Principles of Digital Communications

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

Lecture - 64
Channel Coding : Decoding using Standard Arrays

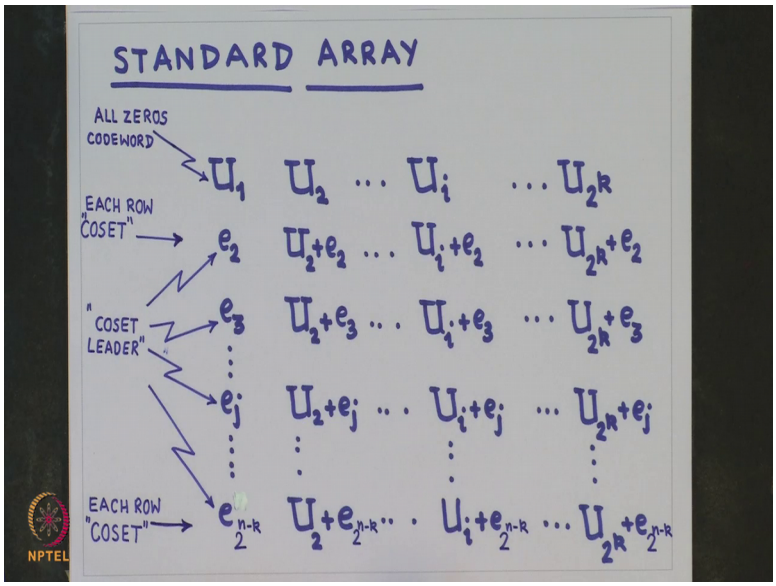
N comma k linear block code has a generator matrix, for which we can define the parity check matrix. Now a codeword consists of symbols 0 and 1. For transmission of this codewords over a physical channel, we need to convert these symbols to waveforms. At the receiver the received signal is passed through match filters and sampled to obtain a received vector.

When the elements of this received vector are compared with the zero thresholds and converted to symbols 0 and 1 then the output of this process we get what is called as received codeword. So, this received codeword or the code vector or multiplication by the transpose of the parity check matrix provides the syndrome

Now, if the syndrome is 0, then the received codeword is decoded as the transmitted codeword. But if the syndrome is nonzero then for error detection decoding, the receiver informs the transmitter for retransmission. But for error correction decoding the syndrome is used to estimate the error pattern, and this error pattern estimate is used to correct the receive codeword to obtain the estimate of the transmitted codeword.

Now, in order to understand the types of correctable error patterns, its important for us to understand the concept of a standard array.

(Refer Slide Time: 02:47)



So, a standard array for n comma k linear block code is formed as follows. The first row of this array is formed by writing all the codewords for the n comma k linear block code and the codewords are denoted by U 1, U 2, U i up to U suffix to 2 raise to k. U 1 is all 0 codeword which is a requirement for a linear code.

Then to write the second row of this standard array, we look among all the end tuples that are not in the first row of this array and choose one of this n tuple, that has the minimum weight and we call it as e 2 and it is written below U 1. And then we write U i plus e 2 below U i for all I from 2 to 2 raise to k.

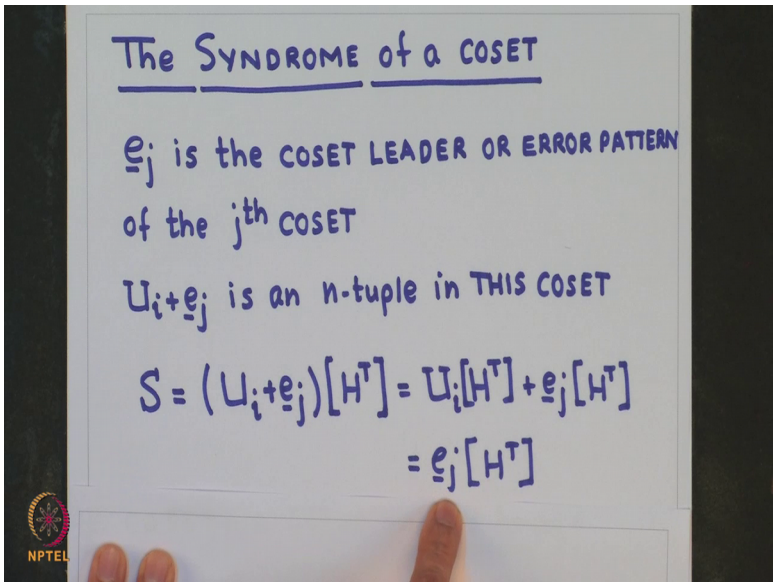
Then the third row of the array is completed in a similar way from the binary end tuples that have not been used in the first 2 rows, we choose one with minimum weight and call it as e 3. Then the elements of the third row become U i plus e 3 and this process is continued until no binary n tuples remain to start a new row.

Now, since there are 2 raise to k codewords, corresponding to 2 raise to k messages in n comma k linear block code we have 2 raise to k columns in the standard array. And now since the total number of all n tuples in the n dimensional vector space is 2 raised to n, the total number of rows which we will have in the standard array will be 2 raise to n minus k.

So, the standard array is of size 2 raise to n minus k multiplied by 2 raise to k. Now each row of this standard array is called a Coset. And the first element of each row or each coset that is e j in general is called the Coset leaders.

Now, let us all the elements of this standard array are distinct. Now let us calculate the syndrome of any element in a coset. So, let us say we look at the coset with the coset leader e j.

(Refer Slide Time: 06:19)

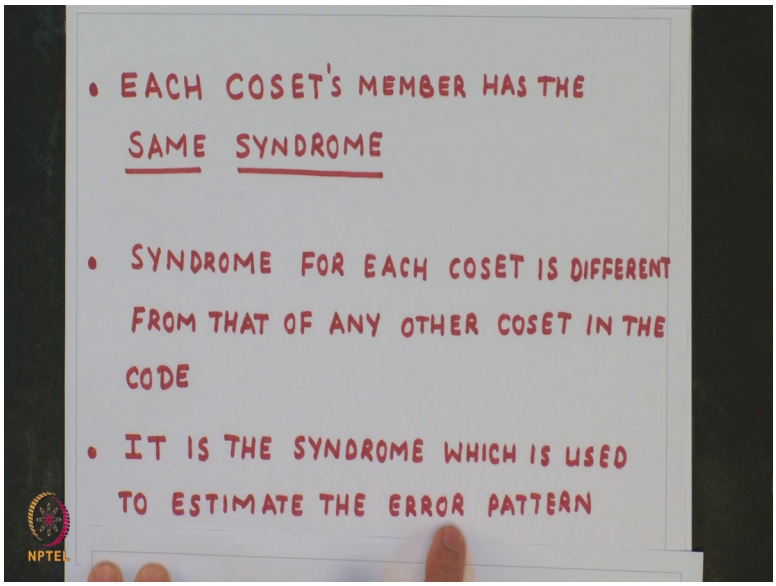


Then if we calculate the syndrome for an element in this jth coset, that is U i plus e j we will obtain the result, which says that the syndrome is equal to the syndrome of the coset leader.

Now, let us assume that the transmitted codeword gets corrupted by additive error patterns on the channel. And we assume that the error patterns with low weight have high probability of occurrence, than the error patterns of higher weights.

So, with this reasonable model we should choose the coset leader at any stage of standard array formation that has minimum weight. And therefore, the coset leaders are also called as correctable error patterns. So, your U 1 which is all 0 codeword could be considered as e 1 error pattern which is all 0 either error pattern.

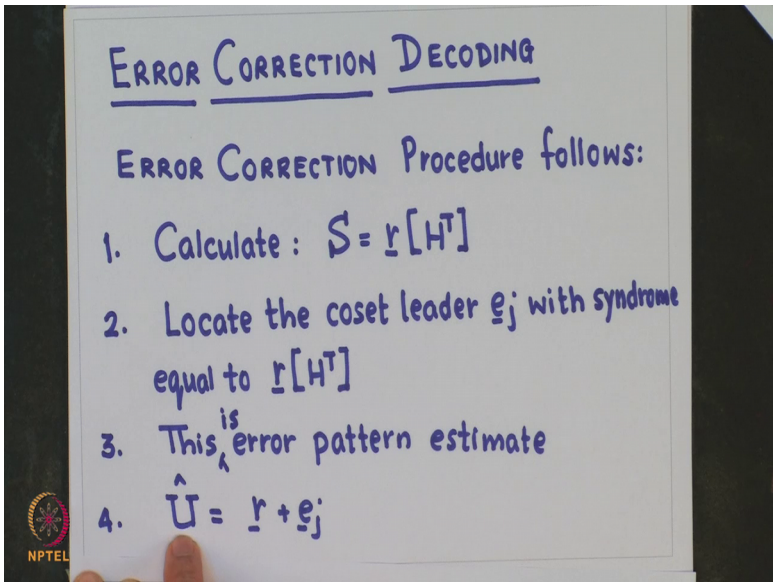
(Refer Slide Time: 07:59)



So, each coset member has the same syndrome and the syndrome for each coset is different from that of any other coset in the code.

This is because of the property of the parity check matrix which we have studied earlier. So, it is the syndrome which is used to estimate the error pattern and the syndrome of the coset leader or the correctable error pattern is sufficient to decide the syndrome for the complete coset to which that error pattern or coset leader belongs.

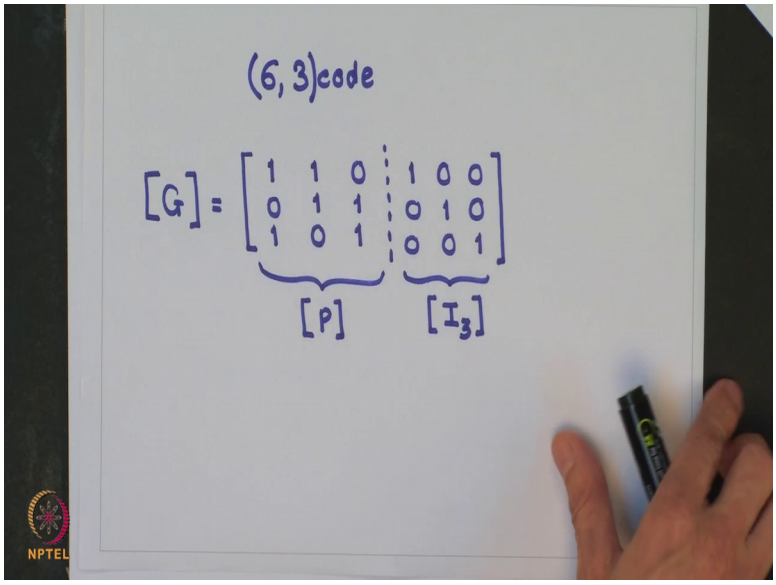
(Refer Slide Time: 08:43)



So, based on these discussions we can give the algorithm for error correction decoding as follows. So, calculate the syndrome based on the received codeword by multiplying the receive code vector the transpose of the parity check matrix, then locate the coset leader e j with the syndrome equal to this syndrome and then, this forms the error pattern estimate and from this error pattern estimate we get the estimate of the transmitted codeword by adding it to the received code word.

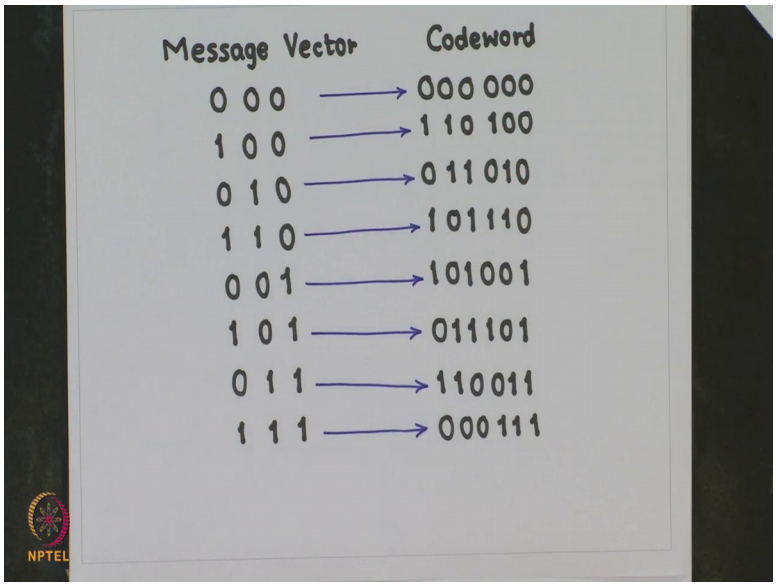
Now, let us take an previous example of 6 3 code for which the generator matrix is shown here.

(Refer Slide Time: 09:38)



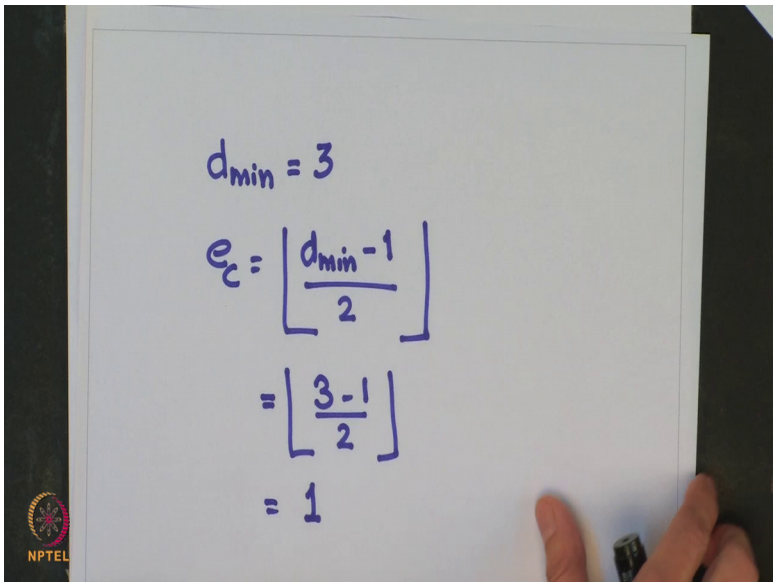
For this generator matrix the message vectors and the codewords are shown here.

(Refer Slide Time: 09:50)



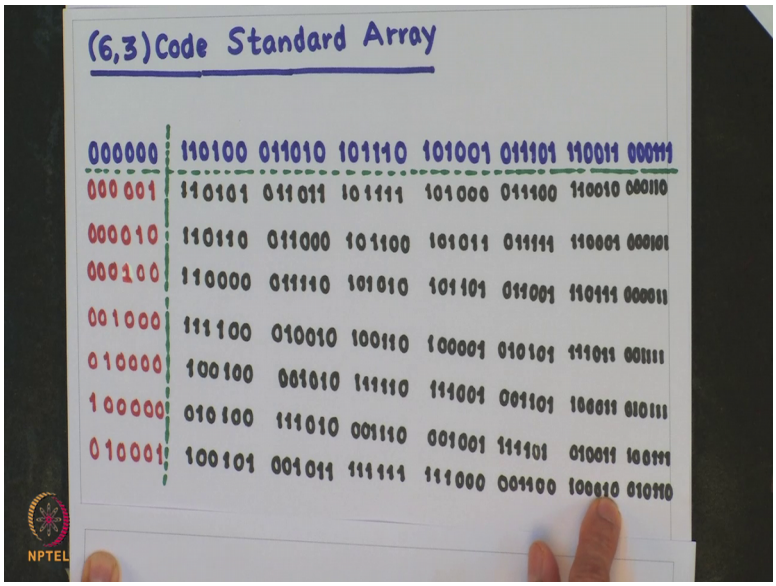
And we see from the code that the minimum distance of this code which is the minimum weight of the code is 3; and from this we can conclude that the error correcting capability of this code should be equal to 3.

(Refer Slide Time: 10:17)



And now let us construct the standard array for this 6 comma 3 code.

(Refer Slide Time: 10:26)



So, the first row will consist of all the codewords in this code starting with all 0 code word. And then we choose the next row coset leader as that 6 tuples which has the minimum weight. And in this case we can see that we can choose this 0 0 0 0 0 1 as the coset leader, which has the minimum weight of 1.

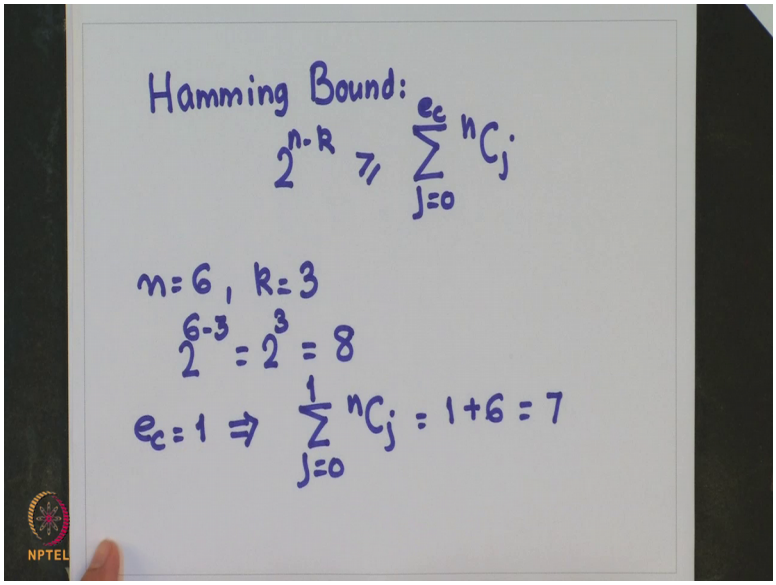
And using this coset leader, we can complete the coset by adding the coset leader to each of the codewords to form the elements in this coset. Having done this, we choose the third row coset leader as that is its tuple which is not there in this 2 rows and also it has a minimum weight. Now it is easy to confirm that this will serve as a coset leader and with this coset leader we complete the coset. So, we proceed this way till we reach up to the 7th coset or 7th row where we have exhausted all one bit pattern.

Now, in this 6 3 standard array, we will have number of cosets which will be 2 raised to 6 minus 3 which is equal to 8 and number of columns will be 8. So, we have 1 more coset which is remaining. So, in this case now we will choose another coset leader for the last row such that it is not existing in this 7 cosets and it has a minimum weight.

So, it is not difficult to verify that, this will serve as a coset leader because this is not existing in the above 7 cosets. And using this coset leader we complete the last coset. So, we see that at this stage of choosing the coset leader, we could have even chosen either this or this as a coset leaders because both this have weight 2.

Now, for this 6 3 code, it we can also see that it satisfies the hamming bound with n equal to 6, k equal to 3 and e c is equal to 1 correct.

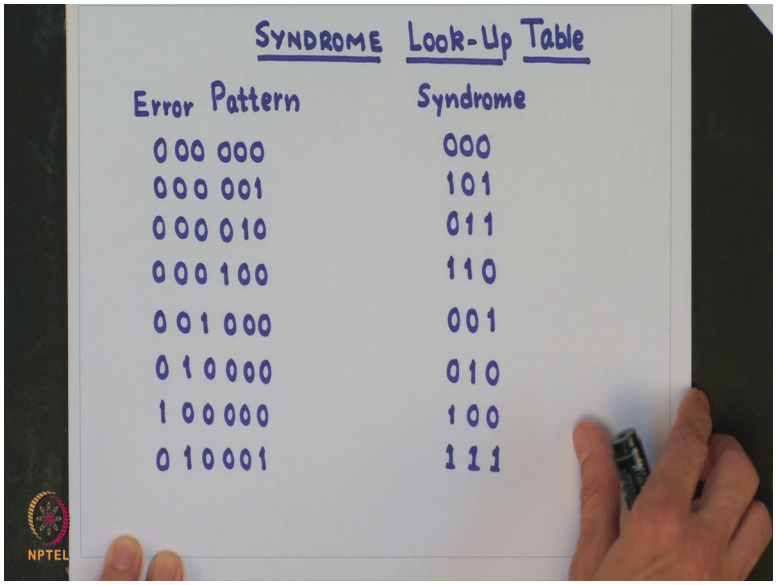
(Refer Slide Time: 13:35)



So, we get 8 is greater than 7. So, it satisfies the hamming bound. And now for this error patterns in the standard array, we can find out the syndrome and those syndromes are listed in the table here.

So, in a practical application the received code words syndrome is evaluated and say for example, we get the syndrome to be 0 01.

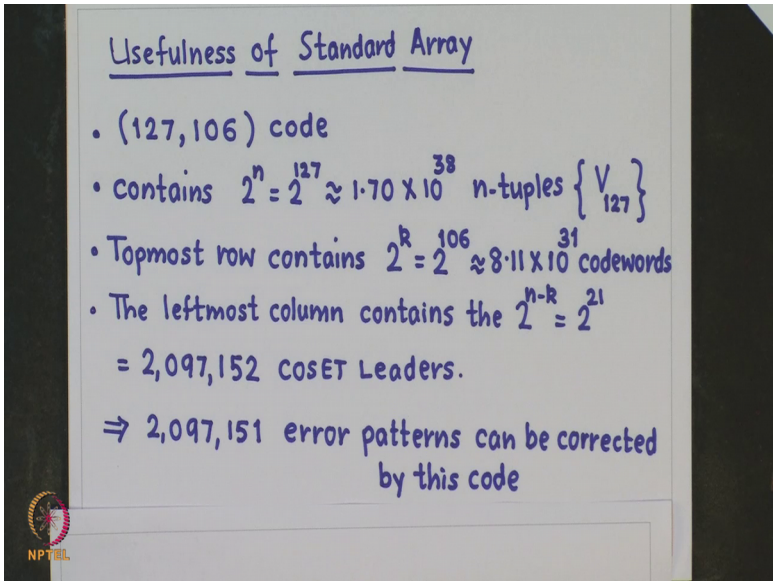
(Refer Slide Time: 14:03)



So, from this table we know that the error pattern is 0 0 1 0 0 0. So, this error pattern is added to the received codeword to obtain the estimate of the transmitted codeword.

Now, the usefulness of the standard array is highlighted, when we use high rate code. For example, Bose Chaudhuri Hocquenghem codes which are popularly known as BCH code which are a generalization of hamming codes that allow multiple error correction.

(Refer Slide Time: 14:57)



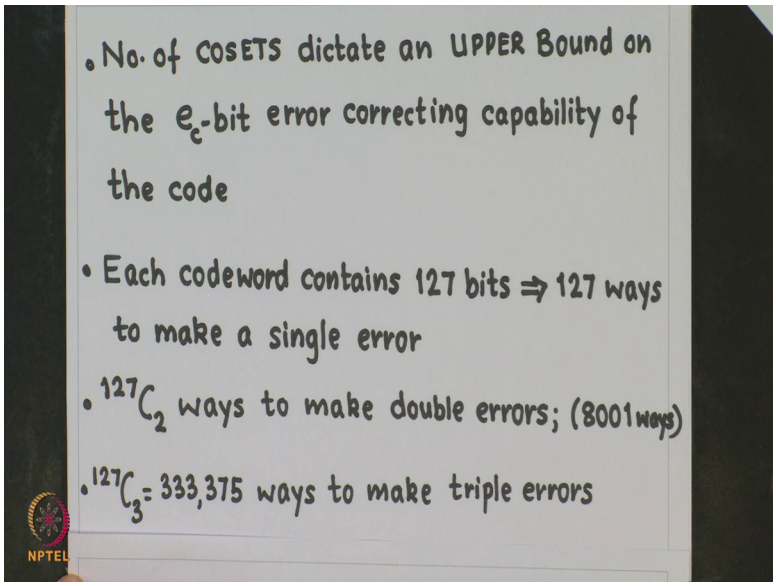
So, let us take 127 comma 1 0 6 BCH code.

So, to find the standard array for this code would be very difficult, but it is not necessary for us to find the standard array to understand the error correcting capability of this code. So, let us follow the following procedure to understand the error correcting capability of this code.

Now, this code would contain 2 raises to n tuples and the topmost would contain 2 raise to k code words. Now the leftmost column will contain 2 raise to 21 coset leaders or we could say error patterns. So, out of this so many coset leaders, the first coset leader which also corresponds to the first code word is all 0 error pattern, in which we are not interested. So, these are the number of error patterns which can be corrected by this code.

Now, we will show that the number of cosets dictate an upper bound on the bit error correcting capability of the code. Now since the code is 127 tuple, this implies that each code word contains 127 bits.

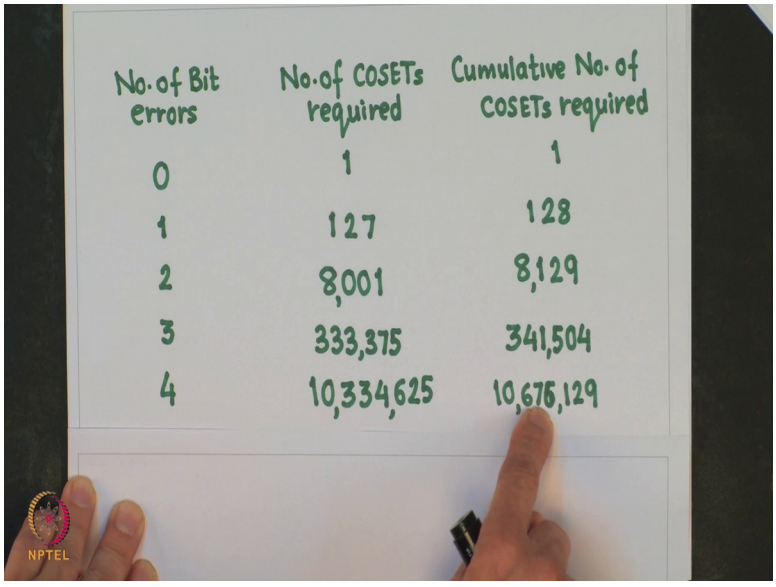
(Refer Slide Time: 16:34)



So, this implies that there are 127 ways to make a single error and similarly there are 127 C 2 ways to make double errors; and there are 127 C 3 number of ways to make triple errors.

So, this calculation is indicated in this table here.

(Refer Slide Time: 17:07)



For the number of bit error 0 obviously, number of coset required is 1 and the cumulative number of coset required is also equal to 1. Now if you are interested in correcting all 1

bit error patterns, then we require 127 cosets because each codeword is 127 tuple. So, this would require 128 cumulative number of cosets.

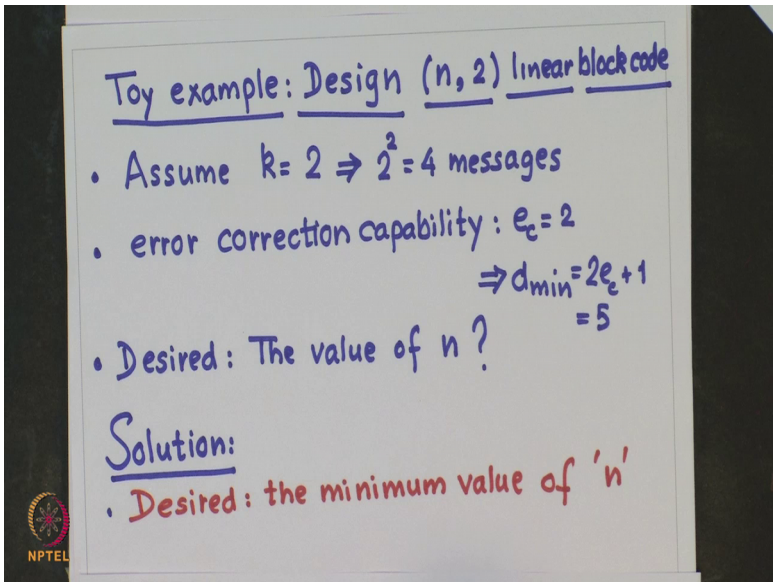
And now we move over to number of bit errors to be equal to 2, for which the number of coset required would be 8001 and the cumulative number of coset required would be given by the summation of this plus sorry summation of this plus 8001. And so many number of cosets are required to correct all the errors up to 2 bits including; obviously, the 1 bit. And now we move over to the number of bit errors to be 3, and we see that these are the number of coset required and this is the cumulative number of cosets required.

So, after exhausting all error patterns up to 3 bit errors as coset leaders to form the coset, still there are unused rows which means that more error correction is possible. But we cannot fit all possible 4 bit error patterns into the first column of the standard array, because the number of coset required for correcting 4 bit errors is indicated by this number in the table, and this is more than what is remaining correct. So, this is also clear from the value here there to correct up to 4 bit error the total number of coset required is this, but what is available is only this many.

So, it cannot correct all 4 bit - error patterns. So, what this says that 127 comma 1 0 6 code has a hamming bound that guarantees the correction up to an including all 3 bit errors.

Now, let us take one toy example for our design of a linear block code.

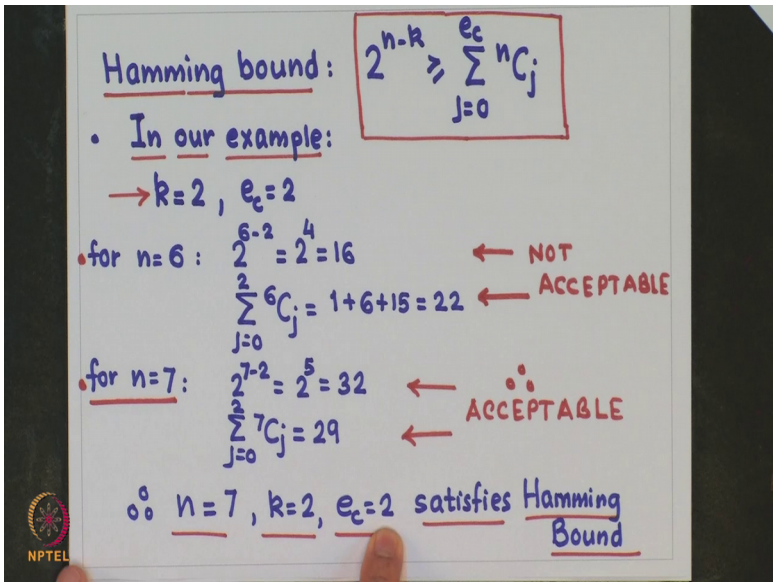
(Refer Slide Time: 20:00)



In this toy example I assume that k is equal to 2 that means, there are 4 messages. So, I have to generate for code words and the I want the error correction capability of this code to be e c equal to 2, which imply the d minimum should be equal to 5 and what is desired is the value of n.

Now, the solution is that, we should try to find out the minimum value of n because by choosing the minimum value of n, we will get better bandwidth of efficiency or spectral efficiency. So, based on this reasoning we can get the value of n from the hamming bound.

(Refer Slide Time: 20:53)

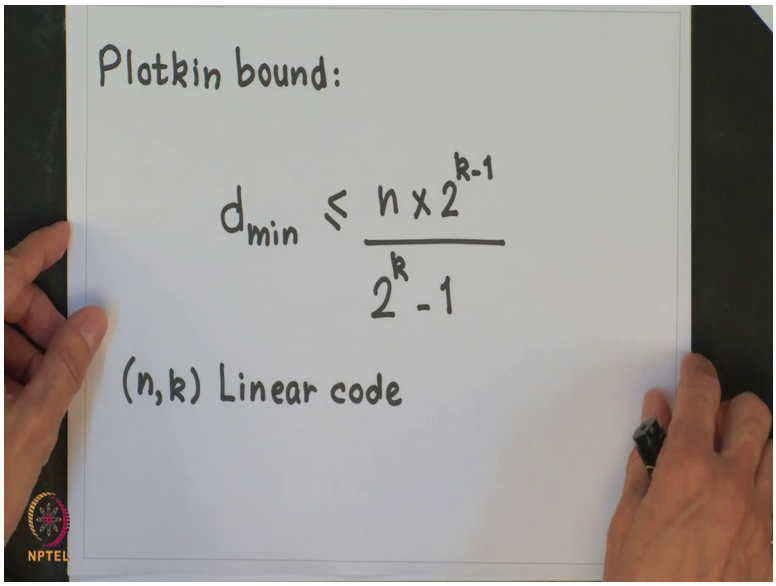


We know the hamming bound has this inequality, for our example k equal to 2 and ec equal to 2 let us choose n equal to 6.

If we choose n equal to 6 and we plug in these values in this inequality, we find that this n equal to 6 is not acceptable because it does not satisfy the hamming bound. And for error correction up to 2 bits its necessary that hamming bound is satisfied. So, let us choose n equal to 7. So, if we choose n equal to 7 we find that the hamming bound inequality is satisfied. So, n equal to 7, k equal to 2, ec equal to 2 satisfies hamming bound and we could have 7 comma 2 has a linear block code.

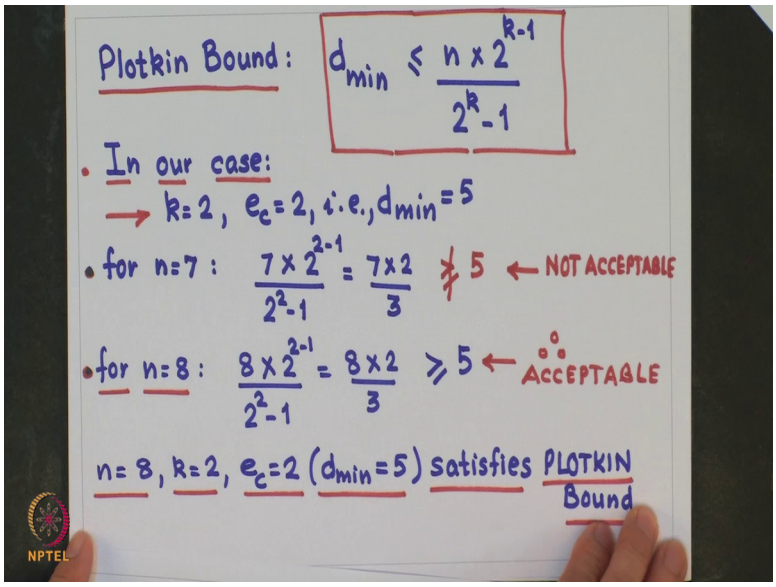
However the dimensions of such a 7 comma 2 code will not meet our stated requirement of ec equal to 2 bit error correction capability and d minimum equal to 5. So, what this implies that there is an existence of another bound, which this code should satisfy.

(Refer Slide Time: 22:19)



And this bound is stated here without proof, which n comma k code should satisfy in order to achieve the desired error correcting capability and this is known as Plotkin bound.

(Refer Slide Time: 22:48)

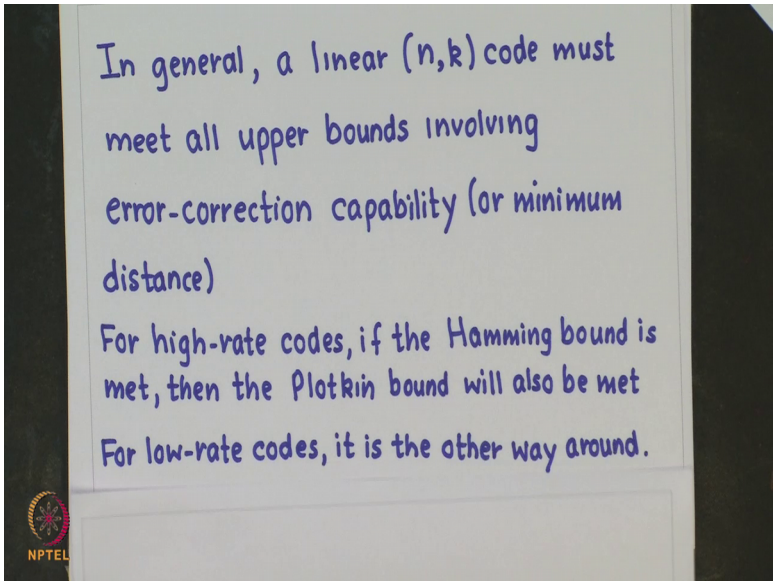


So, for our example if we take this n equal to 7, and substitute in this Plotkin bound, we get this evaluate this for n equal to 7 and we find that this quantity is not greater than 5 because the required d minimum is 5 for ec equal to 2. So, n equal to 7 is not acceptable

because it does not satisfy the Plotkin bound. So, we go for a higher value of n and we choose n is equal to 8 and we see that this satisfies the Plotkin bound.

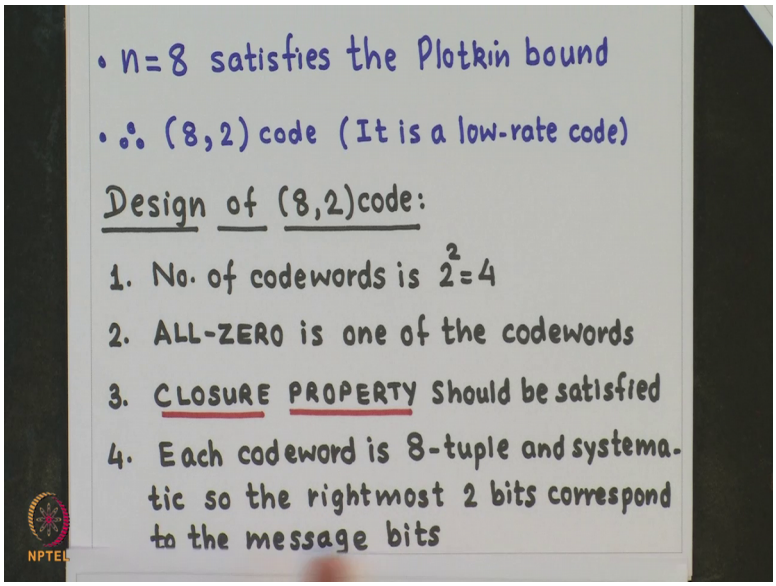
So, now n equal to 8, k equal to 2, e c equal to 2 d minimum equal to 5 satisfies both the Plotkin bound and the hamming bound. So, in general a linear n comma k code must meet all upper bounds involving error correction capability or minimum distance.

(Refer Slide Time: 23:37)



So, for high rate codes if the hamming bound is met, then the Plotkin bound will also be met. So, you can verify this for our example of bch code 127 comma 1 0 6, it satisfies the hamming bound and it satisfies the Plotkin bound. And for a low rate code like the one we design, 8 comma 2 it is the other way round if the Plotkin bound is satisfied then the hamming bound will also be satisfied.

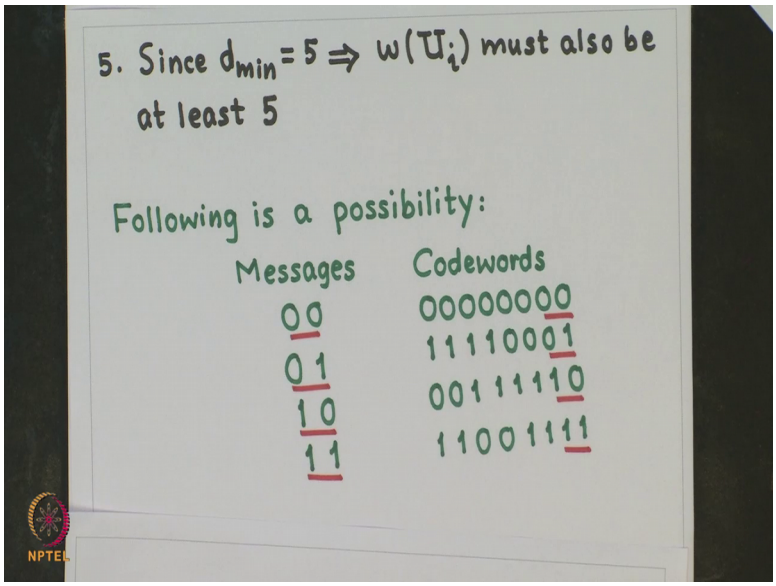
(Refer Slide Time: 24:26)



So, now n equal to 8 satisfies the both Plotkin bound and the hamming bound so, we could freeze on value of n to be 8 and now this will be the acceptable code, which is a really a low rate code and the final solution has to be found out for the code words of this code, and since there is no general procedure to design the code, but we should remember the following points while designing the code and these are as follows.

We require the number of code words to be 4, all 0 is 1 of the code words and the closure property should be satisfied this is an important condition, then each code word is 8 tuple and we want it to be systematic. So, the rightmost 2 bits corresponds to the message bit and since we want d minimum equal to 5, this implies that the weight of the code word in the code must also be at least 5.

(Refer Slide Time: 25:21)



So, following is a possibility for the code words of this 8 comma 2 code. So, d minimum that is the minimum distance of a code decides the error correcting and or error detecting capability of n comma k code, and this code should satisfy both the hamming bound and the Plotkin bound.

With this we come to the end of a study of channel coding for this course, and we also come to an end of our course. Hope that the course material has challenged you at every stage of your learning, and you have benefited out of this course many thanks.

Thank you very much.