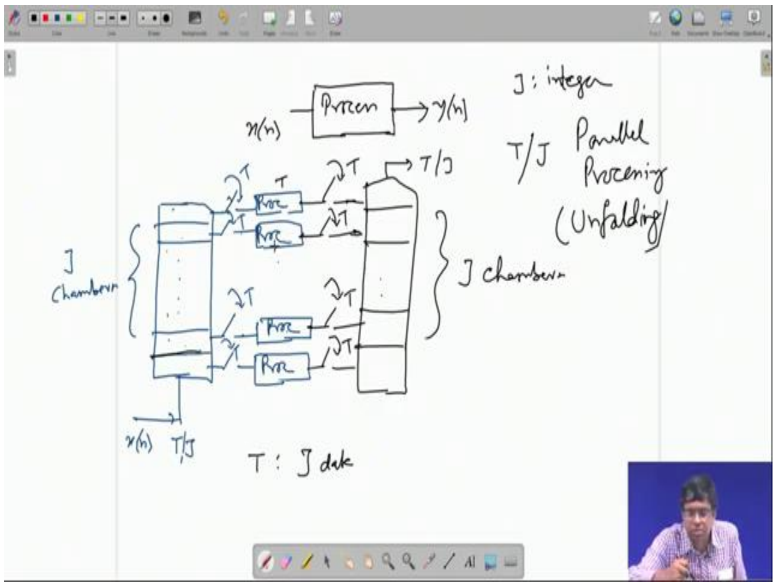
VLSI Signal Processing
Professor Mrityunjoy Chakraborty
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology Kharagpur
Lecture 13
Parallel Processing in DSP by Unfolding

So we start our next topic which is Parallel Processing in Digital Filters. We have already seen that by retiming you can minimize the critical path and therefore the speed of the circuit, the throughput can go up because critical path is the bottleneck, longer the critical path time I mean longer will be the clock, input data clock because you can, until and unless the input job is over through the DFG you cannot give a new data and input job there will be only if you spend critical path amount of time so that determines the clock frequency, input clock frequency. Therefore, by retiming if you can minimize the critical path you can boost up the input rate, lets us find.

(Refer Slide Time: 01:11)



So, suppose there is a process, there is a job or a process like this, this takes input xn produces output yn. This entire thing you have supposed done retiming, you have done retiming and you have minimized the critical path to capital T so capital T unit of time, it could be capital T micro second or nano second whatever, that is the critical path bound and by retiming any further, you cannot improve that critical path, you cannot minimize any further, this is the best you could do suppose.

So, you have retimed it first, you have brought out the critical path of the system to T, which means the best is this, I mean input clock period, I mean you can bring down to the level of capital T but not shorter than that T or in terms of speed, your maximum speed can be 1 by capital T but not more than that, T cannot be made shorter and 1 by T cannot give it higher.

So that is the bottleneck, that by retiming, you could go up to some point, that you can minimize the critical path only to some level, say may be T, so input clock cannot be made shorter than T or a speed wise cannot be, speed cannot be go up above 1 by capital T.

But, suppose I still become greedy, I want to process data still at a faster rate, then how to go about it. For instance suppose I say that let J be an integer, J be an integer. J equal to 1 is very silly. So, say J equal to 2, 3, 4, 5 anything. I say that input clock instead of T let it be, let it be reduced further, 2 by T by J where T by 2 or T by 3 or T by 4 like that, which means speed will go up accordingly, if it is T by 2, speed will go up to 1 by T by 2 that is 2 into 1 by T that is twice. If it is T by 3 it will go up 3 times.

So, I say that I have done retiming and therefore as such I cannot bring down the critical path to below T, and therefore input clock cannot go, clock period cannot go shorter than, cannot become shorter than capital T or clock period or clock rate, input rate cannot go up and above 1 by T but still I am greedy, I say no, I want the input clock to be, clock period to be reduced from T to a fraction. T by 2 or T by 3 or T by in general J, but J is an integer. How to do. Then I say we will go parallel, parallel processing, there is a game for it in DSP it is architectural, it is called unfolding.

The way we go above, this is this, we use a buffer, there is a input buffer, here you give input xn at the first rate, that is the desired rate, that speed, clock period here is, T by J. That is speed is J into 1 by T, J times fast as we want and then let that be J chambers, total J chambers is may be having 16 bit or 32 bit or 24 bit whatever be the data size, is may be one cell like that, another one, another cell like that.

Then what happens, I take lines and this is switch, switch is not fade switch, is done by mux multiplexers, controlled by clock. This process, I copied parallely, one copy, another copy, another copy. So, J times the same process is copied. So, you see we are paying in terms of hardware, earlier whatever hardware was required in the one copy of the process now J times that will be required.

And therefore, power required will go up by J what does the switch do, it closes once in every T amount of time and then again opens up, once it closes the data from the chamber moves to the process, the data from this chamber moves to this process, data from this chamber moves to this process, like wise.

So what happening is this, suppose one new data comes then next cycle it moves here, another data come then they move up, another data come. So, in J cycles J cycles I have how many data, how much time, I mean just a minute, data you see data is coming at a period of T by J therefore, you get period of T, how many cycles come, how many data come, J data because one data takes T by J time, another takes another T by J times, dot dot dot. So J data, J number of data they occupy T amount of time.

So, in one T amount of time I have got one data here, another here, another here another here. So, J data have come J number of data, they are filled up the chamber and if taken the time capital T to fill up this chamber. Once that happens then the switches close, this double this data moves fly over to this process inside, and the switches open again, so the process takes gets the input, from this chambers.

And then it starts processing then how much time it takes, process takes T because its critical path is T, I cannot bring it down, so it is taking T amount of time, so next T amount of time, the this process will work on this input, this process will work on this input, this process will work on this input, this process will work on this input, so on and so forth. Next T amount of time and that next T amount of time another block of new data J number of data will fill up the chamber.

So, previous block of (())(07:47) I moved in, they have gone into the process, now if you look for the next T amount of time that time they, this processors they will process this whatever data they received and during that interval I mean the new data then new data then new data so total J number of new data taking T amount of time will fill up the chamber. Then again they will close, so which will close, they will move here, but in that T amount of time, the processes have completed the job so they will come in the output.

Here again I have got switches that is mux they also close, once in every T amount of time, so now the process output has come, they close and open. So, this data from process output goes into and output buffer. Again J chambers, J chambers so in first T amount of time, chamber is this is filled with new one block of data then they move here, then there are process (())(09:01) in next T amount of time and that time this is filled with another block of data then that moves here, process output moves here.

In the next T amount of time, I will clear this chamber by using a faster clock T by J period, so T by J period for sending out this data then this data this J data, so total I have got J data and T by J is the period by which I have sending out, output clock period. So total T amount of time will be next T amount of time will be used here to clear this buffer.

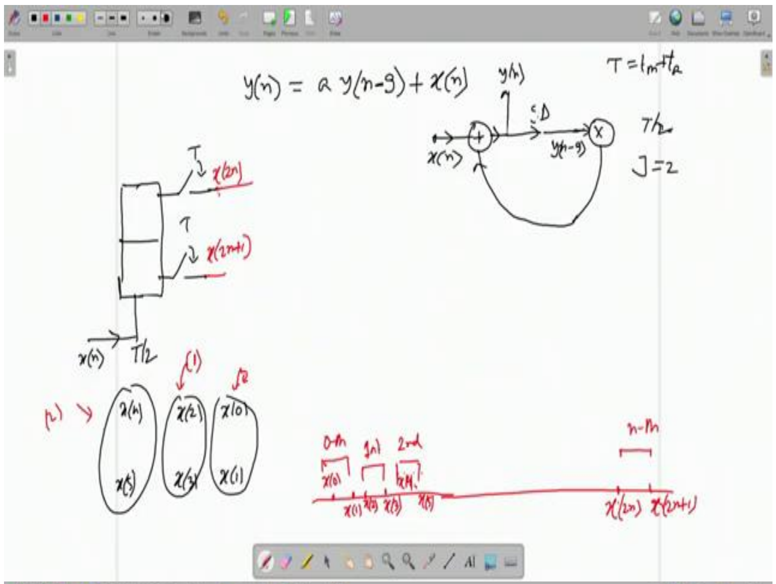
During that time this processes will complete that process, output will be form they will again fill up the, this new this buffer output buffer, once this is clear and another new block will move here, and this will go on.

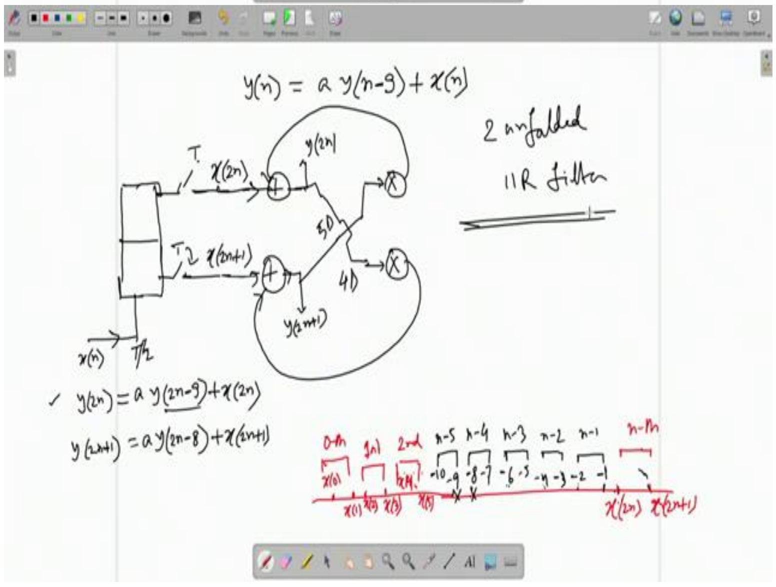
So, input output rate you see, that is high. Because clock period has, input period has gone down to gone up input period. Clock period has gone down from capital T to T by J, output clock period has gone down from T (())(10:00) T by J, suppose input and output they are running at a faster clock, but system is working at a slower clock because if once he gave re T amount of time only new data come and they are processed at a clock of period capital T.

And then again they are moved out to the new buffer, this buffer because the output switches also close, once in every T amount of time. But, filling up this buffer or clearing up the buffer they are working done at the faster rate, period of T by J, so that J data are fade in, in T amount of time J data have fade out in T amount of time.

This is the whole idea of parallel processing which you can do, now in the case of digital filters, I will take an beautiful example of paradies book, I will take an IIR filter in fact, and then apply this concept there to obtain a parallel realization and then many you know in many things which are lying inside and not visible here they will show up.

(Refer Slide Time: 11:09)





So, let us take an IIR filter of this type, instead of n minus 1 I take n minus 9, 9 th order plus xn that is all. So, what is the basic DFG, source node I am not showing, this is xn, this is plus, this plus suppose yn, I am coming here suppose yn minus 9 just a minute, suppose I, the past data so yn minus 9 this has been correctly generated in the past suppose, using that and using xn will generated the current output yn correctly.

Suppose it correctly generated, so it will be multiplied by a built in constant a, there is a built in constant a I am not showing. So, a into yn minus 9 will come up and that has to be added with xn so this that adder, and then this is the output. It is output I take as yn and yn if I delay by 9 cycles so I get yn minus 9 and then the loop is complete.

This is what we have done many times, so this is the basic DFG alright, this is basic DFG. Write down I am not doing any retiming what you guess critical path it was say upstairs this is separated by delay 9 delay. So the 2 times added and multiplier will not adder but if you look at the lower bunch it has no delay.

So first multiplied time Tn then add up time Ta. So, total critical path will be Tn plus Ta. Since here a critical path is a T, which in this case is Tn plus Ta you can retiming and you can bring down one some delay from here to here, so it will be Tm when (())(13:25) change my logic so I am not doing that Tm plus Ta, now we want to, I want to apply input at a faster rate, that is instead of T, I want T by 2, that is J equal to 2, 2 parallel.

So, that means I should have a buffer of just two cells, earlier it was J number of cells, J is 2 here J is 2 here, so input will go in at a rate T by 2, rate means period T by 2 alright, as before there will be switches like this, now you see first x we started is equal to 0, x0 comes in then x0 moves here, x1 moves here, x0 comes in and in T by 2 amount of time, then x0 moves here in next T by 2 amount of time x1 moves in, so in a total time of T x0 here, x1 here then they close here, so I have got x0 here, up stair and x1 down stairs they move in.

Next x2 comes in next T by 2 and then moves here and next T by 2 x3 comes, so x0 followed by x2 followed by x3. So this is, this first goes in then this goes in and like that then again x4 comes, x5 dot dot dot. So you see and in the actual time scale x0 and x1 they come one after another they are not simultaneously coming but because of this buffering they are simultaneously available in the system that is x0 and x1 coming together into the system x2 and x4, x3 coming together into the system x4 and x5 coming together in the system like that.

So that means if you have a time scale and this time scale which I call timing diagram it is very important for doing these jobs, it will be like this. Suppose your x0 is here, data x0 then x1 these two together will move in into one cycle, one system cycle that is x0 here x1 they move into it and only processed by the process is here in one system cycle, system cycle period is capital T it is slower clock input period is T by 2.

There are two clocks always up working input period, input and output clock which are faster T by J here T by 2 system working at I mean as taking data once in every T amount of time and using a clock of T.

So, input clock is slow I mean system clock is slower by a factor of 2 as compare to input clock, so once x0 and x1 are here they will move in and they will be processed by the two processes parallely in one system clock together that particular system clock x0 and x1 will be available together and equally 0 th system clock and 0 th system clock, I have to give it them, so I could get 0 th system clock x0 and x1 they are simultaneously available, 0 th is system clock.

In next system clock first x2 comes in one T by 2 time, it moves here in the next T by 2 time x3 comes, x2, x3 then they move in here. So, in next once this T amount of time is gone after another T amount of time x2 x3 come that means the next system cycle in one system cycle x0 x1 came then the switches opened up next system cycle is after another T amount of time who comes x2 x3, that means x2 x3 that will be system cycle 1, this will be system cycle 2, this is your x2 x3 this is fast or like that.

x0 x1 they are together coming processed in one cycle I call it 0th, means 0s cycle they move in, then after another T amount of time x2 and x3 have filled it up I mean moves in and that is called they are processed in the next cycle and that cycle is called fast, so in th system cycle I have got x0 x1 here, in first system cycle, system cycle look here actually, there system in 0 th system cycle who have come x0, x1 done, processing.

The next system cycle who have come, x2 x3 because they already occupied here, get another next system cycle who comes x3 x4, so 0th who come x0, x1. So x0,x1 together like this, then next cycle who come x2 x3 together, x2 here x3 like this, so this is what 0 I am writing this original time scale fast as 0 comes as far the input data line is constant then x1 comes, then x2 comes then x3 comes and like that.

But, actually x0 and x1 they are fade to the system together in a same clock 0 th system clock, 0 th system clock, then again x2 x3 comes, x2 comes after x1 then x3 after x3 it takes T by 2 time it takes T by 2 time, so total T. And then they move here so the system gets it in the next system clock but together x2 x3 that system clock is called clock number one, system clock one, earlier we got x0 x1 at system clock 0, that is how I name them or number them.

So, in general n th, n th system clock will up who, you see 0 th has x0 x1, first has x2 x3, then you have got, second has x4 x5 alright, so n th will be what, the second means 4 5 first means 2 3, so n th will be 2n, x2n x2 2 n plus 1, just check if n is 0, x0 x1 here, if n is 1 x2 x3 here, if n is 2 x4 x5 here dot dot dot.

So even odd, even point and the next odd point they come together again next even point, next odd point they come together, next even point next odd point they come together, so 2n th point and 2n plus 1th point, they come together in the n th system cycle. Out of which, like here x0 came here x1 came here they moved out, x0 here x1 here, next time who comes, x2 comes to this path x3, x2 here x3 here.

Next time who x4 x5, x4 here x5 here. So in the n th system cycle x2n so, who comes here is x2n in the n th system cycle, and who comes here is x2n plus 1. So x 2n, x 2n plus 1 and therefore at 2 n th point I have to generate y2n at 2n plus 1 th for this I have to generate 2n plus 1 they will be in the output buffer and y2n will move from first then y2n plus 1 will move next, that is how the output sequentially formed, so I am taking x of 2n x of 2n plus 1 parallely.

I generate y of 2n, y of 2n plus 1 parallely and then put them in a buffer and then they will move out at the faster clock by T by 2. So first y2n then y2n plus 1 and so and so forth. So, input output rate will be first as I explain in the previous slide. So, y2n if have to generate, what is y2n, if this equation simply replace n by 2n, y2n is a into y 2n minus 9 plus x2n and y 2n plus 1, replace n by 2n plus 1, 2n plus 1 minus 9, so 2n minus 8 plus x2 n plus 1, first consider this, I want to generate y2n, alright.

What is we have this y2n so, x2n is coming here with that there is an adder so, there will be an adder a into some data a into some data, so there will be a multiplier with a built in constant a that will multiply y2n minus 9, how to generate that, that is a key issue, similarly x2n plus 1 is just coming this is an adder, so it will be adder and it with a into something so the built in multiplier I have generate y2n minus 8.

Suppose here in the first case by hook or crook I am able to generate y2n minus 9, y2n minus 9 I am able to generate. So a times that, that is suppose here by hook or crook some way I am will able to generate this y2n minus 9 so that into a, a is built in if that output I add, result will be y2n, but I am assuming here I have been able to generate y2n minus 9 how, that is the main issue I am coming. Similarly here in the your case suppose I am able to generate y2n minus 8 by hook or crook if I could give it to its input y2n minus 8 that into a will come out with that if I add that with x2n plus 1 then the output will be y2n plus 1.

Now, let us what is here the time scale, what is a target here, 2n minus 9 so if you go one cycle ago, it is 2n I would writing minus 1, minus 1 means actually 2n minus 1. I am not writing 2n minus 1 fully here because it will be two congested, if I keep writing the full expression 2n minus 1, 2n minus 2, 2n minus 3, 2n minus 4. So I am simply writing minus 1 but, in your mind it be clear that actually I making 2n minus 1.

So, it is 2n minus 1, 2n minus 2 and this will be n minus 1 th clock, system clock. So, one system clock ago 2n minus 1 and 2n minus 2, x of 2n minus 1, x of 2n minus 2 that parallely available, x of 2n minus 2, x of 2n minus 1.

But, I am targeting 2n minus 9. So I go further back minus 3, minus 4 and that is at n minus 2 there is 2 cycles ago, system cycles ago x of 2n minus 4, x of 2n minus 3 they are parallely available, but our 2 is to n minus 9, so I go further back n minus 3, so minus 6 minus 5.

So, 3 cycles ago I have 2n minus x of 2n minus 6, 2n minus 5 parallely available. Go further back n minus 4, so I got 8 minus 7. So, 4 cycles ago, I had x of 2n minus 8 here, x of 2n minus 7 here. And then, if I go another cycle earlier to this, n minus 5 earlier minus 10 minus 9.

This is the point of interest to be here because 2n minus 9, so x2n minus 9 means what, how many system cycle ago, 1 2 3 4 5 so if I delay if I delay y2n which is coming here by one system cycle that takes me to hit this place like the even point, even point to one cycle ago, this position, y2n was here one cycle ago here y2n plus 1 here, one cycle ago here.

Even odd, even odd parallel parallel, with another cycle ago this even would go here this odd will go here like that. Now, you see I am landed at what point, minus 9. So this are odd location this are even odd, that means form here if I go back by one cycle, that is the odd output if I delay by one cycle this output I am here 2n plus 1, minus 1.

Then if I go further by one cycle, 2n plus 1 minus 3, if I go back further by one more cycle 2n plus 1 minus 5, then 2n plus 1 minus 7. So, this point y2n plus 1 if I delay by 5 cycle then I am here alright, because this is the odd point, y2n plus 1 if it is delayed by 5 cycle this goes here then here, then here, then here, alright. If x2n is delayed by 5 cycle then it goes here by one cycle then here, here, here, here alright that means this odd point now needs to be delayed by 5 cycle to take me here.

Very simple, 5 cycle is 5 into 10, so 2n plus 1 minus 10, 2n minus 9 because, one system cycle ago means 2 clock input cycle ago. So when you go here basically you are crossing 2 input cycle then only you are going here another 2 input cycle minus 1 to minus 2, minus this way when you are going here 1 1 2. So, every time you move from one system clock to another system clock you are basically going by 2 input clocks here, going back by 2 input clocks.

If you are going, if you are starting from here by 1, then another 1, another 1, another 1 and then landing here, so 5 times you are jumping so 5 into 2, 10 input cycles you are covering, you are going back by the input cycles, faster cycles. So, 2n plus 1 minus10, is 2n minus 9 that is what, I have here. So I delay this fellow by how much, 5 system cycles sorry, that will generate y2n, and y2n plus 1 I need y2n minus 8, so this guide, 2n minus 8, is the even point so if I started x2n go back by 1 system cycle or 2 input cycles I am here. Another system cycle I have minus 4 so, 2 system cycle or 4 input cycle takes me from here to here.

Then another system cycle or 2 input cycle here, then one more I am here, so, 1 2 3 4, 4 system cycle ago, x2n a was here 2n minus 8 that means to generate x2n minus 8 or other y2n minus 8, I take current yn 2n and delay it by 4 cycles alright. So, this is you see at 2 parallel or 2 unfolded IIR filters.

I stop here today, stop here now. In the next term I will take up a more general DFG not just this example and I, show how to unfold the DFG by factor J, and then I will give some fundamental results, using those result you can come back to this apply the results here, you will get the same structure, thank you very much.