Signal Processing Techniques and Its Applications
Dr. Shyamal Kumar Das Mandal
Advanced Technology Development Centre
Indian Institute of Technology, Kharagpur

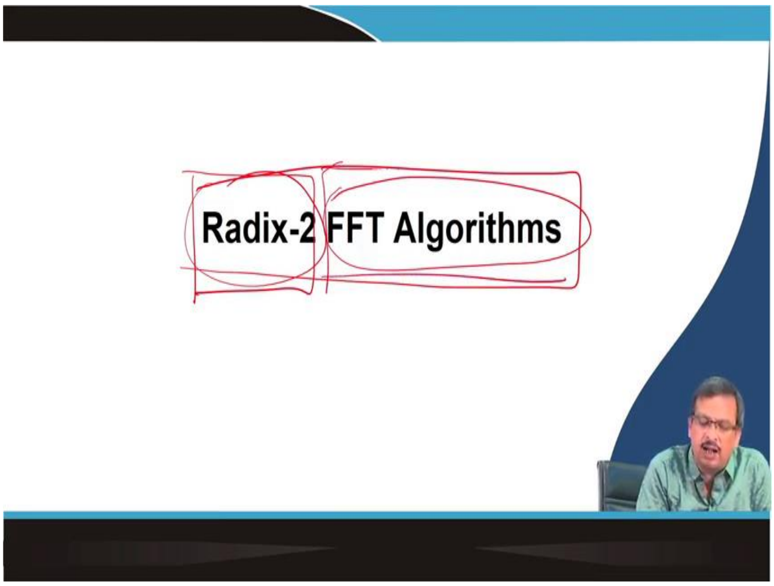
Lecture - 32
Radix - 2 FFT Algorithms

So, students, last week we discussed the divide-conquer method for efficiently calculating the discrete Fourier transform. So, the whole purpose of the divide-conquer method is to improve the efficiency of the calculation of discrete Fourier transform. As you know, if I directly compute DFT, the computational complexity is the order of N squares, which means the number of multiplications is nothing but an N square, and the number of additions is N into N-1.

So, N square complex multiplication, again, if I converted to the real multiplication, it will be 4 into N square. So, I want an efficient algorithm that, with the computational complexity, will be reduced compared to N square. So, the divide-conquer method is one of the methods we have seen which can improve the efficiency of calculating discrete Fourier transform.

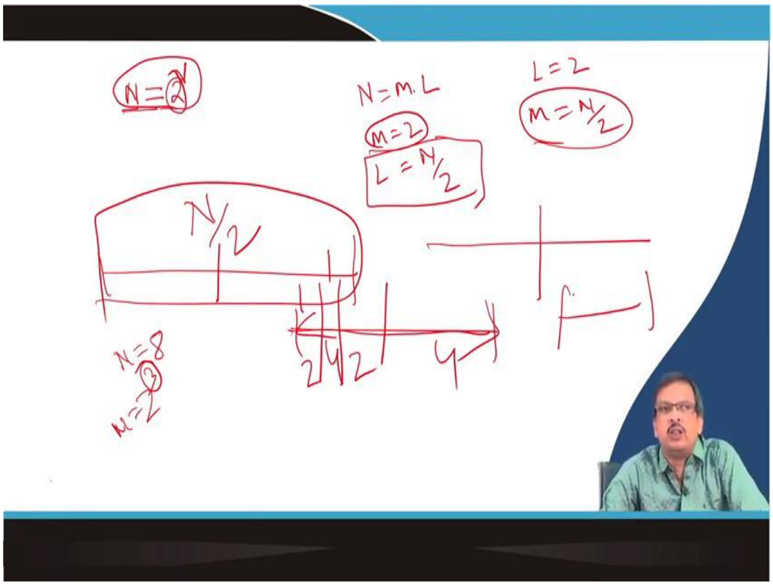
So, I can say the fast Fourier transform. Today, we discuss a special algorithm, commonly known as an FFT algorithm, called the Radix(2) FFT Algorithm.

(Refer Slide Time: 01:42)



So, today's lecture will be on the Radix(2) FFT algorithm. So, if you look at this term, it contains two words: the first word is called radix(2), and the next one is the FFT algorithm. So, 2 phases, I can say. So, radix(2); what do you mean by radix(2)? What is the meaning of a radix(2)? Now, suppose I say what the mean is; so, I am explaining what the meaning of radix(2) is.

(Refer Slide Time: 02:14)



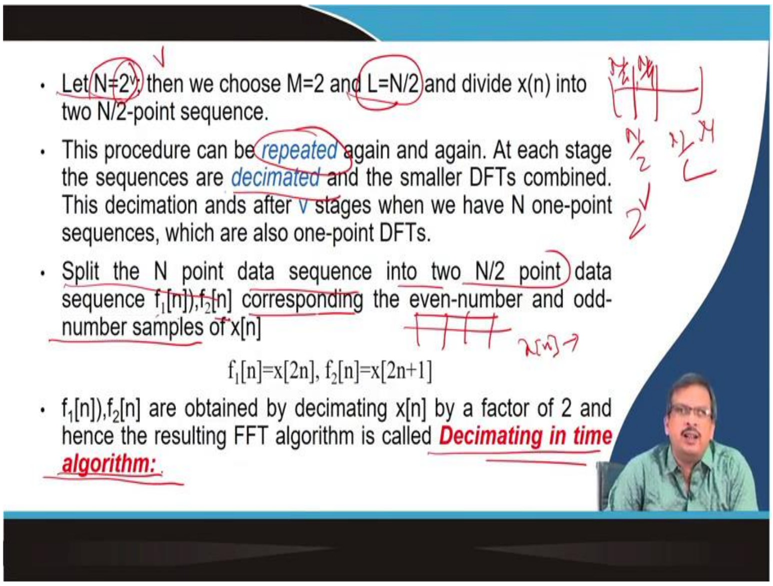
Now, if I say the length of the DFT is N and N can be expressible in terms of 2^V, where V is an integer. So, in the divide-conquer method, I said the limitation is that N should not be a prime number. Here, I said the limitation in radix(2) I said N can be expressed in 2^V. So if I say if I apply the divide-conquer method that N is equal to M into L, let us say M is equal to 2, then L is equal to N/2; see or if I say L is equal to 2 then M is equal to N/2.

So, whatever I can say. So, if I say that M is equal to 2, then the entire length of the signal is divided into 2 sections, or L is equal to 2, M is equal to N/2. So, what I said is that if my N is expressible in 2^V, com V is an integer, then I can compute discrete Fourier transform of N point by dividing whole N in every time by 2.

Since I am dividing the sequence every time 2, it is called the radix(2) algorithm, or I am expressing N as a 2^V term, which is why it is called the radix(2) algorithm. So, every time, the whole length of the signal will be divided by 2. Now, as you know, think about who has already done the binary search algorithm; what is a binary search algorithm?

That is a normal search; if the computational context complexity is N, then in binary search, it is nothing but a log to N because every time I divide the sequence into 2 parts, I apply the search algorithm. So, that reduces the computational complexity; here also, every time I divided the sequence into 2 sequences and that reduces the computational complexity.

(Refer Slide Time: 04:42)



So, in the radix(2) algorithm, what we said is let N be represented in 2^V, where V is an integer M equal to 2, then L equal to N/2 or N equal to M equal to 2, then L equal to N/2. So, whatever you can say, divide-conquer methods, either M or L, M is the number of columns, and L is the number of rows.

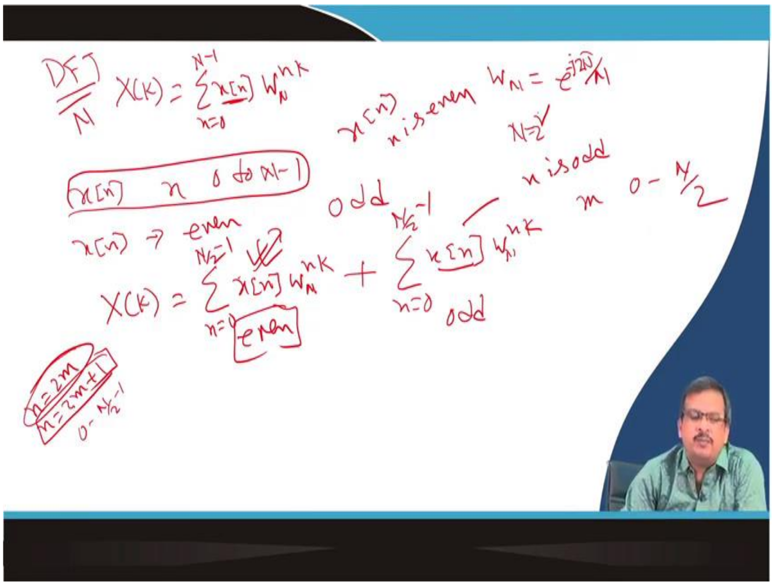
So, this procedure that divides the signal into two terms lets us N, then N/2, N/2, and then N/2, which can be divided again. So, N/4, N/4. So, this procedure will repeat unless I get 1 1-point sequence. So, 2^V, I am dividing the sequence in a 2 way every time in 2. Let us say I said I have a sequence of length 8, and N is equal to 8. So, how do I express N? N is equal to 2^3. So, I have a sequence length of 8.

So, the first time, I divided it into 4 4; the second time, I divided it into 2 2, and the third time, I divided it into 1 1. So, if you see how many divisions are possible? 3 time. So, that is called a number of stages. So, what are we dividing? We divide the input sequence or the output sequence; let us talk about the input sequence. So, what are we doing? We decimated the input signal. So, when we divide the input signal, then we call decimation in time, decimating in time, or if I divide the output signal, then it is called decimating in frequency.

So, let us say I describe the decimating in time algorithm first. So, what is required? We split the endpoint data sequence into two N/2 point data sequences, F1[n], and F2[n], corresponding to the even number and odd number. So, what am I doing? I am splitting x[n] in two data sequences, dividing by 2.

Since the sequence is expressed in terms of 2 to the power something, that is why it is divisible every time is divisible by 2, and when I divide when I decimate the signal time domain signal, then it is called decimating in time algorithm. So, what is the algorithm? So, how do I do that? Let us say go into it.

(Refer Slide Time: 07:45)



So, what is there? I have

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{nk}$$

So, this is a discrete Fourier transform, an endpoint discrete Fourier transform. This is the endpoint discrete Fourier transform, where ωN equals $e^{-j2\pi/N}$, ok or not. So, as I said, I want to divide the sequence into 2 sequences. So, x[n] has a length of n; so x[n], n varies from 0 to N-1.

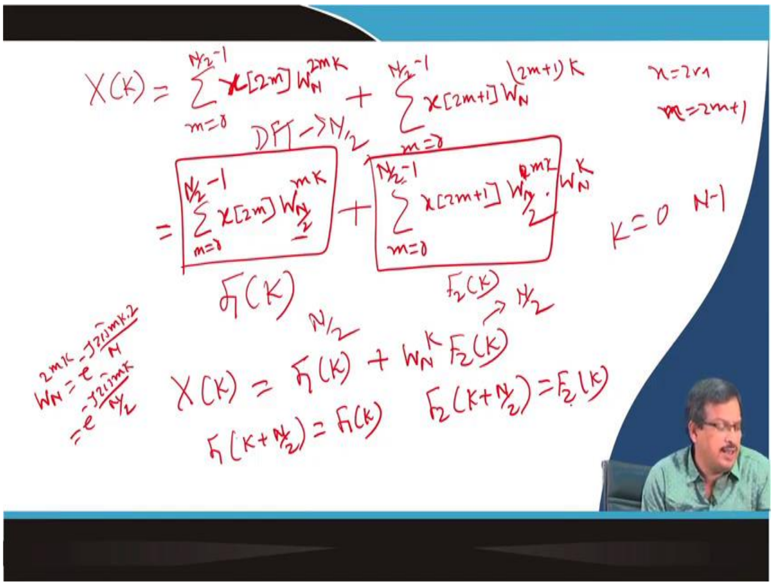
So, x[n] has a length of n capital N and N in the form of 2 to the power something, 2^V, V is an integer. So, N is divisible by 2; so, let us say I want to divide this x[n] into 2 parts. So, how can I divide? Let us say so: N is an index. So, how can I divide it? So, you cannot say that ok 0 to N by N/2 and N/2; it is a sequential division.

So, let us say instead of doing that, I am except I am accessing the signal or I am dividing x[n] in terms of 2 sequences: one is called an even sequence, and another is called an odd sequence. So, I can say X(k) is equal to n equal to 0 to N/2, even sequence also divided by 2, length is N/2 minus 1 x[n] WN n k even. This I am accessing the signal which is even, now I am accessing the signal n equal to 0 to N/2 minus 1 x[n] WN n k.

I am accessing the signal, which is odd. So, what is odd? The index of the signal is odd. So, here x[n], where n is even, here x[n] where n is odd. So, the index of the signal, x[n], is stored in a memory location; I am accessing the signal even index, odd index, even index, and odd index. So, if I say that in mathematics, how do I write an even number?

So, I can say that n is equal to 2m, representing the even number, where m is also an integer and n is equal to 2m, plus 1 always represents an odd number, which is where m is also an integer. So, I can say if m varies from 0 to N/2, I get N/2 number of even number, and if m varies from 0 to N/2, then I can say N/2 minus 1, I can get N/2 number of odd number.

(Refer Slide Time: 11:21)



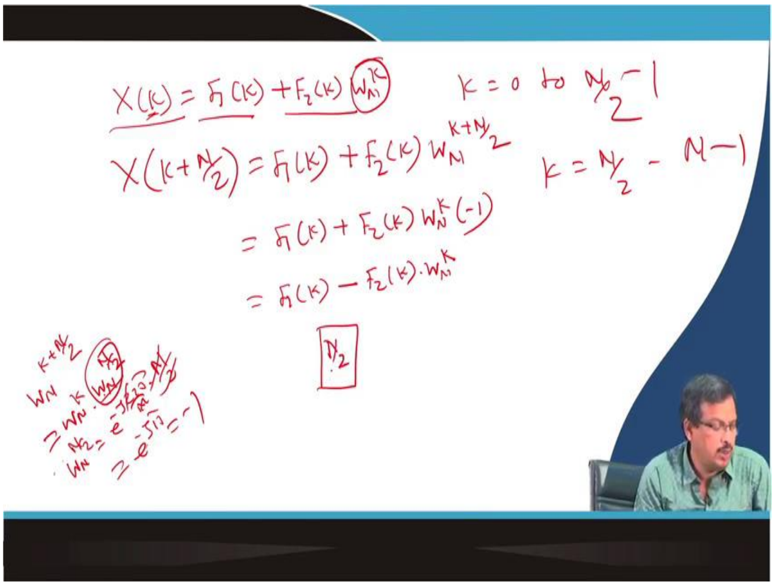
So, I can say X(k) can be written as so, and X(k) can be written as m equals 0 to N/2 minus 1. So, x[n] I am accessing the signal which are even. So, x[n] is replaced by 2m x[2m] into ωN 2m k plus m is equal to 0 to N/2 minus 1 x[2m] plus 1 into W N/2 m plus 1 k. So, n is replaced by 2m, and n is replaced by 2m plus 1.

So, for odd, it is 2m plus 1; for even, it is 2m; now, if I do that. So, how do I write this one down? m is equal to 0 to k capital N/2 minus 1 x[2m] into W N/2 mk; can I write that or not? So, what is there? Now I know what WN is. W N/2 m k is nothing but an equal to e to the power minus j 2 π m k into 2. So, 2 is again there divided by N.

So, I can easily write down $e^{-j2\pi mk/(N/2)}$. So, that is why I write N/2 here, plus m equal to 0 to N/2 minus 1 x[2m] plus 1. So, there is a W N/2 m k; so, W N/2 m k into WN k; again this can be written as this one. So, I can say that this one is a DFT equation whose length is N/2.

So, let us say this is F1[k], and this one is also an DFT length whose length is N/2; let us say this is F2[k]. So, how can I write? I can write X(k) is equal to F1[k] plus WN k F2[k], now say X(k), k varies from 0 to N-1, but F1[k] the length is N/2 and F2[k] length is N/2. So, if the length of the DFT is N/2 then N/2 is the period. So, I can say the F1[k] plus N/2 is equal to nothing but a F1[k]; similarly, F2[k] plus N/2 is nothing but a F2[k].

(Refer Slide Time: 14:52)



So, I can say that X(k) is equal to F1[k] plus F2[k] into ωN k, for k is equal to 0 to N/2. Now for X(k) plus N/2, so, k varies from N/2 to N, N-1 because X(k) varies from 0 to N-1 minus 1; so, it will be N-1. So, I equal to F1[k]. You know the k F1[k] plus N/2 is nothing but the F1[k].

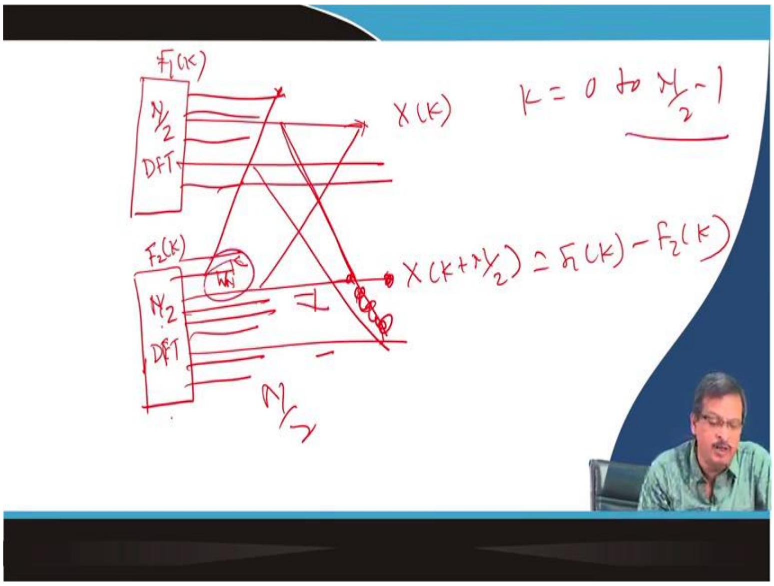
So, it becomes F1[k] F2[k] because F2[k] plus N/2 is nothing, but the F2[k]. So, I do not compute F2[k] again and WN to the power k plus N/2. Now what is WN k plus N/2 is nothing but a WN k into WN N/2. So, what is this one? WN is N/2 is equal to $e^{-j2\pi/N}$ into N/2.

So, N N cancel, 2 2 cancel

$$e^{-j\pi} = \cos(\pi) - j\sin(\pi) = -1$$

So, I can say this can be written as F1[k] plus F2[k] into WN k minus 1 because WN N/2 is equal to minus 1 ok. So, I can say that F1[k] minus F2[k] into WN k. So, if you see that X k can be represented in a 2 part F1[k] F2[k] and F2[k] multiply will be. So, I can say that suppose this is my F1[k]. Let us say this is N/2 point DFT ok. I will write in here.

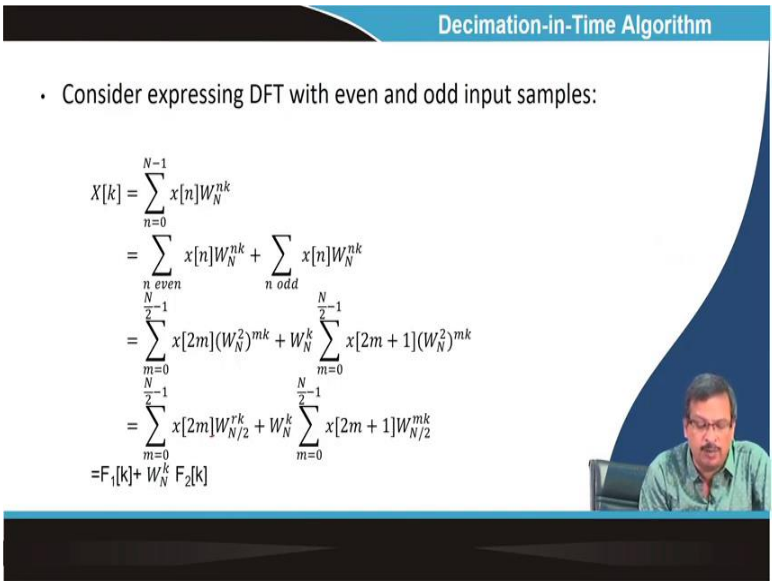
(Refer Slide Time: 17:09)



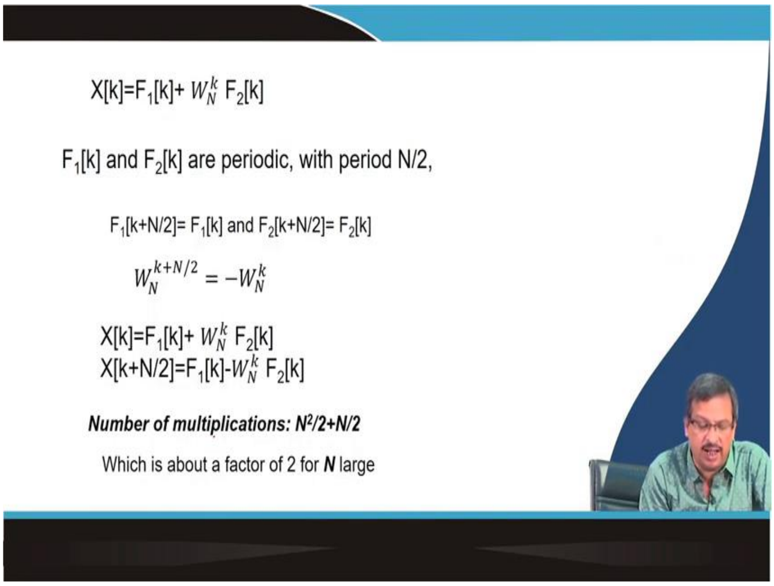
So, this is my F1[k], this is N/2 point DFT and here is this is F F1[k], this is F2[k], this is N/2 point DFT, length is N/2. So, here I can say that F1[k] and F2[k] will be added together to get X(k), where k varies from 0 to N/2 minus 1, but when it is N/2 plus 1, I can say this is nothing, but an F1[k]. So, this is nothing but a minus 1; I can say, let us say, give it in here, not this point. So, I can say minus 1.

So, this point is X(k) plus N/2 if I say it is nothing, but an F1[k] minus F2[k] multiplied by this F 2; if you see here, F 2 is multiplied by WN k every time. So, I can say F 2 is multiplied by WN to the power k. So, whatever the F2[k] number is there. So, the N/2 point will be there, the N/2 line will be there and here will be the N/2 line. Now, every line will be added together here up to N minus 2, and every line will be subtracted from this one. So, this one will be multiplied.

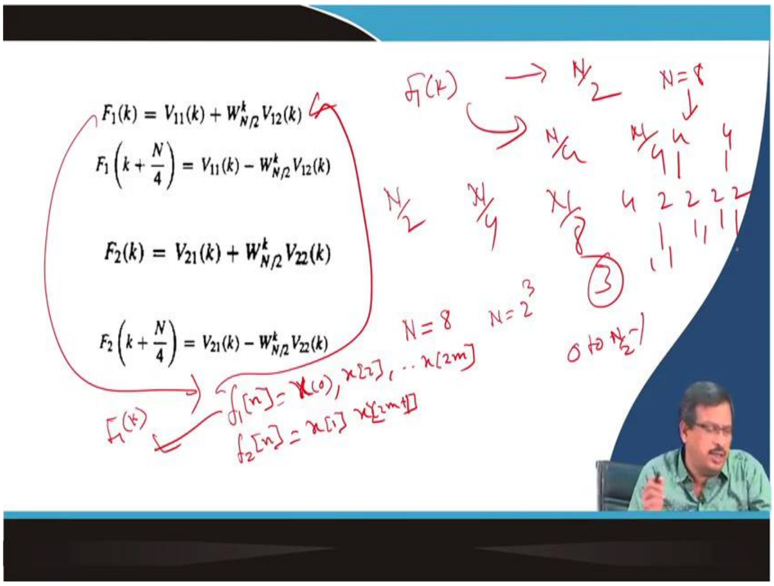
(Refer Slide Time: 18:53)



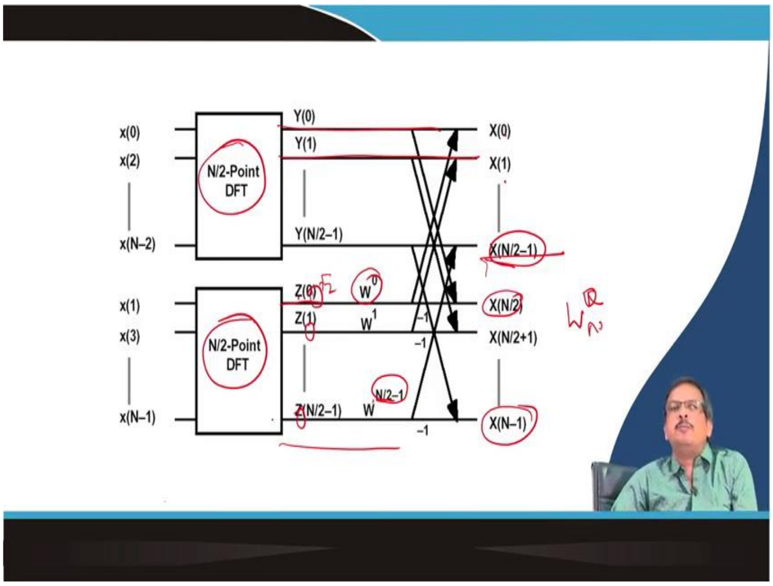
(Refer Slide Time: 18:54)



(Refer Slide Time: 18:54)



(Refer Slide Time: 18:57)

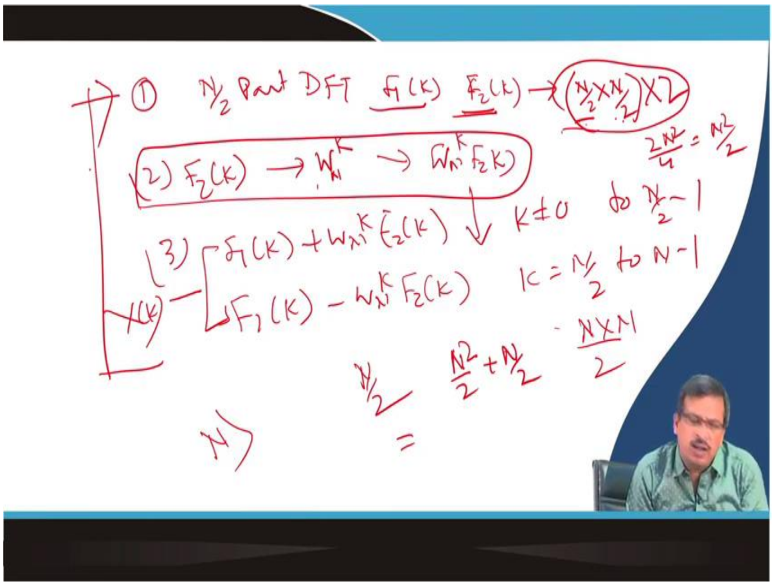


So, if I draw the picture directly and correctly, you can see it will be this kind of picture. So, this is N/2 point DFT; this is N/2 point DFT, forget about the signal. So, I get X(0) X(1) X[n-2] x[n] by 2 to N-1. So, I can say, and this one, this F 2 is let us say this is Z is multiplied by ωN k. So, k equal to 0 to k equal to N/2 minus 1 clear.

So, what I have to do, instead of directly computing X(k), is divide the sequence into 2 parts; the m sequence varies from 0 to N minus 2 because the length of the even signal is N/2, and the length of the odd signal is N/2. Then, I compute N/2 point DFT, and then I know the output is nothing but an I f every N/2. first, N/2 is F 1, and the second N/2 is f 2.

So, F 1 plus F 2 for k equals N/2 minus 1, and the next one is F 1 minus F 2, and every time, F 2 will be multiplied by WN to the power k. So, what is the computational complexity if I say in type decimation in time algorithm? So, what is the stage required for decimation in the time algorithm?

(Refer Slide Time: 20:33)



So, stage 1 number 1 stage is that I have to compute N/2 point D DFT for F1[k] and F2[k]. Then, in the second step, F2[k] will be multiplied by a number by a complex factor, which is ωN to the power k. In 3rd stage, I have to add them, either F1 add F1 plus ωN k F2[k].

So, in the second stage, I am producing ωN k F2[k], now F1[k] once will be plus F2[k] for k equal to 0 to k equal to 0 to N/2 minus 1 and F1[k] minus ωN k F2[k] for k equal to N/2 minus N/2 to N-1. So, I get X(k) from these two equations. Now, what is the computational complexity? In this stage, instead of required, I require N/2 into N/2 complexity into 2 because 2 DFT complexity of F 1 is N/2 complexity of N/2 into N/2, the complexity of F 2 is also N/2 into N/2.

So, N/2 into N/2 plus N/2 into 2 into; so, N 2 into 2 into N square by 4, I can say N square by 2 ok. Now, what is the complexity of the second stage? A number of multiplication; so, here, the multiplication can be N/2. So, I can say total multiplication is N square by 2 plus N/2. So, the total number of multiplication required by the methods I said is N square by 2 plus N/2 instead of N into N.

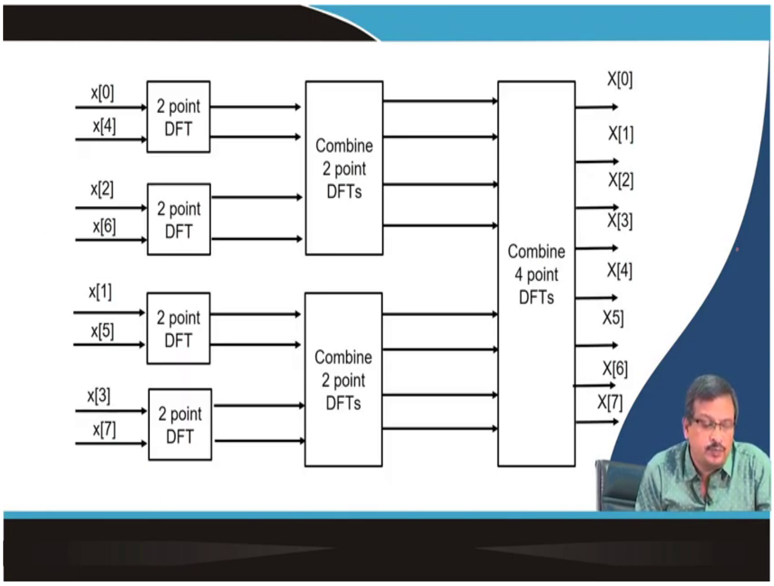
So, I can say if the N is quite large, then I can say it is reduced by half; if the N is very large; so, it reduces the complexity by 50 per cent, half 1 by 2. So, if you see this is a single stage, now this stage can be repeated for another one. What is that meaning? This means that now I can say F1[k], whose length of the DFT is N/2 again, can we apply the same principle, which can be divided into 2 parts.

So, N/4 and N/4. So, F1[k] has a sequence that sequence can be treated as a signal and take N/4 into N/4 4. Similarly, F2[k] also can be computed. So, for each stage, N/2 is one stage, and N/4 is another stage, and N/8 will be another stage. So, a number of stages depend on how you express N.

If N is equal to 8, then I know the number of the stages is 2^3, 3 stage is there; I can divide the signal into 3 stages. So, when I compute F1[k], what is the signal index? Let us say F1[k] time domain signal is small f1[n] is nothing, but the X(0) small x(0) small x(2) then it is nothing, but a x[2] into m, m vary m is varies from 0 to N/2 minus 1 and f2[n] is nothing, but the x(1) x[2m] plus 1.

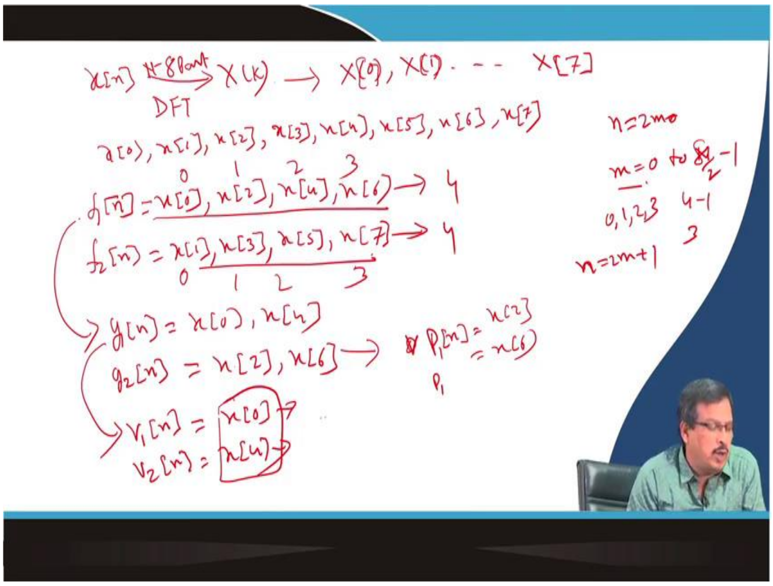
So, I accessed those indexes that are odd, varying from 0 to N/2 minus 1. So now, this f s1 sequence is again divided into sequences. So, this division is going on going on, and ultimately, I get the signal where it is not divisible this 1. So, if I say the N is equal to 8, then first, you divide the signal N/2 N/2 4. So, 8 is divided into 4 4, then 4 is divided into 2, then 2 is divided into 1 1 1 1 1 1; 1 odd signal, 1 even signal, 1 odd signal, 1 even signal. So, if you see that, how do I do it?

(Refer Slide Time: 26:06)



So, now let us say take an example 8 point DFT, then you can understand how this radix(2) algorithm works.

(Refer Slide Time: 26:19)



Let us say I am not showing you the slide; I will show you the slide later on. Let us say I have a signal x[n], and I want to take 8-point DFT. N is equal to 8, and I calculated X(k). So, what is the signal up there? x[0] index(0) x(1), x(2), x(3), x(4), x(5), x(6) and x(7). So, a number of 8 signals are there, similarly X k also X(0) capital X(k), X(1) dot dot dot up to X(7). Now, I said decimation in time.

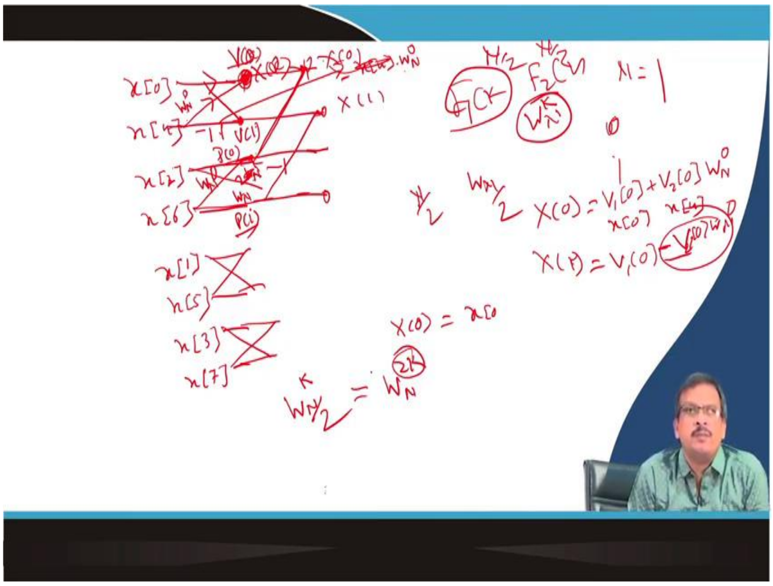
So, every time, I divided the signal into 2 parts. So, I want to group f1[n] is the even signal. So, I want to access the index n, which is expressible by 2m, m varying from 0 to N/2 minus 1. So, what is N/2? In the case of 8/2 - 1 means 4 - 1 means 3. So, if m is equal to 0, I get x(0); if m is equal to 1, I get x(2); if m is equal to 2, I get x(4); if m is equal to 3, I get x 6 done 0. So, m varies from 0, 1, 2, 3.

Now, what is f, and what signal is f2[n]? It is n is equal to 2m plus 1. So, m is 0; that means, x[1], m is 1 x[3], m is 2 x[5], m is 3 x[7]. So, this is 4 numbers of signals, and here are also 4 numbers of signals. Now, again, this f 1 can be treated as a sequence whose index is 0 1 2 3. So, again, I divided it into g 1 sequence, which is even and f 1 is divided by g 1 and g 2. So, g 1 in even sequence again if I consider 0 1 2 3; so, it will be x(0) and x(4), g 2 will be x(2) and x 6.

Now, again, if g is divided by another stage, let us say v 1; so, v 1 n small v 1 n will be x(0), and v 2 n will be x(4). So, this is an even signal; this is an odd signal; further division is not possible. So, I divided the signal in this way. So, what is the group of the signal? 0

4, similarly if I divided this sequence, let us say v instead of v, let us say p 1 p 2. So, p 1 n will be this sequence, this sequence. So, x(2) and x 6, similarly, I divided this sequence into 0 1 2 3. So, 1 and 5 will be clubbed together, and 3 and 7 will be clubbed together.

(Refer Slide Time: 30:16)



So, when I say I have an 8 n equal to 8 x[0] to x[7], the sequence is club like this x(0) with x(4), then x[1] with is x[2] with x[6]. So, this is even sequence, this is odd sequence, even sequence, odd sequence. Then x[1] will be clubbed with 1 find, then x[3] will be clubbed with x[7].

Now, if you look at the final, the length of the DFT N is equal to 1, and N is equal to 1. So, the signal, so, the length of the DFT, is 1. So, the signal is even, the sequence length is 0 and the odd sequence length is also 1, signal length single length sequence. So, when I want to compute this one, let us say this is X k. So, I want to compute X k; so, X X(0) is equal to I can say even 0.

So, let us say this is represented by, let us say, V 1 and V 2. So, X(0) is equal to V1(0) plus V2(0) into WN k is 0. So, what is V2(0)? So, what is V1(0)? x(0). What is V2(0)? x(4). So, if I say that X(0) k is equal to 0, that means these two will be added together, and this will be multiplied by WN to the power 0.

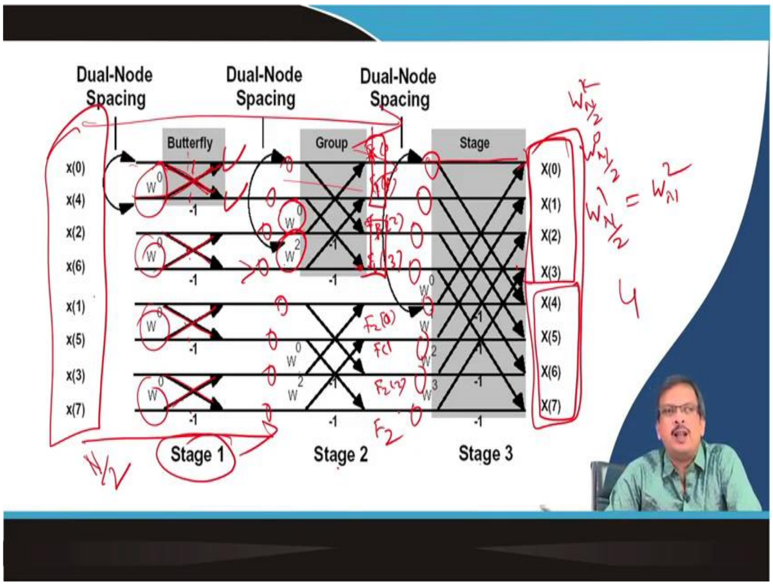
So, I have multiplied x(4) with WN to the power 0, then added up with X(0). Now, what is this one? So, what is X(1)? X(1) I know this sequence is periodic, so I can say X(1) is nothing but a V1(0) minus V2(0) into WN to the power 0. So, I have to subtract instead of plus; so, I have to subtract 1 and then. So, this will create this point: it is nothing but an x(4) multiplied by WN to the power 0 and multiplied by minus 1.

So, it is created minus x(4) WN to the power 0, which is this part. Similarly, the same thing will happen here. The same thing will happen here, and the same thing will happen here. Now, if I say this length here, I get the length of the DFT is 2. So, here I get X(0), X(1), X(2); if I say X(0) is nothing, but this one x, this one will be multiplied by this one. So, I can say X(0) will be then x[0], or I can say this output us this output is, let us say, V 1; so, V capital V k.

So, V k 0 V k 1, and this is also, let us say, instead of P k 0 and P 1. Now, I am saying that these 2 sequences again will be combined; this one will be combined with this one to get that. So, a V k and P k will be combined in positive and then again, it will be multiplied by 1 to get this one. So, you get this one, but what is required? Again, I have to multiply by WN to the power k. Now, here, if you see WN in the power k, So, k is equal to k, which is equal to what?

Here k what is the length? Length is here length is not N, there will be N/2, if you see when we are calculating F1[k] and F2[k], we multiply by WN to the power k. The length of the DFT when it is F1[k] length is N/2 and F2[k] length is N/2, then it is WN k, but when it is N/4, then it will be W N/2 . So, W N/2 to the power k; so, which is nothing, but a WN 2 k. So, k is equal to 0, and it will be WN 0; here, it will be WN to the power 2, and k is equal to 1; so, 2 will be there.

(Refer Slide Time: 35:17)



So, I can say that the whole thing will look like this: W 0 butterfly W 0 butterfly W 0 butterfly W 0 butterfly, here I am combining. So, this is ω 0; so, ωN/2 k. So, ωN/2 0 ωN/2 1 which is nothing but a ωN square. So, it is nothing also not ω WN square W square not ω W W square. So, this W square will be multiplied by this line minus 1; I get this 1.

So, I get X(0). Let us say I get another let us say F 0 F 1 F 1 0 F 2 0 F 3 0 F 4 0 sorry F 1 0 1 F 1 2 F 1 3, here I get F 2 1 F 2 2 F 2 0 F 2 1 F 2 2 F 2 3; now I combining at the eighth output stage. So, I know F 1 plus F 2 in case of X(0) up to N/2, this portion it will be plus and this portion it will be minus. Similarly, here also this portion plus this portion minus, here this portion plus this person minus. Now, what should be my algorithm?

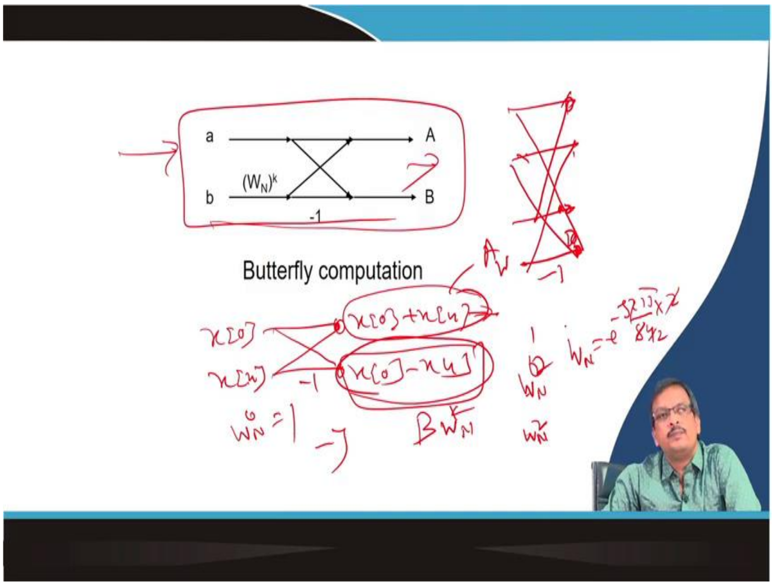
So, what is my algorithm? What do I observe, and what should be my algorithm? So, if we see that this is stage 1, the signal is flowing from this side to this side. But, when we understand, we understand from this side to this side. How do we understand? We understand that N/2 N/2. Again, I go for this. So, again, on this side, it will be x(1) x(1) x(1), and what is the signal flow? We take the 2-point DFT, then 4 2 points combined, then 4 points combined, and I get the complete DFT.

So, signal flow is like this, this stage to this stage, because this portion, I know I can calculate this weight factor and I can flow the signal in this direction, I get this one. If you see every stage, there is a 4 butterfly; this is called butterfly. If you see 2 wings of a butterfly, that is why it is called a butterfly. So, at this stage, there are 4 butterflies. So, if

I say, in general, for every stage, there will be a N/2 number of butterflies. If it is 8, then you see it is 4 butterflies; if it is 16, in every stage, there will be 8 butterflies.

So, observation is that here also that if you see 4 butterflies there, this line, this line, along with this line 1 butterfly, this line and this line another butterfly, this line and this line 1 butterfly and this line another butterfly. So, 4 number of butterflies, here also if you see the 4 number of butterflies, this line with this line, this line with this line, this line with this line and this line with this line.

(Refer Slide Time: 38:37)

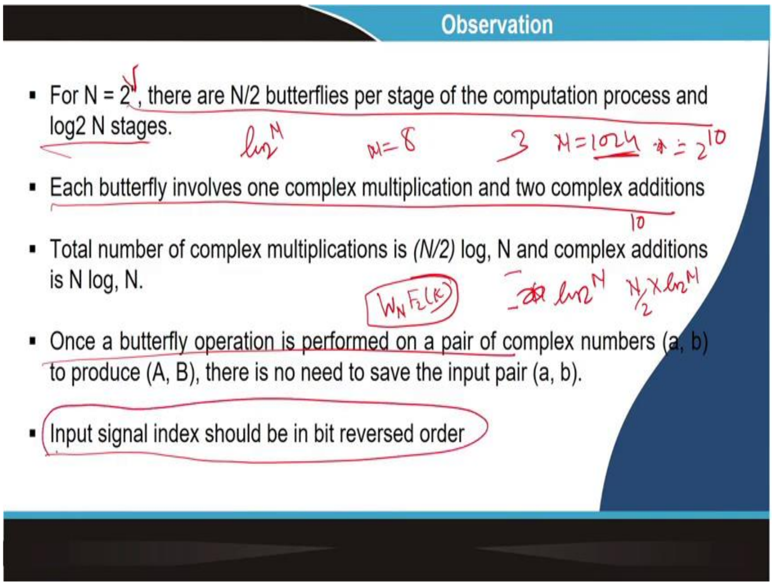


So, there are 4 butterflies, and every butterfly will look like this; this is the generalization form of the butterfly. So, here is the input and output. So, initially, I take input x(0) and x(4), and I compute WN to the power 0, which is 1. So, I can say I know the value of x(4), I know the value of x(0). So, I can say this one is nothing but x(0) plus x(4), and this one is nothing but an x(0) minus 1 minus x(4).

So, I got this point, and this point, then again, this one will be this is the output. So, this is A, and this is B. Now, A B will be multiplied by WN k again. So, in the next stage, in the case of 4 points, it is nothing but the WN square or 1 0 or WN to the power square. So, there is a 2 index(0) and square. So, I know the value of WN W a WN is equal to $e^{-j2\pi/N}$ is equal to 8; so, 2 π by 4. So, in this case, it will be 1, but in this case, it will multiply by 2.

So, it is π by 2, this is minus j. So, then I can say that this portion will multiply with minus j, and this portion will be stored in A and propagated for the next stage. And, then again, this one will be multiplied with the added with this one, and this one again will be multiplied with this one; I have already got multiplication I got. So, this one will be minus with this one, I get this two, then again this one, sorry, I guess these two, these two; then this one with this one and this one with this one. So, 1 2 3 4, I get that way.

(Refer Slide Time: 40:29)



So, now I can say the observation that if N is equal to 2^V, there are N/2 number of butterflies for every stage. And how many stages will be there? Log 2 log 2 N. If it is 8 N equal to 8, there is a 3 stage; if equal to N equal to 1 0 2 4 N equal, then there will be a 10 stage because 1 0 2 4 is nothing but a 2 to the power 10. So, it is nothing but a 10-stage. So, each butterfly involves one's complex multiplication, where WN will multiply with F2[k].

So, since the length of the butterfly is 1, I can say one complex multiplication and two complex addition; one is addition, one is subtraction, and subtraction is also addition. So, I can say how many stages there will be. The stages will be N/2 number of; so, I can say there is a log.

So, how many stages are there? So, there will be a log 2 N number of stages there; each stage requires 1 number. So, I can say it is nothing; how many stages are there? N/2 number of stages are there. So, N/2 into log 2 N, each stage N/2 butterfly, each butterfly one complex multiplication. How many stage? Log 2 N. How many butterflies? N/2.

So, I can say the number of stages multiplied by N/2 is the computational multiplication required for computing discrete Fourier transform using the FFT algorithm. Now, one butterfly operation is performed, and the output can be stored in an input. Another one is that the input signal index should be bit reversed.

So, in the next lecture, I will explain what is the meaning of bit reversal.

Thank you.