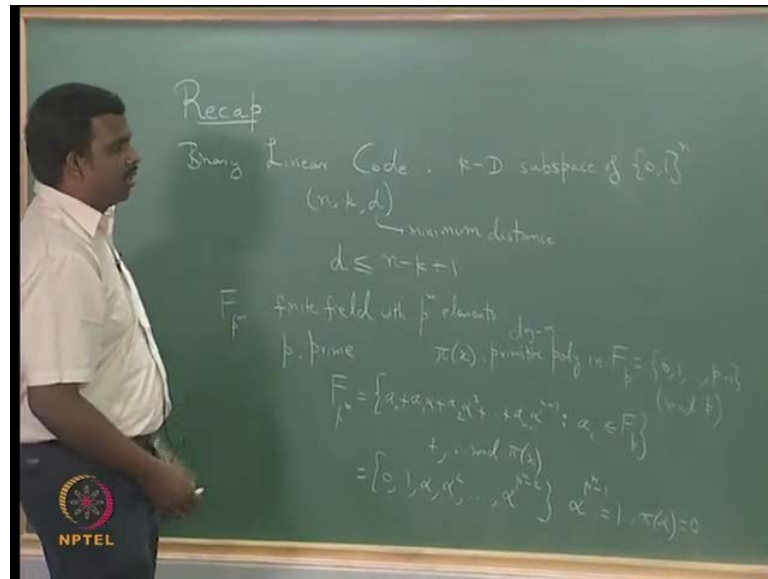


**Coding Theory**  
**Prof. Dr. Andrew Thangaraj**  
**Department of Electronics and communication Engineering**  
**Indian Institution of Technology Madras**

**Lecture - 13**  
**BCH and RS codes I**

(Refer Slide Time: 00:13)



So, let us do a quick recap. So, the first thing is well I guess, this is kind of summary of whole code. So far, we began with a linear code, binary linear code which is basically a subspace of, k dimensional subspace of, right. This is how we started and we have a notation for it n k d and d was the minimum distance, was the definition for that. And all possible n k d, I mean all kind of n k d are not possible for given n and k, there are some pounce and how height it can be. In case one of the bound is basically, the single ten bound say d is less than equal to n minus k plus 1.

In case this bound is applies, it can be used for linear codes, non-linear code, binary codes, non-binary codes anything you want. So, it is a very, very interesting and important bound. This codes that meet the bound are called maximum distance separable, it is that some where we saw, the binary codes that only two of those which meet the m d s bound. What are the two binary codes that meet the of m d s bound?

Student: ((Refer Time: 01:52))

Reputation code and two even weights. So, those two weights and then we saw this, a definition of this Galva field  $F_p$  to the  $m$ , this finite field with  $p$  power  $m$  elements. And  $p$  has to be a,  $p$  has to be a prime number and  $m$  is some integer, positive integer and how do you construct it? You construct it by beginning with  $a$ , for instance  $\pi$  of  $x$  which is the primitive polynomial in  $F_p$ .  $F_p$  is an easy enough field,  $F_p$  is what?  $0$  into  $p$  minus  $1$  modulo  $p$ , addition and multiplication. So, this is a easy enough field and this is a degree  $m$ , primitive polynomial and  $F_p$  I had a definition for it and also give you a result, that such polynomials always exist. So, we do not want worry about it system.

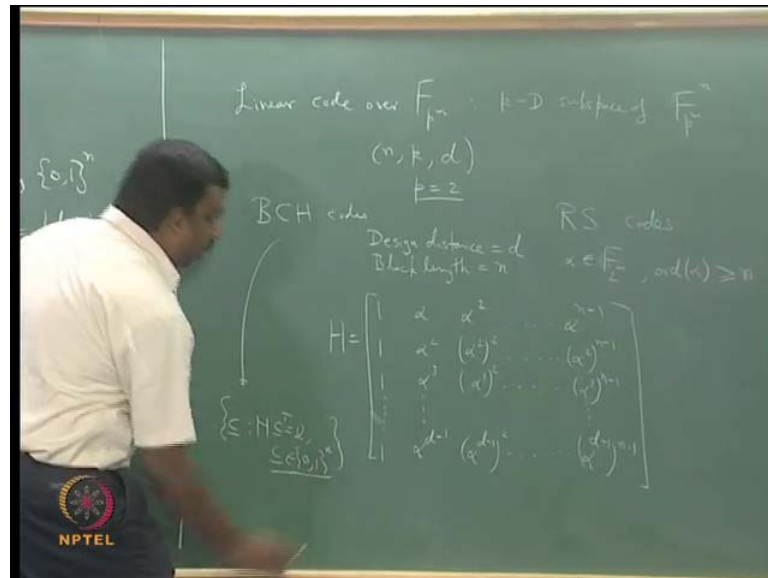
So, once you have that, you can construct  $F_{p^m}$  in kind of like a similar way, but in a different, slightly different style using polynomials. So, what is that, what is the definition? So, you have to look at polynomials, let us say some  $\alpha$ , right. You want to look at polynomial of degree less than or equal to  $m$  minus  $1$  and this  $a_i$  should come from  $F_p$ . So, this is, this is a useful way in which is to think about finite field  $f_{p^m}$ ,  $F_p$  to the  $m$  and here addition and multiplication you would do, modulo  $\pi$  of  $x$ . Once again, it is a degree  $m$  primitive polynomial, this addition is really not I mean modulo  $\pi$  of  $x$ , it does not play any role, addition is just vector or just polynomial addition, no problem.

But in multiplication is clearly  $\pi$  effects will play a role and the crucial another way you think about it is that is that there is one specific element in this case, you can take  $\alpha$  itself to be that specific elements, which will be primitive in this field. So, what does it mean? You can also, since this  $\pi$  effects is a primitive polynomial whatever I am using is  $\alpha$ , that itself will generate the entire field. So, what will it, what you can write this is, also is,  $0, 1, \alpha, \alpha^2, \alpha^3, \dots, \alpha^{p^m - 2}$  is such the half of to the power  $p$   $\pi^{m-1}$  is  $1$  and then what else,  $\pi$  of  $\alpha$  is equal to  $0$ .

If it is this is another equivalent way of describing the field. So, this notation is called rowan notation, this notation is called the vector notation or the polynomial notation. This is useful for multiplication, this is useful for addition, if you have table to go for one to the other you are well set in any finite field. So, this, this is the finite field in  $p$  to the power  $m$  elements, there are other results, there are no other finite field, any to finite fields of the same size are isomorphic etc. etc. which makes these very special structures. So, then what we will do? We went from binary linear codes to let us say, non-binary or ((Refer Time: 06:13)) or the larger code with the larger alphabet. So, that we saw some

better flexibility, as you could get other code which were also mbs I gave you some very simple examples.

(Refer Slide Time: 6:25)



So, that was the kind of motivation, how does the definition work? It is very simple, again a linear code over  $F_p^m$  is basically a  $k$  dimensional subspace of, instead of  $0, 1$  which is binary we would take  $F_p$  and  $n$ , which is the  $n$  dimensional vector space over the field,  $F_p$ . So, that is the way we think of the linear code again, once again, you will have  $n, k, n, d$ . So, it is a larger field you have more code words, minimum distance is define definitely and. So, that, there were curious to motivations to go to these finite field. So, one of the motivation was, you could not get, you could not you could not meet the bound in binary so that one of the motivations.

The other motivation which is also equally important is that, there is no easy way to construct parity check matrixes, for an arbitrary minimum distance. So, minimum distances is also related to the error correcting capability. So, the error correcting capability is, right. So, if I want able to correct let say, one error it is not too bad, but even for correcting two errors, which requires a minimum distance of what? A five, you need a minimum distance of five to correct two errors, it is not very easy to come up with a generic method for constructing parity check matrixes. That was the other problem, we could not really check minimum distance etcetera.

So, I said both problems are solved by moving to finite fields and defining parity check matrixes over finite field. So, in fact as it turns out you have a parity check matrix over the finite fields, with the elements from  $F_p$  to the  $m$  and using that we define both binary code or ((Refer Time: 08:34)) code. So, that was the idea so we design, define, define this BCH codes and resolament codes, which basically had a pretty much same parity check matrix with elements from  $F$  to the  $m$ .

So, usually we stick to the  $p$  equals 2, right that is the common alphabet here. So, how did that parity check matrix look? If you want a design distance  $d$ , let say the block length  $n$  and  $k$  will come to the dimension later, it is kind of find defined differently. So, if what, if have block length  $n$  and design distance  $d$  what do you need to start of? Parity check matrix  $H$ , yes of course, but how do I construct the parity check matrix?

Student: We need some alpha...

You need some alpha, right. So, what is that alpha? You have to tell me what that alpha is? We need, what? Alpha to have some property.

Student: Order should be at least  $n$ ...

Order should be at least  $n$ , you need some alpha belonging to some  $GF(2^m)$ , whose order is at least  $n$ . Why do I need that? Otherwise in my construction I will have minimum distance 2 so I do not want that, right. So, I do not want that, I do not want that to happen in my construction. So, in the construction you require that alpha belongs to let say,  $GF(2^m)$  is such that order of the alpha is at least  $n$ . So, usually it will be equal to  $n$ , you can take it as equal to  $n$  if you like, that is one choice, that is certain kind of choice. Now, also take some other element it does not matter.

Suppose they have an element like that, then I can construct a parity check matrix, which looks like this. So, the first row is 1, alpha, alpha square, so on till alpha to the power  $n$  minus 1. What is the second row? Alpha squared, alpha squared whole squared, so on till alpha squared ratio to the power  $n$  minus 1. The third row is alpha power 3, alpha power 3 squared, alpha power 3 ratio to the power  $n$  minus 1. The last row is  $d$  minus 1, you have to go all the way to  $d$  minus 1. So, this is the magic parity check matrix it somehow solves all the problems at least the way, which we are looking at this if you want  $n \times k \times d$  codes, which are good respect to the bound, reasonable good with respect to the bound

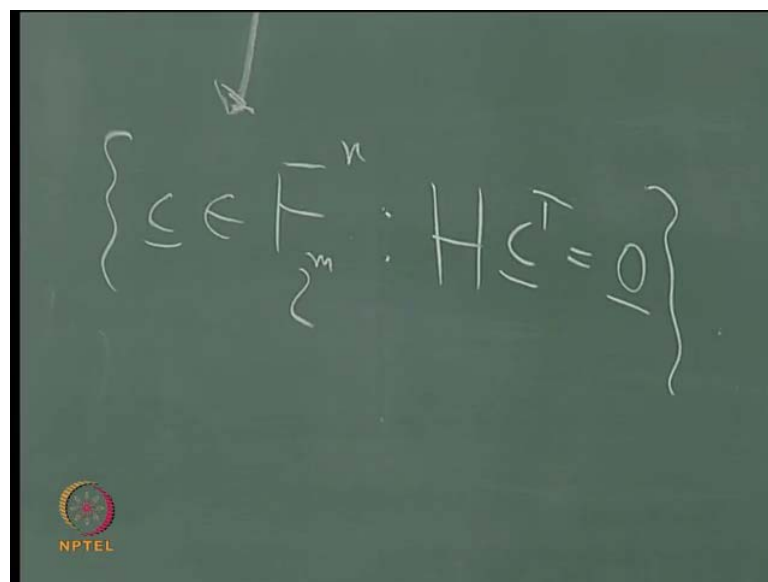
may be. And then they that, that, then you should be able to construct the parity check matrix for a given error correcting capability.

So, we did not quite see the proof of the error correcting capability yet, but any way. So, let us begin with this constructions, this constructions solves pretty much all the problems that we are looking. So, the first thing we did was, try and understand  $k$ ,  $k$  so here is a parity check matrix. So, first of all the definition of bch and resolvment code, sorry for back tracking little bit. So, this is a parity check matrix, elements are from  $F_2$  to the  $m$ , when u design, the, define the bch code what do you do?

Student: ((Refer Time: 12:19))

All code words have to be binary as well. So, you need two conditions for the bch code, bch code basically becomes set of all  $c$  such that,  $c$  transpose 0 and then what? So, basically  $c$  has to be binary. So, this is an important thing, another way to describe the same thing what was said just now, set of all  $c$  such  $c$  in a larger field, then you have to intersect it with the binary. So, these are both ways of defending this.

(Refer Slide Time: 12:58)


$$\left\{ c \in F_2^m : Hc^T = 0 \right\}$$

The resolvment code on the other hand, has no such restrictions. So, here is simply, simply set of all  $c$  in such that. So, according to a definition  $h$  is a perfectly valid parity check matrix for the resolvment code. Even for the bch codes, it is a very valid parity check matrix, but the only problem is, it is not binary code. For a binary code, you expect

a binary parity check matrix, here you have a non-binary parity check matrix. But in spirit as far definitions is concerned, it is exactly the same you can definitely give this definition, there is no problem. We also saw some very simple examples, for very small  $n$  or  $g = 4$  etc. how this definitions play out? We got some good feel why one code is contained in the other, which one is contained in the other one?

Student: ((Refer Time: 13:55))

Yes obviously, intersecting with a smaller set that we will be small. So, BCH is contained in the convolutional code, alright. So, this is a lot of recap I know, but think it is good we did it, if you have any questions now it is a good time to ask about any of these things particularly, assuming you seen the tutorial problems little bit, may be you have thought of some questions, something that disturbing you.

Student: Sir, how do we exactly transmit this, if the alphabet is not from binary?

So, I will come to that there are several ways of doing it but I will repeat the question, the question was BCH code is fine, alphabet is binary you are, you are transmitting it. What about non-binary codes? Suppose, the alphabet here is  $F_2^m$ , how do you transmit the  $F_2^m$ ? How do you do it?

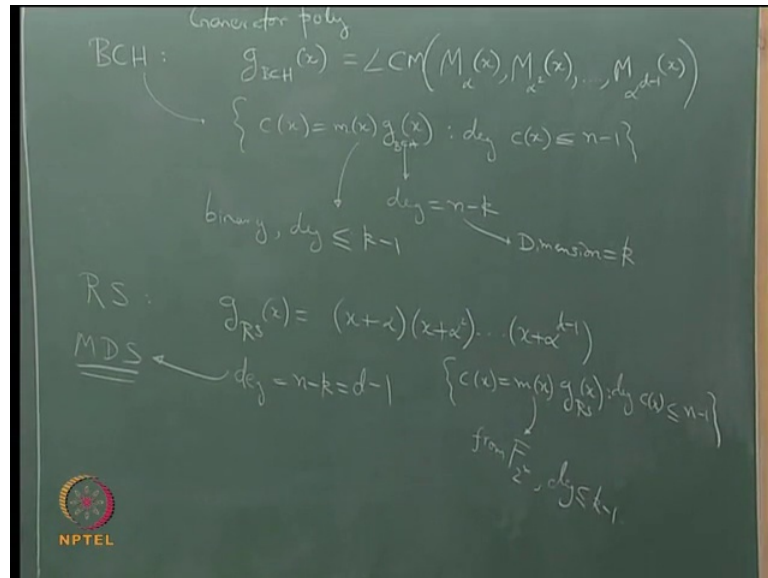
Student: Vector notation.

Yeah, so there is several ways of doing it, one way is to think of  $F_2^m$  basically in vector notation, which is, which is basically embeds, right. Once you have embeds you can send embeds it is no problem, another way to think about it would be, say for instances instead of so in digital communication you have this constellation, right. So, you have BPSK, which is basically just binary constellation, you also have larger constellation, if  $m$  is a 8 you could do like 256 QAM. Then map directly each element of field to a constellation point, which you will believe me if I tell you that, you can send constellation points, right. So, it is something like that.

So, as a rule you, do not have to worry about of sending a finite alphabet whatever it is, right. Just map it was suitable constellation, you can send any way, but remember that point in mind, there is a catch there, you will see when you look at the equivalent binary

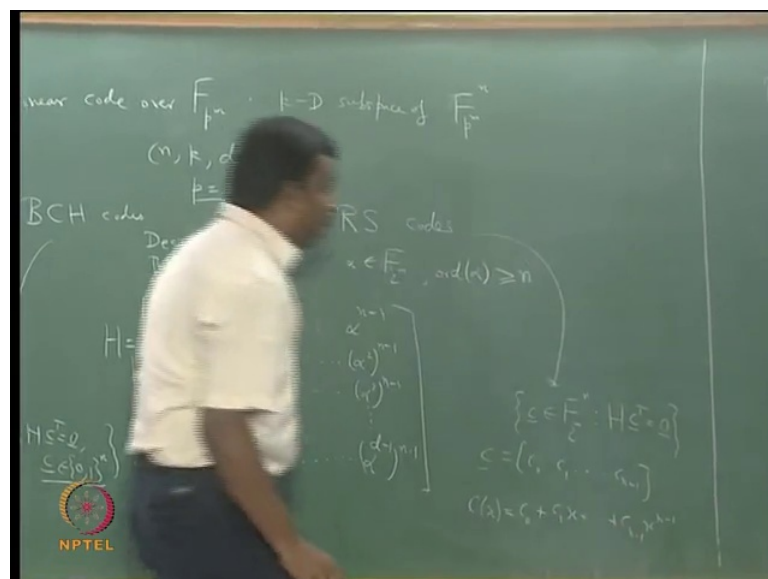
code, the picture will be little bit different. But anyway we will come to that later, for now this is important.

(Refer Slide Time: 15:54)



The next thing we saw was  $k$  and for the  $k$  there are two different results. One for BCH and one for RS, for BCH we defined  $g_{\text{BCH}}$  of  $x$ , which were called as the generator polynomial. So, by the way I mean to understand these code words better, we have to think of each code word as a code word polynomial. So, what, how did we do that change?

(Refer Slide Time: 16:26)



So, basically when you think of  $c$  as a code vector, you are saying it is  $c_0, c_1$  to  $c_{n-1}$ . I can equivalently think it is a polynomial  $c(x)$ , which is  $c_0 + c_1 x + \dots + c_{n-1} x^{n-1}$ . And then in terms of the, in the polynomial world this parity check criteria are very easy to interpret. Basically all the, all the criteria saying is that any code word polynomial has  $d-1$  roots, what are the  $d-1$  roots?

Alpha, Alpha square, alpha power 3 and so on till alpha point elements and then you go to a minimum polynomials etcetera. and then you work it out very carefully. You would get this description for the BCH code, you have a generator polynomial  $g(x)$ , which is the lcm of the minimum polynomials of alpha, alpha square so on till. Did I use this notation or did I use some other notation? Did I use this notation, that is good.

So, it is the least common multiple of all this minimum polynomial. So, what so great about  $g(x)$ ? So, it turns out the BCH code is the set of all  $c(x)$ , which can be written as  $m(x)g(x)$  such that, the degree of  $c(x)$  is less than or equal to  $n-1$ . So, that also we showed kind of roughly we showed why that has to be true, it is easy to see also, for the BCH case you have the added constraints that, the  $c(x)$  has to be binary.

And when alpha is a root of a binary polynomial, it carries along with it all its conjugates also. So, it is very similar to the real and complex scenario. So, the entire minimum polynomial has to divide  $c(x)$ , which is true because  $c(x)$  is binary. So, that was ((Refer Time: 18:27)). So, clearly now it is very easy to count the size of the code,  $c(x)$  is all possible  $m(x)$  into  $g(x)$ , such the degree is lesser than equal to  $n-k$ ,  $n-1$ .

So, all you have to do is, you have to look at the degree of this  $g(x)$ . Suppose the degree is  $n-k$ , you can say  $n-k$  for some  $k$ , then you can show that  $m(x)$  has to have degree less than or equal to what?  $k-1$ , right. Multiplying these two guys, this has degree  $n-k$ , if I want the product to have degree less than or equal to  $n-1$ , I should have form of  $x$  degree less than or equal to  $k-1$ . So, that is the only constraint, it can take any  $m(x)$ , right, binary in the case of BCH. So, this is going to be binary less than or equal to  $k-1$ , how many such  $m(x)$ 's are there?

Student:  $2^{k-1}$ .



$2^k$ , that made very good sense finally because that is exactly our number of code words in the BCH code,  $2^k$  to the power of  $k$  and so  $k$  is our dimension. So, this is, this is a nice notation to use,  $n - k$  for the degree of the generator polynomial. So, it is a little bit different, when you say generator matrix, it is always a  $k$  by  $n$  matrixes, but in the polynomial world the generator polynomial has actually degree  $n - k$ . So, it is reverse usually the parity check matrix is what will have an  $m - k$  by, but here the generator the polynomial itself has degree  $n - k$ . So, you should remember this definition. So, that was for BCH code and that is the degree.

So, dimension equals  $k$  so from here dimension is equal to  $k$  for resolvments codes also you have a generator polynomial, but in this case it is much, much simpler. The reason is simple is when I say polynomial  $c$  of  $x$ , with coefficient  $F_{2^m}$ , has a route from  $F_{2^m}$  then everything is very consistence, only thing that has to happen is  $x$  plus or  $x$  minus that route has to divide your polynomial. So, we are in characteristic 2 so minus and plus are exactly the same so you do not want to worry about that also. So,  $g$  RS of  $x$  becomes a very, very simple polynomial, which is simply the product of, plus  $\alpha$ , plus  $\alpha$  power  $d - 1$ .

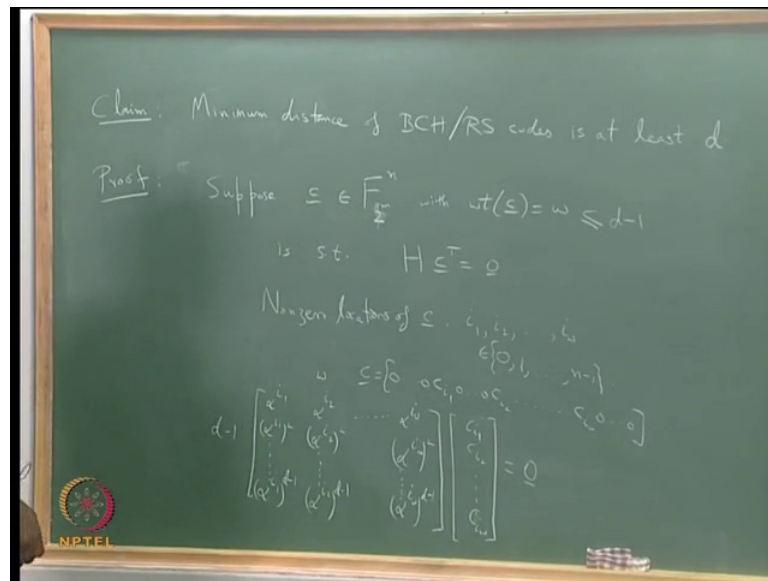
So, I want you to compute it is a very easy computation, the degree of  $g$  RS of  $x$ . So, why I am interested in computing degree of  $g$  RS of  $x$ , I am sorry it should be BCH here, Why am I interested in degree because that degree is  $n - k$ , right, from my resolvment code if you want to find dimension, all I have to do is find the degree of the generator polynomial and that degree would be  $n - k$ . So, the degree here equals  $n - k$  equals what?  $d - 1$ . So, like I said this matrix solves many of the problems that we wanted, one of the problems was  $n - k$  equals  $d - 1$ .

So, that Cleary shows this resolvment codes are mbs. Notice the certain difference here in this case, the set of all code words is again a  $c$  of  $x$  equals  $m$  of  $x$   $g$  RS of  $x$ , again you have degree of  $c$  of  $x$  is less than or equal to  $n - 1$ , but you do not have the binary restriction. So, what happens for  $m$  of  $x$ , it is from  $F_{2^m}$  and degree less than or equal to  $k - 1$ . So, that makes big, that makes the big difference between these two cases. So, I think this probably last thing we did in the previous lecture. So, it is good to go back and revisit this, this is something new we have not seen it before it is important. Nobody is depressed and happy that, we have found maximum distance separable codes,

no, you are happy, you are worried about what exam question might come from, that is a problem with learning it in a class room.

So, of course, I mean to claim really that this is mds, I have to show you that the minimum distance is  $d$ , right. Of course, we do not show minimum distance is  $d$  it is not really mds, but already motivated I have said design distance, I should be able to show minimum distance is  $d$ , which is what will do next.

(Refer Slide Time: 24:50)



So, the proof is not too bad, it is spread straight forward in various ways, it uses very elementary methods. So, we will, will, will say that. So, the claim here is minimum distance of BCH, RS codes is at least  $d$ . So, I only have to show at least  $d$  to claim the mds, why should I have to show only, it is greater than or equal to  $d$ . Yeah, the bound is anyway true,  $d$  cannot never be greater than  $n$  minus  $k$  plus 1.

So, once I show it is,  $d$  is greater than or equal to that already have  $d$  is less than or equal to from the single ten bound. So, I can see both and say  $d$  equals  $n$  minus  $k$ , no problem that is what very easily, this is what I want to show, to show that what I will do is, let us say we will do it proof by contradiction, this proof is by contradiction. What is contradiction? You have to assume it is not true and come up with some, something which is evidently not true.

So, what do we assume is not true I am going to say suppose, a vector  $c$  which is in  $F^p$  with weight of  $c$  equals  $w$ , which is less than, less than  $d$ . So, less than or equal to  $d - 1$  is such that, such that what?  $H$  times  $c$  transpose is  $0$ . So, this is my contradictory statement saying, suppose you have a code word  $c$  which has weight less than are equal to  $d - 1$ , yes of course, this is the same as the minimum distance right, minimum distance and weight same for linear codes. Why should I, should I also worry about  $f^2$  are not, I am sorry  $f^2$  power  $m$ ? Should I, Is it enough if I show  $f^2$  power  $m$  or I should worry about  $f^2$  also?

Student: ((Refer Time: 27:14))

Yeah, see remember  $f^2$  is contained in  $f^2$  power  $m$ . So, if I show this then the binary also comes for free, for proofs incredibly similar so do not want to worry too much about it. So, this is the idea so so what we start by doing is, let us say weight of  $c$  is  $w$ . Let us say the non-zero locations of  $c$  or let say,  $i_1, i_2$  so on till  $i_w$ . So, these guys are between  $0$  and  $n - 1$  so these are the non-zero locations of  $c$ , what do I mean by non-zero locations of  $c$ ?

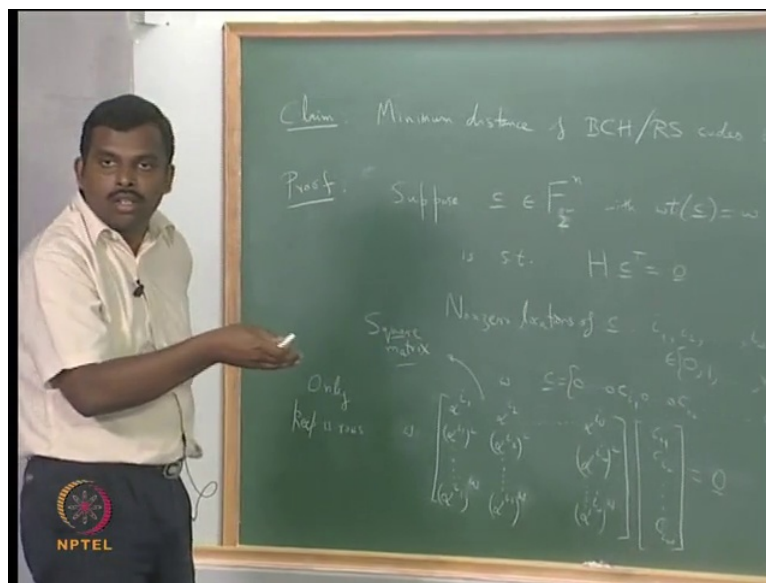
So, in these positions in the vector  $c$  index from  $0$  to  $n - 1$ ,  $c$  can be non-zero. In other positions what should it be? Should be  $0$ , that is also assumed. So, basically  $c$  will look like some, you can think this as some ordered sequence also, that you can image that  $c$  will look something like  $0$  for a while and then it will be something at  $i_1$  and then  $0$  again and  $c_{i_2}$  so on till  $c_{i_w} = 0$ . So, this is the picture that you can keep in your mind,  $0$  everywhere except, but the  $i_1, i_2$  to and  $i_w$  position and the indexing is from  $0$  to  $n - 1$ , that just for convenience, you can also keep it for  $1$  to  $n$ , but resisted small shift.

So, what does  $Hc$  transpose equal  $0$  imply? Where ever there is a  $0$ , that column does not participate in the equation at all. So, we multiply it out and keep only the non-zero terms, you will see, you will get a smaller equation than this, several of columns will go away. So, what exactly the equation that I get here. So, you will have so  $\alpha^{i_1}$ ,  $\alpha^{i_1^2}$  so on till to  $\alpha^{i_1^{d-1}}$ , right. This will be the first column, I will write down the final thing later. Second column  $\alpha^{i_2}$ , is it right or am making mistake? It is fine.  $\alpha^{i_2^2}$   $d - 1$ , I think I have done an interchange between may be the way you used to.

So, there it might have been  $\alpha^{i-1} \alpha^2$  per  $\alpha^d$ , but  $d$  minus ratio to  $i-1$ , I am just doing the, using the competitive of multiplications, it is not a big deal.  $\alpha^w$ ,  $\alpha^w$  squared so on down to  $\alpha^w$  raised to the power  $d-1$ . What is the dimension of this matrix now?  $d-1$  cross  $w$ ,  $w$ . So, remember that  $d-1$  cross  $w$  times what, times  $c_1, c_2, \dots, c_w$  equals 0 when you say 0, it is all 0, column matrix is this clear? It is okay no? So, we can go back and check that, I mean what, what I might have written here is  $\alpha^{i-1}, \alpha^2$  so on, but I can bring the  $\alpha^{i-1}$  inside and take the square outside that is the same thing.

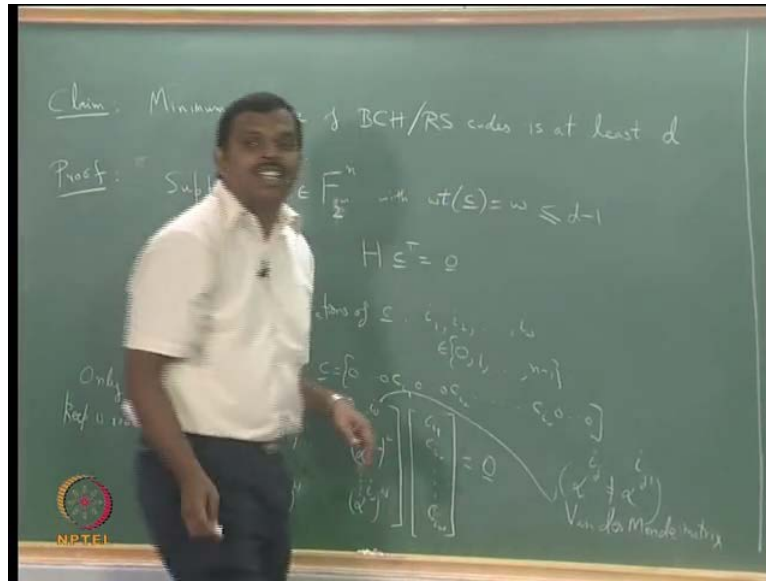
So, so  $w$  is less than or equal to  $d-1$  so I can choose to keep  $w$  rows and drop the other rows. So, you might ask why I want to do it, was then I would get a square matrix, square matrices you have determinant, which is a much much more well studied and understood idea, which is what we are going to use next. Since  $w$  is less than or equal to  $d-1$ , I can without any problem only keep  $w$  rows here and drop the remaining.

(Refer Slide Time: 32:00)



So, if you keep, only keep  $w$  rows what will change here is, this will become  $w$  rows, what will change here instead of  $d-1$ , I will have, I will have  $w$  is that okay? that is fine, when I, when I did this, this matrix become square, all right. Remember my  $\alpha$  is such that, order of  $\alpha$  is at least  $n$ . So, clearly there is no danger of any of these things repeating or any of these things is becoming equal to the other.

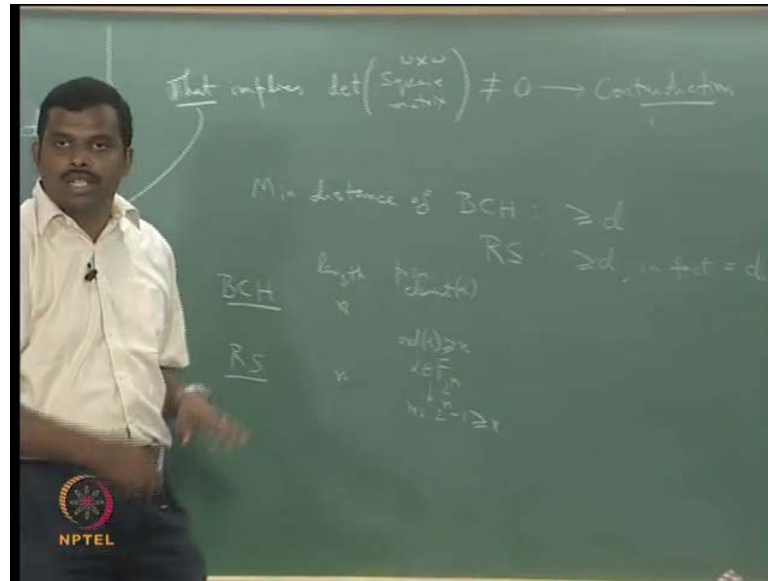
(Refer Slide Time: 33:04)



So, what will happen the square matrix is alpha power  $i$   $i$   $j$  will not be equal to alpha power  $i$   $j$ . So, in this case all these experiments are, all these entries are different. So, this square matrixes has a name you might have seen it before when you solve any matrix determinant problem, it is called the vandamonde matrix. So, its determinant is a well-studied problem from ages ago. So, this matrix, when this condition is true it is called the vandamonde matrix and one can show its determinant goes to 0, only when, let me put it in another way. As long as this condition is satisfied, its determinant is non-zero.

So, you can prove it using so many ways, I mean you think of it as, you can subtract the two columns and then you see there is a common factor that pulling out. In fact the determinant of this has a very simple expression, simply the products of the form alpha power  $i$  1 minus alpha power  $i$  2 or alpha power  $i$  1 minus alpha power minus  $i$  3. So, it is a product of all those things. So, basically the final statement that you can make now is, it is a vandamonde matrix and the entries are not same. So, determinant of that square matrixes is non-zero.

(Refer Slide Time: 34:59)



Now that implies, I am sorry, what is this that, by the way? This guy, the fact that none of these things are same and it has that shape, it became a Vandermonde matrix, a Vandermonde matrix with non-identical columns. When you have that determinant of that matrix, square matrix it is not 0, and you have a contradiction. So, if you have a matrix with non-zero determinant, clearly you cannot have a non-zero vector in its null space, that is, that is not got to happen. So, we have a minimum distance results. So, that gives us, it is greater than or equal to  $d$ , for RS again it is greater than or equal to  $d$ , but then what happen because of this. So, what did we show? We also showed that  $d$  has to be.

So, by the single ton bound it cannot be greater than  $n - k + 1$ . So, you have here two different things, which tell you that for the RS case, it has to be in fact equal to. For BCH you cannot, in general say that is equal to  $d$  because you do not have any result on the degree of the, degree of GPC. In this case degree of RS becomes equal to the  $d$  minus 1 so who know from there that nice bound of  $k$ , alright. So, to summarize for BCH codes and for resolution codes, we have block length and so we need a primitive element. Remember, order of alpha should be at least  $n$  alpha is from some  $f$  2 power  $m$ .

So, you have to pick  $m$  in such a way so what is an easy way of picking such an alpha? Or what is one way, it is not so easy, what is one way of picking such an alpha? Is there any readymade method that you can think of? So, my question basically is how do I pick  $m$ , is there a very smart way in which you can quickly pick  $m$ .

Student: ((Refer time: 38:51))

What will I tell you m is 1000, what you will take m as?

Student: 2 power m minus 1 equal to m

Equal to m, okay. So, you take n equal to 2 power m minus 1, what if there is no such m? for instance 1000, clearly there is no such m, what will you do?

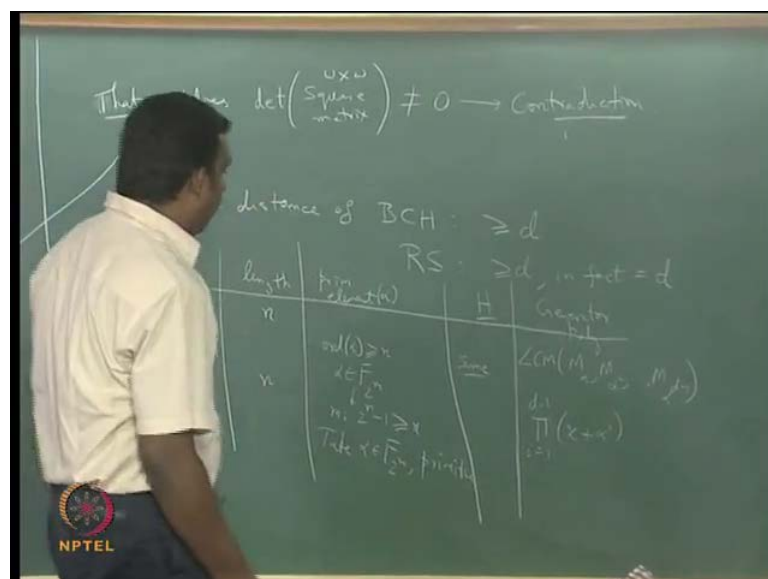
Student: next

What has to be created?

Student: 2 power m minus 1.

2 power m minus 1 has to be greater than n, why should 2 power m minus 1 is greater than n, what is the idea there? So, the idea is you go to m large enough so for 2 power m minus 1 is at least n and then take alpha to be a primitive elements of that. If it take, take it to be a primitive element, what do you know the order is? Order is equal to 2 power m minus 1 and that would be greater than n. So, that is strategy you can use, in case. So, basically idea here is to pick m such that, 2 power m minus 1 is greater than or equal to n. So, that is the first step.

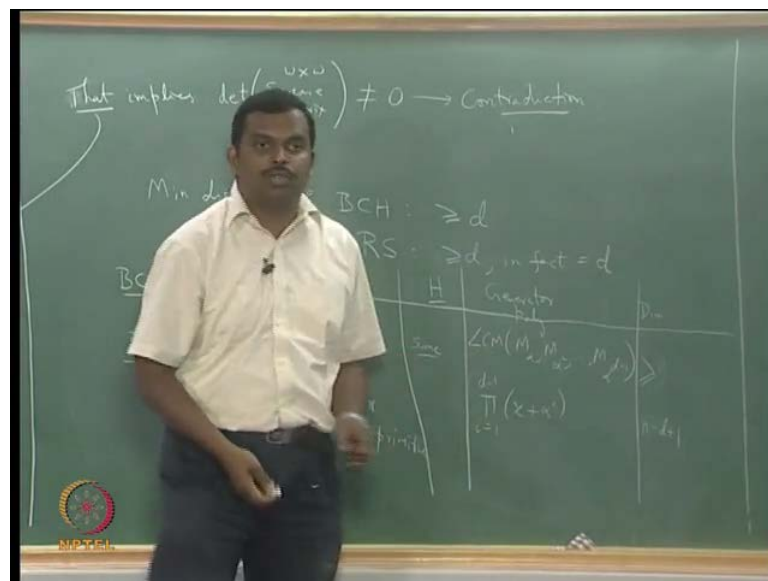
(Refer Slide Time: 39:57)



And then take alpha belonging to  $F_{2^m}$  as primitive, as a primitive element. So, once you take it to a primitive element, what will happen? Order of alpha will be  $2^m - 1$ , which is greater than or equal to  $n$  clearly by choice. So, any  $n$  you can go to a large enough  $n$  and get the alpha satisfied. So, parity check matrix is identical it is so the same for both. But what happens to the generator, may generator polynomial  $k$  for BCH you have the LCM of, the LCM formula for resolvment what do you have, is that okay? Occasional, folded on due to the AC system, I think it is totally off, not getting any air-conditioning at all, are you responsible for that or not.

So, maybe I should draw some lines here to remarket what I am saying, what else should I add and the next thing I should add is for dimension and minimum distance, what about dimension? The dimension for the resolvment code is simply  $n - k$ ,  $n - d + 1$ . So that you can quickly read out because the degree of generator polynomial is simply  $d - 1$  and that is  $n - k$ . So, you do the shift you get  $n - d + 1$ .

(Refer Slide Time: 42:05)



The dimension here is a bit more complicated, but it will be definitely smaller than  $n - d + 1$ , in fact if you pick  $m$  as greater than 1, it will be definitely much smaller. What you can show here is a very simple bound, you can show the dimension greater than or equal to, okay let me say, let me say how do you come up with the bound on the dimension of BCH codes real quick? What can happen in the worst case, in the worst case all these guys can be?



Student: Different

Different, but that cannot really happen, why cannot that, all of them cannot? Because, the  $\alpha$  and  $\alpha^2$  has a same primitive polynomial, same minimum polynomial, I am sorry. So, for the odd numbered guys you might have all different minimum polynomial, in which case what should you do? You will be taking the product, when you do it, when you take the product it will become, the degrees will add. So, I need a bound on the degree, what is the maximum degree of minimum polynomial? If I take an element of  $\mathbb{F}_{2^m}$ , what can be the maximum degree of polynomial?

Student: At max  $m$

At max  $m$ , it cannot be greater than  $m$ , why cannot it be greater than  $m$ ?

Student: It should be maximum only

Yeah, I think I gave these results, maybe I did not prove it very rigorously, you might have forgotten it, I said the degree of the polynomial has to divide  $m$ , you remember, yeah exactly. So, if you take a particular element, you would have count the number of conjugates it has, in no circumstances can it have more than  $m$  conjugates. So, if you, if you keep on squaring it, once you go to  $2^{2^m}$ , you definitely repeat.

So, it cannot have more than  $m$  conjugate so that is an idea. So, based on that you can do a bound, you can say it is greater than or equal to something and think about how you may want to do that. I am not going to write down the exact expression, you can look at the number even guys here and then do that times  $m$  etc. So, it is, it is a possibility definitely it will be smaller than  $n - d + 1$ .

(Refer Slide Time: 44:13)

Generator Poly	Dim	Dist
$\langle M_0, M_1, \dots, M_{d-1} \rangle$ $\prod_{i=1}^{d-1} (x + \alpha^i)$	$\geq ?$	$\geq d$
	$n - d + 1$	$d$

So, then the minimum distance, in this case is equal to  $d$ , here it is, at least. So, I should point out that the exact dimension and minimum distance of BCH codes are tough to compute in most cases. Well, dimension may not be that difficult to compute, you have to do just some simple operation, minimum distance is a much more difficult problem. So, it continues to be a research problem, may be not of so much interest as it was before because it is an old code people do not use these things so much anymore. Many cases, many interesting practical cases it is been solved, only for some extreme case people still keep working at it. So, that table kind of captures the pictures roughly.

So, if it is clear are the any, anything that disturbing you, it is simple enough for this point, right, there is nothing, I guess the proof here is the only thing which is little bit more involved and the proof just comes from this Vandermonde matrix property. So, essentially this, this kinds of matrices have a non-zero determinant as long as the columns are not identical.

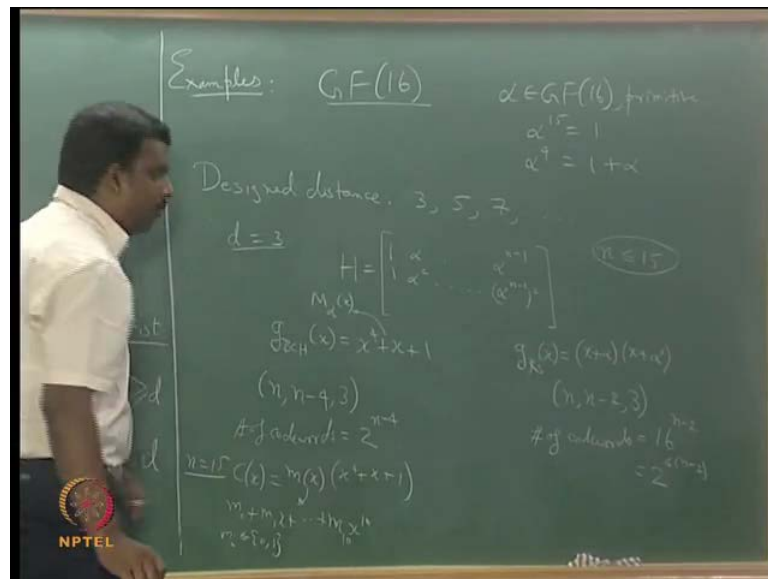
So, that is the basic idea and using that you build ((Refer time: 45:47)), see this is the most crucial thing in the entire construction of BCH code and insolvents codes, right? I might have started at one point, I would have gone on here and there, but ultimately this is the most crucial aspect if you think about it very carefully. All the other things you can manage, it is interesting that you have a generator polynomial, it is interesting that code

words have a very simple descriptions etc. etc. But this is the most crucial property, which give you minimum distance.

Student: ((Refer time: 46:14))

I do not know the first paper, I mean that question was, how was the first construction? How did people think about it? I do not really know, but this is the heart of it. I know for instance Reed Solomon, Reed and Solomon when they describe their constructions, they did not use this language. They used completely different language which is also, which is also very interesting, I think there is a, there is a tutorial problem in your tutorials on that, you can take a look at it, it is an interesting way of doing. So, let us see some examples, we are going to see a whole bunch of examples and then that will drive home the point.

(Refer Slide Time: 47:18)



So, will begin with some, the simplest and most interesting example, will begin with GF 16. So, we will do constructions with GF 16. So, when I say we will do constructions with GF 16, it imposes some restrictions on the block length. What is the maximum possible block length that I can use? If I say I am going to limit myself to GF 16, 15 right, I cannot go beyond 15, the order of primitive element is 15, I cannot go beyond 15, keep that in mind. So, I will start with alpha and GF 16 being primitive, remember alpha power 15 is 1, then alpha power 4 is 1 plus alpha. So, had it been primitive polynomial I chose for the constructions for GF 16.

So, I get that, get the ((Refer time: 48:21)). So, I have been talking about design distance  $d$ , so it is much more common to use error correcting capability, design error correcting capability. So, what is designed capability so if you want a designed error correcting capability of 1, what should your design distance be?

Student: 3

3, if you want designed error correcting capability of 2, what will be your? 5. So, basically it means, you only have to look a designed distances, which are 3, 5, 7 so on. You do have to look at 4, 4 does not give you anything new, you cannot correct anything interesting, 4 gives you something interesting may be, but does not give you anything which is useful for error correction. So, usually the design distance are 3, 5, 7 so on alright. So, let us start with 3, if you start with 3. So, 3 is not really that interesting because we already know how to do 3, right, you know how to do it, it is not so difficult, but still we will, we will start with 3, parity check matrix in a very simple way, I do not want to write this whole parity check matrix over and over again.

So I will, I will simply say, I will write some very short hand notation, hopefully it is very clear to you. I am not going to write all the rows, all generality may be I will write some 1 or 2 rows, hopefully you can interpolate from any way. So, let us say  $d$  equals 3 for parity check matrix is going to look like, what? We will have powers of  $\alpha$  so on then  $\alpha$  squared that is it, right. Last guy is going to be  $\alpha$  power  $n$  minus 1, is that okay? So, you have just two rows and  $n$  columns, remember  $n$  is going to be always less than or equal to 15 for me because I have limited myself to GF 16, you can go all the way to 15, if you like, whatever  $n$  you like you can stop that.

So, this is  $d$  equals 3 so for  $d$  equals 3, what is the GBCH? Your GRS is really simple  $x$  plus  $\alpha$  times  $x$  plus  $\alpha$  squared. So, GBCH will simply be the LCM of two minimum polynomials, what are the two minimum polynomials? minimum polynomials of  $\alpha$  and minimum polynomials of  $\alpha$  square, which is also the same as the minimum polynomial of  $\alpha$ . So, simply you will get the minimum polynomial of  $\alpha$  itself and that happens to be?

Student:  $x$  power 4 plus  $x$  plus 1

So, you get  $x^4 + x + 1$ . In this is the minimum polynomial of  $\alpha$ . So, the RS code we now know is  $(n, k)$ , what is  $k$ , what is the dimension for RS code? is it 2 or  $n - 2$ ? Little bit confusing, right. So, remember degree of the generator polynomial is the best way to start when you want to find dimension, first find the degree of the generator polynomial, what is the degree of generator polynomial?

Student: 2

2 and that 2 equals what?

Student:  $n - k$

$n - k$ . So,  $k$  is  $n - 2$  so  $n - 2$  and we know, the minimum distance in this case is equal to 3. So, these are the three parameters for the resolution code. So, now I am going to ask a very difficult question, how many code words are there in this resolution code?

Student:  $16^{n-2}$

$16^{n-2}$ , do you agree, is that okay? How many number of, number of code words is what? How many of you think that is most bizarre thing you have ever heard? Does it make sense?  $16^{n-2}$ , how do you list out all the code words of the RS code? You have to take this  $m(x)$ ,  $m(x)$  is what, that is like the message polynomial whatever. So, it is a  $m(x)$ , you have to take  $m(x)$  of degree less than or equal to  $k - 1$  or degree less than or equal to  $n - 3$  and multiplied with this  $g(x)$ . How many possible  $m(x)$  do I have?  $16^{n-2}$ . So,  $16^{n-2}$ .

So, remember this is, these are codes over GF 16, that is another way of thinking about it, and these are codes over GF 16. So, moment I say the dimension is  $k$ , how many code words do you have?  $16^k$ . Dimension is  $n - 2$  you have  $16^{n-2}$  code words. Suppose I say  $n$  is 15, how big is that number, how many digits will that number have? It is, it is a big number that is why I want you to convince your word, take  $\log_{10}$  of that number. So, when is 15, 13 times  $\log_{10}$  of sixteen,  $\log_{10}$  of 16 is fairly large, right. So, you have that many digits. So, is that okay, everybody okay? Alright. So, such a large number, so that is what want to impress upon you, it is really really large number.

So, so far we have been talking about a 7 comma 4 code, we had just 16 code words, I mean it is a small code. Now talking about a huge code and you have a very compact description for it, what is the compact description for the huge code? You simply multiply any polynomial with  $x$  plus  $\alpha$  and  $x$  plus  $\alpha$  square. So, the number of bits so number of message vectors is  $16^{n-2}$ .

So, if you want to count the number of message bits, you have to write it as  $2^{n-2}$ , when  $n$  is 15, you are looking at 52 bits. So, you can take 52 bits, convert them into 13 symbols over GF 16 and then make this polynomial  $m$  of  $x$  and then multiply  $m$  of  $x$  with  $g$  of  $x$ . Remember the multiplication have to be done in GF 16, you cannot just change that all of the sudden.

So, you will get one big polynomial  $c$  of  $x$ , that will be a code word polynomial, right and that will have 15 symbols from GF 16, if you want you can convert that back into bits, you will get how many bits? 60 bits, is that reasonable, think about what it is, alright. So, that is the picture in, picture in the Reed Solomon world, what happens in the BCH world? It is again little bit similar, but slightly different also, if you want to write down the three parameters, block length is  $n$  fine and less than or equal to 15, what about the dimension?  $n-4$  and what about minimum distance? The only thing I can say this point is, greater than or equal to 3.

Can somebody come up with may be, a way of showing the minimum distance will be equal to 3, what is the way of showing its equal to 3?

Student: There will be  $\alpha^2$

What is the only way of showing minimum distance equal to 3? You have to find the code word of weight 3, which is binary also, right. There are two problems that it has to be binary and then code word of weight 3, I am saying there is a code word of weight 3 staring at you in the face, right there on the board, which is that code word? Can you identify on the board a code word of weight 3? Remember always, it is good to think of code word polynomials, instead of code word vectors, what are all the code word polynomials? All the polynomials which are products of GBCH of  $x$ , for a instance if I take  $m$  of  $x$  to be simply equal to 1 then what is definitely a code word?  $x^4 + x + 1$  is the code word polynomial. What is, what is the weight of that code word

polynomial? 3. So, there is answer staring you on the face right there, that is the code word of weight 2, is that okay?

So, little bit of switch to go from vectors to polynomial suddenly, think about what I said once again very carefully. How do you find all the code words of the BCH code? We have this polynomial notation, right. So, we take  $m$  of  $x$  and multiply with  $g$  of  $x$ , what is  $m$  of  $x$ ? It can have only binary coefficients and it can have degree less than or equal to  $k$  minus 1. So, you remember right, I wrote that down ok, but remember binary 0 and 1, one, one such valid  $m$  of  $x$  is  $m$  of  $x$  is equal to 1, basically 1 with 0's everywhere else. You state that simply get GBCH has a code word, that has weight 3.

So, I can see easily the minimum distance will be equal to 3. Now how many code words will this have?  $2^{n-4}$ . So, this seems a little bit more manageable, you are not too scared of it, even if  $n$  is 15,  $2^{11}$  is 2048. It is not so big, you are not too scared, only 11 bits, you can take 11 bits and then map it to 15 code word bits directly by the multiplication. I will draw the picture of that also so on enough, but I want you to see that, any questions? So, so if you have come up with code words of these codes. So, if you want to generate let us say, 10 different codes words of BCH code, how will you do it?

Student: Come up with 10 different codes.

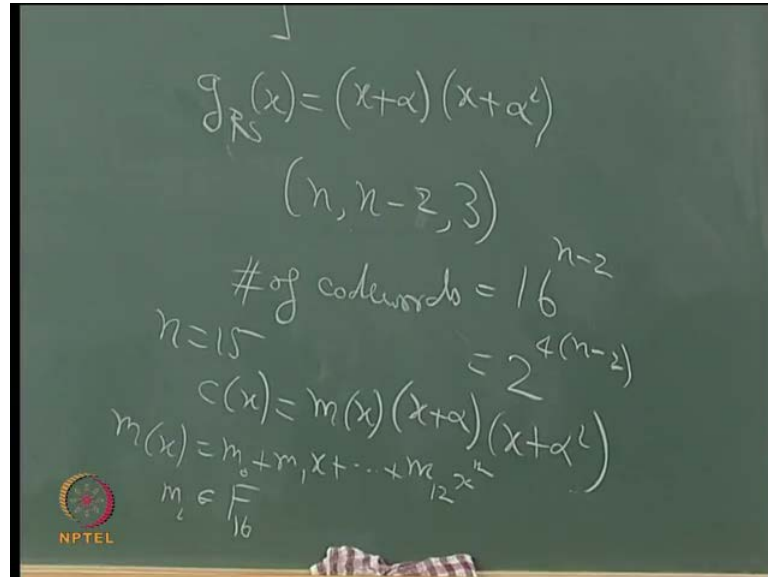
Yeah, you will find 10 different message polynomials so easily come up. So, remember a code word polynomial  $c$  of  $x$  is  $m$  of  $x$  times  $x^4$  plus  $x$  plus 1, it is very easy multiplication. Remember what is this  $m$  of  $x$ ?  $m_0$  plus  $m_1 x$  plus so on till  $m_{10} x^{10}$  to the power 10, what are these  $m_i$ 's?  $m_i$ 's is either 0 or 1. So, as simple has that you have 11 different message bits, you fill them up with an arbitrary 11 bit vector anything you like. You will get a message polynomial, message polynomial you multiply with your generator polynomial, you will get a polynomial of degree less than or equal to  $n$  minus 1, whatever your  $n$  was, this case will get 14 of it is less than or equal to 14.

And that will have binary coefficient and those are here valid code words. This is one way to think about it, here what would you do to get a code word polynomial? It is a little bit more slightly different, right? So, this is  $n$  equals 15, I am sorry.

Student: ((Refer time: 01:02:02))

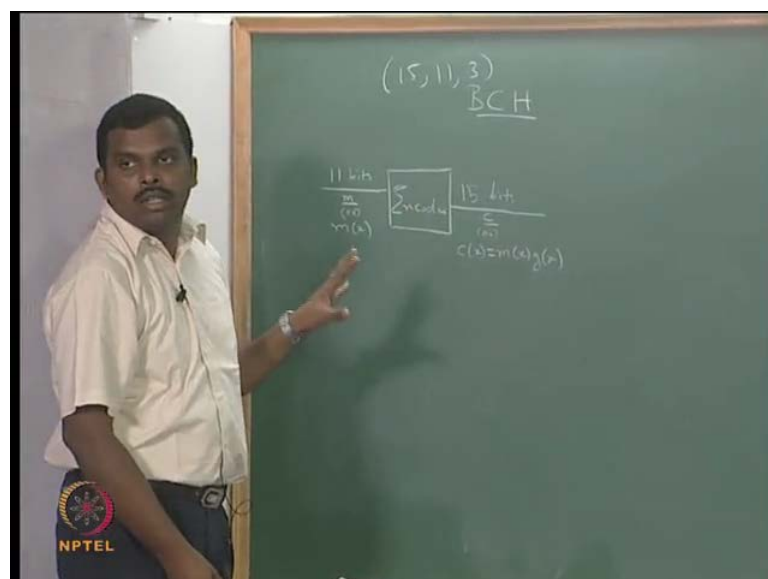
Yeah. So, in the picture will be very similar except that this polynomial will not be  $x^4 + x + 1$ , it will be  $x^2 + \alpha$  into  $x^2 + \alpha^2$ . And the coefficients will go not just till 10, it will go all the way till 13 well, 13 so 12, right.

(Refer Slide Time: 01:02:22)



So, you multiply with  $g$  of  $x$  in case if you want, I can write that down. So, if you pick  $n$  equals to 15 your code word  $c$  of  $x$  is going to be  $m$  of  $x$  times  $x^2 + \alpha$  times  $x^2 + \alpha^2$ . And your  $m$  of  $x$  is  $m_0 + m_1x + \dots + m_{12}x^{12}$ , each of these  $m_i$ 's are now from  $F_{16}$ , alright. So, if I have to draw a picture, I think picture is also important here to see something, quickly draw a picture.

(Refer Slide Time: 1:04:00)



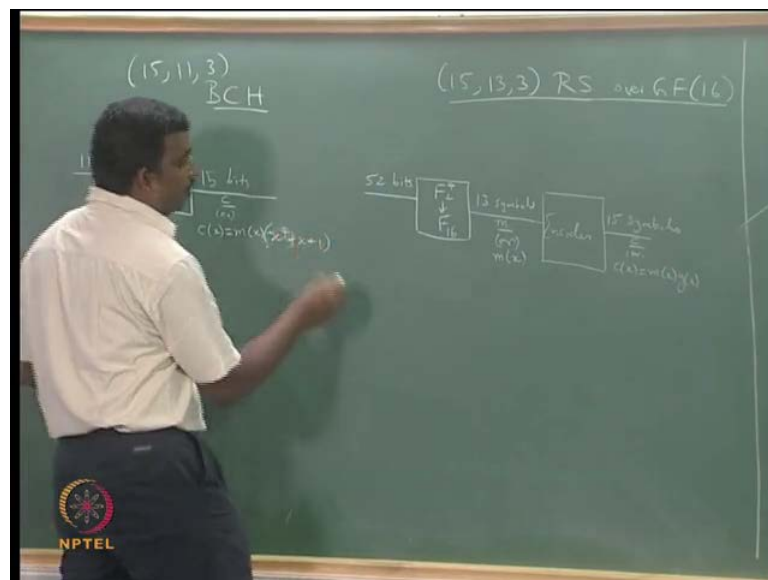


So, this is how a bch code, the 15, 11, 3 BCH code, this is how it would work. So, you would have 11 bits, it is let us call it a message vector  $m$  or message polynomial  $m$  of  $x$ , you are going to do an encoding and you would get how many bits? how many bits?

Student: 15 bits

15 bits, we can call it a code word vector  $c$  or a polynomial  $c$  of  $x$ , this will be let say  $m$  of  $x$  into  $g$  of  $x$ , let say this how you are choosing to do, your encoding. In fact there are other ways to encode, but this is may be the way we pick. So, that is 15 bits, this is how the encoder would operate for the BCH code.

(Refer Slide Time: 01:05:01)

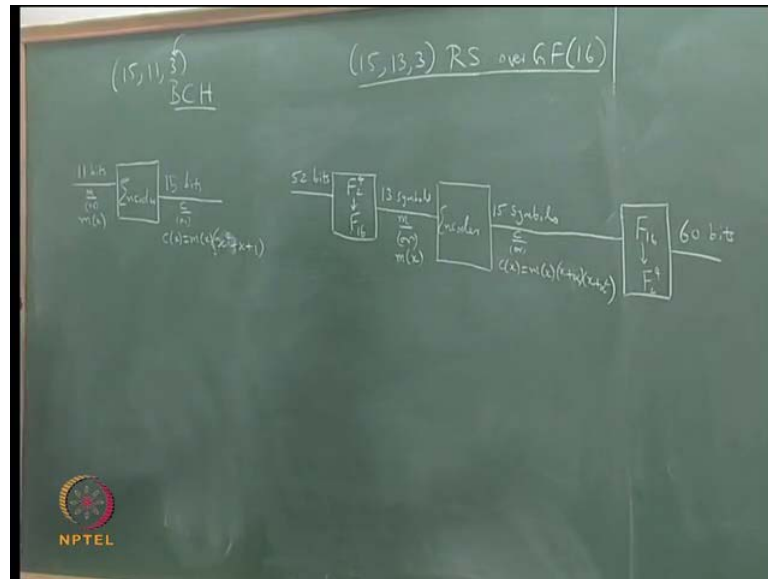


What would happen for the 15 comma 13 comma 3 Reed Solomon code over GF 16, you would first have how many bits? 52 bits, which you can think of as a binary message, what will you do here? You go from  $F_2$  to  $F_{16}$ , what is this notation  $F_2$  to  $F_{16}$ , what does it mean? So, I take four bits at a time, then converted into a symbol over  $F_{16}$  or some alpha power some  $i$ .

So, it is 4 bit vector represent some alpha power something, I convert to that, after that what do I get? I get 13 symbols, I have to call it symbols or elements of  $F_{16}$ . I could call it a message vector  $m$  or message polynomial  $m$  of  $x$  and then what do I do? I use my encoder, once again you do multiple things here. Let us say will do multiplication with you get a code word  $c$  or  $c$  of  $x$  would be  $m$  of  $x$  into  $g$  of  $x$ , remember the  $g$  of  $x$  here is

different. So, I should may be say what the  $g$  of  $x$  here is also, here it is  $x$  power 4 plus into  $x$  plus 1, here it is  $x$  plus  $\alpha x$  plus  $\alpha$  square. It can be implemented, right if you are doing like a chip in FPG or VLSI, you can do it or you can write a computer program to do it, MATLAB program etc. anything you can actually do this, it is not very difficult.

(Refer Slide Time: 01:07:13)



And then what we will do? So, you have to do a  $f 16$  to so I am running out of space, just let me erase some more of this. So, it is good to think of it is being converted back to bits, even if you are doing something else here, you can go from  $f 16$  to  $f 24$ , so that you get 60 bits, this is how an encoder would operate. So, there are several differences between these two pictures, many of them are easily apparent to you. So, here you have a much longer number of bits, you are dealing with a longer block, here you are dealing with a shorter block. There is a saturation about the minimum distance, which I want to particularly emphasis with you.

So, when I say minimum distance is 3 here, what does it mean? If I take any two code words that are going out here, there will always be separated by at least 3, if they are distinct, if they are same of course, they are same, they are distinct. Then they will be separated by at least 3 bits, what will happen here, where does where does it apply? Here it applies, right if I look at any two symbol that are going out here, they will be separated in at least 3 symbols. So, that is different from saying 3 bits, it could be more than 3 bits,

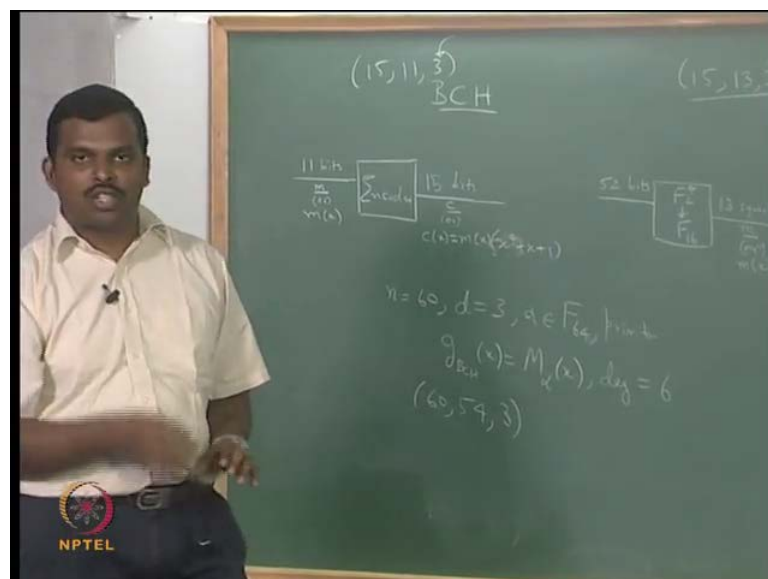
but it will be at least 3 bits. What about the outer picture there? Suppose I say from 52 to 60, I look at two 60 things that are going out here, what will be the separation between those two?

Student: At least 3.

At least 3, all I can guarantee is at least 3 right because 3 symbols can be different in exactly only 1 bit position in the worst case and that would differ in only 1 bit. So, if you see what you have achieved over all, you are going from 11 bits to 15 bits, that is at least different of 3, here you are going from 52 bits to 60 bits with once again at least a difference of only 3. So, ultimately when you use a Reed Solomon code which is mds and all that, the final binary code is far from being mds. So, 60 52 has an n minus k of 8, but you still only have a minimum distance of 3.

So, that is a bit of catch, you should be aware of that when use it you do not get binary mds code suddenly. So, gone to a larger field and solve the problem. So, you do not necessarily get a binary mds code, but it is useful in several other ways this, this notion of symbols having a minimum distance is also very useful. It is not that it is useless, but it should be aware that finally, when we do this we do not get the same mds properly, it comes only at the 16 level. So, let us do, let us do a slightly more complicated example. It is only 3 minutes so it is good to do a complicated example. Suppose, I want a BCH code with n equals 60 suppose, I want bch code just to do a comparison here.

(Refer Slide Time: 01:10:58)



Suppose, I want a BCH code it say  $n$  equal 60 and  $d$  equals 3, how should you go about doing it? First step is to find an  $m$ ,  $m$  such that  $2^m - 1$  is greater than 60. What would be that  $m$ ? 6. So, I can start with an  $\alpha$  and  $f$  64 which is primitive and define a parity check matrix etcetera, etcetera. If I do that, just  $d$  equals 3 so my GBCH will be what? It will be LCM of  $m$   $\alpha$  and  $m$   $\alpha$  square, but again  $\alpha$  and  $\alpha$  square have the same minimum polynomial. So, it will be simply  $m$   $\alpha$  of  $x$  and what will be  $m$   $\alpha$  of  $x$ , which will be primitive polynomial used for a design of  $f$  64. So, in particular what will be the degree of this?

Student: ((Refer Time: 01:11:60)) 6.

6, right. Degree equals 6, why I am worried about degree, why did I asked this question about degree? Yeah, I want to find the dimension of this code, if you want to complete this picture, I should know  $k$ , right, what is its  $g$  of  $x$ ? So,  $g$  of  $x$  has degree 6 so what is this code now,  $n$  equal 60  $k$  is 54 and  $d$  at least 3. And this case believe me, it is true, you can go see the tables minimum distance will be equal to 3.

Now I want, want you to tell me why I ask this question, why did I ask this question? I want you to compare the final binary thing you got with the Reed Solomon construction and a direct BCH construction. What would you say is better? The direct BCH is better, why is that we are getting two more bits are supposed to Reed Solomon, but what would be an disadvantage of the direct BCH bit, for two more bits is it worth that what are you paying for two more bits?

Student: All the math will be with 0 64.

All the math will be with 0 64, for just two bits you have to do is all the arithmetic. Well, the encoded seems like there is no arithmetic, but the  $d$  coded believe me, tell me the arithmetic over  $g$  of 64. Now you have to do operations over  $g$  of 64, for just those two more bits of information. So, in Reed Solomon you can work with just GF 16 which is a much smaller, smaller field, much smaller table to keep away. So, we will stop here for this lecture, we will break and then will pick up again for the other one, but do a good picture to have in you. We will see more examples, will see many more examples to write down the point, but this is an important enough point to know, I can say that.