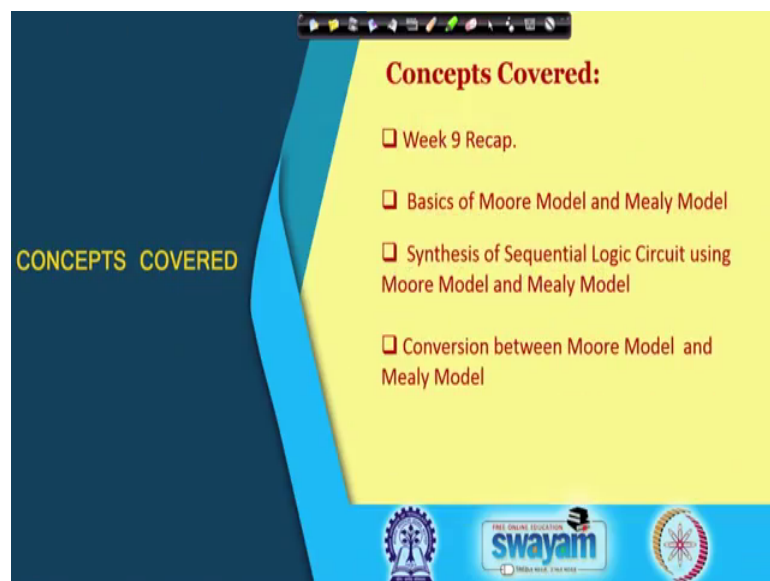


Digital Electronic Circuits.
Prof. Goutam Saha
Department of E & EC Engineering
Indian Institute of Technology, Kharagpur

Lecture – 46
Synthesis of Sequential Logic Circuit: Moore Model and Mealy Model

Hello everybody, we are in week 10 of this particular course. In the previous class we had seen previous week we had seen counter design different aspects of counter.

(Refer Slide Time: 00:27)



So, we shall begin with a quick recap of that and whatever we had done the designing aspect of it the counter circuit, if you remember that with clock trigger it was moving from one state to another and output was generating the output was generated based on how it was getting decoded ok. A general type of Sequential Logic Circuit Design we will also consider an external input ok.

So, in this class we shall consider that circuit design sequential circuit design; where the external input will also be there. And we shall take 2 routes; one is called Moore model based route another is Mealy model based route.

(Refer Slide Time: 01:20)



Week 9 Recap.

- A counter keeps a record of the number of times a particular event has occurred by advancing its state which is unique for each count. Modulo- n counter has n different states. It is also called divided-by- n counter. For m flip-flops used in counter design, $2^m \geq n$.
- In asynchronous counter, clocking of consecutive flip-flops are through rippling and triggering happens in different point of time. Here, the propagation delay is cumulative.
- A decoding gate connected to the outputs of a counter is activated only when the counter content is equal to a given state.
- In synchronous counter, every flip-flop is triggered simultaneously which avoids accumulation of delay and possible glitch.
- Mod- m counter cascaded with Mod- n counter gives Mod- $m \times n$ counter. Counting sequence changes with order of cascade. Mod 2 then 5: BCD; reverse: Bi-quinary.
- Using asynchronous reset, counting state 0 can be enforced early and in a clock cycle in which a specific count is reached. But, a glitch may occur in decoding.
- Sequential logic circuit design concept can be applied to synchronous counter design for different modulo numbers without any glitch.

Logos at the bottom: Swamyam, and other institutional logos.

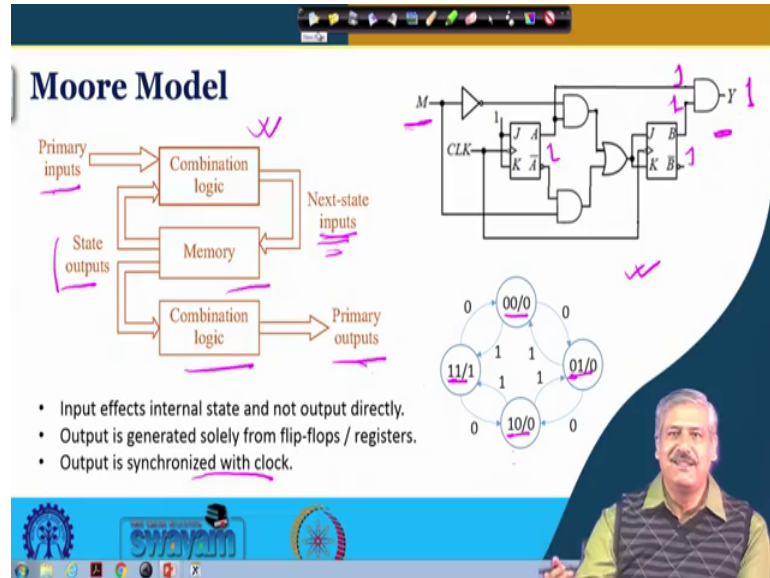
And so let us see before that quick recap, so, we had seen that counter a records number of times a particular event has occurred by advancing state which is unique ok. And the states could be can be sequentially going up, and called up counter and can come down then its called down counter. And in asynchronous sequential asynchronous counter we had seen that the consecutive flip flops are triggered a through a rippling ok.

So, one flip flop output these triggering the next flip flop clock ok. So, that way there is a delay between the triggering of consecutive flip flops. And this delay is cumulative ok, which can cause issue in certain cases. Decoding is very important and later on we had seen that for certain cases there could be glitches and that glitches need to be removed for sequential logic operation. synchronous counter we had seen that all the clocks are getting triggered at the same time. So, at same by the same clock edge in its regard counter and we had seen cascaded counter if mod 2 is cascaded with mod 3, then mod 6 counter was there and 2 is 5 mod 10 counter was there.

So, in general you can see that m with n it will be m multiplied by n that will be the mod value ok. And the order of these cascading is also important that is what we had seen. So, mod 2 mod 5 if you put one after another then you get BCD counter. If you have mod 5 first and then mod 2 then it is Bi quinary counter ok. And to get different kind of modular number we can use asynchronous reset, but that now for a glitch and we

can have standard sequential logic circuit design concept applied for counter design where these glitch can be removed ok.

(Refer Slide Time: 03:43)



And he said in this class we are looking at Moore general approach for sequential logic circuit design not specifically tied being tied with counter was a specific case. So, this is a Moore general case, in which we have got input to the circuit sequential logic circuit and it is generating certain output and inside the circuit the there is an advancement of state changes one after another ok.

So, these are the 3 important things right so these primary inputs where we are seeing primary input this there will be certain input which will be generated inside the circuitry ok; by those can be considered you know intermediate input ok. And accordingly there will be intermediate output. So, these primary inputs that is coming from the external world all right, they are affecting changes in the state ok, but it is not affecting change in the output that is made available to the outside world.

So there is no direct connection, you can see the primary output that is generated from combinational logic is developed from the state or the memory elements or the flip flops or the registers ok. These memory elements or flip flops or the state gets influenced by this primary input of course, so, indirectly it effects not directly right. So, that is what you see in Moore model that the primary outputs are generated solely from the states ok. Whereas, the state the change of the state that occurs based on certain logic certain input

that are presented before the flip flops at the input of the flip flops. So, that is generated that is generated through primary inputs and the previous state the current state, so, together they derive they drive the next state ok. So, that is certain logic combinational logic will be there by which you are generating the next state value. So, this is a general block diagram level representation of Moore model ok.

So, output will be generated solely from the states. So, whenever state change takes place after that, as long as that state remains so; that means, that for that clock period the output will become available would remain available ok; and it is synchronized with the clock. So, state clocking state change output available after whatever this combinatorial logic propagation delay is there is it clear.

So, that is how Moore model works right. So, primary input to for changing the state together with the current state and this state only defining the primary output. Now we had seen such circuit before Moore model we have seen before, if you remember though it was not explicitly a primary input as such, but if you remember the up down counter this is a section of it just we have taken 2 bit counter so mod 4 right.

So, there was an external inputs say M and m u could change any time. So, if you keep M is equal to 0 right at any given state, so state is 0 0. So, it is up count mode so M is equal to 0 it was going to 0 1 ok and M is equal to 1 it was going to down count mode 1 1.

So, that is the way state transition diagram would be written ok. Then M if it is at 0 1 M is equal to 0 it will go to 1 0 up count and if it is 1 it is down count so from 0 1 it will go to 0 0. So, this is the way you can these 4 states you can see for 0 it is going from 0 0 0 1 1 0 2 1 1 and for 1 ok. So, it will go from 0 0 to 1 1 to 1 0 to 0 1 and go back going back to 0 0 right.

So, in this particular case what we had seen that output can be generated since it is a mod 4 counter a count of you know 11 sorry 1 1 is taken place. So, this is 1 and this is 1 in we are discussing it writing in binary ok. So, when both of them are 1 then the output is 1. So, it is solely derived from the flip flop outputs. So, flip flop outputs defining the state. So, M is not directly affecting the output ok.

So, of course, it is a case of a Moore model right and here we write the output alongside the a state value within the state ok. So, this each circle defines the state, so 0 0 the output is 0 0 1 output is 0 1 0 output is 0 and 1 1 output is 1 is it clear ok. So, this is one thing, but still we are analyzing it we shall look at the synthesis part later ok. So, this is the objective of today's class we shall come to that.

(Refer Slide Time: 09:34)

Mealy Model

Primary inputs → Combination logic → Primary outputs
 State outputs → Combination logic → Next-state inputs
 Next-state inputs → Memory → State outputs

- Input effects internal state and output directly.
- Output is generated from flip-flops / registers and input.
- Output is not synchronized with clock, may change if the input changes during a clock period.
- Input transients / glitches may affect the output

Mealy model so from the discussion we had on Moore model you can expect that Mealy model will be related to it, but somewhat different yes in these of course, the primary input and the current state ok.

So, together the current state together they have a part of the combinatorial logic ok, which will be deciding the next state this is similar to Moore model. So primary input and current state values together through a combinatorial logic is defining the next state inputs and next state is obtained ok. But another part of the combinatorial logic is also taking primary input and the current state and generating the primary output the primary output again what I mean by this is the output that is made available to the external world ok.

So, secondary input or secondary output these are internally generated intermediate values within the logic circuit. So, I am not talking about that. So, this one the primary outputs now is having input also from primary input ok. Earlier in Moore model it was only from the memory or the state values flip flop values is it clear, but that is the

difference. Now it looks I mean what is a big deal about it, but when you go forward we shall see that this has got you know lot of significance, right now what we can understand what we can understand is that, the output need not will not be synchronized with the clock ok.

So, whenever input change in between a clock cycle if the input changes, so output will be changing ok. Because if such is the logic such is the combinatorial logic right because it is derived from input also right, any glitch any transient value in the input will go directly to this output ok. Now if these are certain issues associated with it which one can consider disadvantage or you know problematic the other issue is that if you look at the response time ok.

So, whenever input changes output can respond immediately, after whatever this combinatorial logic propagation delay; for Moore model if input changes accordingly the state will change and then state will generate the output ok. So, some amount of time is required in between for the clock getting triggered so depends on the clock period ok. And also because current state and primary input together can generate output number of states required which will be cleared or later maybe less which can affect less amount of hardware ok.

So, these are certain things pros and cons of two models right and accordingly to kind of you know finite state machine Moore machine or Mealy machine can be develop ok. And each has its own as we said advantage disadvantage it accordingly depending on our requirement we shall be using it clear.

And have you seen some such circuit before Mealy model based circuit ok, yes we had seen this is also from our this second part of the discussion sequential logic based circuit on the earlier classes. So, there we had seen one such circuit ok.

So, there if you see that if this is X of these external input so this is connected directly to the output generating combinatorial logic over here this and gate and this is Q bar right. So, what does it say the corresponding state transition diagram is this we have analyzed before. So, I am not doing it again. So, you refer to that particular class in the beginning of the sequential logic circuit part that particular week 7 ok. So, week 7 you can see it right.

So, here whenever these input is at state 0 sorry the flip flop is at state 0 ok, the circuit is at stage 0. So, at that time if a 1 comes right immediately the output is getting generated output is getting generated as high ok, otherwise the output remains 0 ok. So, 0 and 0 here means it is 1. So, this one and whenever it becomes 1 it does not depend need to depend on the clock. So, that is why it is called not synchronized with that clock right. But again the response is immediately it can respond. So, now let us look at because we are now look into the synthesis design aspect of it.

(Refer Slide Time: 14:44)

Synthesis with Moore Model

Problem Statement:
A sequence detector for '110' from a binary data stream is to be designed.

State definition:
 a: No bit is correctly decoded (initial state)
 b: 1 bit is correctly decoded
 c: 2 bits are correctly decoded
 d: 3 bits are correctly decoded

State transition diagram

Example:

CLK	0	1	2	3	4	5	6	7	8	9	10
Input	0	1	0	1	1	1	0	1	1	0	0
State	a	a	b	a	b	c	c	d	b	c	d
Output	0	0	0	0	0	0	0	1	0	0	1

So, for that we take up we one problem which we first define very simple problem. So, something which you had seen before in a different context; so a sequence detector right. Did not we see it for the shift register and other applications in those cases we have seen sequence detector right. So, here you are we are saying it in a different point of view. So, the sequence to be detected is very simple 1 1 0 ok. So, first bit should be 1 second bit should be 1 and third bit should be 0 as soon as it is done, then the output will say output of this circuit will say that these sequence this bit pattern has been detected ok. So, it will go to 1 otherwise it will remain 0 is it clear.

So, if it is shift registered based or register base going in these 3 such flip flops will be required. And we will push it and then we will be having a X nor kind of you know logic circuit which we had seen before ok. So, here let us see whether we require 3 flip flops or less ok. So, now for that the first step required is defining the states ok. So, the

problem now we are breaking into sub problems ok. So, these are the this is what needs to be done ok.

So, for that the circuit should move from 1 state to another. So, we initialize the circuit with a state say a ok, which means that none of the bit in the prescribed sequence is correctly decoded which is reasonable is not it ok. Because this circuit is just to reset is just you have began you have begun to sample the input data stream ok. Now if you receive 1 bit correctly, so you say the circuit moves from state a to state b. So, that is b 2 bits detected correctly 3; 3 bits detect correctly d ok.

So, a is no bit b is 1 bit c is 2 bit and 3 d is 3 bits directed correctly is it fine. And these are the this is the requirement 3 bits are detected correctly right and whenever you go to state d; so Moore model it will generate output so that is the idea ok. Now we need the state transition diagram, and once we have the state transition diagram state table and all and those journey we shall we know that part we shall see later.

So, first let us see what would be the state transition diagram for this one. So, here is the state transition diagram. So, you begin with state a these 0 means no output is 0 of course, no bit is detected output should be 0. So, that part we shall take discuss later. So, first is a now when it is at a input data stream is coming it can it can be 0 it can be 1 both are possible is it ok. So if 0 comes, then it will stay at a and if it 1 comes right it has to go to state b because b refers 1 bit is correctly decoded.

So, first bit decoded is properly decoded out of 1 1 0 ok; so it goes to b. When it is at b if it receives 1 then second bit is correctly detected. So, it goes to c so that is your c that is what is happening here. But after getting 1 if it receives 0 then what it has to do it has to do all over again this 1 1 0 1 1 0 right. So, after receiving 1 if it receives 0 it just to restart the process so; that means, it has to go back to state a which signifies that no bit is currently detected; because the sequence at this could be 1 0 1 1 0 after b received a and then this c bits right clear.

Now, at c possibility is that it can get a 0 or it can get a 1 right. So, c means you have already got 2 thing detected correctly detected right. So, if 0 comes all 3 are done, so you will go to d over here right. But instead of 0 if you get a 1 1 1 1 what will you do you will stay at c because after that if a 0 comes you should be able to detect the sequence ok.

So, that is the logical development. So, at c if you receive a 1 you stay at c ok. Now when you are at d 1 1 0 is currently detected. So, 1 is there output is 1 right. And when at d if you receive a 0 right so; that means, 1 1 0 then again 0 right. So, after that if you keep continuing detecting the sequence then again you would require 1 1 0 kind of thing. So, basically that is the initial state you have to go back that is no bit is properly detected. So, it is going to a and after 1 1 0 if you receive a 1 ok. That means, first bit is correctly detected for the next sequence which could be 1 1 1 1 0 1 1 0 right.

So, that is what you see over here that it is going to b, this is the corresponding state transition diagram ok. So, a output is 0 b output is 0 c output is 0 and d output is 1 is it fine. Now if you look at an example case you can do it yourself you can take any random you know input you know getting 0 first bit second in the clock cycle 1 0 1 1 0 this way right. And you can follow this state transition diagram and see what is happening.

So, state initially it is 0 so it is at a it receives a 0 it goes to state a it receives a 1 goes to state b, it receives a 0 goes to state a b from b it goes to 0 a ok. It receives a 1 it goes to b receives a 1 goes to c receives a receives a 1 1 1 again 1. So, it will remain that c, then 0 it goes to d. And whenever it goes to d it generates a 1 ok. So, the next clock cycle it is happening; so 1 1 0 has taken you know place over here in 4 5 6 and in 7 it is happening.

Similarly, 7 8 9 10 if you look at another 1 1 0 is over here a next clock cycle it is getting completed here 9th what the output will be at (Refer Time: 00:00) cycle is it fine.

(Refer Slide Time: 22:22)

Synthesis with Mealy Model

State definition:

- a: No bit is correctly decoded (initial state)
- b: 1 bit is correctly decoded
- c: 2 bits are correctly decoded

Example:

CLK	0	1	2	3	4	5	6	7	8	9	10
Input	0	1	0	1	1	1	0	1	1	0	0
State	a	a	b	a	b	c	c	a	b	c	a
Output	0	0	0	0	0	0	1	0	0	1	0

Problem Statement:

A sequence detector for '110' from a binary data stream is to be designed.

State transition diagram

Now, compared to this same problem; if we try to realize using Mealy model ok. So, state definitions are similar right similar. But we are not defining d here why let us see we did not initially, we can define d but we will say we shall see that d is not required 3 bits correctly decoded. Why? Let us see because we can use the input to get the output directly.

So, state a 0 comes it will remain at state a initial state. So, 0 output is 0 1 comes it goes to b output is 0 right, because 1 1 0 is required then only the output will be work. State b 0 comes it comes back to the initial state similar to what we had done before output is 0. State b which is 1 bit correctly decoded 1 comes. So, 2 bits get correctly decoded it will go to c which it has an output is 0 at that time is it fine.

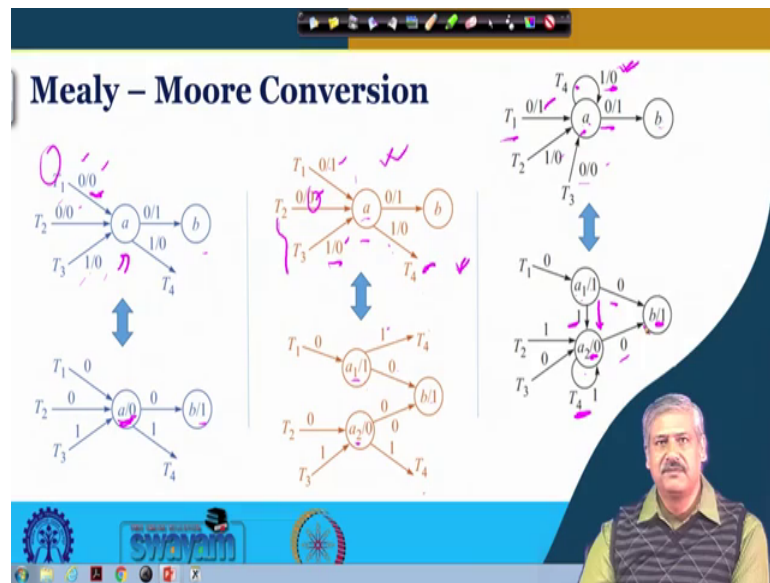
Now, at state c it can receive 0 or 1 and we know in Mealy model current state value, if you remember the model and the input and this is the combinatorial logic together you can generate the output right. So, c and input has come 0 you need not go to the next step immediately, based on this that it is at c and input is 0 right c means 1 1 2 bits a properly detected and next bit is also in the right order you can generate the output in the same clock cycle itself right. And 1 1 0 then you can go back to the initial state a is it clear.

So, that is why d was not required and at c if it receives one; that means, 1 1 again it has got 1 right. So, it will remain at state c with output 0 is it fine. So, this is the corresponding Mealy model that you get for the same problem right. And if you look at the that same example the example that I had we had discussed before the input is coming in this manner ok.

So, say state change is taking place, so 0 if a comes you know if a is the initial state 0 comes, it is a 1 it is b next state is b 0 comes, again it goes back to a 1 comes it is b again 1 comes it goes to c again 1 comes it is c right. Now if 0 comes, right you can take immediately take the decision. In the same clock cycle so it is at c the output is certain value and together with the flip flop output which is presenting c. And then these input x has come, so you have a combinatorial logic circuit together in that clock cycle itself we cannot take the decision.

So, earlier it was in 7th and similarly for the next case was in 10th. Now it is in 6th and 9th 1 cycle before is it fine ok. So, this is how it works.

(Refer Slide Time: 25:53)



Now, it is possible to have a conversion rule established between Mealy and Moore models. So, if you design it in one model right your state transition diagram. And then you decide it to move to another model right, you can start the exercise right from the beginning or you can use this conversion rule and get the circuit get the corresponding step transition diagram of the other type ok.

So, you see there are some basic rules the rules are like this, so here in this state this is the Mealy model. So, all the paths that are leading to state a from other places you see it is 0 and this is 0 0 0 1 0, so all the outputs are 0. So, you can very well take the input over here this sorry this output 0 within this one this state without any ambiguity no conflict and you can write 0 0 1 here right.

So, output will be generated only when of course, it goes to state a not in the that previous clock cycle whatever actually was causing it that particular state value and 0 together output was 0; so it is not like that. Next clock output will be generated from this only because it is Moore model, but we can write it without any ambiguity. Here it is 0 1 so there is no other thing o, we can take the 1 inside it. So, this is the usual thing where we have no you know confusion or conflict compared to that see the next 1. So, what this state a ok. So, here T 2 and T 3 these 2 path 0 0 and 1 0 the output is 0 and zero, but for T 1 the output is 1 ok.

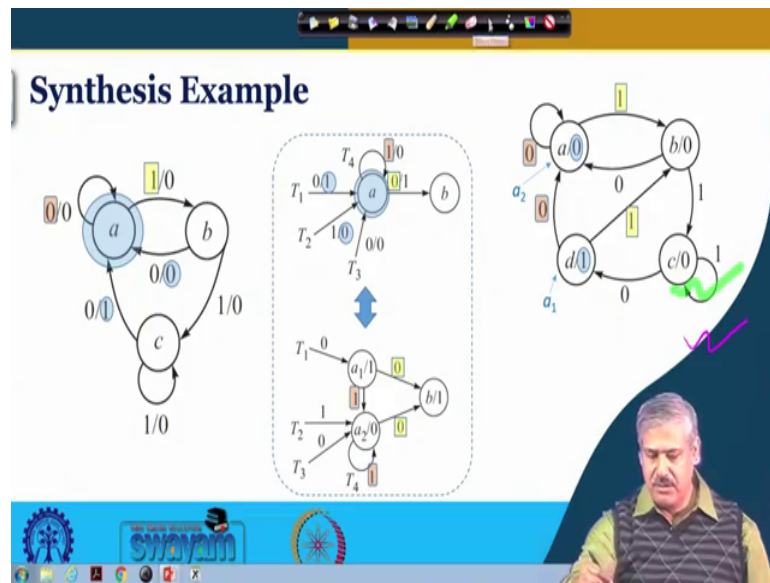
So, if you put inside a 0 good model means output cannot be generated from the input and previous state is not it. So, it has to be from the state in which it goes into right, but you cannot put 0 or 1 I mean both the things are not possible in 1 state. So, what you need to do you need to split the state. So, you can see that a is got a has got split a 1 and a 2 ok.

So, in 1 where T 1 is leading right So, 0 to 1. So, basically you have put the 1 and T 2 to T 3 these 2 are leading you are put a 2 which is output is 0. And now we was going from a b and T 4 were there. So, same thing b and T 4 there is no conflict over here, so 0 0 so there will be 1 path going for 1 to T 4 another path to T 4 and b there is only there is no conflict only 1 1. So, inside the 1 you will take the inside b we will take the 1 is it clear. And the third thing in a situation like this, in a situation like this where there is a basically conflicting requirement in the output.

So, T 1 0 1 T 2 T 3 it is 1 0 and 0 0. So, basically the output is 0 and 0 and here the output is 1 ok. So, we know that we need to split. But in addition to that what you see a there is a T 4 unlike this particular case where for 1 it is staying in a and the output is 0 right. So, when you split it, so this T 4 will be part of this 1 a 2 where the output is 0 ok.

So, that is what you have done; and if it goes from T goes to a 1 which is output 1 via T 1. So, 0 1 after 0 it has got here, if it is 1 then it will stay back here. So, which is output is generated 0. So, if it is getting a 1 it will coming to a 2 that is 0 right. And a is going to b and the output of is 1 so output is contained in b and for both the cases now a 1 and a 2 right for 0 it will go to b is it clear and similarly the other way the opposite is also true.

(Refer Slide Time: 30:38)



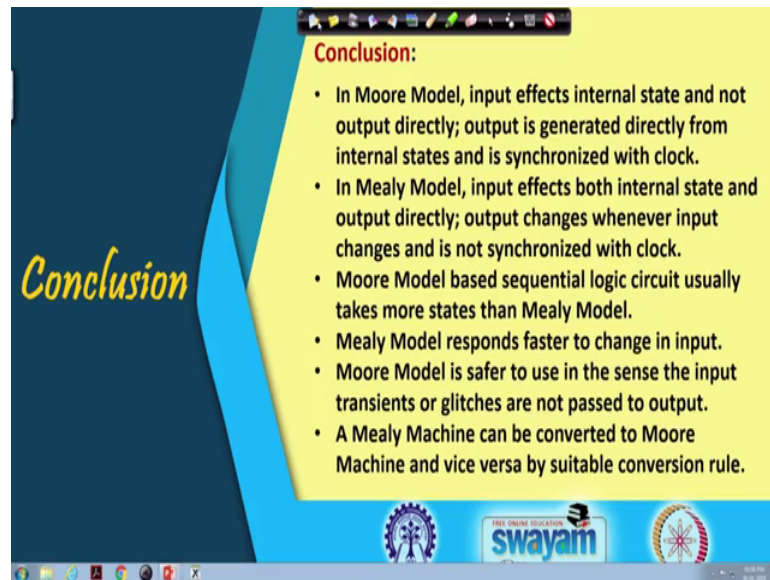
And 2 n we shall just look at the Mealy model and Moore model that we had developed if we apply this rule. So, either we can get directly the thing; yes we can get it right. So, this was the Mealy model ok. And this was the Moore model. And if you see for b so this is only case where the output is 0, so, we can take it inside there is no problem for c there is no conflict, this is 0 this is also 0. So, you can take 0 inside there is no problem ok. So, where is the problem is therefore, a and I have marked it. So, this is 1 and this is 0 these are the problematic case right. So, we need to split the a. And in the design process we did not do it now we can see that if we split it you can name it a 1 a 2 or it has been named there a and d ok.

So, one of them is having 0 so is a is having 0 and d will be having the other one is having 1 is it ok. So, it is similar to your a 1 a 2 following this principle right ok. Now the other cases when this a was there, so 0 when it is 0 it will be looping itself. So, it is 0 it is looping itself like similar to in this example 1 ok. And whenever it is at 0 and if it gets a 0 then it should go to a that has what has happened you can see similar to if it gets a 1 it has come to a 2 so similar over here ok.

The other thing from a it was going to b for 1 at the input now it has been split. So, for both the cases a and d it will go to b next state right. So, that is what you see over here for 0 these are the 2 things that they (Refer Time: 32:40) visited. So, similarly over here what you can see ok. So, the opposite when you apply there also you have to look at

where these values are different and then you can think of joining them ok. Otherwise this cases are straightforward I mean you know there is no such option for joining on the cases, over here like this we can joined to get a Mealy model like this or Mealy to Moore we have already seen is it fine.

(Refer Slide Time: 33:19)



Conclusion:

- In Moore Model, input effects internal state and not output directly; output is generated directly from internal states and is synchronized with clock.
- In Mealy Model, input effects both internal state and output directly; output changes whenever input changes and is not synchronized with clock.
- Moore Model based sequential logic circuit usually takes more states than Mealy Model.
- Mealy Model responds faster to change in input.
- Moore Model is safer to use in the sense the input transients or glitches are not passed to output.
- A Mealy Machine can be converted to Moore Machine and vice versa by suitable conversion rule.

So, if this we would like to conclude. So, we have seen that Moore model input effects internal state, and not output directly output is generated directly from the internal states ok. And it is synchronized with the clock, Mealy model input affects both the internal state and output directly output changes whenever there is a change in the input which is not necessarily synchronized with the clock; if some transients are there it will get go to the output directly ok. And usually we have seen that Moore states were required 3 states were required in Mealy and it was 4 for the given problem.

Mealy model responds faster 1 clock cycle ahead we had seen compared to Moore model. Moore model is safer in the sense that the out input does not directly go to the output. So, transient's glitches will not get passed. So, when the clock trigger comes, why which input is sufficiently stabilized at that time only the output will be available ok. And we have seen that Mealy machine can be converted to machine. Mealy machine Moore machine and also the opposite is also possible ok, by applying suitable conversion rule.

Thank you.