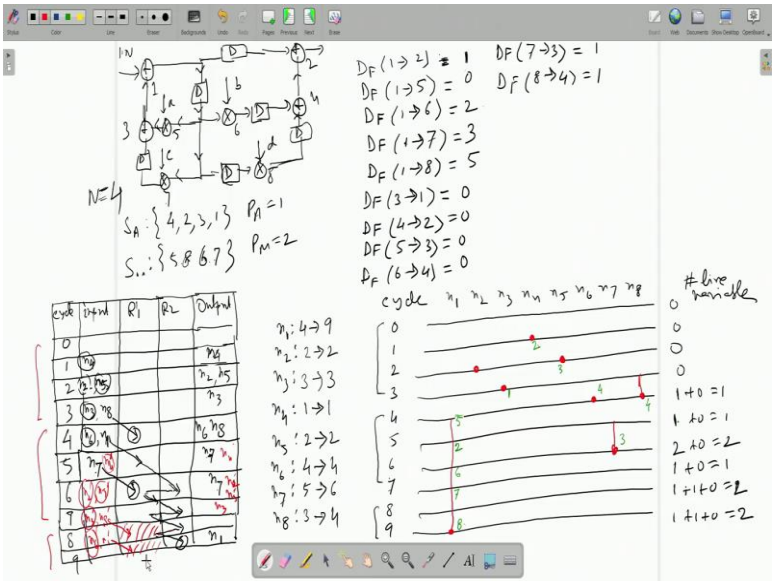
VLSI Signal Processing
Professor. Mrityunjoy Chakraborty
Department of Electronics and
Electrical Communication Engineering
Indian Institute of Technology, Kharagpur
Lecture: 27
Life Time Analysis of Storage Variables in a Digital Filter

(Refer Slide Time: 0:27)



So, in the last class we are considering the delay optivation for this digital filter which we have studied earlier in the context of general folding there is the folding the multipliers and adders. So, all these figures were known to us we opt out this DF u comma v, u arrow v that is the folded delay expressions.

This derived given any support that is folded by 4 times, so system is operating at a faster clock, 4 times the input clock. And these are the schedules pipeline latches, these figures are there. Now, first we have to do the lifetime analysis and lifetime chart. Lifetime analysis means the rest is the variables adder 1 output. This adder 1, schedule is have been 3, that is 4 l plus 3, that time it takes data as input and after adding with whatever is coming from the other end other other input type.

There is a latency here because there is a pipeline latch of 1, so output comes at cycle 4. So, n1 starts at 4, it is born at 4, and it is required here you see 1 to 2, I need 1 delay. So, 4 to 5, then 1 to 5, no delay, so 4 to 4, all right, 1 to 6, 2 delay, so 4 to 6, 4 to 4 plus 2, that is 6, 1 to 7, 3 delay, so cycle 4, so 4 plus 3, 7, 4 to 7 and 1 to 8 5. The recycle from born, so it will be taken up to 4 plus 5 9, 9 is the maximum figure that is when you have shown here.

We have shown here this thing going up to 9, 4 to 9 but remember 1 thing, that in cycle 4 when it is born, that time only it is used with 0 delay, means it goes to 5 5 means multiplier, it by hardware, we adder output, there will be 1 adder output, 1 adder the other output will go with 0 delay there is n1 goes to 0 delay to multiply input.

And it will be born in cycle 4. Then it will correspond to other 1 output because adder 1 schedule is 3, 3 plus 1 internal delay, 5 times delay by 4 in cycle 4 it will be born. Since 1 to 5 is 0 delay, same cycle it will be used at the input of 5 that is, multiplier. So here, just for our convenience I write n1. So, not n1 it will, I write as 5. 5 that means n1 is born at the output of adder 1 at cycle 4 adder at cycle 4 adder unit and in the same cycle it will go to the input up 5.

What is the device 5? 5 is a multiplier, 0 delay that is the same cycle 4 and 4, 4 to 4, there is 1 to 2, 1 delay. So, if the adder 1 output is born in cycle 4, it will be used up at the input of adder 2 at cycle 5. So, I write 2 here, I write 2 , I write 2, device 2. So, it will be used up at this cycle. At the exact what cycle, cycle 5 cycle 4 it is born 1 cycle delay, so 5 and what does it go to 2 that means adder 2 input.

Because you have drawn a vertical line, but what is interesting to see is that I mean during this journey itself and various cycles, I will take this and give it to various either adders or multipliers, those things should be visible here. For me to draw the final circuit you know I mean, when you become a custom create, you do not have to write those figures just for our convenience, I was writing.

Then again, 1 to 6 2 delay, so cycle 4 it is born, so it will be given to 6 6 is a multiplier, so multiplier input at 4 cycle 4 plus 2 6, that will be cycle 6, it will go to 6, multiply 6 1 to 6, 2 delay born at cycle 4, so 4 plus 2 6 in cycle 6 it will go to device, what device, device 6, so multiply 6. 1 to 7 3, so born in cycle 4 and 3 cycle delay to 7. So in cycle 7, it will go to device 7 also, which is multiplier and then 1 to 8, 5 so cycle 4 born and 4 plus 5, 9.

So, in cycle 9 it will go to the input of multiplier 8, the cycle 9 it will go to multiplier 8, these are just writing for our convenience. And then here n2, n2 is the output which is just does does not come back to the circuits or forget about n2, it is born in the schedule of 2 is 1, so 1 plus 1 internally the 2, so it is born n2 is born in cycle 2 and consumed because it is sent out it does not come back to the circuit, so 2.

The n3, n3, 3 schedule here is 2, so 2 plus 1 pipelining 3, so it is born in cycle 3 and 3 to 1 it goes to 1 only and 0. So, 3 born (())(5:55) 3, 3, but it goes to adder 1 that is why, here I write 1 that is, n3 will go to 1 and just for our convenience actually when you get use to it, you never draw it because that is not the custom just for our rough practice we are doing it there is n4 n4 n4 4 is a adder if say rule is 0 0 means, it is bound at cycle 1 0 plus 1 delay 1 internal delay is 1 and 4 to 2 is only line here, 0 delay.

So, it will be born in cycle 1 consume in cycle 1 and, what does it go to adder 2, so I write 2 in 5 is a multiplier 5 schedule is 0. There are 2 internal delays pipelining delays so, 0 plus 2, 2 so it is born in cycle 2 and 5 occurs only once 5 to 3 0 so also can you be in the same cycle. But it goes to 3, so I write 3 then 6, 6 is a multiplier at 6 schedule is 2 and 2 more delays internally, so 2 plus 2 4 in cycle 4 it is born. And 6 occurs only once 6 to 4, 0 delay. So cycle 4 born, cycle 4 dead, but it goes to where 6 goes to 4 adder 4.

So, I write 4. Then 7, 7 also occurs 1, 7 to 3 1, but 7 schedule is 3, 3 plus internal digit 2, so 3 plus 2 5. In cycle 5, 7 is born, and 7 to 3 1 delay, so it is consume is the next cycle during the next cycle, then cycle 6, and that time, what does it go to adder 3, so I write adder 3. And lastly, 8 n8 8 schedule is 1, 2 delays internally. So, 1 plus 2 3, n8 is born in cycle 3. Only 1 delay, so it will consume in the next cycle, dead the next is a cycle 4 and 8 goes to 4. So this is 4.

Now, last class, I did a counting of the number of like variables. So, you take just because there are periods like period is 4, because every 4 internal cycle another new set of data come, new set of variables are generated and so you have to take multiple occurrence, you know, 1 period that goes like this another period starts at 4 goes down for the, the other period starts at 8 goes further down and all that but as I told you, you do not have to count on the repetition cycles from 1 only going by the backward calculation you can find out the number of variables that are alive.

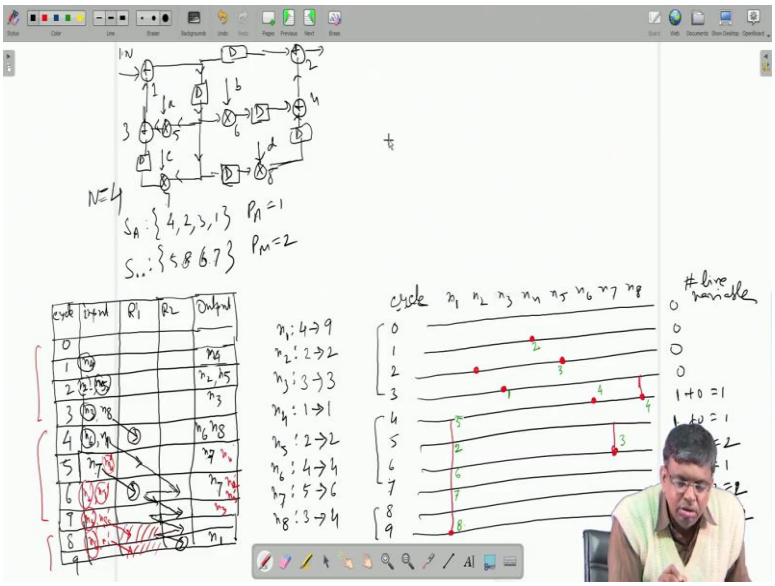
So, for that he had to start 0 here only born and dead 0 born and dead 0 born and dead, born and also born here, here born and dead and born here to 0. Here, this born and dead, no problem, but here I need 1. So, 1 forget about this 1. Then here again. This was only born here I need for this, not for this 1 again. Then here 1 1 2 then here, 1, here, 1 here 1. Now, you consider this 1, actually this 1 and then go 4 cycles back, 1 plus 0 actually this entire thing starts at here.

So, at this 1, you will get this 0 coming here. So,1 plus 0 1. So, we like this 1 will go up by 4, 1 plus 0 1. Next is to go back by 4 through 6, you go to 2, 2 plus 0 2. And then 7, line 7, it is 1, again, go back by 4 over here, so 1. Then comes this 1, you go back by 4 plus here you have 1 and another 4, 0, so 1 plus 1 plus 0 2. And this 1 also go back by 4 here, 1, another 4 here, 0, so 2.

Because when you come down to this you have to consider overlap of 2 blocks, this block another block and the 1 that starts at 8. So, 3 blocks, that is where you get when you are going back and calculating going back twice from here to here and in further. But anyway, we get the magic figure 2, what is the 2 that in any cycle maximum number of live variables is 2, that is we need to store maximum number of variables, how many per cycle 2.

Which means 2 registers will be enough for us, now how to do that, how to use the registers that we will see and we already discussed 1 method called forward backward data location. So, they are used 2 registers R1 and R2. Okay, and I go from column, cycle 0 to cycle 9.

(Refer Slide Time: 10:47)



Let us look at this chart. First variable that gets generated is n4. It is generating cycle 1 but consumed there only. So, n4 a new stop, but remember where does it go to n4 it goes to cycle 2, adder 2 device 2 this I have to you know I mean use when I draw the final hardware circuit, there n4 which comes from 1 adder, what does it go to? It should go to the input of adder 2 at device 2 which is again adder, so it will go back to the input of adder itself, then cycle 2, 2 fellows are born and dead, n2 and n5, no need to put a circle here.

So, here I have got n2 n5, n2 n5, that is in cycle 2, we will write it more clearly, then cycle 3, cycle 3 who is born n3, and n8, and then it continues. So, n3 which is born and dead, so n3 here, comma n8, n8 continues so it will go to the input of R1, 1 cycle n8, and cycle 4 is over, it is consumed. So, here it is over so n8. There were cycle 3, n8 was born. Now cycle 4, cycle 4 3 fellow, I mean, these 2 fellows are born, n6 which is born and consumed.

And who is who else cycle 4, n1, n1 is the fellow who has the longest journey and n1 will go to go down so it will go to now, so n8 was given to R1 input cycle 3 it came to output upon 1 in cycle 4 that time, n1 is given to the input of R1, so n1 comes to the output of R1 in the next cycle, cycle 5 it is connected, but it can go further because the journey is long it goes up to cycle 8.

So, it is only cycle 6 it goes here. I have to go for that but then there is no further register but do not worry, too is the figure. Magic figure, it says that we 2 registers I can do all storage into recovery from subsequent blocks. So I do not I will not be running sort of registers. I just for temporarily stop here.

As I do, another example last time, so these my n1, then. So, there are cycle 4 in cycle 5 who is born, n7 born, and consumed, Is it consumed? No, it is not consumed. If cycle 5 n7 is born, it goes further. So, that was a mistake n7 and it goes to cycle 6 and then gets consumed. So, it will go like this obviously, n1 was given to R1 input in cycle 4 in cycle 5 that goes to output of R1 and so, input of R1 is free, n7 goes there, n7 goes to the output of R1 in cycle 6, no further storage required, so this is your n7.

Now, you see, I got this fellow n6, I can bring it here because R1 output is consumed who is consumed n7, no need to give it to R2 input. So, I take this n1 which is standing at R2 output here, I bring it back to the input of R2 and move it to output of R2. So, I go to 7 but I am still have to I have 1 more cycle. If 7 8 9, I have 2 more cycles alright, so I go to 6 to 7, 7 to 8, 8 to 9, so again I bring it back here from R2 output to the input R2 it comes to the output of R2 in cycle 8, again I bring it back to the input of R2 it comes here and then it is consumed.

So n1, remember during this journey, like in this cycle, in this cycle here here or here, it is stamped and it is used else where, here it is used in cycle 2 in device 2 here it is used in device 5, this multiply 5, so on and so forth. So, you do not have to just look at the final usage, intermediate usage also important in drawing the final circuit.

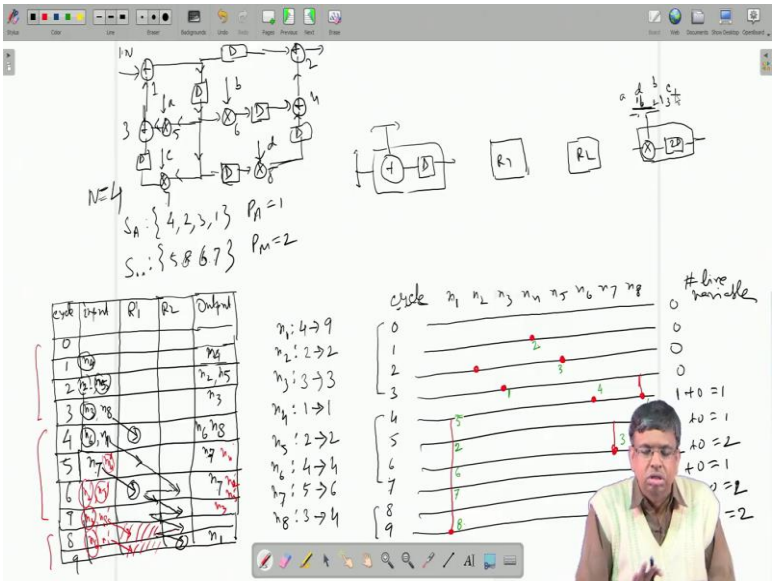
Now, you see, this is periodic it is periodic, so here 0 1 2 3, 1 period 4 5 6 7, 1 period and here another period starts, so 4 is 0 of the new period that time is nothing that 1 is equal to 5, so this time n4 of the new fellow, new block, n4 so I call it is n4 prime, so this will go here n4 prime, so you know overlap I mean it does not interfere, was born and dead and that is because I arrived at the figure 2, by these chart, so I will never (())(19:36) in shortage of any register.

Then 6 basic only 2, that I mean 2 and 5, so that it will be n2 prime, n5 prime both are consumed. So, I will have n2 prime n5 prime, I am not bothered about the red colored figures that 7 beats equivalent to 3, n3, so, there will be n3 prime if it will come here, and there will be n8, so covalently n8 prime, this will go 1 cycle here.

So, will go to the output of R1 and this is coming to the input of R2. So, they are not interfering. So, I that is why has this, that is why I did not bring it to this, I could have brought it here, or I could have brought it here I am not, because this register have to leave, keep free for this red color variables. Similarly your next 1 is when you cycle 8 equivalent to cycle 4 that was n6 and n1.

So, now it will be n6 prime n1 prime, n6 prime is consumed, n1 prime will start moving, so again, like that. So you see, there is no interference. No, there is no conflict. I am happily able to do the allocation to register. So, this is the filter we have to follow. Now I have to draw the circuit. The circuit is very complicated, this entire example is given in Perry's book, this gets erased, unfortunately, we will have to get a n1 n2 n3 n4, like that.

(Refer Slide time: 21:45)



How to do the circuit, 4 adders to replace by 1 adder. So I draw that adder, internal delay, this is 1 input other input, inputs are switched with the adder. What R1 R2 and then I have got the multiplier to delay 1 is built in, this is built in stuff, will be coefficients ABCD then draw, these switched.

Consider n1, n1 goes to, let us start with n1 or something else, let us see, in order to be fine but before that let me see these multiplier ABCD, A is for multiplier 5, 5 as a schedule 0. So the built in thing, this is as we put, I think I have to re draw it, this is not very convenient here. This is my adder, input is switched, this input is switch this adder output, this is my R1 the multiplier, the output, the 4 inputs, a, a is for multiplier 5, multiplier schedule is 0.

So, it will be touching it at 0 at 0. Then b b 6 6 is 2, so who is number 1? Number 1 is 8. So d, 8 is d that means, d will be here this is a cycle 1. Then b, b 6, 6 equal to, so this is a 2 and this is c, c get 7 7 is 3, so a d b c. Since the time is up for this half. I will continue with this in the next class, this has to be drawn is a pretty elaborate affair. So, thank you very much for this.