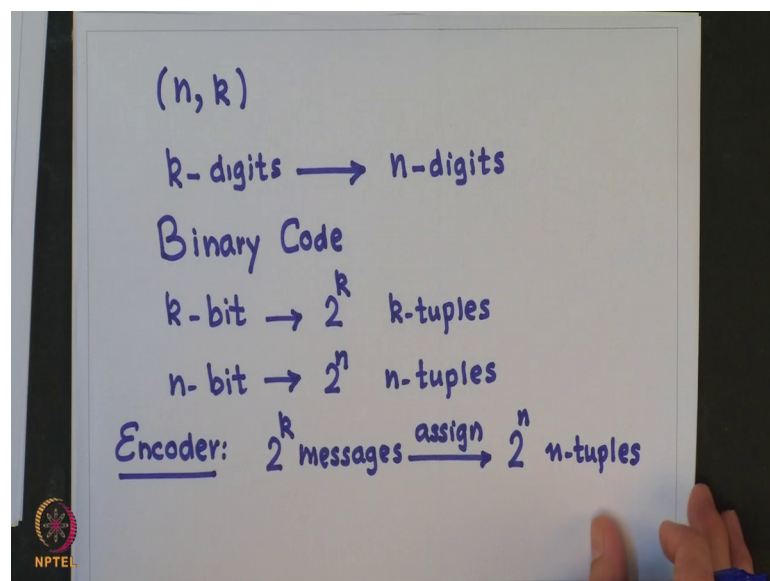


Principles of Digital Communications
Prof. Shabbir N. Merchant
Department of Electrical Engineering
Indian Institute of Technology, Bombay

Lecture - 61
Channel Coding - II

In the earlier class, we have introduced Channel Coding and also learnt what is a block code. So, a block code is defined as follows.

(Refer Slide Time: 00:38)



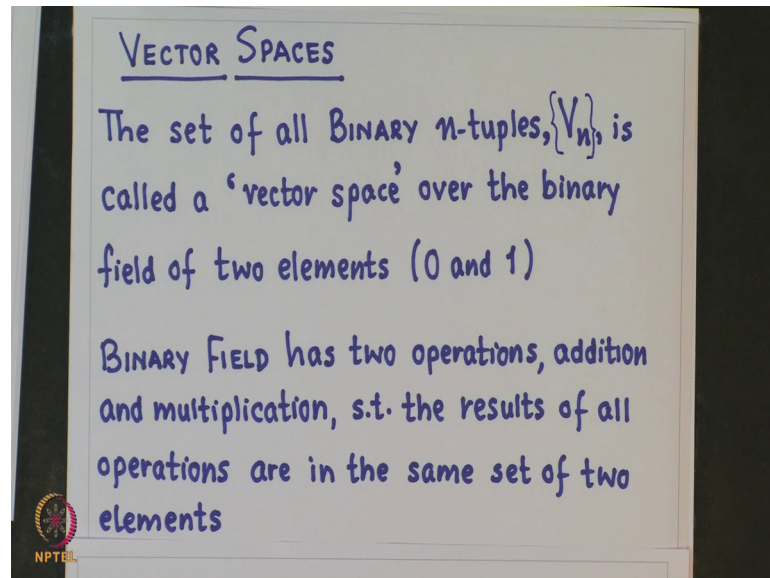
The notation uses n comma k , a block of k message digits is mapped to a larger block of n codeword digits and the digits are chosen from a given alphabets of elements.

Now, for our study, we will restrict to binary code. So, in that case we will have k bit messages, which form 2 raised to k message sequence also known as k -tuples and we will have n bit blocks, which will form 2 raised to n distinct sequences referred to as n -tuples. So, the job of the encoder is to take each of this 2 raised to k messages and assign 1 of the 2 raised to n , n -tuples.

So, here we will have message sequences which are k bit long and here we will have the codewords, which are n bits long. So, what we get is that, 2 raised to k messages k -tuples uniquely get mapped to 2 raised to n codeword n -tuples. Now before we proceed ahead

we need some concepts from vector spaces and subspaces. So, let us define the vector space.

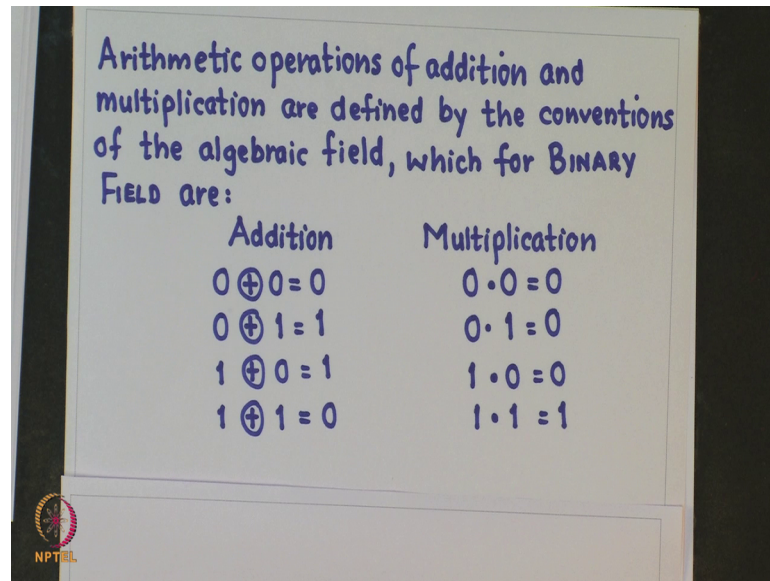
(Refer Slide Time: 03:21)



The set of all binary n-tuples which is denoted by this curly bracket and this n denotes that we are talking about n-tuples is called a vector space over the binary field of 2 elements 0 and 1.

So, binary field has 2 operations addition and multiplication such that, the results of all operations are in the same set of 2 elements that is 0 and 1. So, arithmetic operations of addition and multiplications are defined by the conventions of the algebraic field and which for binary field are as follows.

(Refer Slide Time: 04:17)



Arithmetic operations of addition and multiplication are defined by the conventions of the algebraic field, which for BINARY FIELD are:

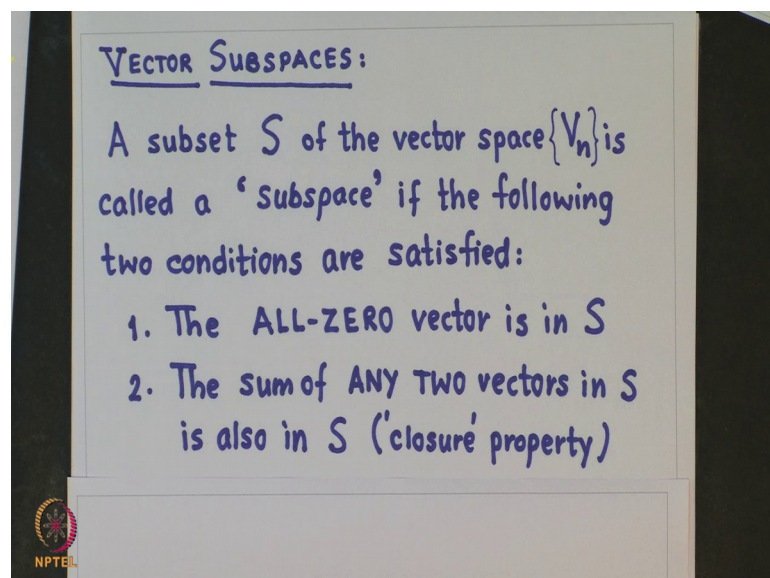
| Addition | Multiplication |
|------------------|-----------------|
| $0 \oplus 0 = 0$ | $0 \cdot 0 = 0$ |
| $0 \oplus 1 = 1$ | $0 \cdot 1 = 0$ |
| $1 \oplus 0 = 1$ | $1 \cdot 0 = 0$ |
| $1 \oplus 1 = 0$ | $1 \cdot 1 = 1$ |

NPTEL

Addition is given by this rule, multiplication is given by this rule. So, we see that this is an exclusive or operation and this is an end operation in Boolean terms.

So, now let us define a subspace of a vector space. A subset S of the vector space is called a subspace if the following 2 conditions are satisfied.

(Refer Slide Time: 04:57)



VECTOR SUBSPACES:

A subset S of the vector space $\{V_n\}$ is called a 'subspace' if the following two conditions are satisfied:

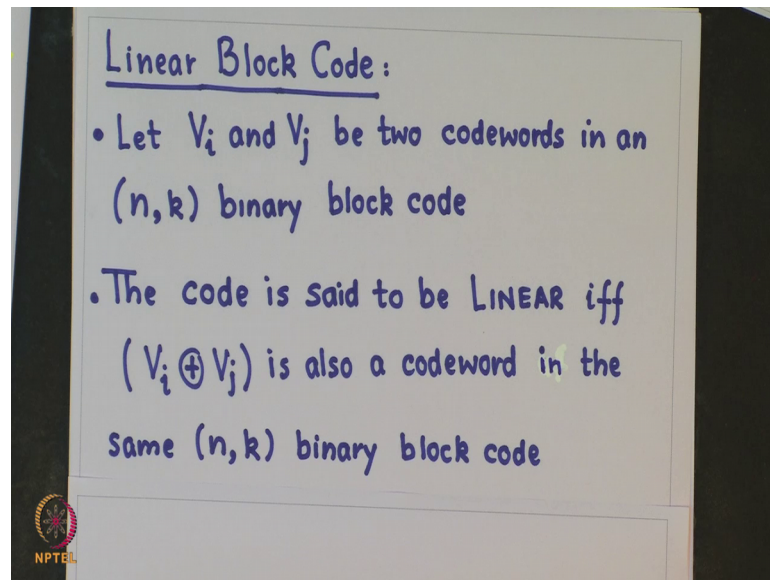
1. The ALL-ZERO vector is in S
2. The sum of ANY TWO vectors in S is also in S ('closure' property)

NPTEL

The all zero vector is in S and the sum of any two vectors in S is also in S this is known as a closure property. Now we see, why we require all 0 vector because if this condition has to be satisfied, then I can choose 2 vectors to be the same vectors. So, when you take

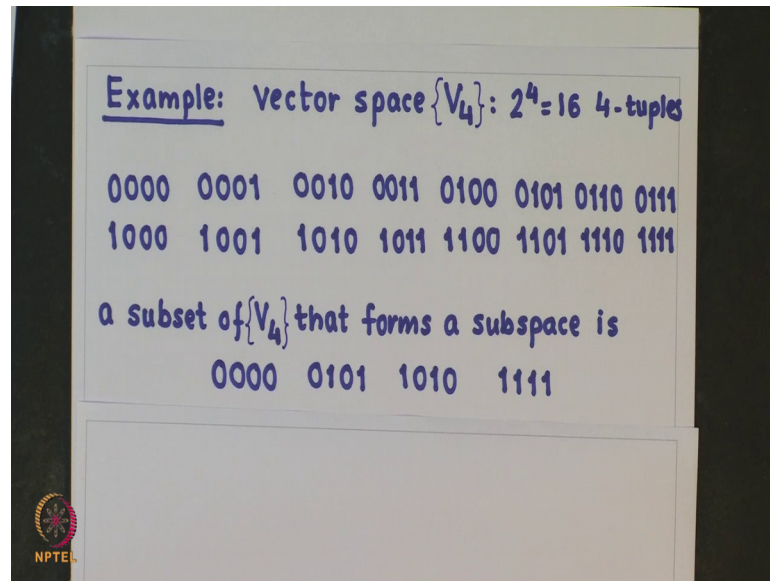
the same vectors and if you add them using the rules of addition, just discussed we will find we will get a 0 vector. And by definition we require that the summation of this 2 vectors should be in S and therefore, we see the need for the all 0 vector in S ok. Having defined this will study a special class of block code that is known as linear block code.

(Refer Slide Time: 05:59)



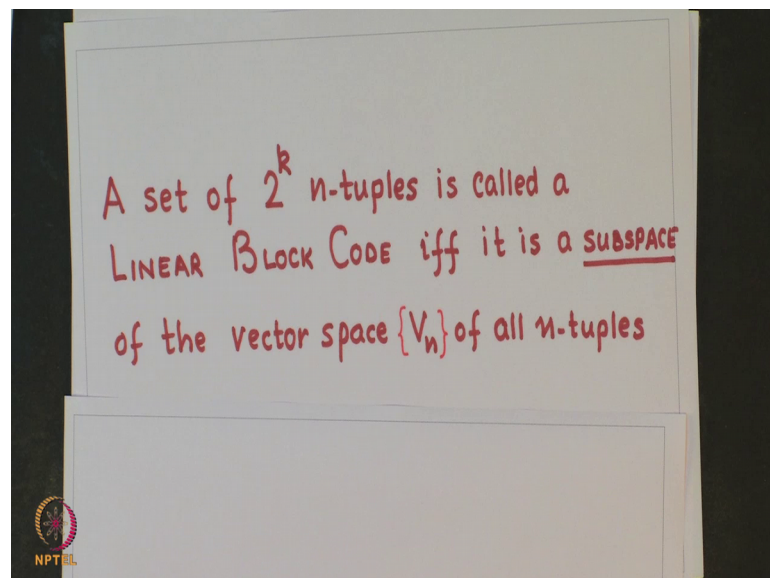
So, the definition of the linear block code is as follows. Let V_i and V_j be two codewords. Remember this V_i and V_j they correspond to 2 different message sequences say m_i and m_j respectively. Now if we have these 2 codewords which will be each of this will be n -tuple and it comes from the binary block code, then the code is said to be linear if and only if $V_i \oplus V_j$ is also a codeword in the same n, k binary block code. If this condition is satisfied for any block code then we say that, that block code is a linear block code an example of that would be let us consider a vector space V_4 .

(Refer Slide Time: 07:12)



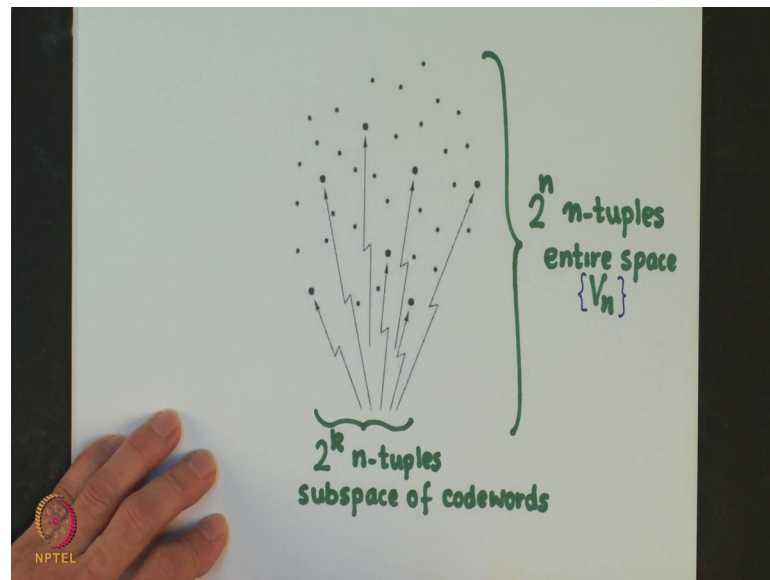
So, which denotes that there will be 2 raised to 4, 16 4 tuples the length of the each of the vector here is 4 bits. And these are all four bit vectors which we can form. Now a subspace of this is chosen by this vectors and it easy to verify that this four vectors satisfy the definition of subspace. So, first of all, all 0 vector is included in this subset of V_4 and next if we take the summation of any 2 vectors, we will get a vector which lies in the subspace.

(Refer Slide Time: 08:26)



So, based on this definition of linear block code and a subspace, we can say that a set of 2^k n -tuples is called a linear block code, if and only if it is a subspace of the vector space V_n of all n -tuples. So, this can be actually visualized with the help of this figure here.

(Refer Slide Time: 08:53)



So, imagine the vector space V_n , which comprises of 2^n n -tuples. So, within this vector space there exists a subset of 2^k n -tuples which make up a subspace.

Now, this 2^k vectors of points are shown here among the 2^n points and these bigger dots represent the codeword of this subspace. So, a message is encoded into any 1 of the 2^k allowable code vectors and then transmitted. Now because of noise on the channel a perturbed version of the codeword, which will be one of the other 2^n vectors in the n -tuple space will be received.

Now, if the perturbed vector is not too dissimilar from the valid codeword, the decoder will be able to decode the message correctly. So, the basic goal in choosing a particular code is similar to the goals in selecting a set of modulation waveforms and this goal can be stated as follows. We want to choose 2^k n -tuple subspace of codewords from this entire space, in such a way that we get the coding efficiency by packing the entire space with as many codewords as possible.

So, what we are indirectly saying that, we want to have small amount of redundancy which implies that we do not require excess bandwidth. And the second condition is that we want this codewords to be selected in such a way that, they are as far apart from 1 another as possible. So, that even if the vectors experience, some kind of corruption during transmission they may still be correctly decoded with a high probability.

(Refer Slide Time: 11:24)

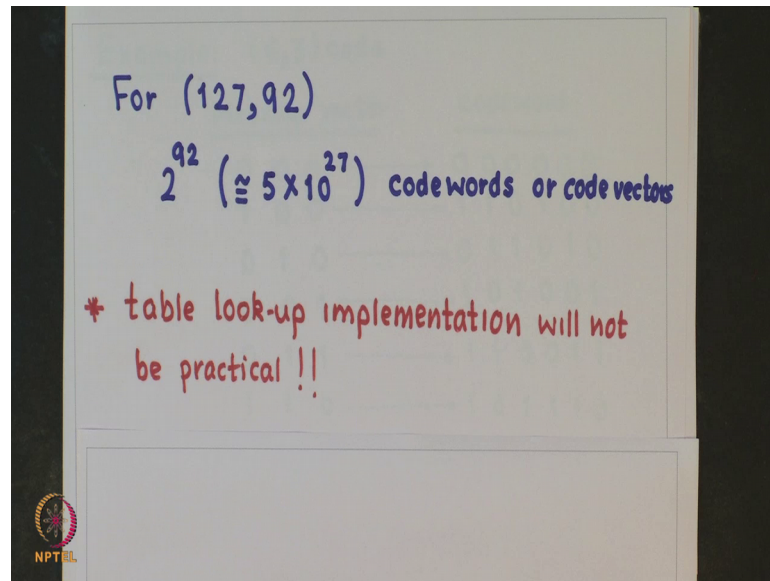
Example: (6,3) code

| <u>Message vector</u> | <u>codeword</u> |
|-----------------------|-----------------|
| 0 0 0 | 0 0 0 0 0 0 |
| 1 0 0 | 1 1 0 1 0 0 |
| 0 1 0 | 0 1 1 0 1 0 |
| 0 0 1 | 1 0 1 0 0 1 |
| 0 1 1 | 1 1 0 0 1 1 |
| 1 1 0 | 1 0 1 1 1 0 |
| 1 0 1 | 0 1 1 1 0 1 |
| 1 1 1 | 0 0 0 1 1 1 |

.Now, an example of 6, 3 linear code is given here. So, we have message sequence of 3 bits. So, there are eight messages and corresponding to these eight messages we have 6 tuples. And we can see that we will have 2^3 that is 8 n-tuples 6 tuples that is and out of that we have chosen 8 6 tuples in such a way that it forms a subspace and it forms a linear block code.

Now, suppose the example instead of 6 comma 3, if I had 127 comma 92 block code. Now in that case I would have 2^{127} codewords or code vectors. Now obviously, the job of the encoder is to assign one codeword to each of the message vector.

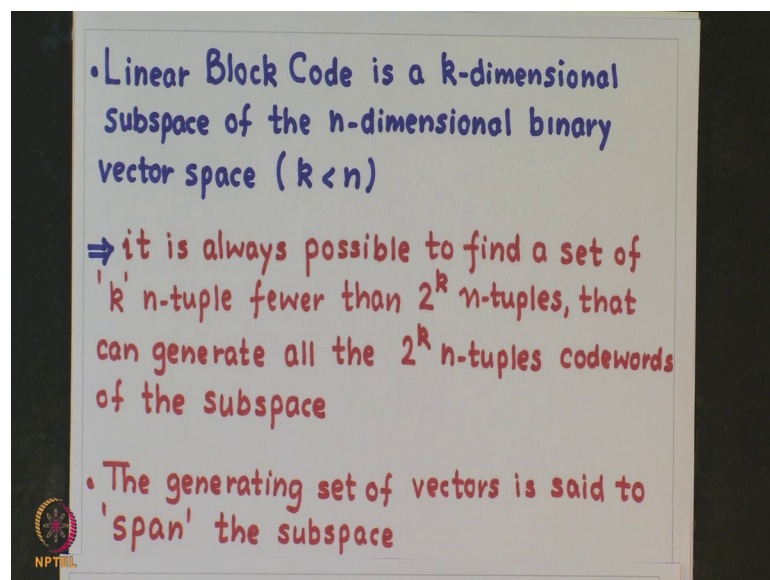
(Refer Slide Time: 12:42)



Now does this imply that we will have to set up some kind of a table lookup type of thing and if we assume table lookup then; obviously, in this case it will fail because the requirement would be very huge and implementation will not be practical.

So, let us look at a feasible implementation of a linear block code.

(Refer Slide Time: 13:09)

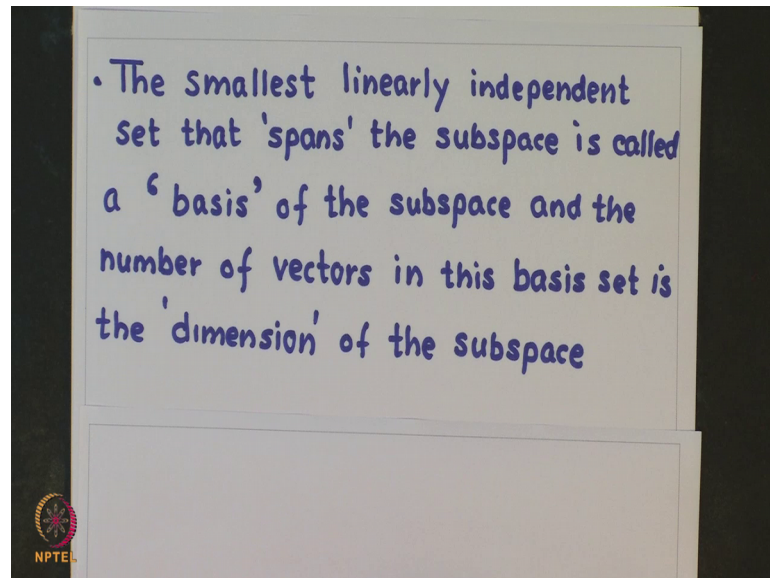


Observe that linear block code is a k -dimensional subspace of the n -dimensional binary vector space, where k is less than n this is the definition of a linear block code. So, what this implies that, it should be always possible to find a set of k n -tuple vectors which are;

obviously, fewer than 2 raised to k n -tuples, that can generate all the 2 raised to k n -tuples codewords.

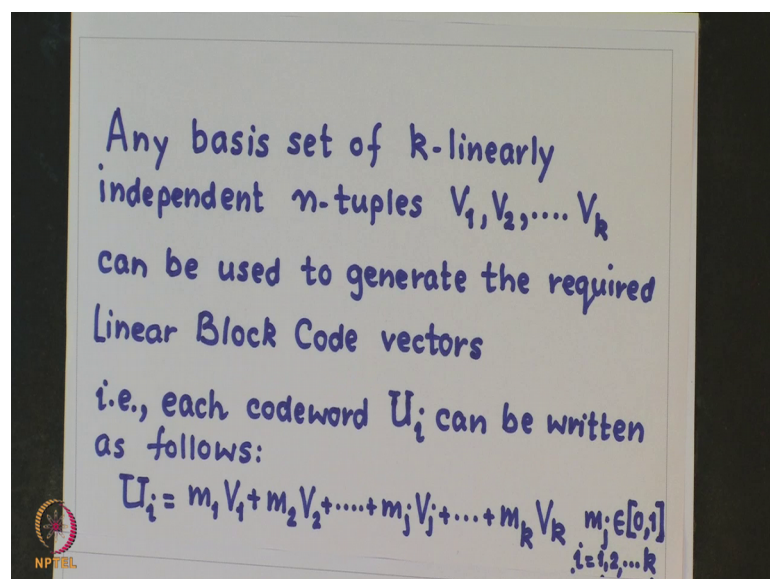
So, this generating set of k n -tuple vectors is said to span the subspace.

(Refer Slide Time: 14:09)



And the smallest linearly independent set that spans the subspace is called a basis of the subspace and the number of vectors in this basis set, is the dimension of the subspace. So, any basis of k linearly independent n -tuples can be used to generate the required linear block code vectors.

(Refer Slide Time: 14:40)

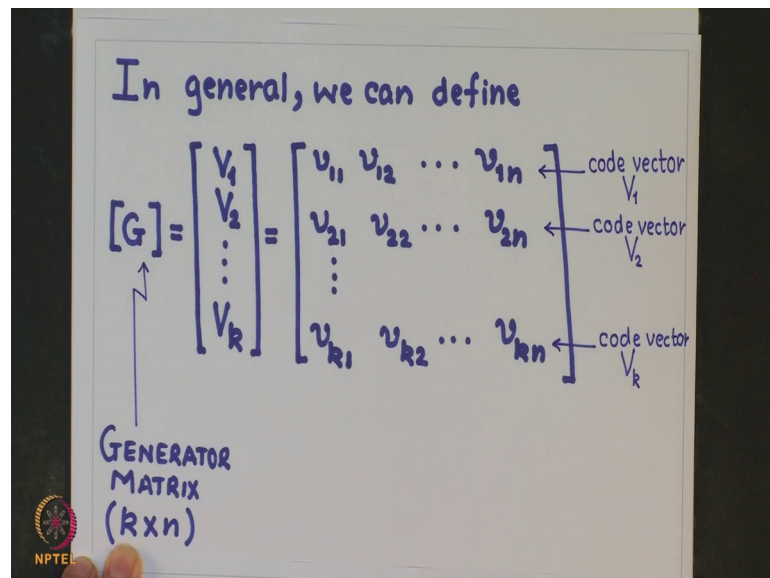


So, we can conclude. So, what this means that, each codeword which I denote by U_i the capital U denotes codeword and also a vector row vector.

So, this can be written as follows; U_i is equal to m_1 which is the first bit of the message sequence multiplied by the vector V_1 plus m_2 times V_2 . So, remember this V_1, V_2, V_j are all vectors whereas, m_1, m_2, m_j are the scalars and m_1, m_2, m_j are the components of the message vector, which is formed out of the message sequence. So, your m_j is going to be 0 or 1 and your i will be from 1 to up to k because the dimension of the subspace representing the linear block code is k .

So, we can in general write this in a matrix form as follows.

(Refer Slide Time: 16:12)



In general, we can define

$$[G] = \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_r \end{bmatrix} = \begin{bmatrix} v_{11} & v_{12} & \dots & v_{1n} \\ v_{21} & v_{22} & \dots & v_{2n} \\ \vdots & \vdots & \dots & \vdots \\ v_{r1} & v_{r2} & \dots & v_{rn} \end{bmatrix}$$

Labels on the right side of the matrix:

- code vector V_1
- code vector V_2
- code vector V_r

Label at the bottom left:

GENERATOR MATRIX (k x n)

. So, first we define what is known has a generator matrix of the size k by n and there are k rows and n elements. K rows corresponding to the k independent vectors which form the basis of the subspace that is why k and each of this vector has n components because the code vectors are n -tuples and the vector elements for the code vector V_1 are v_{11}, v_{12} up to v_{1n} and similarly the components of the code vector V_k are v_{k1}, v_{k2}, v_{kn} .

So, this code vectors they form the rows of this generator matrix. So, now, this equation which we have written is equivalent to writing in matrix form using the definition of generator matrix as follows.

(Refer Slide Time: 17:35)

Handwritten notes on a whiteboard:

$$\underline{m} = [m_1, m_2, \dots, m_j, \dots, m_k] \leftarrow \text{row vector}$$

message vector

$$\underline{m} \rightarrow \underline{U} \leftarrow \text{codeword generated}$$
$$\underline{U} = \underline{m}[G]$$

NPTEL logo is visible in the bottom left corner.

So, message vectors are the message bits m_1 to m_k , it is a row vector and the corresponding code vector a codeword generated by the encoder is denoted as U . So, that U would be equal to this is a row vector multiplied by the generator matrix.

And this is equivalent to this, here this is corresponding to the codeword is corresponding to the message m_i . So, this is a message m_i . So, this is a vector remember this is a vector here correct ok. So, for the earlier example which we had studied we can obtain the generator matrix as follows.

(Refer Slide Time: 18:42)

Handwritten notes on a whiteboard:

Earlier example:

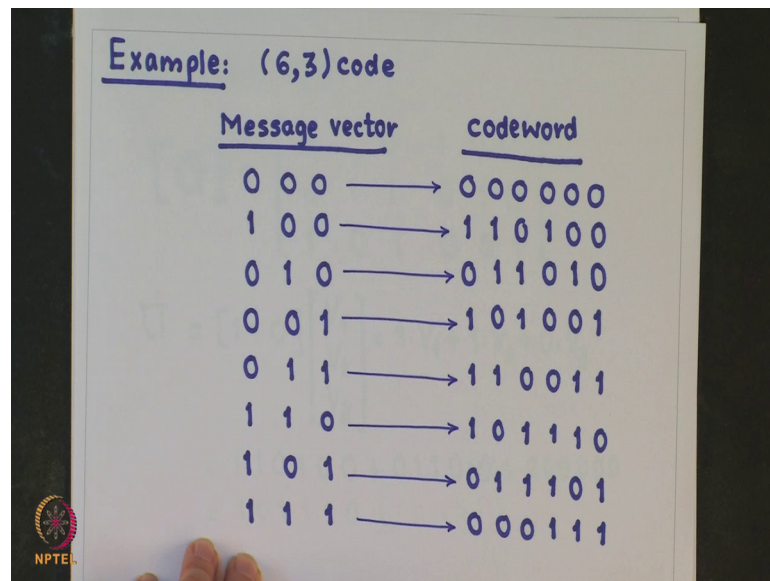
$$[G] = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$
$$\underline{U} = [110] \begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix} = 1 \cdot V_1 + 1 \cdot V_2 + 0 \cdot V_3$$
$$= 110100 + 011010 + 000000$$
$$= 101110 \text{ (check!)}$$

NPTEL logo is visible in the bottom left corner.

Generator matrix is given by this and we can see whether this produces the codewords as expected. So, if you take the message vector to be 1 1 0 and take this V_1 , V_2 and V_3 and generate it using this equation, we get the output to be equal to this and you can check that this is identical to what we had got earlier.

Now, if you look at the 6 3 codeword, which we studied as an example here it has some kind of a characteristic and the characteristic is as follows.

(Refer Slide Time: 19:37)

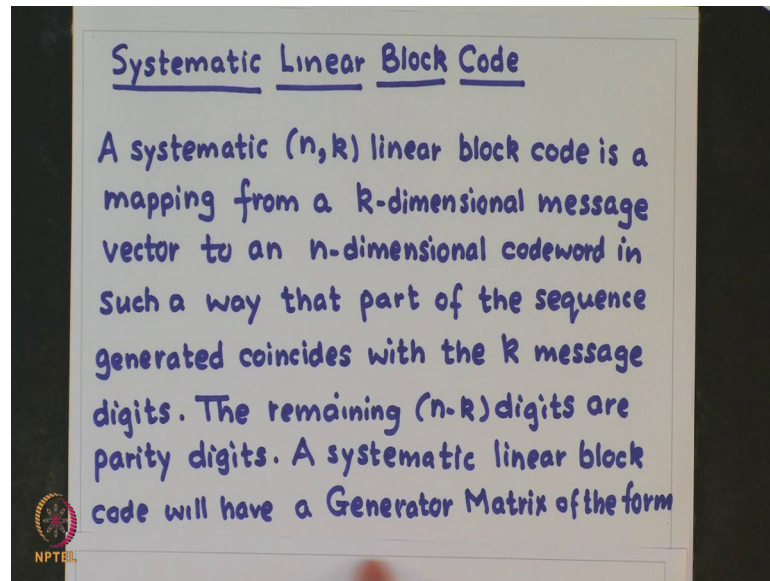


| <u>Message vector</u> | <u>codeword</u> |
|-----------------------|-----------------|
| 0 0 0 | 0 0 0 0 0 0 |
| 1 0 0 | 1 1 0 1 0 0 |
| 0 1 0 | 0 1 1 0 1 0 |
| 0 0 1 | 1 0 1 0 0 1 |
| 0 1 1 | 1 1 0 0 1 1 |
| 1 1 0 | 1 0 1 1 1 0 |
| 1 0 1 | 0 1 1 1 0 1 |
| 1 1 1 | 0 0 0 1 1 1 |

If you look at this message vectors out here and if you look at the codewords, we see that these three message bits they appear at the end of the cordword here. Similarly if you see this, it appears at the end of the codeword here and similarly 0 1 0 it appears at the end of the codeword here and this is true for all the message vectors and the corresponding codewords.

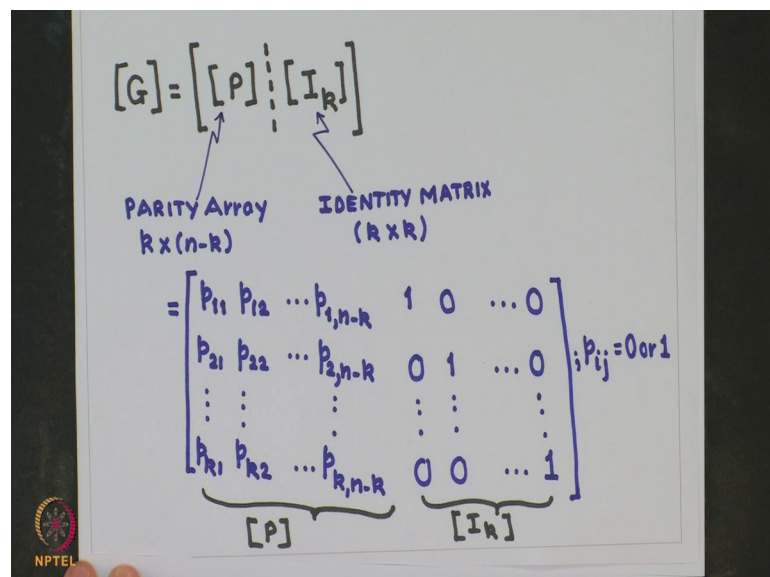
Now, this kind of a linear block code is what is known as systematic linear block code. So, what is a systematic linear block code?

(Refer Slide Time: 20:23)



It is a linear block code and it is a mapping from a k dimensional message vector to an n dimensional codeword, in such a way that part of the sequence generated coincides with the k message digits. In our case k message bits because we are talking about a binary code and the remaining n minus k digits are parity digits. So, a systematic linear block code will have a generator matrix of the following form.

(Refer Slide Time: 21:08)



So, the generator matrix G is partitioned into 2 sub matrices; this is known as p array and this is the identity matrix. So, parity array is of the size k by n minus k whereas, the

identity matrix is of the size k by k and the elements of this generator matrix are given as shown here.

So, this corresponds to the parity array and this corresponds to the identity matrix, where the elements p_{ij} is either 0 or 1.

(Refer Slide Time: 22:07)

for message: $\underline{m} = [m_1 \ m_2 \ \dots \ m_k]$

codeword $\underline{U} = [u_1 \ u_2 \ \dots \ u_i \ \dots \ u_n]$

$$\underline{U} = [m_1 \ m_2 \ \dots \ m_k] \times \begin{bmatrix} p_{11} & p_{12} & \dots & p_{1i} & \dots & p_{1,n-k} & 1 & 0 & \dots & 0 \\ p_{21} & p_{22} & \dots & p_{2i} & \dots & p_{2,n-k} & 0 & 1 & \dots & 0 \\ \vdots & \vdots & & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ p_{k1} & p_{k2} & \dots & p_{ki} & \dots & p_{k,n-k} & 0 & 0 & \dots & 1 \end{bmatrix}$$

$u_i = m_1 p_{1i} + m_2 p_{2i} + \dots + m_k p_{ki}$; for $i=1, 2, \dots, (n-k)$

$= m_{i-n+k}$; for $i=(n-k+1), \dots, n$

Now given this for a message specified by this row vector, the codeword U which is has the elements from u_1 to u_n , u will be obtained by the usual matrix multiplication of the row vector corresponding to the message vector multiplied by the g generator matrix and any of the element of this code vector or codeword u say u_i is given by this expression on the right hand side.

So, this is u_i will be given by this term for i equal to 1 to up to n minus k whereas, for i equal to n minus k plus 1 up to n will be the same as the message bits because this is a systematic linear code.

(Refer Slide Time: 23:43)

codeword / code vector
 $\vec{U} = [u_1, u_2, \dots, u_i, \dots, u_{n-k}, u_{n-k+1}, \dots, u_n]$
 $= [p_1, p_2, \dots, p_i, \dots, p_{n-k}, m_1, \dots, m_k]$
PARITY BITS message bits

Now this vector codeword a codeword vector which is given in the terms of the elements u_i , it is also expressed in a different manner and the first n elements of this codeword or code vector are also known as parity bits and rest of the elements or components in this vector are the message bits, because we are discussing the systematic linear code.

(Refer Slide Time: 24:11)

$\vec{U} = [p_1 \ p_2 \ \dots \ p_{n-k} \ m_1 \ m_2 \ \dots \ m_k]$
where
 $p_1 = m_1 p_{11} + m_2 p_{21} + \dots + m_k p_{k1}$
 $p_2 = m_1 p_{12} + m_2 p_{22} + \dots + m_k p_{k2}$
 \vdots
 $p_{n-k} = m_1 p_{1, n-k} + m_2 p_{2, n-k} + \dots + m_k p_{k, n-k}$

So, this parity bits are obtained as shown on this slide. So, this parity bits p_1 to p_{n-k} are exactly the same as the first $n-k$ elements or components of the vector u .

(Refer Slide Time: 24:45)


For the earlier example of (6,3) code:

$$U = [m_1 \ m_2 \ m_3] \begin{bmatrix} 1 & 1 & 0 & | & 1 & 0 & 0 \\ 0 & 1 & 1 & | & 0 & 1 & 0 \\ 1 & 0 & 1 & | & 0 & 0 & 1 \end{bmatrix}$$

$[P]$ $[I_3]$

$$= \begin{bmatrix} m_1+m_3 & m_1+m_2 & m_2+m_3 & m_1 & m_2 & m_3 \end{bmatrix}$$

\downarrow \downarrow \downarrow

$$\equiv \begin{bmatrix} p_1 & p_2 & p_3 & m_1 & m_2 & m_3 \end{bmatrix}$$


So, let us take a earlier example and obtain the codeword for the any general message m . So, we take the example 6 comma 3 code for which the generator matrix is given as shown here and the codeword for any message vector would be obtained as shown here. So, this example shows how the parity bits are obtained by linear combinations of the message bits. So, these parity bits in fact, provide the ability for error correction and error detection.

We will continue with our study of linear block code and see basically how to detect the errors and correct the errors. So, we will define the notion or what is known as parity check matrix and this we will do in the next class.

Thank you.