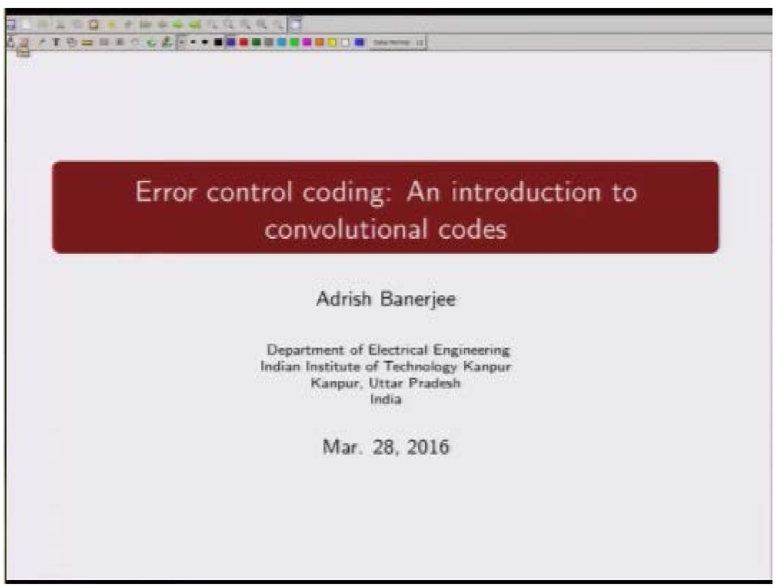
Indian Institute of Technology Kanpur
National Programme on Technology Enhanced Learning (NPTEL)
Course Title
Error Control Coding: An Introduction to Convolutional Codes

Lecture-5B
Problem Solving Session-I

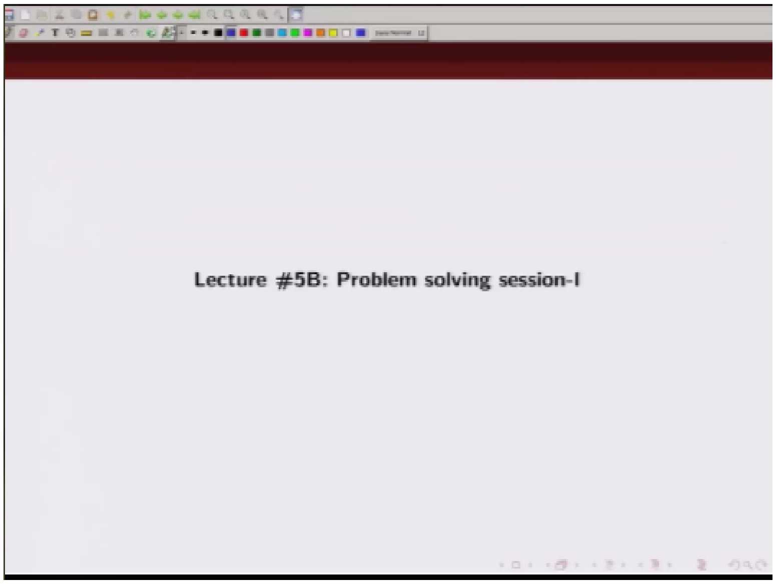
by
Prof. Adrish Banerjee
Dept. Electrical Engineering, IIT Kanpur

Welcome to the course on error control coding, an introduction to convolutional code.

(Refer Slide Time: 00:21)

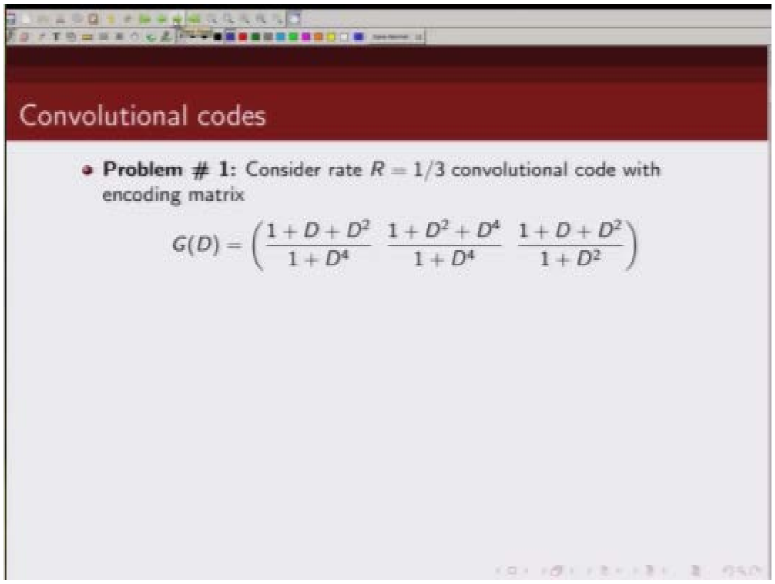


(Refer Slide Time: 00:23)



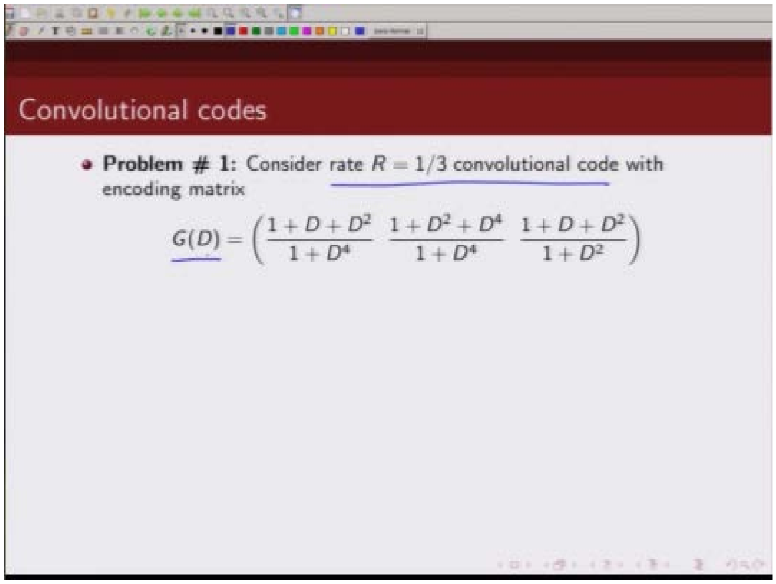
So before we go to concatenated codes let us spend some time solving some problems.

(Refer Slide Time: 00:28)



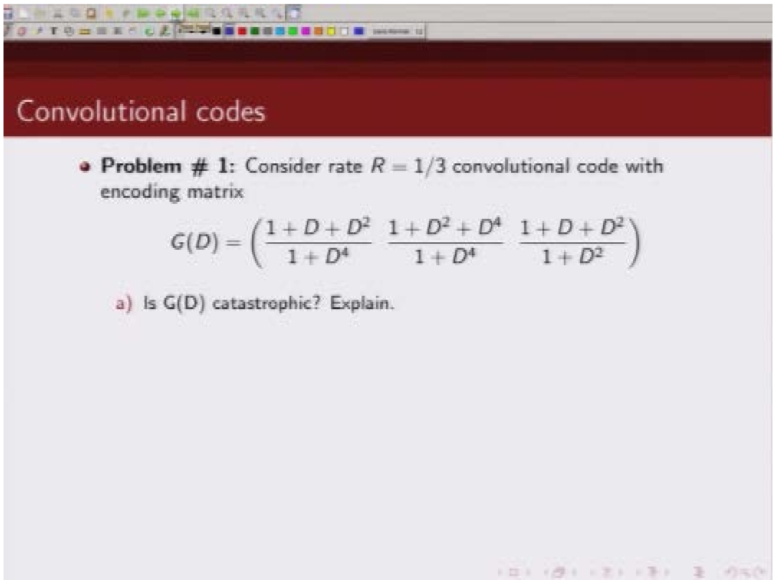
So the first question is.

(Refer Slide Time: 00:32)



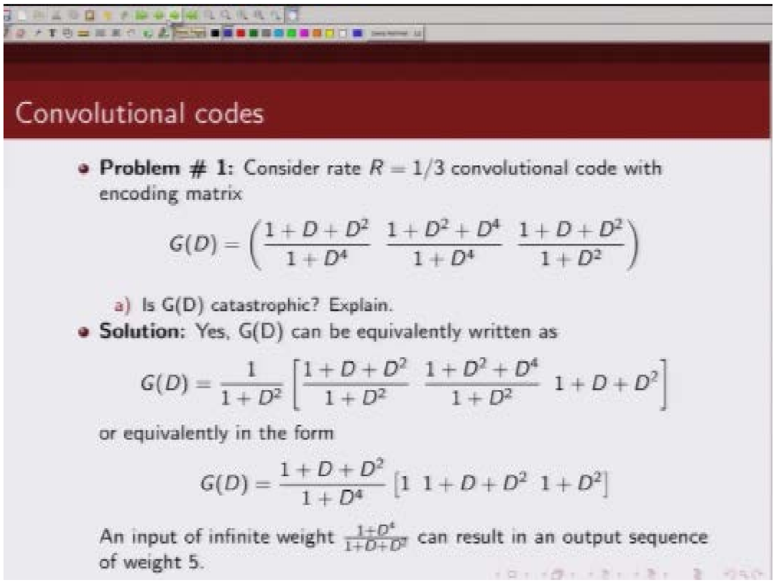
You are given a rate 1/3 convolutional code with generator matrix G(D) which is given by this.

(Refer Slide Time: 00:40)



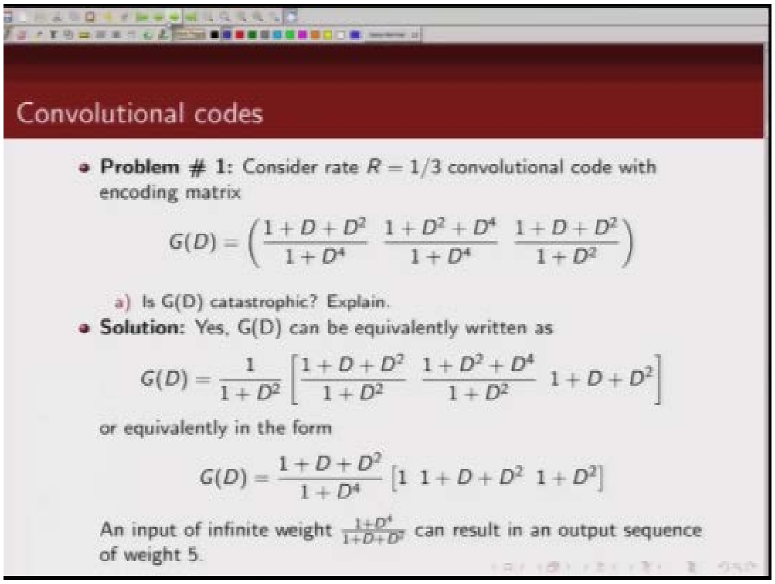
The first question is, is this a catastrophic encoder? Will an encoder which has a generator matrix like this will this result in a catastrophic encoder?

(Refer Slide Time: 00:57)



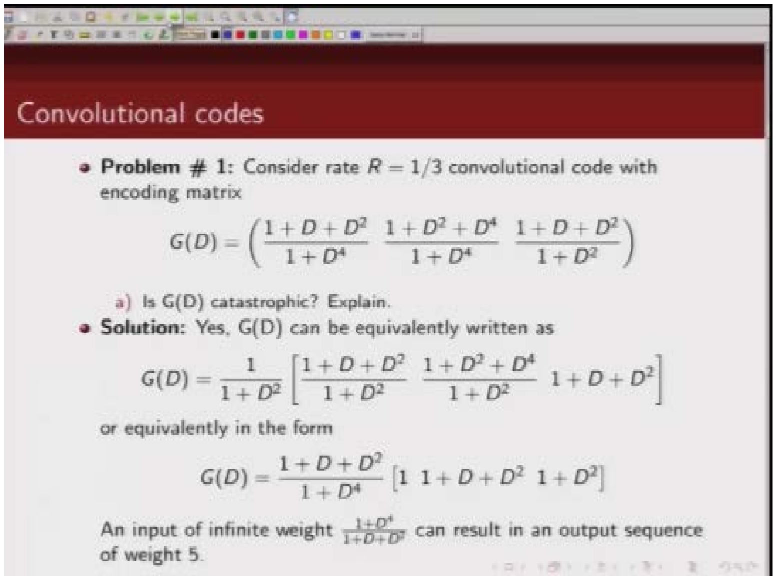
So if you recall, what is a catastrophic encoder? A catastrophic encoder generates the finite weight output corresponding to an infinite weight input sequence. Now if you try to look it in terms of state diagram, in a state away from all zero state there is a self loop around a state where a nonzero input results in all zero output right.

(Refer Slide Time: 01:31)



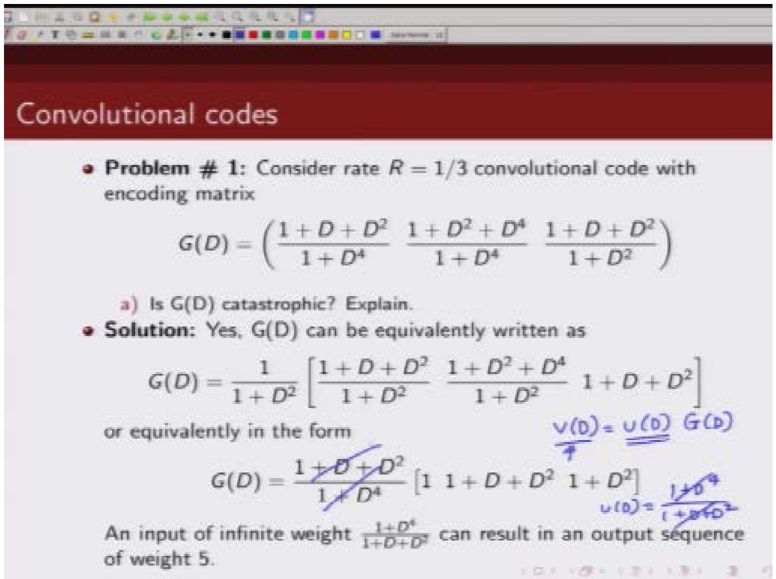
Now let us look at this generator matrix and let us try to simplify, put it in a minimal form. So we can see the denominator this $1+D^2$ is common. So if we take that out we get here $1+D+D^2$ and this is $1+D^2$, this is $1+D^2+D^4/1+D^2$ and this is $1+D+D^2$. Similarly we see in the numerator there is a common term $1+D+D^2$, if we take that out, what we get here is then this is 1 $1+D+D^2$ and $1+D^2$. Now how do we know whether this will result in a catastrophic encoder or not?

(Refer Slide Time: 02:17)



So look at this particular generator matrix, now what is my output sequence?

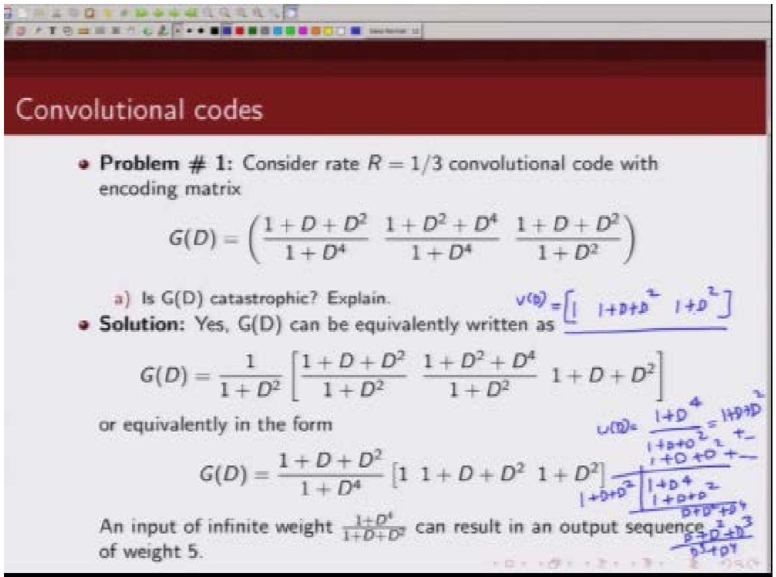
(Refer Slide Time: 02:23)



My output sequence v(D) is u(D) times G(D). Now is there any input sequence which is of infinite weight, but can result in a finite output weight for v(D), now if you pay close attention to G(D) we notice that if our input u(D) is chosen as $1+D^4$, $1+D+D^2$.

If our input is chosen in this particular fashion then what will be the corresponding output v(D). If the input is chosen this way then output will be u(D) times v(D), so this term will cancel these term, so what you will be left with is this. So your v(D) would be 1 $1+D+D^2$ and $1+D^2$ and what is the weight of this [Other language][00:03:27]..जीतकर दिया वो गन्धा वो - Jeet Kar diya, woh gandha woh

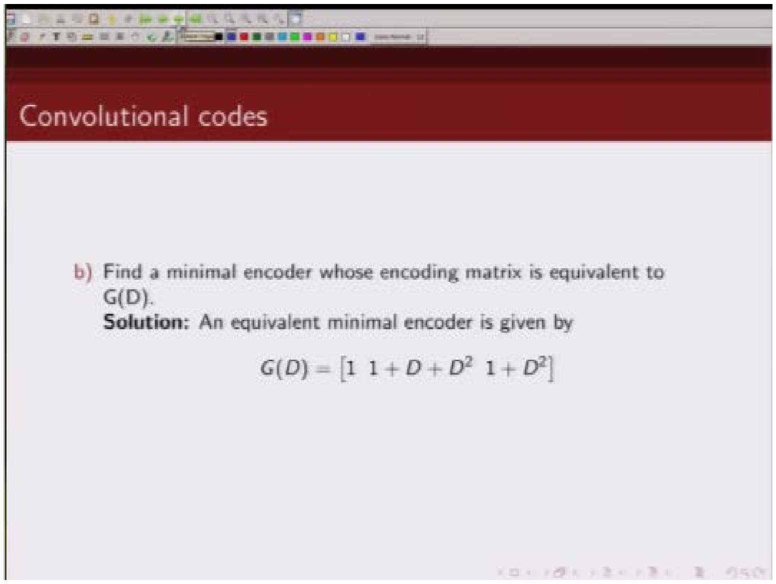
(Refer Slide Time: 03:31)



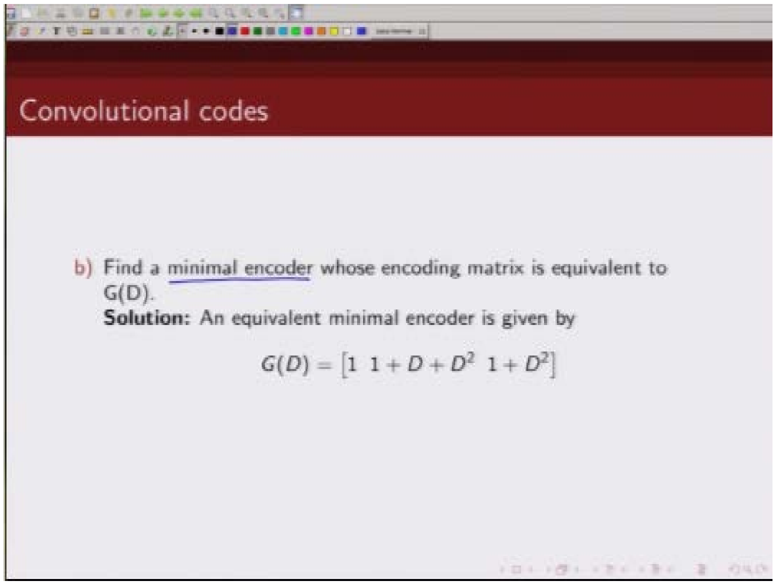
So note here, so the input that will cause this output is given by $1+D^4/1+D+D^2$ right. Now we can expand this, so let us say $1+D^4$ this is $1+D+D^2$, so let us just take 1 $1+D+D^2$ this will be $D+D^2+D^4$, now this will be +D, this will be $D+D^2+D^3$ then this will be D^3+D^4 we can write D^2, so like that basically we can see that this is a infinite series.

The input is an infinite series, $1+D+D^2$ is essentially are infinite series; we can expand it like that. Whereas output is the finite series, it is just 1 $1+D+D^2$ and the third bit is $1+D^2$. So you can see input has lots of ones in it, but the output has finite ones. So this is a case of catastrophic encoder.

(Refer Slide Time: 05:08)

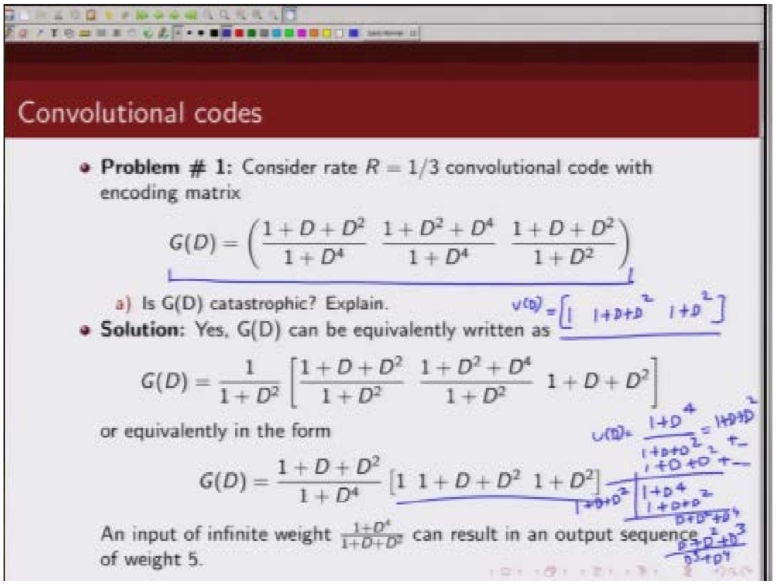


(Refer Slide Time: 05:11)



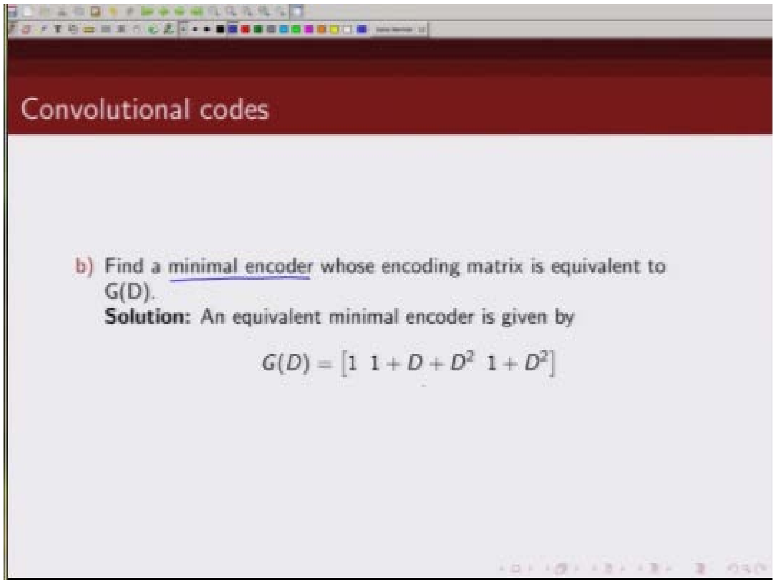
Now the second question is what would be the minimal encoding matrix for the generator matrix given in the previous example.

(Refer Slide Time: 05:18)



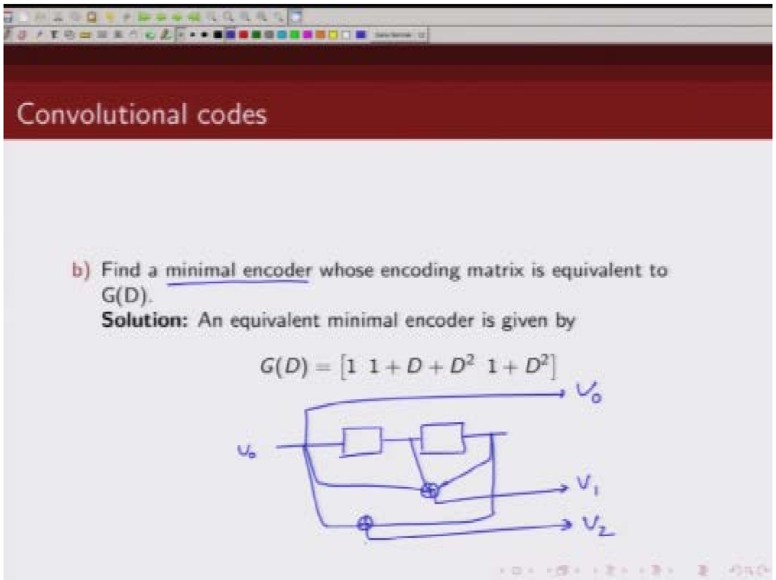
So if I ask you find out the minimal encoding matrix for this encoder, so what do we do, we take out all the common factors, so if we take out common factors when we basically what we get is like this is our minimal encoding matrix.

(Refer Slide Time: 05:37)



And if we can write, if I ask you to draw this encoder.

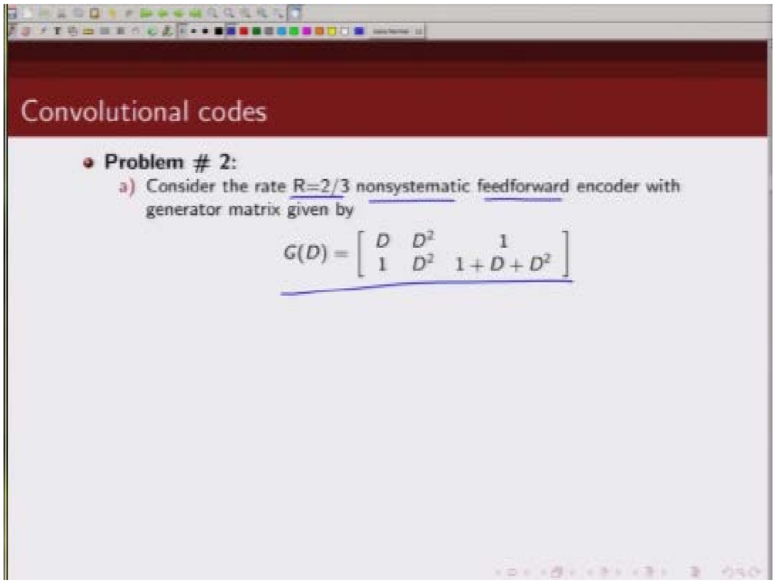
(Refer Slide Time: 05:46)



We can – this is case 1, n is 3, the maximum memory is two, so I am drawing two memory elements here.

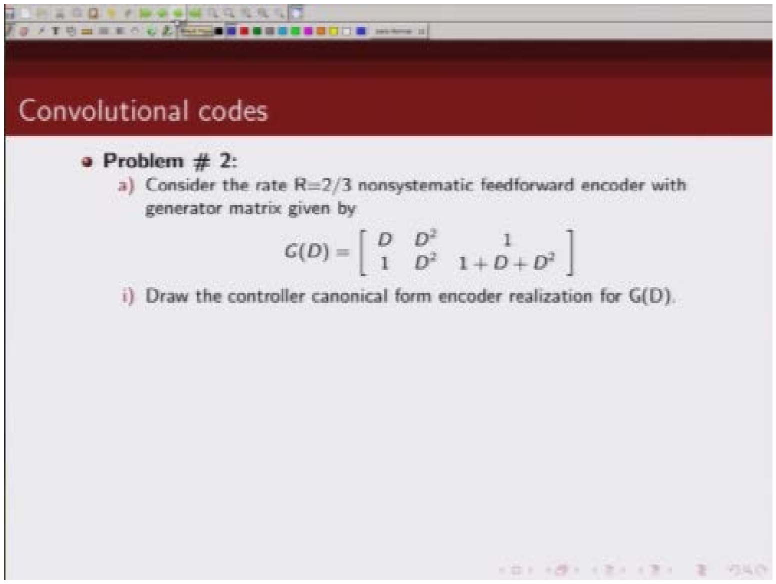
The first coded bit is just 1, so this is the information sequence that goes in, second one is $1+D+D^2$, so that is your let us call it v0, this is v1, this is u0 and the third bit coded bit is 1 and D^2 this is your v2 okay. So this is the minimal encoder for the same generated matrix given in the previous example.

(Refer Slide Time: 06:39)



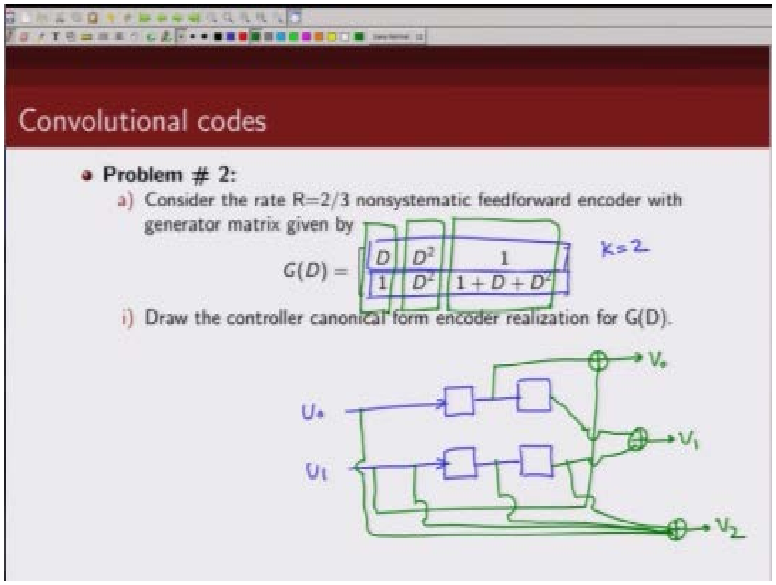
So consider a rate 2/3 nonsystematic feedforward encoder. So this is a generator matrix for a nonsystematic code rate 2/3 and it is a feedforward encoder, there are no feedback polynomials here.

(Refer Slide Time: 06:57)



The first question is draw the controller canonical form realization for this generator matrix. Now in controller canonical form realization we have one set of shift register for input. Now how many inputs do we have here, case 2.

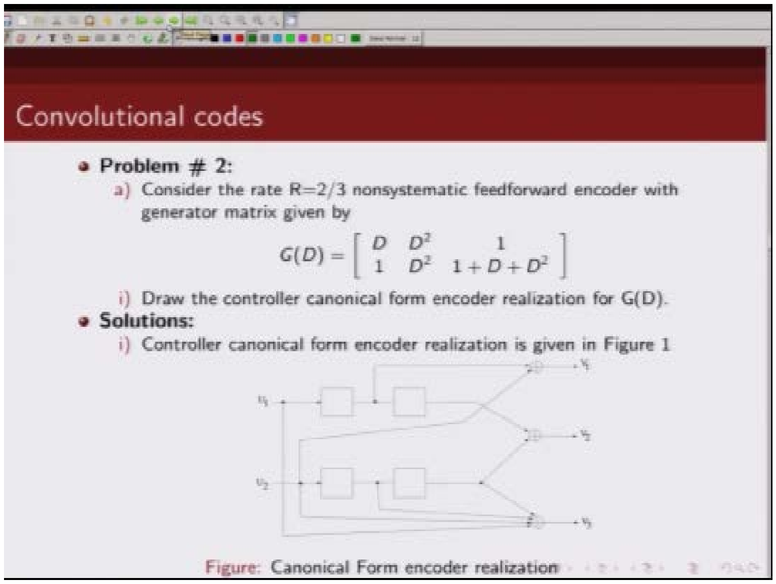
(Refer Slide Time: 07:23)



So we will have two sets of shift registers for this. One for this and second set of shift register for this. Now what should be the maximum memory for each of this shift register, you can see here the maximum power of D is 2. So we should have two memory elements for the first input. Similarly for the second input also we should have two memory elements. Let us call it u0 and u1.

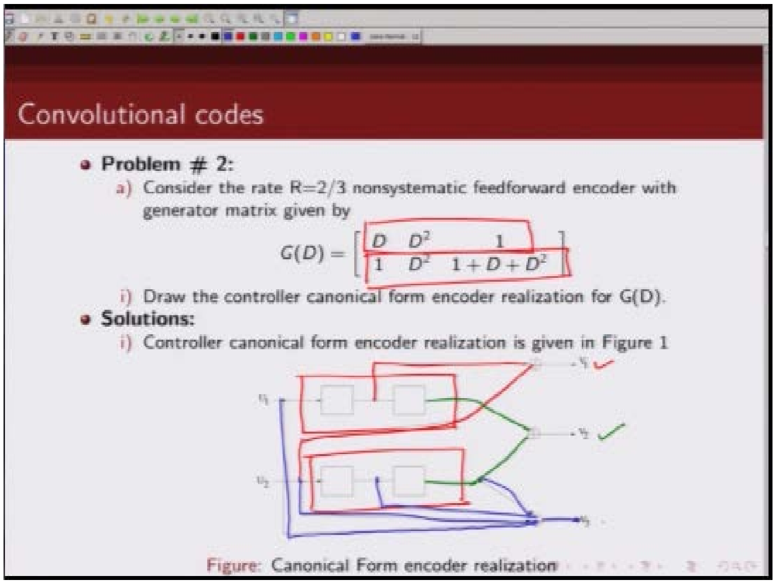
Now there are 3 outputs so the outputs are this is 1 output D times the first and 1 times u_1 so D times the first input is this and 1 times second input so that is this, so this is your first coded bit let us call it $v_{0,}$ now what is the second code bit, this is this term D^2 u_0 is this term, and $D^{2\ u}{}_1$ is this is term so this is your $V_{1,}$ and the third output is this so this is just a minute u_1 and 1 D term and D^2 term so this is your controller canonical form realization for this generator matrix.

(Refer Slide Time: 09:23)



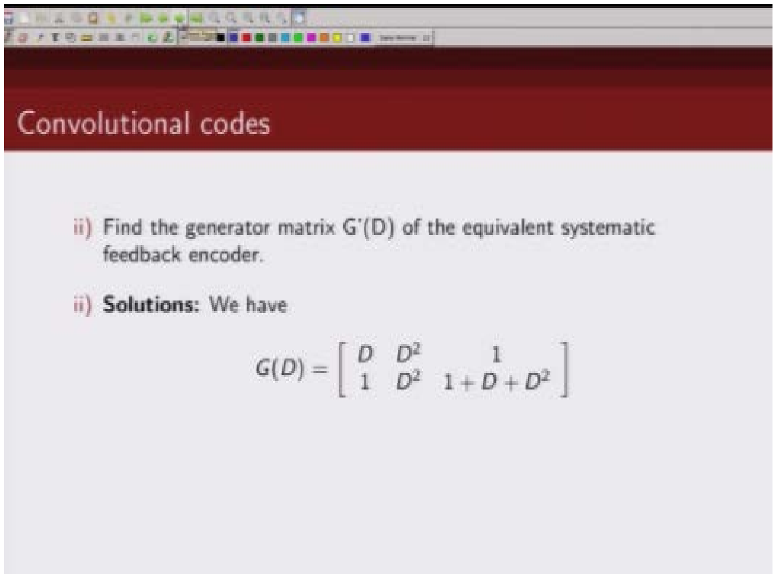
So this is precisely what we have here we can see

(Refer Slide Time: 09:35)



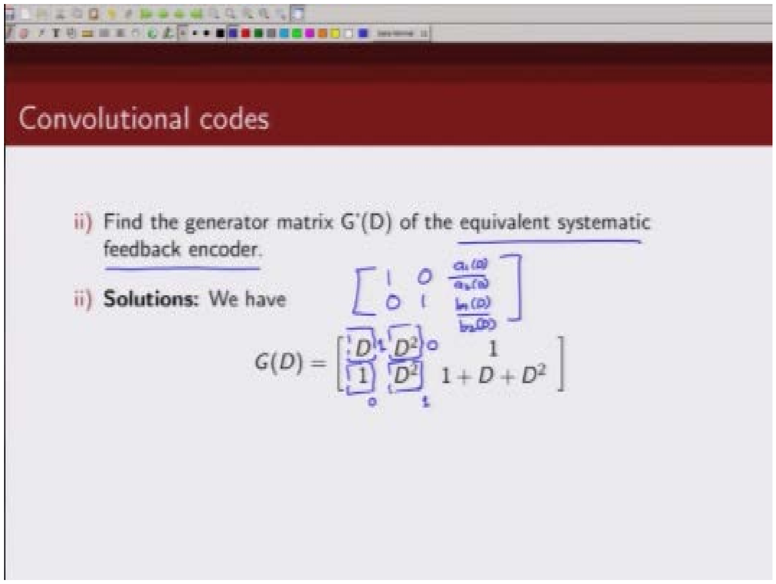
So this shift register is for this input and this shift register is for this input, maximum memory element for the first one is 2, second one also 2, and we can see now the first output is D times u_1 which is this plus u_2 times this, the second output is D^2 times u_1 and D^2 times u_2 so that is this, and the third output is u_1 which is this and this is u_1 D times u_1 D^2 times u_1 so that is your third output okay.

(Refer Slide Time: 10:35)



Now this was a non systematic encoder can we find an

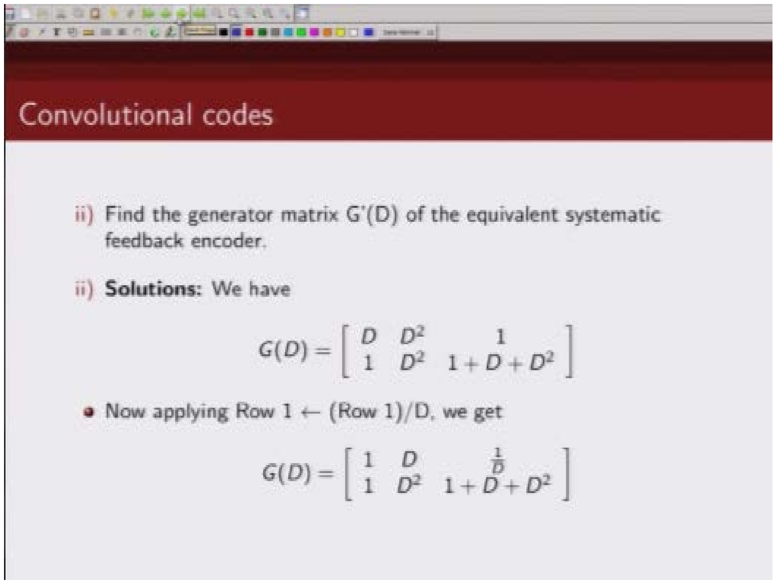
(Refer Slide Time: 10:45)



Equivalent systematic encoder or systematic encoding matrix for this generator matrix, the answer is yes so how do we find a systematic encoding matrix, so this has to be put in the form like this 10 01 and some matrix here, let us call it a_1 D times a_2 D and b_1 D times b_2 D, so we will have to bring this matrix in this particular form so we have to get this to 1, this 2 this has to be changed to 1, this has to be brought to 0, this has this we have to brought to 1 and this we have to brought to bring to 0.

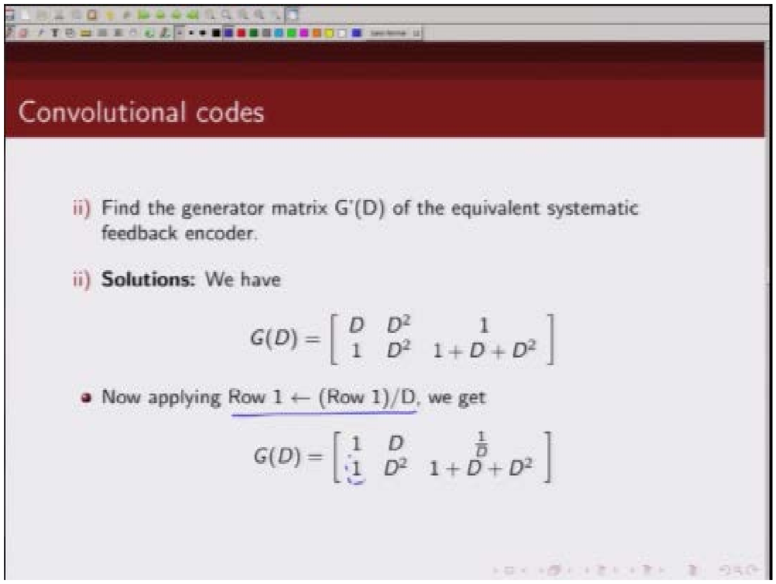
Now we will do elementary row operation to get an identity matrix here so let us do that.

(Refer Slide Time: 11:45)



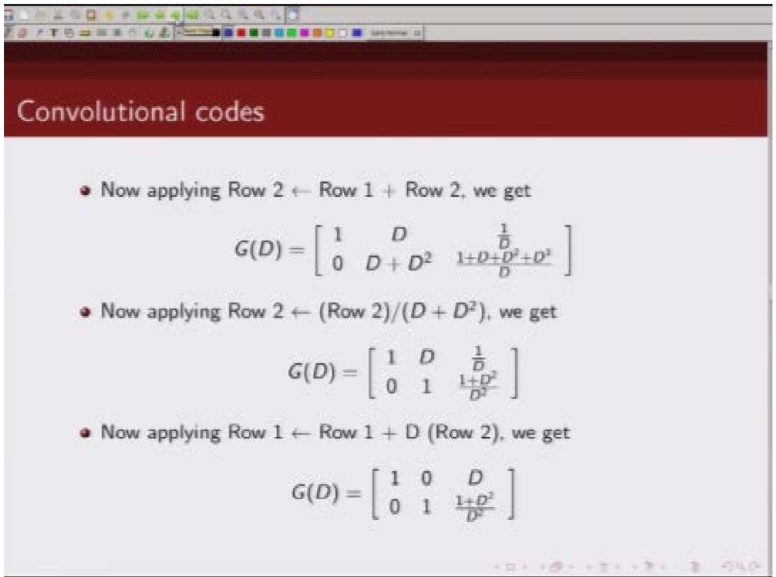
So first thing that we do is we make this a 1, how do we make this a 1, we do this transformation.

(Refer Slide Time: 11:55)



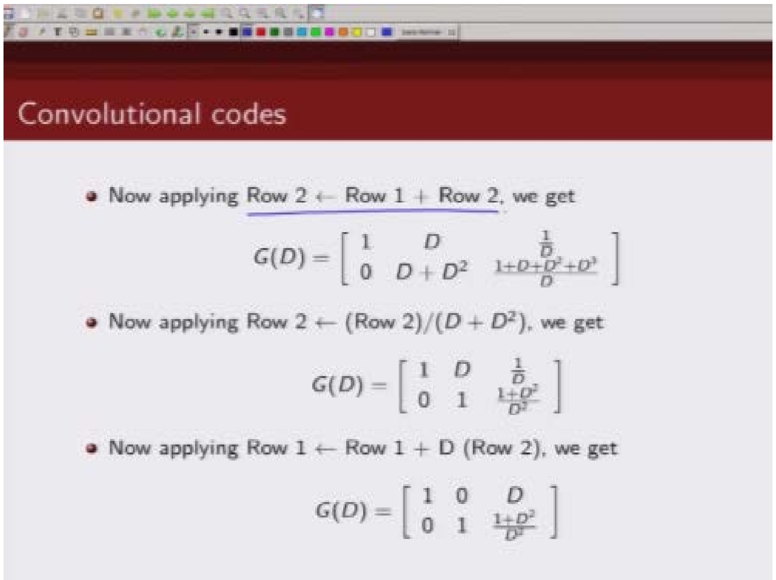
That Row 1 is Row 1/D so we divide this whole thing by D, what we get is 1 D 1/D, next we would like to get a 0 here.

(Refer Slide Time: 12:17)



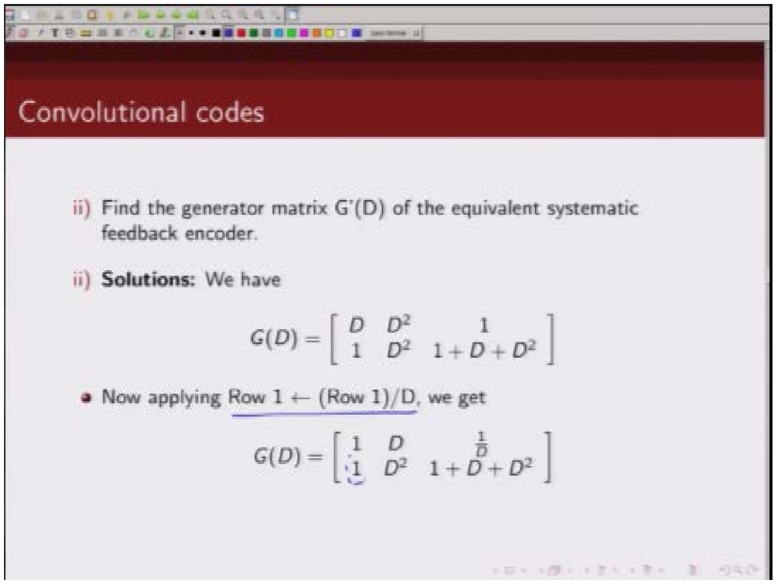
Here we would like to get a 0, how can we get a 0 here, so we will do this transformation.

(Refer Slide Time: 12:19)



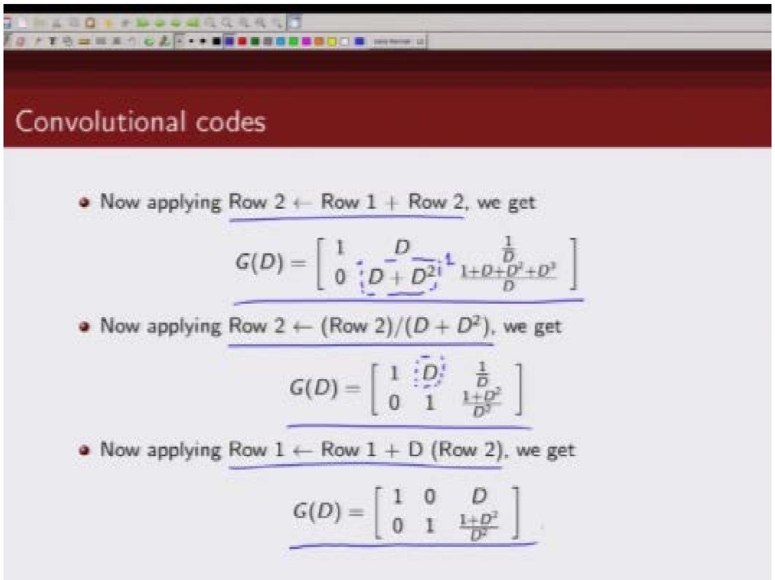
Row 2 is Row 1 + Row2 if we do that so.

(Refer Slide Time: 12:28)



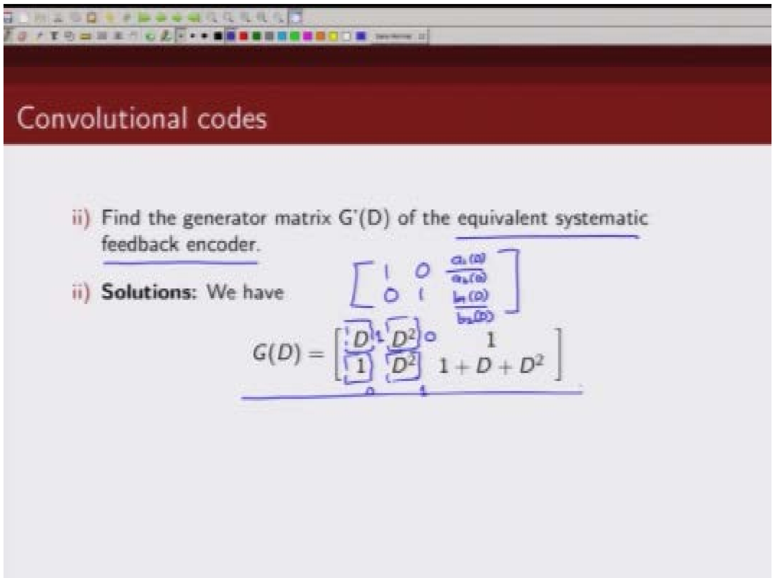
We add these two this will become 0, this will become $D + D^2$ and what we will get

(Refer Slide Time: 12:34)



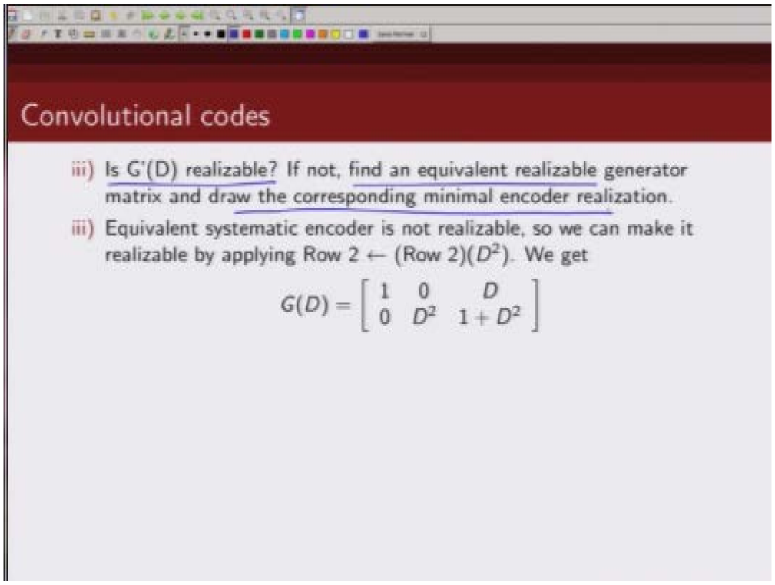
Is this, next we would like to get a 1here, like to get a 1 here, how can we get a 1 here, we divide Row 2 by this so we do this transformation that Row 2 is Row 2 / D+ D^2, and once we do that we get this, next we would like to get a 0 here right, how do we get a 0 here, we multiply Row 2/ D and add it to Row 1 so we do this transformation that Row 1 is Row 1 + D times Row 2 and when we do that we get this, so this is our equivalent systematic encoder.

(Refer Slide Time: 13:34)



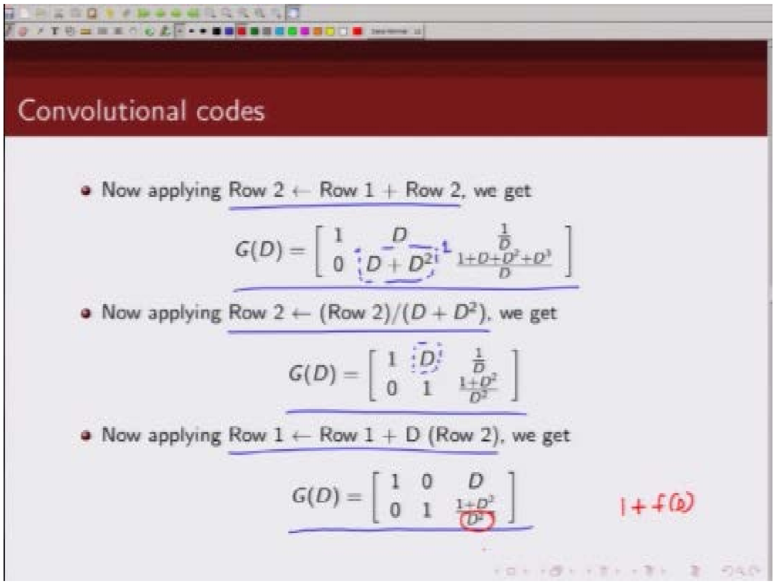
For the generator matrix this okay.

(Refer Slide Time: 13:40)



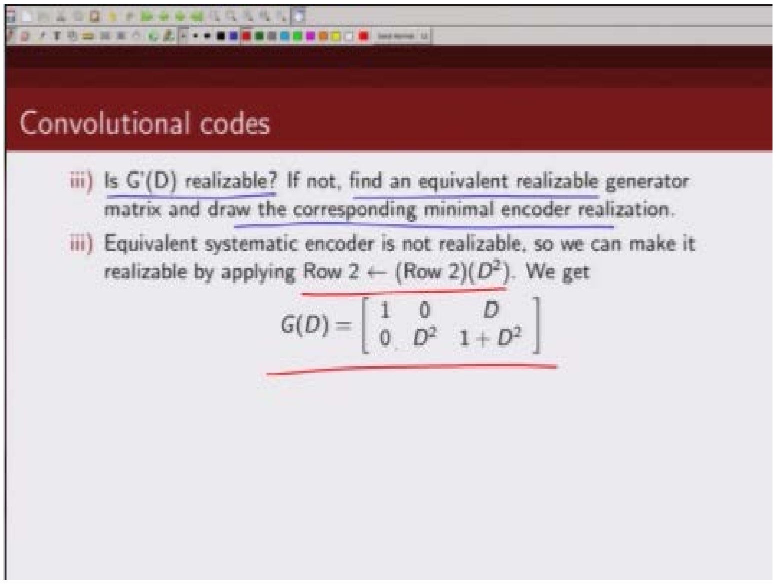
Next is this equivalent systematic generator matrix is it realizable, if it is not find out an equivalent realizable generator matrix and draw its corresponding minimal encoder realization now note here.

(Refer Slide Time: 14:00)



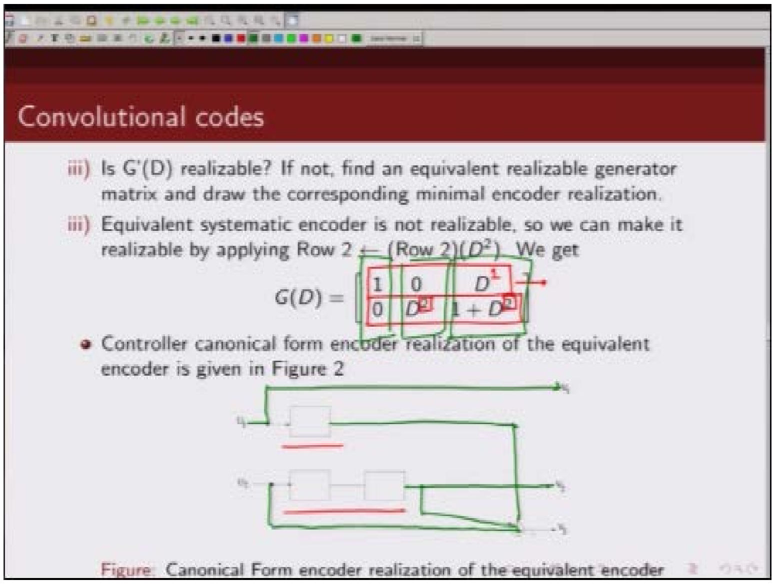
This generator matrix has a term $1+D^2$ in the denominator now this cannot be realizable, so any denominator term that we have it has to be of the form 1+, some polynomial here but here this 1 is not here so we cannot realize of a rational function of this form using our shift register, so this particular equivalent systematic encoder is not realizable however if you multiple this by D^2

(Refer Slide Time: 14:40)



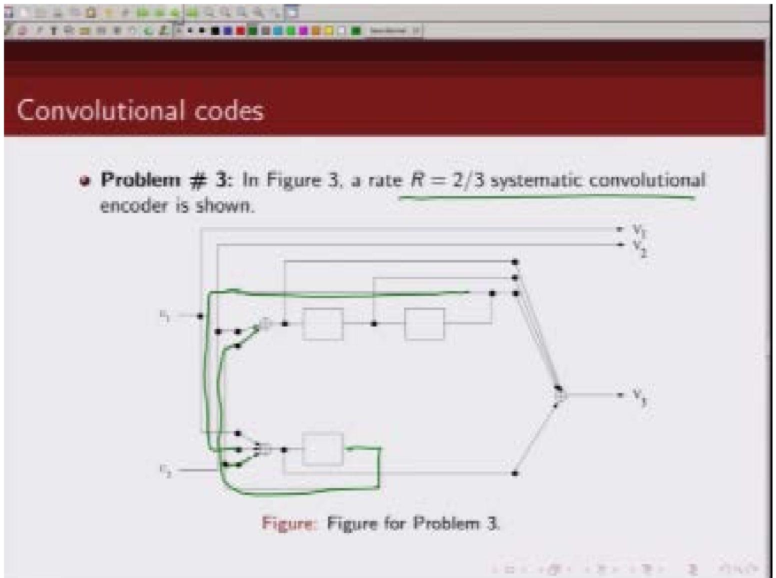
What we get if we do this transformation, what we get is this. This is no longer so what we are getting now is basically a new equivalent encoder which is in the feed forward form and it is realizable.

(Refer Slide Time: 15:01)



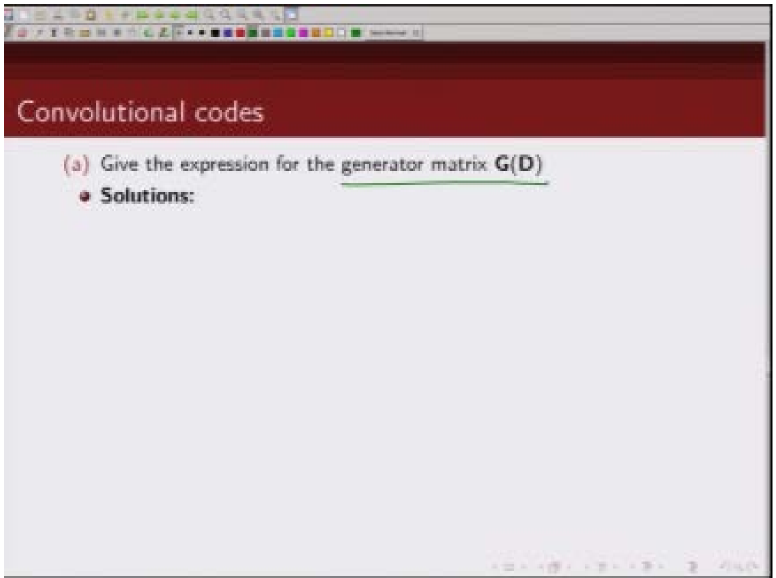
So how do we realize it again if we are using controller canonical form realization we will have one set of shift register for this input another set of shift register for this input. What is the maximum memory for the first row, the maximum power of D is 1 so we will have 1 memory element for the first input and what is the maximum power of D for the second that is 2 here, 2 here, 2 here, so we will use 2 memory element for the second input and again what are our outputs, there are three outputs the firsts output is this. This is u_1 times this is just u_1 so this is this second one is D^2 of $u_{2,}$ so D^2 u_2 is just this term so this is my second output and the third output is this, D times u_1D. Which is this one and one times u2D and D2 times u2d so that's this, this is our 3rd output. Now given below is a rate to third

(Refer slide Time 16:26.1)



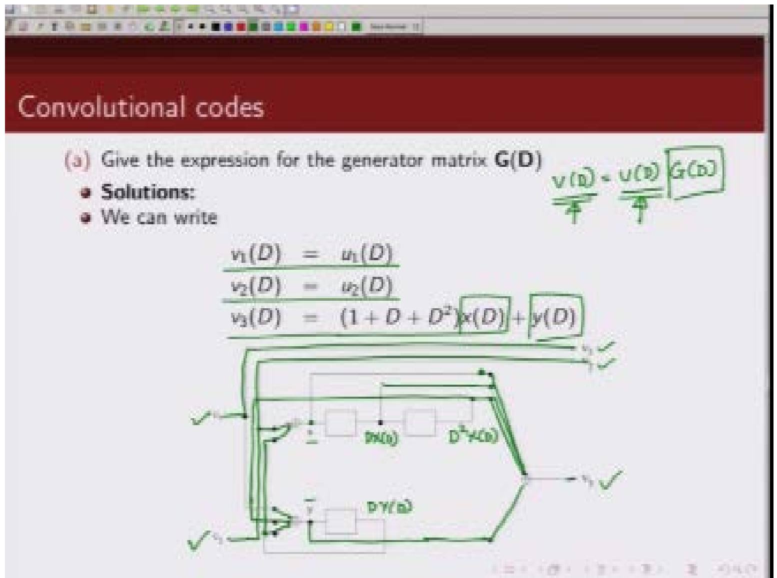
Systematic convolution encoder please note this is leader in the controller canonical form realization, odd in the absolver canonical form realization. Note here the feedback terms that are coming here are not only coming from the same encoders likes this, feedback is not only so if you look at the feedback, feedback from this is going to this encoder and feedback from here is going to this encoder, so not only in feedback is coming to the same encoder but it is also going to the other encoder, so this realization is a very compact realization the question that is been ask is.

(Refer slide Time 17:09.6)



Can you find out the generator matrix code of warning to this encoder, so how do we find the generator matrix?

(Refer slide Time 17:21.8)

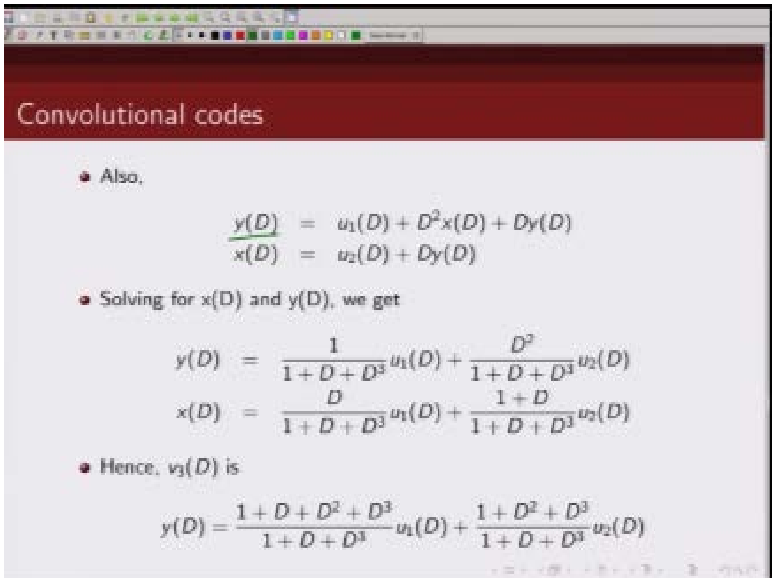


We know this is a relation between the input and the output, so how this inputs are getting map to the output that is governed by this generator matrix. So what we are going to do is we are going to write the output VD in terms of input UD, and then that would give us our generator matrix. So our objective is to write V1, V2, V3 in terms of U1 and U2. Find V use some auxiliary variables x and y which basically will help us find the contents here, so if this is X of D this term will be D times X of D and this will be D squared times X of D.

Similarly if this is Y this term will be D times Y of D. So what is V1 of D, V1 of D is U1 of D, you can see U directly goes, this input directly goes here, so V1 of D is U1 of D. Similarly this input UD directly goes the output here so V2 of D is U2 of D. Now what is V3 of D, V3 of D is this term which is X of D, this term D times X of D and this term which is D square X of D. So it is this term plus this term, so it is these 3 terms, now what is this term, this is Y of D so we have written V1 of D, V2 of D, V3 of D in terms of U1, U2 X of D and Y of D.

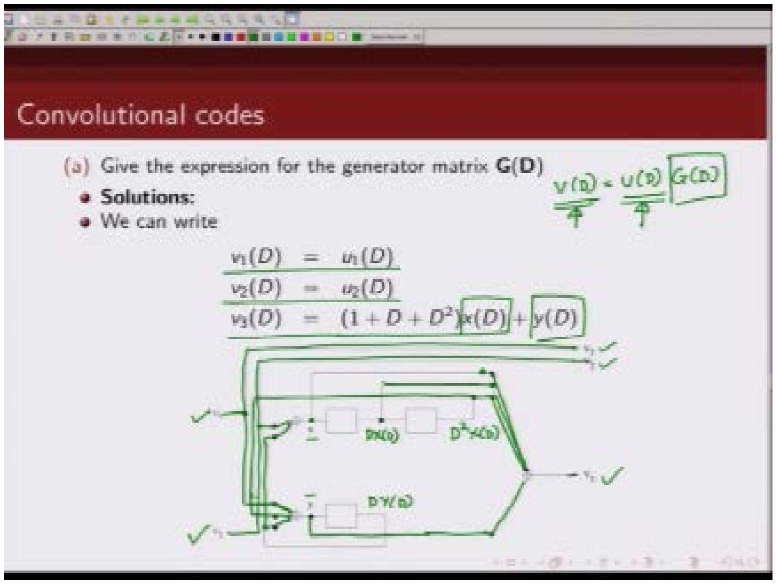
Now note we need to get rid of X of D and Y of D and we have to write these in terms of U1 and U2. Now what is X of D, X of D is this and this, similarly what is Y of D, Y of D is this term, this term, sorry this term and this term okay. So we can write two more equations for X of D and Y of D.

(Refer slide Time 20:15.1)



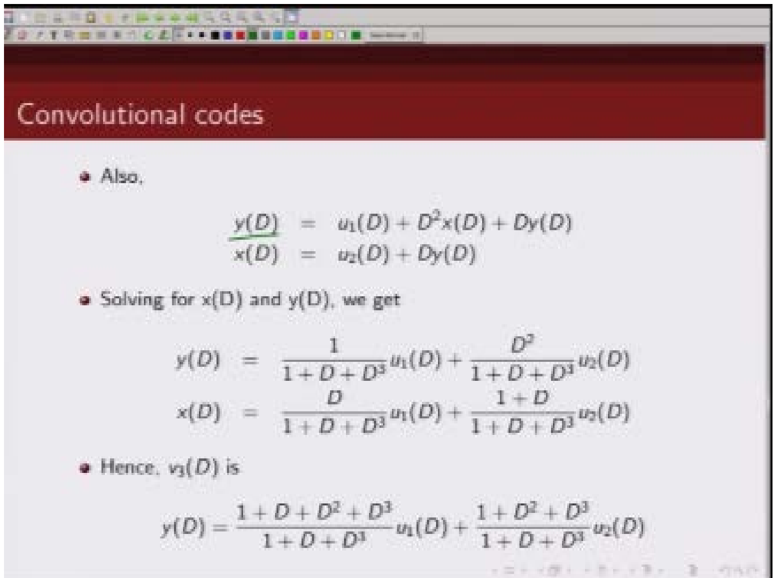
So again Y of D as I said is U1 of D, Y of D is U1 of D which is this one, this is U1of D.

(Refer slide Time 20:23.2)



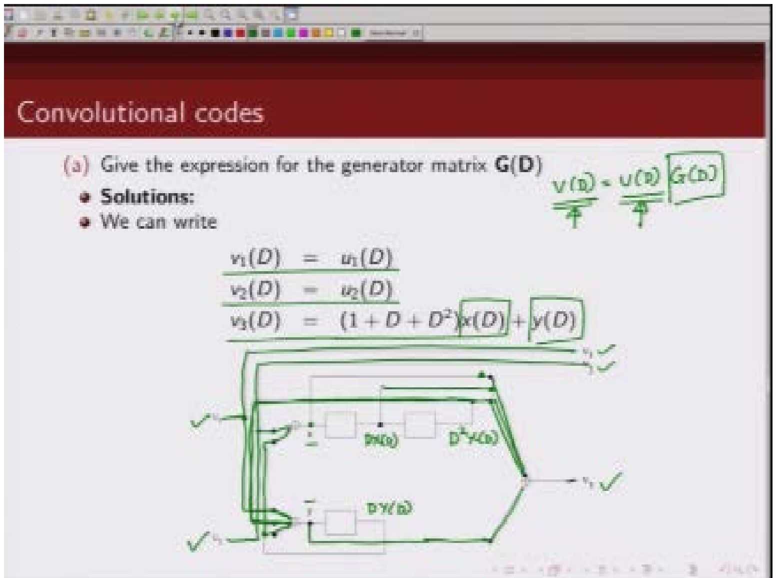
Plus D^2.

(Refer slide Time 20:29.8)



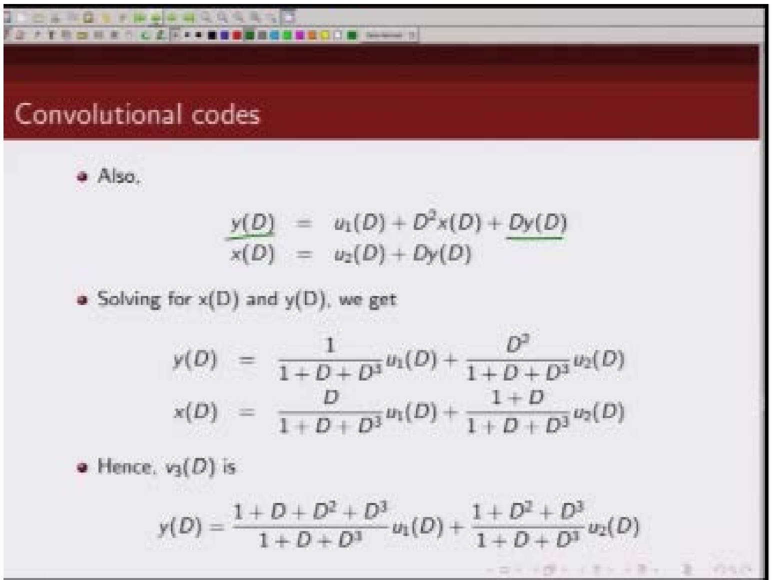
X of D.

(Refer slide Time 20:31.8)



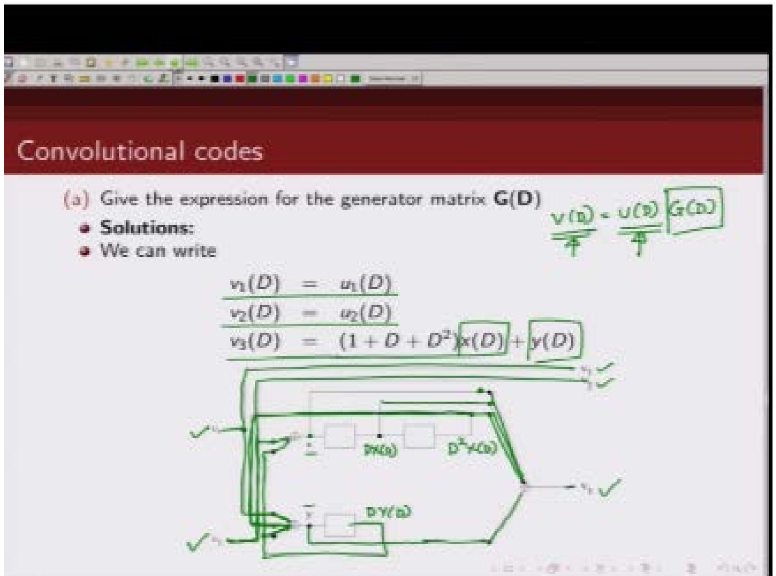
D² X of D is this term, D² is X of D is this term which is coming here, this term and there is another term here.

(Refer slide Time 20:47.8)



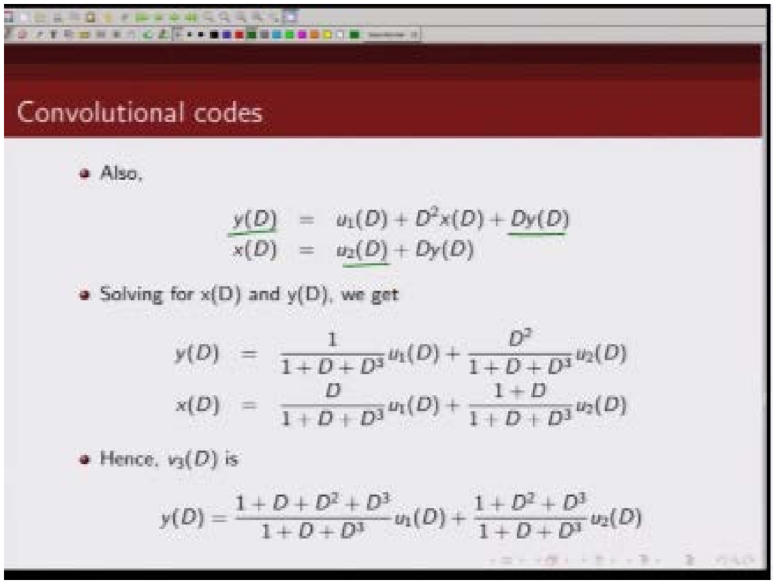
Which is D times Y of D.

(Refer slide Time 20:49.0)



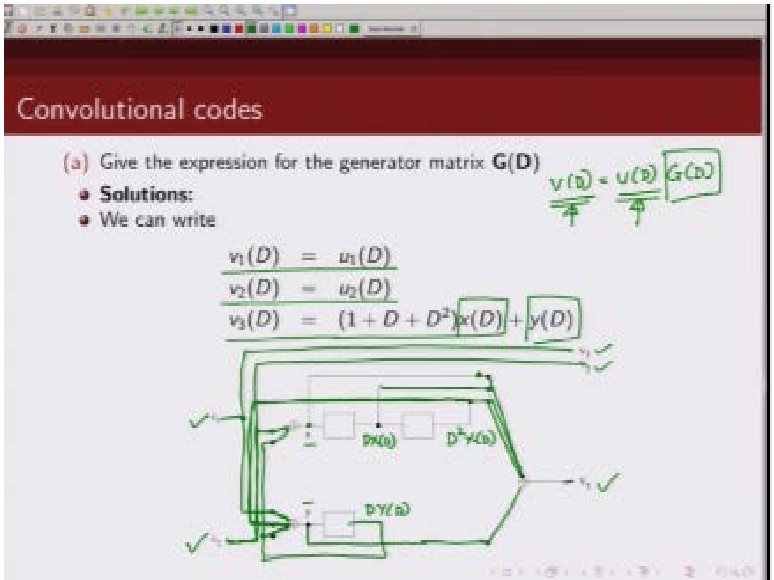
So D times Y of D note here the third input here is this one which is D times Y of D. Similarly X of D is first one is this term which is U of D.

(Refer slide Time 21:06.3)



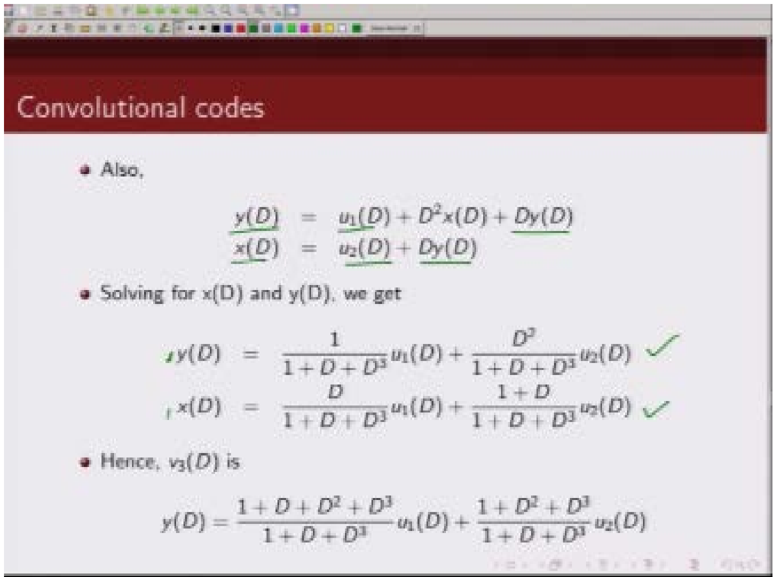
So this is U of D and the second term is.

(Refer slide Time 21:09.2)



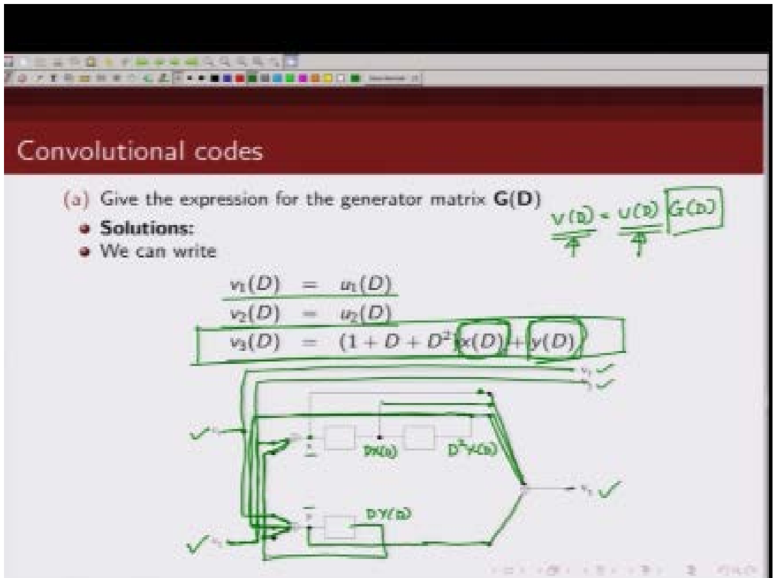
This term which is D times

(Refer slide Time 21:14.4)



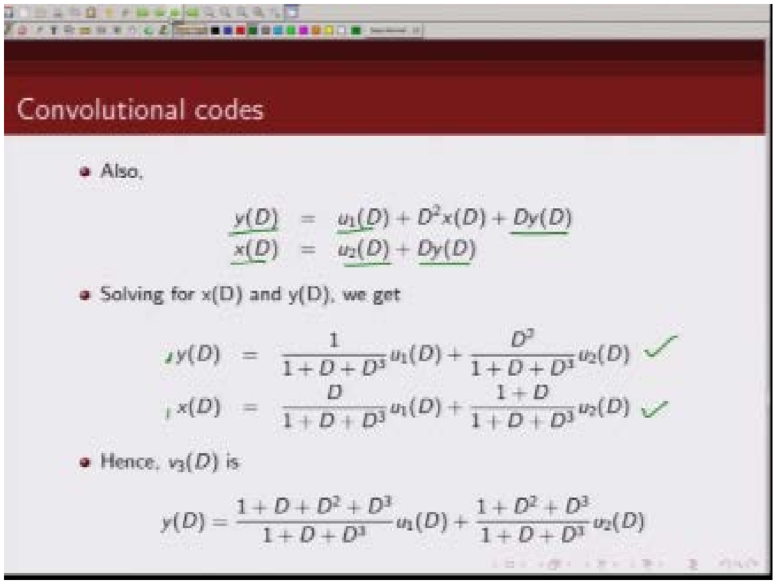
Y of D this one okay? So now we have got equations of Y of D, of X of D, in terms of U1D and U2 D so let us write bring Y of D at one side and X of D of one side and write them in terms of Y of D and X of D in terms of U and D and U2 D. So if you solve this what we get is Y of D is given by this and X of D is given by this, now we plug these values of Y of D and X of D given by this.

(Refer slide Time 21:54.3)



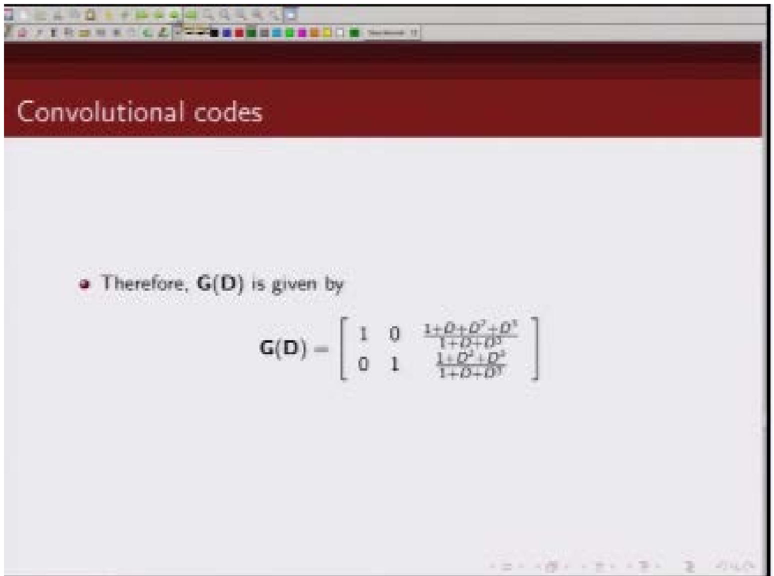
Into here, into this expression of Y3 of D, so we plug this value of XD and YD which we just compute it, we plug those values in here if we do that.

(Refer slide Time 22:10.7)



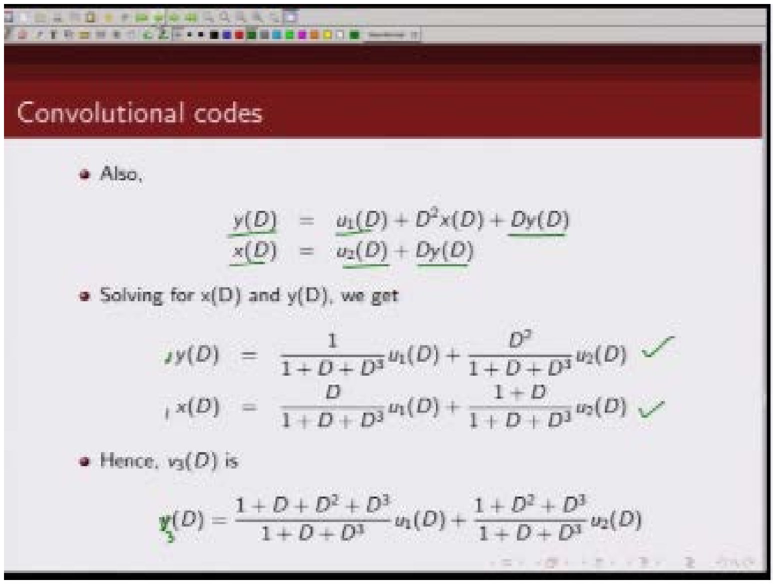
We will get the expression of Y3 of D, okay?

(Refer slide Time 22:18.3)



Now so if we do that

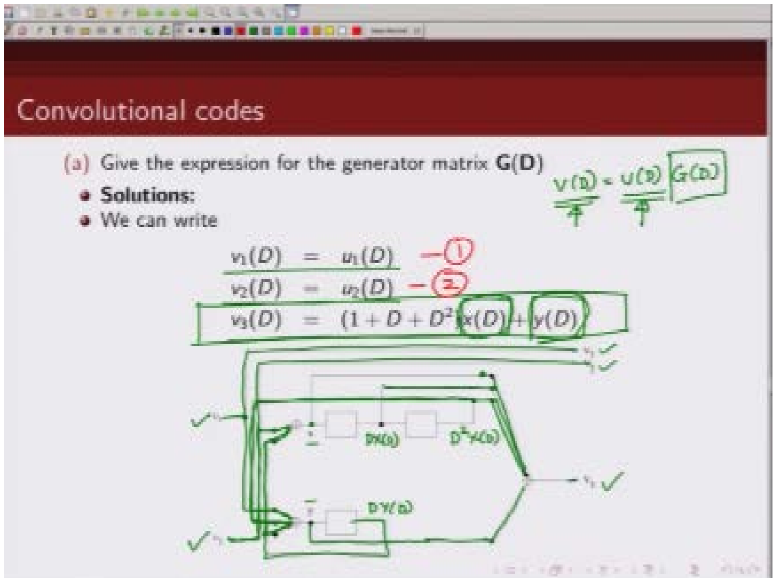
(Refer slide Time 22:21.3)



Finally this is 3 of D okay, so if we do that what we get is then V3 of D is this times U and D plus this time U2 of D.

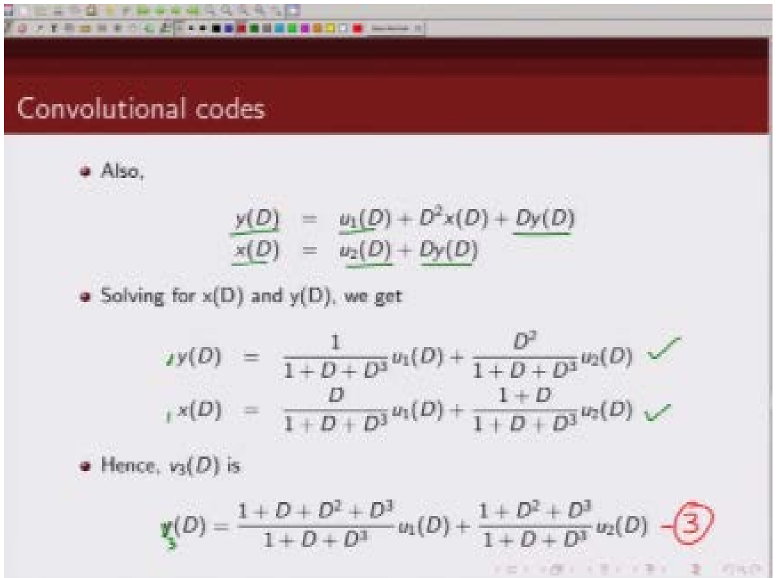
So now we are

(Refer slide Time 22:37.3)



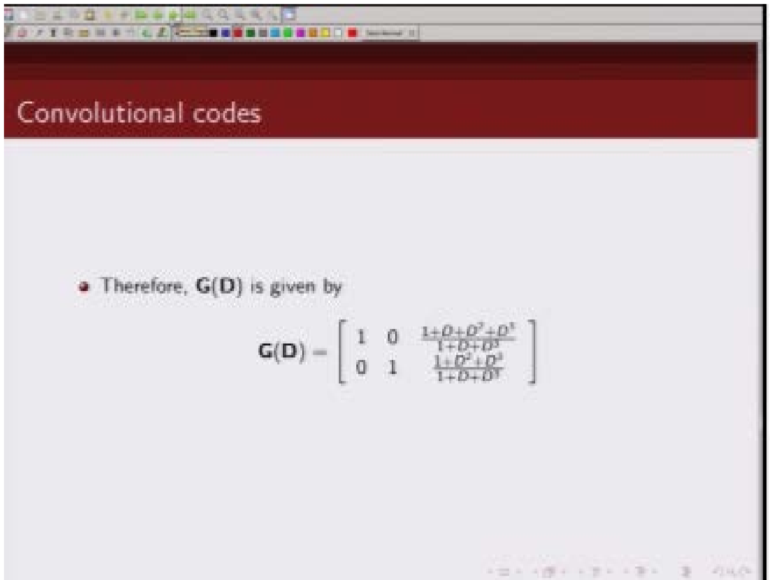
In a position to write the generator matrix the first equation that we will require is this one, second equation we will require is this one and

(Refer slide Time 22:48.2)



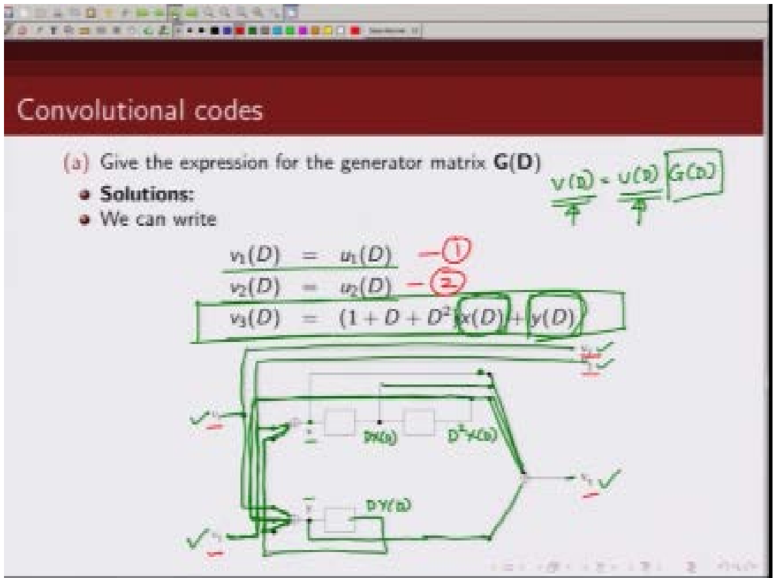
The third equation that we will require is this one.

(Refer slide Time 22:51.7)



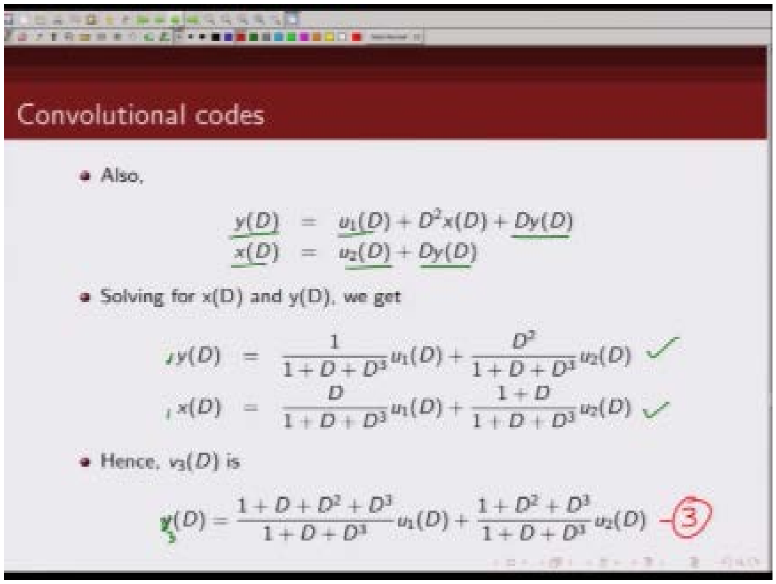
Right, so you can think of it as like this so we have

(Refer slide Time 22:56.4)

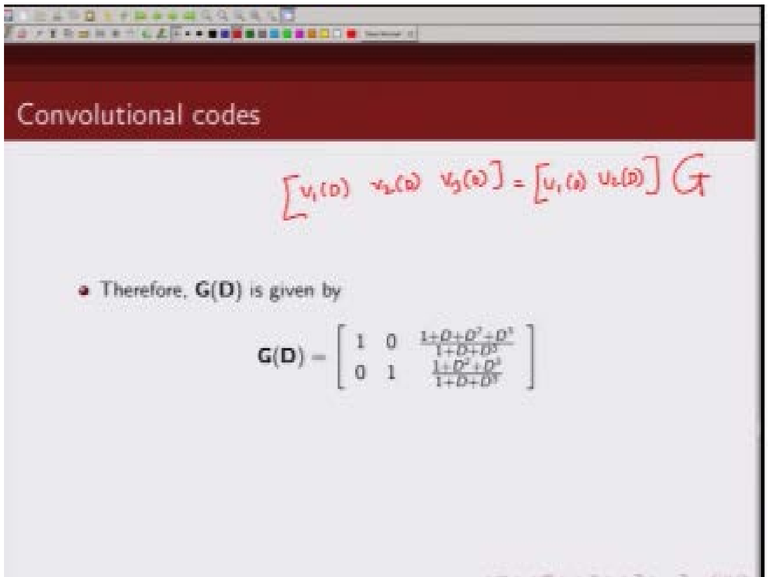


3 output V1, V2, V3, two input U1 U2.

(Refer slide Time 23:02.4)

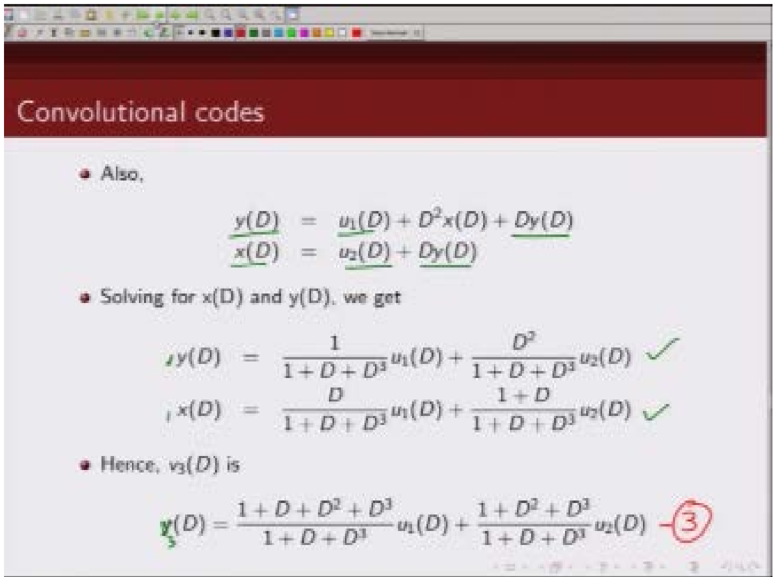


(Refer slide Time 23:03.5)



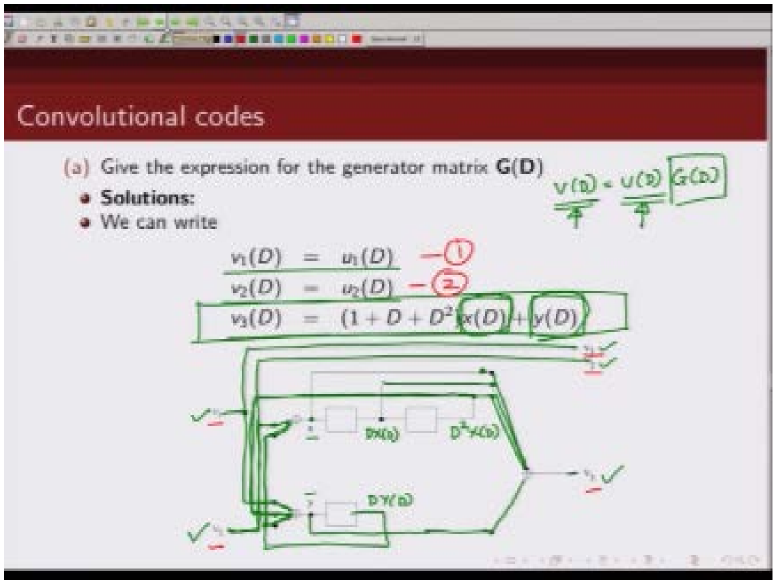
So we are writing V1 (D), V2 (D), V3 (D) in terms of U1 (D) U2 (D) and this G matrix so what is V1 (D)?

(Refer slide Time 23:23.9)



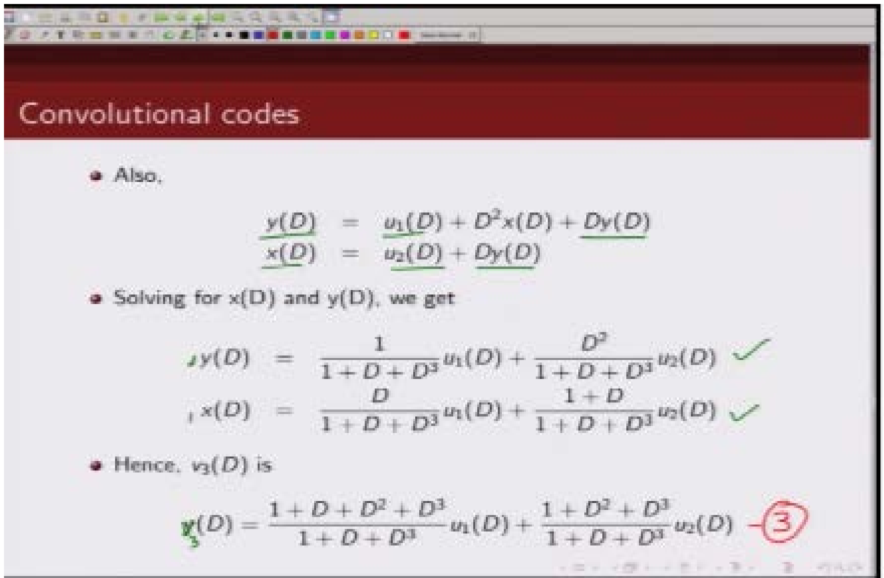
V1 (D) is

(Refer slide Time 23:24.4)



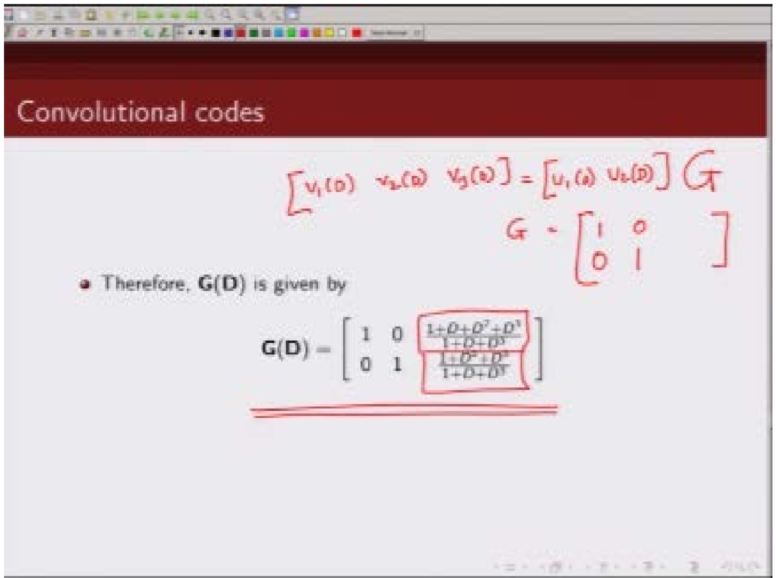
U1 of D.

(Refer slide Time 23:27.4)



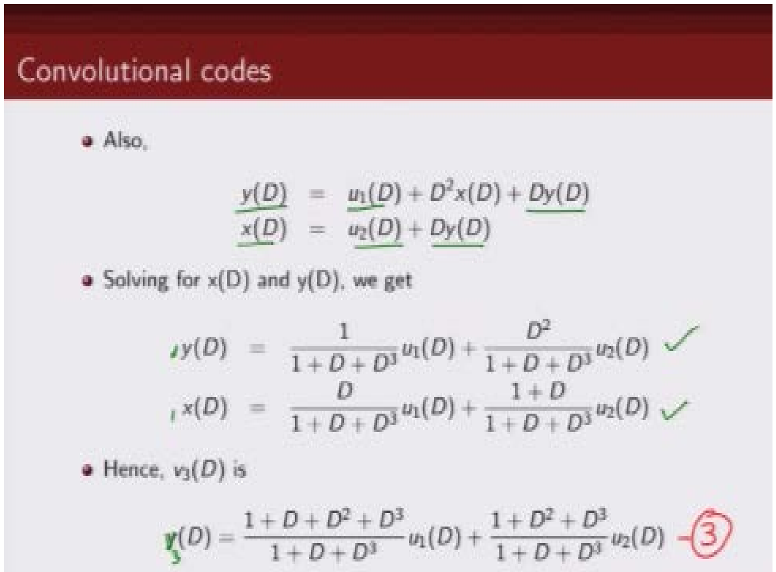
So then our G

(Refer slide Time 23:29.6)

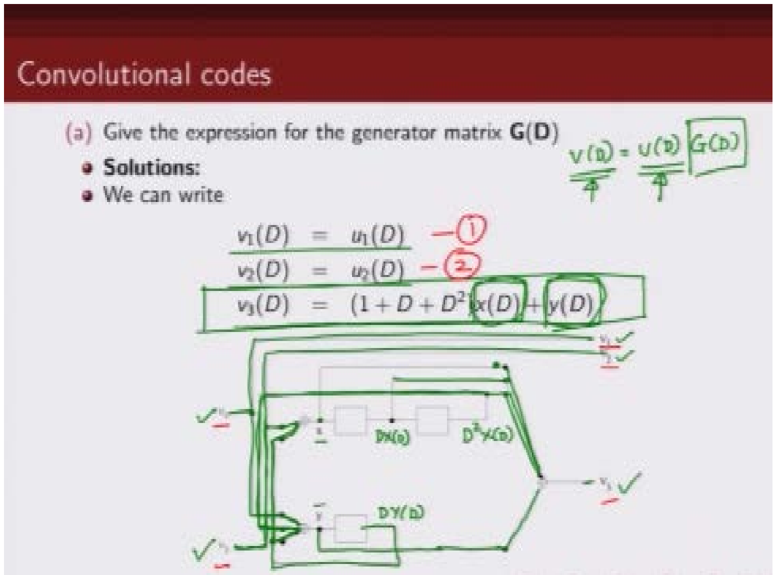


Matrix here, again this G matrix is 2 cross 3 so V1 (D) is U1 of D so we got one zero V2 of D is U2 (D) so we get 0 1, and what is V3 of D, V3 of D is this, this times U1 of D and this time U2 of D, so this will be our final generator matrix corresponding to the encoder that is shown in this figure, okay?

(Refer Slide Time: 24:01)

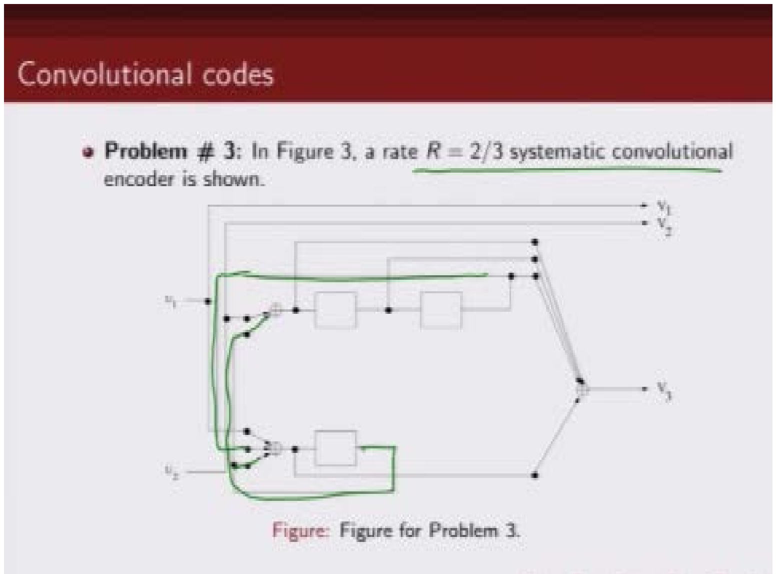


(Refer Slide Time: 24:02)

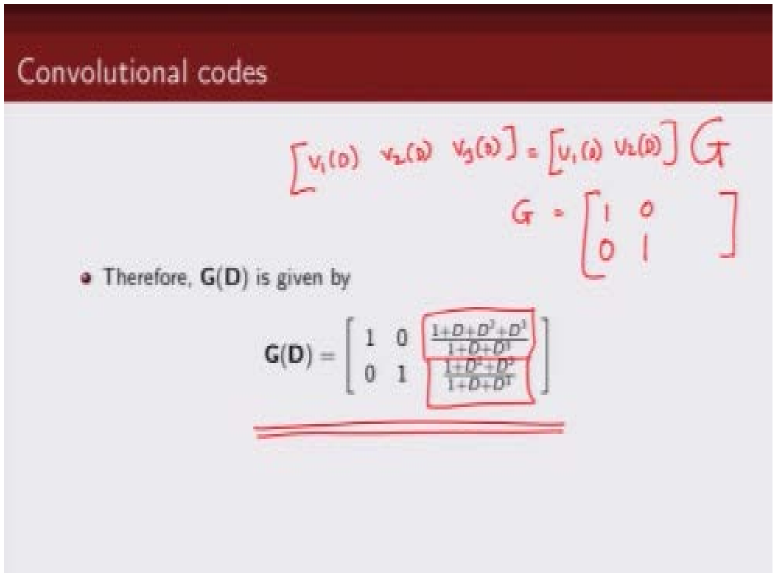


(Refer Slide Time: 24:03)

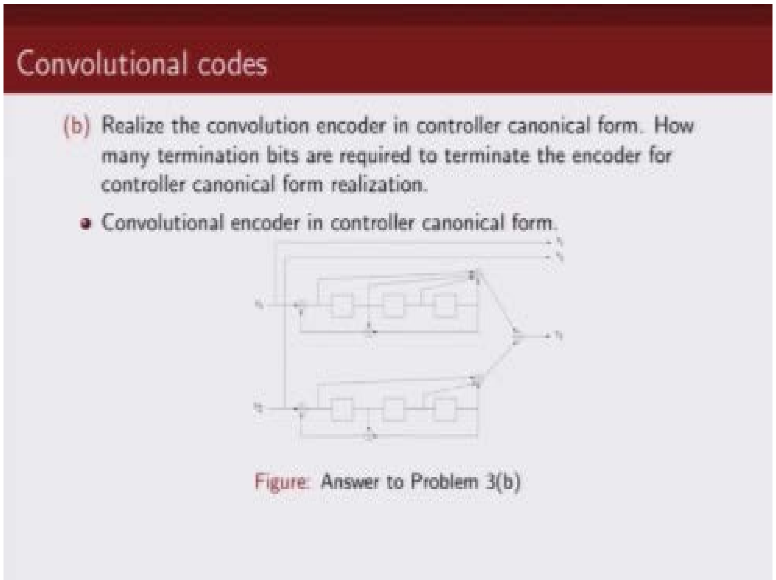
(Refer Slide Time: 24:06)



(Refer Slide Time: 24:09)



(Refer Slide Time: 24:10)



Now the next question is, can we realize this encoder in the controller canonical form? So the answer is yes we can realize it.

(Refer Slide Time: 24:15)

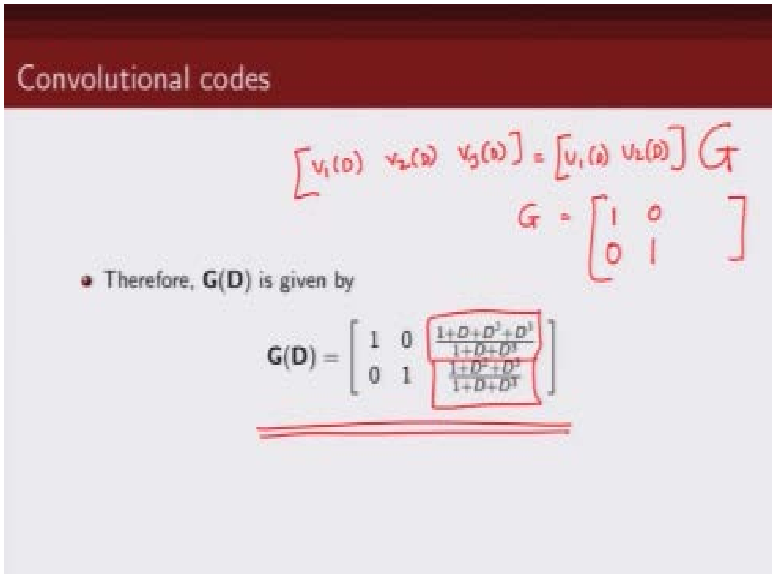
Convolutional codes

(b) Realize the convolution encoder in controller canonical form. How many termination bits are required to terminate the encoder for controller canonical form realization.

- Convolutional encoder in controller canonical form.

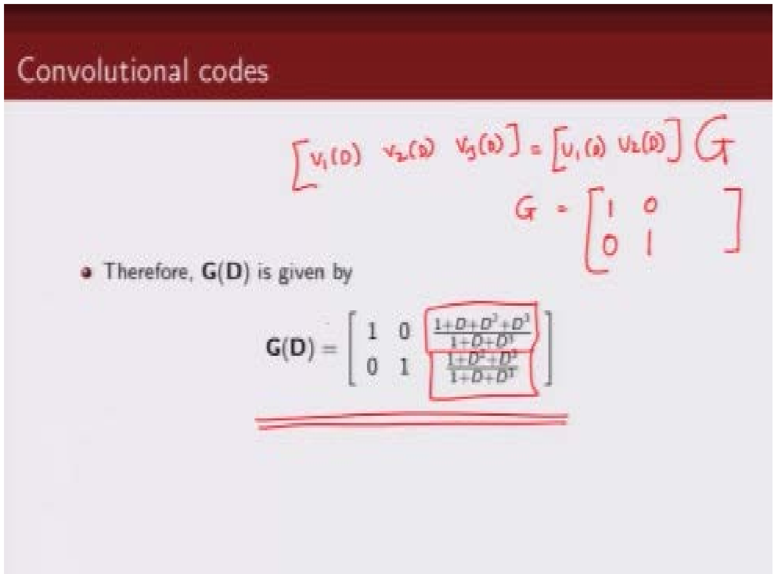
Figure: Answer to Problem 3(b)

(Refer Slide Time: 24:21)



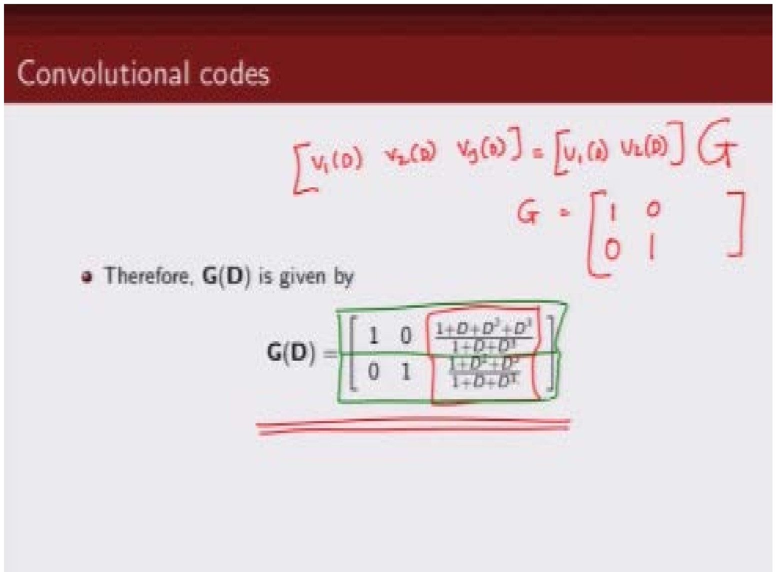
We already have the expression for generator matrix so to realize it in controller canonical form again.

(Refer Slide Time: 24:32)

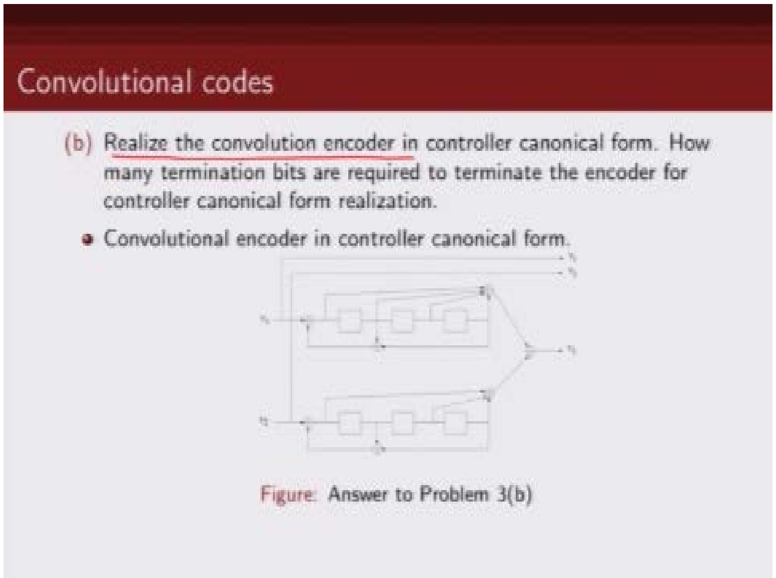


So there is one set of shift register for each input so these, this is one input, this is second input right? Please note this is a feedback polynomial so we would require a feedback a polynomial and now maximum degree here is three and maximum degree here is also three so.

(Refer Slide Time: 24:36)

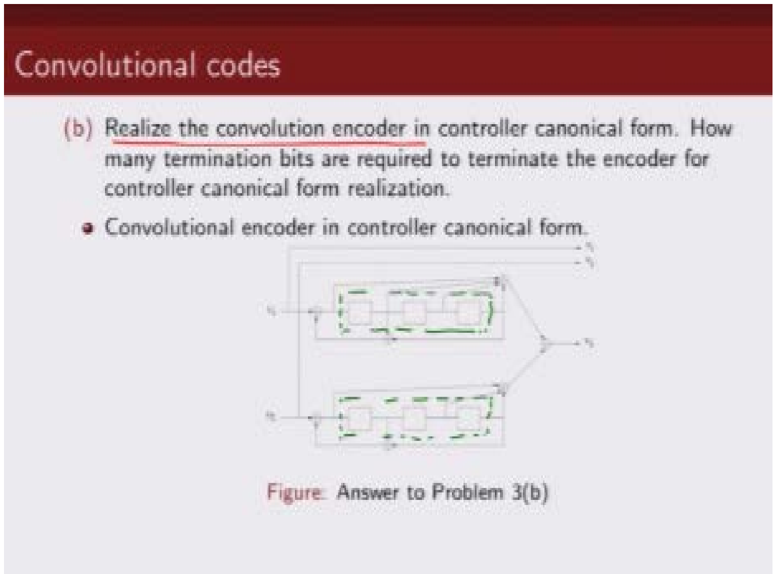


(Refer Slide Time: 24:53)

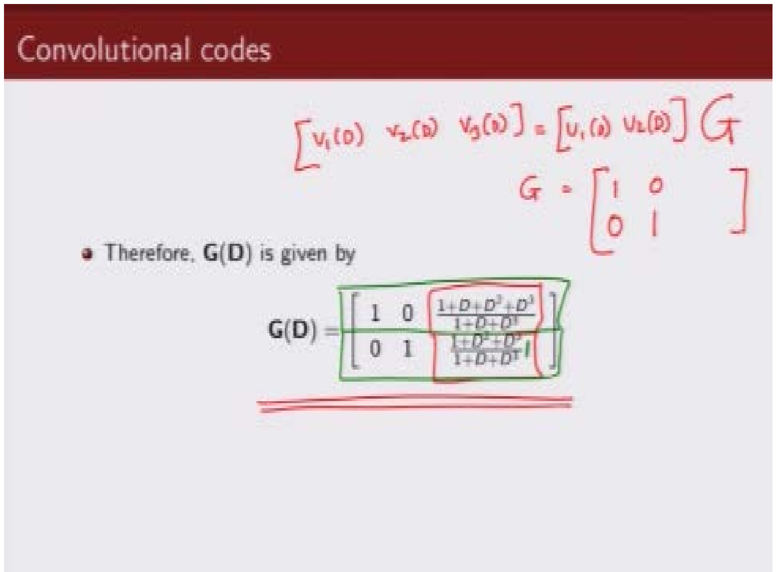


We will require two set of shift register first one is this one, please note this has three memory elements and similarly second shift register, this also has three memory elements.

(Refer Slide Time: 25:02)

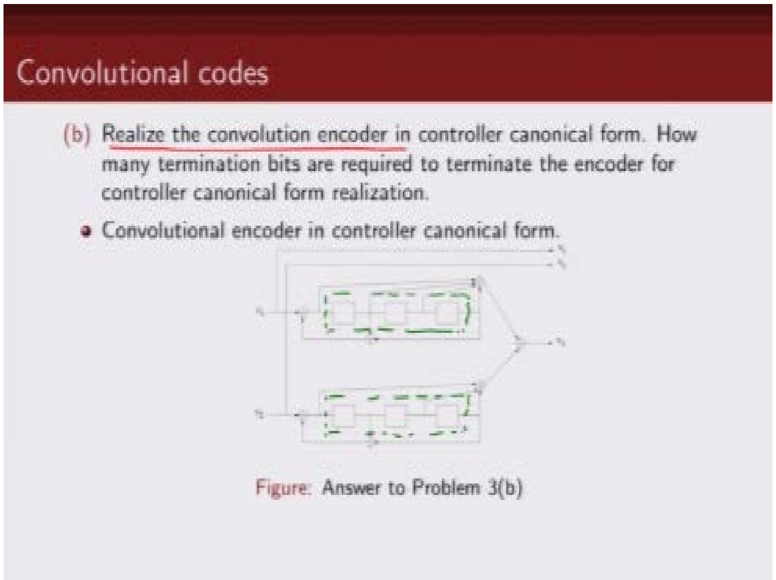


(Refer Slide Time: 25:15)



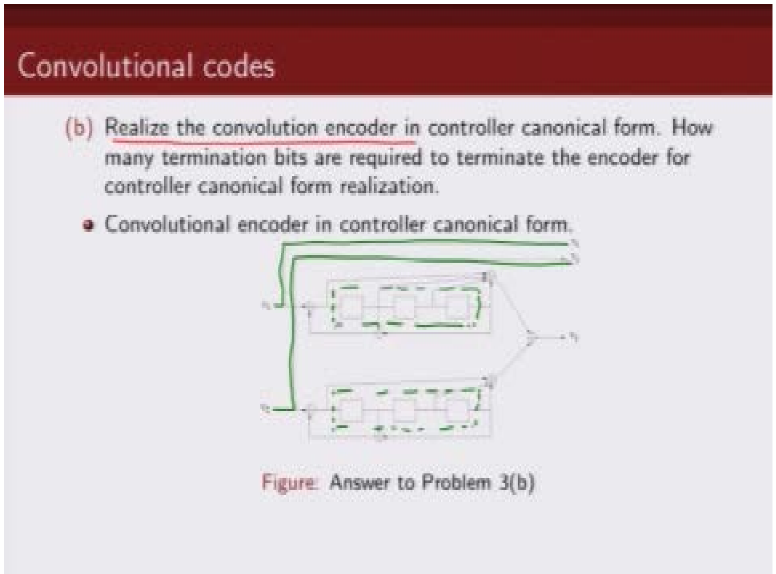
That is because the maximum degree of this rational function is three and similarly maximum degree of this rational function is three and we just implement this.

(Refer Slide Time: 25:27)



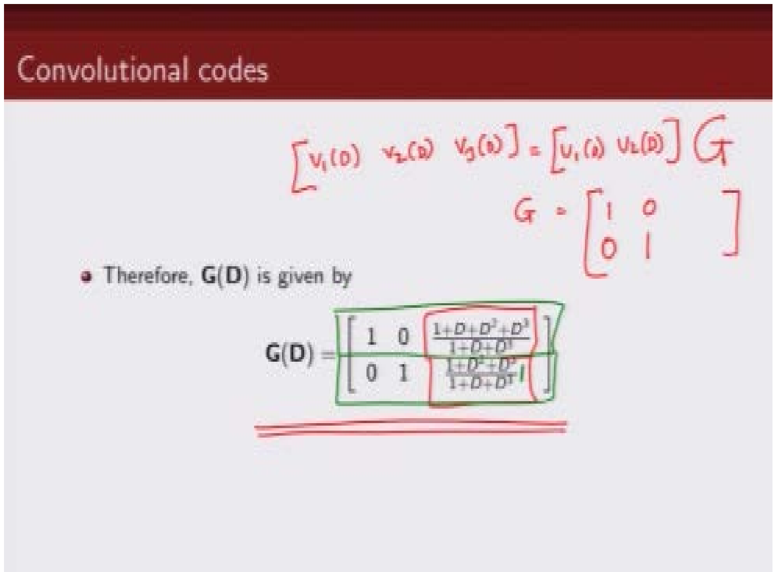
So, v_1D is just u_1D so that is this, v_2D is u_2D that is just this.

(Refer Slide Time: 25:33)

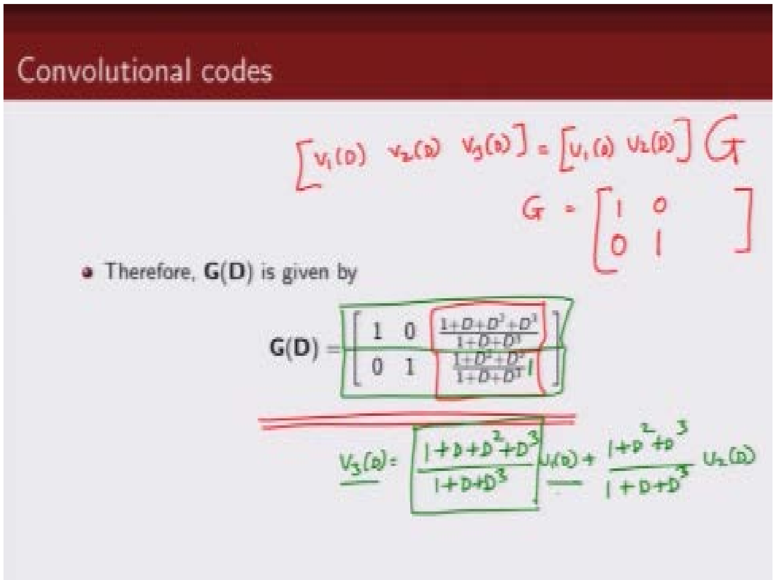


Now what is v_3D?

(Refer Slide Time: 25:42)

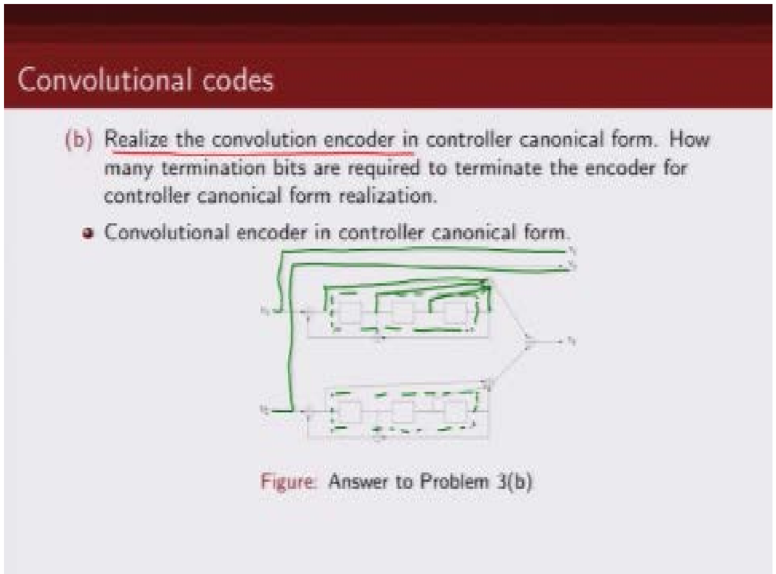


(Refer Slide Time: 25:48)



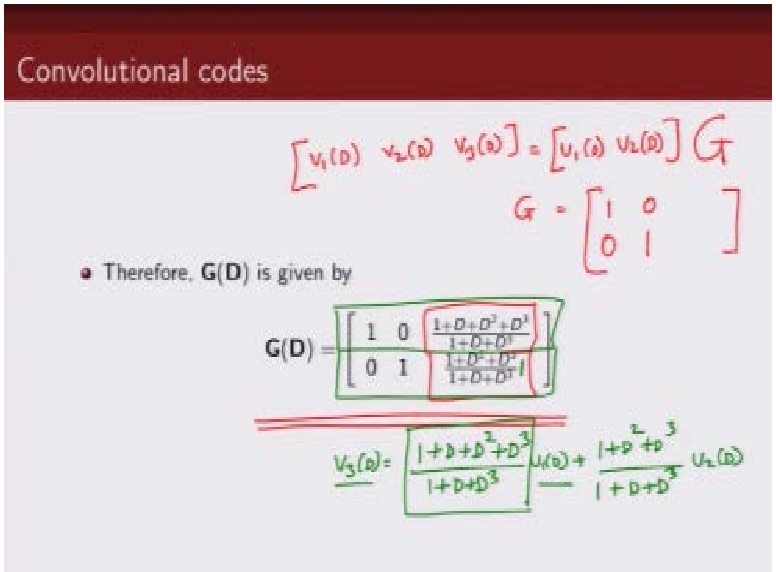
v_3 (D) is 1+D+D^2+D^3 /1+D+D^{3,} u_1 (D) +1+D^2+D^3 /1+D+D^3 u_2 (D) right? So relationship between v_3 and u_1D is given by this so let us implement this so numerator has 1+D+D^2+D^3.

(Refer Slide Time: 26:20)



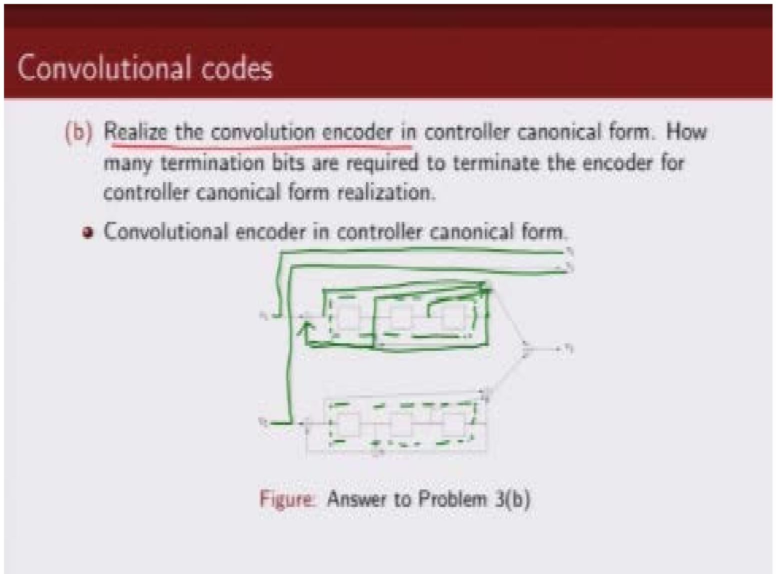
So you can see here this is my 1, this is my D, this is my D^2 and this is my D^3.

(Refer Slide Time: 26:32)



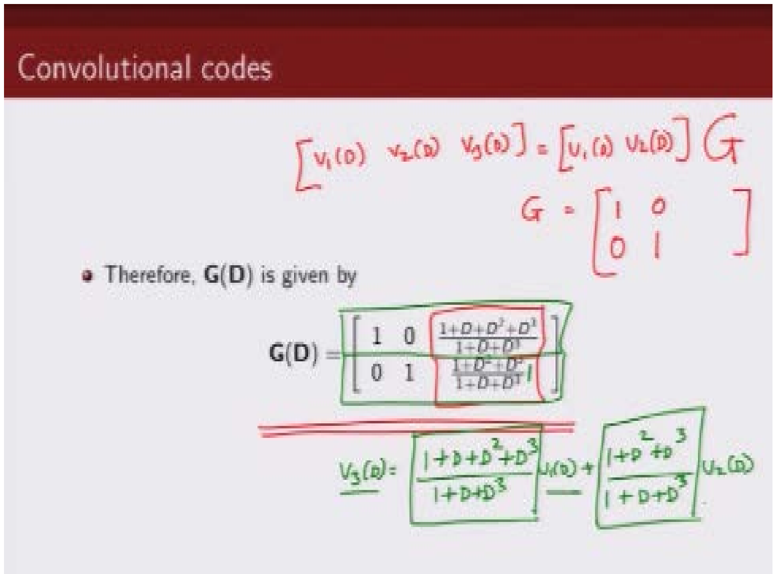
And similarly the denominator has $1+D+D^3$.

(Refer Slide Time: 26:38)



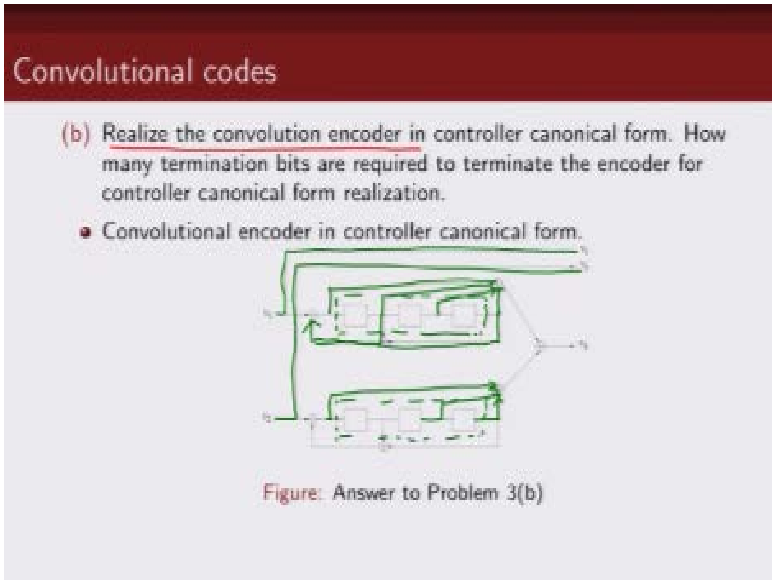
So the denominator this is the one term, this is the D term, and this is the D^3 term, so this part is implemented, next is this.

(Refer Slide Time: 26:38)



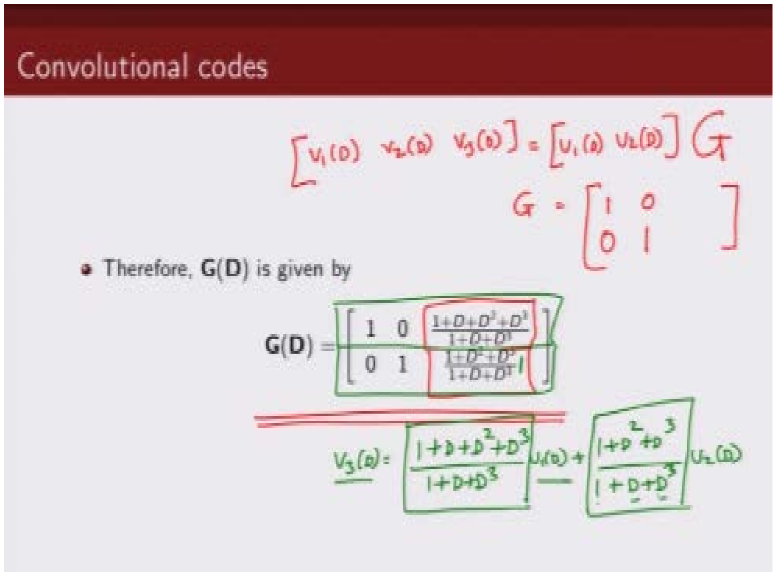
Following the same procedure we can find out the mapping between u_2 (D) and v_3 (D), the feed forward connections are 1, D^2 and D^3.

(Refer Slide Time: 27:09)



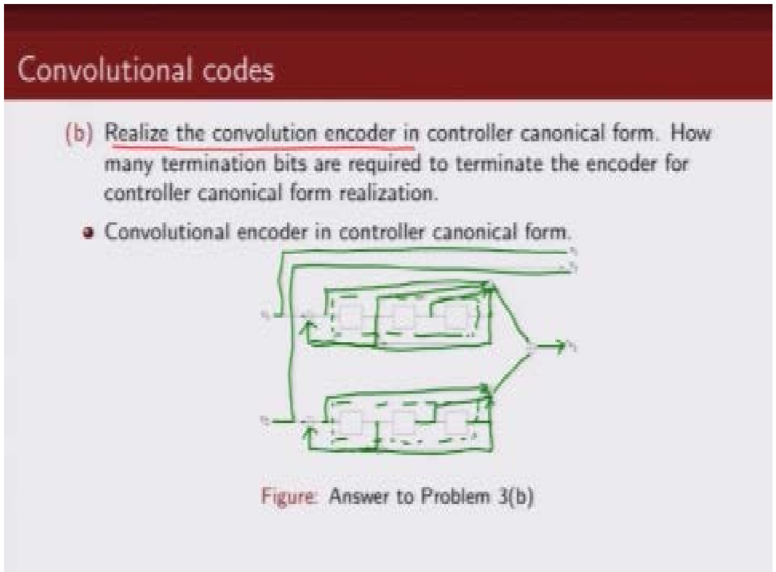
So then this is 1, this is D no connection, D^2 is this and D^3 is this.

(Refer Slide Time: 27:20)



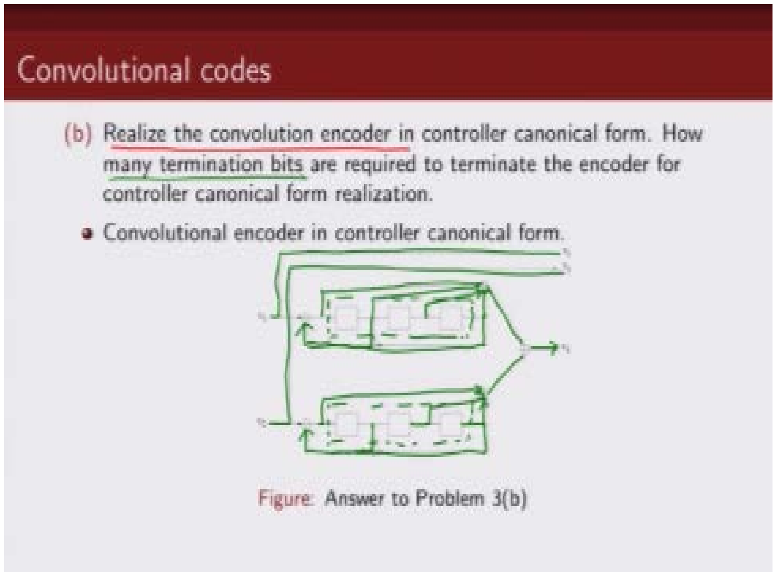
Similarly the feedback connections are 1, D and D^3.

(Refer Slide Time: 27:25)



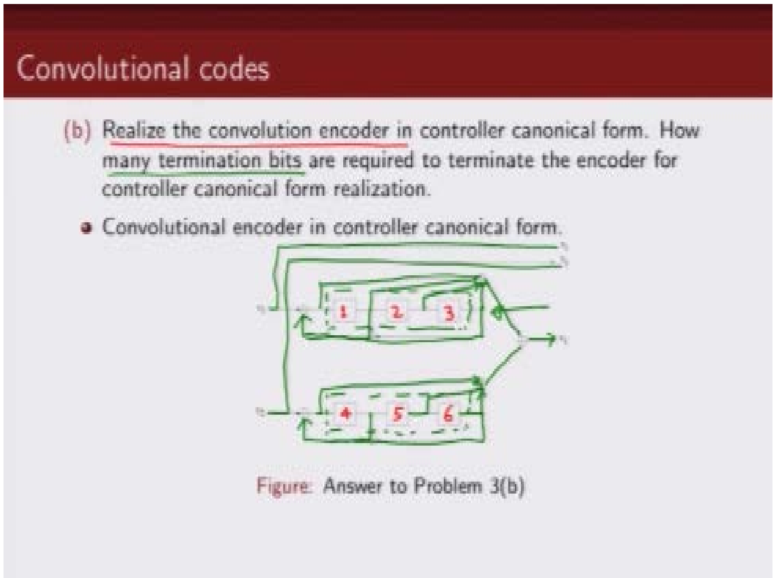
So feedback connections so this is the 1, this is D and this is D^3, and v_3 is a combination of these two so this is my v_3. So I hope this is clear how we can realize this encoder using controller canonical form realization.

(Refer Slide Time: 27:53)



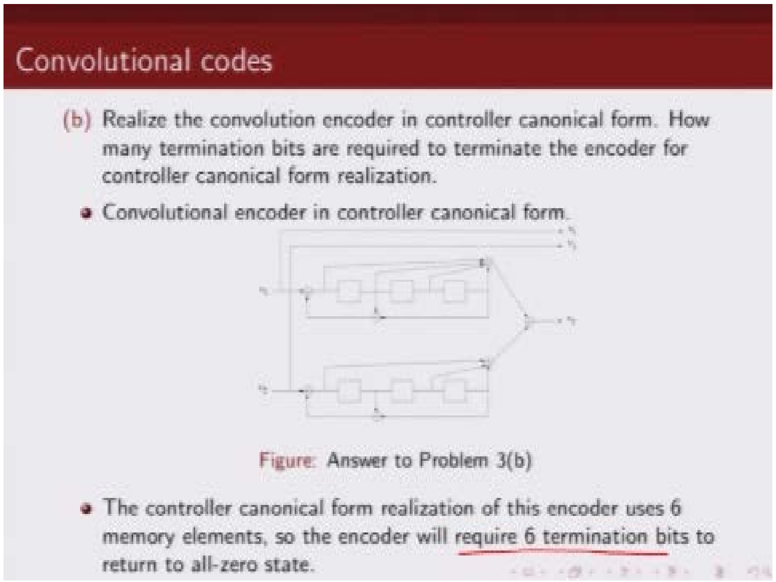
Now the next question is how many termination bits are required to bring this encoder back to all zero state, now what does termination means? Termination means we are bringing, bringing this encoder back to all zero state, so no matter what the state is if you want to bring them back.

(Refer Slide Time: 28:13)



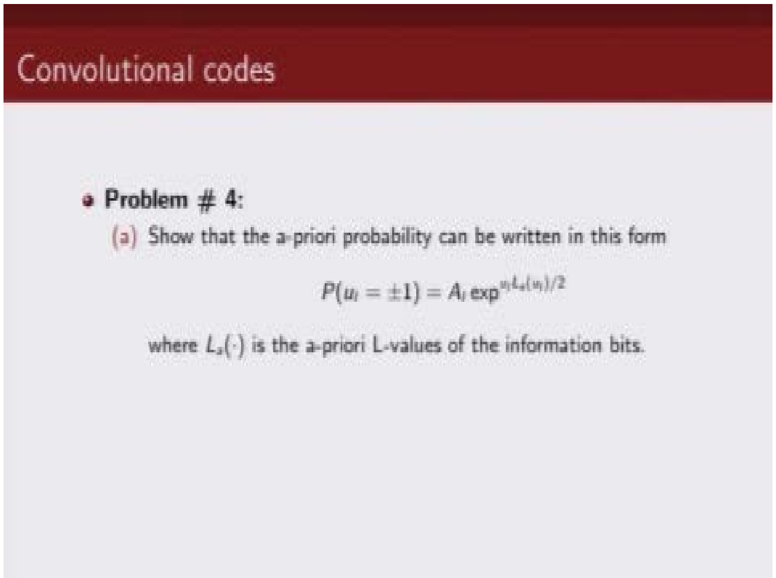
Into an all zero state the number of termination bits required is equal to how many memory elements we have, so in the controller canonical form realization to bring this shift register the first shift register if you want to bring it to all zero state we would require three bits because we have three memory elements here, 1, 2 and 3, similarly for this shift register we require additional three bits, so 4, 5 6. So total we require six termination bits, three to terminate this encoder and three to terminate this encoder.

(Refer Slide Time: 28:58)



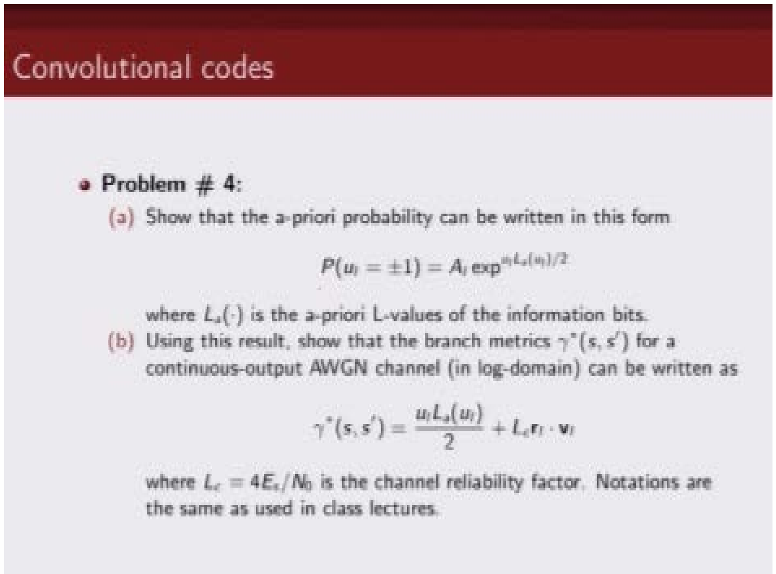
So we require six termination bits.

(Refer Slide Time: 29:01)

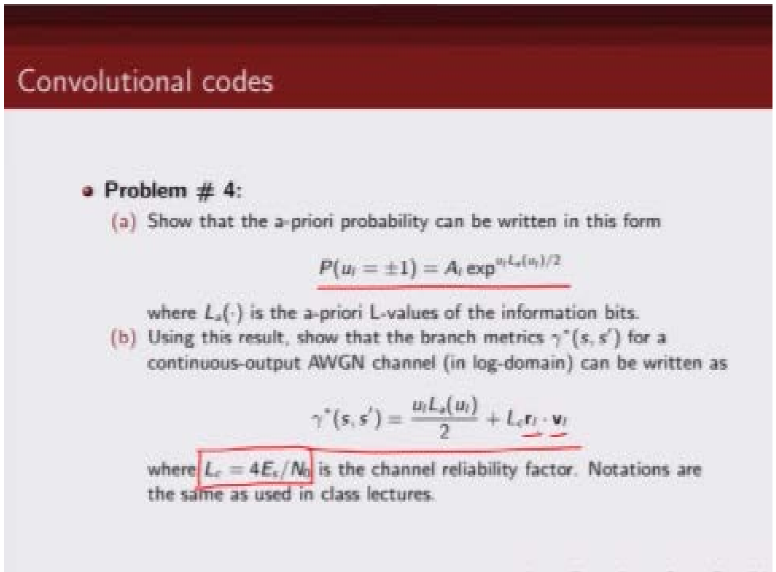


Finally let us come to the Bezier algorithm that we talked about so the first question is.

(Refer Slide Time: 29:10)

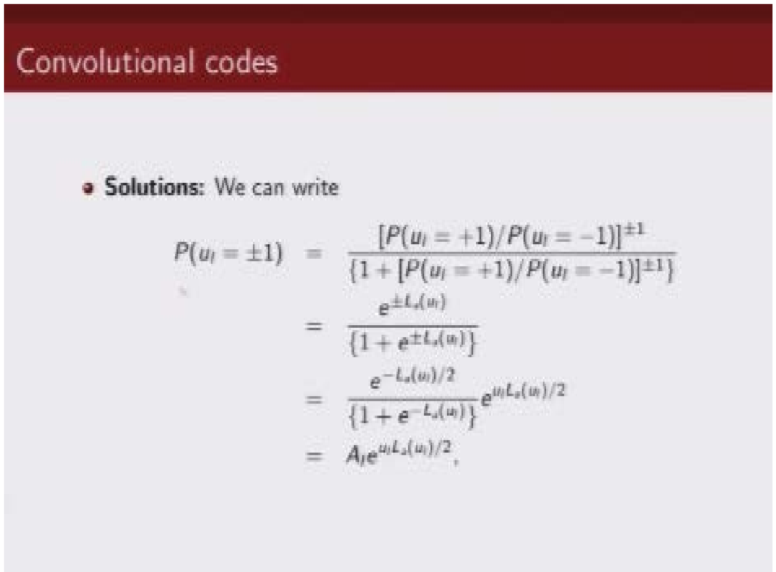


(Refer Slide Time: 29:12)

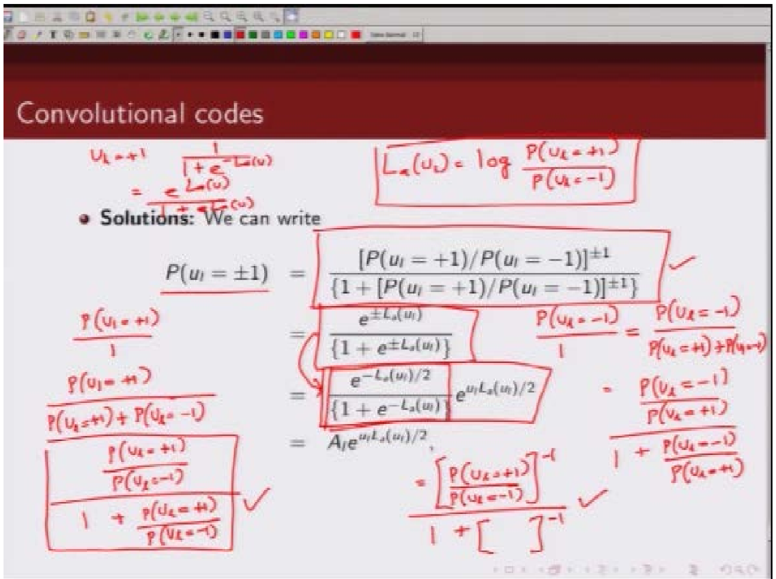


Can you write the a-priori probability in this particular form and also the branch metric in log domain can it be written in this particular form. Now u_l is my input $L_{a,}$ is the APP value for the a-priori inputs, L_c is a reliability factor which is given by $4E_s/ N_o$, other notations are same as which are used in the lecture, v is code word, r is a received sequence, so can we write these in terms like this? So let us look at it.

(Refer Slide Time: 29:53)



(Refer Slide Time: 29:54)



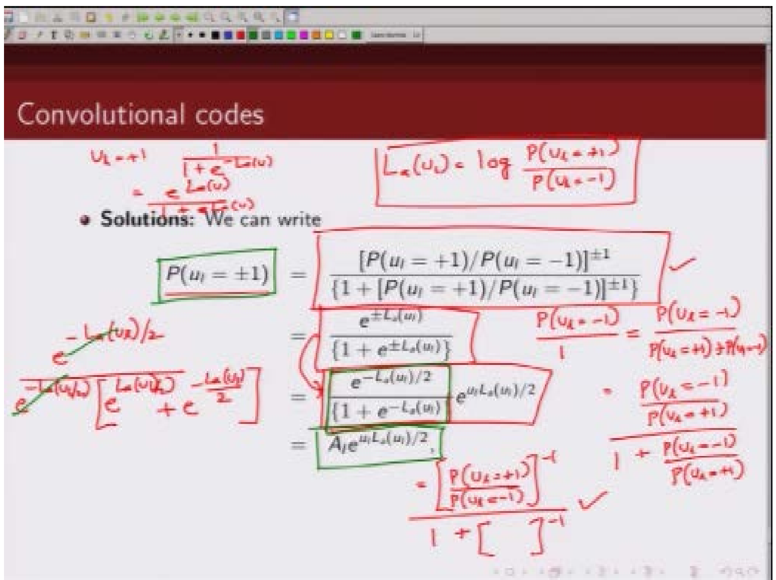
So what is the $P(u_l = \pm1)$ let us take like +1, let us say what is the $P(u_l = +1)$ now this can be written as this by 1 so I can write this $P(u_l = +1)/ P(u_l = +1) + P(u_l = -1)$ now write it this way, right? And if I divide by a $P(u_l = -1)$ then what I get is $P(u_l = +1)/ P(u_l = -1)/ 1+ P(u_l = +1)/P(u_l = -1)$. So this is what I will get of the form here, you can see here the form for when u_l is +1 I get $P(u_l = +1)$ I get in this particular form.

Now let us look at what is the $P(u_l = -1)$ again I can follow the same procedure I can write this as same as this by 1 or I can write $P(u_l = -1)/ P(u_l = +1) + P(u_l = -1)$ and I can divide this by $P(u_l = 1)$ so this will be $P(u_l = -1)/P(u_l = +1)/ 1+ P(u_l = -1)/P(u_l = +1)$, right? Now this I can also write as this is equal to $[P(u_l = +1)/P(u_l = -1)]^{-1} / 1+ [\ \ ]^{-1}$. So if I combine this and this what I get is the first step here okay? I can write by combining this and this, I will get this. Now if I write this ratio of probabilities in terms of L values so what is this L value of u_l this is log of $P(u_l)$ being +1/ $P(u_l)$ being -1. So this can be then written as $e^{La (u_1)}$, so if I do that if I plug this in first line what I get here is this term okay?

Now note I can further simplify this into this expression, you can see when u_l is +1 when ul is +1 what do we get, when u_l is +1 this is $e^{La (u_1)/2}$ and $e^{-La/2}$ so this is will be basically 1 so this will be one times $1+e^{-La(u)}$ which can be written as $e^{La(u)}/1+ e^{La(u)}$, this is precisely what I have written here. And if u_l is -1 this will be $e^{-La (u)/2}$ and $e^{-La/2}$ so this term will become in that case $e^{-La(u)}$

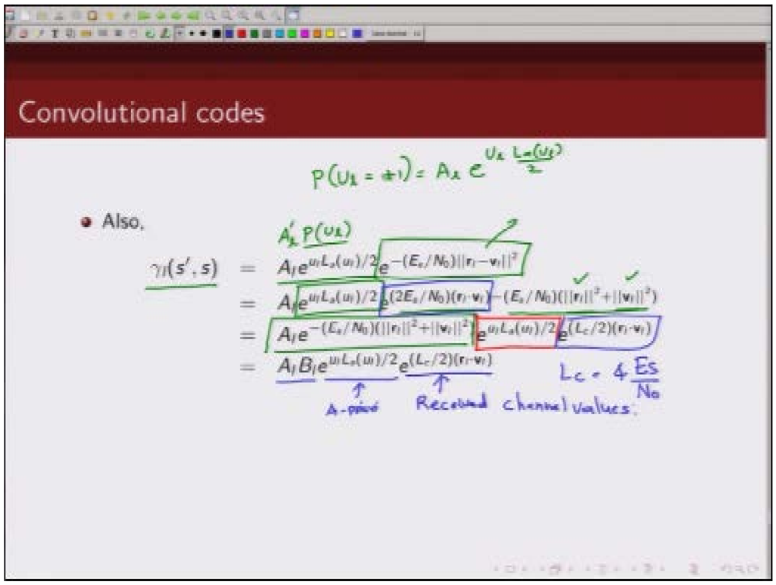
$1+e^{-La(u_1)}$. So this term can be written in terms of this right? And what is this term, what is this term, this I can simplify this term, let us make some space.

(Refer Slide Time: 34:41)



I can simplify this term as $e^{-La(u_1)/2}$ and I have $e^{-La(u_1)/2}$ this is $e^{La(u_1)/2} + e^{-La(u_1)/2}$ so what I am doing here is I am writing this particular term. So this I can write as this and this, so this cancel out and this is $e^{x} +e^{-6}$ this will be cause of x and that is the symmetric function. So it does not depend on sign of u_l, whether u_l is +1 or -1 does not depend on that. So I can then write this in terms of this expression.

(Refer Slide Time: 35:48)

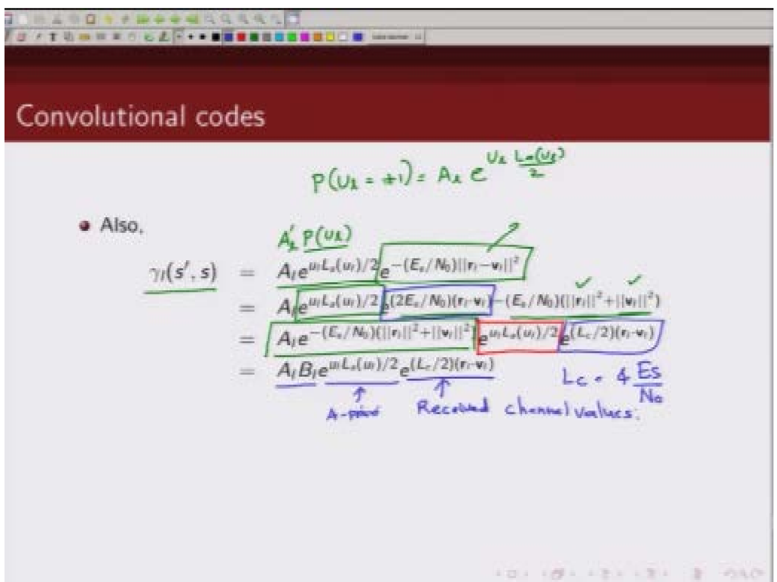


So will use the expression that we derived in the previous slide for a-priori value which was u_l being +1 or -1 as $A_l\ e^{u_l L_a(u_l)/2}$, we will use this expression to simplify the expression for branch matrix for our BCJR Algorithm. Now note if you recall we have written the expression for branch matrix as a-priori P (u_l) and then we had for AWGN channel we had this expression and of course there was some constant factor, we did not depend on u(n) right. So what we did just now was we derived that this a-priori probability can be written in this particular fashion right.

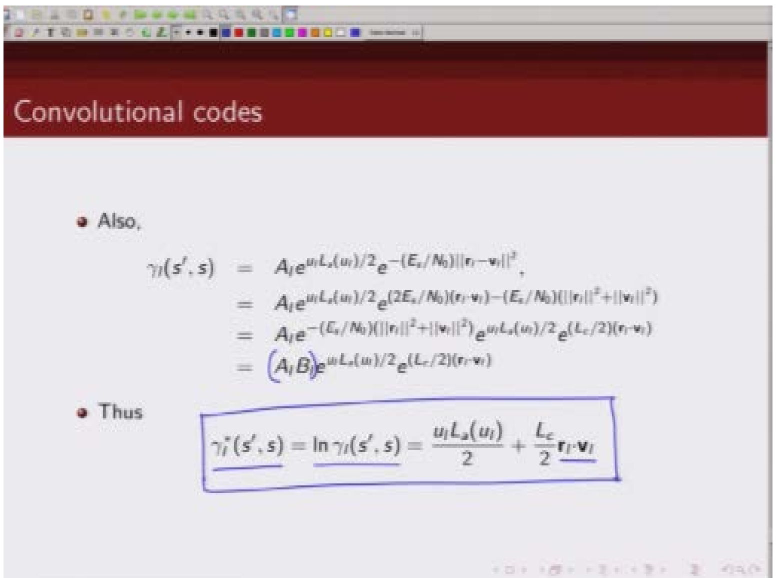
Now let us further simplify the expression for branch matrix. So this we can expand as $r^2 + v_l^2$ plus two times dot put it of $r_n\ v_l$. Now this does not depend on choice of v(l) and if v(l) is mapped to +1 and -1 v_l^2 will be 1 so this also will be a constant term, so this term would then not depend on choice of v(l). So what then we will be left with is, so this term we can just take out as some sort of constant which does not depend on choice of v(l) and what will be left is this term which we are writing here, which we are writing here and the next term that will be left is this term which we are writing here.

Please note Lc is four time Es/N0 so that is why we are writing it is, it is e $^{Lc/2}$ and dot product between r_n v_l. So then we can just simplify this expression as some constant terms multiplied by this a-priori, this a-priori term and this is a term which depends on received channel values. So this is a simplified expression for branch matrix computation for our BCJR algorithm.

(Refer Slide Time: 38:52)



(Refer Slide Time: 38:54)



If we are considering additive white cosine noise channel. Now if we consider branch matrix in, in the log domain then log of this term will be some sort of constant we just ignore it because this does not depend on choice of v(l), u(l), so then this will become u_l L value a-priori L value by 2 plus Lc/2, and dot put at between the receive sequence and the transmitted code word. So this will be then our simplified expression for branch matrix computation for BCJR algorithm over additive white cosine noise channel. Thank you.

Acknowledgement

Ministry of Human Resource & Development

Prof. Satyaki Roy

Co-ordinator, NPTEL IIT Kanpur

NPTEL Team

Sanjay Pal

Ashish Singh

Badal Pradhan

Tapobrata Das

Ram Chandra

Dilip Tripathi

Manoj Shrivastava

Padam Shukla

Sanjay Mishra

Shubham Rawat

Shikha Gupta

K. K. Mishra

Aradhana Singh

Sweta

Ashutosh Gairola

Dilip Katiyar

Sharwan

Hari Ram

Bhadra Rao

Puneet Kumar Bajpai

Lalty Dutta

Ajay Kanaujia

Shivendra Kumar Tiwari

an IIT Kanpur Production