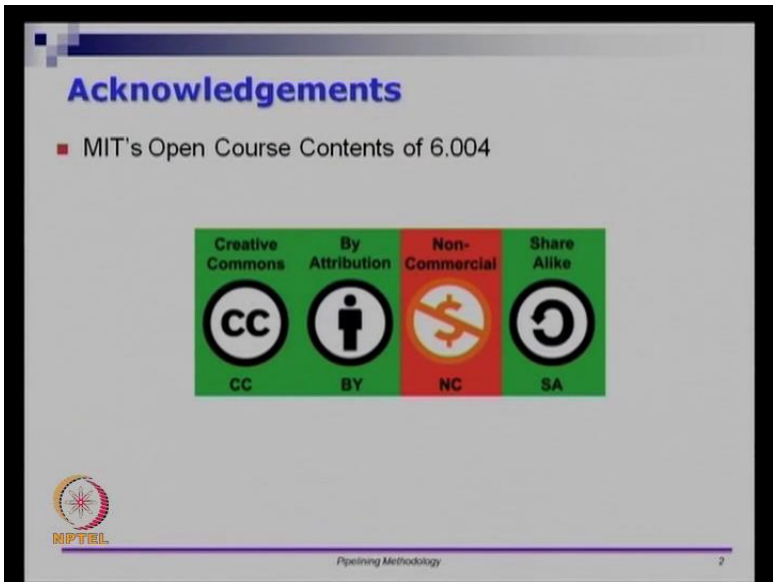
Digital Circuits and Systems
Prof. Shankar Balachandran
Department of Electrical Engineering,
Indian Institute of Technology, Bombay
And
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Module - 44
Pipelining Terminology plus Methodology

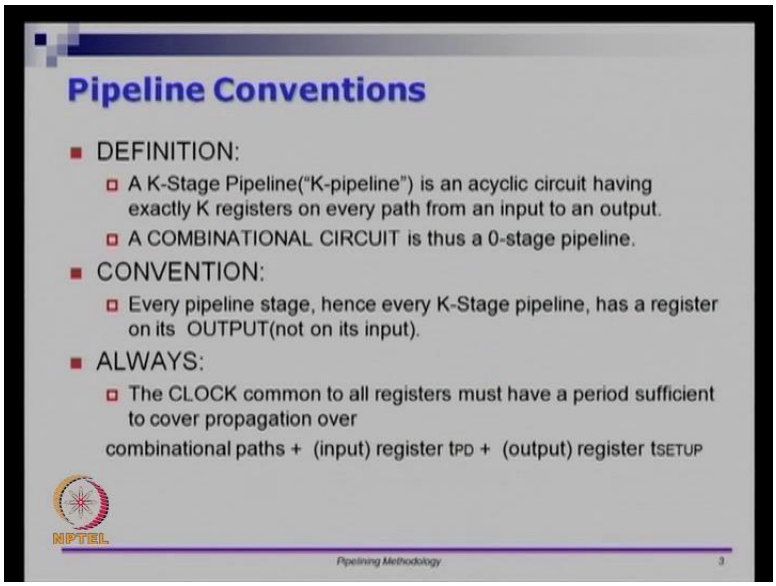
Welcome to module 44. So in this module, we are going to look at some terminology related to pipelining and we will also learn, how to - methodology for pipelining circuits. So, if I give you a circuit, how do we go about pipelining that, we will learn the pipelining methodology.

(Refer Slide Time: 00:34)



So, again as before I acknowledge MIT's open course ware of 6.004 course for the material for this.

(Refer Slide Time: 00:40)



So, let us look at pipeline conventions. So, some of the definitions, so in the previous lecture, we looked at two definitions, namely latency and throughput. In this lecture, we are going to learn another convention or a definition, so we are going to call something a K stage pipeline. So, a K stage pipeline is an acyclic circuit, which means there is no path coming from output of something to the input of the same circuit itself.

So, K stage pipeline is an acyclic circuit, so you may have some combinational logic, you may have some registers, you have combinational logic, you have registers, you have combinational logic, you have registers and so on. You do not have anything feeding back at any point of time, so that is what acyclic means. So, a K stage pipeline is an acyclic circuit, which has exactly K flip flops and every path from input to the output. So, this is a careful thing, so it has exactly K registers, so this exactly is a key thing.

So, any input that is given from the external world, goes through exactly so many stages or exactly K flip flops to go to the output. So, if you use this definition a combinational circuit is also a zero stage pipeline, because a combinational circuit does not have any flip flops at all, so it is a zero stage pipelines. So, let us now look at a convention, so every pipeline stage means after every combinational block, if I put a register; that is a pipeline stage. So, every pipeline stage and hence, every K stage pipeline has a register on its output.

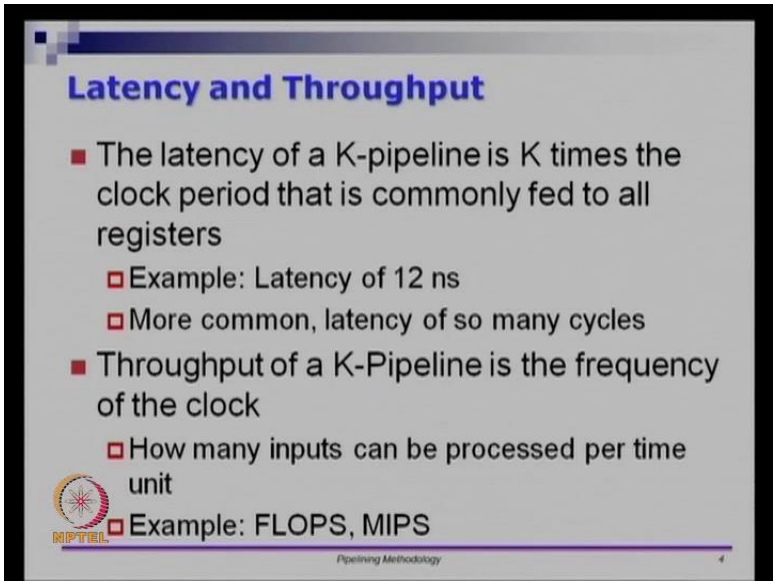
So, what I means is, when I come from the external world, I have some input and I go

through some combinational logic, flops, combinational logic, flops and so on, the last stage will not come out of combinational logic. It will insert come out of a flip flop at the end. So, the outputs are always registered in a K stage pipeline. And the other convention that we have is, there is a clock that is common to all the registers and it must have a period that is sufficient to cover the propagation over the worst case.

So, I have a single clock that is going to supply to all the flip flops and the single clock should accommodate or the time period of this clock should accommodate for the worst case path across all the stages. So, what is a worst case path that we can have? We should look at combinational paths, plus the input registers propagation delay, plus the output registers set up time.

So, we already looked at this, so this is the clock to queue delay, plus the logic delay, plus the set up time, so that is the worst case time period. We already talked about this in week 6, when we talked about sequential circuits; it is a same thing here. So, the pipeline circuit is a sequential circuit, except that, it has the special structure that there is no feedback path from one pipeline register to, so there is no feedback path in the circuit itself, it is acyclic. So, the combinational paths plus the clock to queue delay plus the set up delay, across all the different stages of the pipeline, you take the worst case and your clock period must be wider than that.

(Refer Slide Time: 04:08)

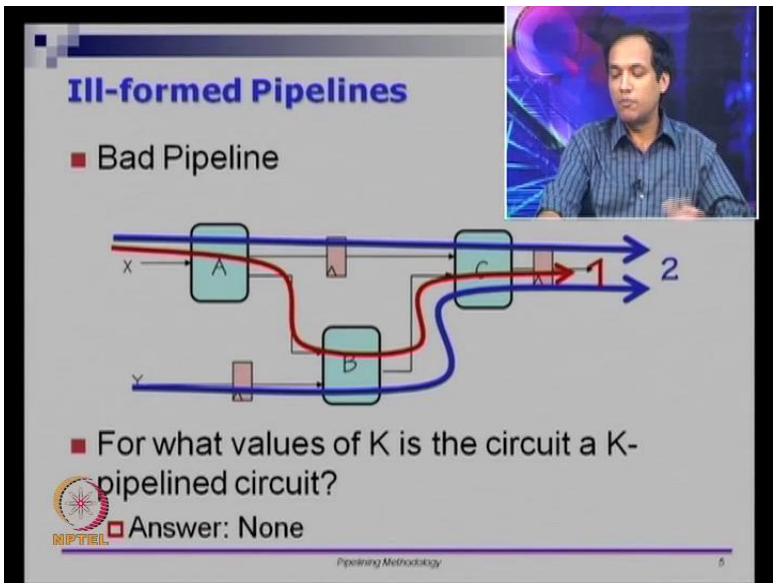


So, we already defined latency and throughput. So, latency of a K stage pipeline is K

times the clock period; that is commonly fed to all registers. So, sometimes we will refer to the latency with exact time units, we may say that, latency is 12 nano seconds. So, for instance, when we wash clothes, we said the latency of the pipeline system was 120 minutes, but more often than not you will refer to them as cycles. So, we will say that, the latency of this pipeline K stage pipeline is K cycles.

So, if you want the actual measure of time, you take the number of stages K, multiply it to the clock period that will tell you, what the actual latency is in terms of time units. And the throughput of a K pipeline circuit is the frequency of the clock that you are given. So, how many inputs can be processed per unit time is, usually the throughput and the examples are the number of floating point operations, you do per unit time or the number of millions of instructions, you can process per unit time and so on, are all actually measures of throughput. So, in the laundry example we said, our throughput is 1 by 60 loads per minute.

(Refer Slide Time: 05:24)



So, I am going to post a circuit and I am going to ask this question. For what values of K is the circuit of a K pipeline circuit? So, let us look at this circuit here, we have a combinational logic A and a combinational logic B and we have a combinational logic C. So, assume that, there were no flip flops, these pink color flip flops assume that, they were not there to begin with and if they are not there, then it is a zero stage pipeline.

So, any input that x or y that I give has to wait, so A has some delay and after that delay,

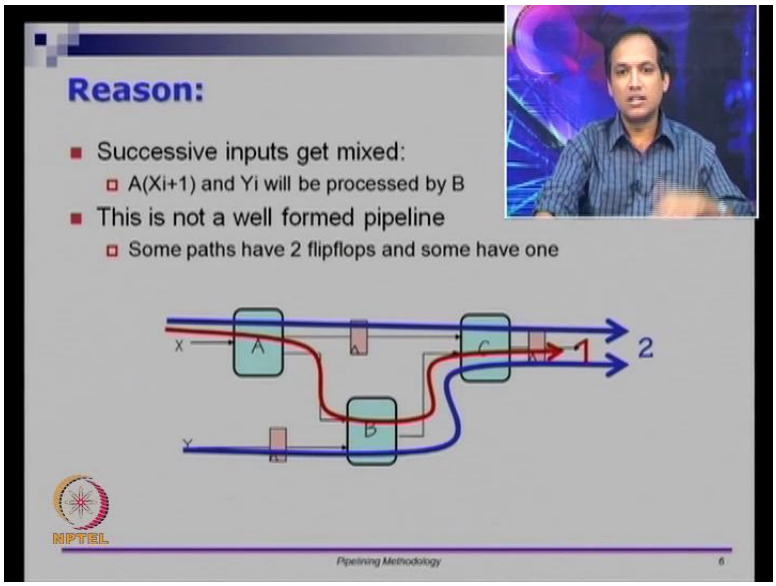
it will appear at C; B has some delay, so B itself will have to wait for A and only then, it can produce stable output at B. And once, these two inputs are stable, C will process and produce the output; that is for combinational logic. However, this question is, if I have put these flip flops here, let us assume that each one of this is a single bit value, if I put flip flops here, is this a valid case stage pipeline?

So, let us go back to the definition of a case stage pipeline, from every input to the output, we go and count the number of flip flops. So, let us take x input and let us take the output, let us see, how many flip flops is there in this path. So, through A I go, A is completely combinational, no flip flop yet, 1 flip flop here in this stage. And the output of this is going through some combinational logic and coming to this another flip flop here that is another flip flop stage.

So, this path, this blue path is two cycles or two flip flop stages. So, this y goes through 1 flip flops stage or one pipeline stage and another pipeline stage here, y also this blue path for y is also two pipeline stages. However, this red path is going through some combinational logic, some more combinational logic and another combinational logic and goes through 1 flip flop. So, this is a flip flop, this is a path which has only one pipeline stage.

So, if you remember the definition of a K pipeline circuit, it says all the paths must have the same number of stages. The blue paths have two stages, the red path has one stage, we cannot call it a one stage pipeline, because the blue paths have two stages, we cannot call it a two stage pipeline, because the red one has one stage. So, this is neither a one stage pipeline nor a two stage pipeline, we will call such pipelines, bad pipelines. This is not a well formed K stage pipeline. We will see why it is important later. So, it is not a well formed pipeline; that is the first thing that you have to know.

(Refer Slide Time: 08:18)

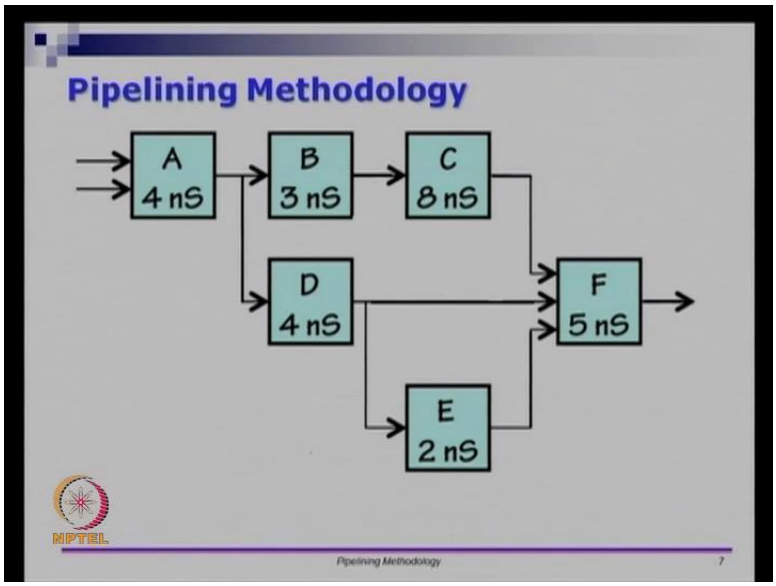


So, the reason why it is called a bad pipeline or a nil form pipeline is that, successive inputs will get mixed. So, if you go and look at the picture here, A of X i plus 1 comes through here, let us say if I place X i here in the next clock cycle, it will come here. So, A of X i plus 1 and B of Y i, will get processed by B. So, if you notice this, A of X i plus 1, which is the current input and Y i which is the previous stages input, which gets registered in this flip flop, they both get processed at B. So, B is not processing inputs that are coming from the same stage.

So, A of X i plus 1 is the current input, Y i was the previous input which got registered here. So, B will process those two together; that is not correct, so this is not a well formed pipeline. So, because B is not getting inputs from the same number of stages, C also by definition will not get it from the same number of stages. So, even if I had a circuit, which contain only A and B and if I put a flip flop here and let us say I put a flip flop here, so the path from y to B's output will be a two stage circuit.

But, the path from x to the B's flip flop would have been a one stage circuit, so that is not a correct pipeline circuit, so this is the bad combination. At any point of time, whatever input is given at the global input, you want to give all the inputs at one time, wait till the circuit produces the output or the chip produces the output and then, give the next set of inputs. So, this one is asking me to mix x and y inputs from multiple stages to be given together and that is not correct.

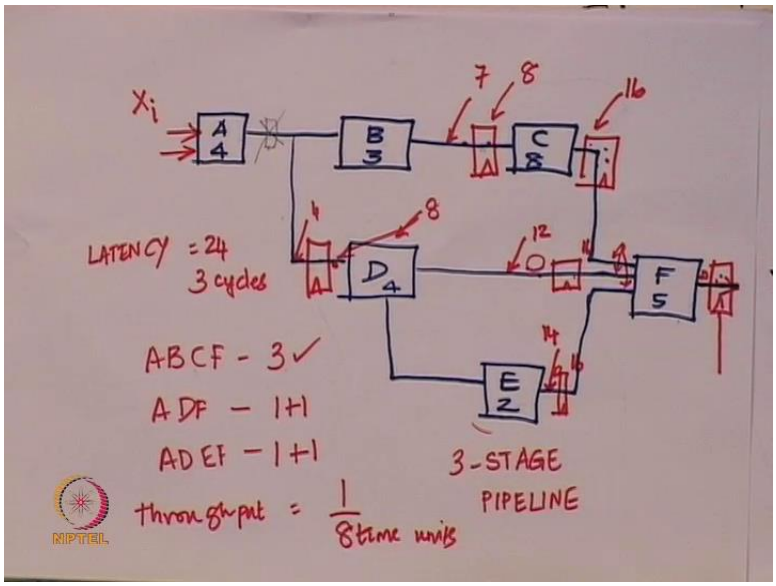
(Refer Slide Time: 10:19)



So, let us look at this slightly more detailed example. So, again let us not worry about what these actual basic blocks are. There are six basic blocks named A to F, these are some combinational circuits and they have these delays. A has a delay of 4 nano seconds, B has a delay of 3, C has a delay of 8 and what not. And if I go and look at what is the latency of this system, any change in input here should go through A, through B, through C, through F and go out. Another path is through A, through D, through F and go out, third path is through A, through D, through E and then, F and go out.

So, there are three different paths, inputs to output is either A, B, C, F or A, D, F or A, D, E, F and I have to wait for the largest of the delays. So, the latency of the system is the largest propogational delay, which is 4 plus 3, which is 7, 7 plus 8, which is 15, 15 plus 5 which is 20, so the latency of the system is 20 nano seconds. So, if I give any inputs here, then the circuit will stabilize, the whole chip will stabilize only after 20 nano seconds. I want to see if I can make this better by doing pipelining. So, as I mentioned, if I want to pipeline, a K stage pipeline will always start with a pipeline register at the output. So, let us look at this picture here.

(Refer Slide Time: 12:04)



So, I have this picture that I have drawn, which is a same picture as what we had earlier and I have the circuit here, which is taking two inputs here and it is producing an output here. So, if I want to pipeline, the first thing you have to do is, put a flip flop right here, so this is going to be a flip flop stage. So, every pipeline circuit should have a flip flop at the output, so I put a flip flop here.

Now, if I go and look at the combinational circuit F here, if I am getting a flip flop, the inputs that are going to come from these three wires, they should all belong to the same stage, they should all in some sense belong to the same stage. Let us look at what is feeding F, F is been fed by C, it is been fed by D and fed by E. If you go and look at C, C among all these blocks, let us assume that, A is a circuit which cannot be sub divided further, B is another circuit that cannot be sub divided further and so on for now.

If I go and look at this, C is a circuit that cannot be subdivided further also and C has the largest delay, it has the largest delay, it has 8 nano seconds of time. So, what I am going to do is, I am going to put a flip flop here at both the inputs and output of C. So, I am talking about a methodology. So, this methodology I will summarize this later, let us take this specific example and let us see, how to pipeline this circuit and do it in a correct manner.

So, there are two goals to the pipeline, one is, I want to improve the throughput of the system, because a throughput in this one is… So, this combinational delay is 20 nano

seconds, the throughput is 1 by 20 nano seconds, so I want to improve that. So, I want to improve that, at the same time, it should be a well formed K stage pipeline. So, I do not have a K in mind, I want to design a well formed K stage pipeline.

So, for a while if I did not put these flip flops, if I put only this flip flop, it will be a one stage pipeline and one stage pipeline is not quite useful. So, the one stage pipeline will still have a latency of 20 nano seconds. So, it is not making any better. So, when you are looking at pipeline circuits, you are asking for either a two stage pipeline or more. So, far I am trying to see we can to make that circuit.

So, if I put of flip flops at the output of C and that the input of C, then what I do know is, the clock period hopefully will be only 8 nano seconds and not more than 8. So, immediately we get a drastic cut down from a clock period of 20 nano seconds that we had earlier, we can now run a clock period of 8 nano seconds. So, this is crucial, because once you place the inputs and output of this combinational block, we do not have to do it all the time, but in this specific example, I have put it here and here.

The thing that I have to now do is, from the inputs if I go and look at it, is this is a well formed pipeline. So, from the input if I start here, I go through this flip flop, I go through this flip flop and I go through this flip flop. So, the path from A to F, so the path ABCF has 3 flip flops, so however, the path ADF has only 1 flip flop and ADEF also has 1 flip flop. This is not a well form pipeline and if it is not a well formed pipeline, then inputs from different stages are getting mixed at different plug points; that is not correct.

So, we want the same set of inputs going through different stages, going to the final output. So, we want to make this, a three stage pipeline. So, I already used 3 flip flops. So, maybe I will make a three stage pipeline or even may be a four stage pipeline. But, as of now I want to see at least can I make a well form three stage pipeline. To make this three stage pipeline, let us look at this picture again, so I have a flip flop here, so I already placed flip flop here.

So, F is supposed to process inputs that are all coming from the same input stage. So, a simple thing to do is, so let us say, if I put a flip flop here for instance, if I put a flip flop here. Then, from this flip flop to this flip flop, so I have already decided a flip flop for this location, it is in the output of C. Given that, now I have choices to where to put for the other ones. So, I need flip flops on ADF path, I need flip flops on ADEF path.

So, I can put it here or I can put it here, let us look at for the ADF path. If I decide to put it here, then if I go and look at what happens. So, a goes through one flip flop here and it goes through the second flip flop, it makes it as two stage path for ADF. However, at this point, if you notice F's input, so you go and look at the flip flop delay from here to here. So, from here to here, it will be 9 units through this path, it will be 4 plus 2 which is 6 plus 5 units through this path that will become 11 units.

If I put a flip flop here, it will become 11 units of delay, whereas, we said the largest combinational block has only 8 units of delay. So, I need to now go and see, whether I can put it somewhere else. So, if I choose to put a flip flop here, then the flop to flop delay is only 5 time units. So, from here to here is also 5 time units. So, these two are good to go.

Now, let us go and look at the ADEF path. So, if I put a flip flop here, then 4 plus 2 is 6, 6 plus 5 is 11; that is still not good. So, if I put a flip flop here; however, it will cut down the delay to 5. So, from this flip flop to this flip flop, all the three things C D and E; all of them take 5 time units. So, this is one stage of pipeline. Now, let us go and see, whether this is a well form pipeline circuit, ADC and F; you have not introduced new flip flops is 3.

For ADF I have introduced 1 flip flop, so this is 1 plus 1 now, so ADF has two stages of pipeline, ADEF has two stages of pipeline. So, it is still not a well formed pipeline circuit. So, this has 3 and these two have 2, I still need to introduce one more flip flop on the path ADF and ADEF. So, now, again we have a choice, if I want to put the flip flops, so let us say there are multiple locations. So, I have already decided to put a flip flop here by the way, so that is not there.

So, I am only looking at ADF and ADEF, so to do that, I can forget the circuit that is on the top here and all I need to do is, I need to put flip flops, either here, I could do it here or I can do it on the branch here. So, let us do that, if I put it on this one, the problem is even the path ABCF will encounter this flip flop. So, ABCF is a three stage pipeline, now if I put this flip flop here, before the branch that will make this 4 and the rest will become 3. So, that is still not a good well formed pipeline.

So, instead of putting it here, if I choose to put it here instead, so this is again another flip flop, if I choose to put it here, now let us go and look at all the paths. So from A, so this

flip flop is not there, so I cross that out, there is no flip flop there. From the input in this path, there are 3 flip flops, in this path, there are 3 flip flops, in this path also there are 3 flip flops. So, the overall circuit is a three stage pipeline.

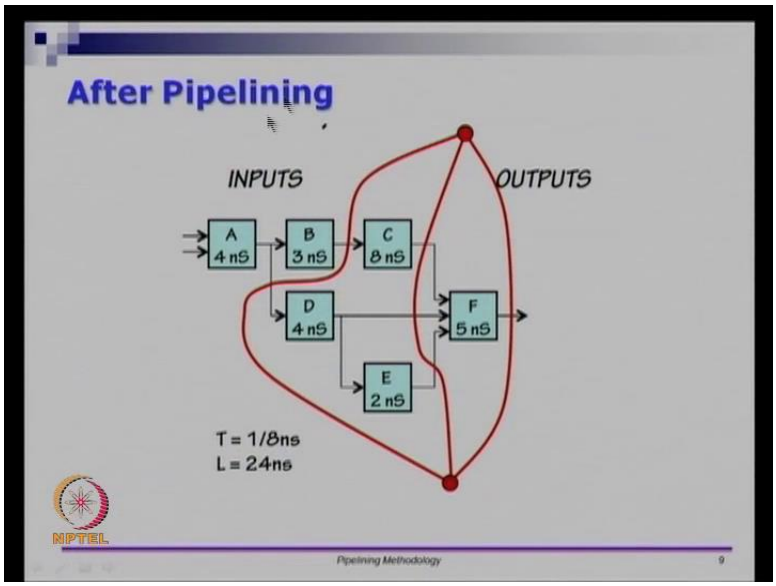
So, this circuit is a three stage pipeline, let us look at what the latency of this circuit is. So, the latency of the circuit is given input here, so it will go through A and B, if I give some input here Xi in 7 time units, I will get this processed here, x processed here in 4 time units, it will be ready here. So, this will be ready at 7 and this will be ready at 4, but I am going to bring in the clock only every 8 time units.

So, the first edge will get registered at 8, so Xi; so A and B of Xi will get ready here at 8 and it will also be ready available at 8. So, now the relevant things for Xi are available here and here, is available here not at the D at the flip flop. Then, to go to this stage from this flip to this flop, this is going to take 8 time units. So, this is that 16, if there is a stable input here, it will take 4 time units to come out of this. So, the input here will be ready at 12 time units, it goes through 4 here plus 2, 6. So, this will be ready at 14 time units.

So, the input to this flip flop is ready, we will get only ready at 16, this will be ready at 12 and this will be ready at 14, but I am bringing in a periodic signal. So, the next clock pulse will come at 8 plus 8, which is 16. So, there is 4 time units of waiting here and two time units of waiting here and at 16, the outputs of all the flip flops are stable. Now, 16 plus 5 at 21, this flip flop input will become stable, but the clock pulse is going to come at only 24, time period 24, so the latency through this system.

So, the latency is 24 time units whatever the time unit is or in terms of cycles, it is three cycles and the clock period is 8 time units. So, the throughput that you can get from the system is 1 over 8 time units, whatever the time units is. So, if this all nano seconds, this is then 1 over 8 nano seconds is the throughput. So, you can process so many inputs in every nano seconds; that is the meaning of this and so this is after the pipeline. So, let us see the circuit here.

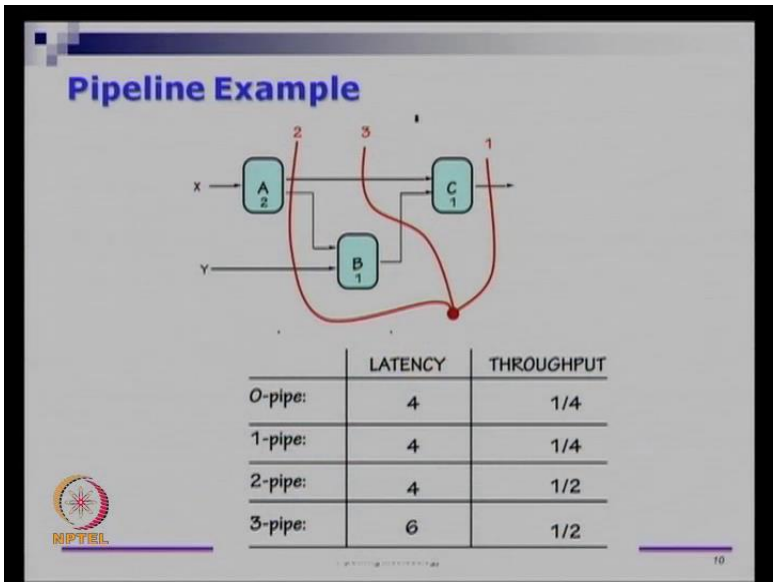
(Refer Slide Time: 23:25)



After pipelining, I have drawn this picture, so the throughput is 1 over 8 nano seconds and latency is 24 nano seconds. And I have drawn this picture to show, this red line means, this is all one stage of pipeline, this red line means,; that is one stage of pipeline and this red line means; one stage of pipeline. If I go and count the number of flip flops, the third stage pipeline is only registering F, so that is 1 flip flop. The second stage pipeline is registering C, registering D and registering E, so there are 3 flip flops here.

And the first stage pipeline is here which is registering B and registering on this branch of A. So, that is two flip flops. So, over all if we put 6 flip flops, the rate at which we can process increases dramatically with a minor increase in the latency. So, the combinational circuit had a latency of 24 time units. Now, we have a latency of 24 time units. So, at the expense of stretching the processing time of 1 input, we get more processing done per time unit. So, at the expense of latency, we get better throughput when you do pipelining.

(Refer Slide Time: 24:38)



So, let us take this other example. So, we have another example with A, B and C and let us assume that, these lines are not there to begin with. If I do a zero stage pipeline, then 2 plus 1 is 3 time units through the path AC, x will take 2 plus 1 plus 1 to go through the path A, B, C and y will take 1 plus 1, so it is 2. The worst case for y to go to the output is 2, x to go to the output to this path is 3 and this path is 4. So, the worst case time units is 4, the propagation delay is 4, I cannot change the inputs x and y till whatever inputs I give has taken 4 time units by the time, 4 time units are over it will be at the output.

So, the latency in the zero pipeline stage is 4 and the throughput is just 1 over latency, it is 1 by 4. If I put just one pipe line register at stage 1, if I just put one register here, then latency is still 4, because I am not changed the overall combinational delay. One stage pipeline must have single register only at the output. So, if I put it only here, then the latency is 4, the throughput does not improve, the latency does not increase. If I make a two stage pipeline, so if I put a flip flop here in the two stage pipeline, so this will take 2 time units.

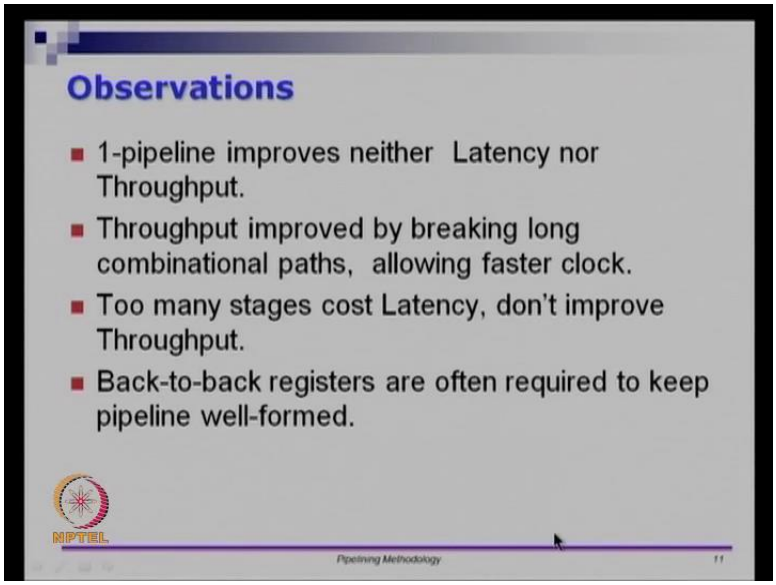
So, the latency is 2 plus 2, so the latency is 4 time units, but the throughput is now better. So, every 2 time units, so there is 1 input that is processed in every half a time unit. So, that is good. The three stage pipeline is, if I put a flip flops here and here, also it becomes a three stage pipeline, you can count it from x one red line, two red line, three red line at cross, from x through this path, there is one red line, two red line in three led red lines at

cross, y also threes will see three red lines, if I put 1 to 1, 3 all of them.

So, when you do that, the latency increases without increasing the throughput. So, this is something that you have to watch out for. So, blindly by putting more and more pipeline stages, you may not actually improve the throughput at all. The latency may increase without increasing the throughput. So, there is nice thing here. So, we kept increasing the latency or kept the latency at the same, but we saw different throughputs.

At some point the circuit is can hit an point of no return, which means, you have increased the latency without increasing the throughput and that is something that you should avoid. In this example, it is very clear that you have increased the latency unnecessarily to 6 time units without increasing the throughput.

(Refer Slide Time: 27:37)

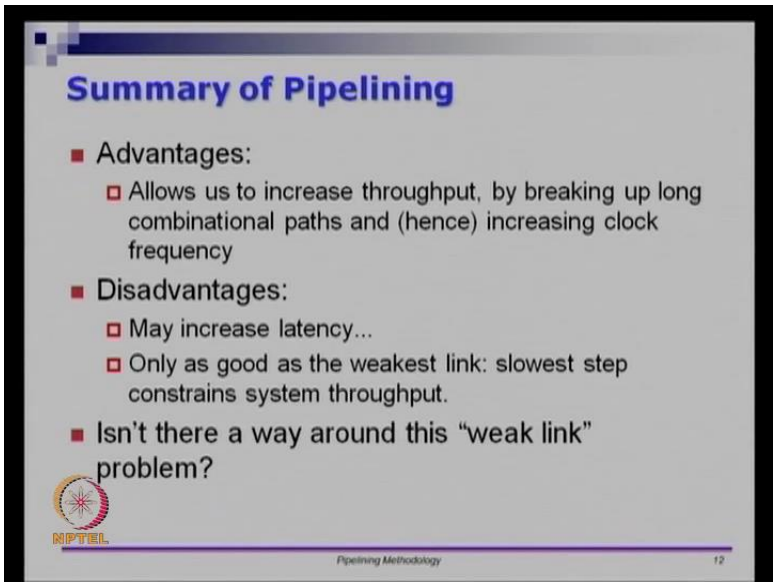


So, the observations are that the one pipeline does not improve the latency or throughput, it improves neither. The throughput is improved by breaking the long combinational paths allowing for faster clock. So, the longest combinational path is 2 plus 1 plus 1, you put more and more and more flip flops along the longest combinational path first, that will improve your throughput.

So, this is the same thing that we did in the previous circuit also. We took the longest path which was through A, B, C and F and on that path, we introduced as many flip flops as we want it. So, throughput is usually improved by breaking the long combinational

paths. Too many stages can cost latency without improving throughput and back to back registers are required to keep the pipeline well formed. So, these are the observations, I have talked about this pictures also, but these are the observations that I making from all these things, the summary of observations.

(Refer Slide Time: 28:35)



So, in summary pipelining is, it gives us several advantages, it allows us to increase the throughput by breaking up the longest combinational path. So, you can do different stages and get things done. So, because it is breaking the longest combinational path with various flip flops, you are actually increasing the clock frequency. So, that is the good thing, you improve the throughput; however, this may increase latency, if you are not careful and the pipelining is only as good as the weakest link.

For example in the previous circuit, I had a 8 time units of delay, if I cannot do anything about it combinational block, then I cannot run the circuit any faster than 8 time units. So, the weakest link is that combinational block C, I can do only as good as that. So, one thing we are going to look at in the next video is, is there a way around the weakest link.

So, the references this picture, this C is the weakest link here, can we somehow get around the weakest link, because this is forcing as to operate at a latency of 24 time units and the throughput is only 1 by 8 Nano seconds, can we do better than that. This is what we are going to do in the next lecture; this brings me to the end of this module 44. In the next lecture, we are going to see something called interleaving.

So, thank you and I will see you soon.