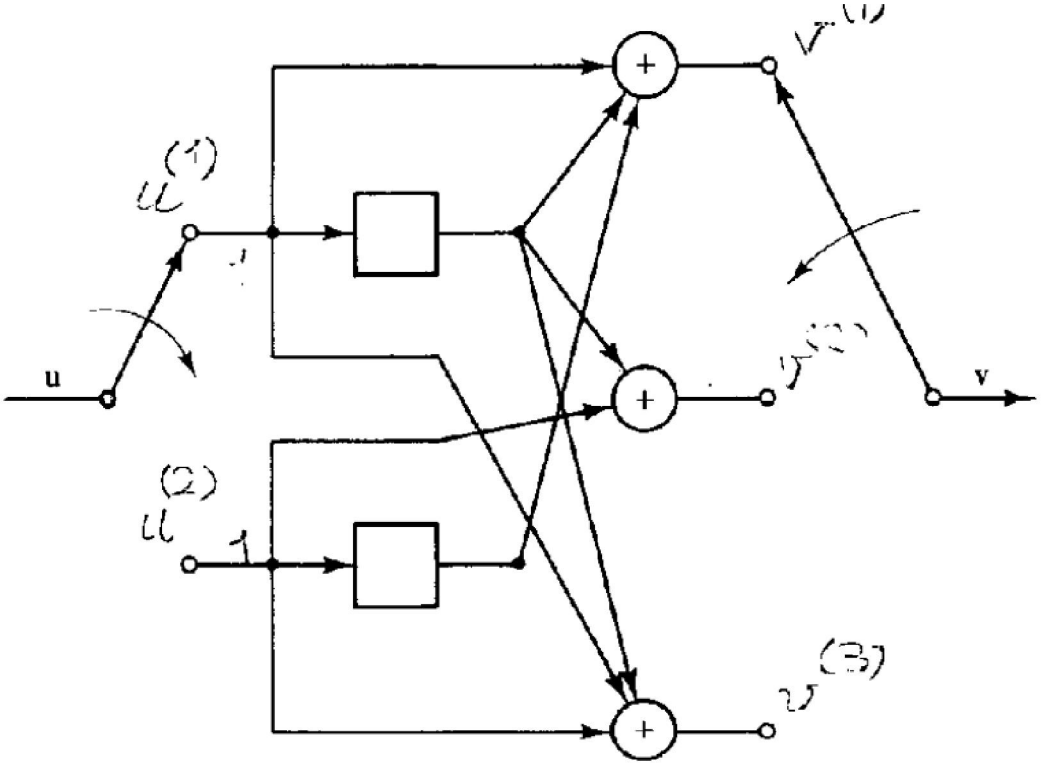

Fig .2

Since $k = 2$, the encoder consists of two $m = 1$-stage shift registers together with $n = 3$ mod-2 adders and two

As can be seen from the encoding circuit. After multiplexing, the code word is given by

$$\mathbf{v} = (v_0^{(1)}v_0^{(2)}v_0^{(3)}, v_1^{(1)}v_1^{(2)}v_1^{(3)}, v_2^{(1)}v_2^{(2)}v_2^{(3)}, \ldots).$$

EXAMPLE 3:

If $\mathbf{u}^{(1)} = (1\ 0\ 1)$ and $\mathbf{u}^{(2)} = (1\ 1\ 0)$, then

$$\mathbf{v}^{(1)} = (1\ 0\ 1) * (1\ 1) + (1\ 1\ 0) * (0\ 1) = (1\ 0\ 0\ 1)$$

$$\mathbf{v}^{(2)} = (1\ 0\ 1) * (0\ 1) + (1\ 1\ 0) * (1\ 0) = (1\ 0\ 0\ 1)$$

$$\mathbf{v}^{(3)} = (1\ 0\ 1) * (1\ 1) + (1\ 1\ 0) * (1\ 0) = (0\ 0\ 1\ 1)$$

and

$$\mathbf{v} = (1\ 1\ 0,\ 0\ 0\ 0,\ 0\ 0\ 1,\ 1\ 1\ 1).$$

The generator matrix of a $(3, 2, m)$ code is

$$\mathbf{G} = \begin{bmatrix} g_{1,0}^{(1)}g_{1,0}^{(2)}g_{1,0}^{(3)} & g_{1,1}^{(1)}g_{1,1}^{(2)}g_{1,1}^{(3)} & \cdots & g_{1,m}^{(1)} & g_{1,m}^{(2)} & g_{1,m}^{(3)} & & \\ g_{2,0}^{(1)}g_{2,0}^{(2)}g_{2,0}^{(3)} & g_{2,1}^{(1)}g_{2,1}^{(2)}g_{2,1}^{(3)} & \cdots & g_{2,m}^{(1)} & g_{2,m}^{(2)} & g_{2,m}^{(3)} & & \\ & g_{1,0}^{(1)}g_{1,0}^{(2)}g_{1,0}^{(3)} & \cdots & g_{1,m-1}^{(1)}g_{1,m-1}^{(2)}g_{1,m-1}^{(3)} & & g_{1,m}^{(1)}g_{1,m}^{(2)}g_{1,m}^{(3)} & & \\ & g_{2,0}^{(1)}g_{2,0}^{(2)}g_{2,0}^{(3)} & \cdots & g_{2,m-1}^{(1)}g_{2,m-1}^{(2)}g_{2,m-1}^{(3)} & & g_{2,m}^{(1)}g_{2,m}^{(2)}g_{2,m}^{(3)} & & \\ & \ddots & & & & \ddots & & \\ & & \ddots & & & & \ddots & \\ & & & \ddots & & & & \ddots \end{bmatrix}$$

.....(6)

And the encoding equations in matrix form are again given by **v = uG.** Note that each set of $k = 2$ rows of **G** is identical to the preceding set of rows but shifted $n = 3$ places to the right.

Hence, the output polynomial of path 1 is given by

$$\begin{aligned} c^{(1)}(D) &= g^{(1)}(D)m(D) \\ &= (1 + D + D^2)(1 + D^3 + D^4) \\ &= 1 + D + D^2 + D^3 + D^6 \end{aligned}$$

From this we immediately deduce that the output sequence of path 1 is (1111001). Similarly, the output polynomial of path 2 is given by

$$\begin{aligned} c^{(2)}(D) &= g^{(2)}(D)m(D) \\ &= (1 + D^2)(1 + D^3 + D^4) \\ &= 1 + D^2 + D^3 + D^4 + D^5 + D^6 \end{aligned}$$

The output sequence of path 2 is therefore (1011111). Finally, multiplexing the two output sequences of paths 1 and 2, we get the encoded sequence

$$\mathbf{c} = (11, 10, 11, 11, 01, 01, 11)$$

Note that the message sequence of length L = 5 bits produces an encoded sequence of length $n(L + K - 1) = 14$ bits. Where L represents number of bits in the input message , and M-shift register requires k=M+1 shifts for a message bit to enter the shift register and finally come out. In this example M=2 ,k=3, and n=2.

Hence, the output polynomial of path 1 is given by

$$\begin{aligned} c^{(1)}(D) &= g^{(1)}(D)m(D) \\ &= (1 + D + D^2)(1 + D^3 + D^4) \\ &= 1 + D + D^2 + D^3 + D^6 \end{aligned}$$

From this we immediately deduce that the output sequence of path 1 is (1111001). Similarly, the output polynomial of path 2 is given by

$$\begin{aligned} c^{(2)}(D) &= g^{(2)}(D)m(D) \\ &= (1 + D^2)(1 + D^3 + D^4) \\ &= 1 + D^2 + D^3 + D^4 + D^5 + D^6 \end{aligned}$$

The output sequence of path 2 is therefore (1011111). Finally, multiplexing the two output sequences of paths 1 and 2, we get the encoded sequence

$$\mathbf{c} = (11, 10, 11, 11, 01, 01, 11)$$

Note that the message sequence of length L = 5 bits produces an encoded sequence of length $n(L + K - 1) = 14$ bits. Where L represents number of bits in the input message , and M-shift register requires k=M+1 shifts for a message bit to enter the shift register and finally come out. In this example M=2 ,k=3, and n=2.

EXAMPLE 6: For the (2, 1, 3) code of Figure 1, the generator polynomials are $\mathbf{g^{(1)}(D)} = 1 + D^2 + D^3$ and $\mathbf{g^{(2)}(D)} = 1 + D + D^2 + D^3$. For the information sequence $\mathbf{u(D)} = 1 + D^2 + D^3 + D^4$, the encoding equations are

$$\mathbf{v}^{(1)}(D) = (1 + D^2 + D^3 + D^4)(1 + D^2 + D^3) = 1 + D^7$$

$$\begin{aligned}\mathbf{v}^{(2)}(D) &= (1 + D^2 + D^3 + D^4)(1 + D + D^2 + D^3) \\ &= 1 + D + D^3 + D^4 + D^5 + D^7,\end{aligned}$$

And the code word is

$$\mathbf{v}(D) = 1 + D + D^3 + D^7 + D^9 + D^{11} + D^{14} + D^{15}.$$

UNIT III
ERROR CONTROL CODING
PART-A(2marks)

1. What is the use of error control coding?
2. What is the difference between systematic code and non-systematic code?
3. What is a Repetition code?
4. What is forward acting error correction method?
5. What is error detection?
6. Define linear block code.
7. Give the properties of syndrome in linear block code.
8. What is Hamming code?
9. When a code is said to be cyclic?
10. Give the difference between linear block code and cyclic code.
11. What is generator polynomial?
12. What is parity check polynomial?
13. How will you convert a generator polynomial into a generator matrix?
14. How will you convert parity check polynomial into a parity check matrix
15. How a syndrome polynomial can be calculated?
16. Give two properties of syndrome in cyclic code.
17. Define Hamming distance (HD).
18. Define Weight of a code vector.
19. Define minimum distance?
20. What is coset leader?
21. What is convolutional code?
22. Define constraint length.
23. What is meant by tail of a message?
24. What is state diagram?
25. What is trellis diagram?

PART-B

1. Explain Linear Block Code. (16)
2. Explain cyclic code in detail. (16)
3. Explain Convolution codes. (16)
4. Write the procedures for designing an Encoder circuit. (16)
5. Write the procedures for designing a syndrome calculator circuit. (16)