

Architectural Design of Digital Integrated Circuits
Prof. Indranil Hatai
School of VLSI Technology
Indian Institute of Engineering Science and Technology, Shibpur, Howrah

Lecture - 50
FFT Architecture (Contd.)

Hello everyone, welcome to the course on Architectural Design of ICS. So, in the last class we have discussed the basics of this FFT Architecture as why we have to know the FFT the hardware architecture of FFT is that; because FFT is one of the major mostly used blocks in digital signal processing. It has a wide application in for video processing, audio processing as well as image processing all the DSP mainly uses; that means, this fast Fourier transform. So, that is why there are several works on this; that means, developing the efficient fast Fourier transform hardware architecture.

So, we have seen that there are two type of architecture; one is this decimation in time, and another one is this decimation in frequency ok. So, in decimation in time the inputs are in that means bit reversed manner. So, that the output comes in the original sequence. Whereas, in decimation in frequency we give the input in original sequence and we get the output gain bit reversed manner. So, today we will see that whenever we will use this butterfly unit ok. So, we have seen the data flow graph of the fast Fourier transform. So, depending on that this all this butterfly unit.

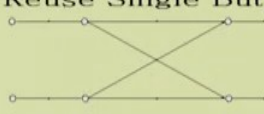
(Refer Slide Time: 01:39)

Page 20/28


Implementation

--- Two Extreme Method

Reuse Single Buttt



Fully Spread



Slow ← ----- Speed ----- → **Fast**

Small ← ----- Area ----- → Large

Complicated ← ----- Control ----- → **Simple**

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

Now, using one single butterfly; that means, this is the serial implementation; that means, if we reuse one single butterfly to compute the whole FFT operation. So, because the butterfly unit is same only the input at different stage the inputs are different ok. Why because, if I consider this 8 point FFT. So, at that time it will be having \log_2 base n it is number of stage so; that means, it will be having 3 stages. So, if we consider that that if we are having this let us say 16 point FFT.

So, at the time the number of stage will be 4, whenever we are considering this butterfly of 2; that means, here we are using this Radix-2 butterfly you need to compute the; that means, the full fast Fourier transform. So, whenever we will use one single butterfly or we will reuse the single butterfly you need to compute the whole FFT operation.

So, at that time the computation time requirement will be much more higher which makes the fast Fourier transform very slow. Whereas, the area will be very much smaller because only one single butterfly we are using ok. So, that the inputs are basically time multiplexed, but here actually the whenever we are doing this or you are using one single butterfly.

So, at that time we need the controller should be designed in such a way so, that each of this; that means, at what stage what will be the corresponding inputs to this particular node that we have to consider very carefully. So, that is why the controller of this particular block it will be much more complicated. Whereas, if we are having this fully spread that means the parallel architecture if we want if we they have just implemented.

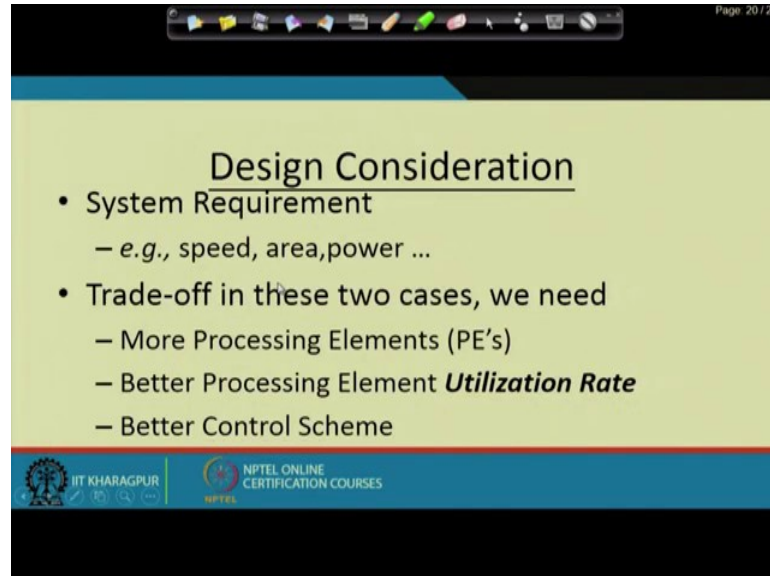
So, at that time speed wise it will be very fast. So, why the computation time will be very fast because all these particular operations are running in parallel. So, it depends upon the in one particular path. So, how many numbers of multiply multiplication and additions were run in; that means, doing depending on that.

So, that will be the critical path for this system, there I do not have to wait for that many clock cycles to what we have to wait for actually in case of serial implementation. So, area wise this will be much larger than this the serial implementation why because we are doing here we are implementing in all this that butterfly in parallel.

So, the controller is also very simple because we do not have to; that means, do the time multiplexing here because all the inputs and all these intermediate signals are already

available in this parallel case. So, that is why the controller is much simpler than this, this iterative architecture or the serial architecture.



(Refer Slide Time: 05:14)



Page 20/28

Design Consideration

- System Requirement
 - *e.g.*, speed, area, power ...
- Trade-off in these two cases, we need
 - More Processing Elements (PE's)
 - Better Processing Element **Utilization Rate**
 - Better Control Scheme

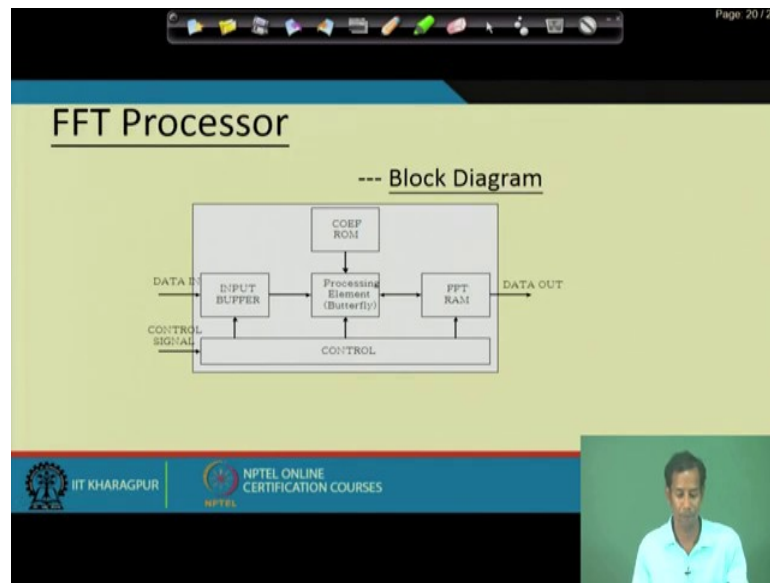
 IIT KHARAGPUR  NPTEL ONLINE CERTIFICATION COURSES

So, whenever we are considering; that means, there on the design perspective. So, at that time the systems requirement should be any of the speed, power or area. There will be trade off in these two cases where actually for the parallel implementation we can increase the area.

But the processing speeds of the design processor, the FFT processor that will be much higher. So, which requires more number of processing elements; and this better processing elements utilization rate ratio as well as this better control scheme in case of; that means, fully parallel design.

So, but in case of serial implementation or this iterative implementation. So, at that time there will be lesser number of processing elements required, but at that time of the utilization of this processing elements as well as this control scheme that will be much more complicated.

(Refer Slide Time: 06:24)



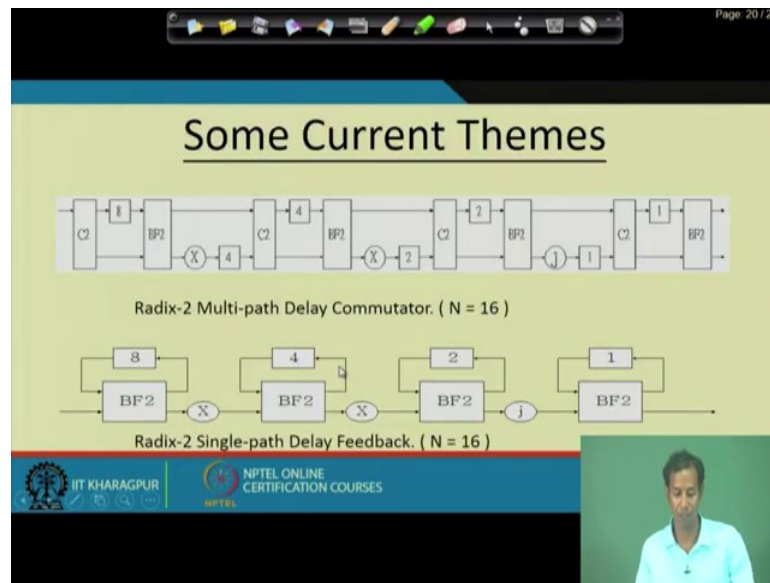
So, if we consider this one of this, this is the block diagram of FFT processors which runs in the iterative mode. So, here we are having only one single processing element block which is nothing, but this butterfly operation then there we are having this coefficient ROM; coefficient ROM means at what stage what will be the value of the twiddle factors so those values are stored in ROM. And then we are having this input buffers, and then we are having this fast Fourier transform RAM.

So, why I need this RAM is that to store the intermediate results and; that means, for one stage whenever we will; that means, compute all this; that means, for the first stage suppose for the first stage all the process butterfly operations has been done then those computed values will be used for the next stage butterfly computation.

So, at that time we have to save or we have to store those intermediate values somewhere. So, that is why we have to use the RAM to store the intermediate values which process that what; that means, this butterfly stage wise and stage wise we also these coefficients are also different. So, that will also store in a read only memory. So, that at every stage we and who will basically control all this; that means, what will be the input at what particular stage, or for which butterfly what will be the coefficient memory and where it will be again stored.

So, everything is the job of this controller. So, that is why in iterative FFT processors this controller design is much more complicated than the this parallel implementation

(Refer Slide Time: 08:20)



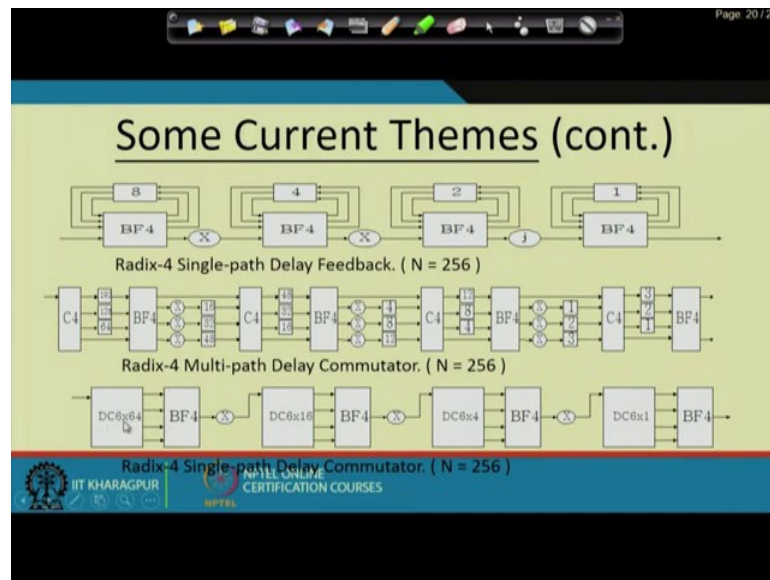
So, the sum of the current architecture actually is that, if we use this pipeline architecture parallel; that means, parallel of this along with this pipeline structure. So, here this is basically this structure is known as Radix-2 multi path delay commutator.

So, here you see we are having this coefficient; that means, storage then we are having these butterfly units. So, then again depending on this this is fully parallel implementation this is not serial, and if we; that means, what is the; that means, usefulness of this parallel implementation is that we can use the pipeline technique here.

So, if we use the pipeline technique. So, at that time the area requirement will be higher, but we can achieve a very high; that means, operating frequency or the computation time for that particular process FFT processor will be much lower. So, we have some of the application like if we have to design this OFDM processor ok. So, in OFDM actually this transceiver we have to design or we need the FFT and IFFT block which will be run in the giga Hertz range.

So, at that time we have to follow this particular architecture where; that means, the area if it is taking more area then it is no problem, but speed is my major concern. So, at that time we have to use this parallel that means implementation along with the pipeline version of this. So, here you see that this is the Radix-2 single path delay feedback. So, here only this butterfly this 2 butterfly basically it consist of 8; that means, 8 times here, 4 times here, 2 times here and 1 times here ok.

(Refer Slide Time: 10:31)



So, then again actually this is for Radix-4 single path delay feedback which is been used for in; that means, 256 point FFT. So, here you see we are having this butterfly unit of 4 which consists; that means, which runs in the loop for 8 times. Then again in the next stage it runs for the 4 times, and in the third it runs for the 2 times and again it runs for the 1 time in the final stage ok.

So, then the same thing the Radix-4 multi path delay commutator for this N point; that means, 256 point FFT they will be like it has to consider this coefficient 4 and then this will be the butterfly unit 4 and then you have to multiply with this 16, 32 and 42 then again something like this.

So, this is the structure for Radix-4 multipath delay commutator for 256 point FFT. And then again this another implementation of this Radix-4 single path delay commutator for 256 point FFT is that here you see we are having this delay commutator of 6 into 64 then again it is going to this butterfly unit of 4. So, whenever we increase; that means, for 256 point FFT whenever we consider this butterfly unit of 2.

So, at that time how many stages we will having that if log 2 base N. So, 2 to the power 256 mean that is 2 to the power 8. So, 8 stage we have to consider, but if we increase the radix to 4. So, at that time the number of stage requirement will be log 4 base 256. So, log base 256 means at that time this number of stage requirement will be divided by 2 ok.

So, but in case of this parallel implementation we can this is also; that means, this parallel; that means, this is nothing, but serial here along with the parallel ok; that means, it is not that sum of this butterfly unit in each stage there basically running in serial, but all as all of these are running in parallel. So, this is the mixture structure of serial as well as the parallel.

So, according to the; that means, the FFT architecture what we have discussed is that; whenever we will use more radix or the serial implementation at that time the number of processing elements will be or the area requirement will be lower, but the computation time will be slower.

But, whenever we will put more number of processing elements with more number of area. So, at that time the processing speed will be the computation time or the speed of operation of the computation of the FFT processor that will be on the higher side, but here we are. So, these two trade-offs if we can implement or if we can design one circuit where these some of the portion is serial and some of the portion is parallel.

So, that will give me the; that means, that a requirement for the processing elements will be 4 in this case, but the time requirement will be like as it is running for 8 times, this is running for 4 times, this is running for 2 times, this is running for 1 times. So, it has to wait for so, 8 plus 4 plus 2 plus 1. So, a total fifteen number of cycles to compute the whole task ok. But, in case of only serial implementation it has to run for; that means, this has to run for $\log_4 56$.

(Refer Slide Time: 14:53)

Distinctive merit of the above

- The delay-feedback are more efficient than delay-commutator in terms of memory utilization
- Radix-4 has higher multiplier utilization ,however,Radix-2 has simpler BF which are better utilized

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, the distinctive merit of this above is that the delay feedbacks are more efficient than the delay commutator in terms of memory utilization and the Radix-4; that means, FFT has higher multiplier utilization; however, Radix-2 has simpler butterfly which are better utilized ok.

(Refer Slide Time: 15:16)

Comparison

Radix / Speed
Low ← ----- → **High**

Control Theme
Simple ← ----- → Complex

Processing Ability / Unit
Low ← ----- → **High**

Combine the advantages
→ Further decompose high radix PE

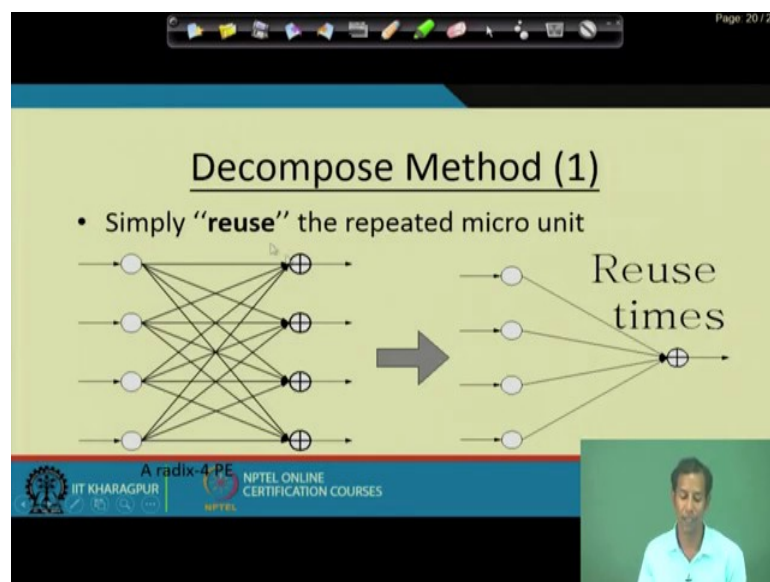
IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, if we compare this in terms of radix along with the speed if we increase; that means, the radix. So, at that time the speed will be higher and if we decrease the; that means,

radix. So, at that time speed will be on the lower. So, the control part if we increase the radix if you increase the radix.

So, at that time it needs or it will be; that means, if we increase the radix. So, at that time the control part will be much complicated and if we decrease the radix, at that time it will be much simpler. So, the processing ability or this the processing elements if we increase the radix. So, at that time it will be higher if we increase the; that means, if we decrease the radix. So, at that time the processing elements requirement will be lower. So, combine the advantage for the decomposed using high radix process, processing elements.

(Refer Slide Time: 16:26)



So, how we can decompose the FFT for a better utilization or for designing one better FFT architecture. So, suppose here we are having this four of this multiplier and then the or addition operation. So, we will just reuse this addition operation the multiplication operations are in parallel, but we will use this addition operation reuse the addition operation ok.

(Refer Slide Time: 16:58)

Page 20/28

Decompose Method (2)

- From algorithm level
 - Applying 3 index:

$$n = \langle n_1 \cdot N/2 + n_2 \cdot N/4 + n_3 \rangle N \quad \text{where } n_1, n_2 = \{0, 1\}; n_3 = \{0 \sim N/4 - 1\}$$

$$k = \langle k_1 + 2k_2 + 4k_3 \rangle N$$

$$X(k_1 + 2k_2 + 4k_3) = \sum_{n_3=0}^{N/4-1} \sum_{n_2=0}^1 \sum_{n_1=0}^1 x\left(\frac{N}{2}n_1 + \frac{N}{4}n_2 + n_3\right) W_N^{(\frac{N}{2}n_1 + \frac{N}{4}n_2 + n_3)(k_1 + 2k_2 + 4k_3)}$$

$$= \sum_{n_3=0}^{N/4-1} \sum_{n_2=0}^1 \left[B_{\frac{N}{2}}^{k_1} \left(\frac{N}{4}n_2 + n_3\right) W_N^{(\frac{N}{4}n_2 + n_3)k_1} \right] W_N^{(\frac{N}{2}n_2 + n_3)(2k_2 + 4k_3)}$$

Summation of n_1

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, then again actually we can actually we can decompose in another method like we can consider; that means, 3 different part where we can process k_1 then $2k_2$ and then $4k_3$ where this n_1, n_2 is 0, 1 and n_3 is 0 to N divided by 4 minus 1. So, this is as we have considered this three index values.

(Refer Slide Time: 17:34)

Page 20/28

Decompose Method (2) cont.

Summation of n_2

$$X(k_1 + 2k_2 + 4k_3) = \sum_{n_3=0}^{N/4-1} \left[H(k_1, k_2, n_3) W_N^{n_3(k_1 + 2k_2)} \right] W_{\frac{N}{2}}^{n_3 k_3}$$

$$H(k_1, k_2, n_3) = \underbrace{\left[x(n_3) + (-1)^{k_1} x\left(n_3 + \frac{N}{2}\right) \right]}_{\text{BF I}} + \underbrace{(-j)^{(k_1 + 2k_2)}}_{\text{BF II}} \underbrace{\left[x\left(n_3 + \frac{N}{4}\right) + (-1)^{k_1} x\left(n_3 + \frac{3N}{4}\right) \right]}_{\text{BF I}}$$

Only real-imaginary swapping & sign inversion

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, then again there are actually this is this particular actually method is known as this decomposition method where on this actually we can compute the first butterfly 1 then again combination of this butterfly ones now I can make this butterfly 2 unit ok.

(Refer Slide Time: 18:05)

The slide displays two butterfly network diagrams for $N=16$. The left diagram shows the first two stages, labeled BF I and BF II, with four butterfly units (BF1, BF2, BF3, BF4) and their corresponding operations. The right diagram shows the full network with stages BF I, BF II, BF III, and BF IV. A red circle highlights a specific butterfly unit in the BF III stage. The slide also includes the IIT Kharagpur and NPTEL logos, and a video inset of a speaker in the bottom right corner.

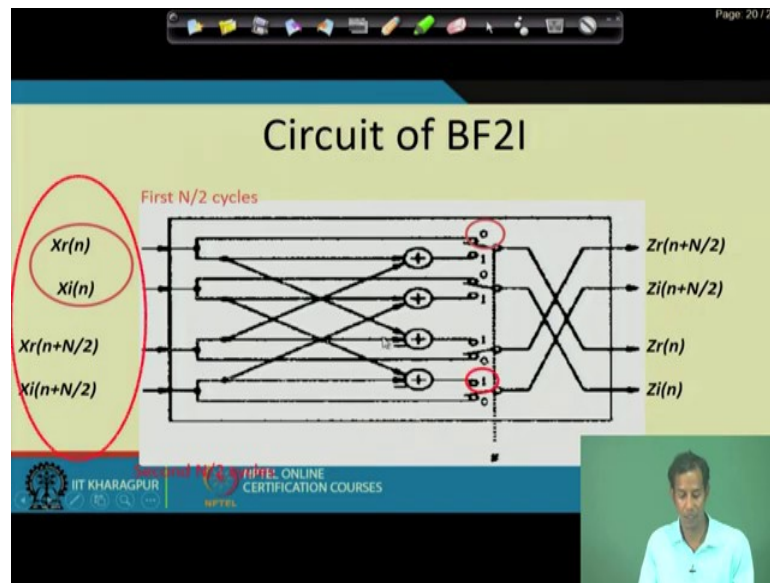
So, what will be the corresponding structure of this is that. So, suppose here we are doing this. So, here the first stage is the butterfly unit 1 and the second stage is the butterfly unit 2. So, in instead of actually if we use this kind of; that means, this decomposition. So, at that time this there will be one single; that means, the trivial multiplication with the corresponding minus 1 term for the imaginary part ok.

(Refer Slide Time: 18:39)

The slide shows a block diagram illustrating the decomposition of a butterfly unit. A large block labeled BF4 is shown to be equivalent to two smaller blocks labeled BF2 I and BF2 II. A 'Control' signal is shown entering the BF4 block and branching to control both BF2 I and BF2 II blocks. The slide includes the IIT Kharagpur and NPTEL logos, and a video inset of a speaker in the bottom right corner.

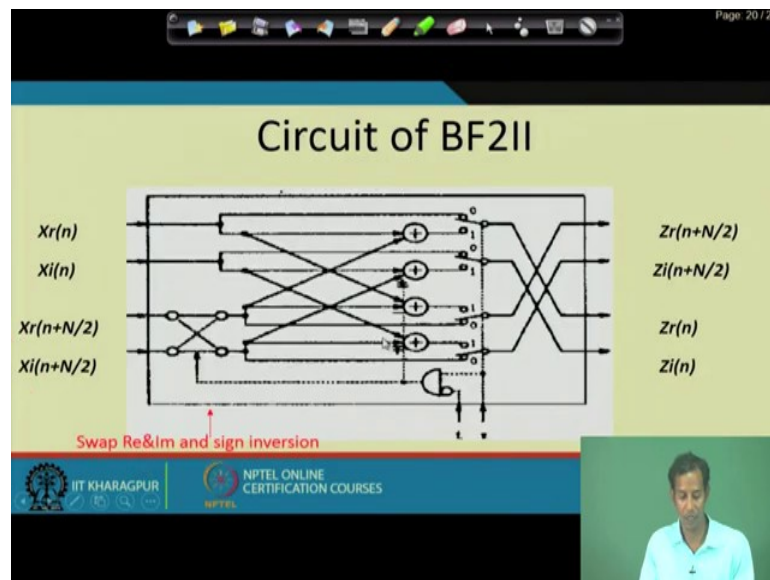
So, then actually this butterfly unit 4 you can just decompose it into butterfly unit 2, 1 and butterfly unit 2 which consists of 2 butterfly 1 unit ok.

(Refer Slide Time: 18:56)



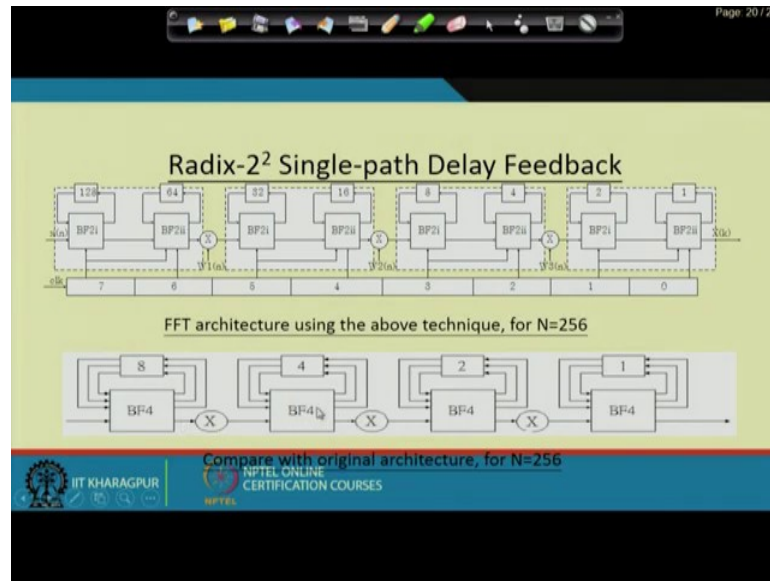
So, this is the corresponding structure of butterfly 2 unit where it takes 4 of this this X_n then; sorry this is the real n this is imaginary and this is the real n plus N by 2; where this is the imaginary n plus N by 2 and it produce the Z_n plus real of n plus N by 2; Z imaginary of n plus N by 2 and Z real of n z imaginary of n ok.

(Refer Slide Time: 19:39)



So, this is the BF2 architecture this is BF2 this is BF1 architecture this is BF2 architecture.

(Refer Slide Time: 19:48)



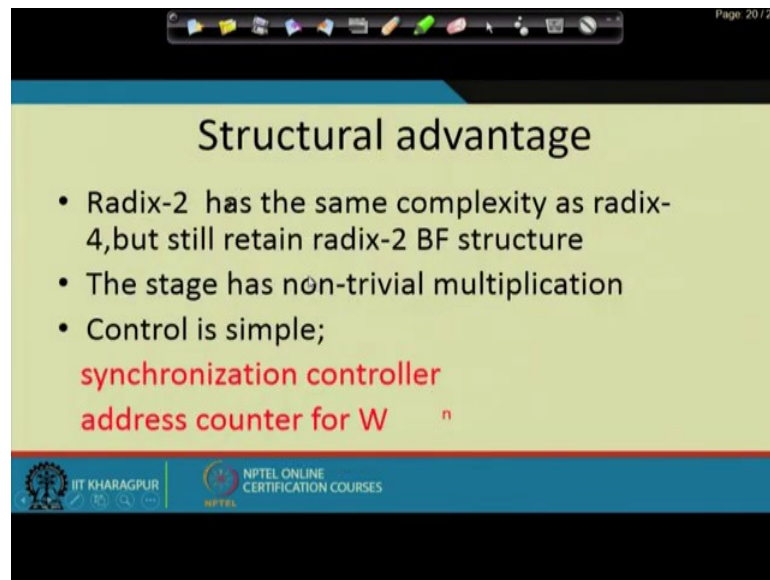
And combining them, now I can implement this. This is Radix-2 square single path delay feedback architecture ok. So, where we have this BF2 I and then we have this BF2 two I considering the above decomposition what we have already used or in the equation we have shown earlier.

So, here for actually in this particular case we have to this is for 256 point FFT. So, in this case we have to run it for 128 times then again in 2 we have to run it for 64 times. Then in the next it will be run for 32, here it will be run for 16. So, all these are basically running in parallel ok.

So, the original architecture for this 256 point FFT which we have already discussed is that it was butterfly unit of 4 it has to run for 8 times. This butterfly in the next stage the butterfly unit 4, has to run for 4 times, then the butterfly unit 4 has to run for 2 times in the third stage.

And in the final stage is it has to run for one number of iteration ok. So, this is this particular this is the newer architecture for implementing the n equals to; that means, 256 point FFT. So, here what is the problem is that as we are considering this Radix-4 butterfly. So, at that time the processing element requirements and the controller part will be much more complicated, but in this case it will be much simpler to implement.

(Refer Slide Time: 21:46)



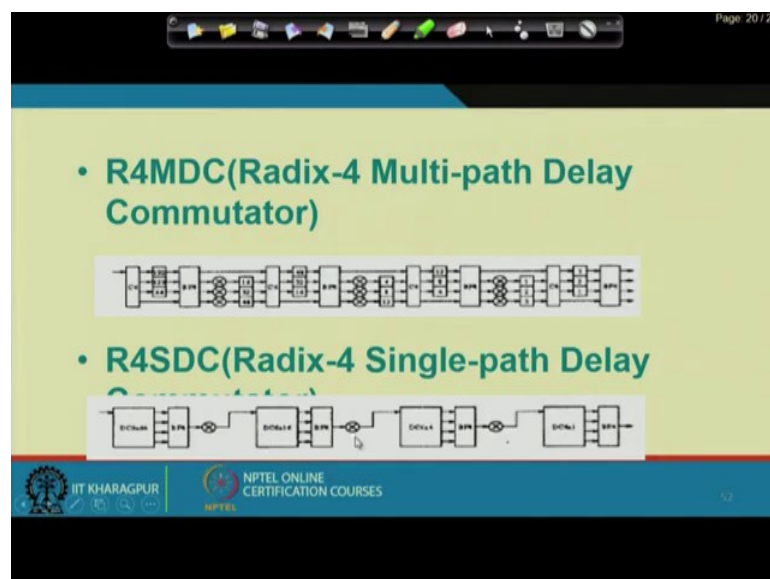
Structural advantage

- Radix-2 has the same complexity as radix-4, but still retain radix-2 BF structure
- The stage has non-trivial multiplication
- Control is simple;
synchronization controller
address counter for W^n

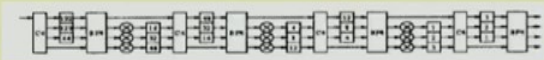
IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, the structure advantage is that the Radix-2 has same complexity as Radix-4, but still retains Radix-2 butterfly structure. The stage has non trivial multiplication the control is also very much simple. So, that the synchronization in the controller and the address counter for this W^n ok. So, that is why in this case the controller as we are considering this butterfly; that means, Radix-4 butterfly unit.

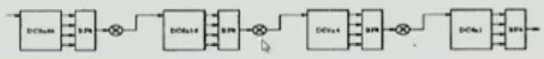
(Refer Slide Time: 22:37)



- R4MDC(Radix-4 Multi-path Delay Commutator)



- R4SDC(Radix-4 Single-path Delay Commutator)



IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, that is why the design that the controller design for this kind of architecture will be more complicated than this one as we are considering only Radix-2 butterfly unit. So, then again this is the actually structure for Radix-4 multipath delay commutator block then we are having this Radix-4 single path delay commutator block. So, here you see as we are having this multi path means we are combining 4 of the paths.

So, here this is single path means only one single path is basically do the multiplication operation is performing after this butterfly by only 1. So, all this here parallel implementation means actually, here also we are following this Radix-4, but the hardware requirement for this particular case will be much higher than this one.

(Refer Slide Time: 23:24)

• **Radix-2 (Radix-2 Single-path Delay Feedback)**

- DIT (bit reverse->normal)
- DIF (normal->bit reverse)

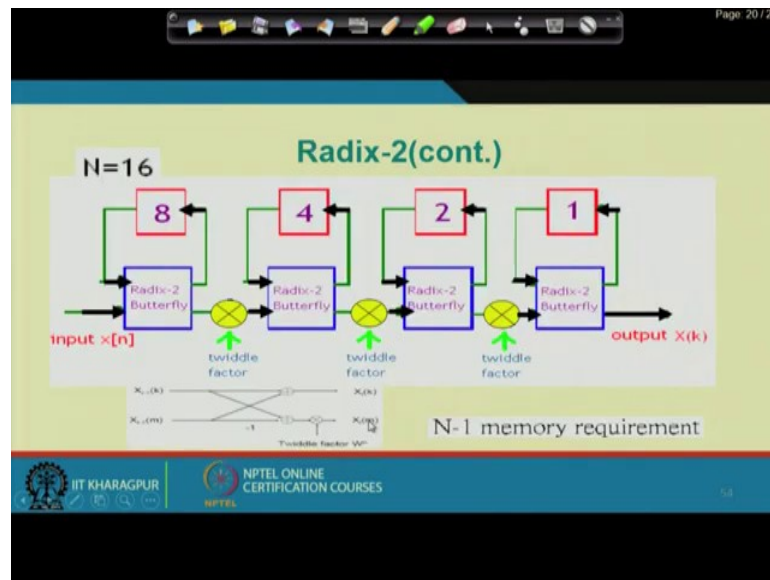
Twiddle factor W^*

Twiddle factor W^*

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, then again if we consider this Radix-2 single path delay feedback. So, as we know that that in DIT it will be in bit reverse to normal, but in decimation in frequency it will be normal to the bit reversed.

(Refer Slide Time: 23:42)

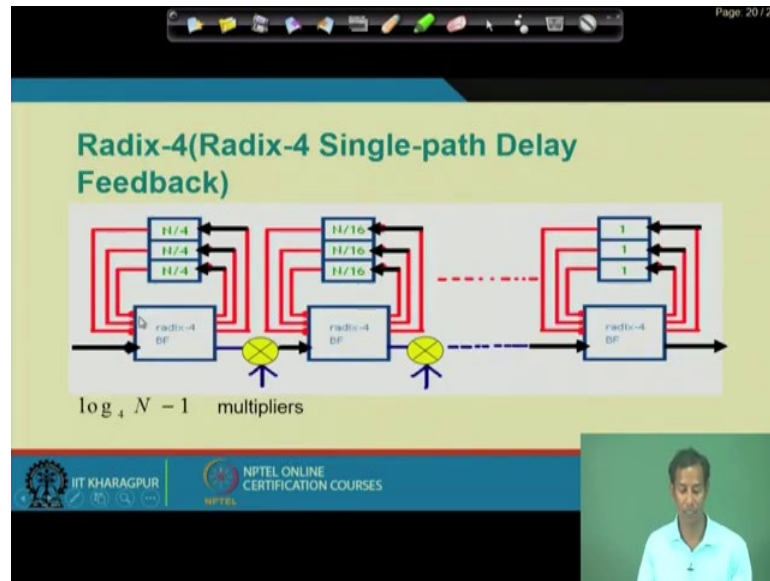


So, then again if we consider this Radix-2 butterfly unit for computation of 16 point FFT. So, at that time the input will be something will be come to the first stage.

And then this Radix-2 butterfly has to run for 8 times then the twiddle factor will be multiplied with each of this, that the unit which will be produced from this butterfly operation. Then again it has to in the second stage it has to run for four times in the third stage it has to run for 2 times and in that; that means, final it has to run for 1 times ok.

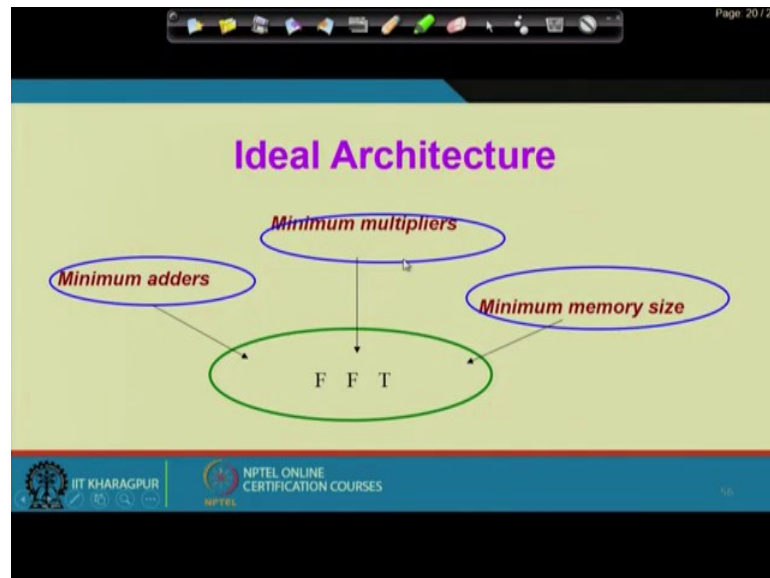
So, this is the basic butterfly structure what we use in this particular processing as the processing element.

(Refer Slide Time: 24:30)



So, in case of this Radix-4 single path delay feedback. So, this is the Radix-4 butterfly then we have to run it for N by 4 times, here N by 16, for the next stage and the final one has to run for only single one. So, here we require $\log_4 N$ minus 1 number of multipliers for this complex multiplication.

(Refer Slide Time: 25:00)

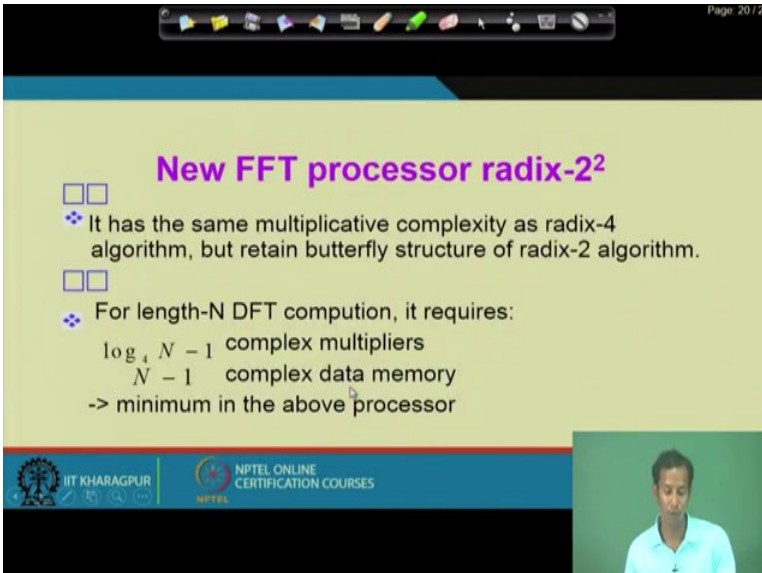


So, the ideal architecture for is that whether I need the minimum number of minimum; that means, the FFT architecture which will require the minimum number of memory size for storing the that means the; that means, the coefficient as well as the intermediate

values the minimum number of adder requirement as well as the minimum number of multipliers requirement.

So, that is why people are basically trying to develop this FFT architecture, considering all these aspects and what I say is that the when the processing elements; we increase the radix and the processing elements if we use at that time the controller will become much more complicated. At that time the memory size either this this will increase the memory size or it will increase the corresponding this multiplier requirement and the addition requirement.

(Refer Slide Time: 26:06)



Page 20/28

New FFT processor radix- 2^2

- It has the same multiplicative complexity as radix-4 algorithm, but retain butterfly structure of radix-2 algorithm.
- For length-N DFT computation, it requires:
 - $\log_4 N - 1$ complex multipliers
 - $N - 1$ complex data memory
 - > minimum in the above processor

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, we have to develop one FFT processors which will be much more; that means, enhanced or if which can use minimum number of this resources as well as minimum number of memory to get the advantage in terms of speed in terms of power and in terms of area ok.

(Refer Slide Time: 26:36)

New FFT processor radix-2²

- ☐☐ It has the same multiplicative complexity as radix-4 algorithm, but retain butterfly structure of radix-2 algorithm.
- ☐☐ For length-N DFT computation, it requires:
 - $\log_4 N - 1$ complex multipliers
 - $N - 1$ complex data memory
 - > minimum in the above processor

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, so the there is another new FFT processor which is this Radix-2 square; that means, and we can also this is not only that you can do this Radix-2, Radix-4 you can also implement Radix-8 algorithm, but; that means, the FFT for this video processing we need the this we need the FFT processors of let us say 8192 point FFT. So, at that time using Radix-2 butterfly unit it consumes too much of cycles to compute the full toss.

So, at that time we have to consider this this we have to consider more number of radix for the processing elements you also can do one thing that is mixed radix architecture is also there; that means, some of the part you can go; that means, compute using; that means, the first stage you can compute using Radix-8 that second stage you can compute using Radix-4 the third stage you can use Radix-2.

So, mixed radix architecture is also there to gain the benefit in terms of low area as well as; that means, a control circuit becomes much more; that means, simpler; as well as you can get or the processing speed in a reasonably higher than the, that means in if instead of using only butterfly Radix-2 butterfly unit ok. So, this is one of this; that means, newer FFT processors where we use this Radix-2 square.

(Refer Slide Time: 28:15)

Page 20/28

Derivation

$$X(k_1 + 2k_2 + 4k_3) = \sum_{n_3=0}^{N-1} \sum_{n_2=0}^1 \sum_{n_1=0}^1 x\left(\frac{N}{2}n_1 + \frac{N}{4}n_2 + n_3\right) * W_N^{\left(\frac{N}{2}n_1 + \frac{N}{4}n_2 + n_3\right) * (k_1 + 2k_2 + 4k_3)}$$

$$X(k_1 + 2k_2 + 4k_3) = \sum_{n_3=0}^{N-1} \left[H(k_1, k_2, n_3) * W_N^{n_3 * (k_1 + 2k_2)} \right] * W_N^{n_3 * 4k_3} \quad \begin{matrix} 0 \leq n_3 \leq \frac{N}{4} - 1 \\ 0 \leq k_3 \leq \frac{N}{4} - 1 \end{matrix}$$

BF1
BF1

$$H(k_1, k_2, n_3) = \left[x(n_3) + (-1)^{k_1} * x\left(n_3 + \frac{N}{2}\right) \right] + (-j)^{(k_1 + 2k_2)} * \left[x\left(n_3 + \frac{N}{4}\right) + (-1)^{k_1} * x\left(n_3 + \frac{3}{4}N\right) \right]$$

BF2

IIT KHARAGPUR
 NPTEL ONLINE CERTIFICATION COURSES

And this is the equation how we do; that means, derive from this equations and then it has to implement like this.

(Refer Slide Time: 28:22)

Page 20/28

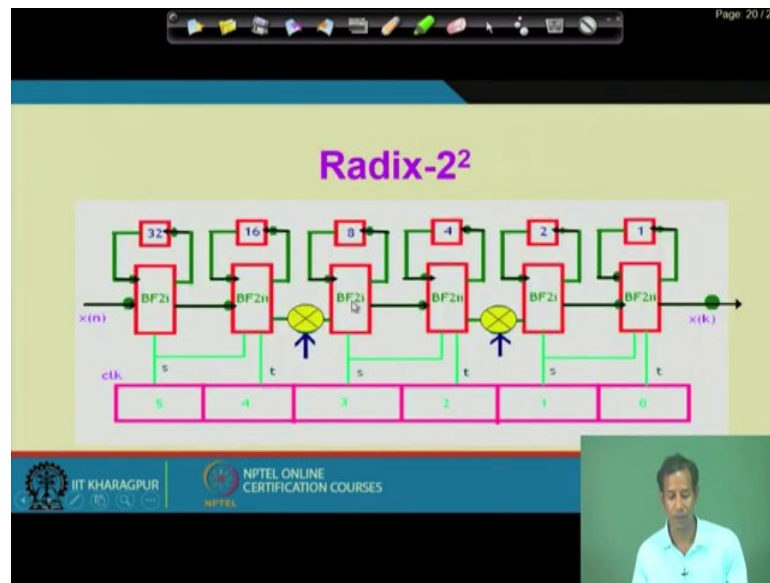
Point Flow

$$H(k_1, k_2, n_3) = \left[x(n_3) + (-1)^{k_1} * x\left(n_3 + \frac{N}{2}\right) \right] + (-j)^{(k_1 + 2k_2)} * \left[x\left(n_3 + \frac{N}{4}\right) + (-1)^{k_1} * x\left(n_3 + \frac{3}{4}N\right) \right]$$

N/A DFT	x(0)
k1=0, k2=0	x(0)
N/A DFT	x(1)
k1=0, k2=1	x(1)
N/A DFT	x(2)
k1=1, k2=0	x(2)
N/A DFT	x(3)
k1=1, k2=1	x(3)

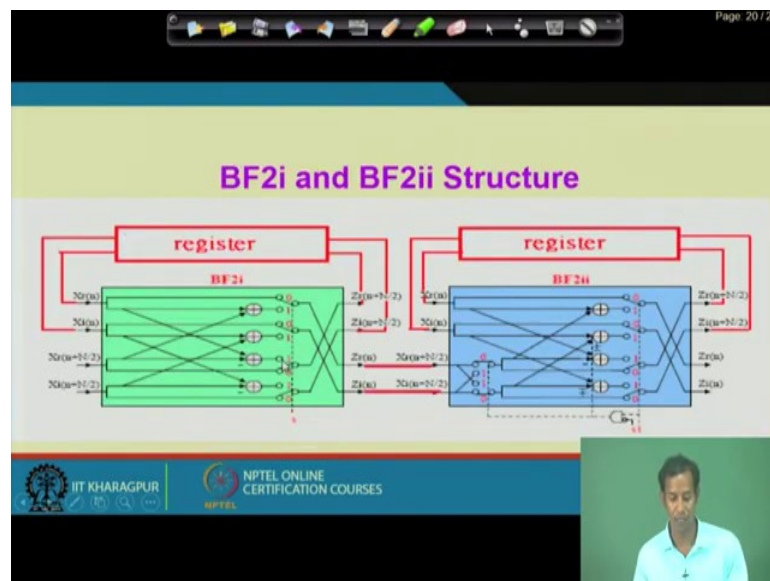
IIT KHARAGPUR
 NPTEL ONLINE CERTIFICATION COURSES

(Refer Slide Time: 28:27)



So, this is the Radix-2 square implementation.

(Refer Slide Time: 28:32)



We have already said; so, in Radix-2 square we are having this butterfly unit 1 and then we have butterfly unit of 2. So, here the intermediate results are being stored and then again it runs in serial.

(Refer Slide Time: 28:50)

Hardware Comparison

	multiplier	Memory size	adder
R2MDC	$2(\log_4 N) - 1$	$\frac{3N}{2} - 2$	$4 \log_4 N$
R2SDF	$2(\log_4 N) - 1$	$N - 1$	$4 \log_4 N$
R4 SDF	$\log_4 N - 1$	$N - 1$	$8 \log_4 N$
R4MDC	$3(\log_4 N) - 1$	$\frac{5N}{2} - 4$	$8 \log_4 N$
R4SDC	$\log_4 N - 1$	$2N - 2$	$3 \log_4 N$
R^2 SDF	$\log_4 N - 1$	$N - 1$	$4 \log_4 N$

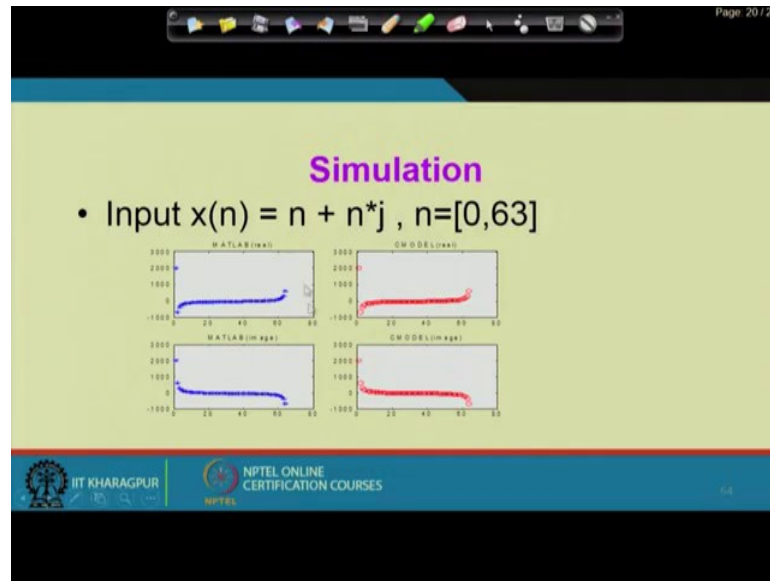
IT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, this is the hardware comparison of different; that means, this is Radix-2 multipath delay commutator block. So, the multiplier requirement is this, memory size requirement is this and adders; that means, requirements this, this is Radix-2 single delay feedback. So, this is Radix-2 square single delay feedback.

So, here you see the minimum number of; that means, multiplier requirement corresponds to Radix-4 single delay feedback, Radix-2 square single delay feedback. So, all are basically $\log_4 N - 1$. Whereas, the memory size for this Radix-2 single delay format, single delay feedback and this Radix-4 single delay feedback is $N - 1$, $N - 1$ this Radix-2 square is a single delay feedback is also the memory requirement is $N - 1$ and; that means, the error requirement is $4 \log_4 N$.

So, if you compare this hardware requirement for this different architecture you find that the Radix-2 square single delay feedback architecture is basically having the advantage in terms of multiplier requirement, the memory size requirement and the adder requirement. So, this is the recent day where; that means, FFT architecture which people have already developed, but it is not that this is the ultimate one still people are trying to use us trying to develop the newer architecture for FFT ok.

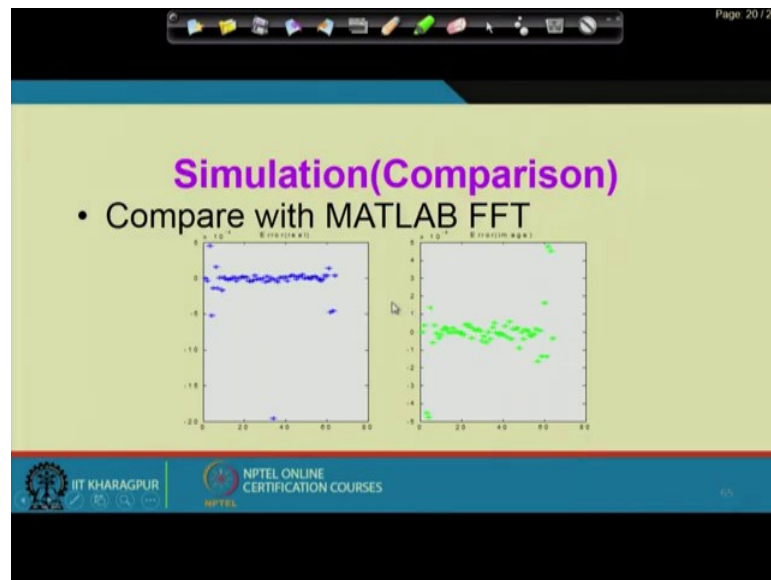
(Refer Slide Time: 30:32)



So, this is the simulation results; that means, you can just write the code in MATLAB and then you can write. As well as you can just do this hardware implementation and then you can run; that means, that how much do; that means, in whenever you are running in MATLAB. So, at that time it is running in the floating point environment whenever you are running in hardware so at that time this is fixed point in environment.

So, depending on this truncation or this overflow, what error or what is the difference between the that actual FFT implementation you are getting in terms of MATLAB or the actually; that means, implementation you are getting in terms of FFT hardware

(Refer Slide Time: 31:18)



So, this is the simulation results and if you just compare this error. So, at that time in MATLAB actually in compared with the MATLAB FFT the error in real part that is in 10 to the power minus 4 , and in case of imaginary part that is also in the power of 10 to the power minus 4 ok. So, whenever we do this FFT architecture at that time initially we have to fix that what should be it is that is what length. So, that this particular this particular error becomes that it will be much more on the lower side.

So, we initially we have to run the algorithm on the MATLAB to fix all these values and then we can go for the hardware implementation using different kind of architecture. And, the ultimate target for designing this hardware hard ware for FFT is that we have to use minimum number of multiplier minimum number of memory requirement memory is requirement and minimum number of addition operations which will give me the benefit in terms of speed power and area.

So, this is it for the FFT architecture if you need to know more then please go through Google scholars find the latest architecture on FFT or if you need more information then please let me know via discussion forum, I will provide you the more information about the recent days FFT architecture for VLSI implementation.

Thank you.