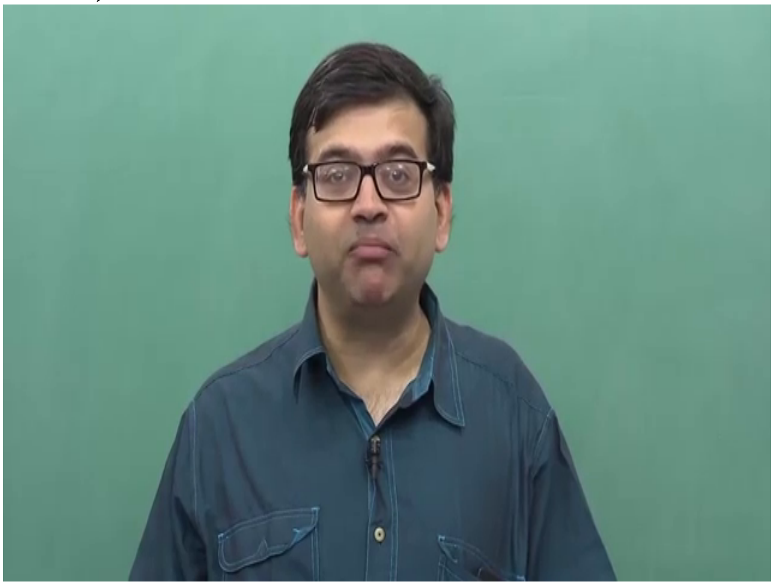
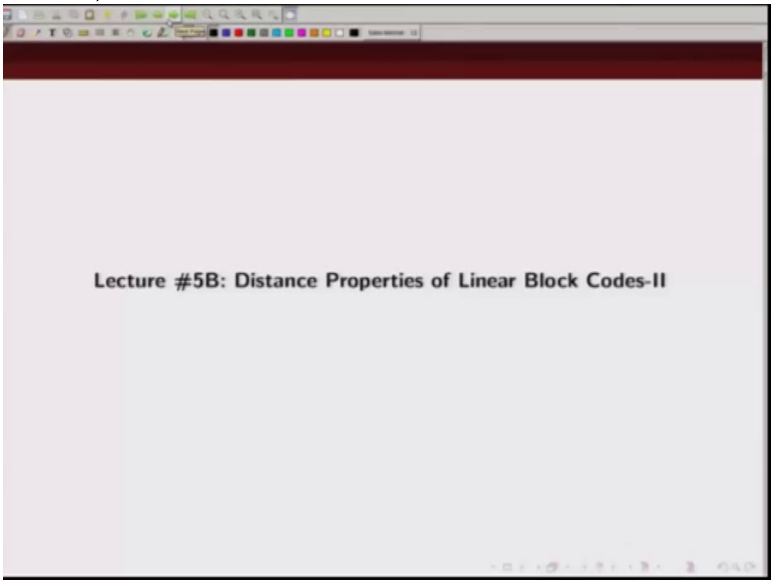
An Introduction to Coding Theory
Professor Adrish Banerji
Department of Electrical Engineering
Indian Institute of Technology, Kanpur
Module 02
Lecture Number 09
Distance Properties of Linear Block Codes-II

(Refer Slide Time 00:18)



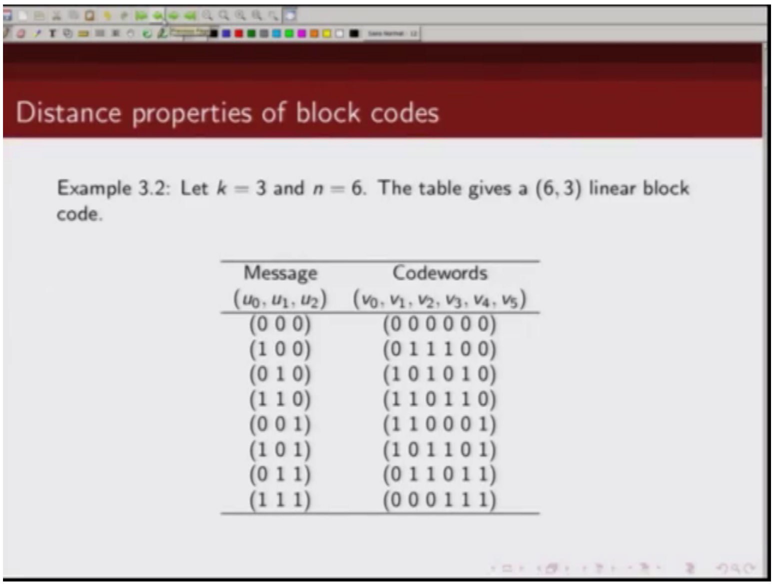
In this lecture we are going to talk about what do we mean by weight distribution of a linear block code and then we are going to talk about how is the error correcting capability and error detecting capability of a linear block code dependent on the minimum distance of a code. So we will continue

(Refer Slide Time 00:39)



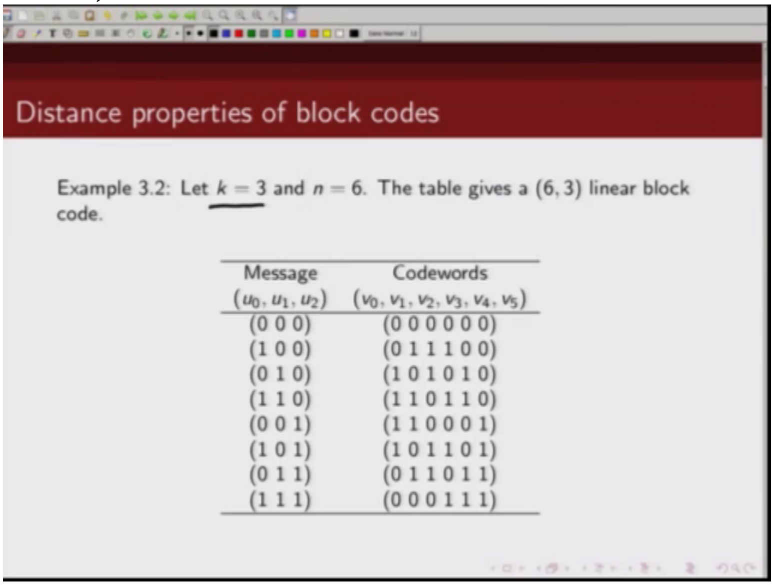
basically our discussion on distance properties that we have started

(Refer Slide Time 00:43)



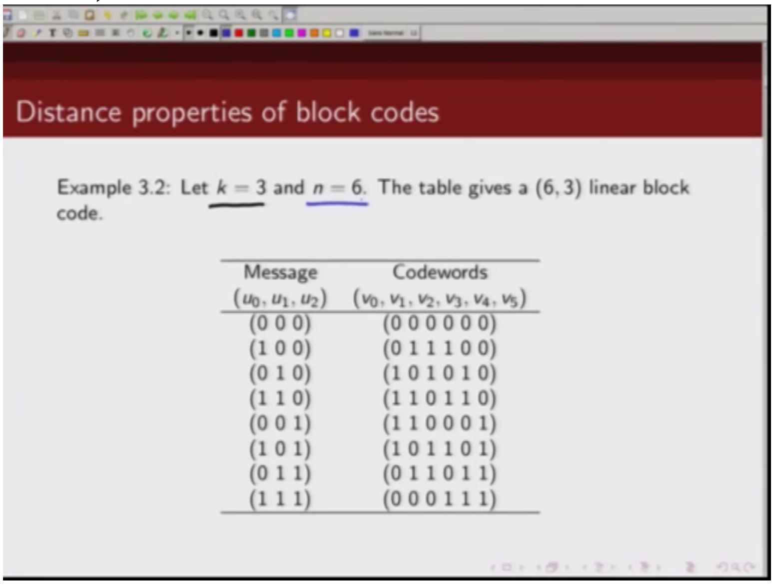
last time. So this is one example of a linear block code where number of information bits is 3

(Refer Slide Time 00:52)



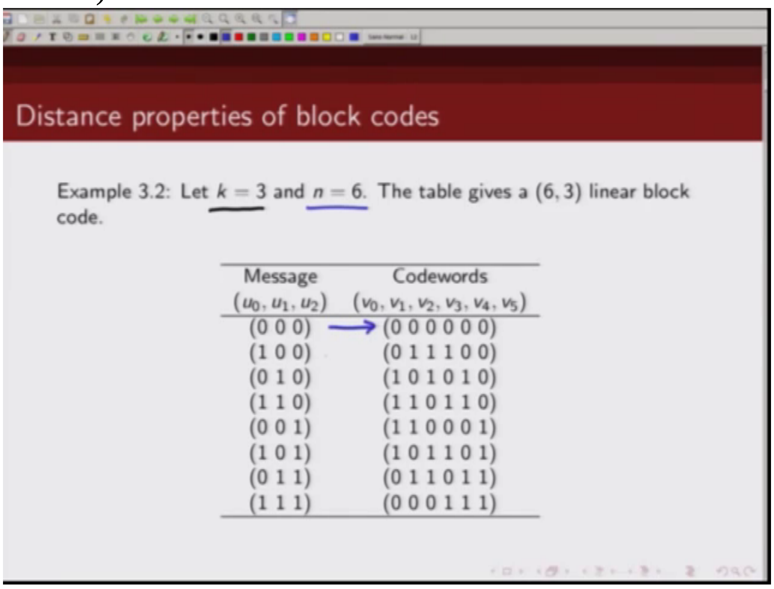
and number of

(Refer Slide Time 00:55)



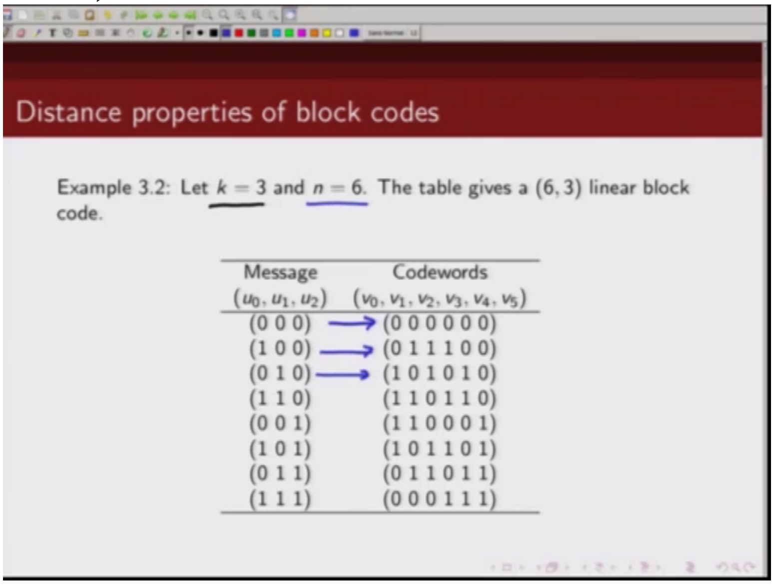
coded bits is 6. This is a list of 2 k codewords which is 8 codewords, message bits and these are their corresponding codewords. So these are the, from 0 0 0 to 1 1 1, these are our 2 k message bits and corresponding to each of our message bits these are the corresponding codewords,

(Refer Slide Time 01:21)



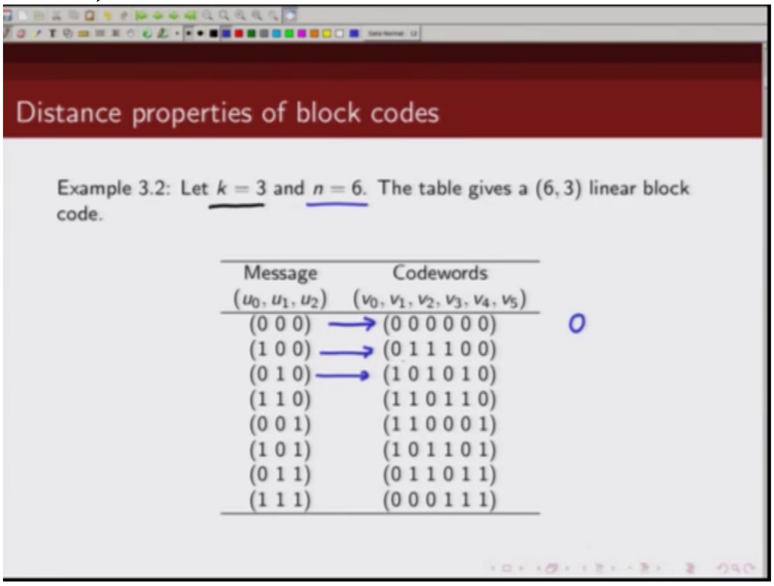
Ok. Now

(Refer Slide Time 01:27)



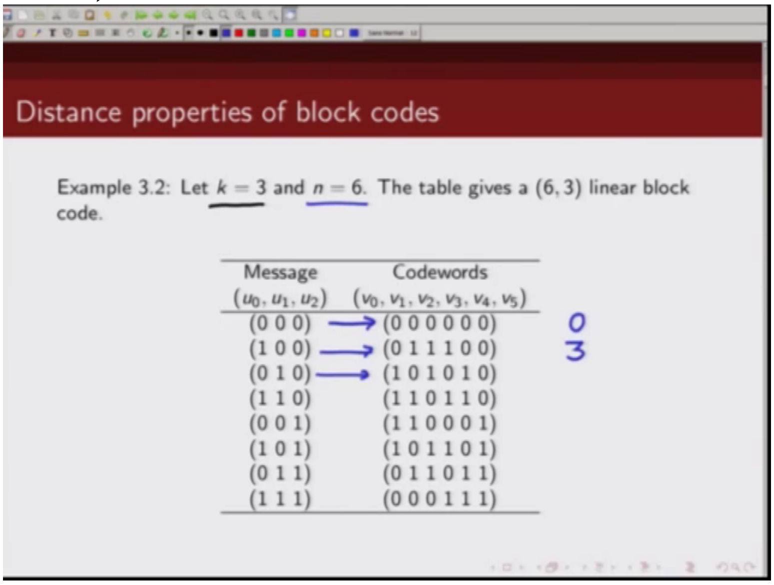
let us look at what is the weight distribution of these codewords. So these codewords, this is all zero codeword, so the weight, Hamming weight for this is basically 0. What about

(Refer Slide Time 01:43)



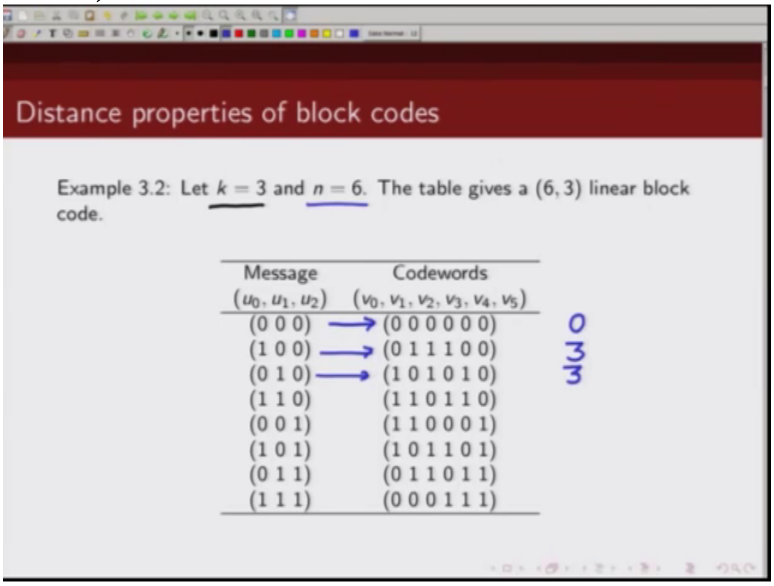
This codeword has 3 1's. So Hamming weight is 3,

(Refer Slide Time 01:51)



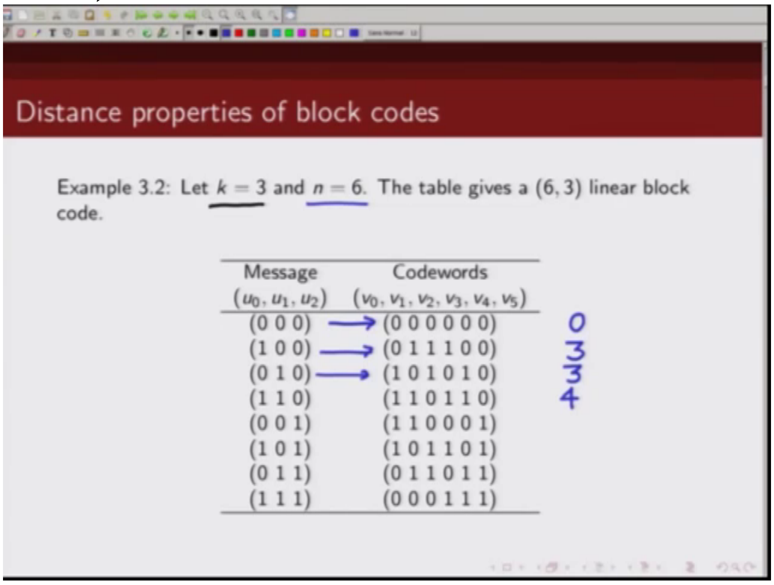
this codeword has three 1's. So its Hamming weight is 3.

(Refer Slide Time 01:57)



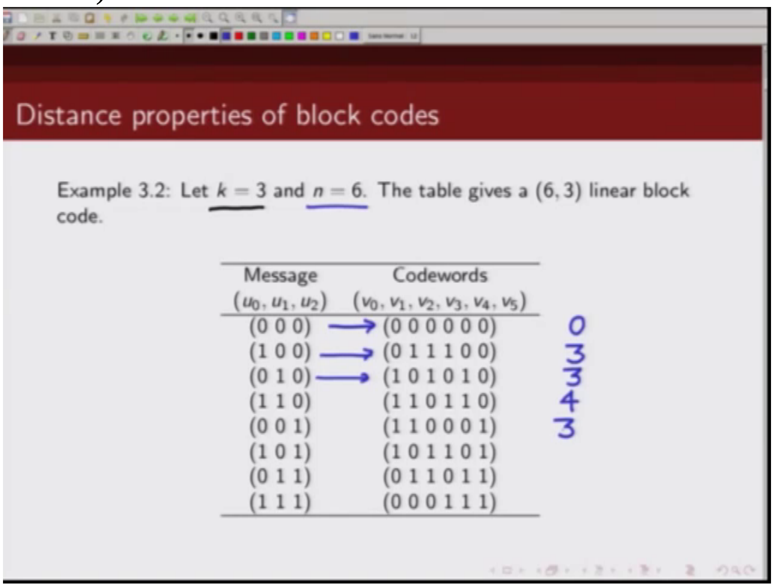
This codeword has four 1's. So the Hamming weight is 4.

(Refer Slide Time 02:04)



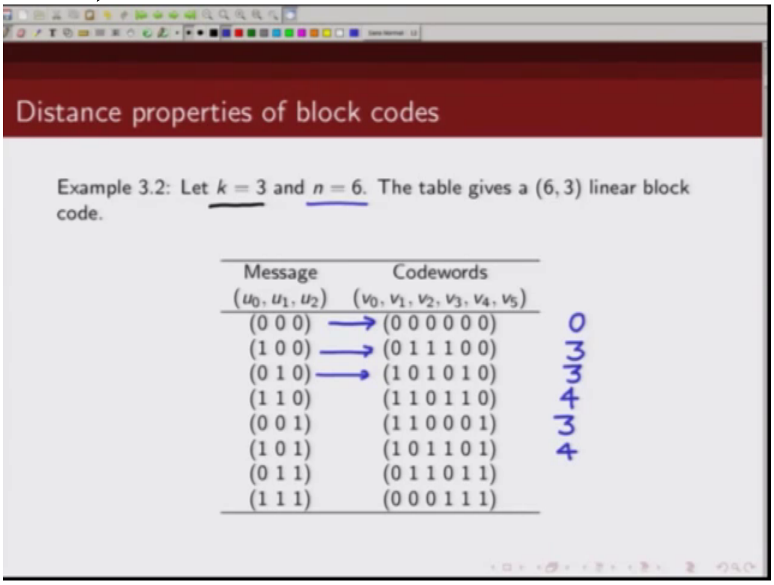
This codeword has three 1's so Hamming weight is 3.

(Refer Slide Time 02:10)



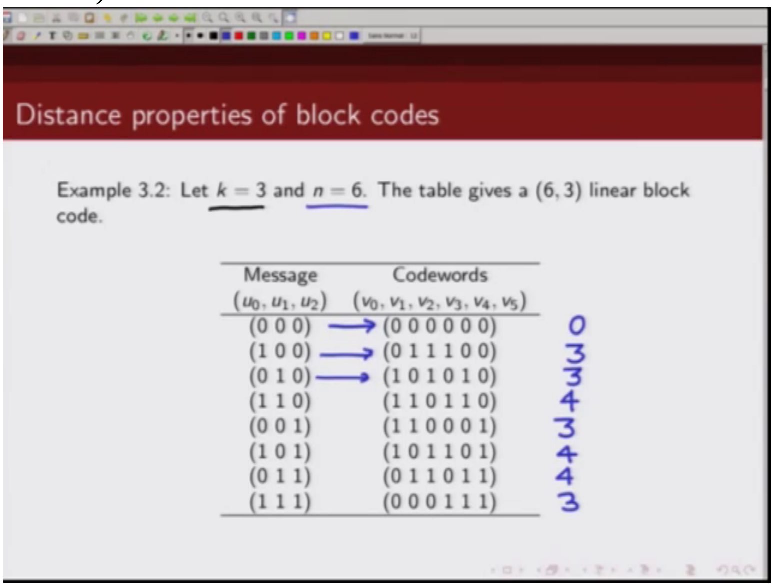
This one similarly has Hamming weight 4,

(Refer Slide Time 02:16)



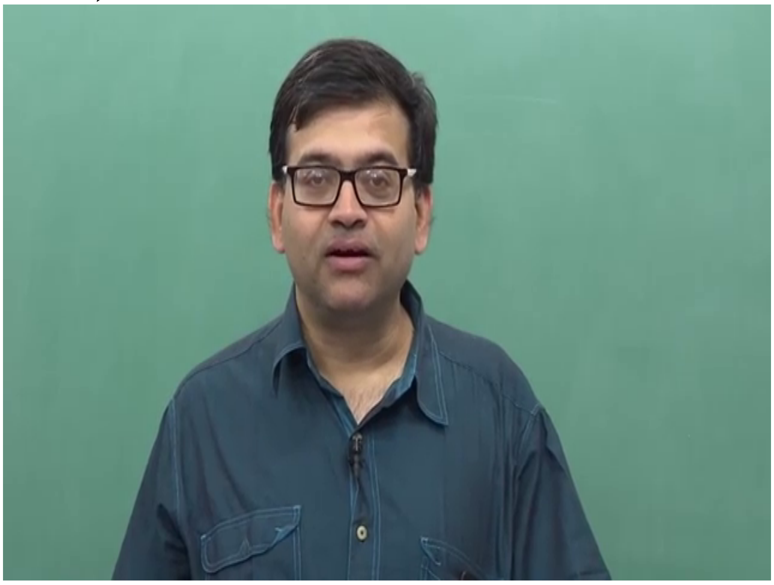
this one Hamming weight 4 and this one has

(Refer Slide Time 02:21)



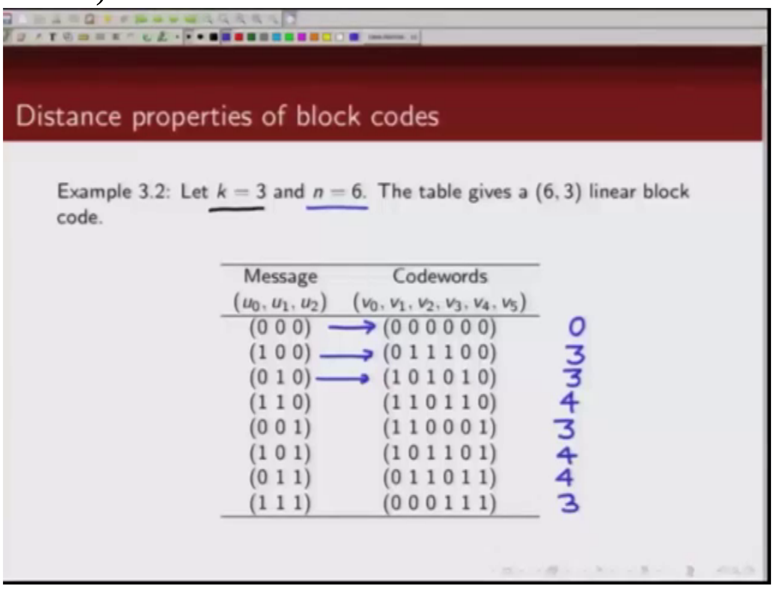
Hamming weight 3.

(Refer Slide Time 02:24)



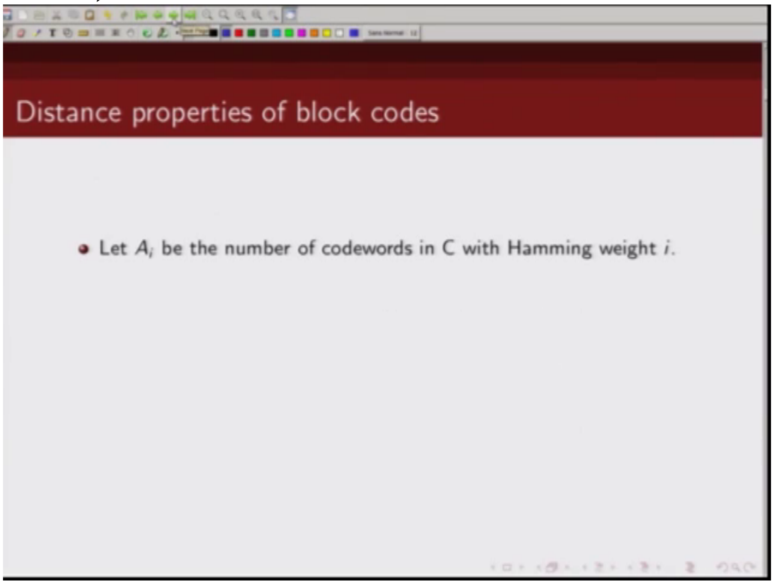
Now what is the minimum distance of the code? As you recall we define the minimum distance of the code as minimum weight

(Refer Slide Time 02:35)



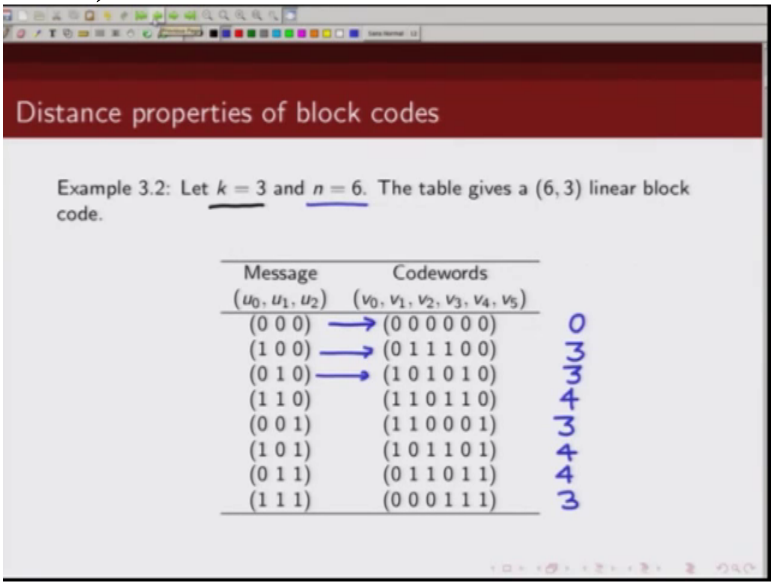
of a non-zero codeword. So what is the minimum weight of the non-zero codeword in this case? Its 3 so minimum distance of this code is 3.

(Refer Slide Time 02:50)



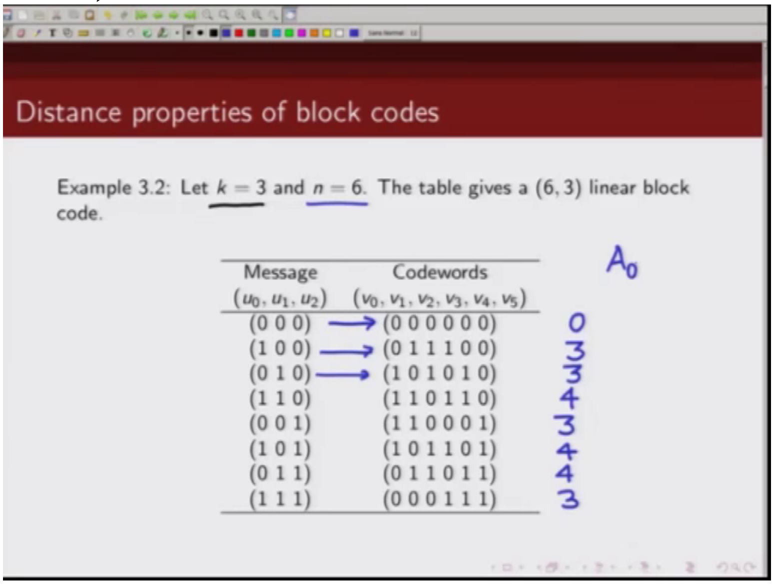
So let a i denotes the number of codewords in C with Hamming weight i. So if

(Refer Slide Time 03:02)



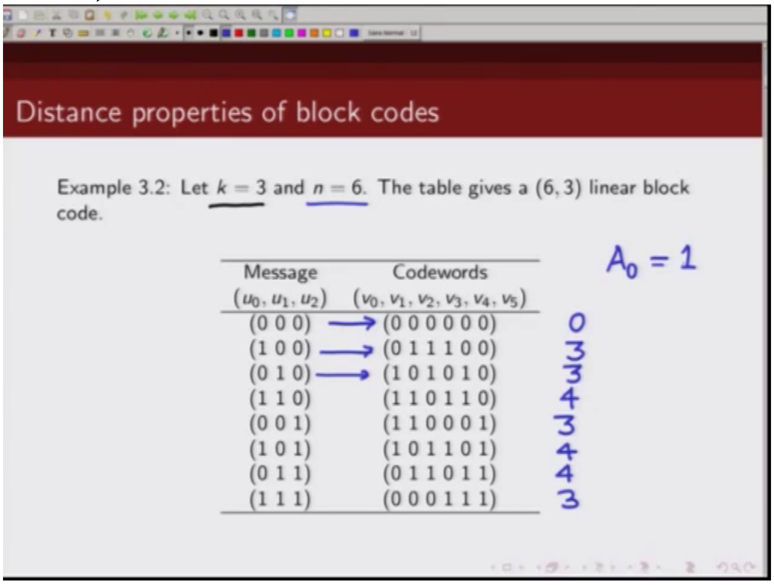
you look here, if I, so I will use a 0 to denote

(Refer Slide Time 03:09)



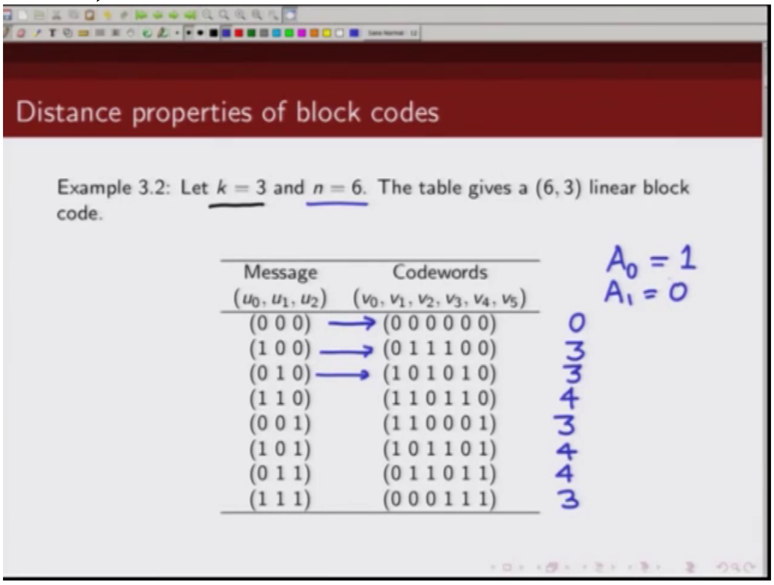
number of codewords which have Hamming weight 0 and that number is 1. Do we have

(Refer Slide Time 03:20)



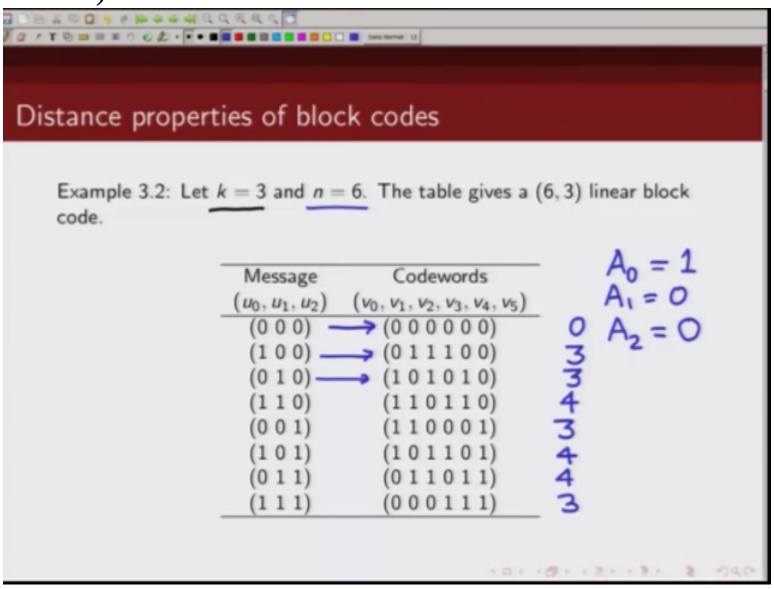
any codeword with Hamming weight 1? No. So a 1 is going to be 0.

(Refer Slide Time 03:28)



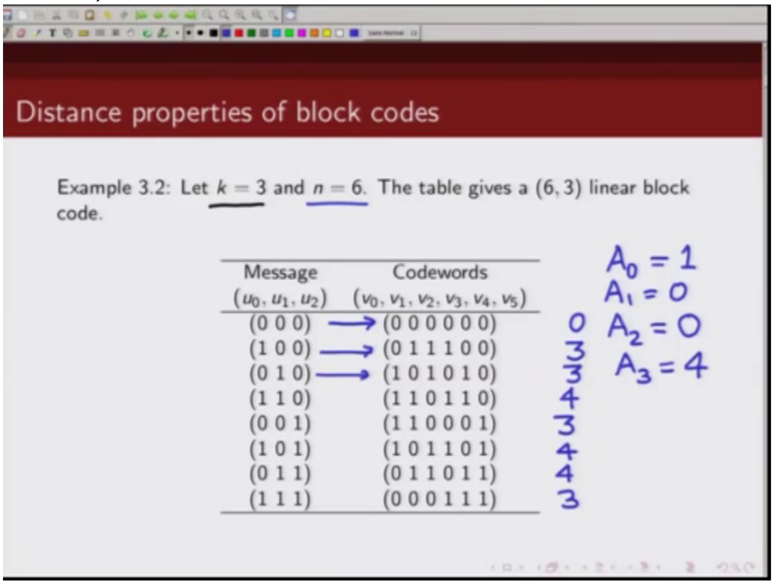
What about a 2? How many codewords we have with Hamming weight 2? Again that's 0.

(Refer Slide Time 03:39)



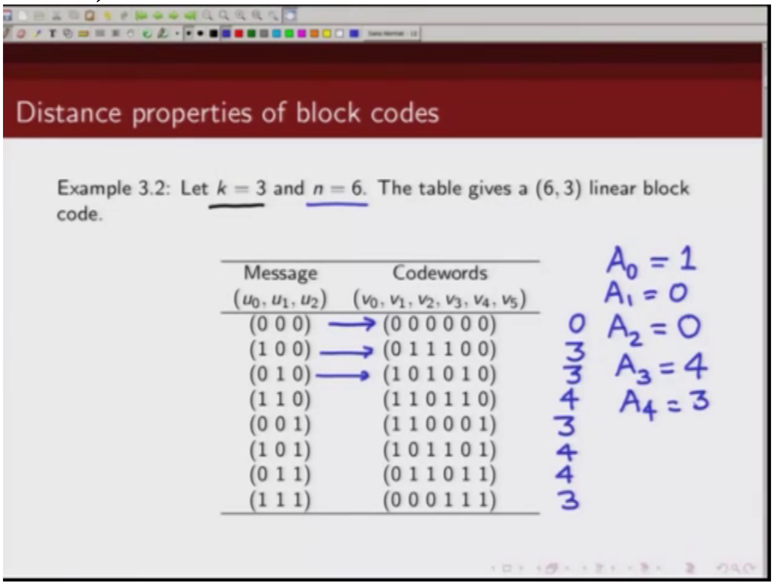
What about a 3? That's basically 1, 2, 3, 4.We have 4 codewords with

(Refer Slide Time 03:51)



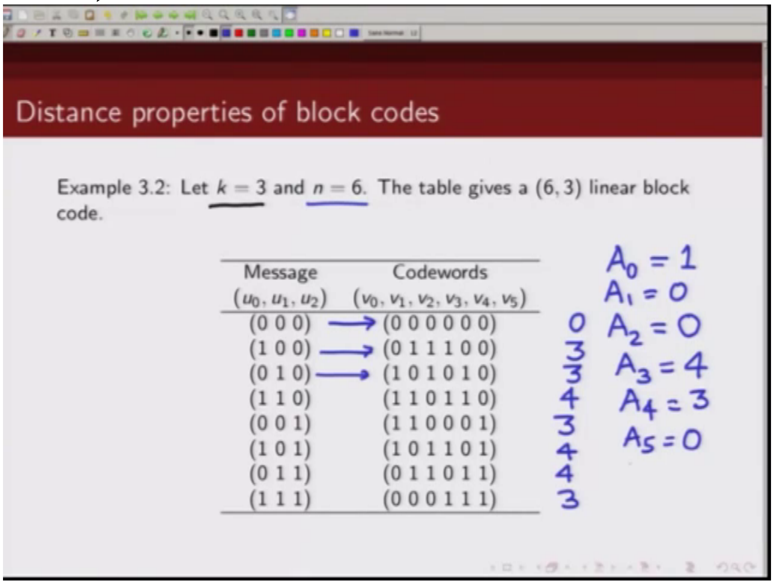
Hamming weight 3, a 4, 1, 2, 3, Ok

(Refer Slide Time 04:04)



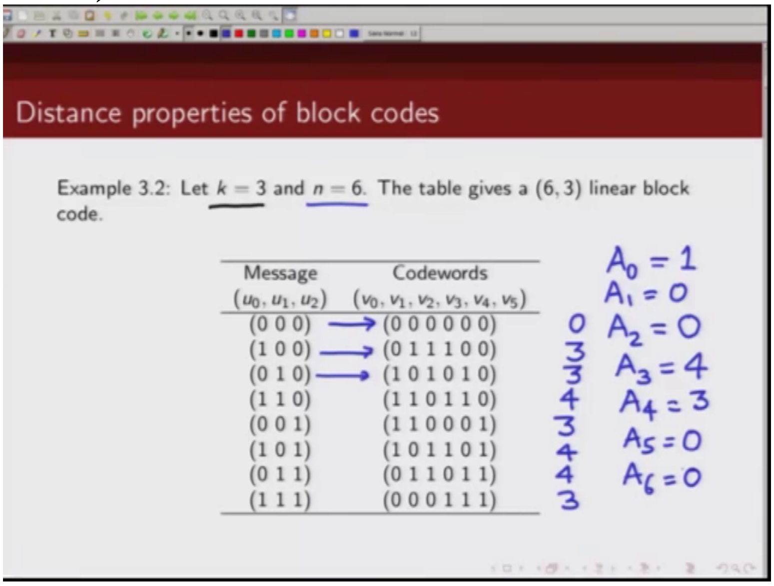
Ok we don't have any codeword with Hamming weight 5 or

(Refer Slide Time 04:10)



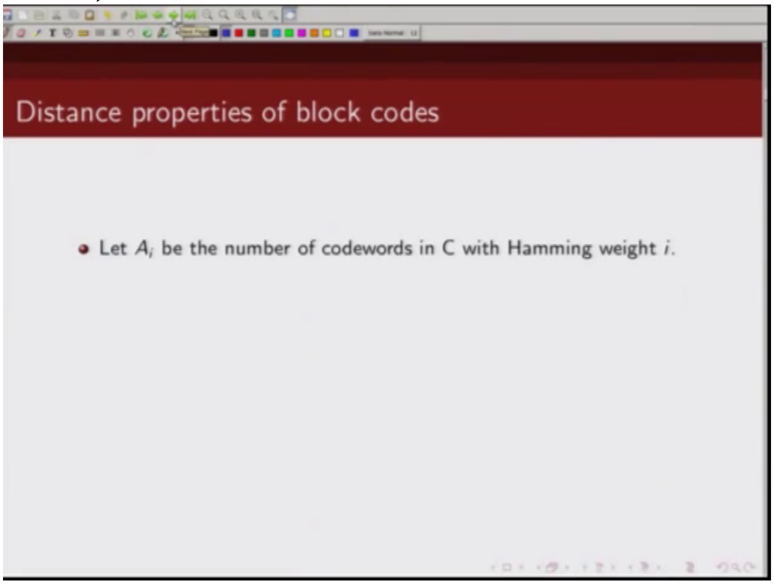
Hamming weight 6.

(Refer Slide Time 04:13)



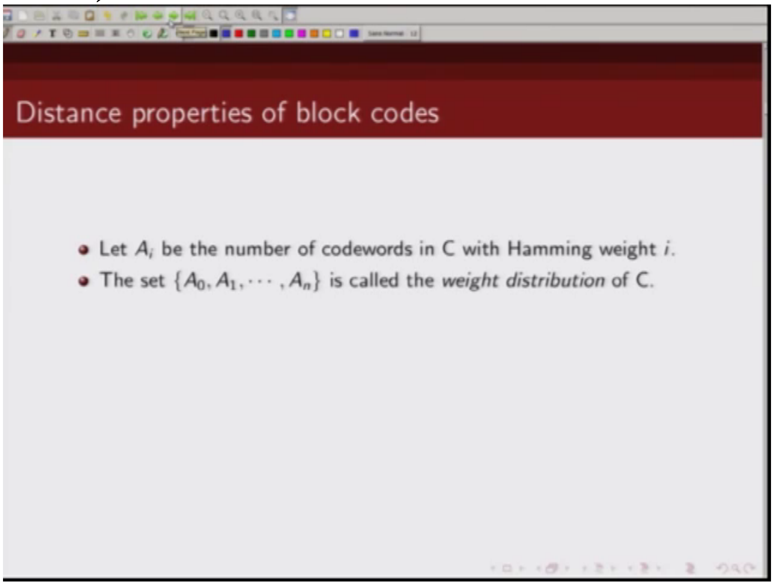
And you can do a quick check, the number of codewords should add up to number of codewords that we have which is 8, 1 plus 4 plus 3, Ok. So we are denoting

(Refer Slide Time 04:24)



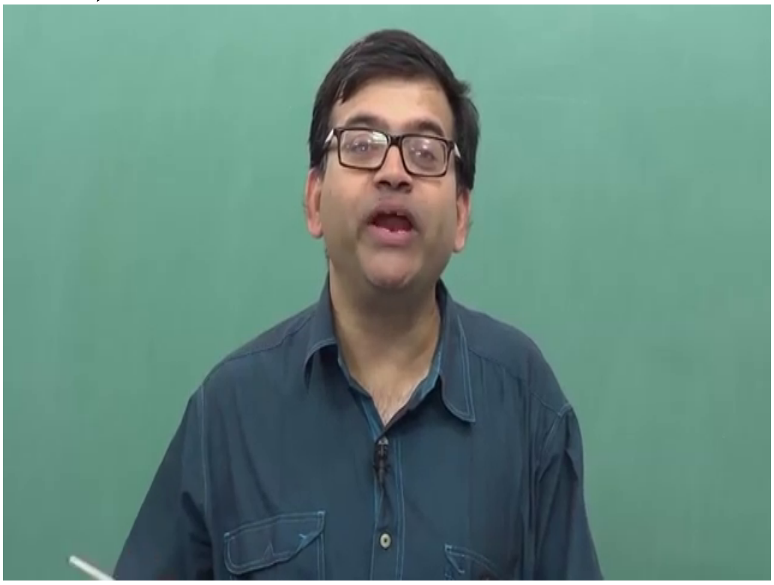
by a i, the number of codewords in this linear block code with Hamming weight i.

(Refer Slide Time 04:32)



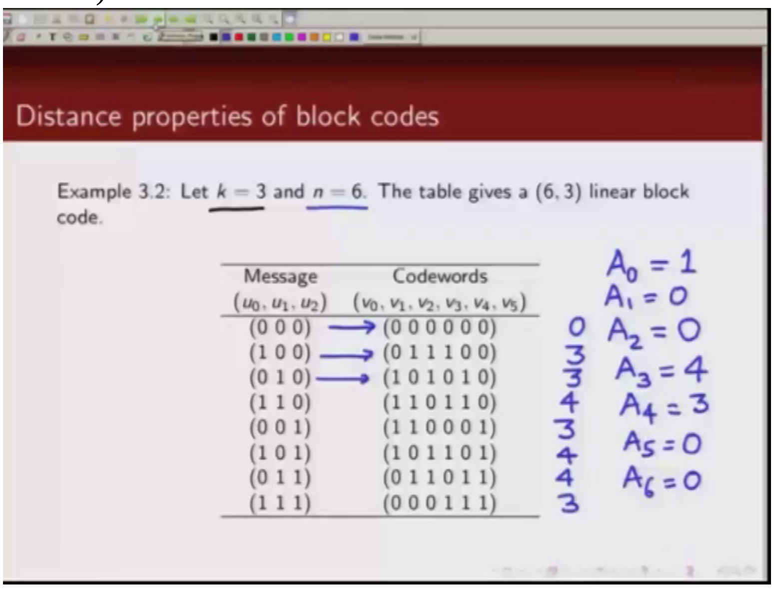
Now this set which describes how many codewords

(Refer Slide Time 04:38)



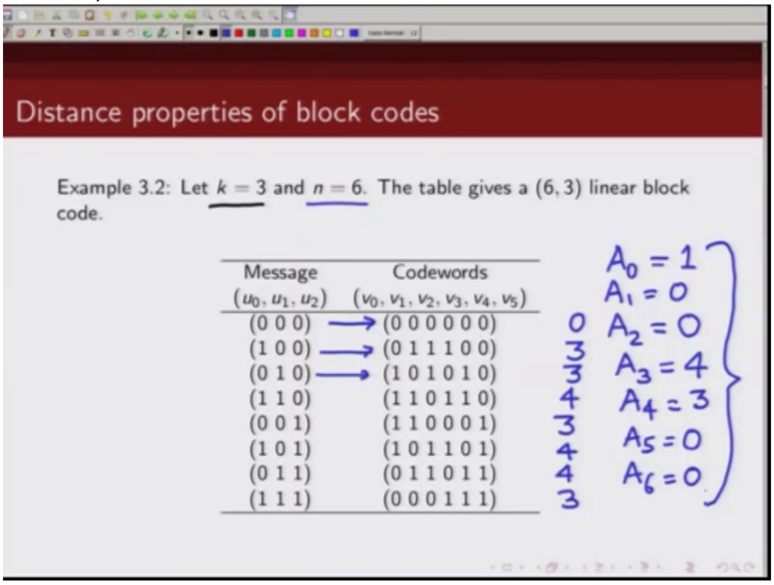
we have of particular weight, this is basically known as weight distribution of a linear block code, see. So for this

(Refer Slide Time 04:49)



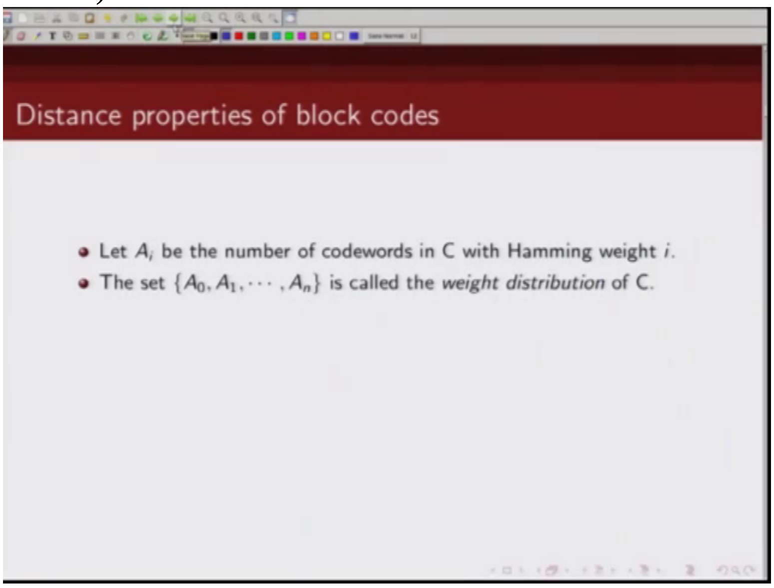
block code, the weight distribution is given by this. This completely specifies the weight distribution

(Refer Slide Time 04:59)



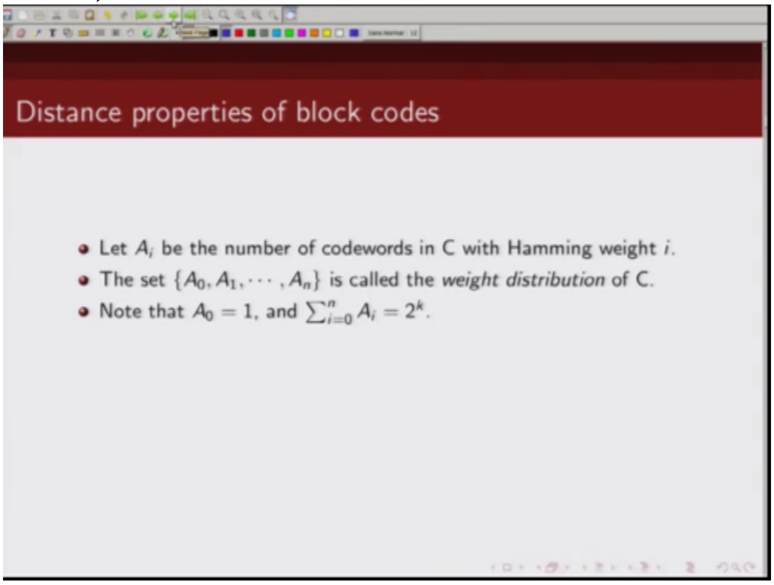
of this particular 6 3 linear block code.

(Refer Slide Time 05:05)



And since we have said

(Refer Slide Time 05:07)

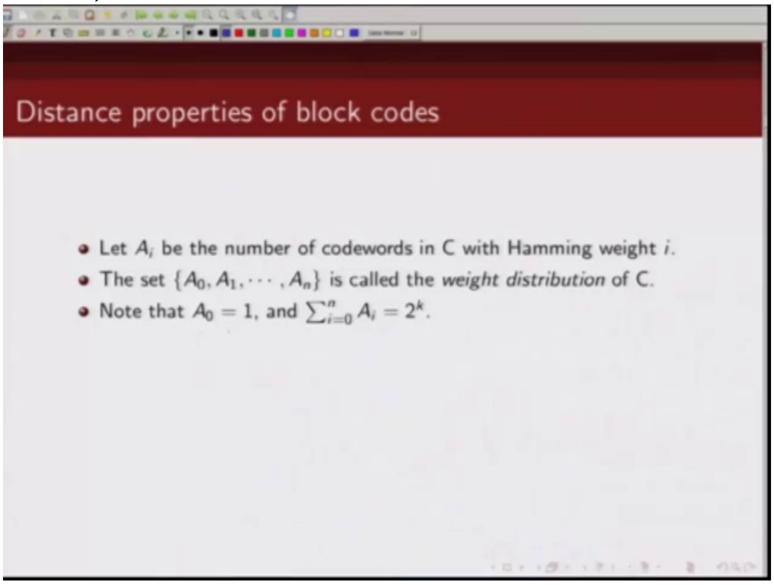


a linear block

(Refer Slide Time 05:08)

code will have an all zero codeword, so a 0 will be

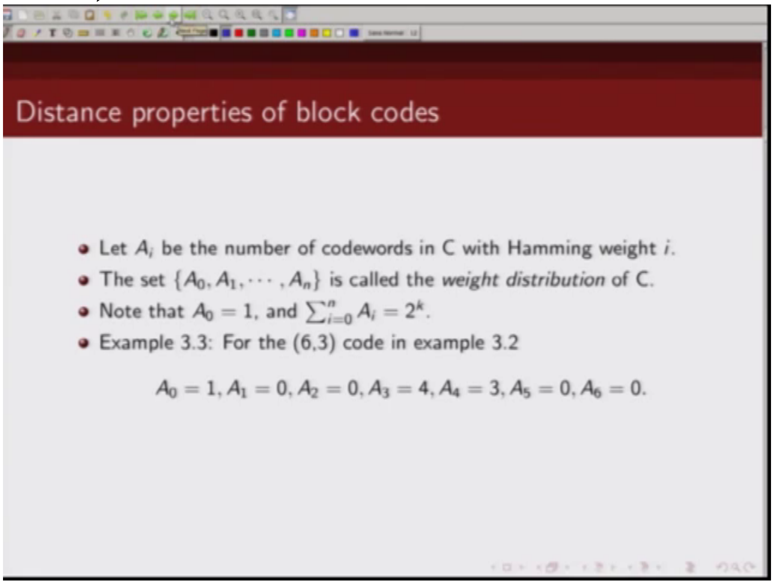
(Refer Slide Time 05:11)



1 and sum of all these codewords, they should all add up to total number of codewords which is 2 to the power k.

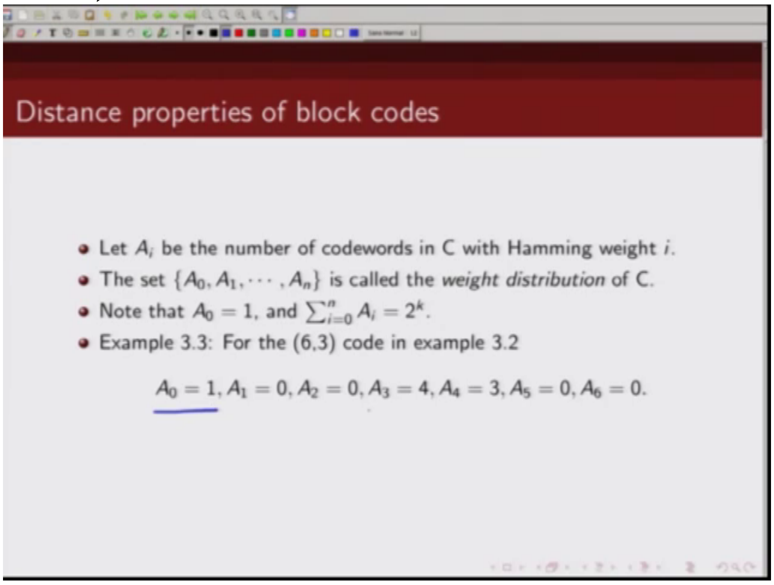
I just worked out

(Refer Slide Time 05:23)



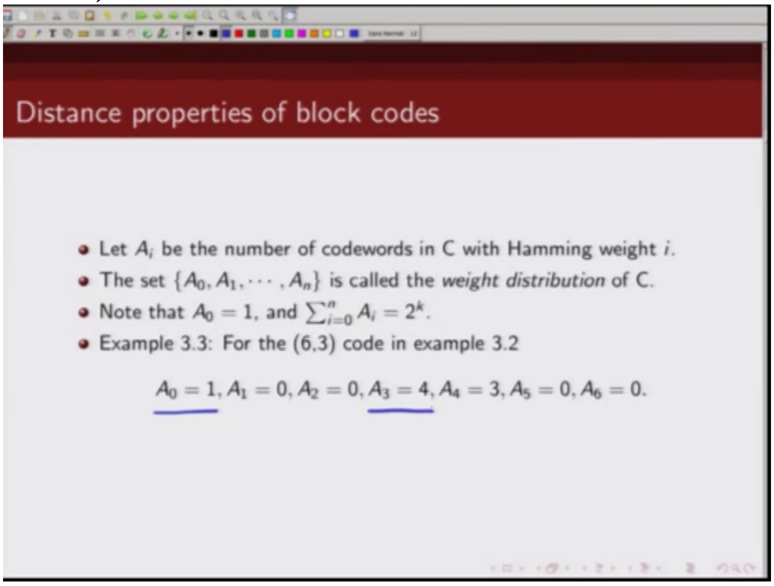
this example for the 6 3 code we have shown in the previous slide and I showed you that in this particular example a 0 is 1, a 3 is

(Refer Slide Time 05:35)



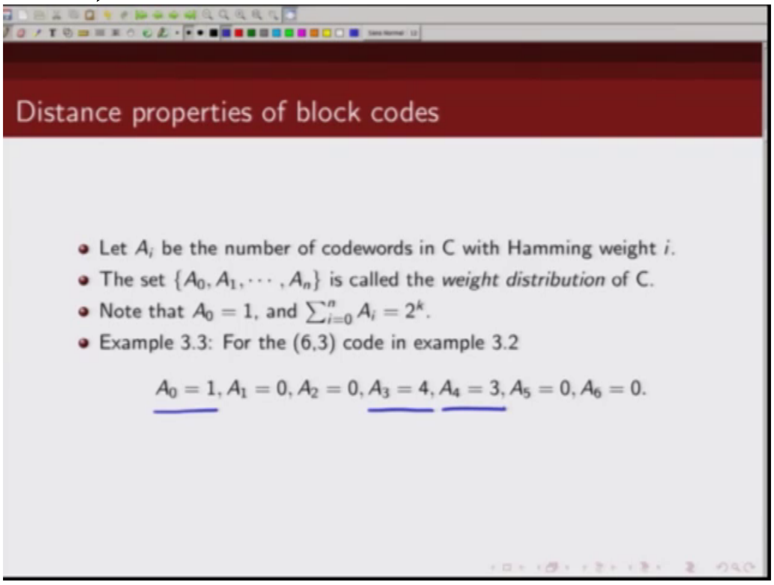
4,

(Refer Slide Time 05:36)



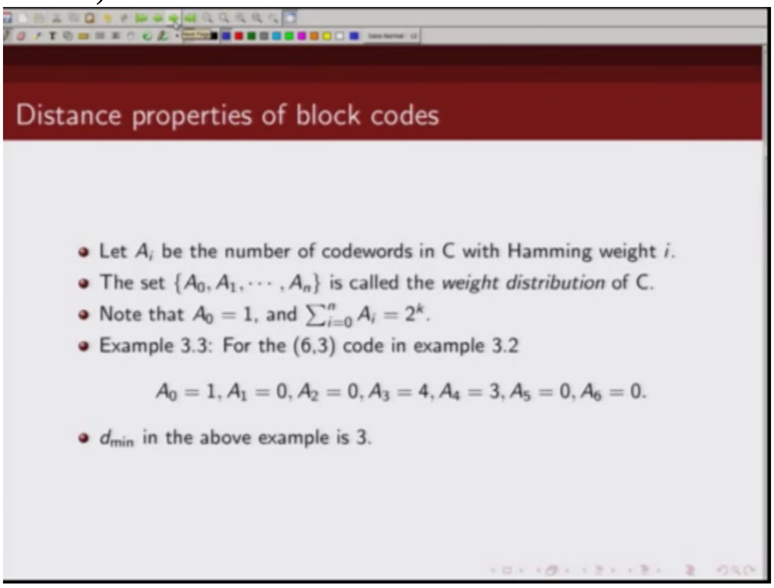
a 4 is 3. Rest all others are

(Refer Slide Time 05:39)



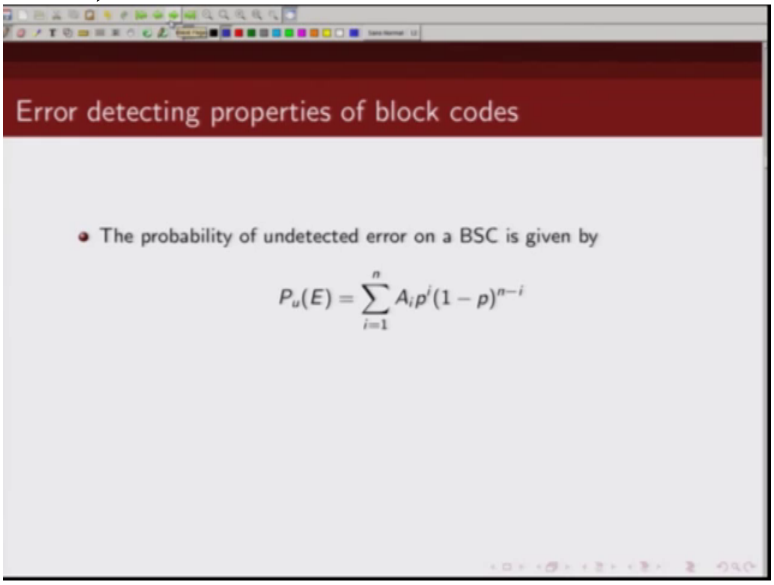
0. And I also showed

(Refer Slide Time 05:43)



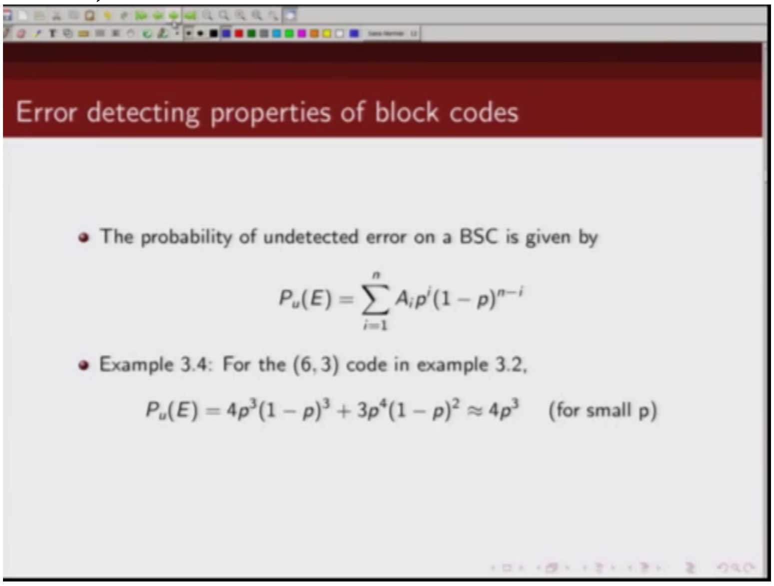
you that the minimum distance of this code is 3 because minimum weight of a non-zero codeword in this example is 3. Now the probability of

(Refer Slide Time 05:58)



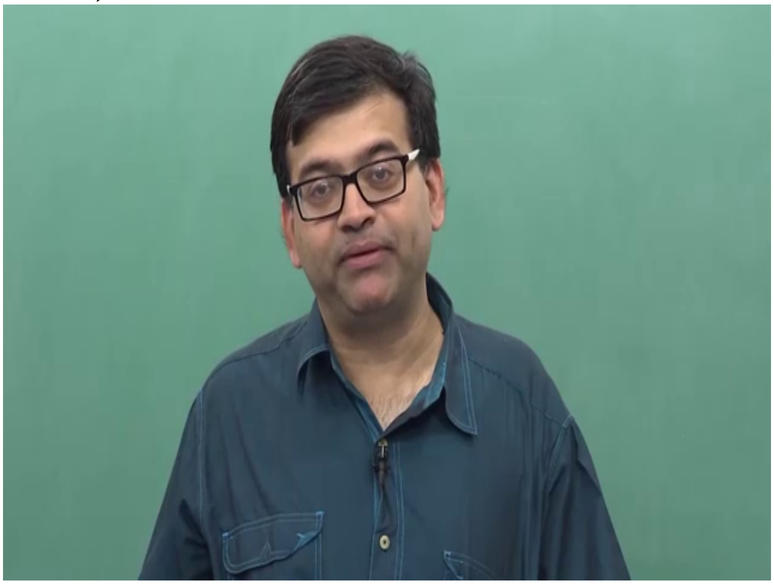
undetected error for a linear block code over a binary symmetric channel is basically related to the weight distribution of the code.

(Refer Slide Time 06:10)



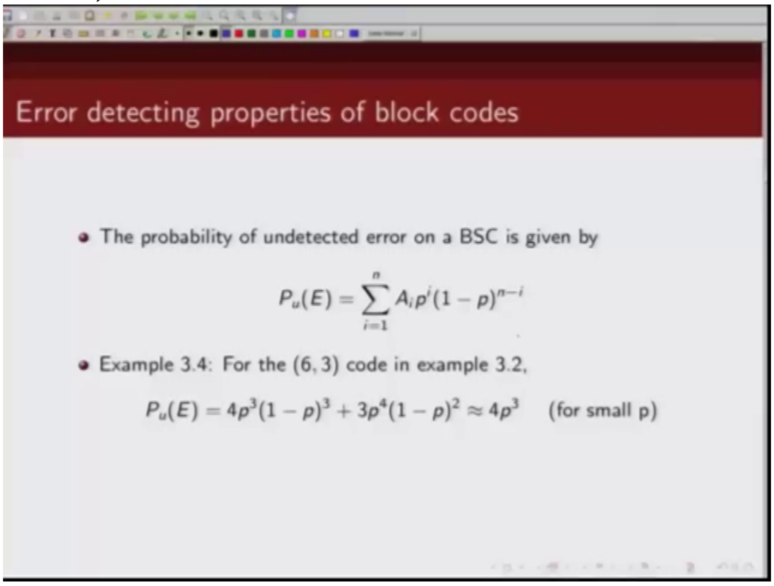
So for a 6 3 linear block code and, so when does an, when does a undetected error happens?

(Refer Slide Time 06:22)



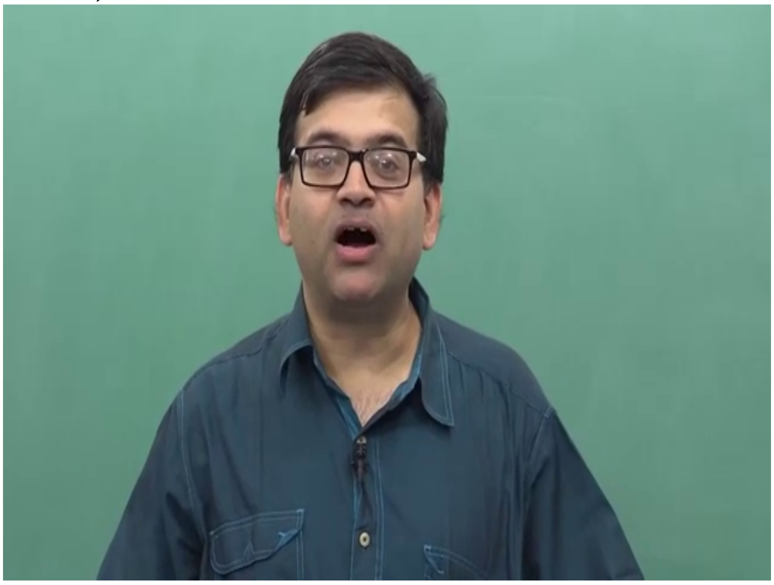
An undetected error happens if, let's say you send one particular codeword and at the receiver you receive some other codeword. So without loss of generality let's assume that we sent a all zero codeword. And at the receiver you received any other non-zero codeword. So if I send an all zero codeword

(Refer Slide Time 06:52)



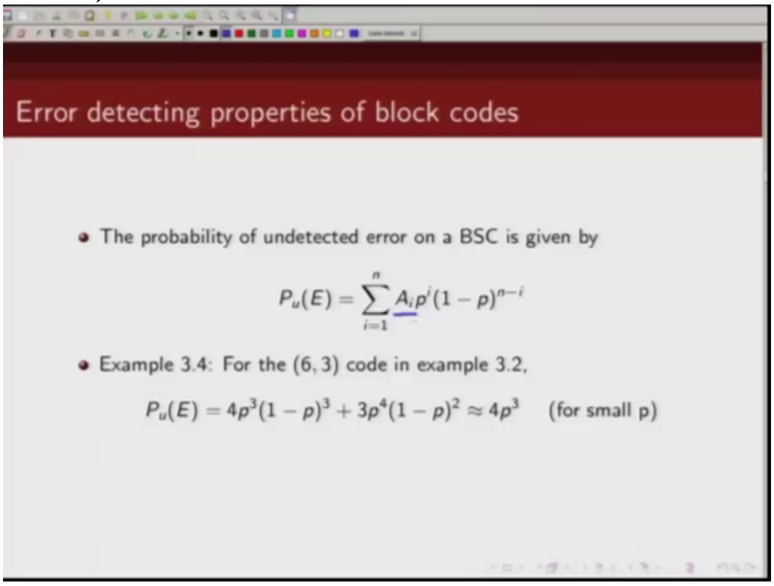
at the transmitter and at the receiver you receive any other non-zero codeword then that will be the case of undetected error. So you can see basically, that's why I have written it as, so what is the probability, when you are sending an all zero codeword, what is the probability of getting another codeword of weight a i or weight i? What is the probability that, when I am sending an all zero codeword and you receive a codeword

(Refer Slide Time 07:32)



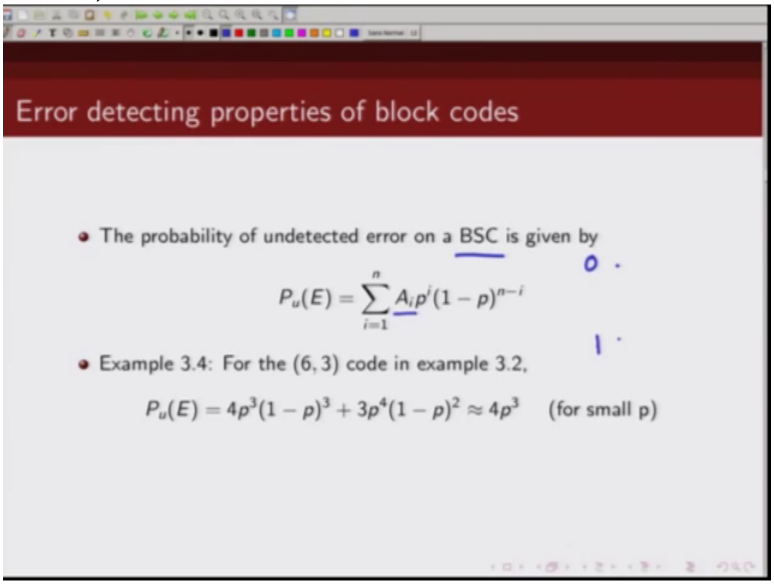
which has weight i? Now that probability is given by,

(Refer Slide Time 07:38)



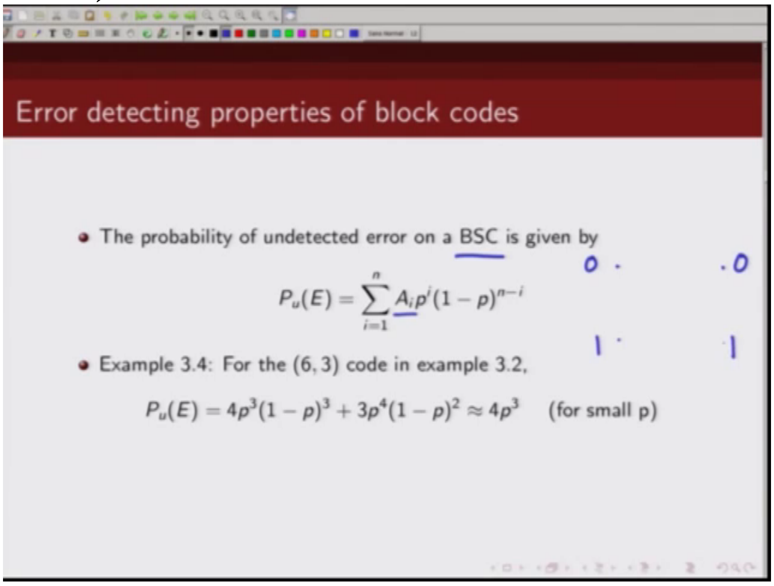
since we are considering a binary symmetric channel, now recall what happens in binary symmetric channel, two inputs 0 and 1,

(Refer Slide Time 07:49)



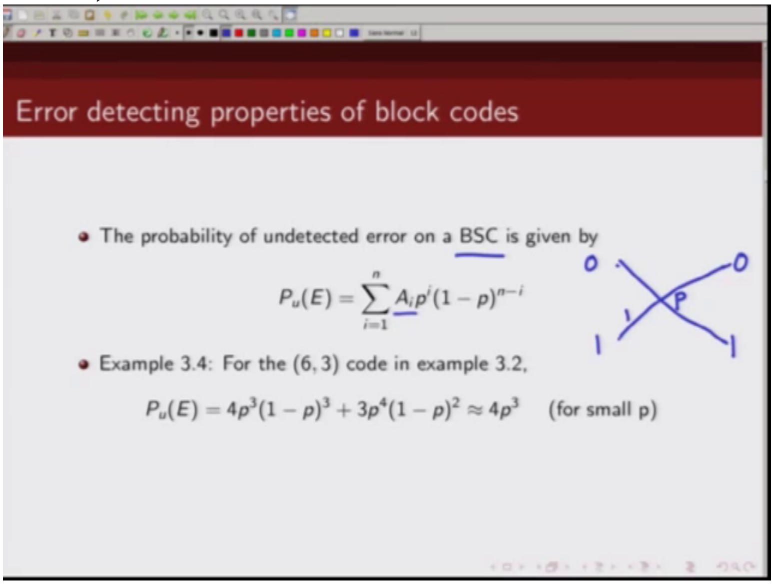
two outputs 0 and 1, and what

(Refer Slide Time 07:52)



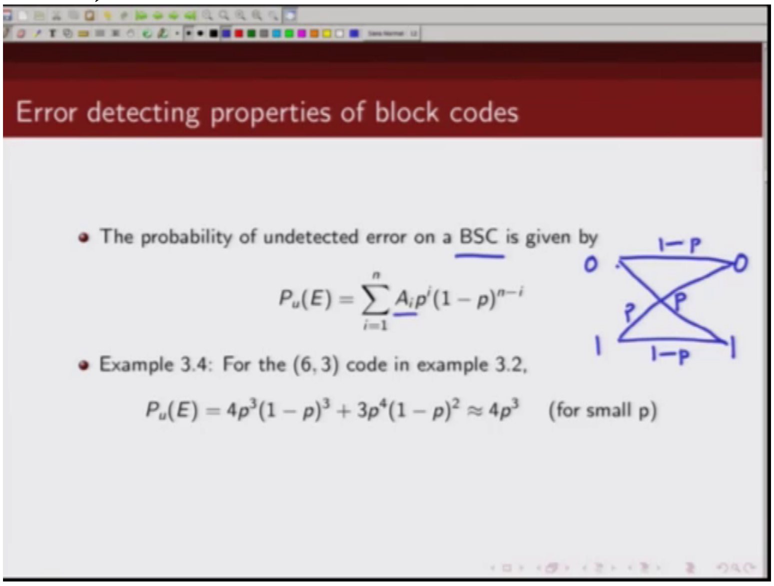
is the crossover probability? That is basically given by p. So with probability p,

(Refer Slide Time 07:59)



0 can get flipped to 1, 1 can get flipped to 0. And the probability of correct detection is 1 minus p. So you are sending a

(Refer Slide Time 08:11)

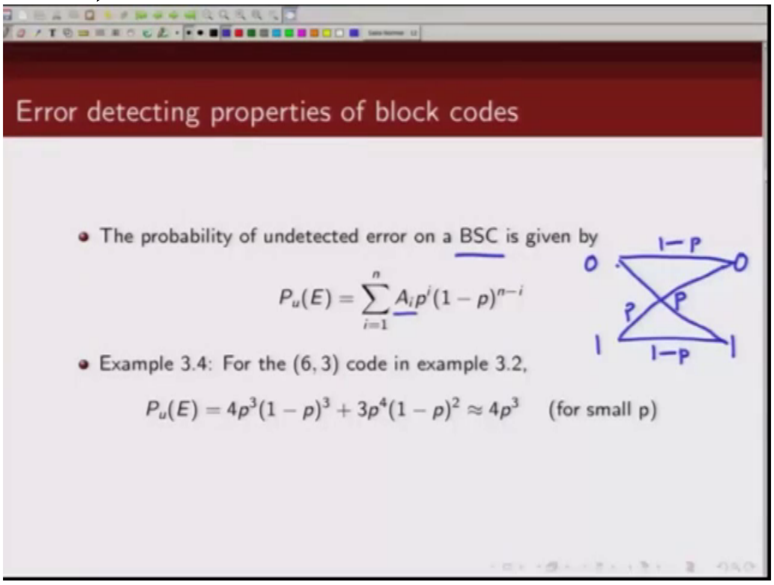


codeword which is an n-bit tuple. Now what's a probability

(Refer Slide Time 08:17)

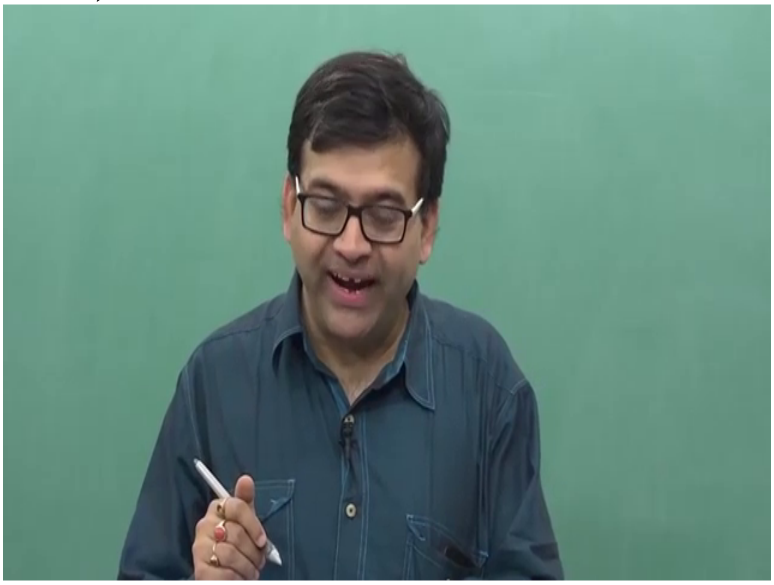
that you are sending an all zero codeword of all zero bits, you receive another codeword of

(Refer Slide Time 08:28)



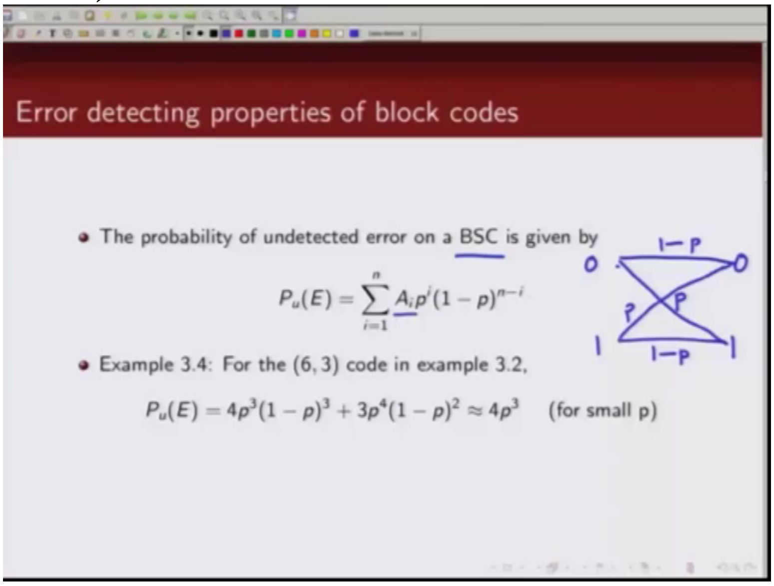
weight i. Now that probability is given by p raised to power i. This will happen when i bits get flipped and n minus i bits do not get flipped. So that probability is given by p raised to power i into 1 minus p raised to power n minus i and how many such codewords exist? That number is given by a i. So the probability of getting a weight i

(Refer Slide Time 09:02)



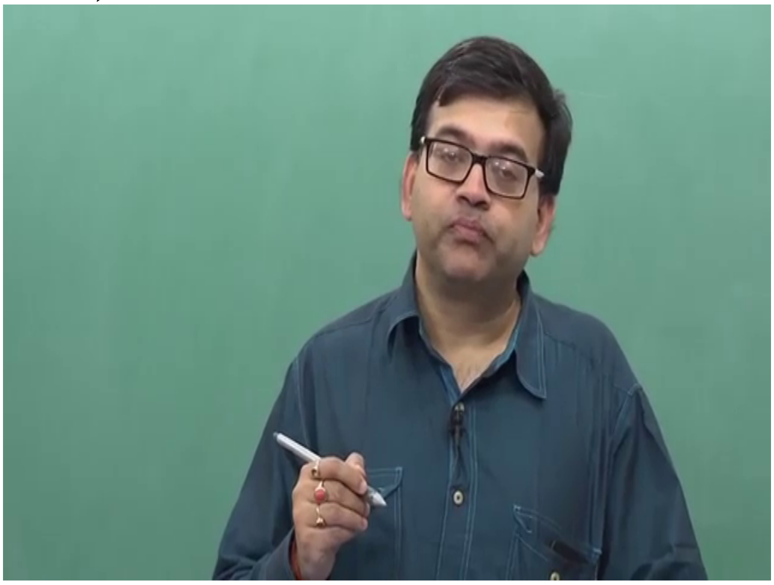
codeword at the receiver; when you send an all zero codeword, that probability

(Refer Slide Time 09:08)



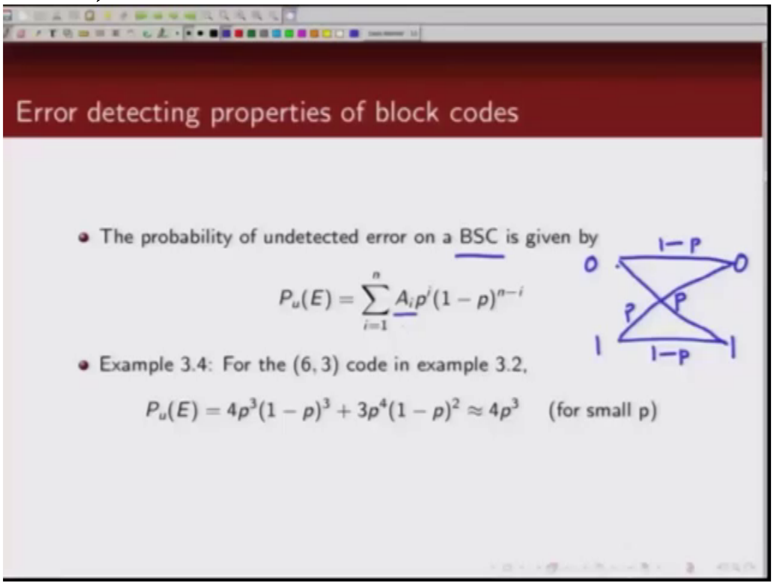
is basically given by this. Ok. Now

(Refer Slide Time 09:13)



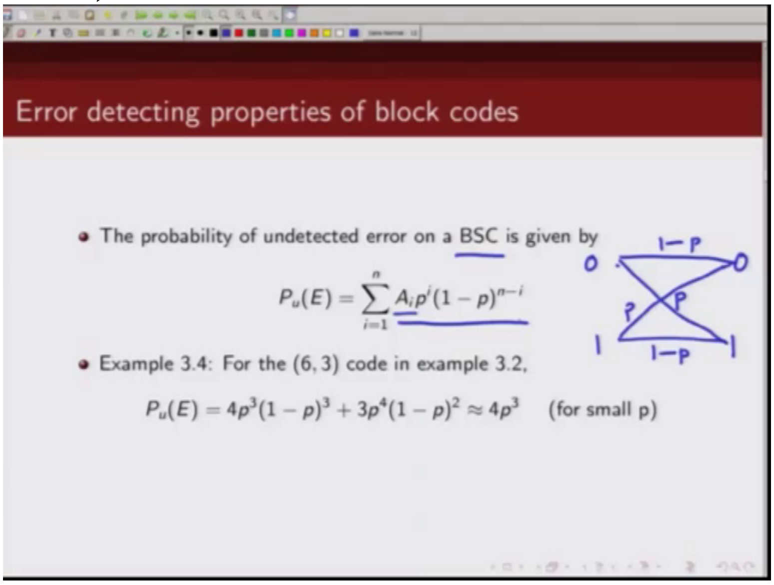
an undetected error will happen if the receiver receives any non-zero codeword. So I have to sum up

(Refer Slide Time 09:22)



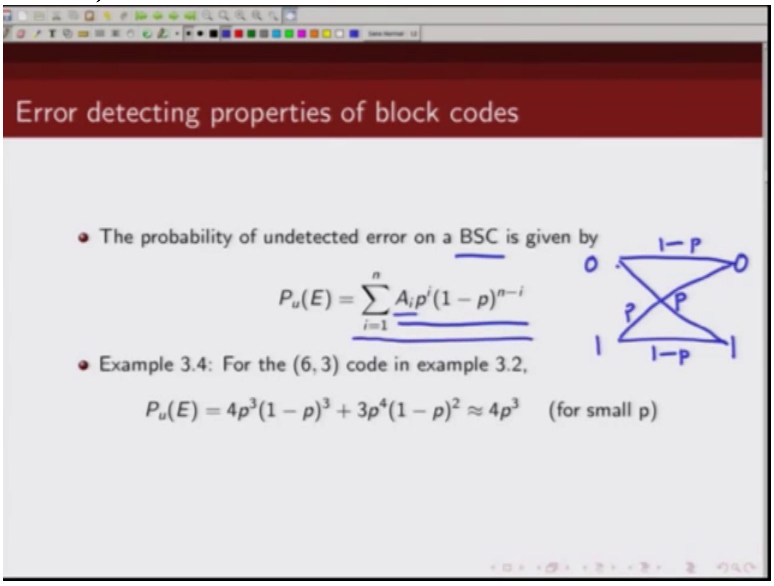
this probability

(Refer Slide Time 09:25)



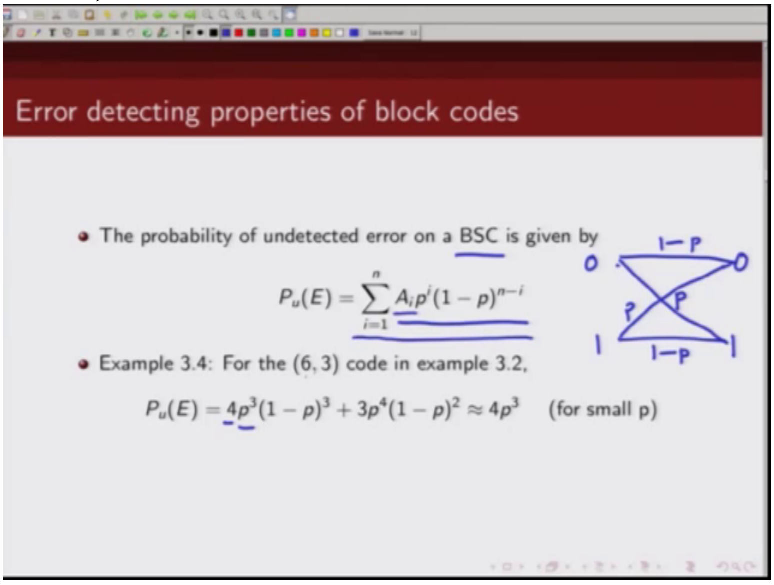
for all i going from 1 to n. So this is my overall

(Refer Slide Time 09:34)



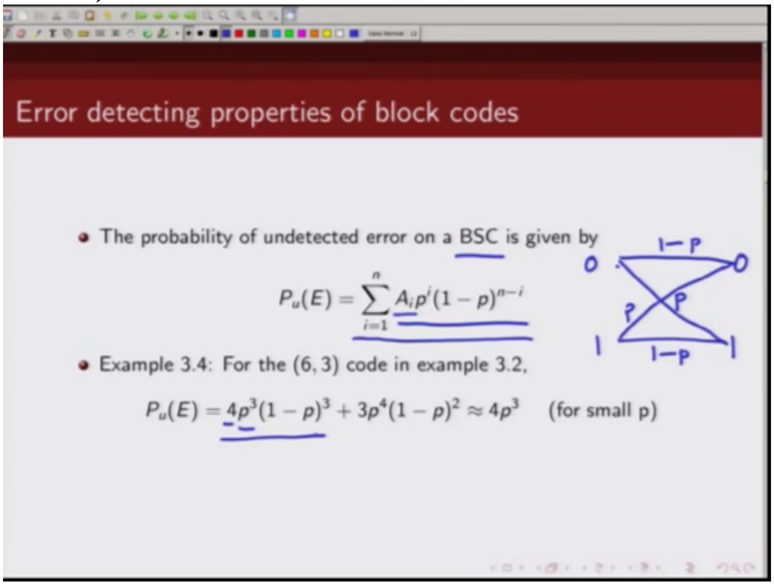
undetected error probability if I send a linear block code over a binary symmetric channel. So for the example that I have considered I know the weight distribution, so if I plug that in here what I get is, so there were 4 codewords with weight 3, so this is 4 p raised to power 3. And what was n, n is 6.

(Refer Slide Time 10:04)



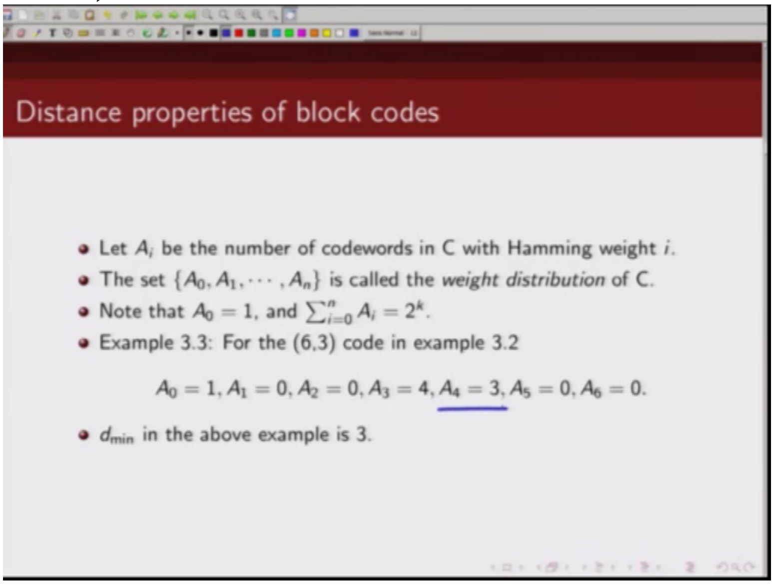
So 6 minus i which is 3 in this case, it's 3. So first term that I will get is this.

(Refer Slide Time 10:12)



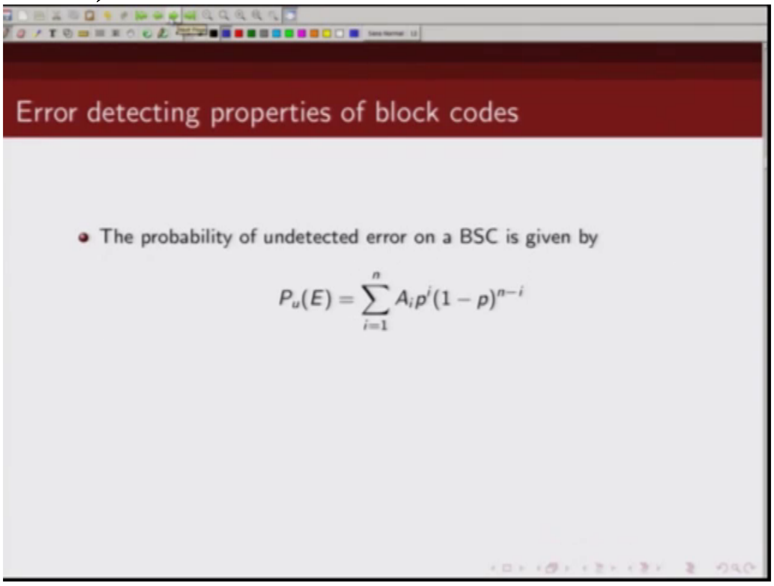
The next term corresponding to

(Refer Slide Time 10:20)



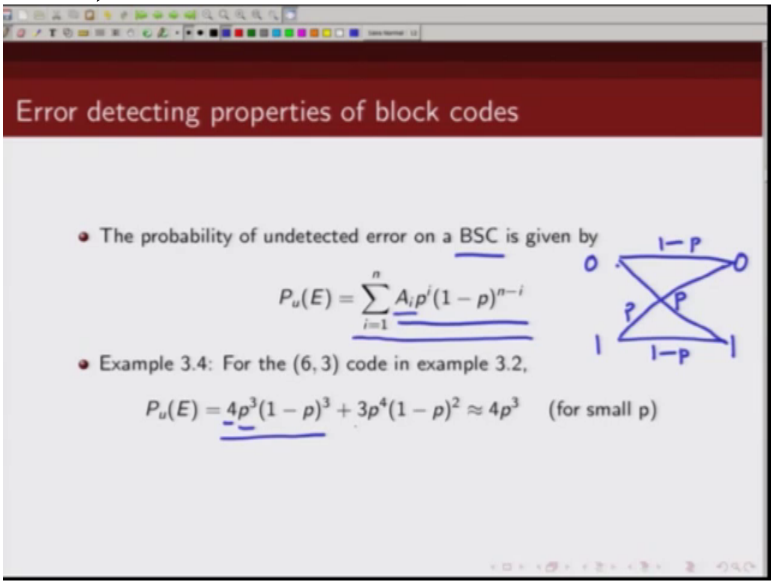
these codewords

(Refer Slide Time 10:23)



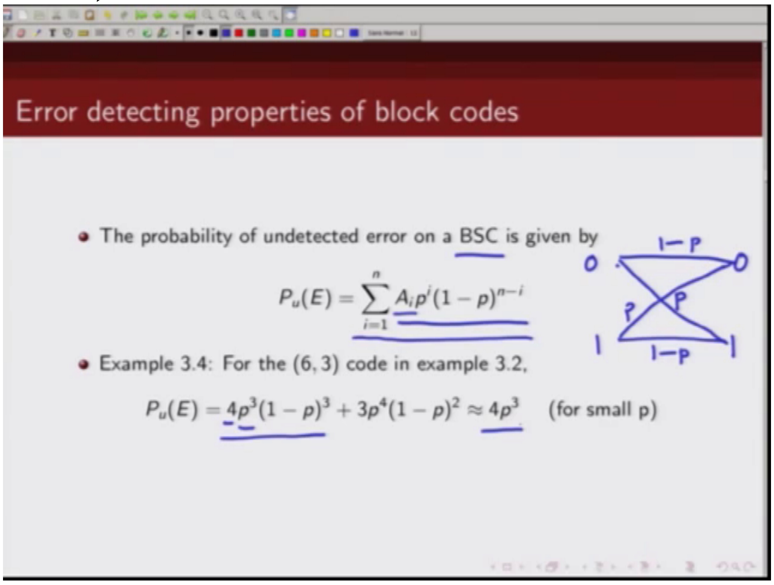
is given, so there are 3 codewords of weight 4

(Refer Slide Time 10:26)



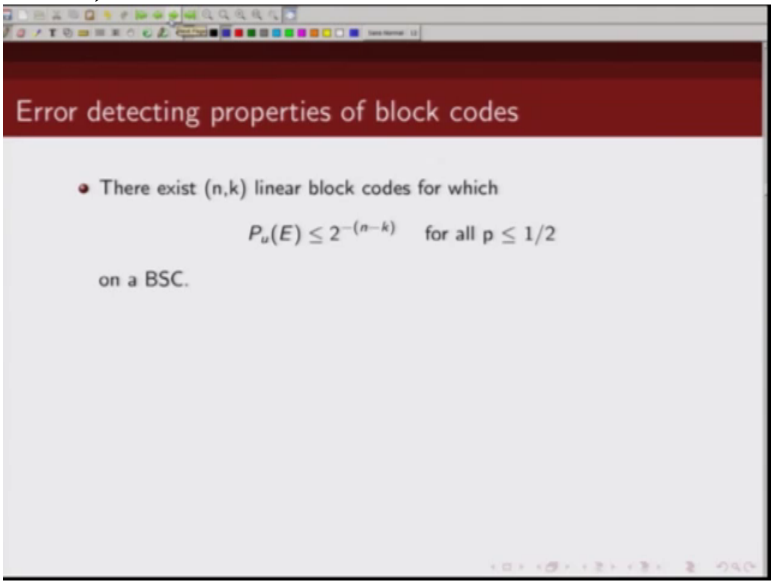
Probability of 4 bits getting flipped is p raised to power 4 and probability of the other 2 bits not getting flipped is 1 minus p whole square. And since p is typically small, I mean I can approximate it, for small p I can approximate this undetected error probability

(Refer Slide Time 10:45)



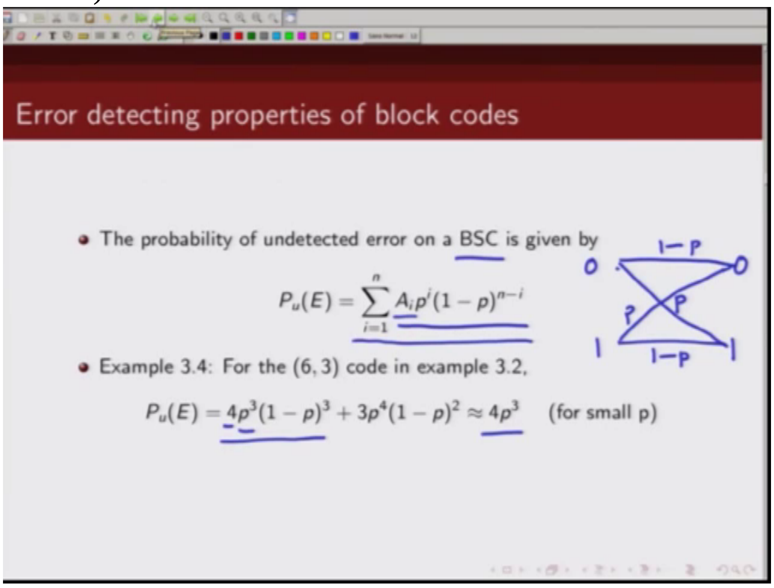
as 4 times p q, because this will be close to 1 and since p is 4, small number, p raised to power 4 will be a small number. So this will be roughly equal to 4 into p raised to power 3. This is for the case when p is small.

(Refer Slide Time 11:05)



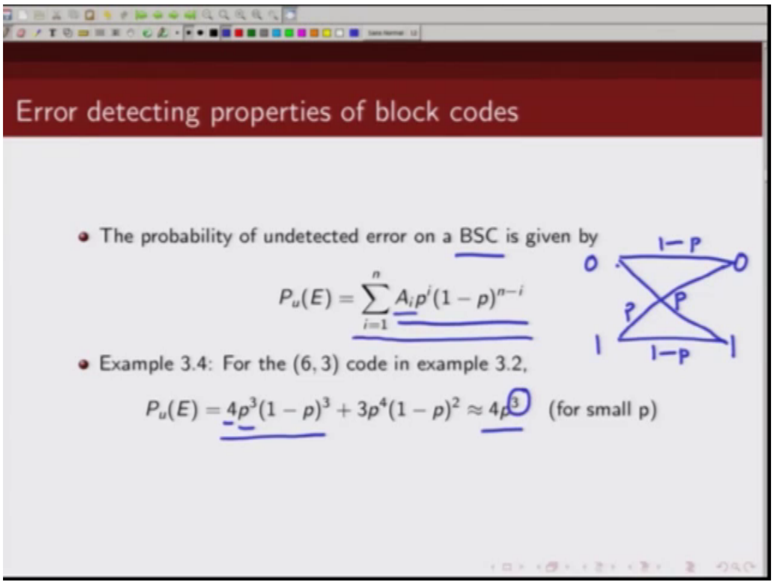
So you can see in general, so in this particular example

(Refer Slide Time 11:11)



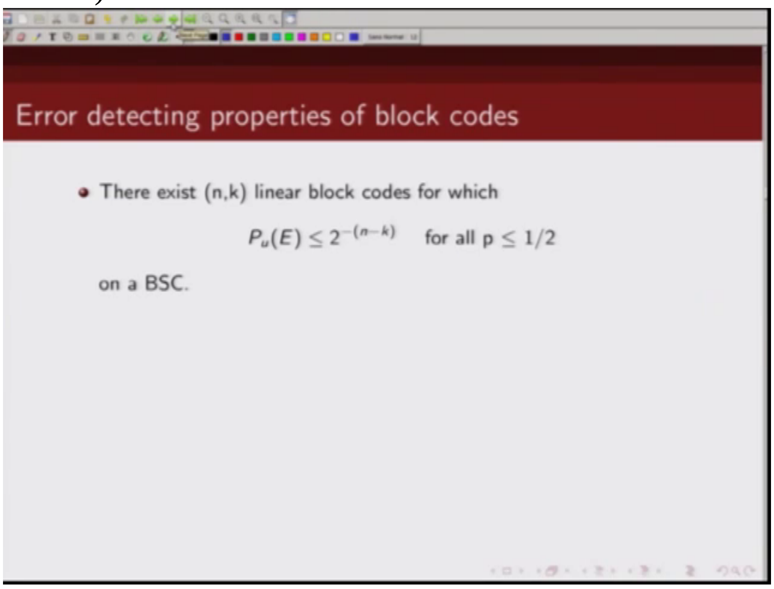
the undetected probability basically

(Refer Slide Time 11:14)



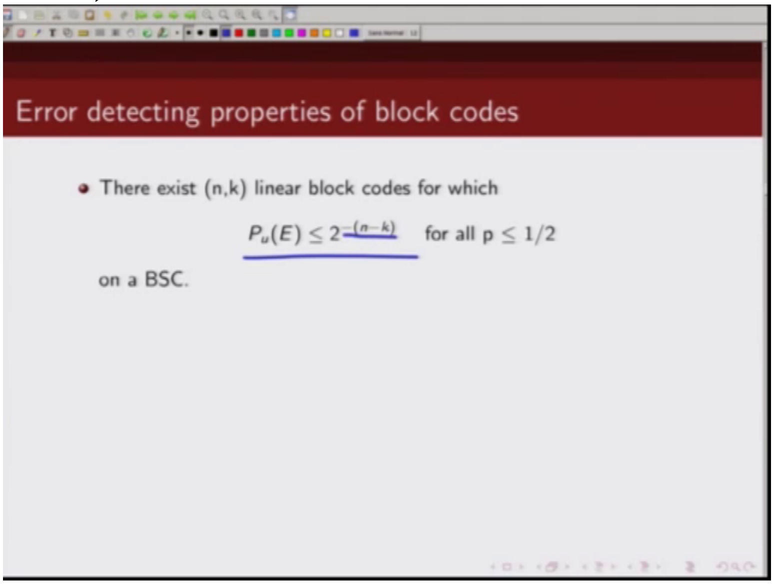
varies as p raised to power 3 which is basically same as n minus k. In general we can show that

(Refer Slide Time 11:22)



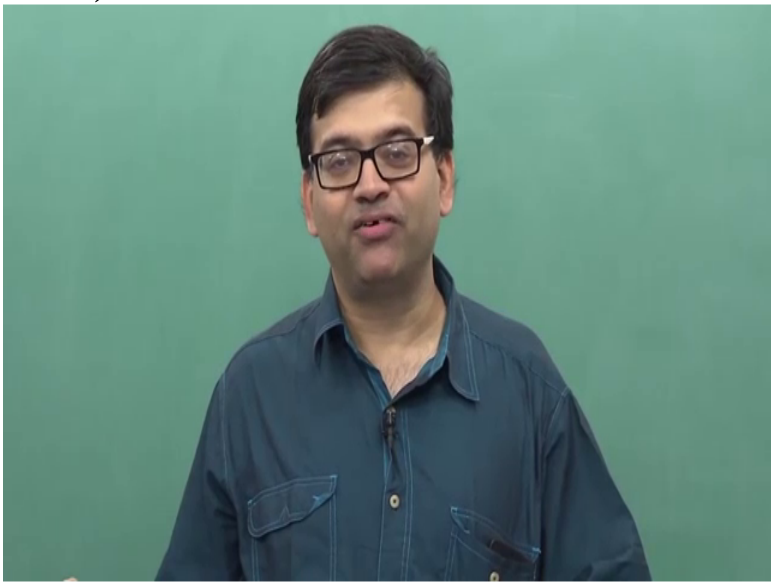
that undetected probability is dependent on how many

(Refer Slide Time 11:27)



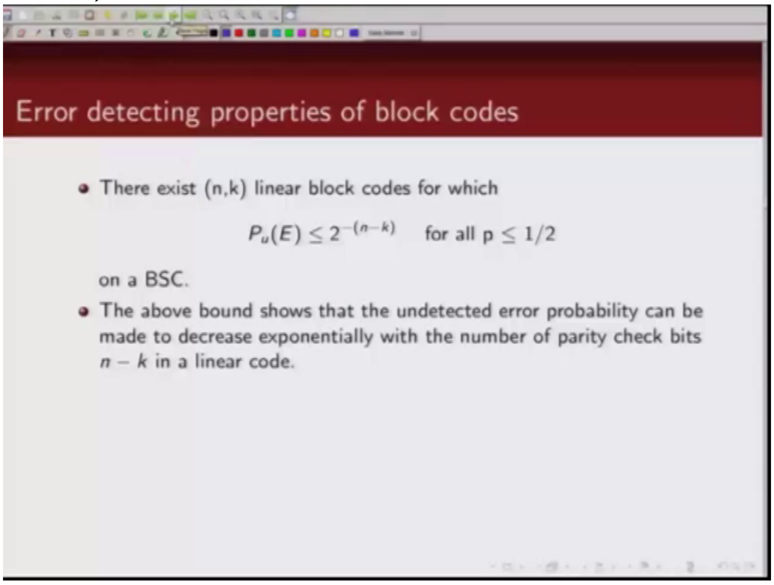
parity bits that we have; so the more

(Refer Slide Time 11:30)



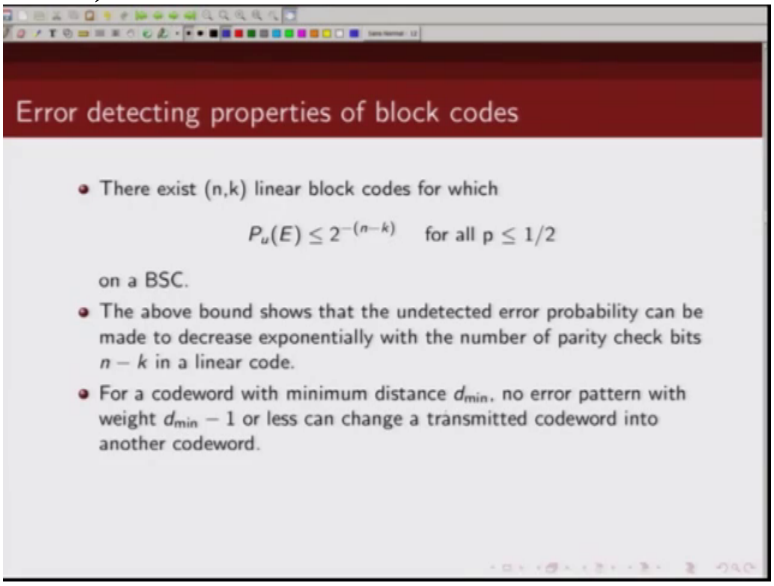
the number of parity bits, lesser will be the undetected error probability. So we can make

(Refer Slide Time 11:37)



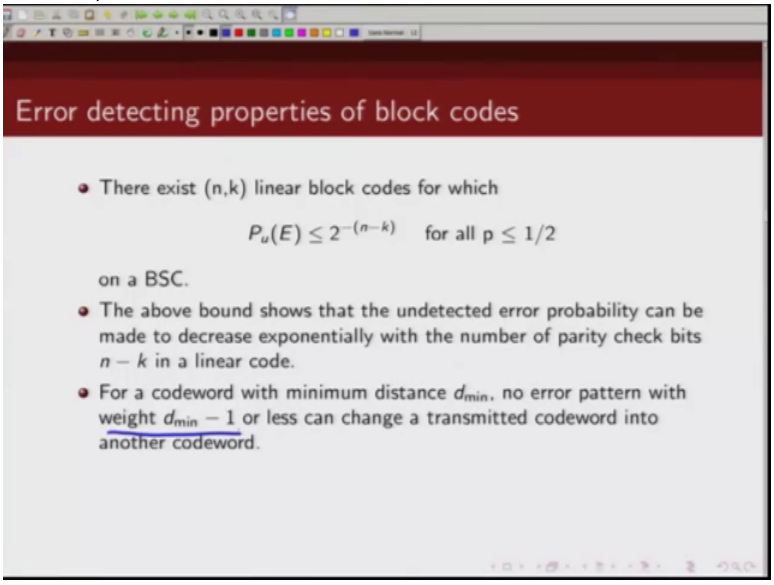
undetected probability go small by increasing the number of parity bits. Now if we have

(Refer Slide Time 11:45)



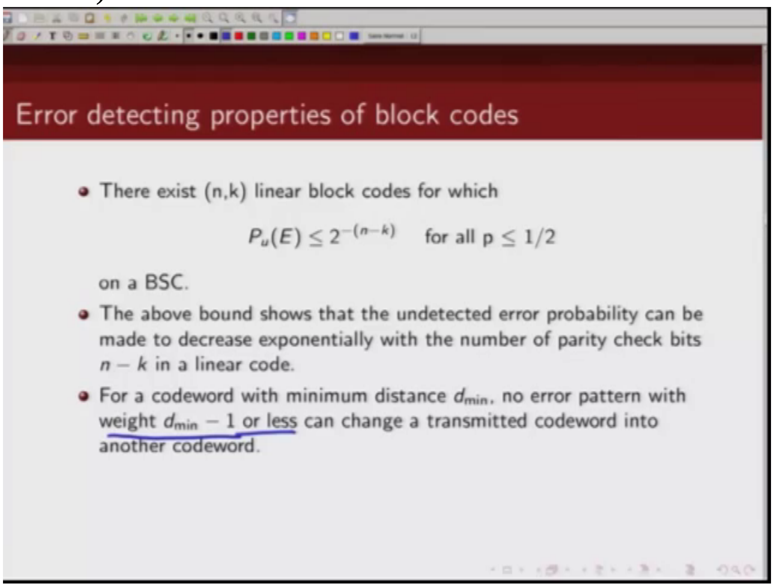
a codeword with minimum distance d min, we know that any error pattern or weight less than equal to d min minus 1 is not going to change that codeword into any other valid codeword. So in other words, if there is an error pattern of weight

(Refer Slide Time 12:14)



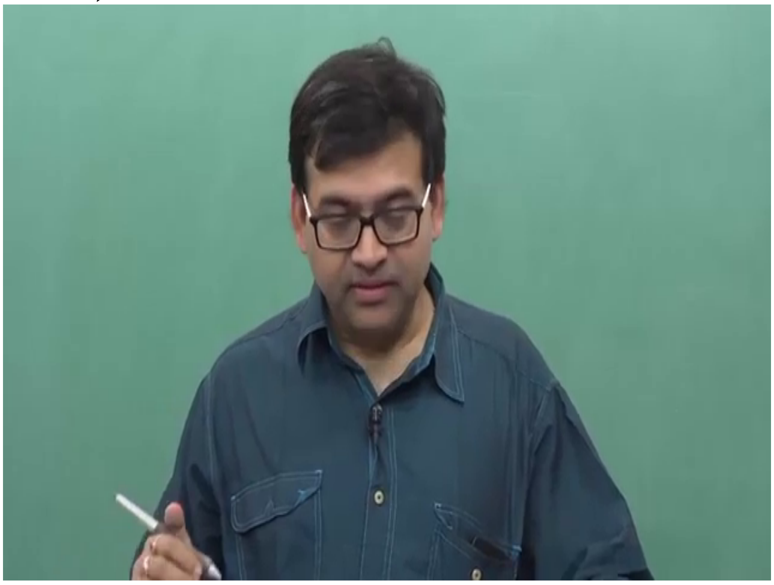
d min minus 1 or less, then

(Refer Slide Time 12:19)



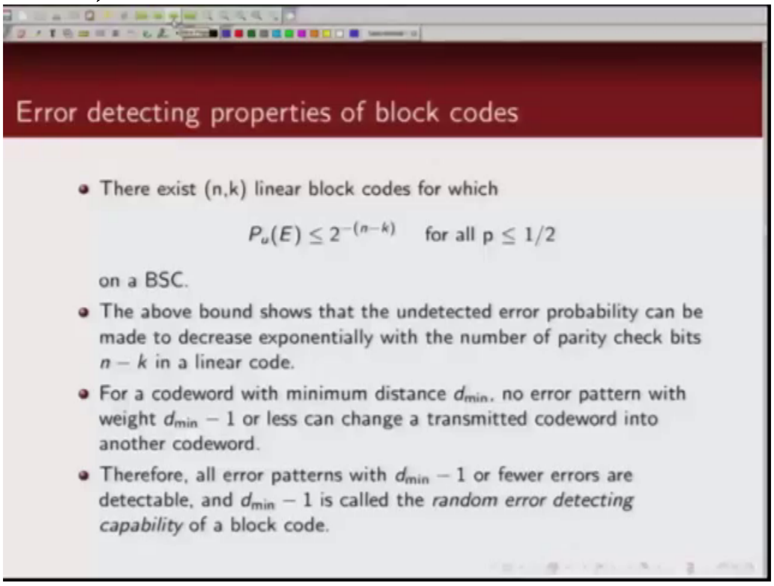
it cannot change a

(Refer Slide Time 12:24)



valid codeword into another valid codeword. What does that mean? It means that we can actually detect any error pattern of weight up to d min minus 1.So

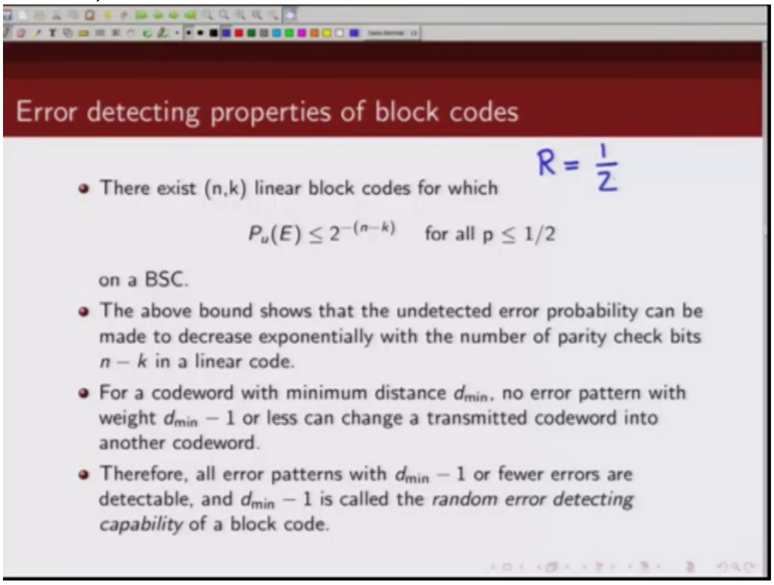
(Refer Slide Time 12:42)



all error patterns of weight d min minus 1 or fewer errors are basically detectable and this is also known as random error correcting capability of a linear block code.

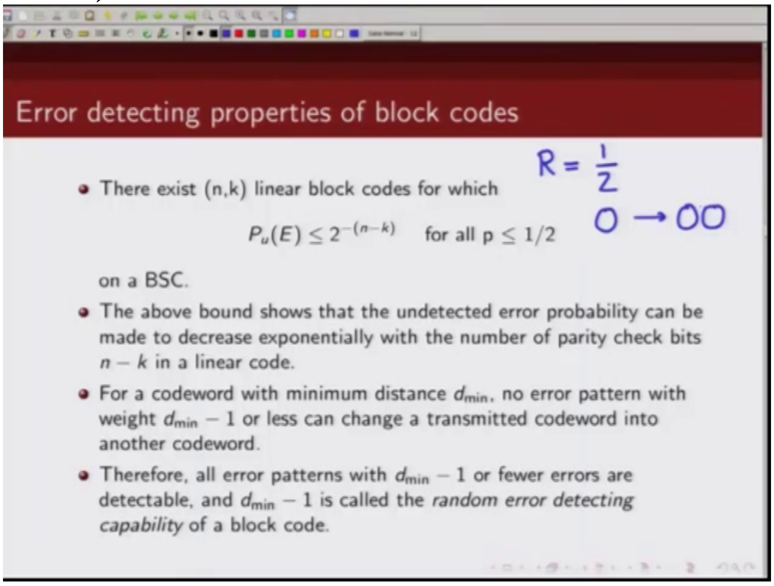
Now take an example of a repetition code that we did in the first class. So let's say we have a rate one half repetition code.

(Refer Slide Time 13:09)



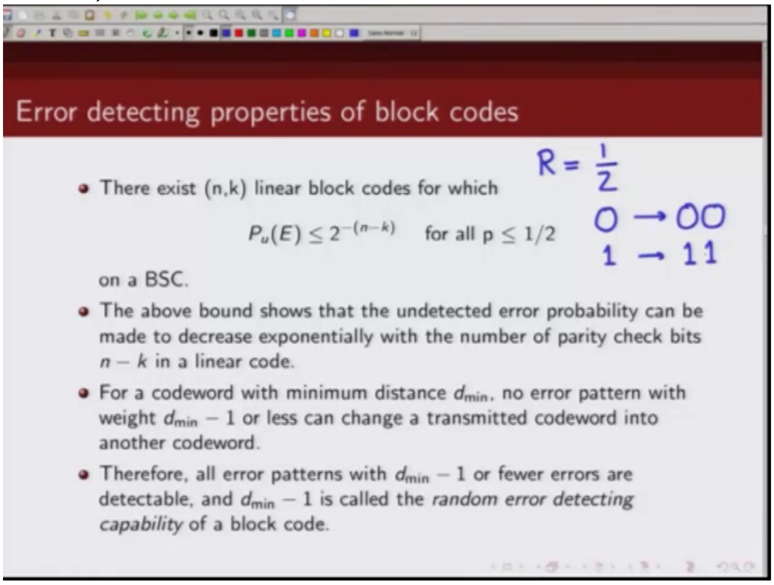
So then for 0, we are sending 0 0

(Refer Slide Time 13:16)



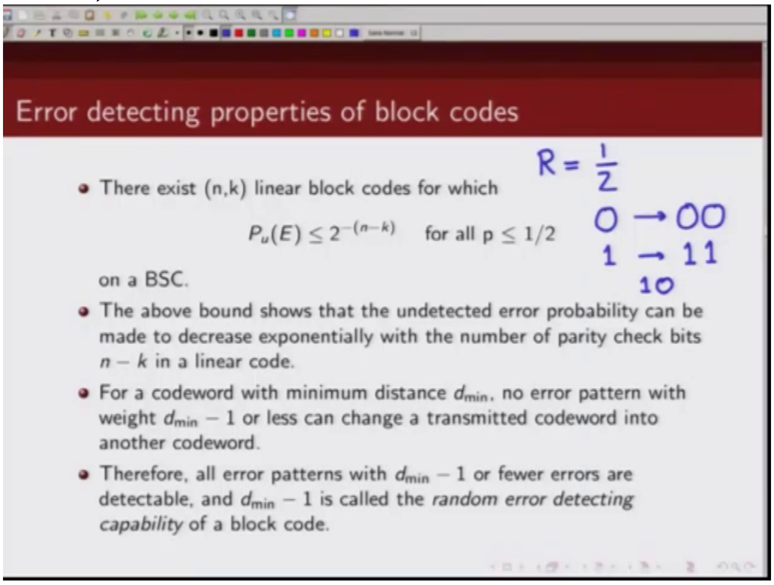
and for 1 we are sending 1 1.

(Refer Slide Time 13:22)



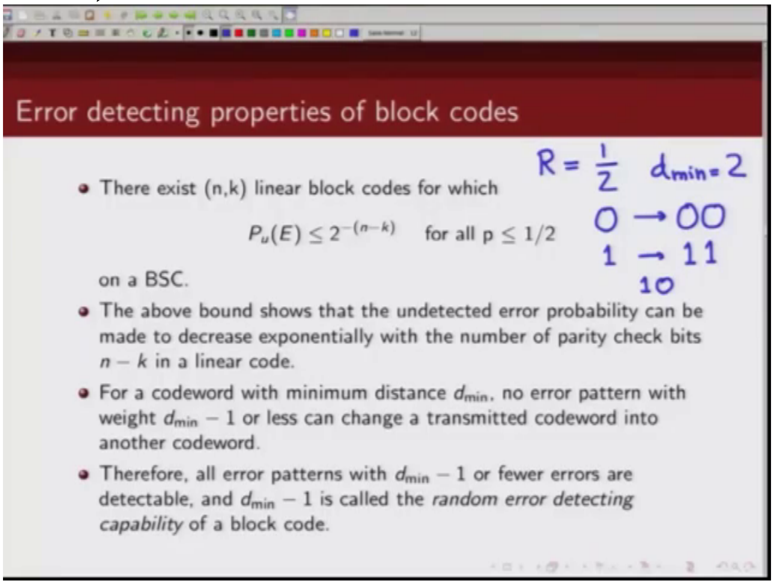
Now let's assume because of error in the channel some of the bits got flipped. So let's say this what we received when we, let's say what we received was 1 0. If you receive 1 0 can you detect?

(Refer Slide Time 13:45)



So what is the minimum distance, first answer this question. What is the minimum distance of this code, this rate one half repetition code? We can see the minimum distance is 2. Minimum distance of this code is 2. So

(Refer Slide Time 14:04)

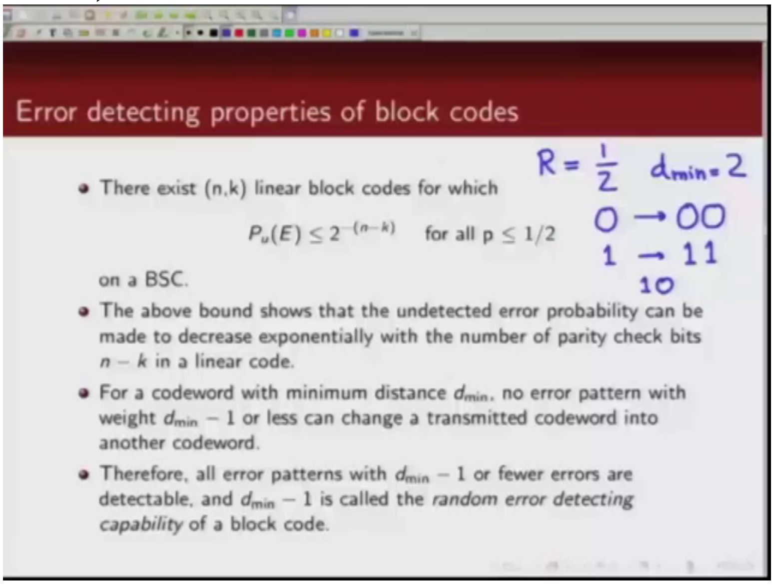


according to this, we should be able all error patterns of weight 1. So let's take an example. Let's say we received 1 0, can you detect the error? Yes we can because since it’s a rate one half repetition code what we expect to receive

(Refer Slide Time 14:30)

either 0 0 or 1 1 if we

(Refer Slide Time 14:35)



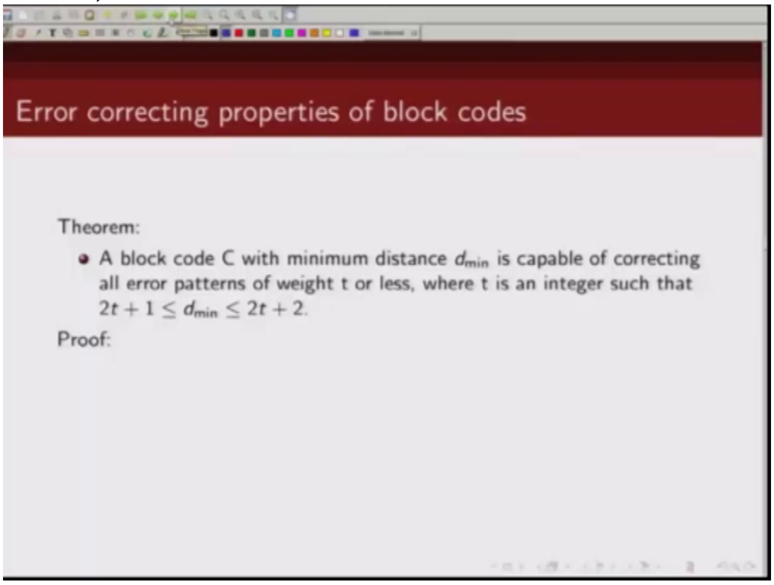
transmit these codewords over a binary symmetric channel. But what we have received is 1 0 which is neither 0 0 nor 1 1. So we are able to detect single error. So to repeat basically, if you have a linear block code whose minimum distance is d min. You will be able to detect all errors, random errors of error pattern up to

(Refer Slide Time 15:09)

d min minus 1.

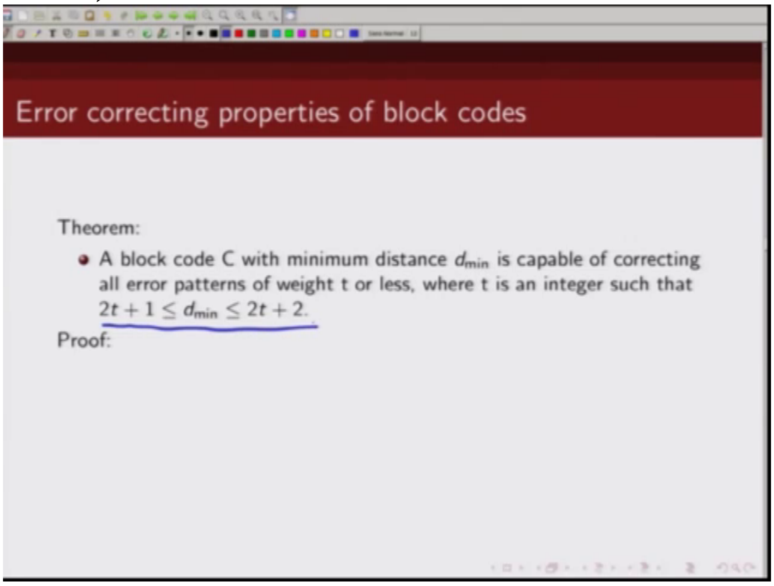
Next we are going to show how is the error detecting capability, error correcting capability of a linear block code related to the minimum distance of a code. So

(Refer Slide Time 15:31)



if we have a linear block code C whose minimum distance is d min where d min satisfies this relation.

(Refer Slide Time 15:45)

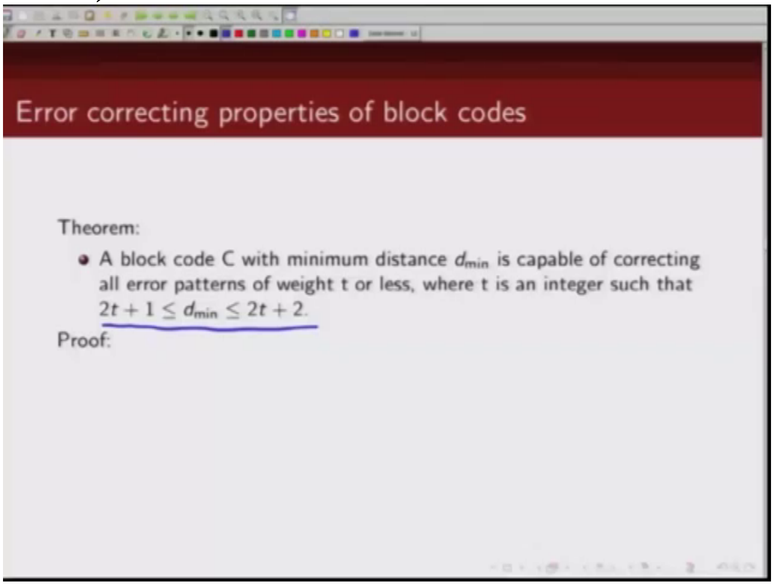


d min is greater than equal to 2 t plus 1 where t is an integer and its less than an integer and it is less than equal to 2 t plus 2. If d min satisfies this relation

(Refer Slide Time 15:58)

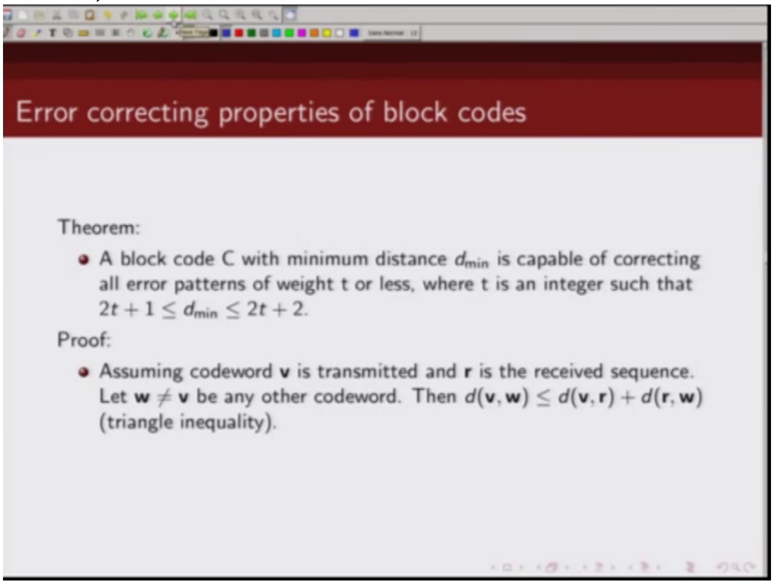
and if we have a linear block code with minimum distance d min then it is capable

(Refer Slide Time 16:06)



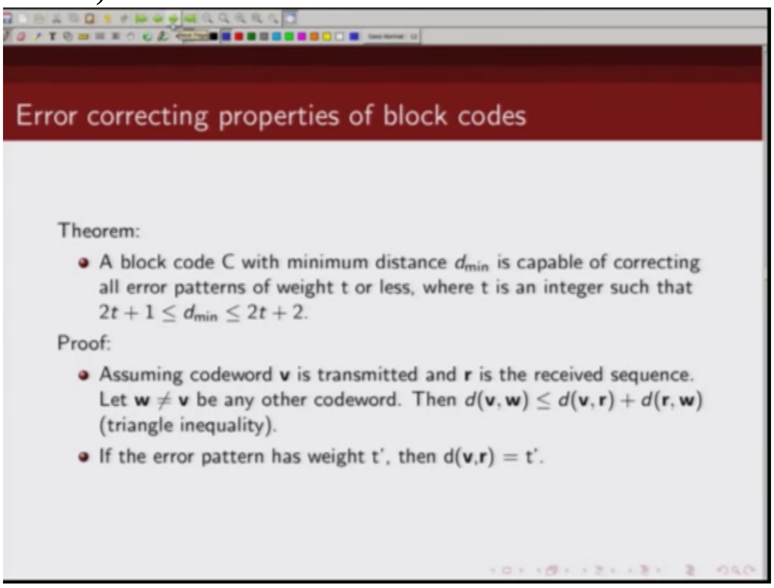
of correcting all error patterns up to weight t. So let us prove this result.

(Refer Slide Time 16:17)



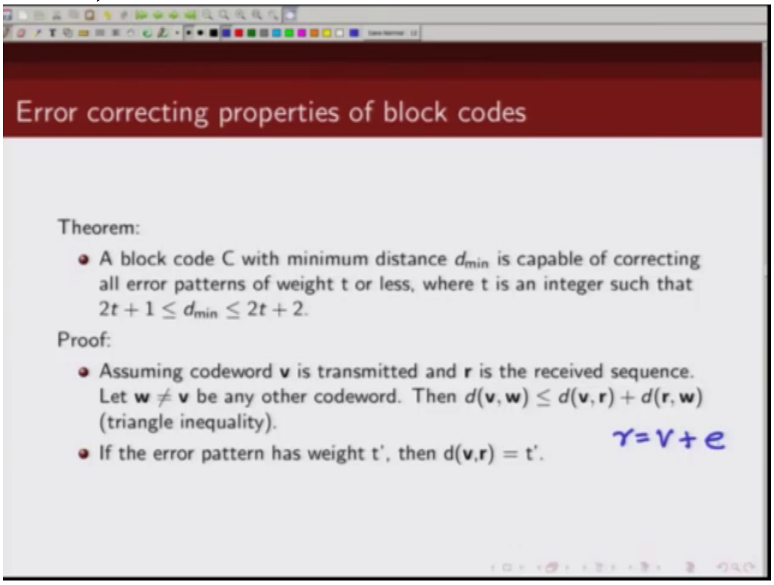
Let us assume the codeword that is transmitted is given by v and what we received is say tuple r. Let us assume there is another codeword w which is not same as v. Now we know from triangular inequality that Hamming distance between v and w will be less than equal to Hamming distance between v and r plus Hamming distance between r and w. Now let us assume

(Refer Slide Time 17:00)



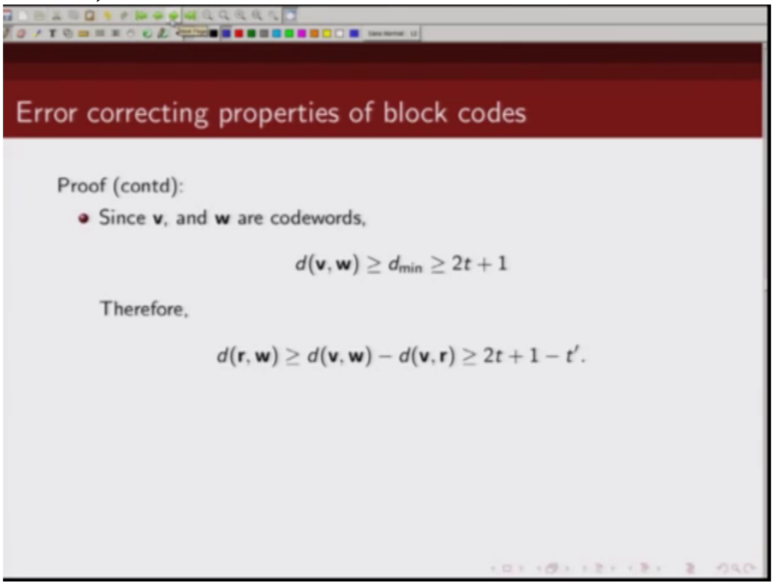
that the error pattern has weight t hat. And what's r; r is nothing but v plus this error pattern,

(Refer Slide Time 17:13)



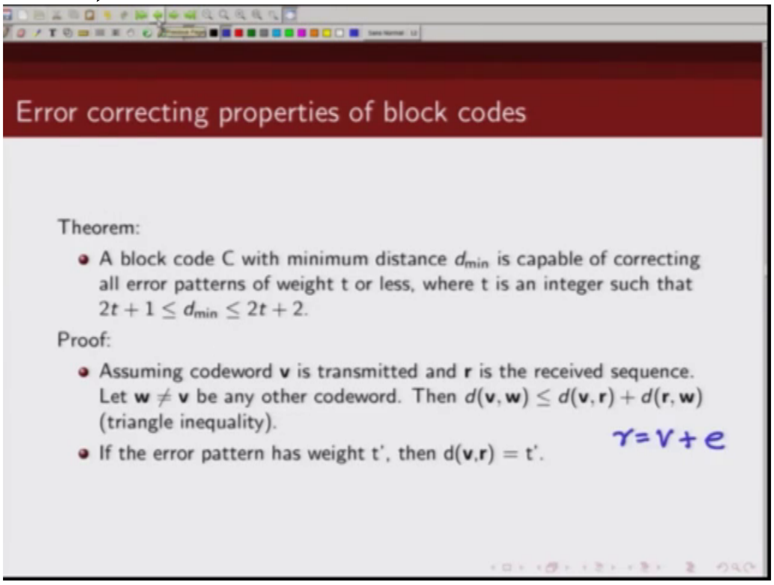
correct? So the Hamming distance between v and r is going to be the weight of this error pattern and which we are denoting by t dash. Now since

(Refer Slide Time 17:29)



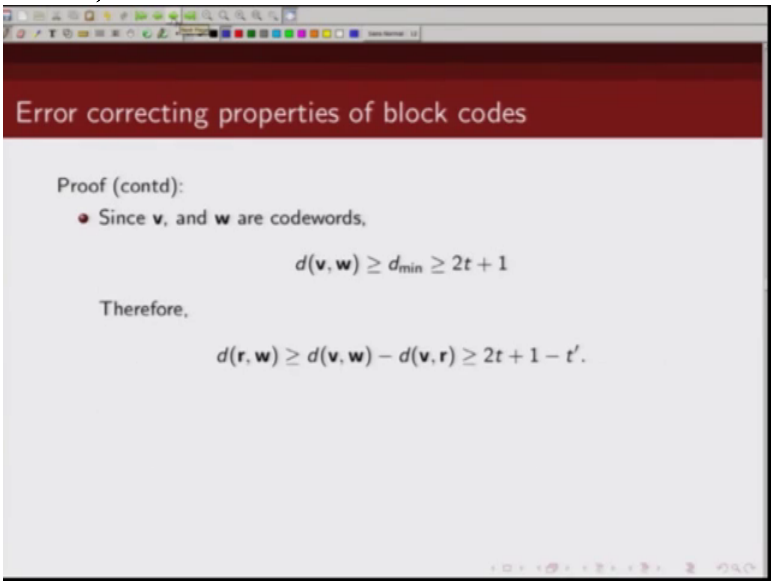
v and w are valid codewords, so the Hamming distance between v and w will be at least equal to the minimum distance of the code. So the Hamming distance between v and w is greater than equal to minimum distance of the code and in the beginning we defined that our minimum distance

(Refer Slide Time 17:53)



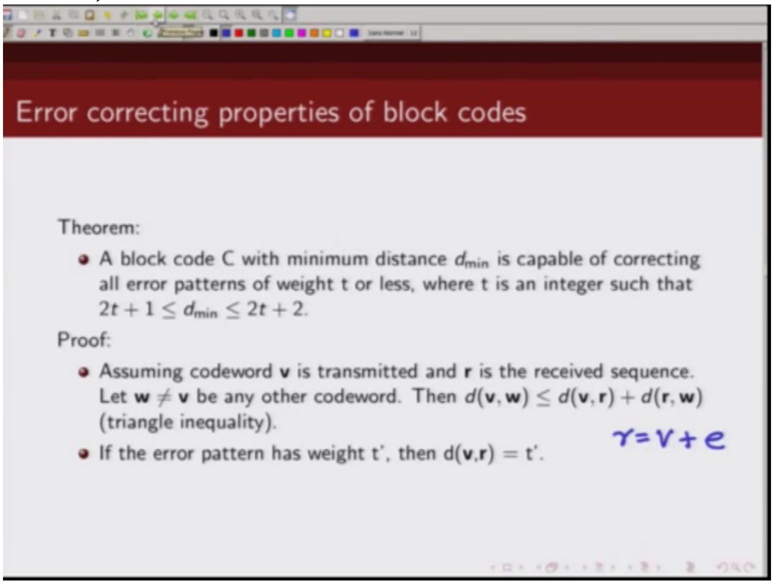
is at least 2 t plus 1. So from these

(Refer Slide Time 17:58)



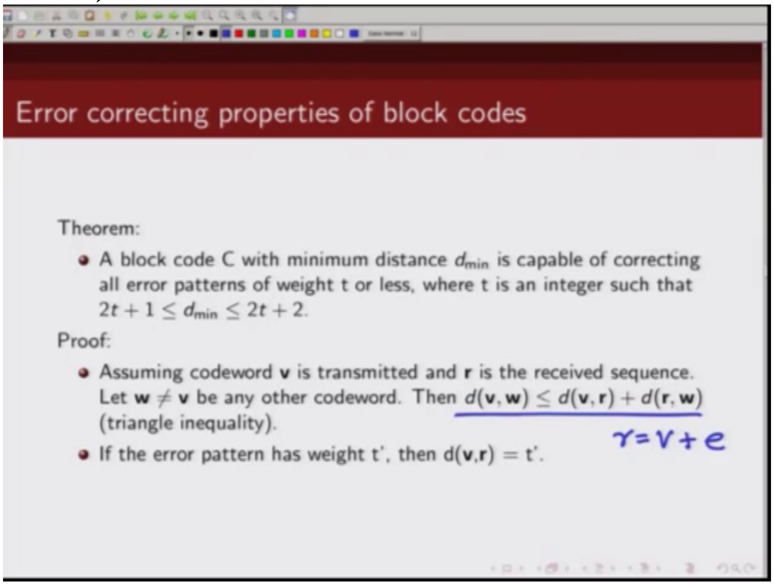
two, we can write that Hamming distance between v and w is greater than equal to 2 t plus 1. Now from the triangular inequality we know that Hamming distance between r and w, this we can see from here, this relationship

(Refer Slide Time 18:24)



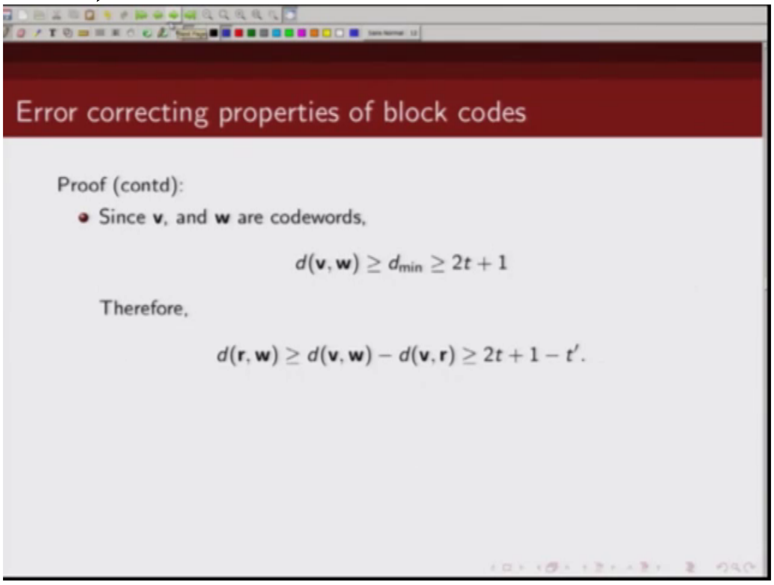
basically triangular inequality

(Refer Slide Time 18:27)



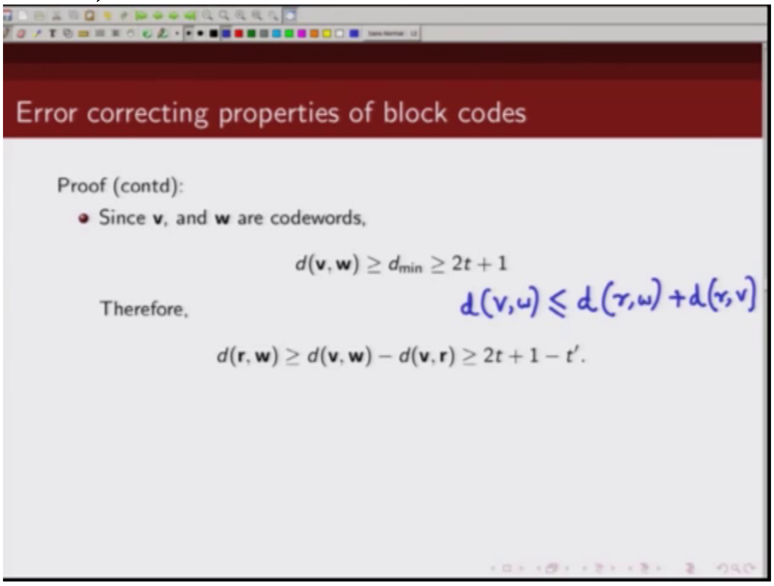
what we have is

(Refer Slide Time 18:29)



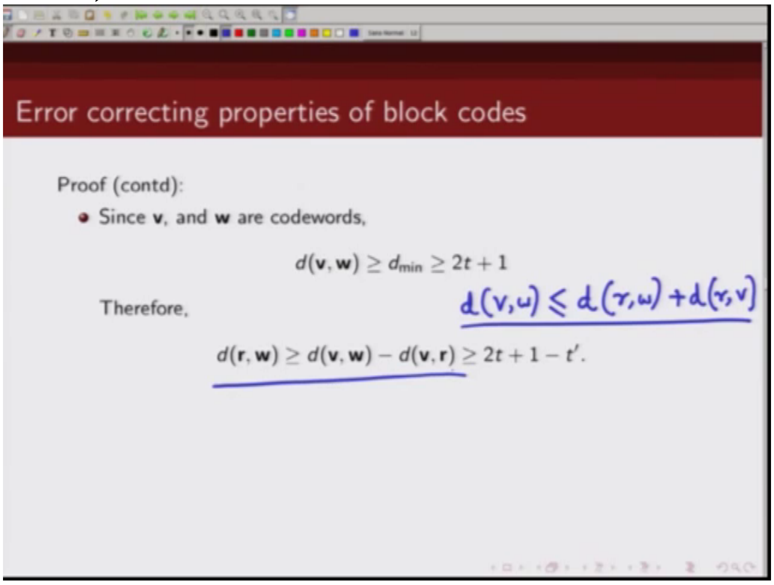
the Hamming distance between v and w to be less than equal to Hamming distance between r and w plus Hamming distance between r and v, right.

(Refer Slide Time 18:48)



Now this we can write as, we can bring this here and we can bring this here, what we can write this as, let us say we can write this, this relation in this particular form.

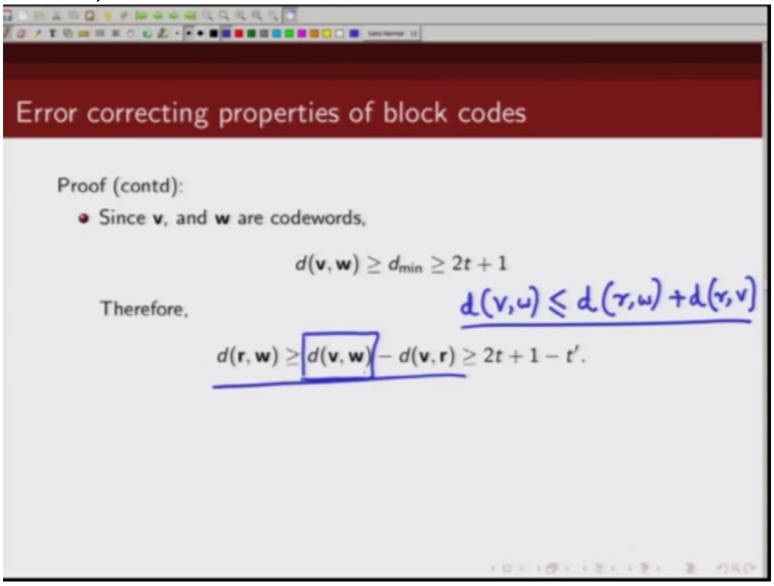
(Refer Slide Time 19:11)



Ok.

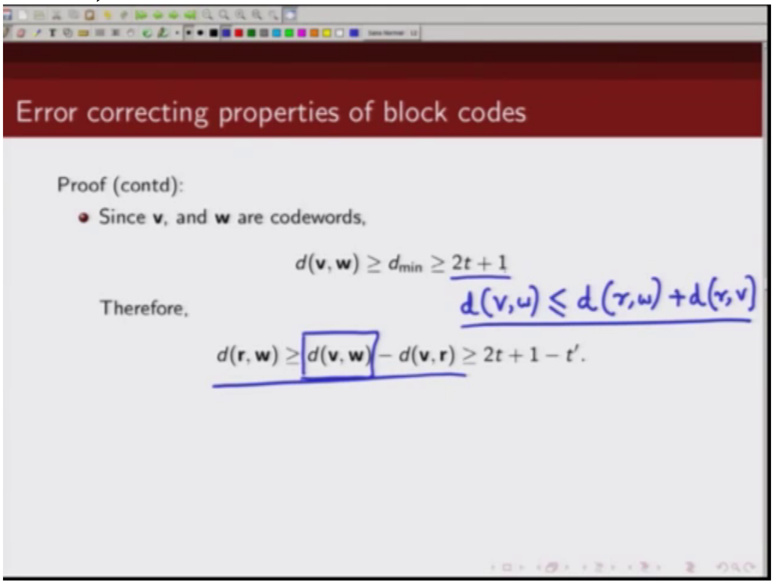
Now what is this quantity, Hamming distance

(Refer Slide Time 19:21)



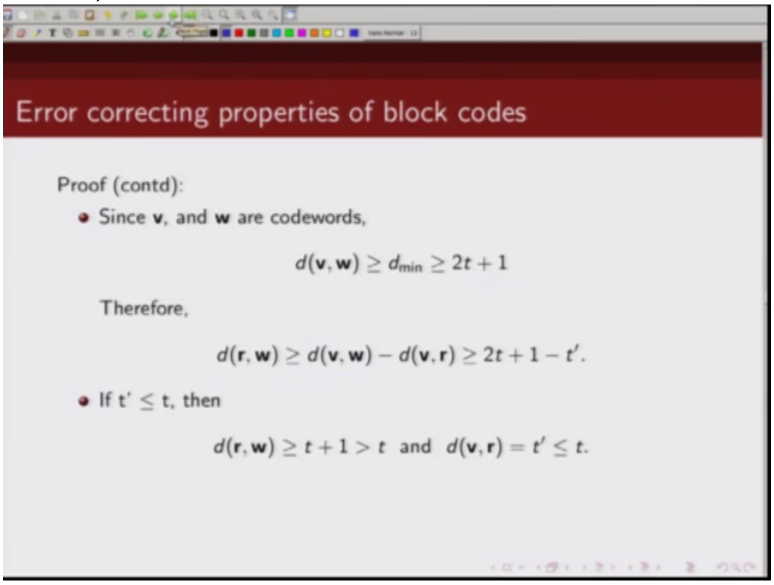
between v and w? The Hamming distance between v and w is at least equal to

(Refer Slide Time 19:28)



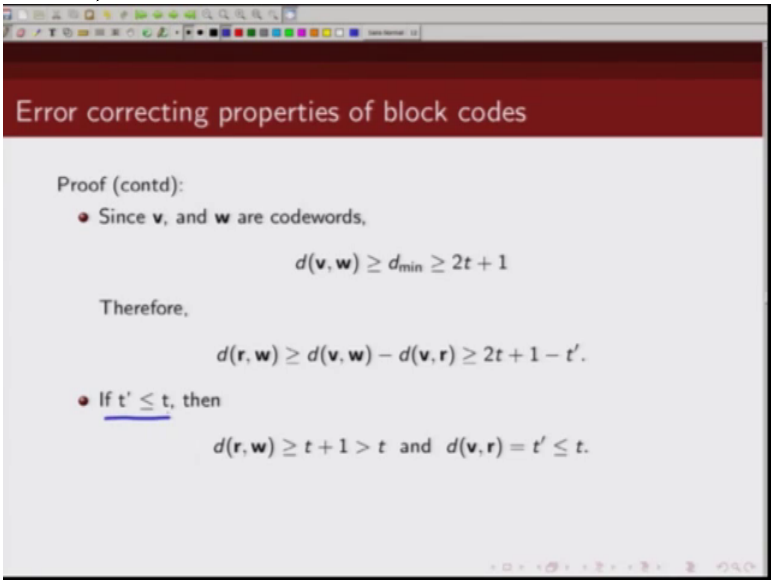
2 t plus 1. And what is Hamming distance between the transmitted codeword and the received codeword? This is we denote it by t dash. So then Hamming distance between r and w is given by 2 t plus 1 minus t dash. Now

(Refer Slide Time 19:54)



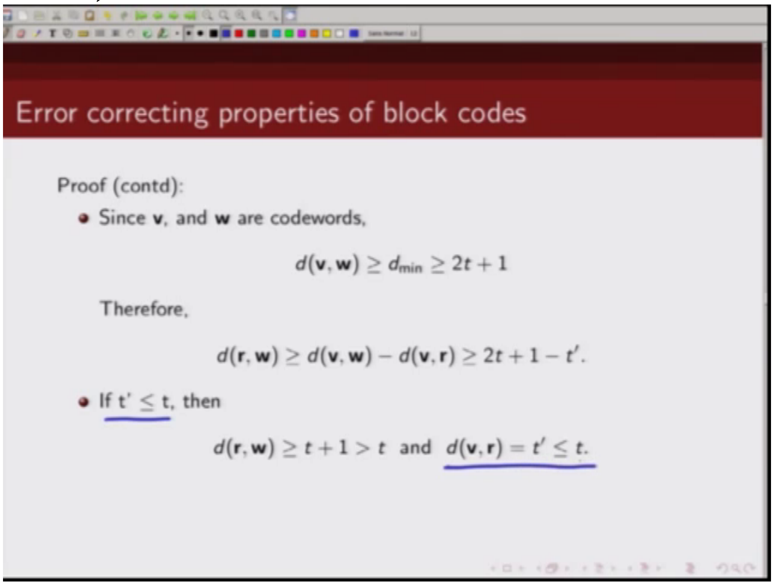
as long as your error pattern is

(Refer Slide Time 20:00)



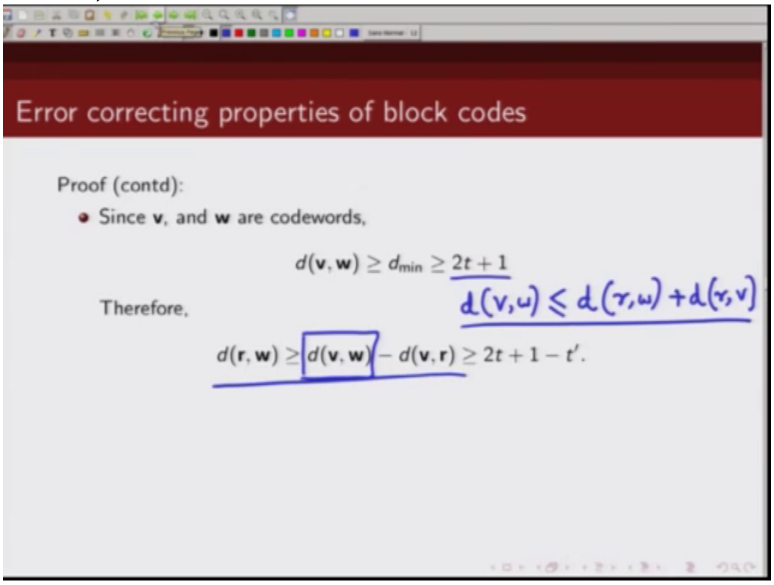
less than equal to t the weight of error pattern is less than equal to t, in that case the Hamming distance between r and w will be, you can plug that value of t here and what we will get is Hamming distance between r and w is greater than equal to t plus 1 which is greater than equal to t where as the Hamming distance between

(Refer Slide Time 20:34)



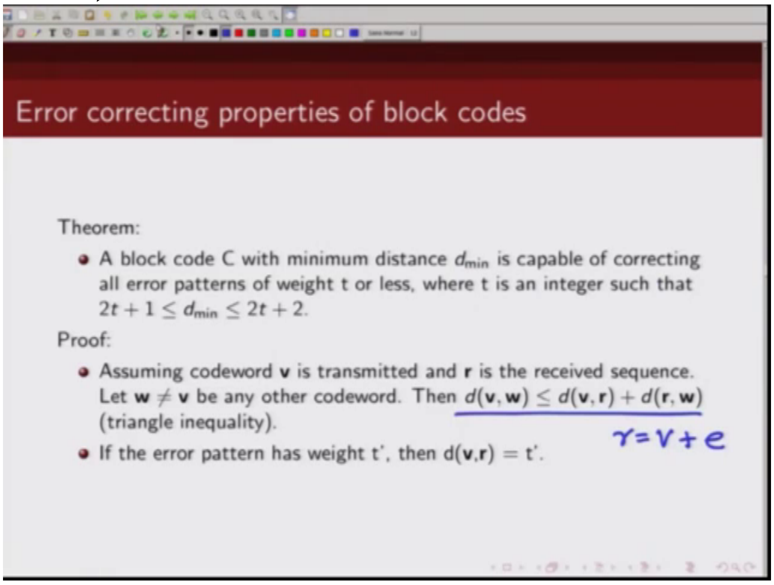
transmitted codeword and the received codeword is t hat which is less than equal to t. What does it mean? It means that the received codeword is closer to v than any other codeword w. So what will be your maximum likelihood decoder for binary symmetric channel will decide in favor of? It will decide in favor of v. So you will correctly decode this received sequence to be v and this was our transmitted codeword. So you will not make an error. So what we have shown here is, as long as your error pattern has weight up to t, those error patterns are correctable provided

(Refer Slide Time 21:36)



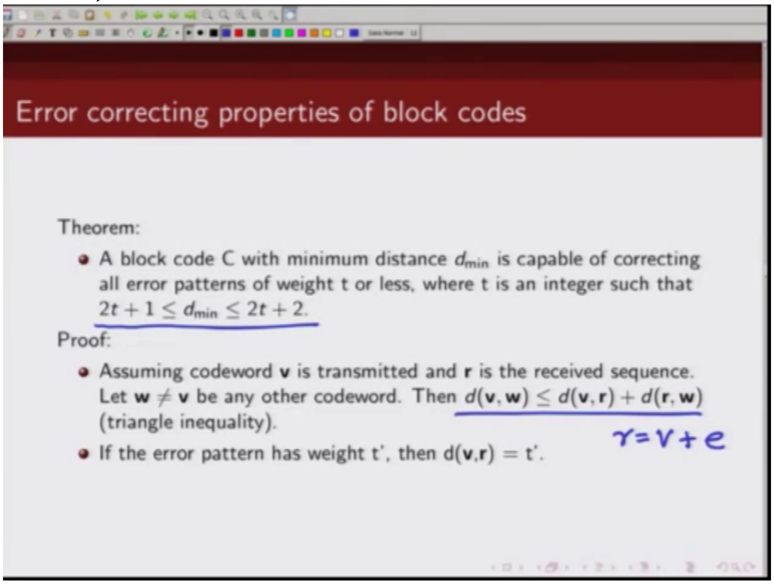
the minimum distance of

(Refer Slide Time 21:38)



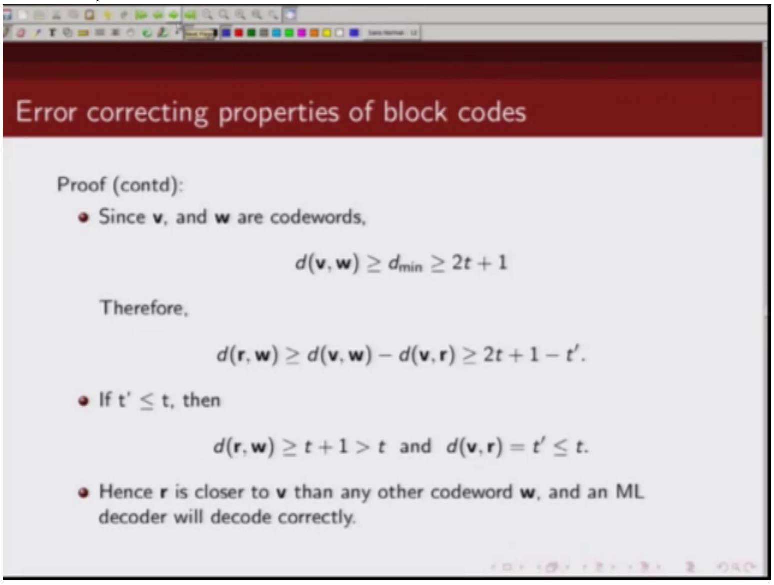
your code is d min

(Refer Slide Time 21:42)



and it satisfies this relationship. So the minimum distance of the code is at least 2 t plus 1, and it is less than equal to 2 t plus 2, then it can correct all error patterns of weight t or less. So as

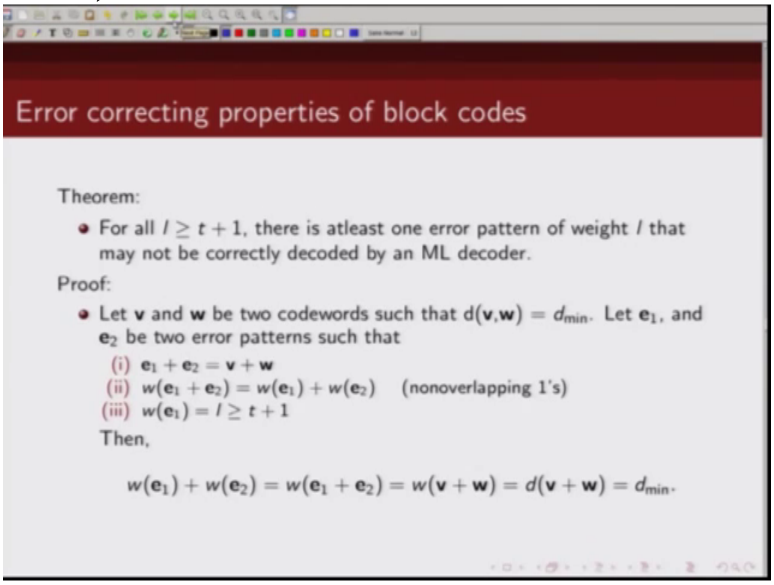
(Refer Slide Time 22:03)



we can see here, the received codeword is closer to v than any other codeword w so it will decide in favor of v and this r will be decoded as v.

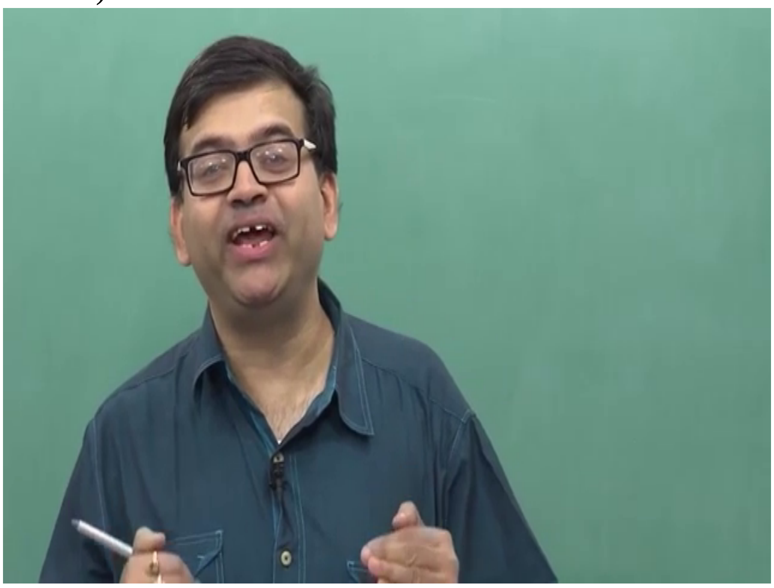
Next we are

(Refer Slide Time 22:18)



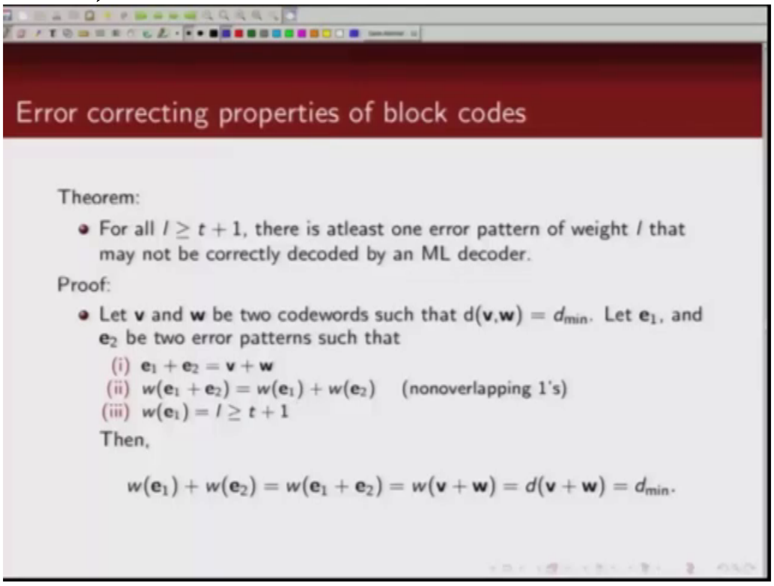
going to show that if there exists an error pattern of weight greater than equal to t plus 1 then our decoder whose minimum distance is at least 2 t plus 1 but less than 2 t plus 2, this decoder will make an error. In other words, it would not be able to correct this error pattern of weight t plus 1. So for all error patterns of weight l, if l is at least t plus 1, then our maximum likelihood decoder may not be able to correctly decode or correct that error. So let's prove this. If v and w are 2 codewords and let's assume that the Hamming distance between v and w is equal to the minimum distance of the code which is denoted by t min. And let e 1 and e 2 are two error patterns which satisfies these 3 properties, and what are these 3 properties? The sum of e 1 and e 2 is the same as v plus w. The second property is, e 1 and e 2, they do not have any overlapping

(Refer Slide Time 23:50)



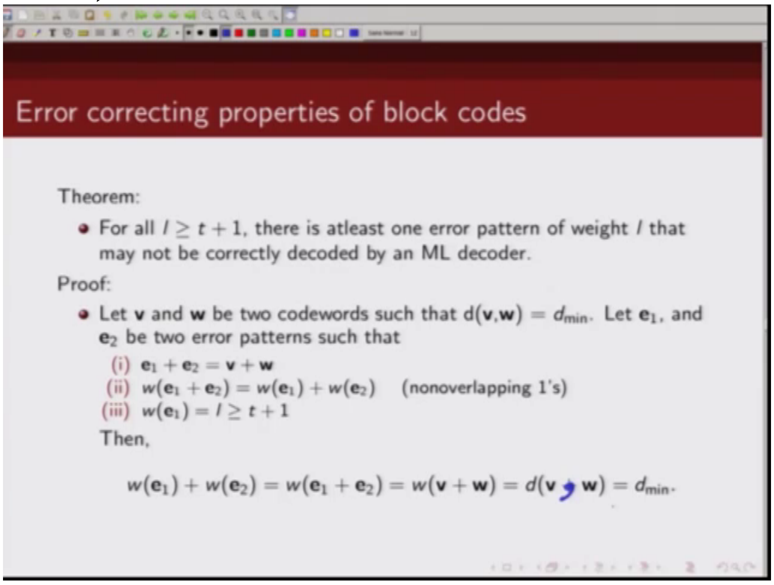
1s. So weight of e 1 plus e 2 can be written as weight of e 1

(Refer Slide Time 23:56)



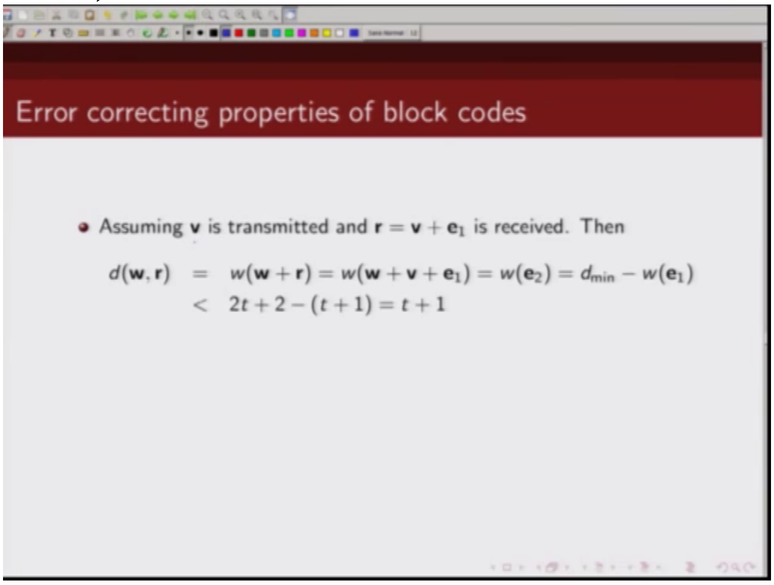
plus weight of e 2. And we will show that if there is an error pattern of weight l where l is at least t plus 1 then our maximum likelihood decoder will make an error in decoding. So the way we have chosen our error pattern, weight of e 1 plus weight of e 2 is given by weight of e 1 plus e 2, this is from 2 and from 1 we know e 1 plus e 2 is nothing but v plus w so this is same as weight of v plus w and this is nothing but this is Hamming distance between v and w

(Refer Slide Time 24:49)



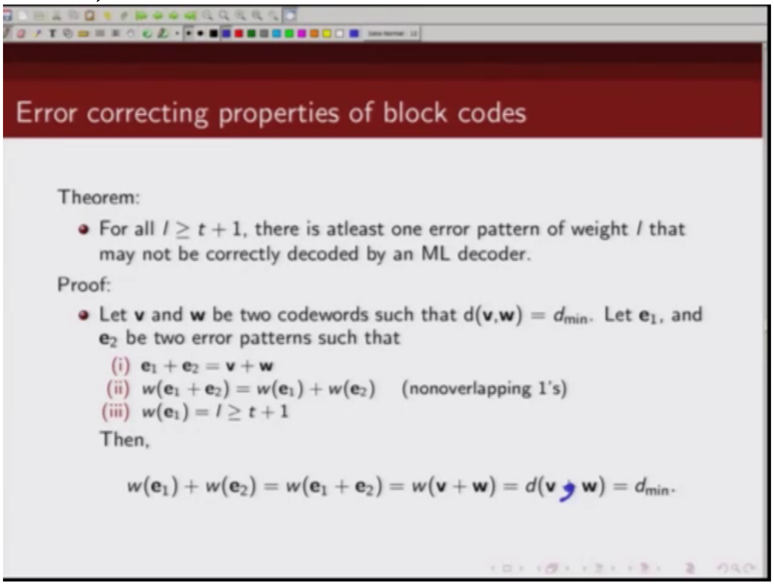
and we have said the Hamming distance between v and w is the minimum distance. So this is equal to the minimum distance. Now let us assume that

(Refer Slide Time 25:02)



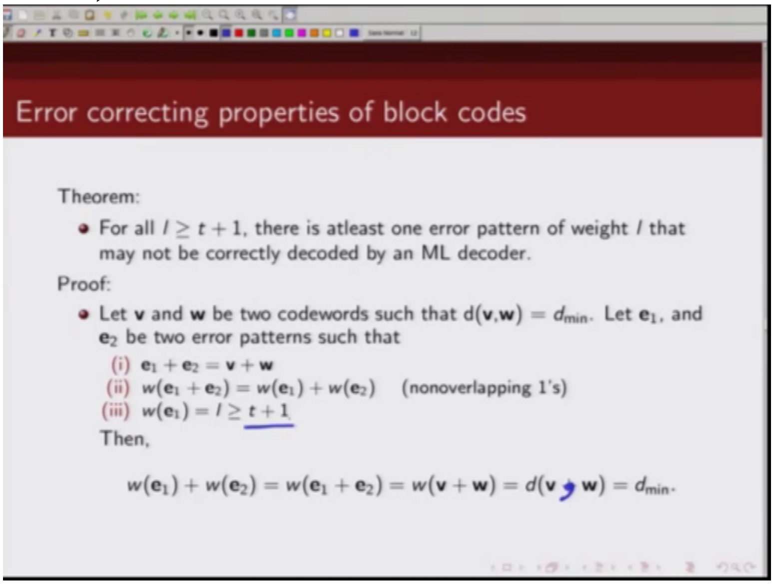
we transmitted this codeword v and what we received is r. So this v got corrupted by this error pattern e 1 which has

(Refer Slide Time 25:15)



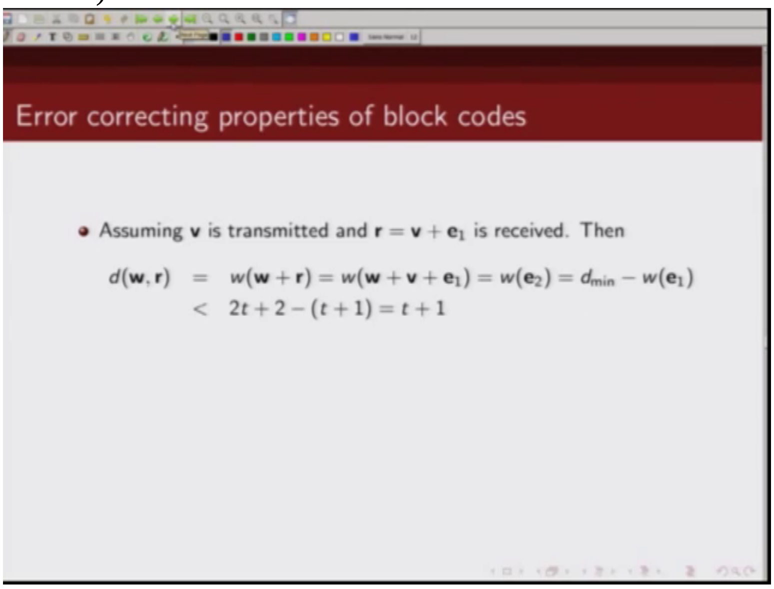
Hamming weight of at least

(Refer Slide Time 25:18)



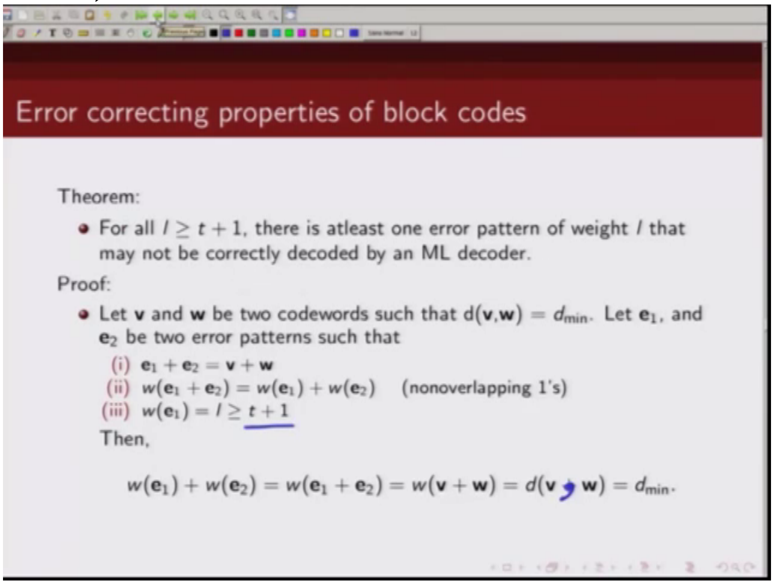
t plus 1. Now

(Refer Slide Time 25:21)



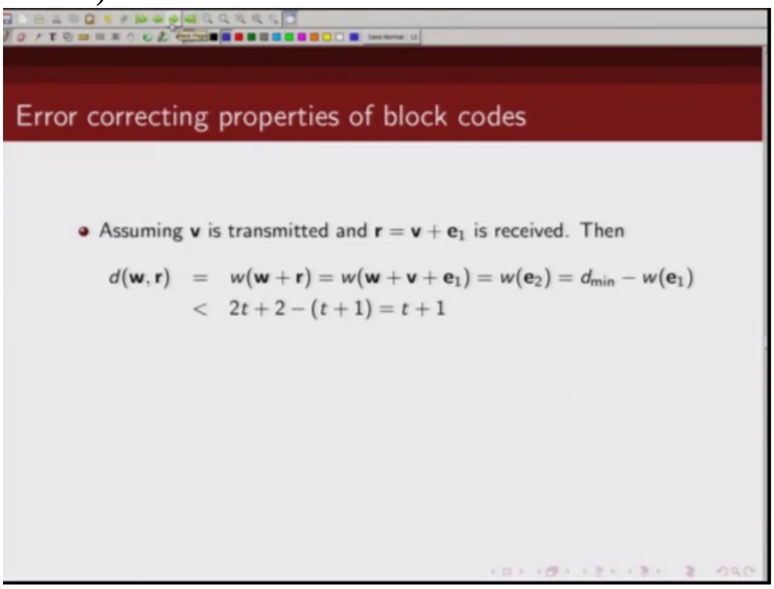
we will repeat the same exercise, we will try to find out the Hamming distance of this received codeword from the correct transmitted codeword v and Hamming distance between the received codeword and any other codeword w. So if we calculate the Hamming distance between w and the received codeword we know that Hamming distance between w and r is nothing but Hamming weight of w and r. And what is r? r is my received codeword, v plus e 1. So I can write this as w plus v plus e 1. Now what is w plus v? From 1,

(Refer Slide Time 26:11)



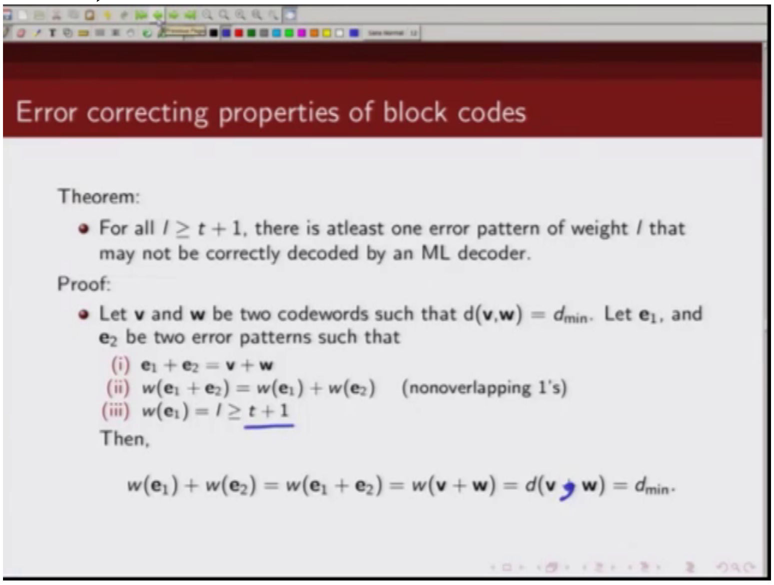
I have w plus v is same as e 1 plus e 2. So then this is

(Refer Slide Time 26:17)



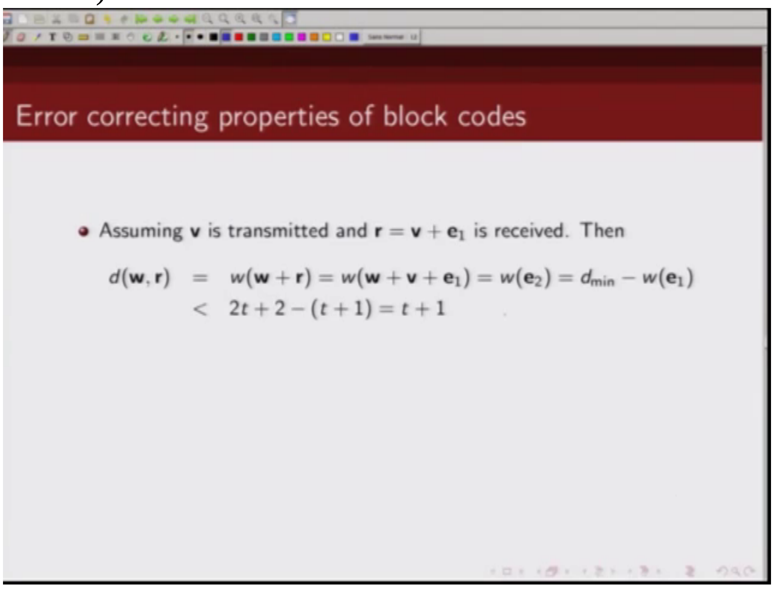
e 1 plus e 2 plus e 1. So e 1 plus e 1 will be 0. So this will be e 2, weight of e 2. And what is weight of e 2? From this relation

(Refer Slide Time 26:31)



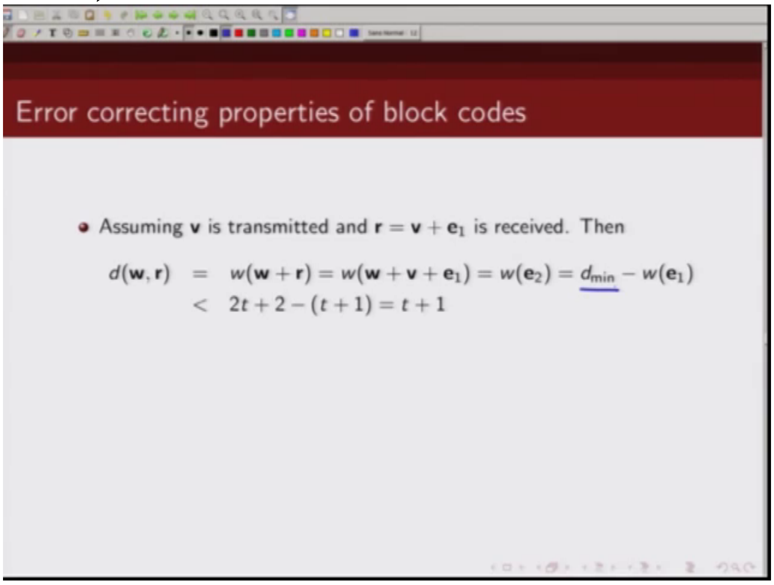
we can see weight of e 1 plus weight of e 2 is d min. So weight of e 2 is d min minus weight of e 1. So this we can

(Refer Slide Time 26:44)



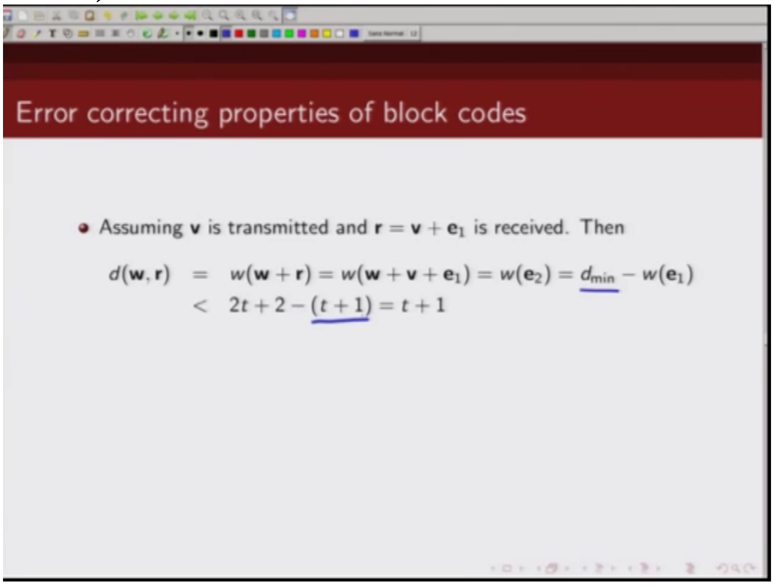
write as weight of e 2 as d min minus weight of e 1. So d min is less than equal to 2 t plus 1

(Refer Slide Time 26:53)



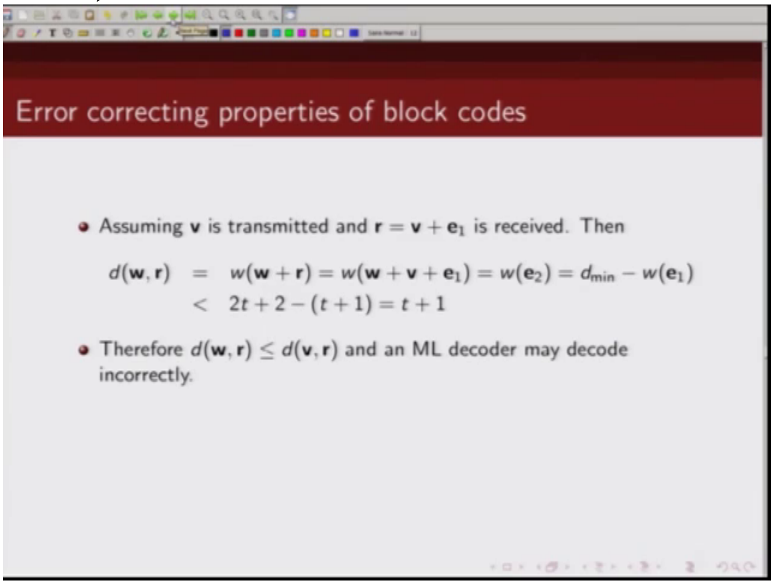
and weight of e 1 is at least t plus 1. So

(Refer Slide Time 26:58)



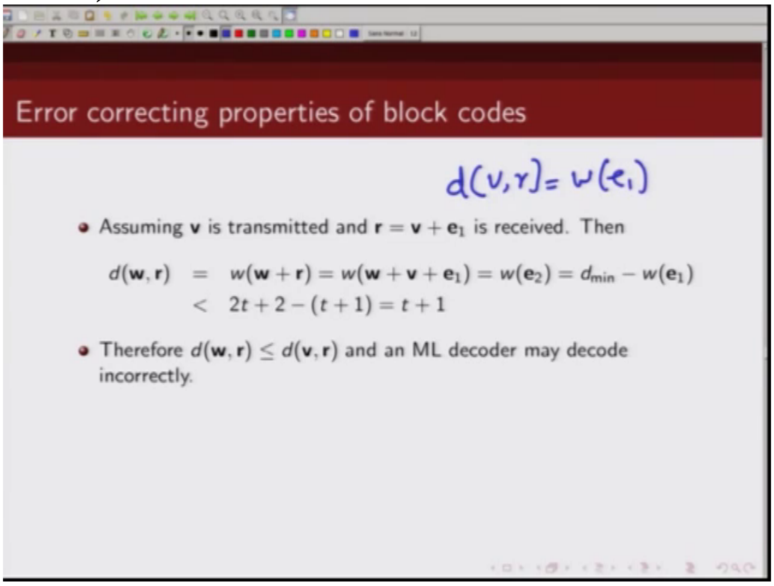
weight of e 2 will be less than 2 t plus 2 minus t plus 1 which is t plus 1. So the Hamming distance between w and r is less than t plus 1. And

(Refer Slide Time 27:15)



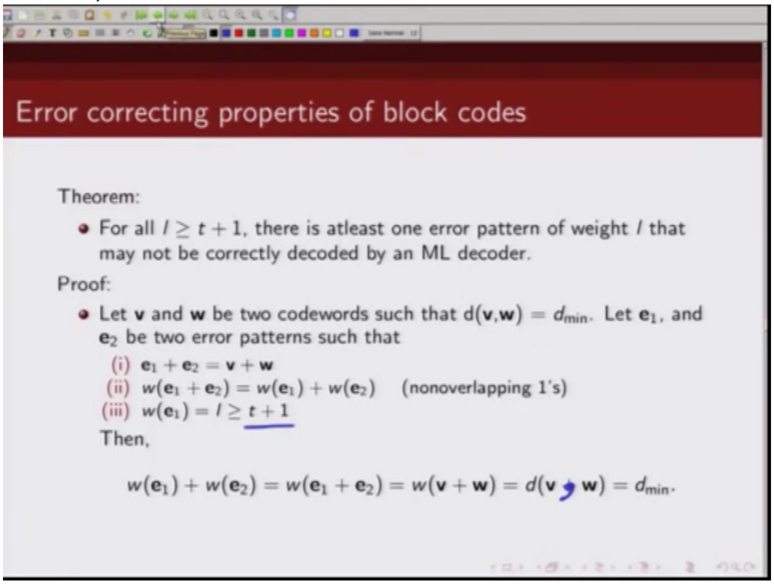
what is the Hamming distance between v and r? This is weight of e 1,

(Refer Slide Time 27:28)



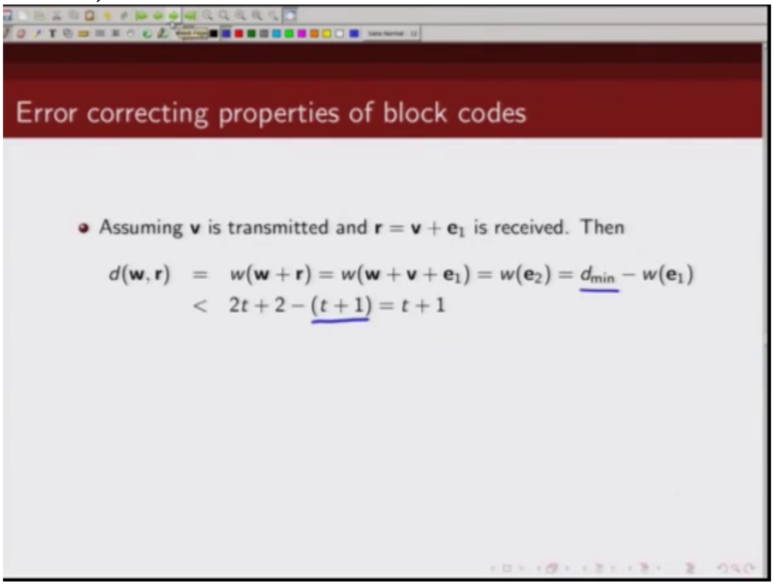
Ok and what is weight of e 1? Weight of e 1 is given by l, which is

(Refer Slide Time 27:36)



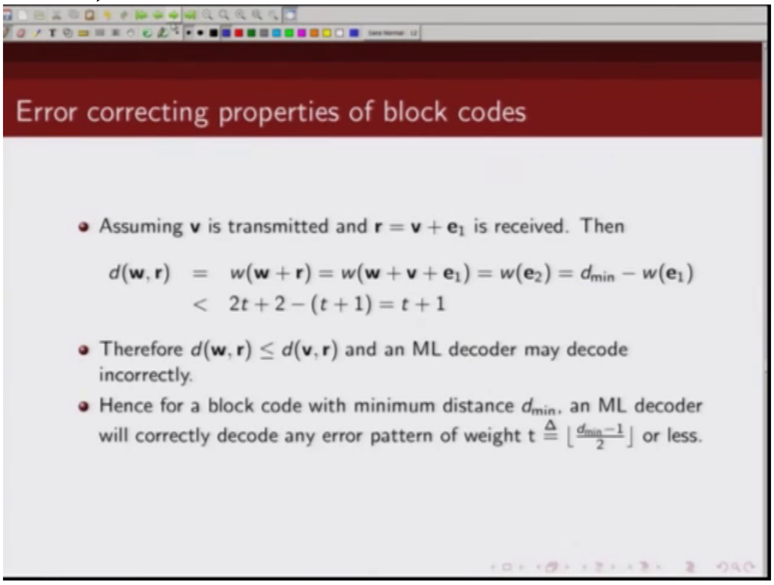
at least t plus 1. So what we

(Refer Slide Time 27:40)



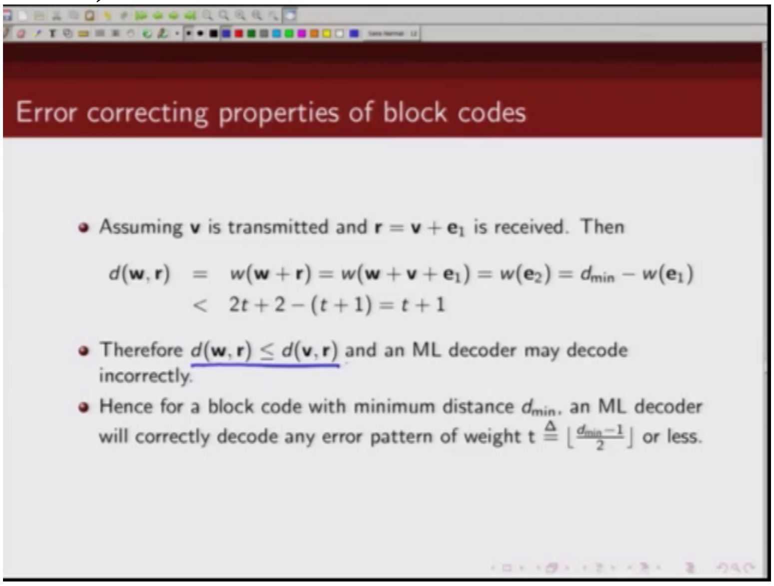
have shown here is

(Refer Slide Time 27:43)



weight of w, Hamming distance between w and r is

(Refer Slide Time 28:01)

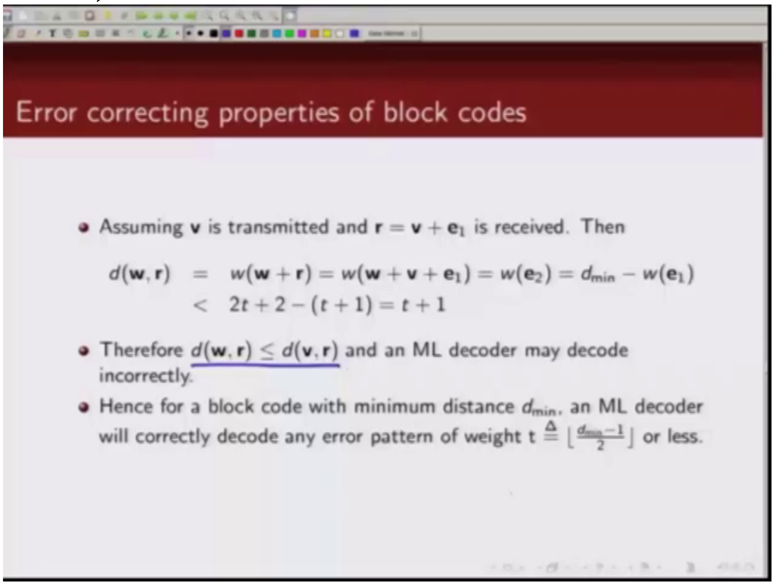


less than t plus 1 where as Hamming distance between v and r is greater than equal to t plus 1. So what we have shown is Hamming distance between w and r is less than equal to Hamming distance between received codeword r and the true codeword which was actually transmitted which is v. So in this case the maximum likelihood decoder will decode in favor of w and not v and will make a mistake. So through this construction

(Refer Slide Time 28:23)

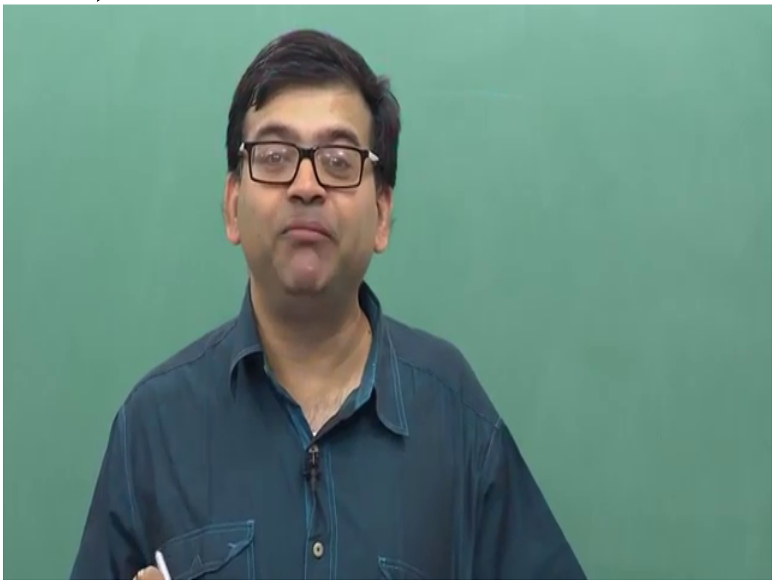
we have shown that if your error pattern is of weight t plus 1, then you are not guaranteed to correct that error. So from this and the previous result we can conclude

(Refer Slide Time 28:36)



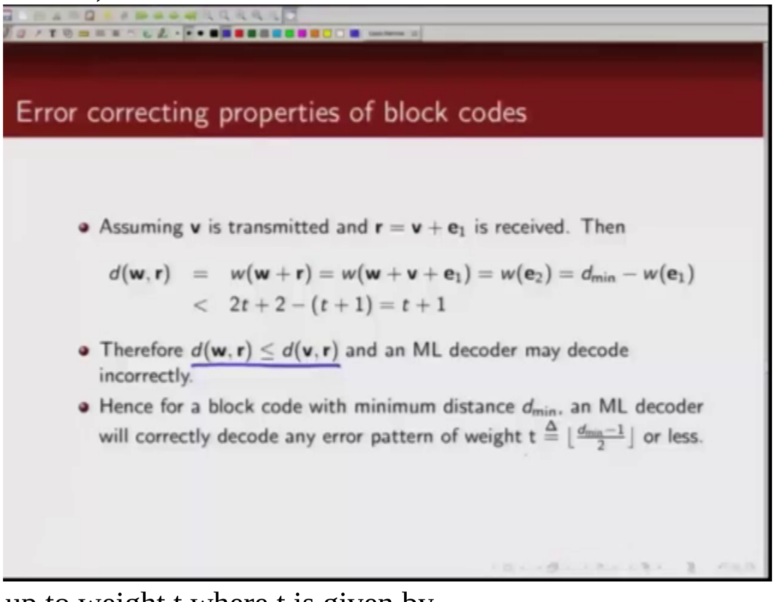
that if we have a block code with minimum distance d min which satisfies relationship that d min

(Refer Slide Time 28:45)



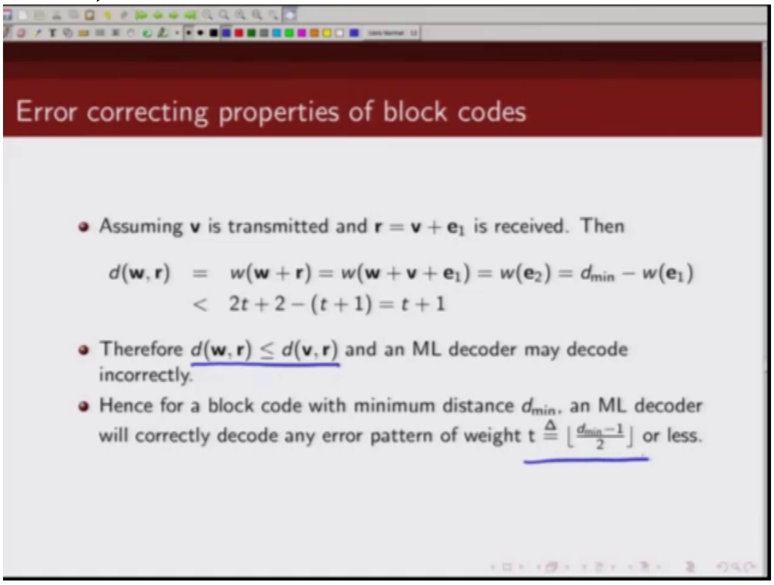
lies between 2 t plus 1 and 2 t plus 2 then this linear block code with minimum distance d min should be able to correct

(Refer Slide Time 28:56)



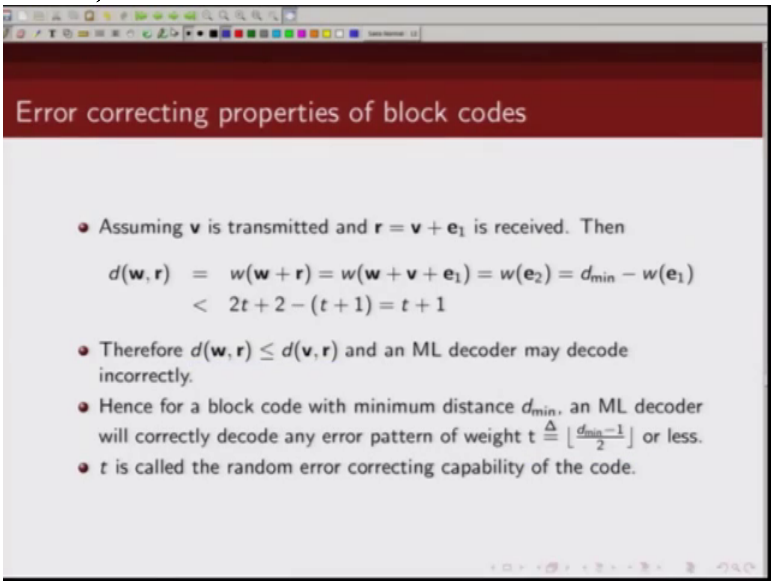
all error patterns up to weight t where t is given by

(Refer Slide Time 29:03)



this. So this t is

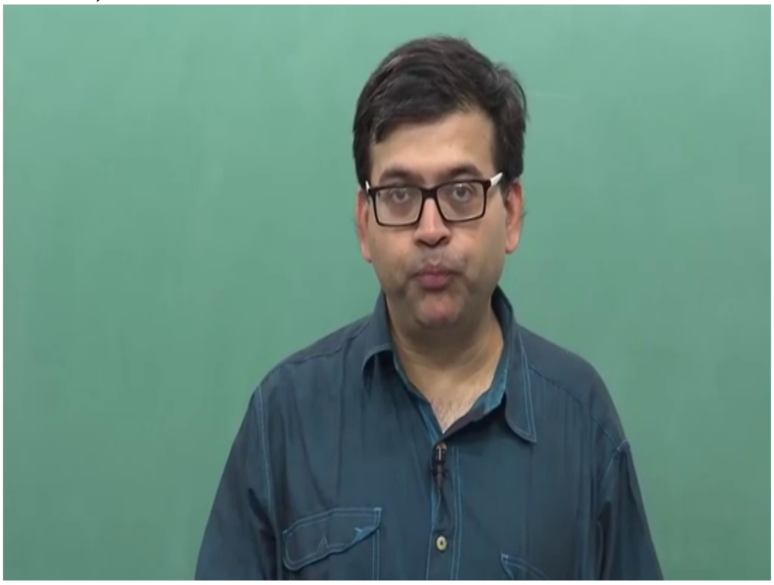
(Refer Slide Time 29:06)



known as random error correcting capability of the linear block code.

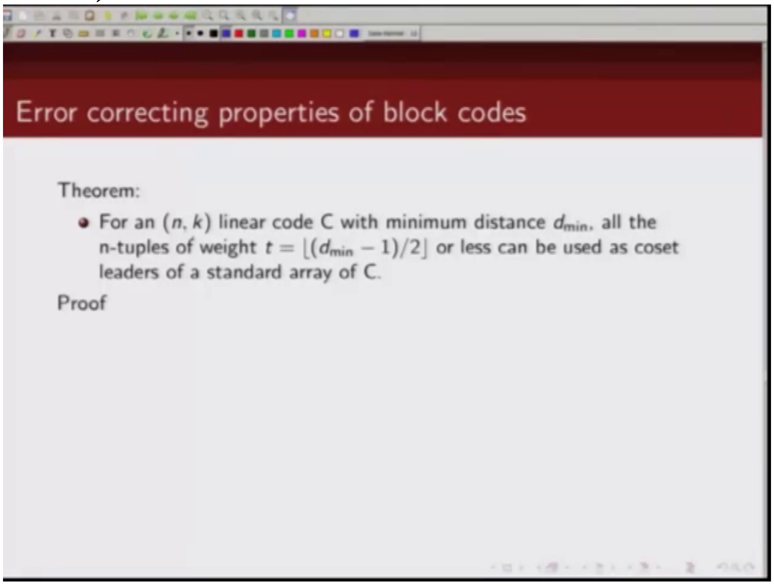
Next we are going to prove a result

(Refer Slide Time 29:15)



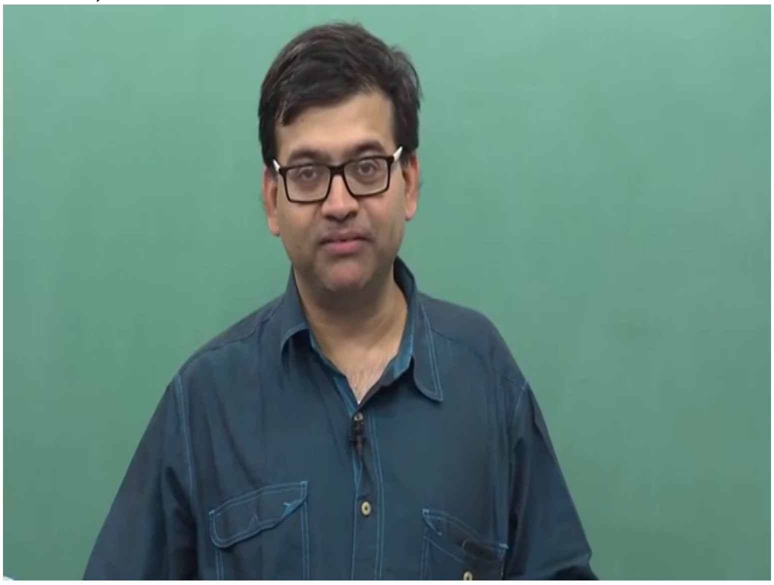
which is as follows. So if we have an n k linear block

(Refer Slide Time 29:20)



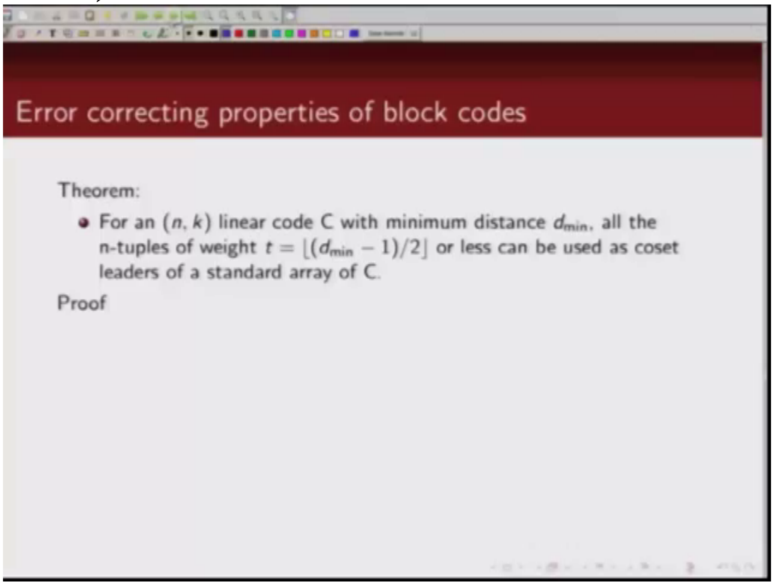
code whose minimum distance is given by t min then we can show where d min lies between 2 t plus 1 and 2 t plus 2, then we can show that all end tuples of weight t or less can be used as coset leader in our standard array. So we are going to prove this result using method of contradiction.

(Refer Slide Time 29:50)



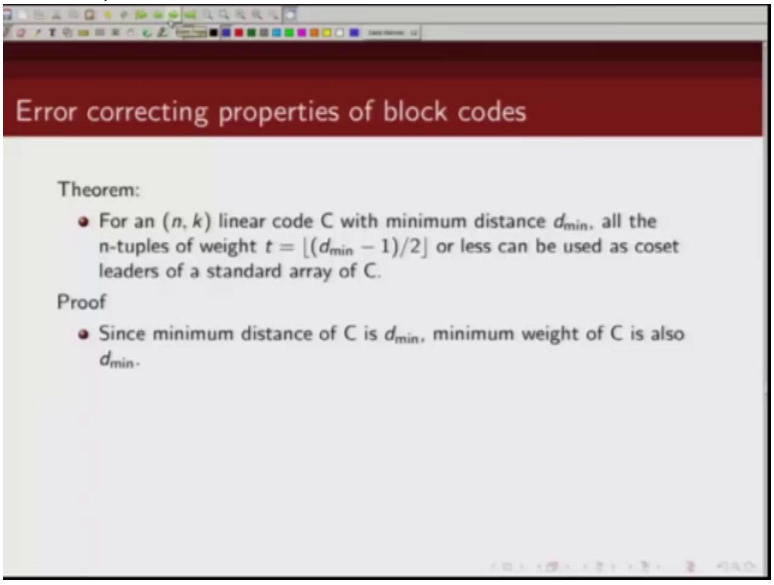
Now let's say, so how method of contradiction work. We will say, let's say they are all error patterns or weight up to t; let's say they are not coset leaders. Let's say, we will assume a scenario where there are 2 such end tuples with weight up to t which are not coset leaders. In other words they lie in the same coset or same row. And then later on we will show that that is not possible. So that's how this method of contradiction will work

(Refer Slide Time 30:27)



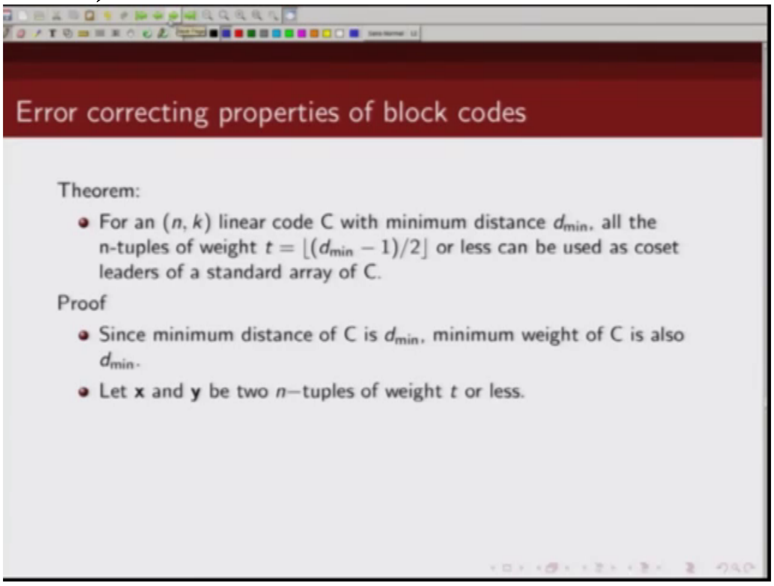
will work

(Refer Slide Time 30:28)



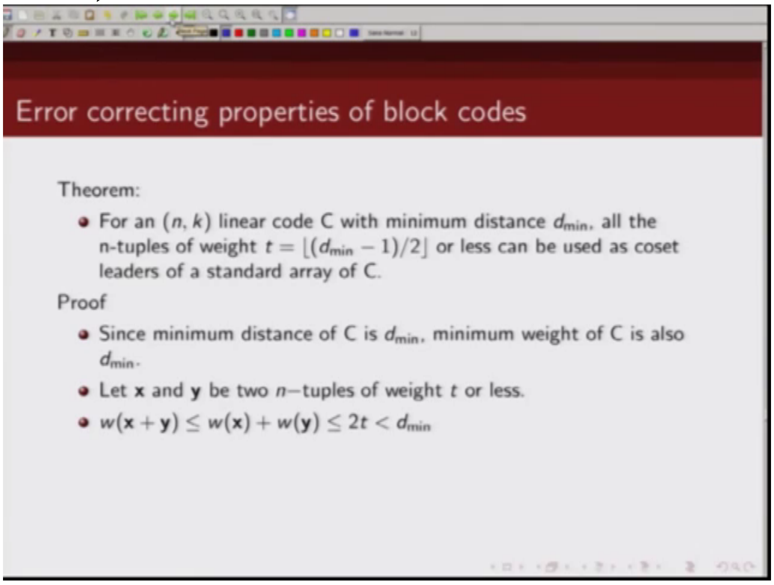
so minimum distance of the code is d min so minimum weight of the code is also d min.

(Refer Slide Time 30:37)



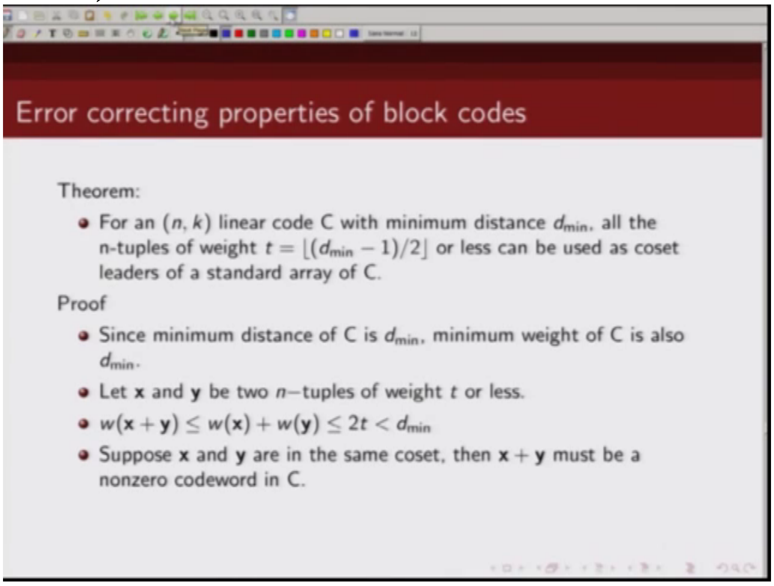
Let x and y are 2 n-tuples of weight t or less. Now

(Refer Slide Time 30:45)



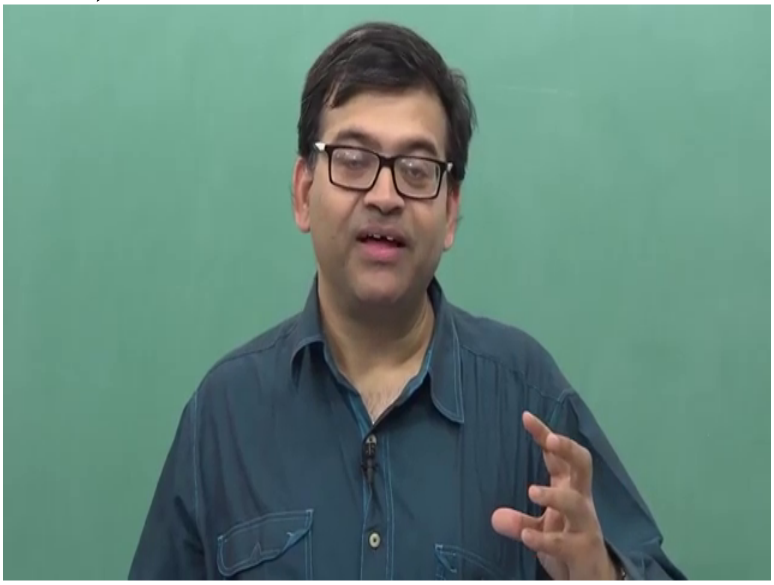
weight of x plus y will be less than equal to weight of x plus weight of y. Why, because there might be some overlapping 1s at some locations of this n-tuple x and y and we are given that the weight of x and weight of y is at most t so then weight of x plus weight of y will be less than equal to 2 t and this is less than minimum distance because minimum distance of code is at least 2 t plus 1. Now let us assume

(Refer Slide Time 31:25)



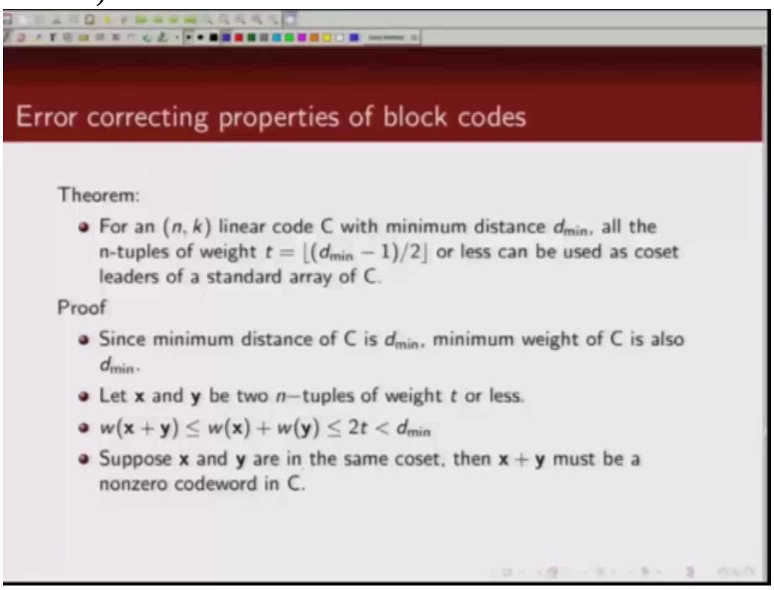
that these x and y which are

(Refer Slide Time 31:29)



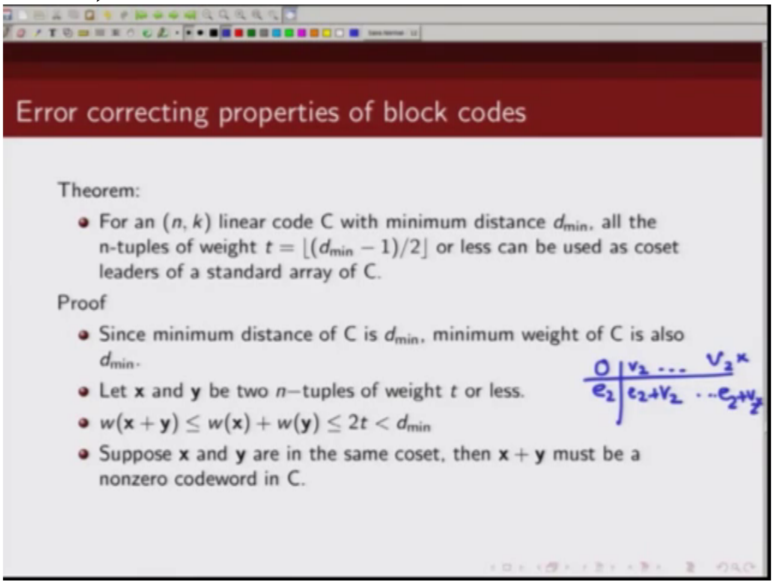
error patterns of weight t or less, let us assume that they are not coset leaders. If they are not coset leaders, let us assume they are in the same coset; they are in the same row. So if we assume x and y are in

(Refer Slide Time 31:45)



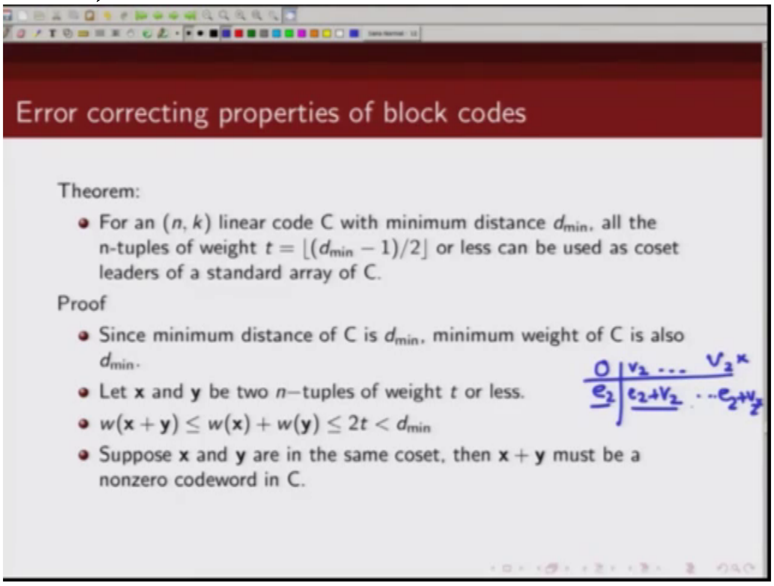
the same row or same coset, then x plus y must be a codeword. Why this is so? If you recall your standard array we had something like this. First row first column was all zero vector and then we had other codewords. And then we had error pattern, let's say e 2. This was e 2 plus v 2. Like, like this was e 2 plus v 2 k. If you look at

(Refer Slide Time 32:27)



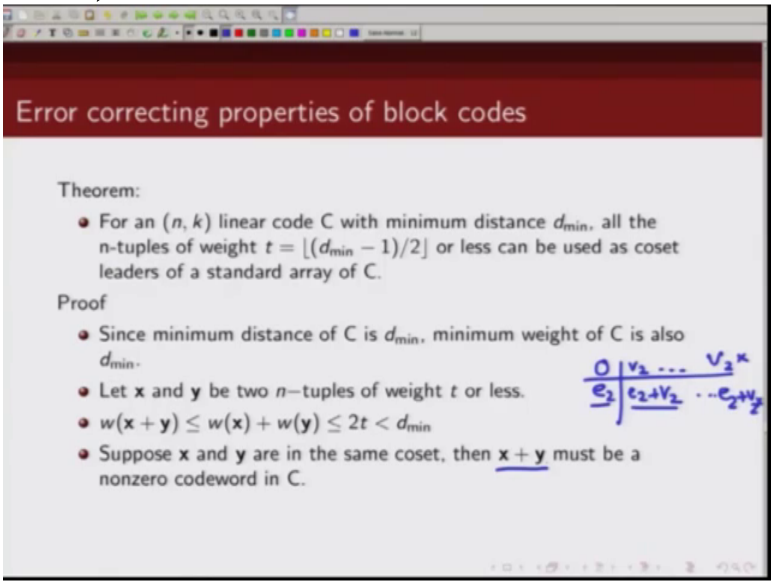
any 2 elements in the same coset or same row and if you add them up what do you get? Let's add this and this, what do we get? e 2 plus e 2 plus v 2,

(Refer Slide Time 32:41)



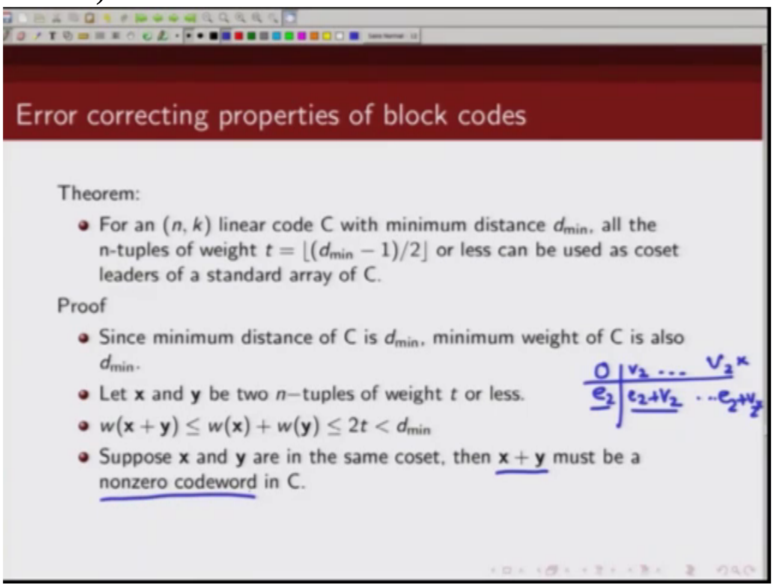
we will get v 2. If we add this and this we will get v 2 plus v 2 k which is another codeword v s. So if we take any two elements in the same coset and we add them up we are going to get a

(Refer Slide Time 33:01)



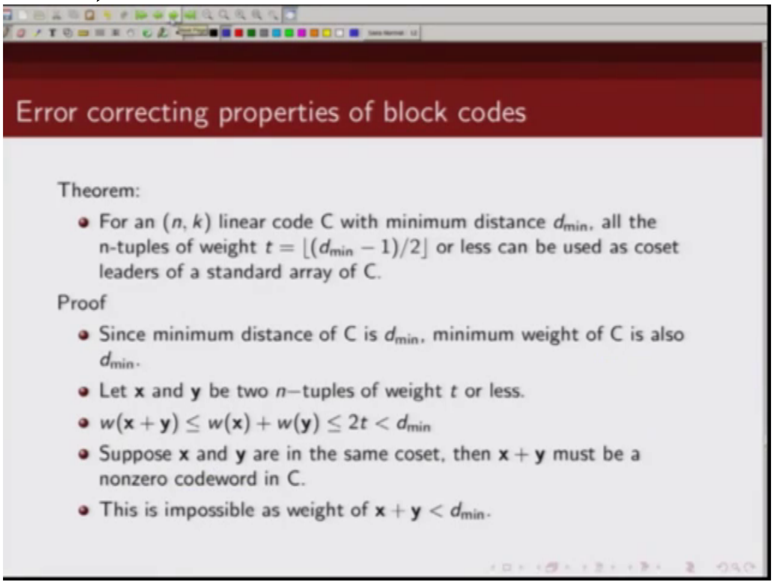
non zero codeword.

(Refer Slide Time 33:03)



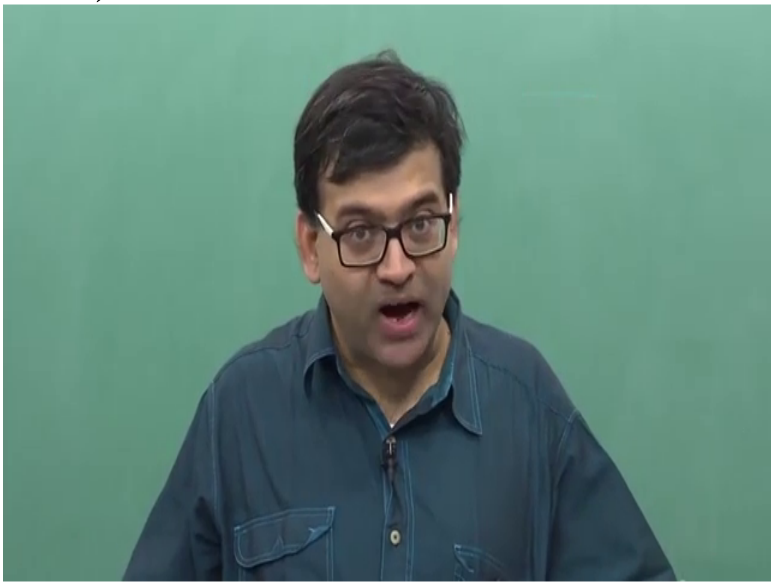
So if x and y are in the same coset then x plus y must be a codeword. This is impossible.

(Refer Slide Time 33:15)



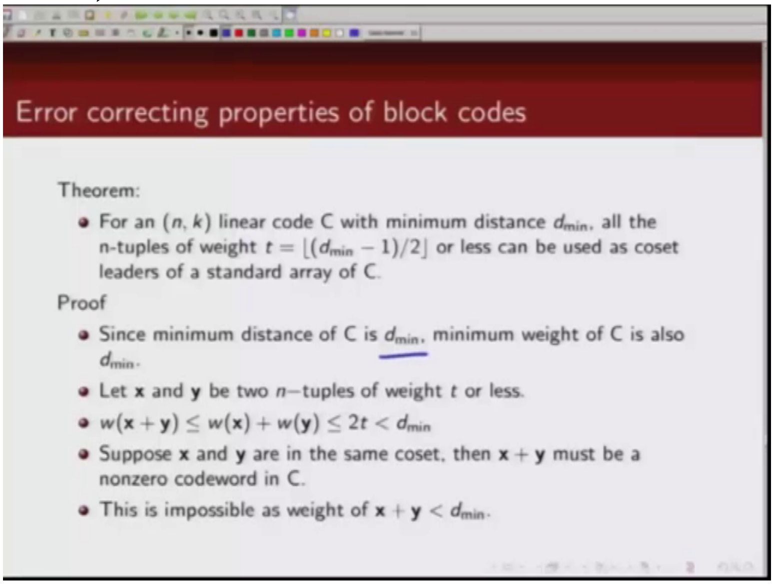
Why? Because if x plus y is a

(Refer Slide Time 33:19)



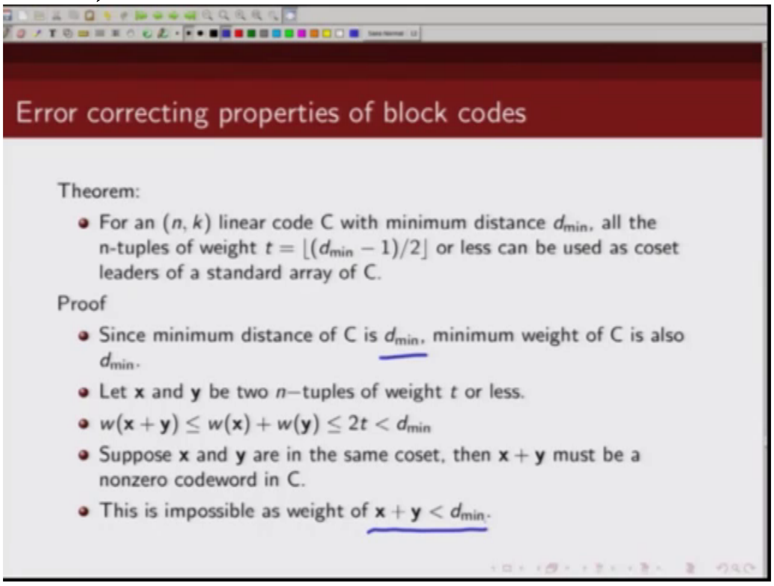
codeword then what is the minimum distance of x plus y? x plus y, minimum distance of that must be d min.

(Refer Slide Time 33:28)



But what is the, what is the weight of x plus y, we just showed in this bullet that weight of x plus y is less than d min. That means weight of x plus y is less than d min.

(Refer Slide Time 33:44)

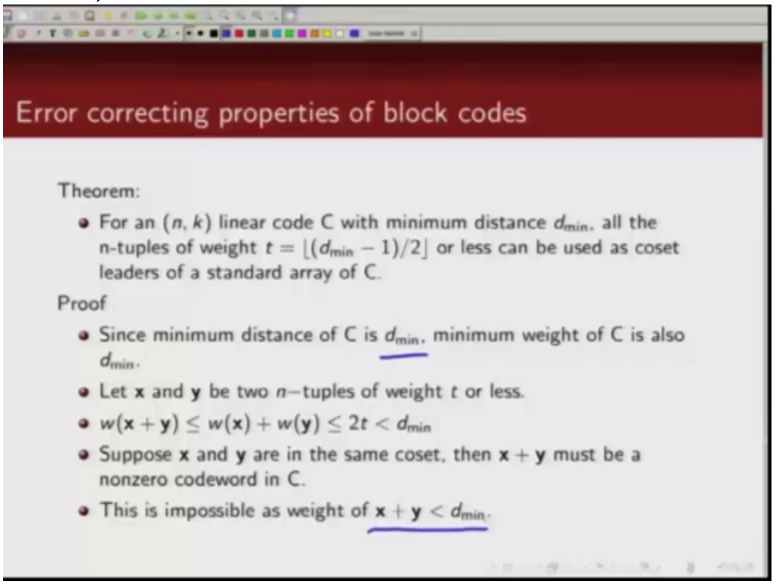


If weight of x plus y is less than d min then x plus y cannot

(Refer Slide Time 33:50)

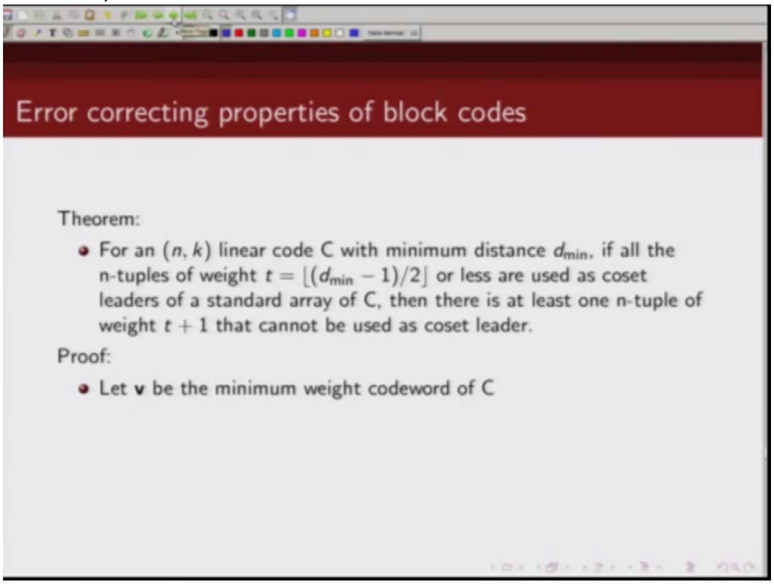
be a non zero codeword because the weight of a non-zero codeword should be at least d min. So our assumption that x and y are in the same coset is wrong. In other words then x and y must be in different cosets, different rows and we can always make these x and y as coset leaders. So this proves our result that

(Refer Slide Time 34:27)



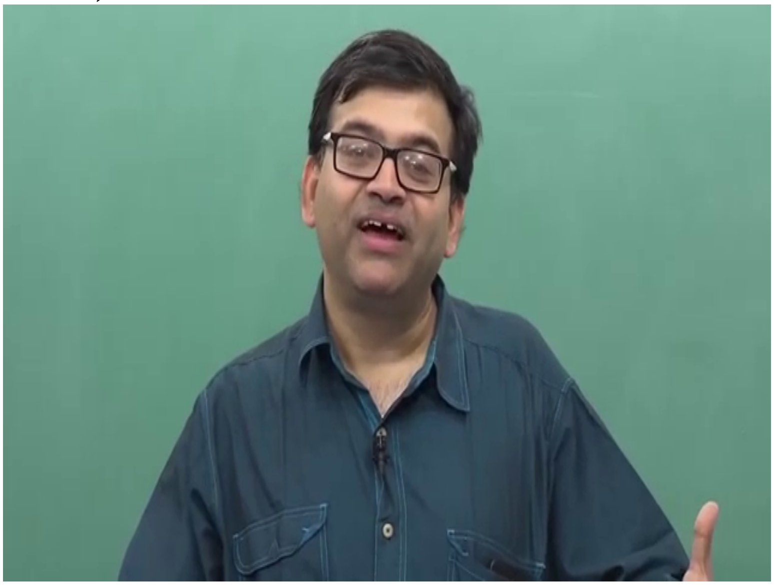
all n-tuples of weight n, of weight t or less can be used as coset leaders in the standard array and we know that uh if we use them as coset leaders, we, those are our correctable error patterns.

(Refer Slide Time 34:46)



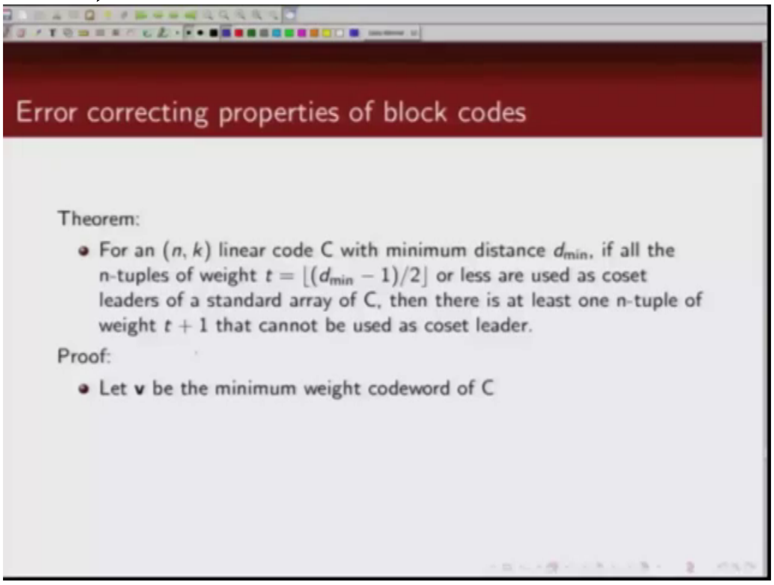
Next I am going to show you a result which is as follows. So if you have a n k linear block code whose minimum distance is d min and if all n-tuples of weight t or less are already used as coset leader then there is at least 1 n-tuple of weight t plus 1 which cannot be used as coset leader. So this essentially is going to show us again the same result that

(Refer Slide Time 35:21)



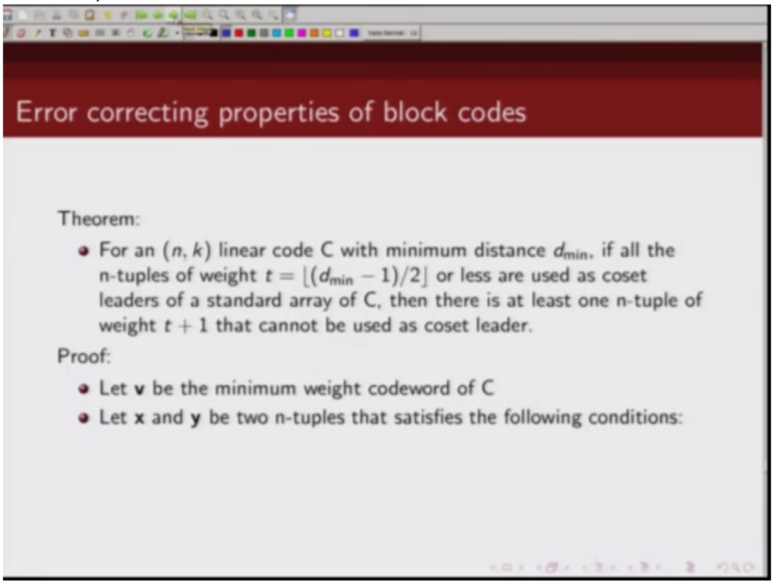
any weight pattern of, error pattern of weight t plus 1 is not guaranteed to be corrected. So how do we

(Refer Slide Time 35:31)



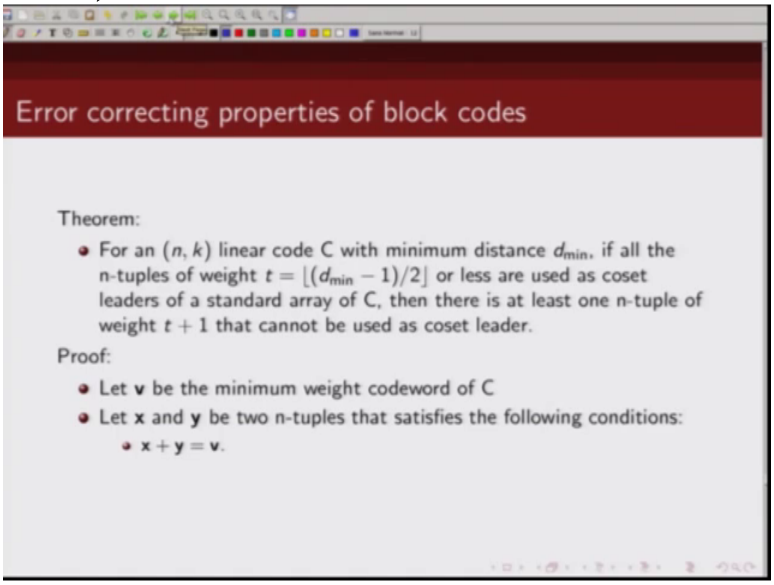
prove it? So let's assume v is the minimum weight codeword of C

(Refer Slide Time 35:38)



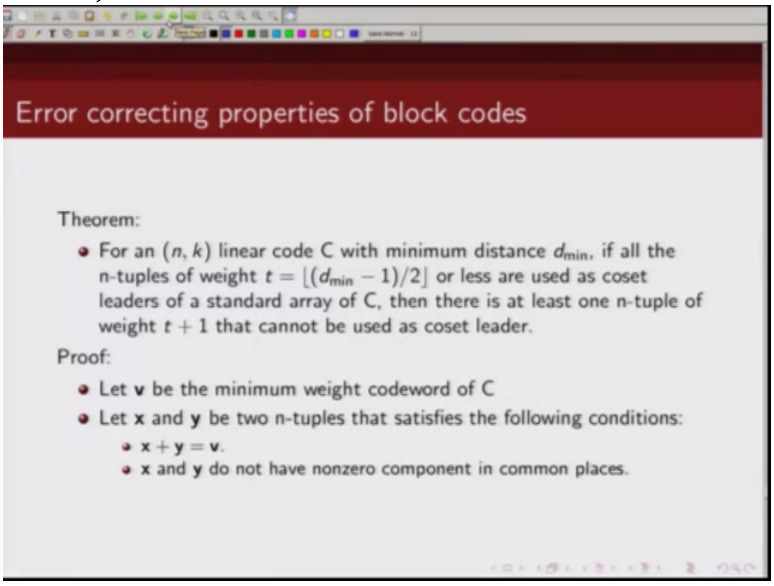
and we have 2 n-tuples x and y which satisfies these following conditions.

(Refer Slide Time 35:45)



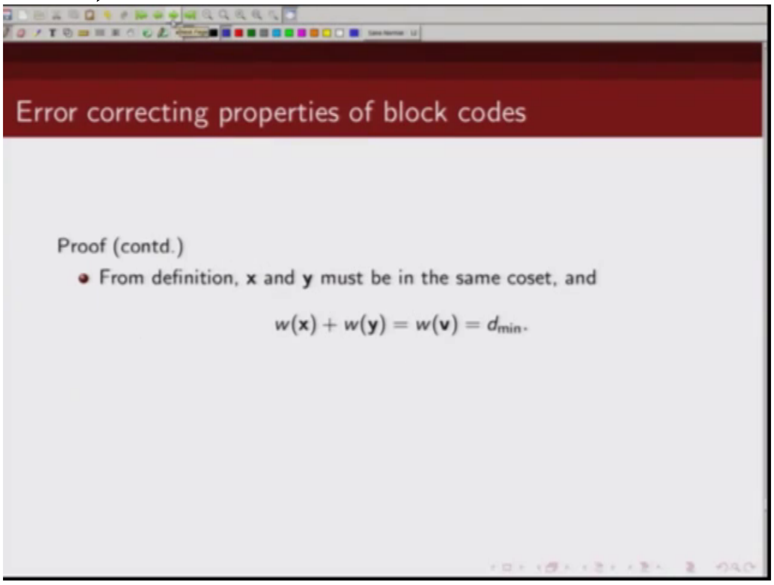
First, x plus y is equal to v, and x and y do not

(Refer Slide Time 35:54)



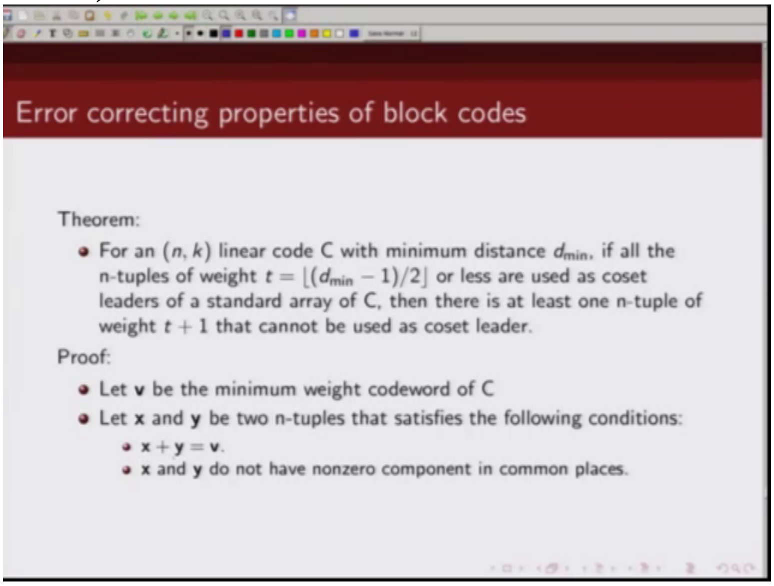
have any component common. So they do not have 1s common in same position. So

(Refer Slide Time 36:04)



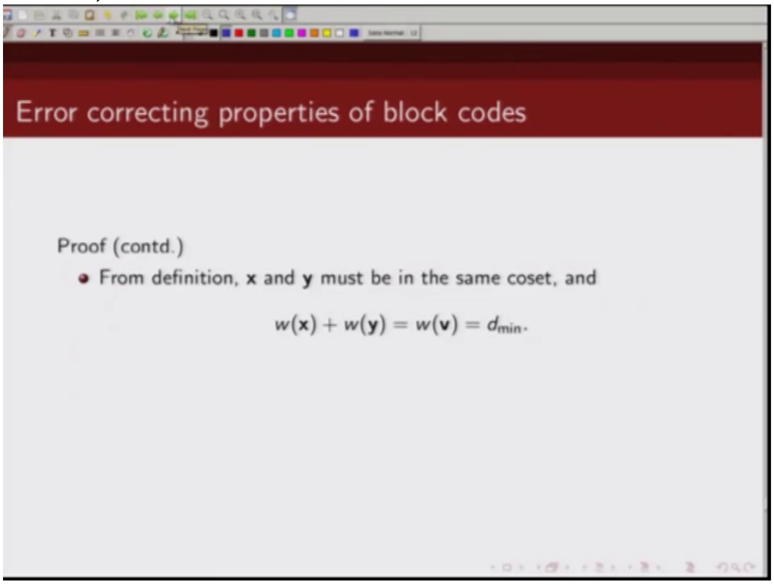
from the definition x and y

(Refer Slide Time 36:07)



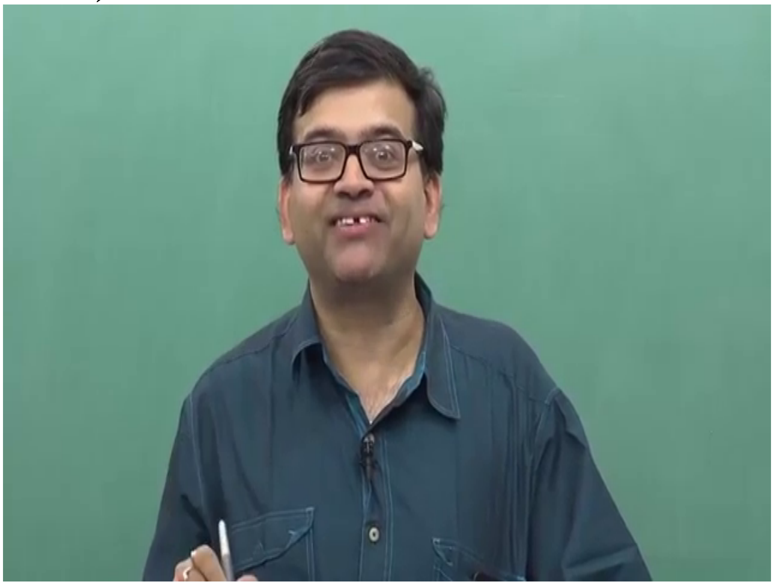
must be in the same coset. Why? Because we know if two elements are in the same coset and if we add them sum is a valid codeword. So x plus y is equal to v which is a valid codeword, then x and y must be in the same coset.

(Refer Slide Time 36:32)



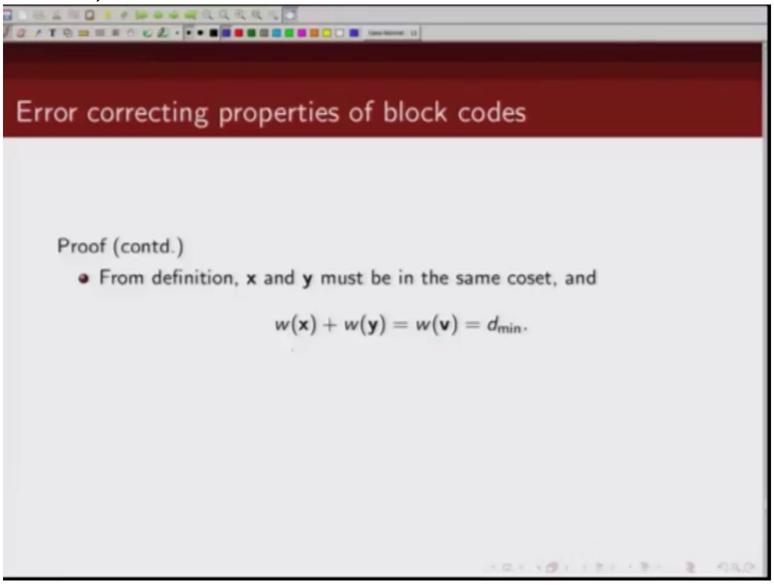
So that's what I said from definition x and y must be in the same coset because x plus y

(Refer Slide Time 36:42)



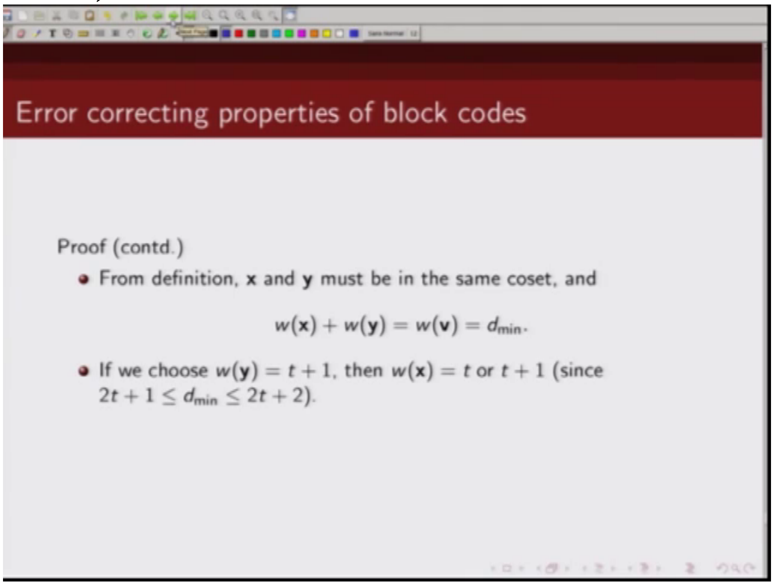
is v which is a valid codeword. And we know that if we add any two elements in a coset their sum is a valid codeword. And

(Refer Slide Time 36:54)



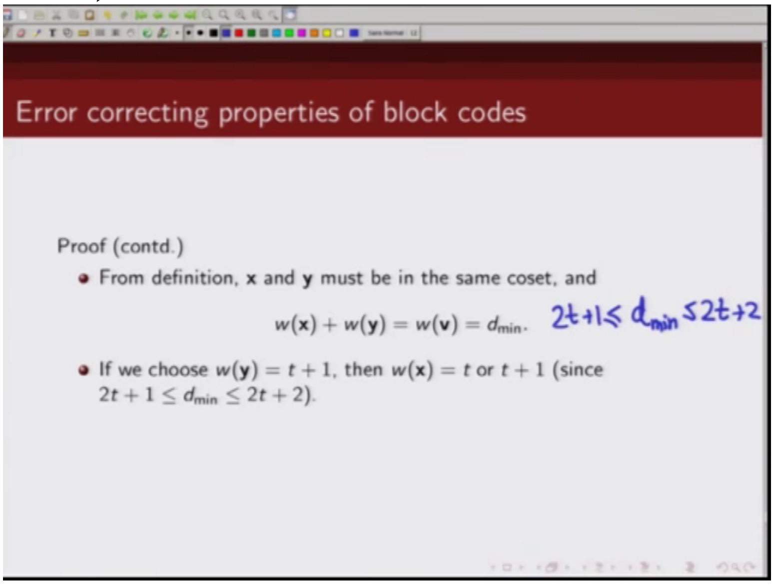
similarly weight of x plus weight of y is equal to weight of v. And we have chosen v to be the minimum distance codeword, so this is given by d min. Now if we choose

(Refer Slide Time 37:09)



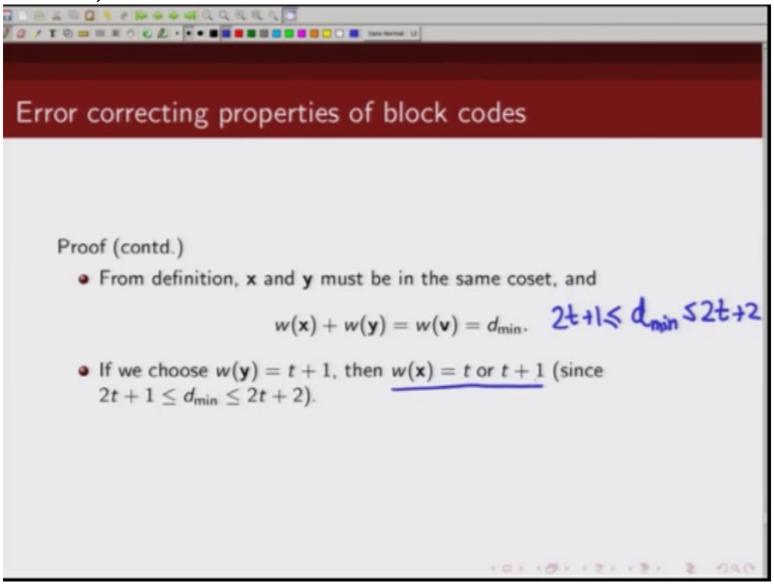
our y to have a weight of t plus 1, then we can see from here d min is greater than equal to 2 t plus 1 but less than equal to 2 t plus 2. So from this and using the fact that d min lies between 2 t plus 1 and 2 t plus 2,

(Refer Slide Time 37:39)



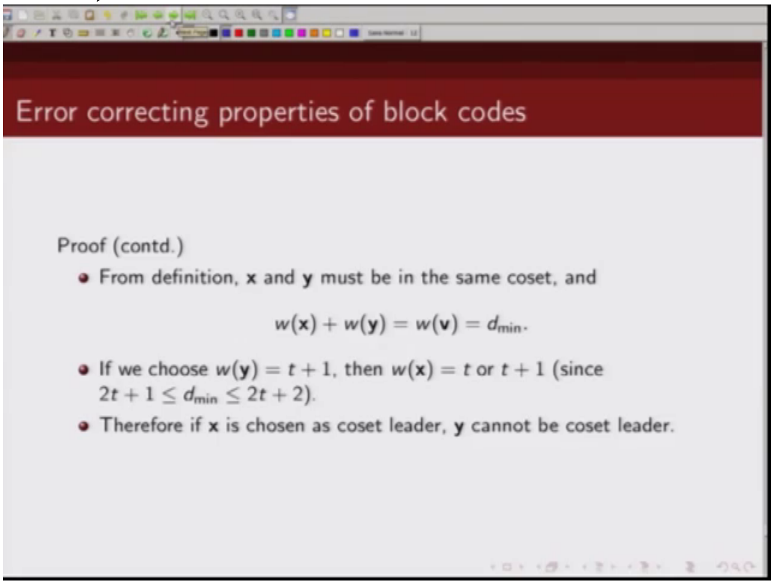
using these 2 results what we get is weight of x

(Refer Slide Time 37:45)



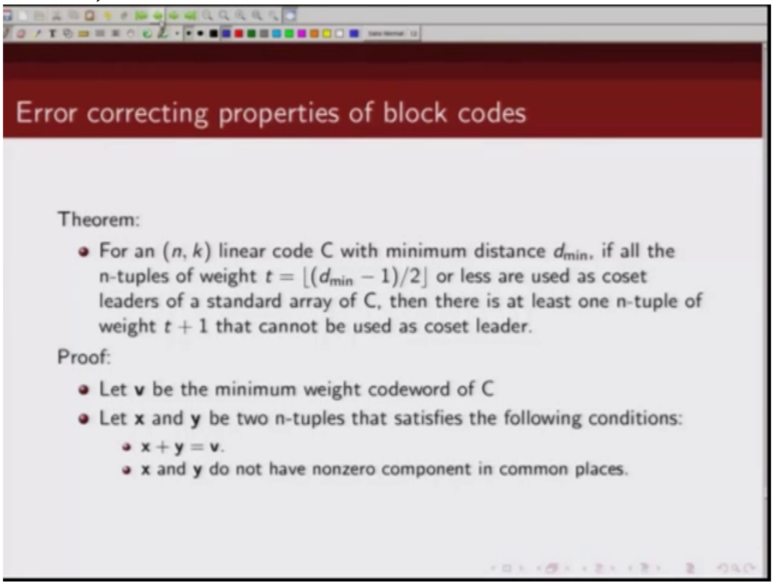
can be t or t plus 1.

(Refer Slide Time 37:53)



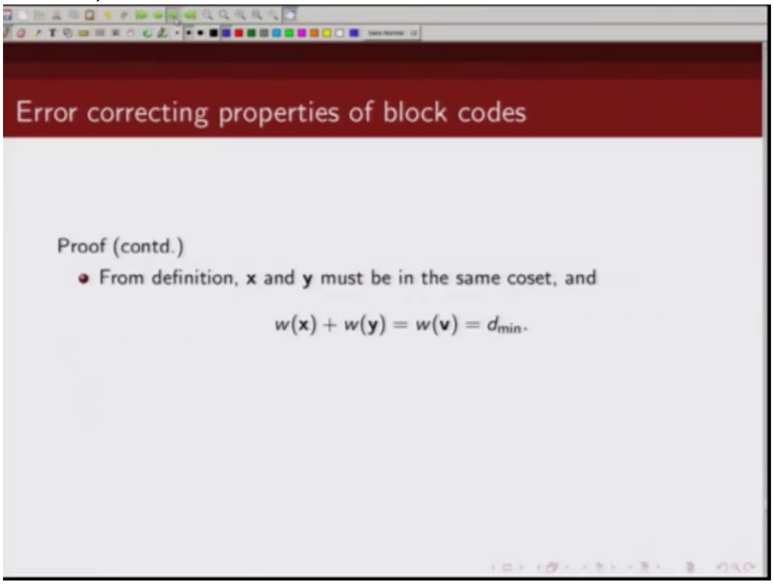
So therefore if we choose x to be our coset leader then we cannot choose y as our coset leader. You can see, because x and y are in the same coset and weight of x is t or t plus 1 whereas weight of y is t plus 1. So I will choose x as my coset leader. And if I choose x as my coset leader then I cannot choose y as my coset leader which proves my result

(Refer Slide Time 38:27)



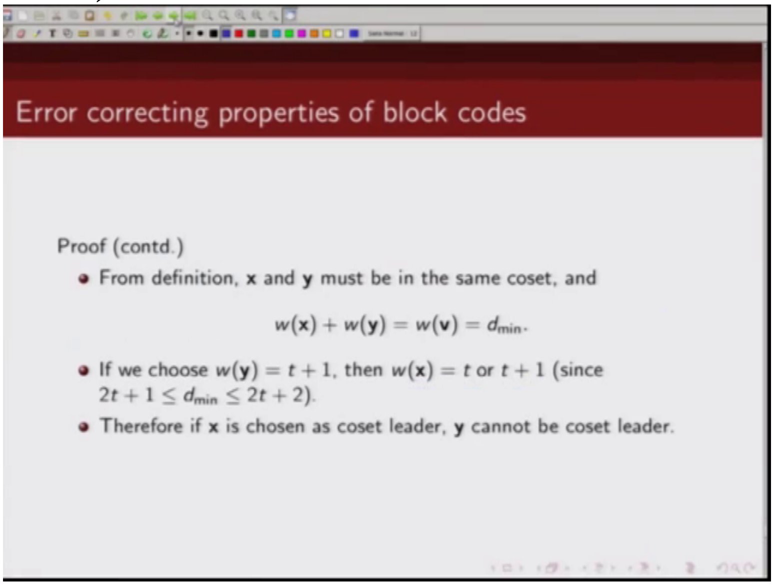
which says that if all end tuples of weight t or less are used as coset leaders then there exist at least one error pattern of weight t plus 1 which cannot be used as coset leader and if this error pattern

(Refer Slide Time 38:44)



pattern

(Refer Slide Time 38:45)

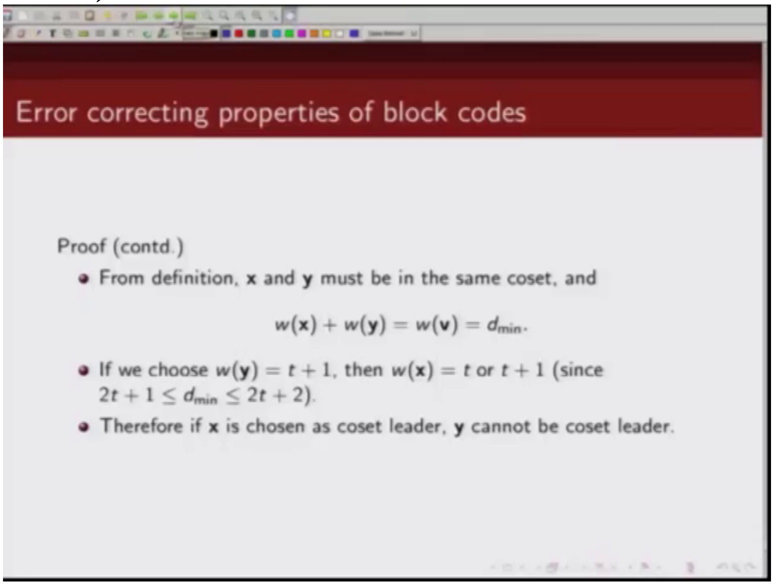


of weight t plus 1 cannot be put

(Refer Slide Time 38:49)

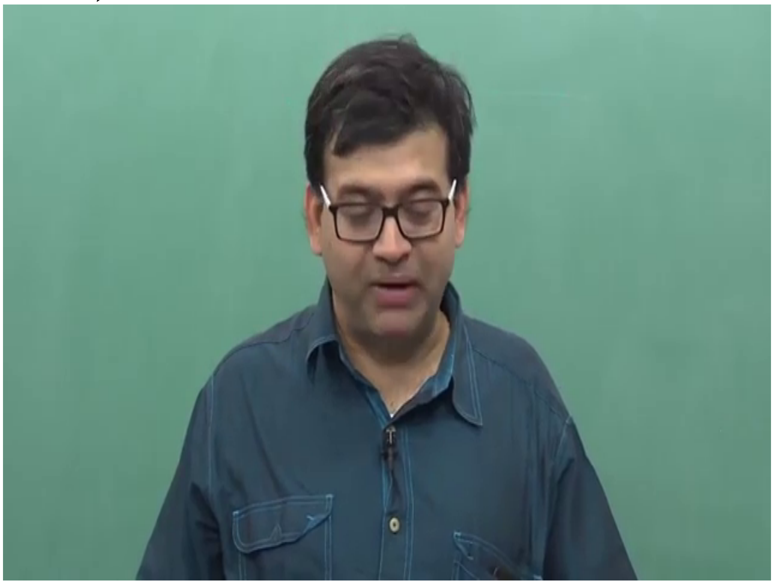
as coset leader then this is not a correctable error pattern.

(Refer Slide Time 38:54)



So with this, I will conclude my

(Refer Slide Time 38:57)



lecture on random error correcting and random error detecting properties of block codes. Thank you