

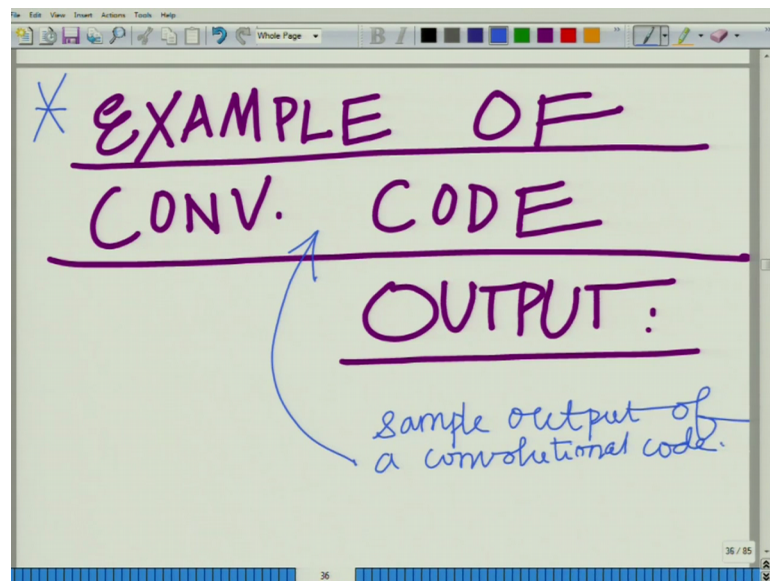
Principles of Communication Systems - Part II
Prof. Aditya K. Jagannatham
Department of Electrical Engineering
Indian Institute of Technology, Kanpur

Lecture - 50

Example of Convolutional Code Output, Convolution Operation for Code Generation

Hello. Welcome to another module in this massive open online course. We are looking at convolutional codes correct which we said our class of error control codes to recover or correct errors that occur over the channel in a communication system, especially a digital communication system. Today let us look at an example of the code generated by this convolutional codes.

(Refer Slide Time: 00:38)



So, an example. So, let us look at an example of the output example of convolutional code. What we want to do is, we want to look at the output as the sample output of a convolution code. So, you want to look at the sample, we want to look at the sample output of a convolutional code.

(Refer Slide Time: 01:31)

sample output of a convolutional code.

consider the bit sequence:

1 0 1 1 0 0 1 0 1 1
 $x(0)$ $x(1)$ $x(2)$ $x(3)$ $x(4)$ $x(5)$ $x(6)$ $x(7)$ $x(8)$ $x(9)$

$C_0(j) = x(j) + x(j-2)$

Binary Field Addition

So, we consider the information bit sequence 1 0 1 1 0 0 1. Let me just write it. So, this is 1 0 1 1 0 0 q 0 1 q q 0 0 1 1 0 0 1 0 1 1. So, this is obviously $x(0)$ is starting with $x(0)$, $x(1)$, $x(2)$, $x(3)$, $x(4)$, $x(5)$, $x(6)$, $x(7)$, $x(8)$ followed by $x(9)$ and what we are going to do is, remember if you start to without definition C_0 of j equals x of j plus x of j minus 2, this is binary field addition remember which is given by xor. We have seen that in the previous module. This is binary field arithmetic. So, this is binary field addition.

(Refer Slide Time: 03:13)

1 0 1 1 0 0 1 0 1 1
 $x(0)$ $x(1)$ $x(2)$ $x(3)$ $x(4)$ $x(5)$ $x(6)$ $x(7)$ $x(8)$ $x(9)$

$C_0(j) = x(j) + x(j-2)$

Binary Field Addition

$C_0(0) = x(0) + x(-2)$
 $= 1 + 0 = 1$
 $x(-1) = 0$
 $x(-2) = 0$

Now, we will realize that to compute C_0 of 0, I need x of 0 plus x of j minus 2, that is x of minus 2. So, I will need this quantity. So, I need to start somewhere. So, what I am going to do is, I am going to set both x of minus 1 and x of minus 2 to 0. Assume that the system starts from 0 0, all right. We are going to see the interpretation of this later, all right. So, what we are doing is, we are setting x minus 1 to 0 as well as x minus 2 to 0. So, x minus 1 equal to 0 x minus 2 equal to 0. So, this implies C naught of 0 x naught of x of 0 is 1 plus x of minus 2 is 0. So, 1 plus 0 is 1.

(Refer Slide Time: 04:04)

The image shows a digital whiteboard with handwritten mathematical derivations. The top part shows the calculation of $C_0(0)$ as the sum of $x(0)$ and $x(-2)$. It notes that at time $j=0$, the code bit 0 is 1 and $x(-1)$ is 0. The bottom part shows the calculation of $C_1(0)$ as the sum of $x(0)$, $x(-1)$, and $x(-2)$. It notes that at time $j=0$, the code bit 1 is 1 and $x(-2)$ is 0. The final result for $C_1(0)$ is 1.

$$C_0(0) = x(0) + x(-2)$$

$$\begin{aligned} \text{code bit 0 at time } j=0 & \rightarrow x(-1) = 0 \\ \text{code bit 1 at time } j=0 & \rightarrow x(-2) = 0 \end{aligned}$$

$$C_1(0) = x(0) + x(-1) + x(-2)$$

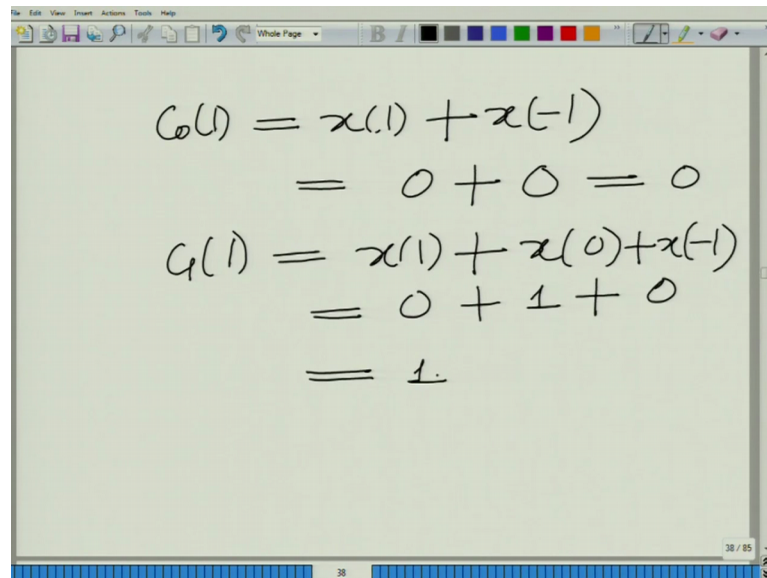
$$= 1 + 0 + 0$$

$$= 1$$

Similarly, see this remember this is code bit 0 at time j equal to 0. So, this is code bit 0 at time j equal to 0, C_1 of 0. This is x of 0 plus x of j minus 1, x of minus 1 plus x of j minus 2 0 minus 2, that is x of minus 2. This is 1 plus 0 plus 0 which is equal to 1. This is code of bit 1 at time j equal to 0.

So, we have C_0 of 0 equal to 1, C_1 of 0 that is code bit 1 at time 0 is also equal to 1.

(Refer Slide Time: 05:06)

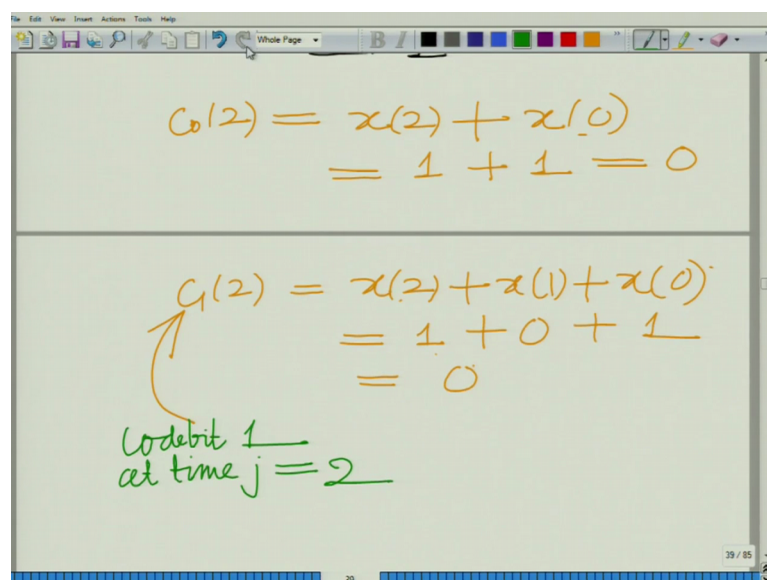


The image shows a digital whiteboard with handwritten calculations. The first calculation is $C_0(1) = x(1) + x(-1)$, which simplifies to $0 + 0 = 0$. The second calculation is $C_1(1) = x(1) + x(0) + x(-1)$, which simplifies to $0 + 1 + 0 = 1$. The whiteboard interface includes a toolbar at the top and a status bar at the bottom showing '38 / 85'.

$$C_0(1) = x(1) + x(-1)$$
$$= 0 + 0 = 0$$
$$C_1(1) = x(1) + x(0) + x(-1)$$
$$= 0 + 1 + 0$$
$$= 1$$

Now, let us look at again $C_0 1$ that will be x of j equal to 1 plus x of j minus 2 x of minus 1 x of 1 is 0 plus x of minus 1 is 0. So, this is equal to 0 C_1 of 1 is similarly x of 1 j equal to 1 plus x of j minus 1 0 plus x of j minus 2, that is minus 1. So, this will be x of 1 is 0 plus x of 0 is 1 plus x of minus 1 is 0. So, this will be equal to 1. Similarly, you can go on generating the subsequent code bits at each time instead, ok.

(Refer Slide Time: 05:52)



The image shows a digital whiteboard with handwritten calculations. The first calculation is $C_0(2) = x(2) + x(0)$, which simplifies to $1 + 1 = 0$. The second calculation is $C_1(2) = x(2) + x(1) + x(0)$, which simplifies to $1 + 0 + 1 = 0$. A green arrow points from the text 'Codebit 1 at time j=2' to the $C_1(2)$ calculation. The whiteboard interface includes a toolbar at the top and a status bar at the bottom showing '39 / 85'.

$$C_0(2) = x(2) + x(0)$$
$$= 1 + 1 = 0$$
$$C_1(2) = x(2) + x(1) + x(0)$$
$$= 1 + 0 + 1$$
$$= 0$$

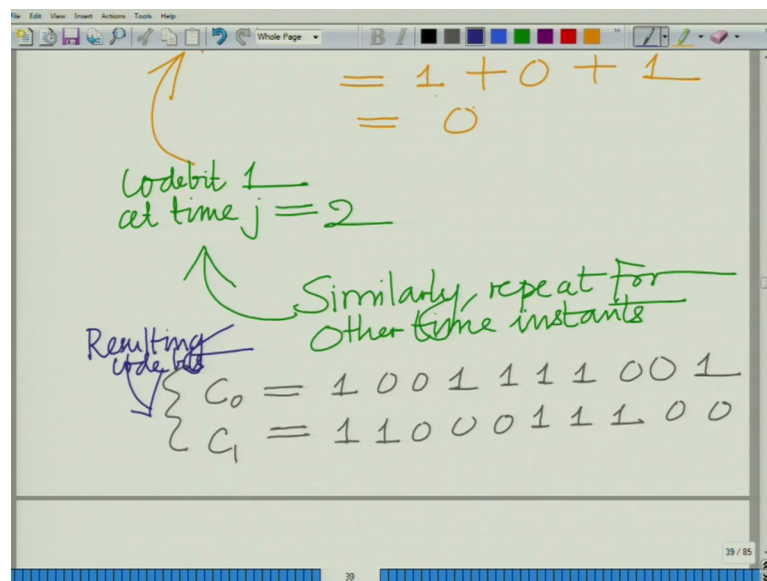
Codebit 1
at time $j=2$

So, for instance C_0 of 2 that is generate one more, this will be x of 2 plus x of 2 minus 2, that is x of 0. This will be 1 plus 1 and binary field addition, this is 1 plus 1 is 0.

Remember we are doing xor and C_1 of 2 is x of 2 plus x of 2 minus 1×1 plus x of 2 minus 2, that is x of 0 which is equal to x of 2 is 1 plus x of 1 is 0 plus x of 0 is 1. Again going by binary field ratio, 1 plus 0 is 1 , 1 plus 1 is 0 .

So, this is 0 . So, C_0 is 1 , so C_1 I am sorry C_1 that is code bit 1 at time instant 2 . So, remember this is code bit 1 at time j is equal to 2 and therefore, now you can construct.

(Refer Slide Time: 07:15)



So, similarly repeat for other time instants and what we will get is C_0 and C_1 . The sequence C_0 and C_1 we can see C_0 is $1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1$ and C_1 is basically $1\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 0$. So, you can see that the resulting code sequence is this. So, this is the resulting and one thing you can see is that we have nine inputs of total number. So, we have ten input informed that is counting from 0 . So, we have number of information bits.

(Refer Slide Time: 08:42)

Resulting code bits

Similarly, repeat for other time instants

$$C_0 = 1001111001$$

$$C_1 = 1100011100$$

10 bits

10 bits

information bits = 10

code bits = 20

$$\text{Rate} = \frac{10}{20} = \frac{1}{2}$$

So, if you look at this number of information bits equals 10 and number of code bits that are output is 10 plus 10 that C_0 10 bits 4 C_0 correct 10 bits and 10 bits for C_1 that is equal to 20. Therefore, now again the rate equals number of information bits by number coded bits. Remember information bits for code bits, this is equal half and this is something that we have seen in the previous module also. The rate basically is the number of code bits is generated for each. The rate is the number of information bits divided by the number of code bits that is basically the inverse of the number of code bits generated per each information bit here. Here you can see in the simple example that the rate is half and of course, this procedure is likely laborious because you have to look at x_j and we have to compute the code bit C_0 of j and C_1 of j . At each time instant we are going to see a much simpler and more elegant procedure to generate the coded bits as we proceed the various models that is based on the state diagram.

Once we construct a state diagram and it trellis for this code, we will see a much simpler and much faster way to generate this code bits corresponding to a given input information bit sequence. Now, let us look at the interpretation why is this known as the convolutional code.

(Refer Slide Time: 10:36)

codebits = 20

Rate = $\frac{10}{20} = \frac{1}{2}$

Convolutional Code:

Why convolutional code?

$$C_0(j) = x(j) + x(j-2)$$

$$= 1 \cdot x(j) + 0 \cdot x(j-1)$$

So, if you look at this notion of a convolutional code, we would like to ask the question why is this convolutional code and to answer that you go back and look at the generation of the code bit.

(Refer Slide Time: 11:38)

$$= 1 \cdot x(j) + 0 \cdot x(j-1)$$

\uparrow $h_0(0)$ \uparrow $h_0(1)$

$$+ 1 \cdot x(j-2)$$

\uparrow $h_0(2)$

$$= \sum_{k=0}^2 h_0(k) x(j-k)$$

convolution

$$h_0(0) = 1 \quad h_0(1) = 0$$

$$h_0(2) = 1$$

Filter h for convolution

So, you have $C_0(j)$ equals $x(j)$ plus $x(j-2)$ which I can write as $C_0(j)$ equals 1 times $x(j)$ plus 0 times $x(j-1)$ plus 1 times $x(j-2)$ which is equal to summation k equal to 0 to 2 $h_0(k)$ rather $h_0(k)$ times $x(j-k)$.

Now, if you see $h_0 = 1$, $h_1 = 0$. So, this is basically your h_0 , this is basically your h_1 and this is basically your h_2 . So, h_0 of 1 is equal to 1 and h_0 of 2 is equal to 0. So, I am writing this as summation k equal to 0 h_0 of k times x_j minus k and this you can see is nothing, but a convolutional operator. This is nothing, but a convolution correct. In fact, this is the filter for the convolutional. So, you can think of the input, the information bit sequence as being input to this filter h_0 which with filter coefficients h_0 equal to 1, h_1 equal to 0, h_2 equal to 1, then the output generate the output coded bits which are formed, which are generated by the convolution of this filter with the input information sequence.

That generates the output coded bits C naught output coded bit bits stream C naught, all right. So, it generate as a convolution of the information, the input information bit sequence with the appropriate filter. Hence, it is one as a convolutional code. So, the convolutional operation forms a basis for the generation of this code. You can similarly look at C_1 of j .

(Refer Slide Time: 14:05)

The image shows a digital whiteboard with a toolbar at the top. The handwritten text on the whiteboard is as follows:

$$\begin{aligned}
 C_1(j) &= x(j) + x(j-1) \\
 &\quad + x(j-2) \\
 &= \overset{h_1(0)}{1} \cdot x(j) + \overset{h_1(1)}{1} \cdot x(j-1) \\
 &\quad + \overset{h_1(2)}{1} \cdot x(j-2) \\
 &= \sum_{k=0}^2 h_1(k) x(j-k)
 \end{aligned}$$

The whiteboard interface includes a toolbar with various drawing tools and a status bar at the bottom showing '42 / 85'.

So, no surprise there if you look at C_1 of j . For instance, C_1 of j is equal to x of j plus x of j minus 1 plus x of j minus 2. This is 1 times x of j plus 1 times x of j minus 1 plus 1 times x of j minus 2 which is equal to summation k equal to 0, again k equal to 0 to 2. This time h_1 of k times x of j minus k , this is h_1 of 0, h_1 of 1, h_1 of 2.

(Refer Slide Time: 15:06)

The image shows a handwritten derivation on a digital whiteboard. At the top, the equation
$$= \sum_{k=0}^2 h_1(k) x(j-k)$$
 is written. Above the summation, the term $1 \cdot x(j-2)$ is written in orange, with a bracket indicating it corresponds to $h_1(2)$. Below the summation, the word "convolution" is written. Underneath, the filter coefficients are listed: $h_1(0) = h_1(1) = h_1(2) = 1$. This is followed by the text "Filter for codebit stream 1" underlined.

So, we have basically in this filter h_1 of 0 equals h_1 of 1 equals h_1 of 2 is equal to 1 and this is you can see this is the convolution operation and this is the filter for the code stream, filter for code bit stream coded bit stream 1. So, the code is generated by convolution of the input information bit stream with the suitable filter. This is known as a convolutional code, ok.

(Refer Slide Time: 15:51)

The image shows handwritten text on a digital whiteboard. At the top, the text "Filter for codebit stream 1" is underlined. Below it, the sentence "Code is generated from a convolution operation." is underlined. An arrow points from this sentence to the text "input information Bit Stream with suitable Filter". Another arrow points from the same sentence to the text "Hence, termed as a convolutional code.".

So, let us note that code is generated from a convolution operator. Convolution operation of the input information bit stream with the suitable filter of course the suitable filter is what determines the convolutional code.

Hence, this is termed as a convolutional code. So, what we have seen in this module is, we have seen a simple example for generation of the bits coded, bits stream of a convolutional code given an input bit stream and we have also seen the motivation or the reason for why this is known as, this code is known a convolutional code because it is generated by a convolution operation of an underlining filter with the input information bit stream, all right.

So, we will stop here and continue in the subsequent modules.

Thank you.