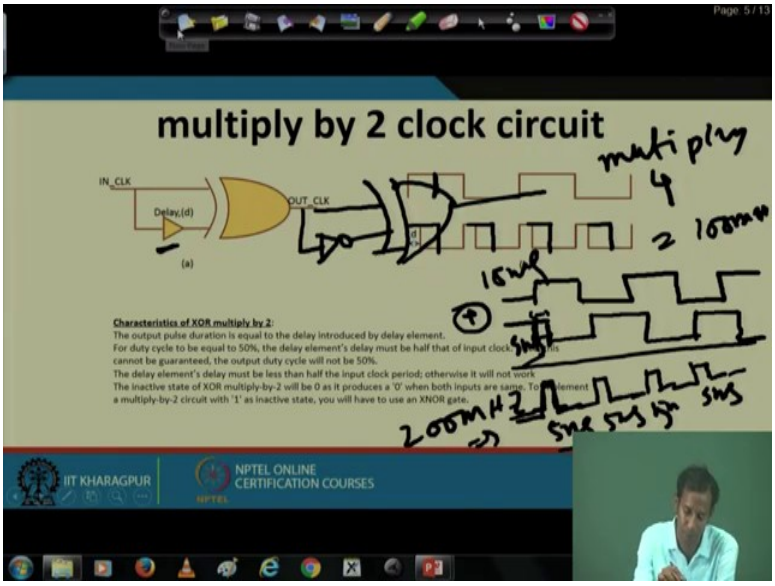
Architectural Design of Digital Integrated Circuits
Prof. Indranil Hatai
School of VLSI Technology
Indian Institute of Engineering Science and Technology, Shibpur, Howrah

Lecture – 59
Design Tips for Basic Circuits Design (Contd.)

Hello everyone, welcome to the course on Architectural Design of ICs. So, we are basically; that means, considering the different circuit design, how efficiently we can do. So, the tips and tricks to do the efficient circuit, how we can in an easiest way, we can do ok.

So, we have seen in the last class we have seen some of the like this edge detector circuit, How we can implement AND gate OR gate using multiplexer; that means, NAND gate any other gates or even in the latch or even the; that means, flip flop that we have already seen.

(Refer Slide Time: 01:10)



So, today we will see one another circuit where you have been asked to design one; that means, clock circuit, which will be multiplied by 2. So, to design this circuit how we can design the circuit of; that means, which will give me the clock signals which will be; that means, the frequency of the clock will be doubled.

So, to do that what we need that, we know that this XOR gate is nothing but, the 1 bit comparator. So, at that time what we are doing is that, in one of this we have just passed this input clock. In another actually part we have just placed this with 1 buffer, we have just passed this corresponding clock signals.

So, whenever this kind of; that means, circuit topology we used. So, at that time what is the delay we used here so, depending on that. So, what will happen if I consider, if I draw the corresponding graph of this. So, if this is my clock and at that time. So, if I just draw that ok, this is my clock signals. So, the inverted clock so, the inverted clock will be what? It will be like this why because there will be some the delay, which is based on this particular delay.

Now, whenever I will XOR these 2 particular signals. So, at that time what will happen? What I will get? I will get for this particular time that means, what is the amount of delay we have provided that much of time, we will get as 1 single pulse. And, then again at this again, we will get 1 single pulse, at this again we will get 1 single pulse, at this edge again we will get 1 single pulse something like this, we will get.

So; that means, now here what is happening, if this is this I consider as output clock. So, though there the output clock frequency will be just doubled; that means, whether if I consider here the period as let us say 10 nanosecond and, if I use that this delay of let us say this 5 nanoseconds. So, at that time these signals will now became of 5 nanosecond, each of this of 5 nanosecond.

So, these 5 nanoseconds are means, now here I can increase the frequency to twice. Twice means here as this time period becomes 5 nanoseconds. So, at that time the frequency of this particular clock will be of 200 megahertz. Whereas, in this particular case when this period of this clock is 10 so, at that time this is 100 megahertz ok. And, here you can see that, it depends upon the corresponding this amount of delay and here you see this is not 50 percent duty cycle.

So, when it will be 50 percent duty cycle when, this amount of delay is just half of this. So, when this half of this delay is mentioned over here. So, at that time these signals will be something like this. Again this will be like this, this will be like this, this will be like this.

So, at that time; that means, I will get 1 particular; that means, 1 clock, which will be having 50 percent duty cycle but, it will be having the frequency which is doubled in this particular case. So, this kind of circuit we can use for multiply by 2. Now, if I if actually if I ask you to design that how you can design this multiply by 4 ok.

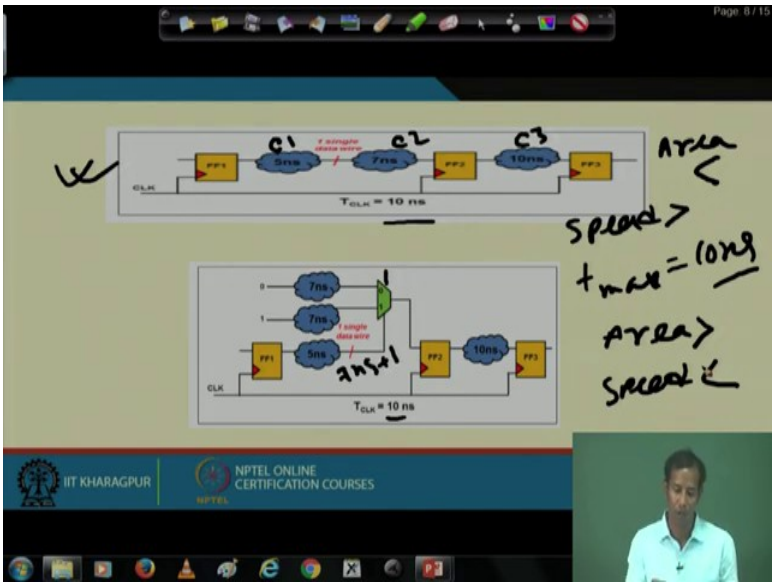
So, then multiply by 4 means, again these particular signals, again if you just extend and you put another XOR gate, where again you use this. So, for this particular circuit, it will be multiply by 4. So; that means, whenever we will do or we will add this kind of chain in series, it will be multiplied by 4, then it will be multiplied by here, it will be multiplied by 2. Then, it will again this 4, again if I add then that will be multiplied with 8. So, something like this if I add this in series, this correspondingly it will be the multiplication of this clock factor will also increase ok.

So, any of this in the power of 2, you can just design the circuit in this manner, which will be where the frequency will be just multiplied in the power of 2. So, and we have already seen that this how we can design the circuit divide by 2, that is also very easy.

Again we will see not only the power of 2 but, the power of something else ok. Power of something else means, that is with the when that is not in the power of 2, rather then it is like 3 5 7 so, how we can design that will also see ok. So, here actually the characteristic of this XOR based multiply; that means, clock multiplication is that, the output pulse duration is equal to the delay introduced by the delay element, for duty cycle to be exactly 50 percent.

The delay element, delay must be half of the input clock whatever, I just said. And, the output duty cycle will; that means, the delay element, delay must be less than half the input clock period, otherwise it will not work. So, this you can already do or you can; that means, using this kind of technique now, you can extend this circuit to implement any of the function so, now then next.

(Refer Slide Time: 08:14)



Suppose, we are having another example suppose, we are having 1 particular circuit something like this. What is there we are having flip flop 1, flip flop 2, flip flop 3 and here this is 1 combinational logic, which produce only single wire, which is connected to this another combinational logic.

Which; that means, consumes 7 nanoseconds of time and there in between this flip flop 2 and flip flop 3, it is having 10 nanoseconds of time. So, that that time period requirement will be how much? 5 plus 7 12 but, I my according to my constraint, I just need this T clock should be 10 nanosecond ok which is the maximum of these 10 nanoseconds.

So, is it possible, if I make some changes or in some architectural changes, is it possible to achieve the corresponding clock cycle time of 10 nanoseconds; that means, keeping the functionalities same. So, all the time whenever we do some kind of architectural modification or some; that means this optimization. But, at that time we always have to remember that the functionality will remain same. So, here keeping the functionality same can I achieve the clock period to 10.

So, I can achieve how I can achieve? As you see there is it is mentioned that only 1 single data where is basically connected from this combinational this if I just write this as C 1, C 2 and C 3. So, C 1 is basically connected to C 2 with a single wire ok. So, for that particular reason as there is only single wire. So, what we have did, if we can compute this 2 in parallel.

So, at that time I do not have to wait, actually here what we are doing. We are finishing this combinational task and then depending on this, then again we are starting the combinational task over here. But, instead of doing that what we can do, if we can compute this 5 nanosecond and 7 nanosecond in parallel so, depending on the 2 value.

One considering 0, another considering 1, means what as this is the single wire. So, this will be either of 0 or 1. So, from the beginning if we consider ok, I will just copy this combinational part twice ok. And, here 1 thing is that I do not have any area restriction, I only have the speed restriction; that means, the frequency I need to improve or the speed I need to improve. Area requirement is not that much constraint. So, at that time only I can do otherwise, I cannot. So, what I will do, I will just parallelly, I will compute this 5 nanosecond and 7 nanosecond by what technique.

So, this as this is the single wire. So, it may be 0 or 1. As we are doing actually it is not basically starting it is operation. Why because, it is not getting the signals from 5 nanoseconds. So, once this 5 nanosecond is done then, only it can starts it computation as this is connected in series. So, that is why it all we all the time it actually it will consume 12 nanosecond of time.

But, what I do I can do is that I copied this 7 nanosecond combinational; that means, C 2 twice. Once I compute using 0, once I compute 1; that means, I do not has to wait; that means, for 5 nanoseconds computation time to get this value whether, this value will be 0 or 1.

So, in parallelly with 5 nanoseconds I have already calculated or compute this 7 0 considering 0 and 1, I have already; that means, calculating. Then, based on this; that means, whether this value will be 0 or 1 so, based on this now I will select which path; that means, this considering.

If it is 0 then, if this particular value; that means, after 5 nanosecond, it finds that this is 0. So, at that time it will select the corresponding path and if it is find that this value is 1. So, at that time it will select this and it will go to the flip flop 2. That means, here I do not have to wait for the computation to be completed for C 1 and then I can take or then I can start my processing at C 2.

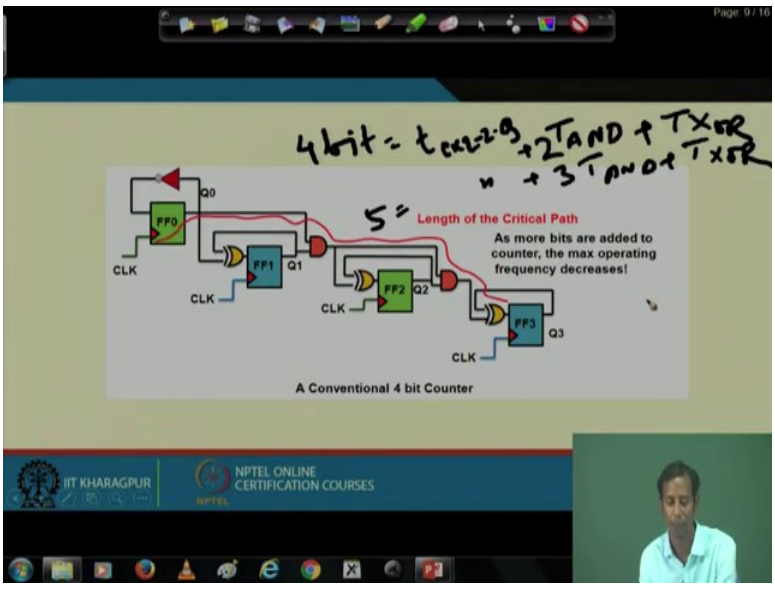
So, parallelly with C 1, I am doing the parallel computation in C 2 and depending on this multiplexers; that means, this based on this multiplexers now, I can select that ok, which path I will select? Whether I will select with the corresponding computation, which is considering 0 or I will consider the corresponding part which is considering the 1 ok.

So, now, as I am running these 3 as parallel so, the maximum delay will be here. The maximum delay will be of 7 nanoseconds plus if I say that ok, this is 1 nanosecond. So, 1 nanosecond of the multiplexer delay will be the maximum of this, which is again less than 10 nanoseconds.

So, in this particular case at that time the t max will be of 10 nanoseconds of time. And, now I can achieve very easily the; that means, the cycle time of 10 nanoseconds. In the earlier case or in this particular design, I never can achieve this clock period to 10. I made the architectural changes and now I can achieve the speed requirement to 10 ok. So, this is the tricks we and this is actually this circuit here, area will be lesser. But, here in this case area will be more, in this case speed is more but, in this case speed will be lesser ok.

So, different architecture different actually aspects, we can follow or we can design ok.

(Refer Slide Time: 15:30)



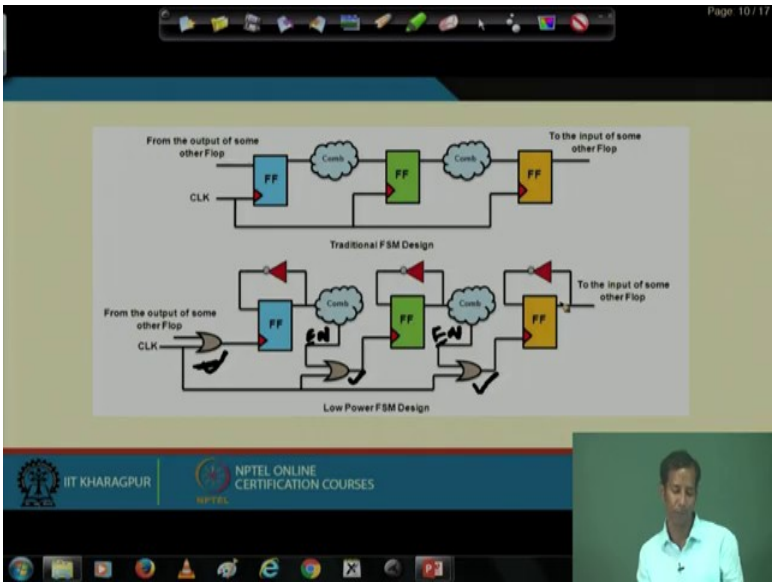
So, then again we will consider another example. So, this is one of the; that means, conventional 4 bit counter ok. So, whenever we actually consider this conventional 4 bit

counter means, it counts by let us say it increases by 1 ok. So, there will be 1 on; that means, there will be these flip flops and the input will be just like, it will be connected like this ok.

So, the very first is the; that means, this is the t flip flop, work as a t flip flop then, depending on this it will be also like whenever, this particular signals will become 1. So, this will be also like act as an inverter and then again if this is 1 then, again it will be just like inverter. So, this is the basic structure of 1 counter. So, it says that if I that means, if I consider the corresponding critical path so, which indicates this red color signals, this red color paths. So; that means, if I increase the corresponding suppose here for 4 bit, I am having this flip flop delay, plus 1 AND gate delay, plus 2 AND gate delay, plus 1 XOR gate delay.

So, if I increase it to 5 bits. So, at that time again I need another AND gate over here and XOR gate at the very end. So, at that time what will happen, the delay will be increased by another 1 AND gate; that means, what for 4 bit. The delay is t clock 2 q plus, 1 to 2 AND gate delay, T and plus 1, T XOR right. For 5 bit what it will be, it will be remain same, plus it will be 3 of T and, plus T XOR. In the same manner, if we increase the corresponding bit for this counter, this number of NAND gate will be increased ok.

(Refer Slide Time: 18:23)



So, now, consider at another example is that, in traditional finite state machine, we are having something like this. We are having this flip flops, flip flop, flip flops and in

between them, we are having this combinational logic, which are placed just in between of the registers but, whenever we consider the low power FSM or this low power finite state machine. So, at that time it is not like this here you see the in traditional FSM, the clock is basically connected synchronously, it is connected to every of this flip flop but, whenever if I ask you to design 1 low power FSM, the same FSM, if you have to consider for low power application.

So, at that time it will not be the same. So, at that time what you have to do, you have to use some kind of enable signals, which will be associated with the clock. So; that means, you have to here, you have to use the gated clock. So, how you can do so, at that time you are having this; that means, you are having this clock signals.

Then, again it is OR with the output of some other flops, which will be the clocks and then every of this particular signals will be now, it will be all these clocks are now it is gated ok. So, how this is gated? Here you see, this clock is now it comes with the enable signal. That means whenever, this particular clock; that means, this circuit is not running.

So, at that time depending on this particular logic, the clock signals will be just 0. If this particular signal, if this particular; that means, FSM is running at the running condition. So, according to the logic, which provides the corresponding enable signal, which comes from this combinational logic, they will be OR with the clock. So, whenever this enable signal is ok, this I need that enable signal make the clock enable and it will be in the running condition.

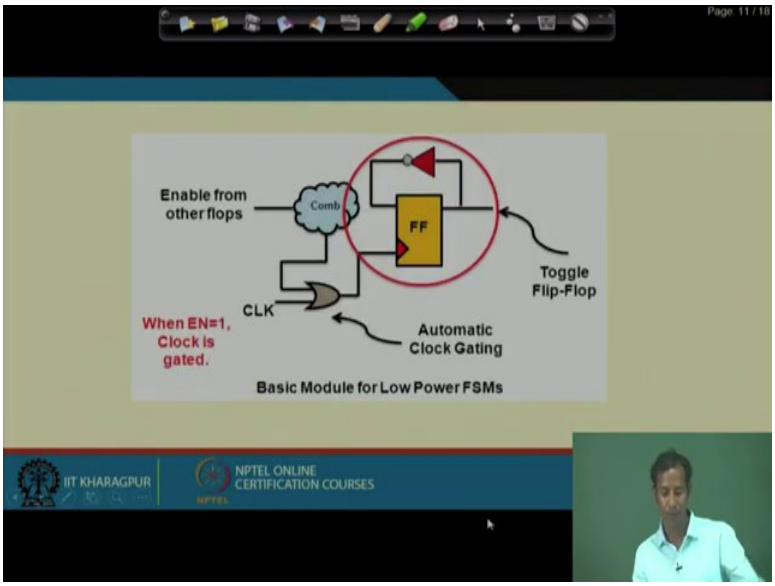
So, at that time every of this flip flop will be connected to the gated version of this particular clock ok. So, the same circuit they will be at that time modified to this particular architecture to achieve the low power. So, whenever we are doing; that means, we are making this traditional FSM to this low power FSM design.

So, at that time what we are doing, intentionally we are putting some of the extra hardware's which is in terms of these gates. Along with whenever we are putting this gates means what now, the corresponding delay will be also increase. So; that means, the speed of this will be also reduced by a factor.

So, area is also increased, the speed is also decreased. To achieve what to achieve a great performance or a good performance in terms of power consumption ok. So, this is one

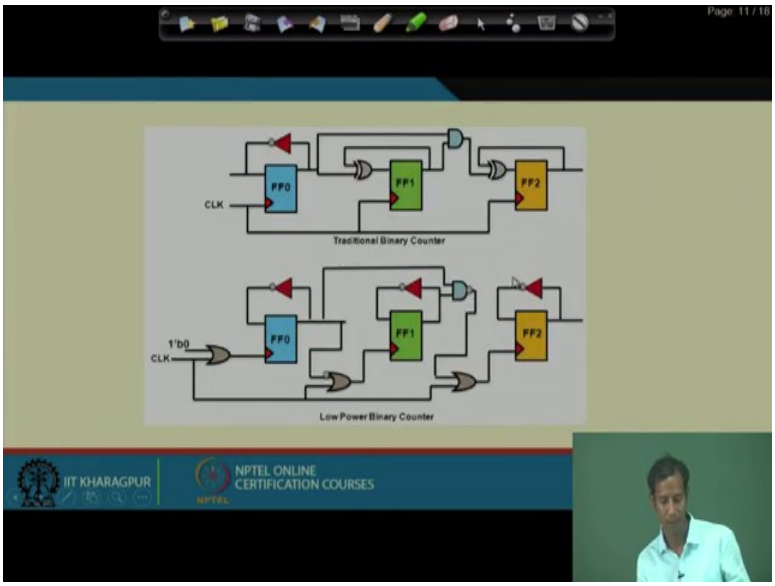
of; that means, the architecture has been modified in this particular case with respect to power consumption aspects ok. So, in the previous example, the circuit was implemented with respect to the speed constraint. So, in this particular case, this particular architecture has been developed in terms of power as a major constraint.

(Refer Slide Time: 22:38)



So, then this is actually this says that when this enable will be coming from this particular combinational loop, at that time this enable 1, the clock is gated and it will be activated. And, if this particular circuit; that means, when this enable is 0, at that time the clock will not be; that means, the clock will not affect the corresponding circuit.

(Refer Slide Time: 23:13)



So, this is one of the example of this is traditional binary counter, what we have already seen that ok. So, it is the first is this t flip flop and then we are having this XOR gate then, again it is ANDed.

So, this if I just make this traditional binary counter to the low power mode. So, at that time what will happen, this will be comes with this; that means, this is 1 tick b0. It is; that means, all the clocks are now gated and this will be the corresponding enable signals to the clock ok.

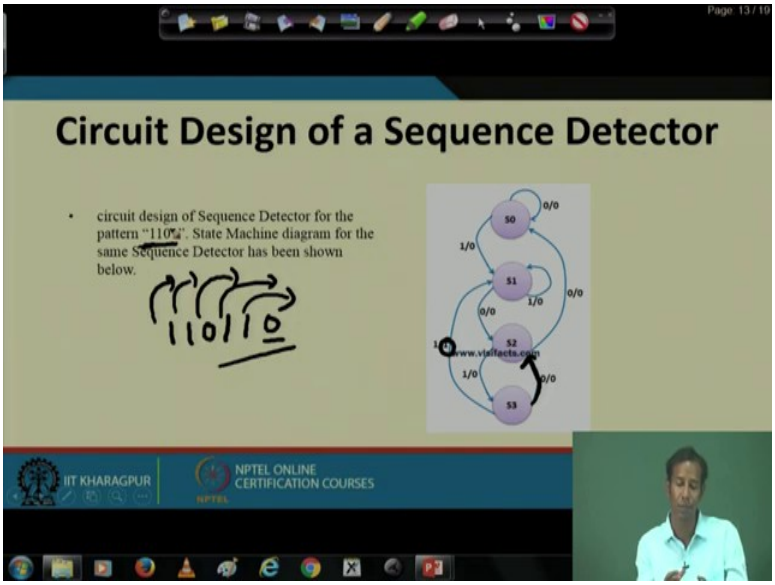
So, whenever this is not running; that means, it if it is all the portions is 0. So; that means, this will be again this will be also 0, this will be also 0. So, at that time the clock will be just clock 2, it will be at that time it will be just nothing but, this particular circuit which will be implemented over here.

So, some kind of modification in the architecture also; that means, in the gates also, it is being done. To make the corresponding traditional binary counter to be work as a low power binary counter ok. So, in case of actually here if you see that I need 1 inverter 2 XOR gate and 1 AND gate but, here what I need I need 3 inverter, 1 NAND gate and 3 OR gate.

So; that means, I have increased the speed sorry increase the area as well as I by putting this clock gating circuit, we are also decreasing the speed. But, we will achieve where we

can achieve the; that means, enhancement in terms of power. The traditional binary counter now, it is converted to low power binary counter ok.

(Refer Slide Time: 25:14)

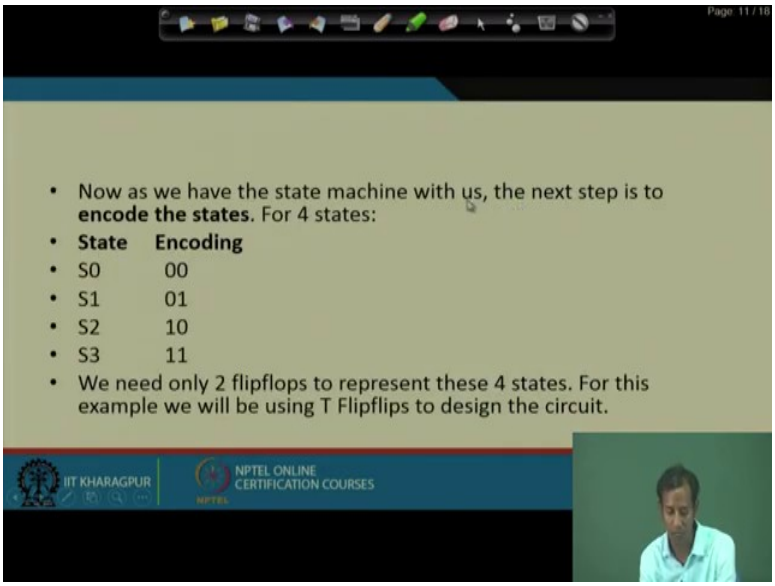


Now, then again another actually very common circuit which is used in digital circuit design that is sequence detector. Now, let us consider this we have to detect 1 sequence which is suppose one pulse is coming ok. So, we have to or the data stream is coming.

Now, we have to; that means, detect that this particular sequence of 1101 so, at that time how can I make the circuit of 1, this detector which can detects the corresponding sequence of 1101. So, at that time at the very beginning what we have to do, we can write the corresponding finite state machine or we can draw the finite state machine.

So, as we are having we have to select 4 different patterns that means, 1101. So, we will be having 4 different states, ok. So, the states are s0, s1, s2 and s3. So, at the very beginning, it says that whenever we are having we are at this particular s 0 state, if it is founded the input bit is 0. So, at that time again it will because, this bit will be 1 ok. So, whenever this bit will be 1 then, only it will go to the next; that means, it will try to go to find the next sequence.

(Refer Slide Time: 26:58)



If it is not 1 so, then again it will try to remain or it will try to check for the next bit ok.

So, that is why this if it is found 0 then, it will be 0 and the output will be also 0. If this s0 state finds 1 so, at that time it will go to what it will go to s1 state. So, in s1 state again, if it is found the in s1 state, it is corresponds to the sequence of 0. So, if it is founds 1, they can again it will select or it will again actually check, whether the next bit is 0 or 1. So, that is why again it will be on the same state only, it will not change it is state. So, at that time if it is found 0; that means, there is a match of the which pattern or which sequence I need to detect.

So, at that time the 0 will go to, if I find 0 then it will go to the next state which is s2. Then, again in s2 if I find 1 then, it will go to s3. Otherwise, if it is found that this is the 0 then, in s2 if it is found that this is 0. So, at that time what will get again it will be like 100, sorry 110.

So, that is not the actual actually pattern which we need then, again it will start finding from the starting 1 ok. So, then that is why it is again coming back to s0 then, again it is it will start finding 1, then 0, then 1 something like that.

So, as in this particular s2, if I find that this is 1. So, it will go to the next step, which is this s3. So, from s3, in s3 now if it gets that 1 so, then again when where it will be back,

it will be back to s1. Why s1 because, again actually it is already 1 ok. Then, again we have to select this as 1101.

So, if I write that, that 11011 something like this. So, at that time initially it has to start with this then, again if it is not matching then, again it will go to the next, it will consider the next bit, like next bit, like next bit. So, something like this 1 by 1, it will be processed through this particular state machine.

So, whenever it founds that this 1; that means, all this 1 0 and then 1 and 1. All the sequence found then, the output will be of 1. In all other case this will be just 0, if it is found that this particular thing is 0. So, then again it will go to s2, the previous state ok. So, now, this is the corresponding; that means, FSM for detecting this particular sequence 1101, if you have to detect one different sequence here. So, at that time this FSM will be also of different ok.

So, now once I can develop this FSM based on the pattern, which we need to detect. We have to calculate that what will be the corresponding circuit through which will implement this FSM. So, this is for today again, we will see how to start or how to basically draw the circuit of this which will give me the, which will detect the sequence of 1011 in a data stream so.

Thank you.