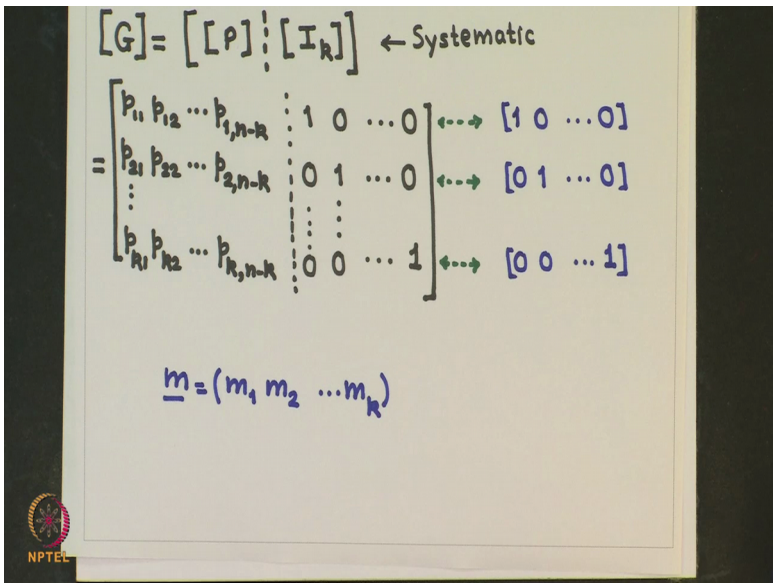
Principles of Digital Communications
Prof. Shabbir N. Merchant
Department of Electrical Engineering
Indian Institute of Technology, Bombay

Lecture - 62
Channel Coding - III

In the previous class, we learned how to generate code word for any message vector for a n comma k linear code using a generator matrix. So, the generator matrix is of the size k by n and the rows of the generator matrix are k independent vectors, which span the k dimensional subspace of a higher n dimensional vector space of n tuples. We also studied a subclass of linear block code known as systematic linear code. In systematic linear code the last k elements of a code word exactly match with the corresponding k elements of the message vector. The generator matrix of such a linear code is expressed as shown here on the slide.

(Refer Slide Time: 01:40)



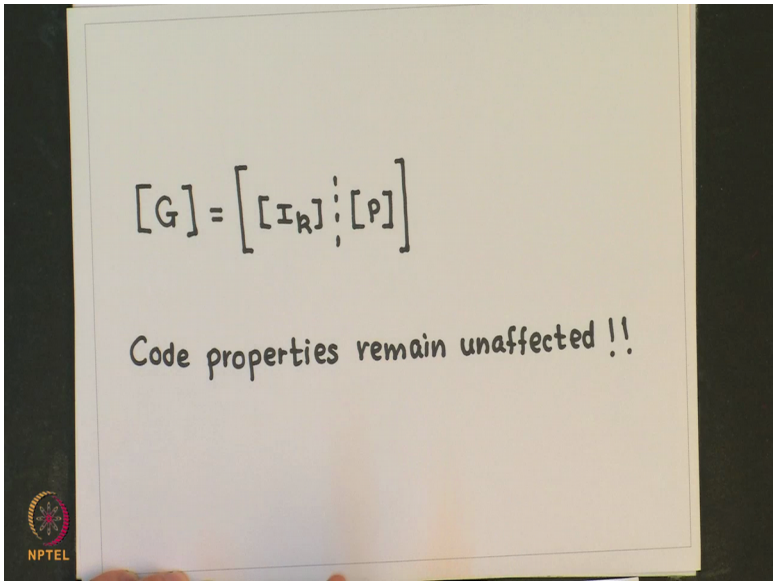
So, this is the parity array and this is the identity matrix of size k by k and the matrix G is of this form.

Now, based on our discussions, we can easily observe that each of these rows of this generator matrix G correspond to the code words of the unit message vectors, where a unit message vector is defined as a message vector with only 1 nonzero element. So, this

row is a code word for this message vector this row is a code word for this message vector and similarly the last row is a code word for this unit message vector

Now, any message signal which is k tuple can be represented in terms of this unit message vectors. So, the code word for this message vector is the linear combinations of this code words for the unit message vectors.

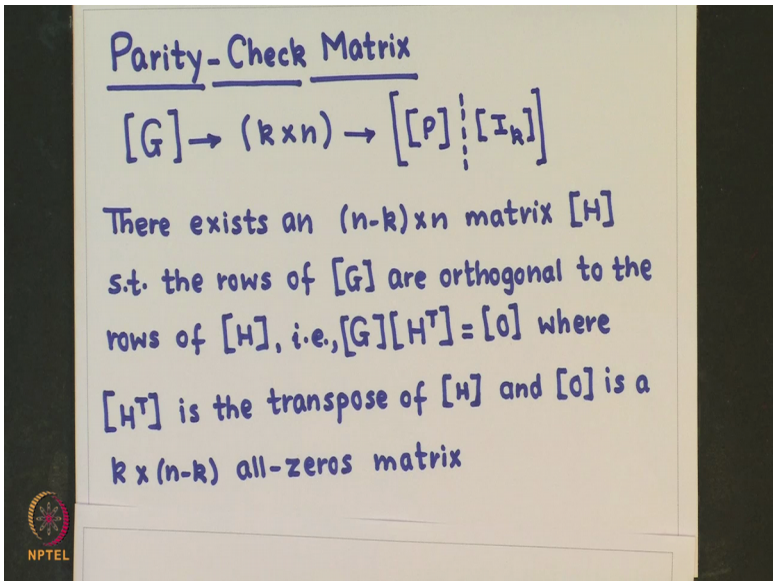
(Refer Slide Time: 03:31)



Now, in the literature the generator matrix is also written in this form where the parity array follows the identity matrix of size k by k. Now in this case the first k elements of a code word will be exactly match with the k elements of the corresponding message vector otherwise the code properties remains unaffected. So, as far as our discussion is concerned we will consider our generator matrix to be of this form.

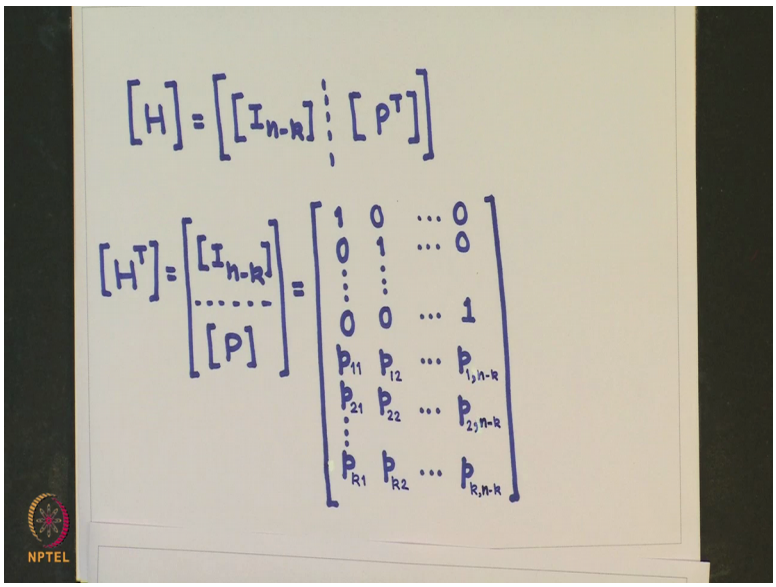
Now, for any generator matrix, there exists what is known has parity check matrix H and the property of that matrix H follows.

(Refer Slide Time: 04:30)



What it says that there exists an n minus k by n matrix parity check matrix H such that the rows of G are orthogonal to the rows of the parity check matrix H, that is G H transpose is equal to a 0 matrix, where the 0 matrix is of the size k by n minus k and it contains all 0 elements.

(Refer Slide Time: 05:09)

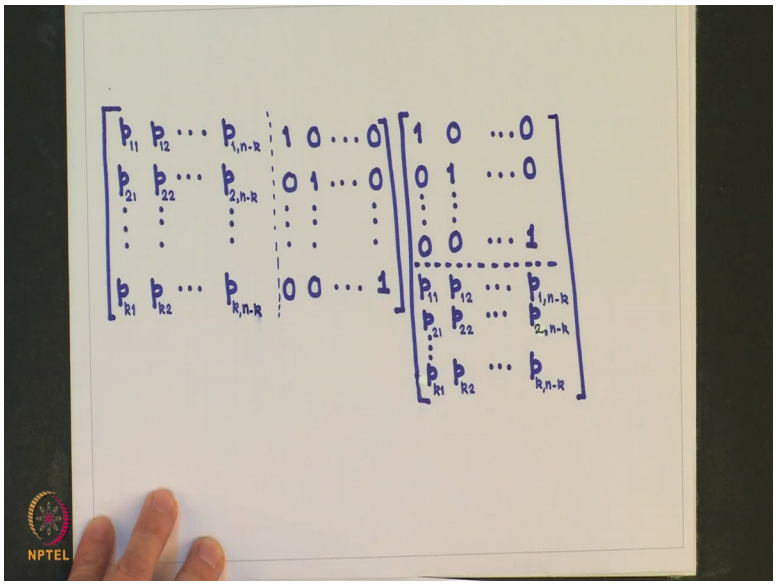


So, the parity check matrix for the matrix G is as shown here on the slide. So, you have an identity matrix of size n minus k by n minus k followed by the transpose of the parity array and so, H transpose would be the transpose of this transpose of this. So, we get the

H transpose matrix this and we know what is the parity array. So, if you substitute the values for that, H transpose matrix is as shown here.

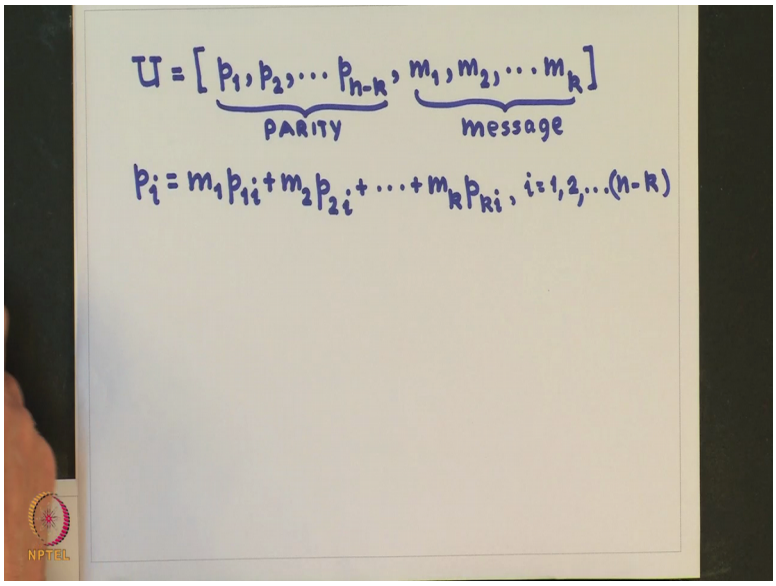
Now, once we write H transpose matrix and G matrix it is easy to see that gh transpose is equal to a 0 matrix.

(Refer Slide Time: 06:09)



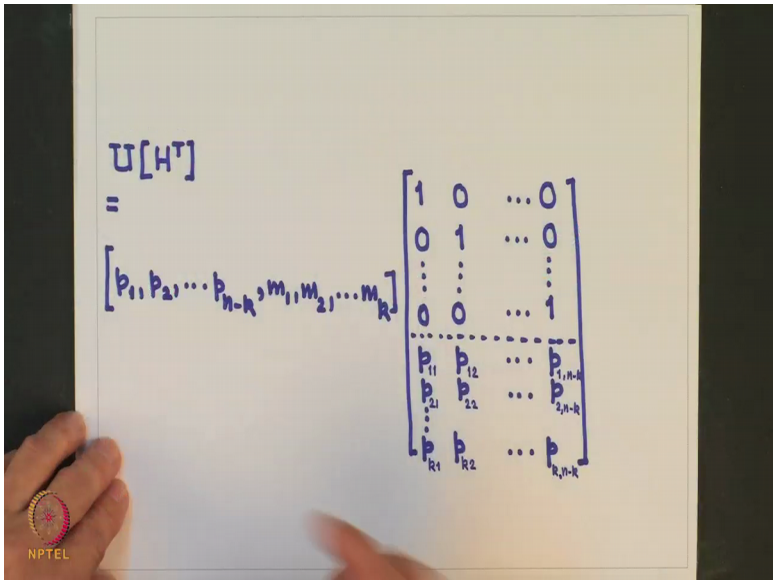
So, this slide clearly demonstrates that gh transpose this is gth matrix and this is h transpose is equal to 0. So, if we take the first row and multiply with the first column out here correct you will get p 11 and rest of the terms here we will go to 0 and then you will pick up again p 11. So, what will be picked up is p 11 plus p 11. So, that will be equal to 0 and then if you multiply this by this again you will pick up p 1 2 and here you will pick up p 1 2. So, p 1 2 plus p 1 2 again if this would be equal to 0 correct using the modular 2 arithmetic. So, any row multiplied with any column out here will give you 0 and similarly you can take any other row in the G matrix and multiply with the column out here it will turn out to be 0. So, G H transpose is a 0 matrix.

(Refer Slide Time: 07:26)



Now, we know that our code vector U is as shown here, we have first n minus k parity bits, followed by k bits representing the message. This is the systematic form of the code word and these parity bits are obtained by this relationship, which we have seen last time it is the linear combinations of the elements from the parity array. And with this definition of our code word, it is now easy to see what happens when I take U code word and multiply with H transpose.

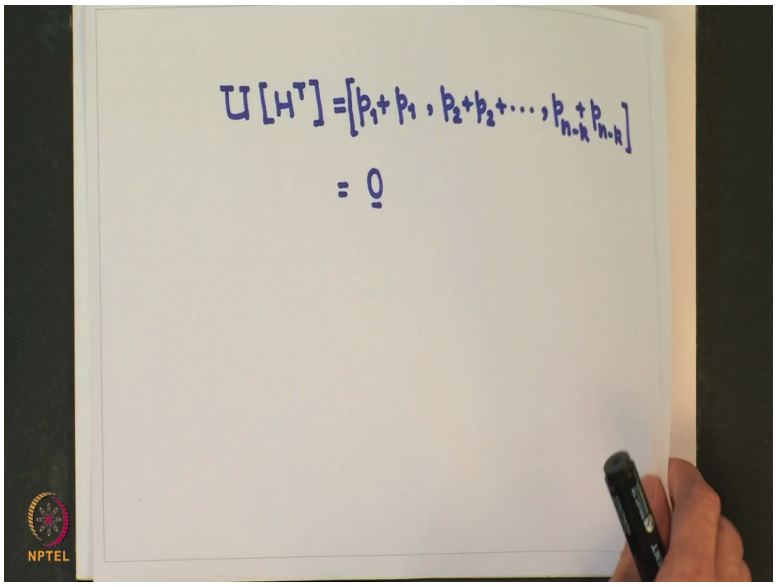
(Refer Slide Time: 08:09)



So, this is my code word U and this is my H transpose. So, if we take this and multiply by the first column out here, we will get p 1 1 plus you will get the linear combinations of this weighted by m 1 2 m k.

And similarly if we take this and multiply with the second column out here, we will get p 2 plus the weighted combinations of these elements by m 1 m 2 up to m k. So, on multiplying U by H transpose it is easy to see that we get the following result.

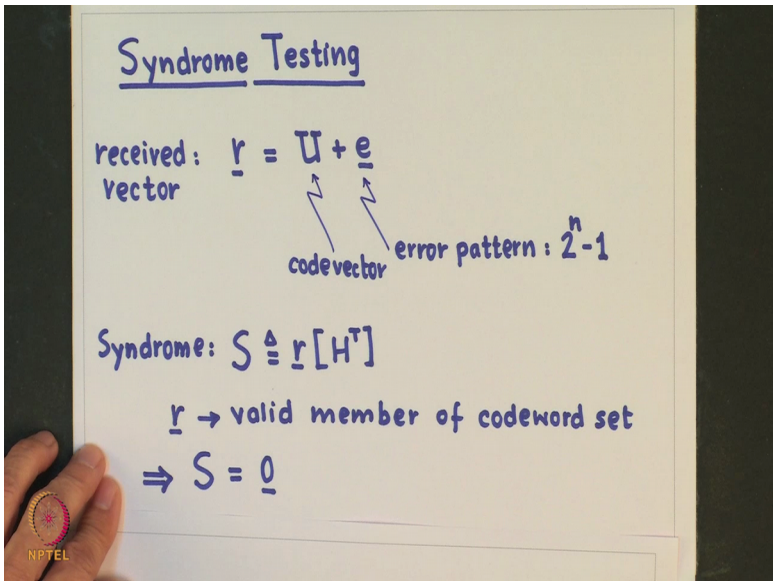
(Refer Slide Time: 09:05)



The first element in this would be p 1 plus p 1 and the second would be p 2 plus p 2 and the last one would be p n minus k plus p n minus k now this will be all 0. So, we get a 0 row vector of size 1by n minus k.

So, what it says that the code word when multiplied by H transpose would give you 0 correct. Now this is a very powerful result which can be used to find out if the received vector is a really a code word or non code word and this can be done as follows. This is known as syndrome testing. So, we have a receive vector which is the transmitted vector U plus let us assume there is some error vector.

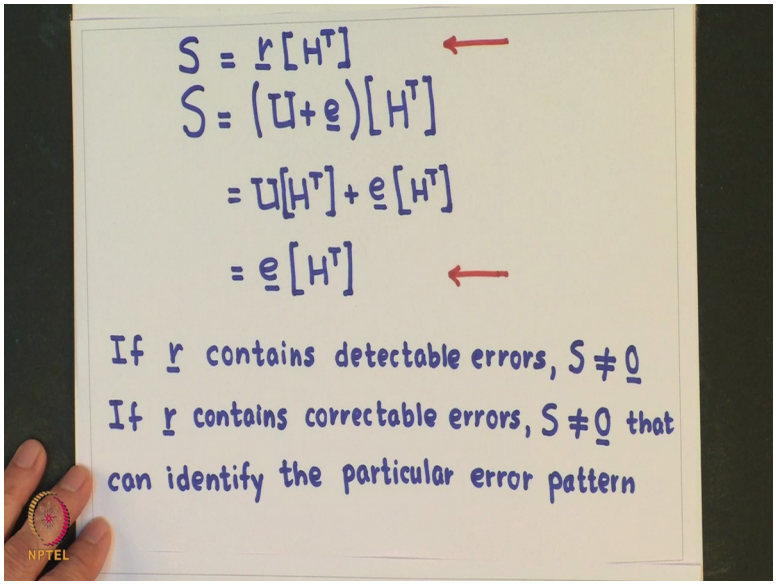
(Refer Slide Time: 10:04)



So, this error vector is also n tuple. So, the receive vector is the sum of the 2.

Now, if we compute r multiplied by H transpose, if r is a valid member of code word set, then this syndrome would be equal to 0 otherwise, it would be non zero this is very clearly seen if we write the expansion as follows.

(Refer Slide Time: 10:50)

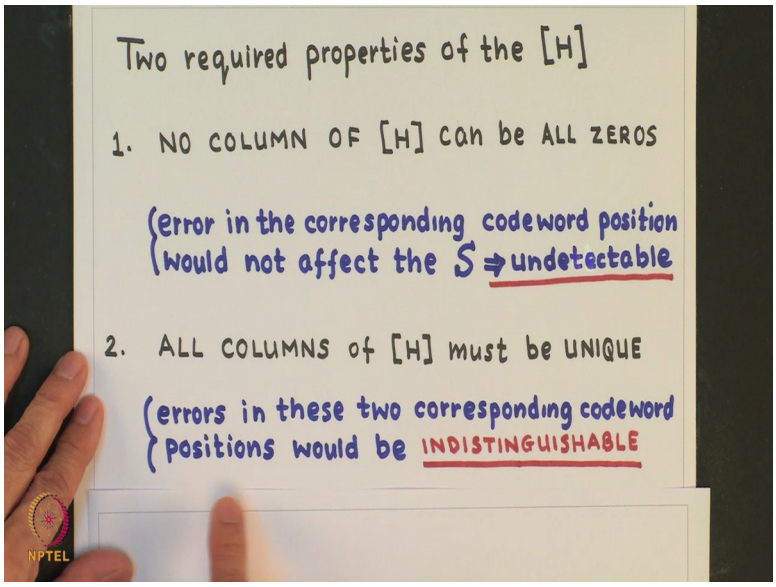


So, S is equal to r multiplied by H transpose r is U plus e multiplied by H transpose. So, I open out the bracket I get U H transpose plus E H transpose. Now we know that transpose is equal to 0. So, the result would be e multiplied by H transpose.

So, what this result shows that the syndrome which we get or multiplying the received vector by H transpose is same as the syndrome, which we will get if we take the error pattern which has corrupted the transmitted code word and multiplied with the H transpose you will get the same result. So, if r contains detectable errors, then the syndrome will be nonzero and if r contains correctable errors, then also the syndrome will be nonzero and, but in this case we can identify the particular error pattern. This will become very clear as we proceed ahead.

But the important thing is that on receiving the vector or multiplication with H transpose the 0 value tells us that the received vector is a code word and if it is nonzero, it says the received vector is non code word what to do later on we will see as we proceed ahead. But there are 2 required properties of this parity check matrix and these are as follows.

(Refer Slide Time: 12:43)



The first is no column of h can be all zeros. So, what this implies that no row of h transpose can be all zeros.

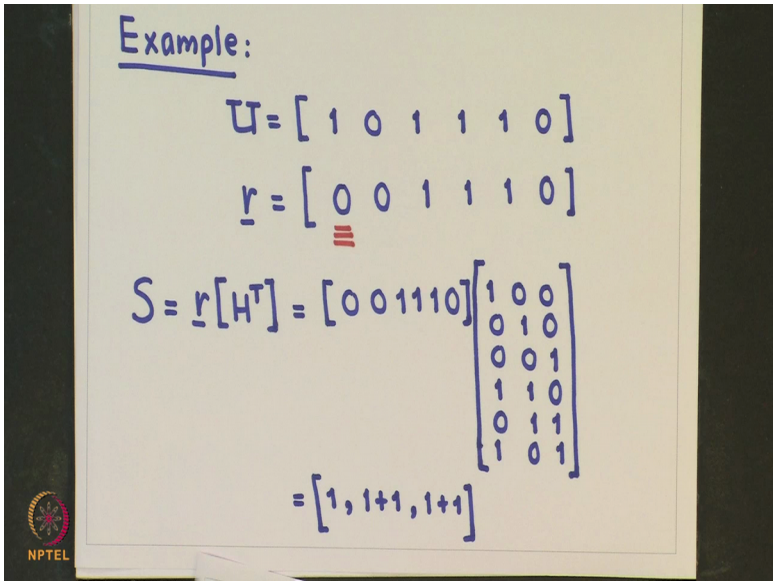
Now, this is easy to see why this condition should be satisfied, because if it were true then error in the corresponding code word position would not affect the syndrome and thus the error would go undetectable look here. If any of this if any of the column of H is 0; that means, the row of H is 0 if there is error in that position this row being 0 the syndrome will not get affected because if the error is there in this position because of all

zeros it will go to 0 there will be no contribution from that error and that is why this will go undetectable

And the second is that all columns of H must be unique. So, what this implies that all rows of H transpose should be unique because if that does not happen then the error in 2 positions will be indistinguishable. For example, if I have error corresponding to this position. So, this will be at this point, if I have an error at this point corresponding to this row and I have errors at this point corresponding to a second row. If both are identical then I am not sure whether the error has occurred here or the error has occurred here correct. So, that is why we say that the rows of H transpose should be all distinct implying that the columns of H should be all unique or distinct. So, if that condition is violated then errors in this two corresponding code word positions would be indistinguishable.

So, let us take one simple example, suppose I have a transmitted code word to be of this form and then let us assume that there is an error in the first bit of the transmitted code word.

(Refer Slide Time: 14:54)

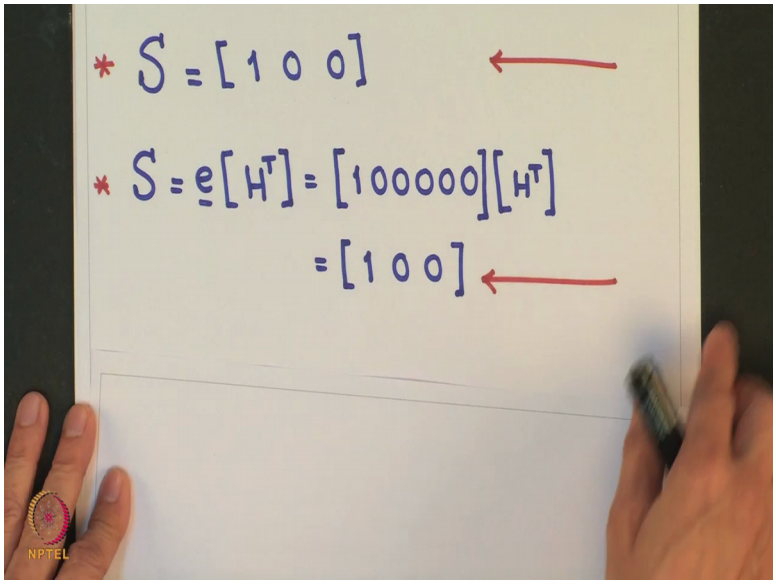


So, we get the received vector to be as shown here. So, there is an error in the first bit.

Now, if we calculate the syndrome for this by multiplying r by H transpose and H transpose we choose the same generator matrix, which we had chosen in the previous

class as an example and for that generator matrix, we can write the H transpose and then you multiply this we get the answer to be after multiplication equal to this which is equivalent.

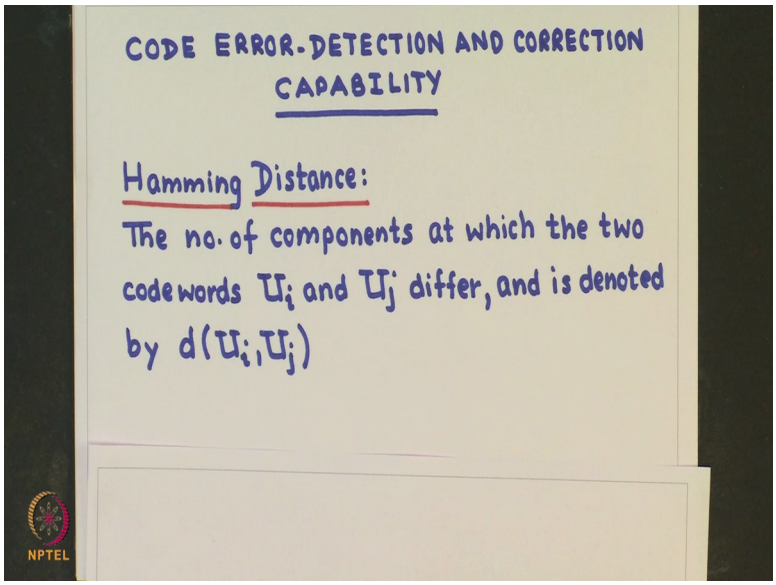
(Refer Slide Time: 15:42)



To 1 0 0, 1 0 0 correct and if we take we know that there is error only in the first bit. So, the error pattern is 10 0 0 0 0 and if we take that error pattern and multiply with H transpose we get 1 0 0 0. So, what this says what we had said earlier that, syndrome obtained by taking the multiplication of the receive vector which H transpose is the same as the syndrome, which we will get or multiplying the error pattern with the h transpose fine.

Now, we would like to understand the error detection and correction properties of a linear code; now in order to do that we need to understand some primaries and let us do that first. So, the first thing is the concept of hamming distance.

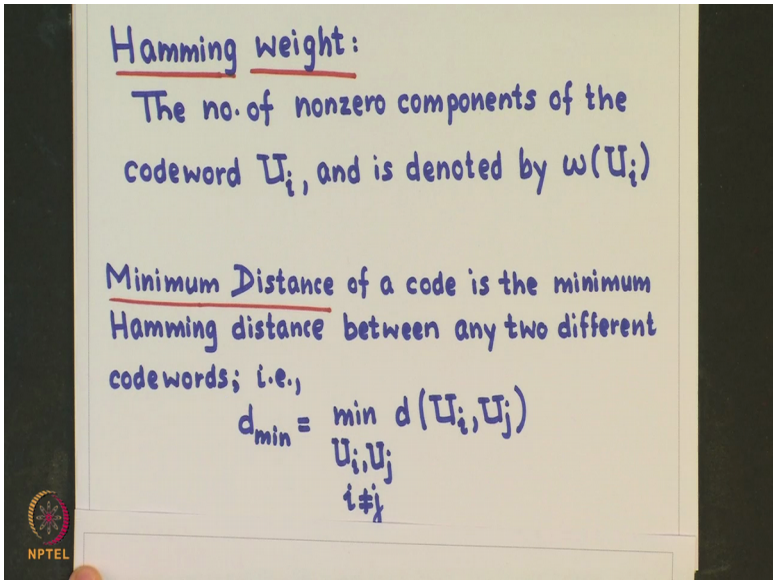
(Refer Slide Time: 16:50)



The hamming distance is defined as the number of components at which the 2 code words U i and U j. U i is corresponding to the message vector mi and U j is corresponding to message vector mj. So, the number of components at which the 2 code words U i and U j differ and this is denoted by this notation is the hamming distance between 2 code words.

Then there is a concept of hamming weight which is defined as the number of nonzero components of the code word U i and this is denoted by w U i.

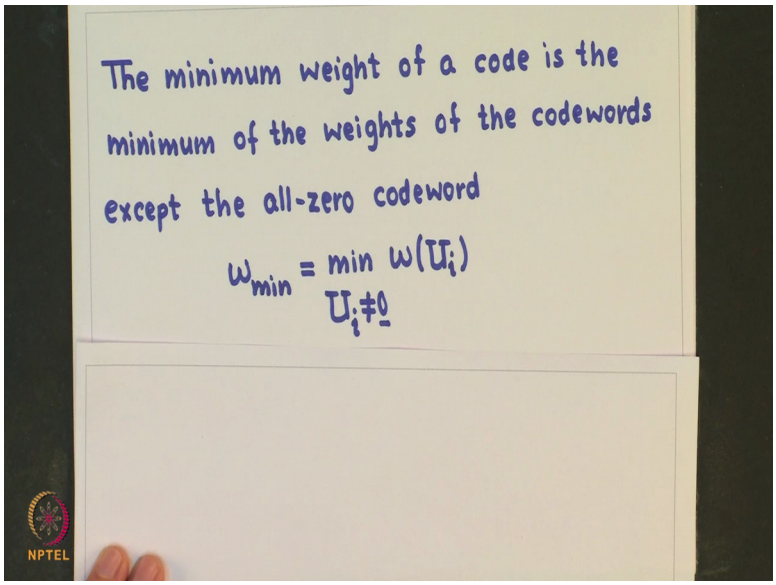
(Refer Slide Time: 17:40)



There is a concept of minimum distance of a code a code consists of a set of code words corresponding to the set of message vectors. So, the minimum distance of a code is the minimum hamming distance between any 2 different code words in the code. So, d minimum is equal to minimum you have to calculate between all U i's and U j assuming that U i and U j are not the same. So, for whichever 2 code word where we get this distance to be minimum that distance between the 2 code word is defined as the d minimum and that form the minimum distance of a code.

Now, the minimum weight of a code is the minimum of the weights of the code words which are there in the code except the all 0 code word.

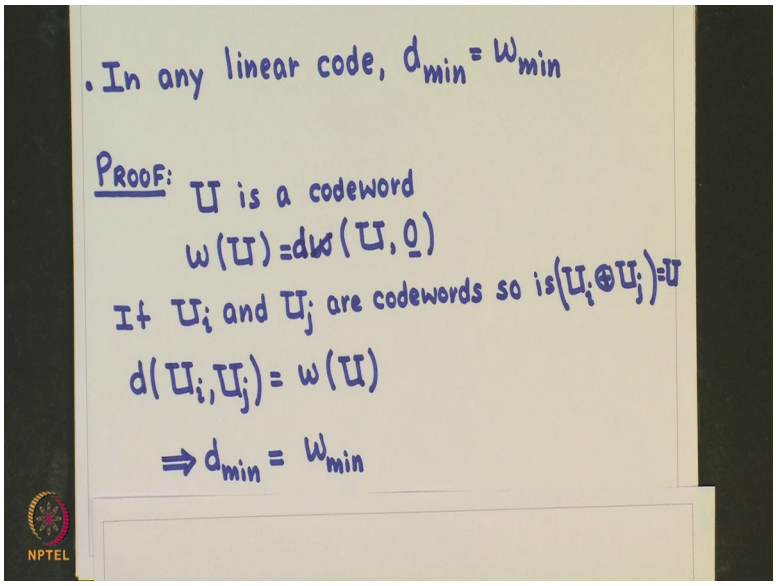
(Refer Slide Time: 18:50)



So, w minimum is minimum of this weight where U i is not equal to 0 vector all 0 vector. Because all the linear codes will have all 0 vector has one of your code word this we have seen in the earlier class this is a requirement for the code to be a linear.

Now there is a theorem, which says that in any linear code d minimum is equal to w minimum.

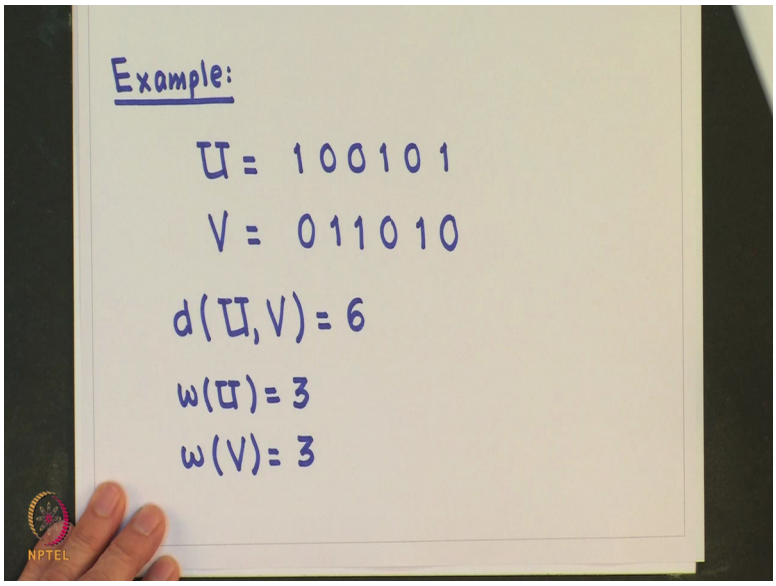
(Refer Slide Time: 19:41)



. So, remember d minimum is the minimum distance of a code and w minimum is the minimum weight of a code word in the code correct. So, the proof follows as follows, if say u is a code word then the weight of this code word is the same as this should be distance. So, this should be distance between U and all 0 code word. So, this is equal to distance correct.

So, now if U i and U j are code words, then if we take the sum of U i and U j then because this belong to a linear code this sum is going to be some code word in that code. So, what it implies that the distance between U i and U j is equal to weight of this code U correct. So, from this it means that an in linear code corresponding to any weight of a code there exists a hamming distance between the 2 code words and corresponding to any hamming distance there exists a weight of a code word. So, in particular it follows that d minimum is equal to w minimum. So, we have proved that in any linear code the minimum is equal to the minimum weight of a code which is w minimum.

(Refer Slide Time: 21:57)



Now, let us take example. So, if I have say 2 code words U and V, which are 6 tuples each, then the distance between the 2 code words is given by the number of components at which the 2 code words U and V differ; so 1, 2, 3, 4, 5, 6. So, the distance here is 6 and the weight of this code U is equal to the number of nonzero components which is 1, 2, 3 and similarly the weight of this code V is 1, 2, 3 we get 3.

Now, having studied this concept of hamming distance, hamming weight; so the next question is how are this d minimum w minimum helpful in deciding the error correction or error detection capability of a linear code. We will try to find the answer to this question next time.

Thank you.