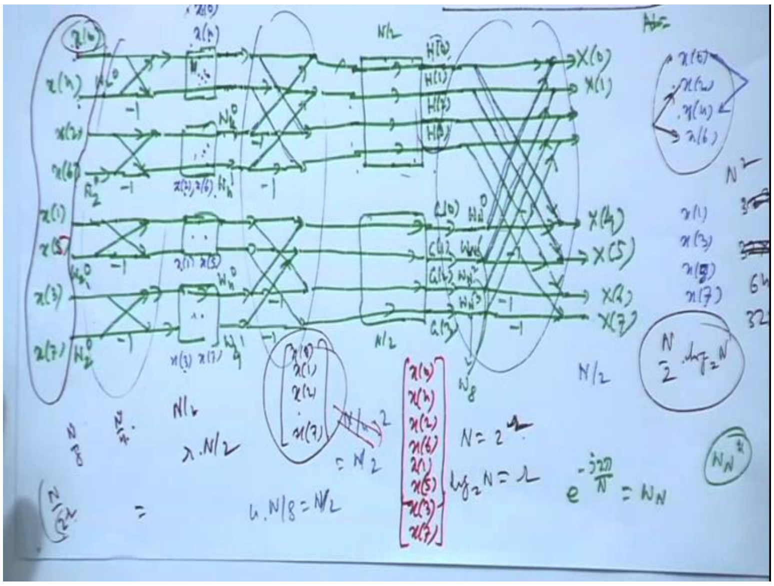
Discrete Time Signal Processing

Prof. Mrityunjoy Chakraborty
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

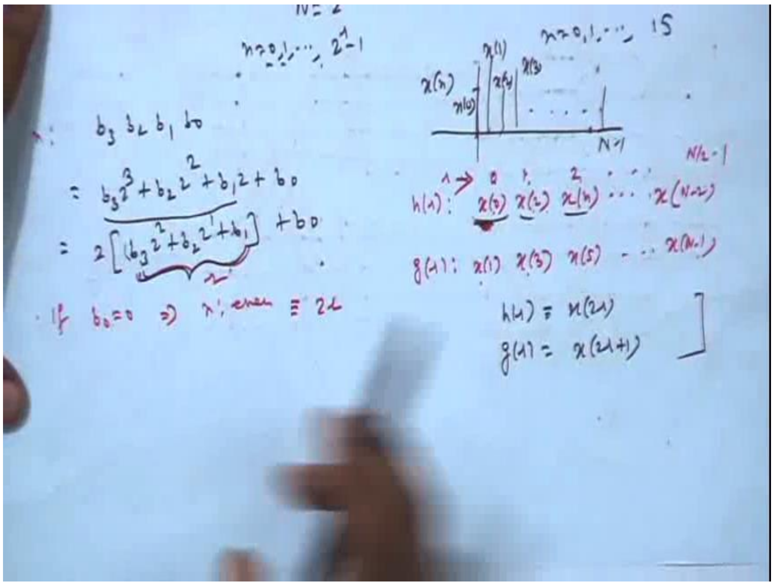
Lecture – 40
Bit Reversal and FFT

(Refer Slide Time: 00:01)



Last class we got c r FFT, I remember we are getting I mean original vector data, vector was x 0, x 1, x 2 up to x 7, but this time data vector got thoroughly shuffled. Now, it is x 0, x 4, x 2, x 6, x 1, x 5, x 3, x 7. So, it is a new orders originally it was x 0, x 1, x 2 up to x 7. Now, it is x 0, x 4, x 2, x 6, dot, dot, dot, as I told. So, for a big FFT say 512 and all that you cannot go back and find out the order right, but this order in these elements have to be elements of the original vector have to be shuffled that can be obtained easily from some observation that suppose N is as I told you some power of r like say N suppose is 16 that is 2 to the power 4.

(Refer Slide Time: 01:16)



So, I will have smaller equal to 0, 1 dot, dot, dot N minus 1 that is 2 to the power r minus 1 in this case smaller n equal to 0, 1 dot, dot, dot up to 15. So, I have originally x n sequence up to N minus 1. So, x of 0 x of 1 x of 2 x of 3 dot, dot, dot, dot, now you remember what I did I took the even point sequence that is I formed a sequence like x 0 and I put an underline indicate this is the position of the origin then x 2 x anyway it is not required because if I had a sequence I am starting from x 0 only say x 0, x 2, x 4, dot, dot, dot and it will go up to because n is even.

So, N minus 1 is odd. So, N minus 2 that is the largest even number. So, it will go up to x. So, in this case N minus 2 in general that is 1 sequence I called it h r. So, h r and other 1 I took g r which was x 1, x 3 dot, dot, dot this order last order h r is x 2 r that is h 0 0, this is the 0 th guy h 0, this is h 1, this is h 2, dot, dot, dot and how many I have got N by 2 elements here. So, this is 0 th first second up to N by 2 minus 1 again 0 th first, second up to N by 2 minus 1. So, r th guy h r is original x 2 r and g r is original x not just 2 r, but 2 r plus 1 like if r is 0 it is x 1, g 0 is x 1, g 1 is x 3, dot, dot, dot this is the situation.

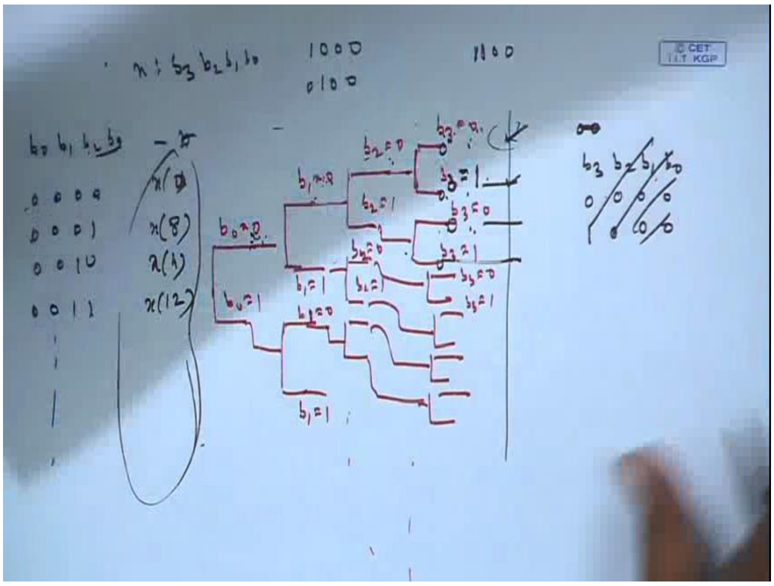
There I wanted to find the N by 2 point DFT of this N by 2 point DFT of this and then total DFT was obtained in terms of them I got a Butterfly diagram, then again the DFT of h r we prove that this particular sequence you consider again divided it into its own even part and odd part. Similarly, here and did the same thing and then their structure was developed that we have done.

Now, what are these indices in terms of r this is 0, 1, 2, dot, dot, dot N by 2 minus 1 in terms of r same here 0, 1 now. So, at 0 th x 0 comes at first, x 2 comes at 2, x 4 comes at r, x 2 are comes dot, dot, dot. Now, you see this indices 0, 1 to 2 to the power r minus 1 like 0, 1 up to 15 these all these numbers can be represented by how many bits; 4 bits like r bits here. So, suppose I am taking the example of this, 4 bits. So, 4 bits or say may be b 3, b 2, b 1, b 0; 0 0 0 0 that is 0 0 0 0 1 that is 1 index 1 0 0 1 0 that is index 2 up to 1 1 1 1 that is index 15.

Now, you see this is equal to b 3 into 2 to the power 3. I am doing for this case where you can easily generalize to the general case 2 to the power 2, if I take out this part, this is divisible by 2 b 3, 2 to the power 2, b 2, 2 to the power 1, b 1 plus b 0. If b 0 equal to 0 then this whole sum is divisible by 2 because, it is 2 into this so that means, even if I call this n, n is even if its 4 bit binary representation b 0 is 0 n is even means n is equal to 2 r; that means, this is r alright.

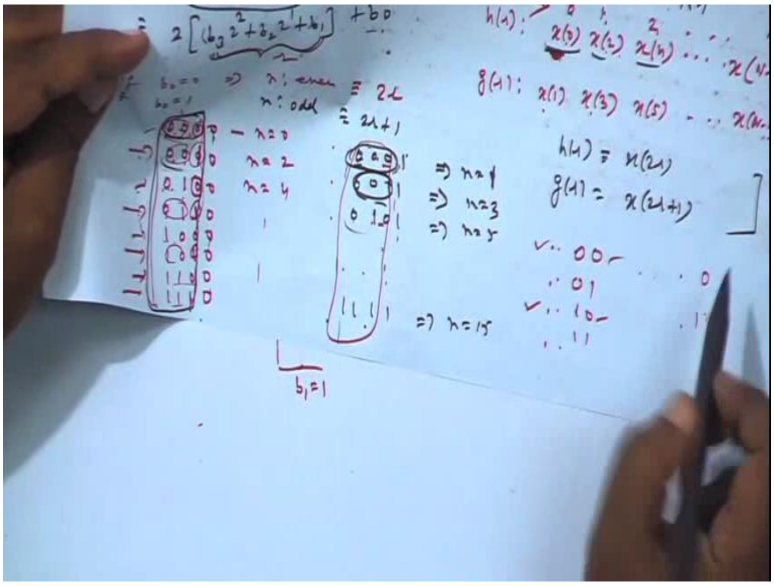
So, what I will do to get these elements x 0, x 2, x 4 I will take the index n and write it in the binary form I will pick up only those cases where b 0 is 0 because as long as b 0 is 0 I get even and then this part this is your r see b 0 is 0 total value is n and n is going to be 2 r because it is even. So, this is r, this r will take these 3 bits; b 1 plus b 2 2 to the power 1 plus b 3 2 to the power 2. So, 3 bits is binary representation right. So, it will take 8 possibilities and that is true because all will take the 8 possibilities 0, 1, 2 up to 7 total is 15, 16; 8 here 8 there.

(Refer Slide Time: 07:00)



So, what I do I have an tree like this b 0 equal to 0 upstairs, b 0 equal to 1 downstairs, b 0 equal to 0 means I will consider 0 0 0 0 then 0 0 naught 1 0 0 1 0 because this I want to hold permanently at 0 then 0 1 0 0 then 0 1 1 0.

(Refer Slide Time: 07:09)



Then 1 0 0 0, then 1 0 1 0 1 1 0 0 1 1 1 0 alright 1 2 3 4 5 6 7 8, this was actually n equal to 0, n equal to 2, n equal to 4, dot, dot. So, I get all those even indices. So, I first what I do I take b 0 and 0, b 0 equal to 0 case. So, b 0 equal to 0 case only this 8 fellows come and then I order them I write them in this order here upstairs, these fellows and downstairs if b 0 is 1, if b 0 is 1 n is odd it is 2 or plus 1. So, again this is a r 2 or plus 1. So, this is r and r will take total 8 values when b 3, b 2, b 1 are 0 it will be 0, when 0 0 1 it will be 1,when is 0 1 0 it will be 2 like that because you see 2 to the power 2, 2 to the power 1 like that alright, but here I will write them in this way 0 0 0 1.

I will hold, this is called this will correspond to n equal to 0 because 2 r plus 1 well this corresponds to no this corresponds to n equal to 1, r equal to 0 alright, n equal to 1 0 0 0 1 then 0 0 1 1, n equal to 3 then 0 1 0 1, n equal to 5 dot, dot, dot, dot, up to 1 1 1 1 n equal to 15. Here, last bit is 1 that is why it is odd. So, b 0 is 1 remaining part b 2, b 1, b 2, b 3, if you take the 2 out. So, these 3 digits like these 3 digits here they will take 8 values and their ascending order 0, 1, 2, 3 like that. So, I hold them after I see here I took b 0 equal to 0 and then I put this array under b equal to 0 this array upstairs here I i am not writing it, but suppose if I order I write all these values 3 values under b 0 equal to 0 case all these 3 values, here all these 3 bits 0 0 0 0 0 and all of them are pushed upstairs 0 0 0 0 0 1 0 1 0 that is these 3.

Similarly, when b 1 equal to b 0 equal to 1, I have got 2 r plus 1 that is n, n is odd it is 2 r plus 1. So, again r will take these 3 bits only the decimal value because 2 r plus 1. So, under b 0 equal to 1 case again I write the same order 0 0 0 0 0 1 0 1 0 dot, dot, dot on this side, but under b 0 equal to 1 case now, what I do I am just considering these 3. So, this will be r equal to 0 r equal to 1 r equal to 2 r equal to 3 dot, dot, dot. So, it is now 8 point sequence in my next I told you I will consider the even point component of this indices. So, 0 this is the r equal to 0 case that is why this fellow is 0 then equal to r equal to 2 case that is why again this fellow is 0 because, now I am pursuing 3 bits only under b 0 equal to 0, I got the even point.

So, this is my r equal to 0, r equal to 1, r equal to 2 like that or I am now considering its own even points are this guy 0 0 0, then next is 0 1 0, then next is 1 0 0, then next is 1 1 0. So, wherever I have got 0 at the l s b I hold it or, but what is that bit that is b 1. So, here again I go up, b 1 equal to 0 case and here b 1 equal to 1 case under b 1 equal to 0 under b 1 equal to 0 I have got this guy 0 0 then 0 1 that is 2 bits 0 0 0 1 1 0 1 1 2 bits. This will be new r, r equal to 0, r equal to 1, r equal to 2, r equal to 3. So, these 4 are pushed up this 4 cases with b 1 0 with b 0 0. So, b 0 0 then b 1 0 and remaining 2 bits 0 0 0 1 1 0 1 1 they are pushed up and similarly, I take the odd case. So, this 1, this 1, this 1 this 1 which have 1 at the l s b, then again odd will be this much 0 0 0 1 1 0 1 1 same thing or we will take those 4 values under b 1 equal to 1 case. So, under b 1 equal to 1 case those 4 will come.

Similarly, here under b 0 equal to 1, case also if b 0 is 1, but I still have this array only here because b 1 is 1, this is 1, 1, but this is same as this. So, here 0 0 0 0 0 1 0 1 0 up to 1 1 1 that is r is equal to 0 r equal to 1 r equal to 2 up to r equal to 7 they were here. So, those are those indices again I divide them into even point and odd point how I look at this guy this l s b. Now, whenever that is 0 I move up that is under b 0 1 b 1 0 and whenever that is 1 I move below. So, here again I give 4 indices that is 0 0 0 1 1 0 1 1 and here also by the same way, it will be same then I have got 4 point sequences this again I break up into 2 part 4 point sequence means 0 0 0 1 1 0 1 1 that is 0 r equal to 0 r equal to 1 r equal to 2 r equal to 3.

So, this and this they are the even points this I push up. So, for them this fellow is 0 this fellow is now b 2. So, I go up under equal to 0, I have got 0 and 1 it has 2 values 0 and 1, 0 and 1, 1 is 0 1 1. So, r equal to 0, r equal to 1, this is my event point, this is odd point. Now, here I go for b 3 equal to 0 b 3 equal because this is b, this is 3 b 3. So, b 3 0, b 3 0, b 3 1, b 3 0, b 3 1 you can easily understand that when b 2 is 1 you mean then I will have the same thing by the same logic. So, under b 2 1 again I will have b 3 0 up b 3 1 down, I am not completing this free, there is no space again b 2 equal to 0, b 2 equal to 1, dot, dot, dot I am not completing it, again if it is b 3 0, b 3 1 dot, dot, dot.

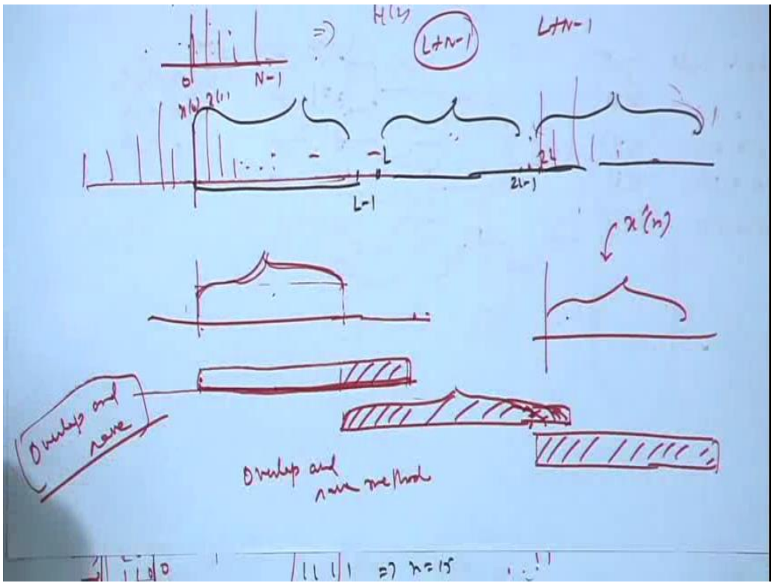
So, who comes on the top of the scale 0 0 0 0, but what is that number actually because that number for that in terms of N, b 3 was the m s b, b 0 was the l s b, b 3 0. So, actually who comes on the top b 3 you can write this say b 2, b 1, b 0. So, 0 0 0 0 0 0 0 0 this guy comes on the top b 3 0, b 2 0, b 1 0, b 0 0, then who comes b 3 1, b 3 1 then 0 then 0 then 0, this way. So, you see if it is a scale who comes first 0 0 0 0 then 0 0 0 1. So, 0 0 0 0 in a scale for the time being forget this, let me write this way b 0 this way b 0 in this reverse product b 0, b 1, b 2, b 3. So, who goes on the top 0 0 0 0, but that is actually in terms of n is what b 3 0, b 2 0, b 1 0, b. So, this guy to 0 then next guy is 0 0 0 1 0 0 0 0 then 0 0 0 1 next guy in the scale. So, next guy in the scale is 0 0 0 1, but what will be the value of small n we have to start from b 3. So, 1 0 0 0 1 0 0 0 because actual number was n that was b 3, b 2, b 1, b 0, alright.

So, this will be 1 0 0 0, 8. So, this will be x 0, x 8 next index is 0 0 0 1 done then 0 0 1 0 next index scale. This is a scale, this guy, this guy, this guy, 0 th guy, first guy, second guy. So, 0 th guy first guy, second guy, third guy, like that 0th guy means 0 0 0 0 then I go like this. So, I get these values b 0 0, b 1 0, b 2 0, b 3 0 that is pointing at the top and, but who will be that value for that have to start with b 3 first, b then b 2 then b 1 then b 0 because that is how number n was b 3, b 2, b 1, b 0 and that is 0 then I have 0 0 0 1 that is the next guy the scale.

This was first guy done now next guy 0 0 0 1 0 0 0 1 or in terms of original name n it will be 1 0 0, 8; 1 0 0 0. So, 8, x 8 then again 0 0 1 0 that is 0 0 1 0 that is number 2, but that will be actually 0 1 this is b 3 0 0 1 0 0 that is 4 then again 0 0 1 1 0 0 1 1 that is the next guy number 3. So, 0 th, first, second, third in this scale. So, now, the third guy is 0 0 1 1 that is why I am pointing here, this is pointing there, this is pointing there this, but third guy means in terms of n b 3 is the m is b. So, 1 1 0 0 1 1 0 0 that is equal to 1 what 12 and dot, dot, dot, dot, and you get that order in which this guy is finally arranged allowed on the scale.

We were started dividing into even part, odd part then again further even part, odd part then again further even part, odd part then it become 1 point even part, odd part and 1 point sequences. So, even odd again, even odd again that is the order. So, in this order 0 th position is 0 0 0 0 here you get b 3 0, b 2 0, b 1 0. So, 0 then 0 0 0 1 means actually 1 0 0 that is x 8 dot, dot, dot, dot, alright this is called bit reversal order. I do not know whether I have time enough to do this, I will leave you as an exercise to you, suppose they have got a filter.

(Refer Slide Time: 20:37)



If I have filter 0 to N minus 1 and you got is h n you got a sequence infinitely long or very long sequence. This is your x 0, x 1, dot, dot, dot, dot, you have to convolve them. Then 1 into convolve is that this sequence suppose, you break into from 0 to l minus 1, then next 1 is l 2, this is 1 block, then next is another block l 2, 2 l minus 1 then another block and then convolution is a linear operation. So, this sequence is summation of this sequence plus this, plus this, plus this, dot, dot, dot.

So, convolution of h n with the overall, this convolution of h n with this plus h n, with this plus h n, with this convolution of h n with anybody say with this that you can do by convolving this with you can bring this entire stuff here, with all those samples here first convolve between h n, and that block and then you can shift it by this much amount, why because in terms of z transform convolution means the product of the 2 z transform H z and z transform of that block, but that block means z to the power minus if it is 2 l z to the power minus 2 l into the z transform of this block when put here.

So, first you convolve between H z and this z transform of this block here, multiply and then z to the power minus 12 means result has to be shifted here. So, if you convolve between say this and this you know by convolution it will stretch the way. If you do graphical color convolution this will be reversed, you have doing only these blocks, suppose you are dealing with only this block and the other one, this is reversed if you keep shifting it you know it will finally, I mean if you reversed h 0, h 1, h 2 that is the way h 0, h 1, h 2. So, you get us output at 0, if you shift it to the left, nothing if you go to shift it to the right, when h 0 will come somewhere here, last guy will still be here. So, you will get over a longer range. So, this output you save.

Then again, this output is stored, then for convolution between this and this, what to do you bring this block as I told you here and convolve this with this. So, you get an output, that output is shift by l because if you call this to be say something say, x prime n this is this block is nothing, but delayed version of x prime n delay by L delay by L. So, better you that means, z transfer from this block alone would have been z to the power minus L into z transform of this x prime n say capital is prime z that into h z.

So, first carry out h z into x prime z which is nothing, but convolution between these 2 and then apply z to the power minus L that means shift the result by L. So, you convolve this with this and shift, so that result again will start upper shifting will start from here and will go like this, but you add this part will be added again, next block and this is x prime z right you start from L, but does not remain for a length of L only it becomes actually L plus N minus 1 total length. So, again it will enter the next block then again you take convolution between these 2 first, you shift it here convolve the 2 and then shift it, shift the result here. So, again there will be like this, this is called overlap and save method.

You can use DFT to do this convolution. As you know you have to do zero padding. So, that the length of the 2 sequence is same. So, this block if length capital L and this is N you have to add some L plus N minus 1 or 2 I forgot that many 0s here and so, total length will be L plus N minus 1, it is N. So, L minus 1 0 is here N minus 1 0 is here to this block. So, total length will be L plus N minus 1 that many point DFT, DFT into DFT inverse DFT you will get this result.

Similarly, this result these are one way called overlap and add this is another one which is called overlap and saved now. I do not have time because this little it will take little longer. So, I would leave it as an exercise to you to read from the book that is not very difficult. So, I conclude this course and I hope by this journey through this course you could gathered some and you could get some useful material in this topic.

Thank you very much.