Architectural Design of Digital Integrated Circuits

Prof. Indranil Hatai
School of VLSI Technology
Indian Institute of Engineering Science and Technology, Shibpur, Howrah

Lecture - 25
Pipelining and Parallel Processing

Hello everyone. So, welcome back to the course on Architectural Design of IC's. So, in the last class, we have seen the, that means, circuit for running average system. So that means, we have seen three different types of architectures. So, if we go for the parallel architecture, so that means, parallel implementation of the running average system, so at that time, it gives me some advantage, it has some disadvantage. So, then we have gone for another architecture which is sequential or serial that means, architecture. So, it is also that means, area I can save whereas, in terms of cost oh sorry, in terms of speed.

So, then we have find out; that means, as in parallel implementation, we are getting the speed in terms of area whereas, we are getting in sequentially implementation of that, we are getting that the area in terms of a computation time or in terms of speed ok. So, to get both the benefit of these two particular circuits, so we have developed one combined circuit. And there we can say that we are getting this with minimal that means, increment from the sequential element or sequential architecture, we are getting this computation time, we are also making lesser and the area requirement also becomes lesser ok.

So, today we will start with another that means topic which is very much common in architectural design of IC's. This is one technique, this is not one basically we will discuss about two techniques which is very much useful in architectural design of IC's. They are one is pipelining and another one is parallel processing ok. So, I think most of you have seen this multi core architecture. So, these are nothing but this parallel processing ok.

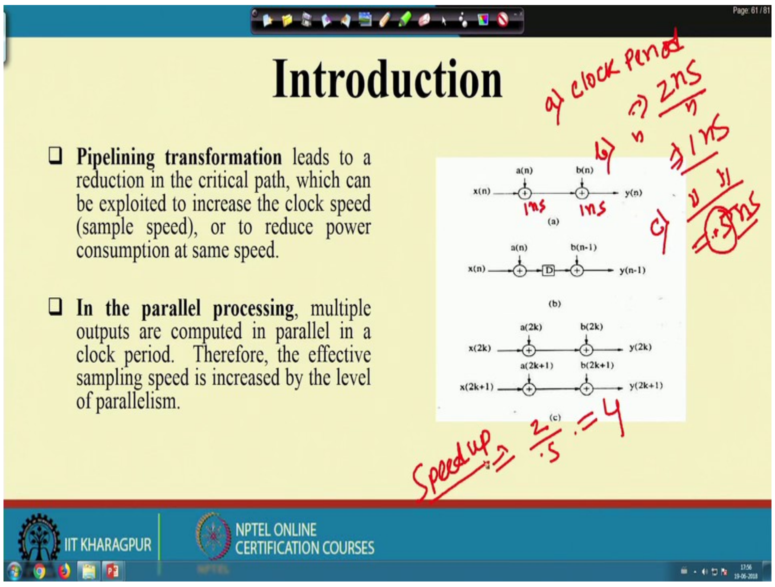
So, to increase the corresponding frequency of the circuit, so in the architecture why I need to learn this two particular like pipelining technique and the parallel processing technique? So, till now what we have seen, suppose we have some algorithm right. So, from that particular point of algorithm now we can choose different type of architectural implementation of that. So, if we have done that architectural implementation, so after doing all this optimization ok. So, whether that is algorithmic optimization or this architectural optimization, the speed is now that is final. So, I cannot go beyond that whatever optimization that means the full optimization of the circuit or the logical expression that we have done ok, but still the requirement of timing that is not meeting.

So, at that time what I will do so that I can increase the corresponding operating frequency of the circuit? So, pipelining is one of the techniques which basically increase the corresponding operative frequency of the circuit. So, we will see that: what is this pipelining technique. And after doing the pipelining, still if it is not meeting the corresponding sampling frequencies to the operating frequency of the circuit, then we can go for the parallel processing too. So, what are the meanings of this pipelining, what are the that means, techniques for doing the pipelining and what is parallel processing that we will see in this particular lecture ok.

So, pipelining transformation says that it leads to a reduction in the critical path which can be exploited to increase the clock speed or to reduce the power consumption at same speed. That means, whenever we use this pipelining technique at that time we can that means, the main fundamental is that once we have done this architectural optimization that means, after doing the algorithmic optimization, we have find out the minimal and after the architectural optimization, we have find out the minimal logical expression for that particular logic or that particular output variables right.

So, I cannot change more than that. So, at that time what I will do, but I need or my requirement of the critical time that is lesser than that, whatever I am achieving after this optimizing, optimization of this architecture as well as the algorithmic optimization.

(Refer Slide Time: 05:04)



Introduction

- **Pipelining transformation** leads to a reduction in the critical path, which can be exploited to increase the clock speed (sample speed), or to reduce power consumption at same speed.
- **In the parallel processing**, multiple outputs are computed in parallel in a clock period. Therefore, the effective sampling speed is increased by the level of parallelism.

So, at that time I have to follow this pipelining technique which can reduce the corresponding critical path by means; that means, it increasing it is increasing the clock speed of the sampling speed. And apart from that it is also reducing the power too. In the same manner, the parallel processing basically what it does it multiples the that means, it copies the number of processing element and therefore, as the processing is basically done in parallel. So, that is why the clock period again I can increase that is more than this pipelining technique ok.

So, it says that in the parallel processing multiple outputs are computed in parallel in a clock period. Therefore, the effective sampling speed is increased by the level of parallelism. So, this means actually in our basic mathematics course, we have learnt these things. Suppose one work I have to do right, so if I can employ five workers to do that job, it takes let us consider one day of time, but if I, that means employ 25 of that means, 25 number of employees or the workers to do the same job.

So, at the time, is that the day will remain same? No, because I am increasing the processing power, I am employing more of the workers means I am increasing the processing powers. So, that means, the computation time will be now reduced. So, here how much time I have increased. So that means, initially it was 5; that means, 5 workers, now it became 25 workers so; that means, 5 level of increment I have done in the processing powers. So that means, the computation time which was one day. So, that will be reduced now by 1 by 5. So, here also the same principle that is works here. That is; that means, whenever we are doing this parallelism that means, suppose the level of parallelism is chosen as n. So; that means, I can increase the clock speed or the sample speed by a factor of n. It is not that only the factor that means, this clock speed or sampling speed it will increase, the corresponding power will also be decreased by the factor of n too.

So, if I use that means, combining this pipelining technique and parallel processing, suppose pipelining level I have used of n and parallel processing level if I used of m level. So, in that in the that means, corresponding if I in one particular circuit whenever we are doing that architecture of that, so at that time both of this technique if we imply or impose on that particular circuit. So, at the time I will get m n, m into n level of improvement in the performance of that particular system ok.

So, how not only the sampling speed; but as well as the corresponding that means, in the power level also I can reduce by power factor of m into n. So, that is how we do and will take one example of that and then we will see how we can reduce that thing ok. So, here you see this is nothing but one that means, one just simple addition operation in chain. Suppose, here what we are doing we are doing an into xn and whenever we are doing this pipelining or parallel processing, at the time what is my intension all the times it should be remembered or it should be in your mind that, the functionality will remain same. It will not change ok. So that means, what we are doing an x n is added then, that is added with b n and we are finally, producing y n.

Now, according to this pipelining technique, what is there that means, in between now this is my critical path this x n to y n or this an to y n any of these two paths is the critical path, maximum of these two which is taking the maximum time for computation, so that is the critical path of this particular circuit, right. So now, I have to reduce this critical path, to optimize the sampling speed of the clock speed I have to reduce or I have to break this path. So, how can I break? So, if I put in between this two, if I put one delay over here or the register over here.

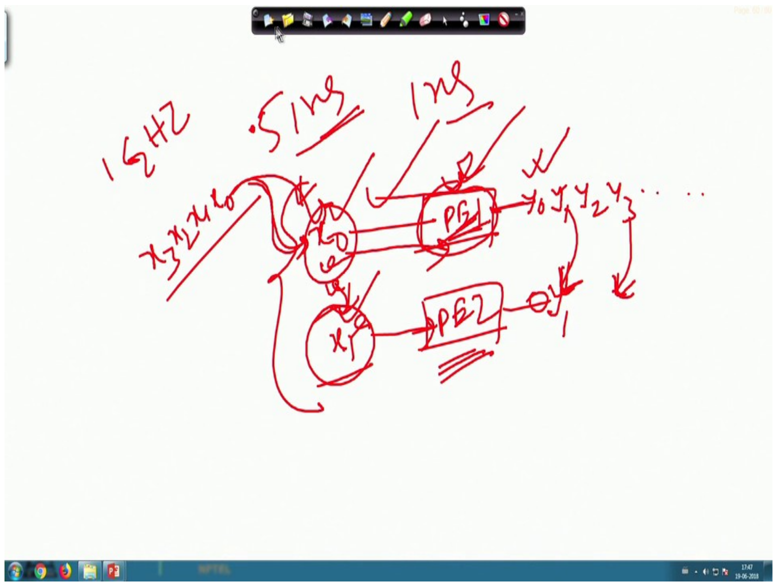
So, at the time these two combinatorial circuits will be act as different. Why different because if as am putting the register, so that means, now the corresponding output from this adder that will be in one clock particular clock that will be stored and then in a next clock it will be added with this. It will not happen in the same clock edge. So that means, now this will act as individual this adder will act as individual of this. So that means, now initially as there were in this particular circuit the delay was two adder delay, but now for these as I have put register in between. So, it the delay of this particular circuit becomes only one adder.

So, that means, I am increasing the operating speed by the factor of 2 ok. So that means, now initially what I require I require two adders. So, here I require only one adder delay. So that means; that means, by 50 percent I have increased the sampling speed. So, if it is it was suppose if it is let us say 1 nanosecond and this is that means, the adder delay is one nanosecond. So, initially the total delay was 2 nanosecond, but now for this particular circuit, it will become only one nanosecond. So that means, if it is 500 megahertz sampling speed. So, here it will become one gigahertz.

So, not actually technically if I say there is not exactly 1 nanosecond because the clock 2 q; that means, the register will also have some delay which will be associated with this. So, that delay also I have to consider for this particular case ok. So that means, it will not be why we are considering because as this in compared to this adder, the delay for this register which is very much nominal. So, that is why we can we can ignore the delay of this register, this d means this is nothing but a register. So, we can that means, ignore those delays and that is why we can say that 1 nanosecond is a delay for this particular 1 nanosecond ok.

Now, what we are doing in the parallel processing? These two particular elements, now I have copied this is one processing element the combination of these, if I say that this is in one processing element. So, if I imply or if I impose two of them, so that means, now the number of samples I can produce two at a time; that means, here I can I can for these particular processing element I can put x 2 k and here I can in that means, put x 2 k plus 1. That means, and I can produce a y 2 k and y 2 k 1 at the same time ok. So that means, if I use two number of process; that means, if I copy the processing element by a factor of 2, again I can increase the sampling speed by the factor of 2 ok, means what basically if I just means what; initially what was happening?

(Refer Slide Time: 13:54)



This x 0, what I have to do this? x 0, x 1, x 2, x 3, the samples are coming like this right. But whenever I have used these two particular processing elements, this is P E 1, this is P E 2, let us consider these. So, at the time what will happen? x 0, I can process over here and x 1, I can process over here. If am having only one processing element, so at that time this particular source will be connected to this. So that means, first it will process x 0, then I will get y 0, then it will start with x 1, then I will produce y 1, then y 2, y 3 something like this it will produce.

But in this case what will happen? So, x 0 will process here and x 1 will be processed here. So, that one is y 1 here, it will come here y 3 will come here; that means, the even samples that will be processed by these if I put the even number of samples at this processing element and this odd number of processing element, if I just want to pass through this P 2. So, at that time, the even number of output I will get sorry odd number of outputs, I will get at this particular output of this processing element block.

So that means, now if my that means, after this processing element, if one gigahertz was the; that means, sampling speed, now I can increase as I need to produce that two of the samples within 1 nanosecond, otherwise what will happen this will remain idle if I just at one if my clock period is now that means, sample clock period is one nanosecond. So, this will come at one nanosecond, then this will come at one nanosecond, but at the time it will just remain idle; one of this particular element that will remain idle. So, the sampling speed I need that will be 0.5, so that within 1 nanosecond, it produce x 0 and x 1 both, so it will take this particular that means, this sample then it will produce y 0 and y one after one nanosecond ok.

So, whenever we are talking this; that means, whenever we are talking about these. So, at that time there are some terms which are very much that means, associated with these particular pipelining and parallel processing things. One is that that latency, what is that latency? Latency is the; that means, the time requirement from applying the input to get the output how much clock period, I need to wait so that is basically known as the latency of that particular circuit. What I said that, after applying the input, after how much clock cycle I have to wait that means, I am getting the output after how much clock cycle that number is the latency for that particular circuit.

So, if I just consider this particular case. So, at this particular case as there is no register in between. So, initially combinatorial circuit means, whenever there is a change in the input immediately, the output basically that reacts or that comes out ok. So, as there is no register in between these two adder path, so whenever I will apply whatever is the delay for that adder, the output will change here. So, again after that again this will be connected. So, this will affect the corresponding y n, but here as I have put the delay in between that means, I will after that means, in one clock after just get the output over here, I have to wait for one clock and after that in the next clock, I will get the y 1.

So, once I got the y 0, after that again immediately I will get the output in a flow. It is something like a, so why this pipelining that means, the terms comes, it is just like if you want to pass the water from a pipe. So, whenever you have connected to the tap. So, at the time whatever is the length of the pipe, so it will take some time from the first flow which is applied in the; that means, the front of that tap to for arriving to the outlet of the pipe. So, it will take some time for traversing the water through the pipe.

So, if I increase the length of the pipe, the more the time I require to that means, to flow the water through the pipe from the top that means, from the tap point. So, here also the same thing, so but the thing is that once I got the output at the; that means, pipes front after that the flow is continuous I do not have to wait. So, initially I have to wait for that particular time and that time is basically very much related to the length of the pipe. So, here also the same thing whatever the level of pipelining, I will introduce into the circuit, so I have to wait for that much of clock period here which is known as the latency for this particular circuit and now the thing is that, now suppose ok, so this is the am I will just discuss that later. So, this is the that means, the latency the definition of latency.

Now, there is another term which is throughput. Throughput means per unit of time how many output I am getting after applying the corresponding input means, what if I have the that means, for this pipelining things what I have done. I have applied this input over here and after that I am getting at the output to which is y n minus sorry n minus 1. So; that means, per unit of time what am getting am applying one and I am getting one output at the end. So that means, the throughput is basically 1 over here.

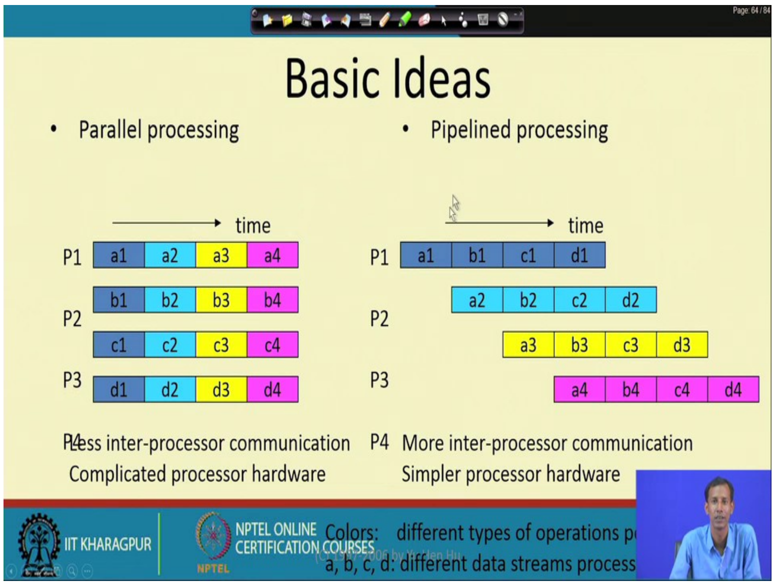
Now if I u that means, if I just copies that processing element as in this case, I have copied that by 2 so; that means, within that unit time I am getting two number of samples at the output which is which have this y 2 k and y 2 k plus 1 ok. So, that is why here the throughput is that the number of output I am getting within the unit time of that that means, within this unit time after applying the inputs. So, that is known as that number is known as the throughput and throughput is very much related to the corresponding this parallel processing. So, what I said latency related to pipelining and throughput is related to parallel processing ok.

So, this two is the two things which I require. And another thing is that that is the speedup ok. So, what is the speedup? Speedup means ah; that means, by what factor I am increasing the sampling speed of the clock speed. So, in this particular case, what is my speedup for the pipelining things what is my speedup. So, initially what I said that there is two adder delays. So, now, I have just come to the one adder delay; that means, the speedup is now 2. So, 2 adder delay divided by 1 adder delay. So that means, the number of speedup is 2, then again as I have employed two parallel processing element.

So that means, again I am increasing the sampling speed now there it becomes the corresponding; that means, the sampling speed; that means, sampling clock period that instead of 1 nanosecond, now it becomes 0.5 nanoseconds. So that means, now I require the corresponding that means, the increasing the speedup by again 2. So, initially in these two particular cases, if this is one nanosecond and this is one nanosecond ok. So, this for this a case the minimum clock period which I require that is 2 nanosecond right but for b case the minimum clock period requirement that becomes 1 nanosecond.

So that means, here am increasing the speed by 2. In the c case the same that means, the clock period will become that will be 0.5 nanosecond. So; that means, I have started with 2 nanosecond after employing pipelining along with parallel processing. So, I have come down to the 0.5. So, the speed is now speedup is now 2 nanosecond divide by 0.5 which is basically 4. So, this is the speedup of this particular circuit ok. So, so this is the that means, fundamentals of pipelining and parallel processing.

(Refer Slide Time: 24:48)



Basic Ideas

- Parallel processing
- Pipelined processing

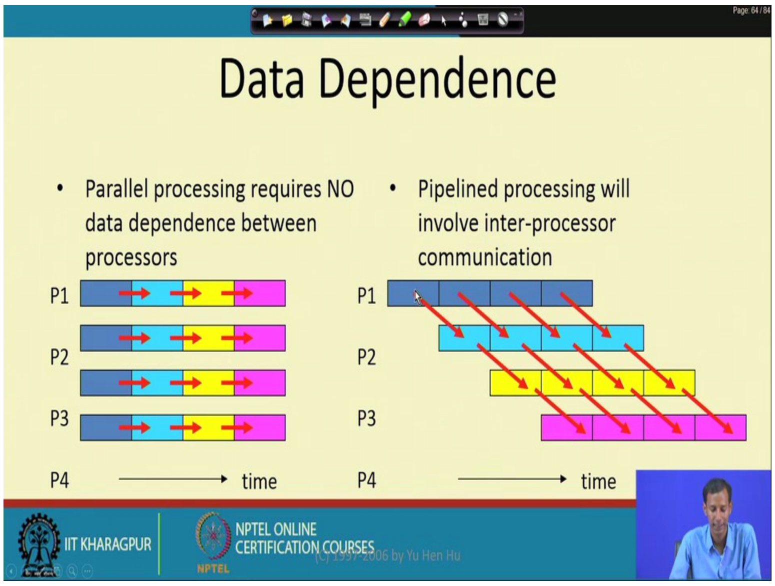
Less inter-processor communication
Complicated processor hardware

More inter-processor communication
Simpler processor hardware

Colors: different types of operations p
a, b, c, d: different data streams process

So, then, so, if I say that this is; that means, here is this is the basic idea. So, in parallel pro in pipelining what we what will done, suppose P 1 is a 1, b 1, c 1 and it passes through the corresponding register. So, now, it will be delayed as it is passed to passes through one that register. So, it will become a 2, b 2, c 2 then d 2; that means, it will be just delayed by 1 clock cycle, then P 3 again it will be delayed again 1 clock cycle. Then again P at the P 4 will be again delayed by another clock cycles, but in parallel processing what happens this a 1 that means, for a it will be processed, for b 2 it again it will be processed, for p 3 again this will be processed; that means, all the processing are running in parallel ok.

(Refer Slide Time: 25:49)



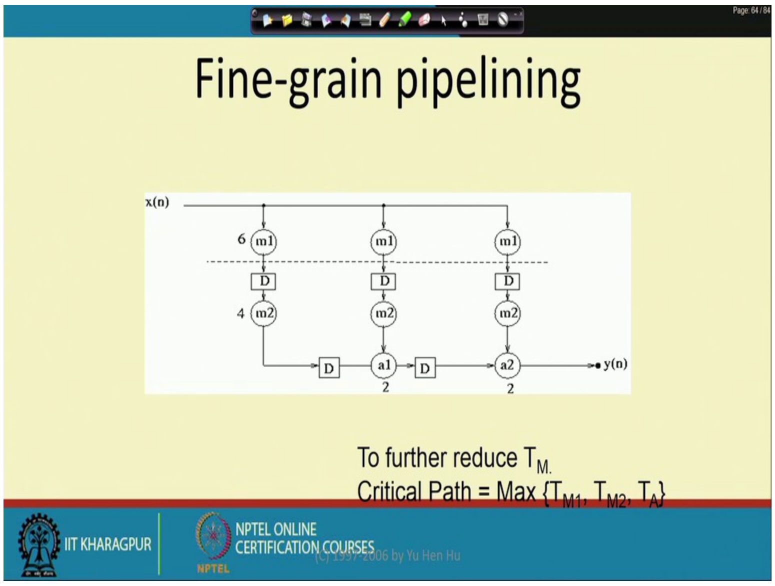
Data Dependence

- Parallel processing requires NO data dependence between processors
- Pipelined processing will involve inter-processor communication

P1
P2
P3
P4
time
P1
P2
P3
P4
time
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
(C) 1997-2006 by Yu Hen Hu

So, this is the basic idea that means, difference or the this is the basic ideas of parallel processing and pipelining ok. So, this is the, that means, the data dependency or the data flow of this particular circuit. So, here you see that data's are basically traversing at this particular point ok. So, they are not dependent on each other, but that means, in this particular case they are interdependent, but in this case in this case they are intradependent; that means, P 1, P 2 is dependent on P 1, P 3 dependent on P 2, P 4 is dependent on P 3. But here P, P 4, P 3, P 2, P 1, all are individual case they are not dependent to each other ok. So, this is the fundamentals of pipelining and parallel processing.

(Refer Slide Time: 26:40)



Fine-grain pipelining

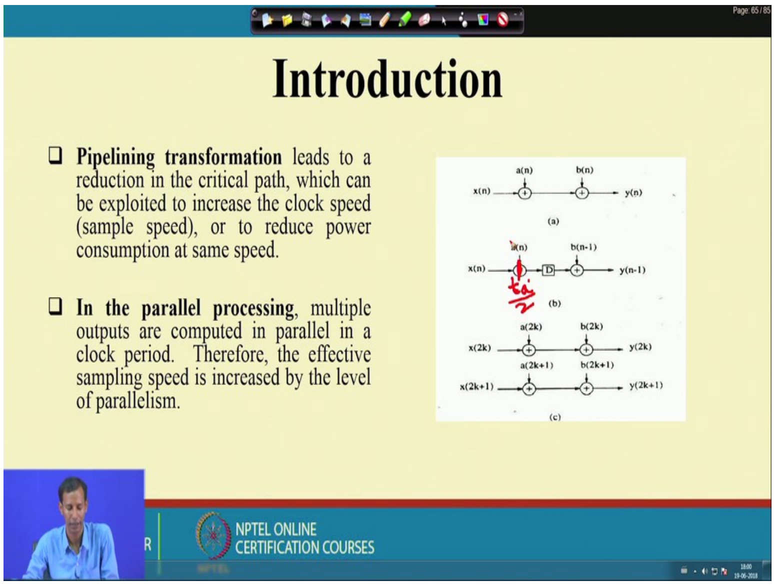
To further reduce T_M.
Critical Path = Max $\{T_{M1}, T_{M2}, T_A\}$

So, again there are that means, two type of that this ah; that means, fine; that means, pipelining technique one is that Coarse-grain pipelining, another one is that Fine-grain pipelining. So that means, whatever we have seen in these particular case, in these particular case that means, this is one adder, this is one adder right in between of these two; that means, the connection level I am putting the register. So, in the operator level they are different acting as two different adders, at that time if I put the register in between path.

So, this is basically known as Coarse-grain pipelining. At that time, what is the limitation that means, I cannot reduce this more than the critical path that is restricted to the delay of this particular adder? So that means, I cannot achieve the clock period lesser than this. So, this is my, that means, the last critical path which of the maximum operating frequency of that circuit which I can achieve.

But I need to increase the corresponding sample speed by more, more number. So, at that time what I will do? If I then if I just break or if I just put register at the operator level that means, if I just break this addition operation into 2 or 3 or 4. So, at that time, again I can reduce the corresponding pipeline.

(Refer Slide Time: 28:38)



Introduction

- Pipelining transformation leads to a reduction in the critical path, which can be exploited to increase the clock speed (sample speed), or to reduce power consumption at same speed.
- In the parallel processing, multiple outputs are computed in parallel in a clock period. Therefore, the effective sampling speed is increased by the level of parallelism.

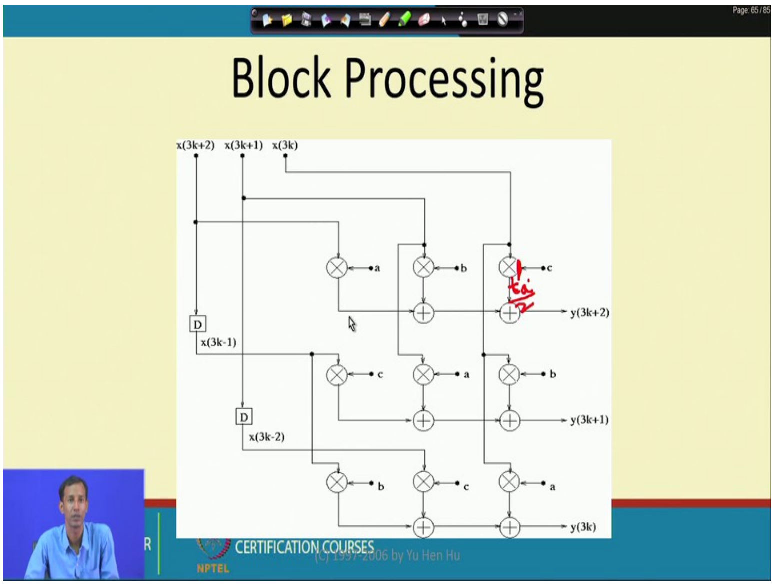
That means reduce the critical path; that means, if I just consider over here, if I consider that means, at this time this is; that means, bounded to this adder delay. Now if I put the register in between of this. So, at the time this will be again become by 2 so; that means, I can increase the operating speed of this by a factor of 2, if I use pipeline at the operator level. So, this particular pipeline technique is known as fine grain pipelining ok. So, if we take the example of that, so at the time you see this is the one multiplier suppose this one is this is the example of one this F IR filtering.

So, here I need multiplier. So, multiplier if it is taking 10 nanosecond, so at the time the delay the critical path for this system that is limited to the adder delay which is 2 and the 10 is the corresponding multiplier delay. So, the total critical path will be if I do not use this corresponding delay in between of this multiplier. So, at the time it will be of 10 here, sorry 10 here and 2 here. So, total of 12 critical that means, nanoseconds critical path. But if I just break this multiplier. So, 6 over at the for m 1 processing and 4 for the m 2 processing; that means, now I am putting this delay for this multiplier block. So, at the time the maximum of this is m 4 plus 2, 6. So, over here is the 6.

So, 4 plus 2, 6, now the critical path will become 6 then what will be the problem; that means, now here I am getting the speedup by again the factor of 2, but the problem is that I am putting three extra register at what cost basically I am getting and putting three extra register along with that the latency, will be increased that is again by a factor of 2.

So, initially if the this 2 was the that means, 2 was latency now it becomes not the factor of 2 that means, I had added one extra latency to the corresponding circuit if I use this fine grain pipelining ok. So, these are the that means, basic idea or basic principle of this pipelining and parallel processing.

(Refer Slide Time: 31:15)



Block Processing

So, this is one a FIR filtering that means, parallel processing architecture of a FIR filtering ok. So, for today, this is it. Next day, we will take one example in the next class, we will go at take one example. And we will see how you can reduce the corresponding, we will start with one particular that means the requirement of the critical path, then we will try to reduce how I can achieve the corresponding critical path by using pipelining and parallel processing.

Thank you.