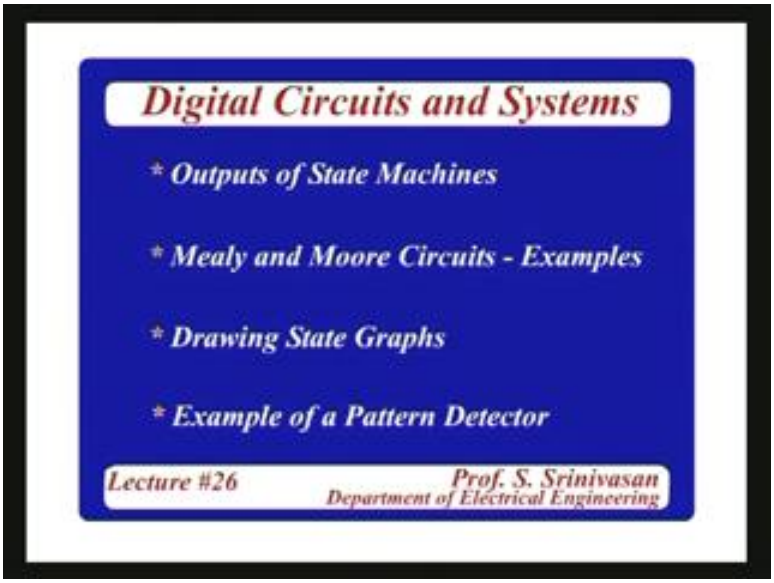
Digital Circuits and Systems
Prof. Srinivasan
Department of Electrical Engineering
Indian Institute of Technology, Madras
Lecture - 26
Mealy and Moore Circuits

(Refer Slide Time: 1:50)



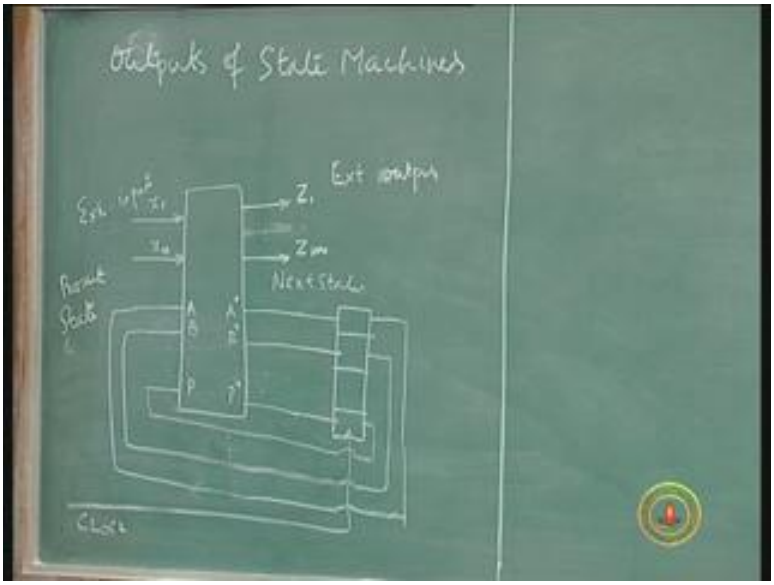
So we have been talking of state graphs and their implementation using flip-flops and combinational logic.

One thing which you have not been emphasizing until now is the outputs of state machines. When you say a machine a machine is described as state graph or state machine or a circuit basically it can exist in several states and it goes from state to state based on the present state information and the input conditions. We are able to draw the state graph based on this information and translate a state graph into a state table and then get a transition table from which we drew the Karnaugh Maps and got the combinational logic to drive the flip-flops or steer the flip-flops we called it steering logic. But we never talked about outputs that is because we took only simple examples of usually counters either sequential or non sequential.

Therefore counters do not have outputs because the states of the flip-flops themselves are the outputs of the counter. But there are circuits in which we have outputs other than the state outputs state information. Example is a traffic light controller, it may exist in several states but then there may be only three types of lights then certain states make certain lights glow so there is a need for outputs in addition to the state outputs and we also indicated this in our original drawing of the generalized state machine. If you remember the generalized state machine, so we will say outputs or state graphs, outputs of state machines.

So, if you remember in the generalized block diagram that we drew we said there can be one or more inputs called external inputs, new symbols x_1 x_n and then the state information which are called state variables which are either the present state or the next state we will call this state variables or we will say present state and next state and next state information which is stored in type of flip-flops clocked to a system clock. So the outputs of this become the input to the state machine for the next state determination so you want to call this A B C and P where these are the corresponding next state values which are stored here and fed back and they become the next state and present state after the next clock trigger.

(Refer Slide Time: 7:06)



And in addition this itself can act as outputs in the case of counters. For example, the state of these flip-flops that is the counting, the sequence through which the counter goes through, it acts as an output. For example, you want to know the number of counts that have expired number of counts that has elapsed till now then you can look at the counter states. Suppose you want to count the number of people getting into a room any time the number of people in the room is giving by the state of these flip-flops.

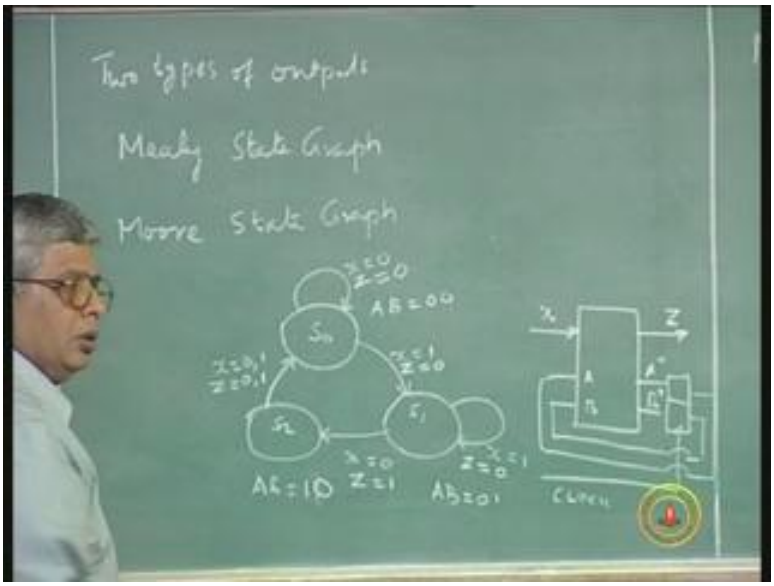
On the other hand there are certain conditions as I said in the case of a traffic light controller. In addition to the state there may be external outputs so you can call them Z_1 Z_2 Z_m may be m outputs n inputs and P state variables which are present state and the next state and you need P flip-flops in order to store the present state information and next state information fed as the present state for the next clock period.

Now when you are drawing the state graph you have to accommodate for this output which you have not done as specifically. The state graphs we have drawn so far in the examples we have worked out we have not specifically taken into account these outputs which can be external outputs. Of course outputs which are state variables themselves we don't have to have a special provision in the state graph because the state information is

anyway there in the state graph and that information also access the output information. But for such outputs which are external to the state variables we have not so far made any provision. So, what we will do today is to see how to consider this output, how to represent this output and how to take into account these outputs in our overall design. There are two types of outputs possible in a state machine and based on this the state machines are classified into two types. One is called the Mealy State Graph or Mealy Machine and the other is called Moore State Graph.

The Mealy state graph is one in which the outputs are defined for a state and an input condition. What I mean is this (Refer Slide Time: 9:15). Suppose I have a simple machine with three states only call this S_0 S_1 S_2 the circuit can exist with one input external input x and then if x is 0 circuit remains in the state S_0 and if x is equal to 1 it goes to S_1 and in S_1 if x is equal to 1 it remains in same state otherwise it goes to S_2.

(Refer Slide Time: 15:28)



Just arbitrarily drawing a state graph:

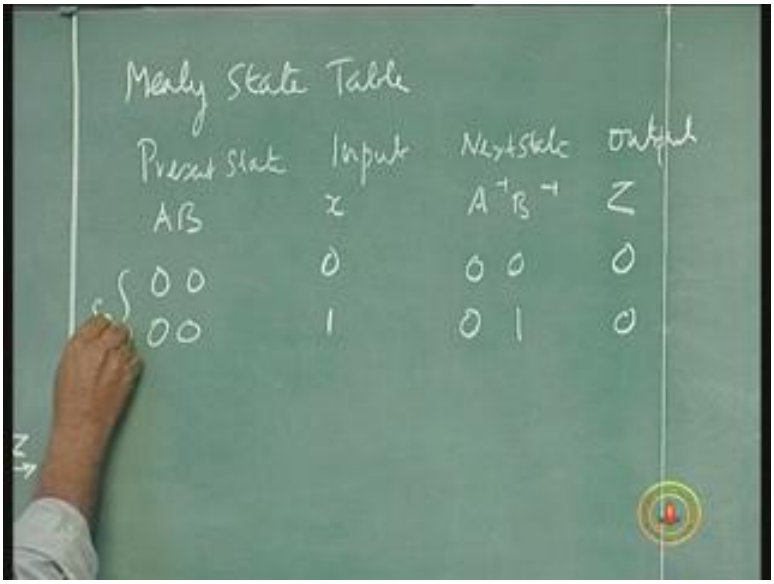
From S_2 it will go to state S_0 whether x is 0 or 1. Just imagine as a simple state graph. Now if you define an output Z for this system there will be state variables, two state variables will be there because there are three states so you need a minimum of two state variables you can call them A B, two D flip-flops so next state information will be a power plus b power plus so forth. But we have not made a provision to show this Z the output which is different from the state variables A and B or A power plus and B power plus. That can be shown in two ways. In a Mealy machine it is part of the state and the input condition. In a Moore machine it is part of the state alone. The output is defined for a given state irrespective of the input condition in a Moore machine. In a Mealy machine the output is defined for a given state and an input condition.

For example, if this is a Mealy graph and this machine is a Mealy machine then for each arrow I should define an input and output so I can say output is 0. The circuit may be in state S_0 when the input x is 0 the circuit remains in state 0 S_0 output is 0. On the other hand when it goes from S_0 to S_1 when x is equal to 1 output may be 0 or may be 1 again I have to define it, it may be still 0 but what I mean is in S_0 this output may be 0 whether the input is 0 or 1 but in both cases you should define it specifically. So I may again put x is equal to 1 and Z is equal to 0 here which means as long as the circuit is in state S_0 the output is 0 Z_0 but I should specifically say this here and here. For each arrow I should give an output value also, it is in a Mealy machine where the output is defined as a combination or output is defined as dependent on the present state and the input condition.

On the other hand, for this one I may have Z is equal to 0 and for this (Refer Slide Time: 13:06) I may have 1 again as an example. I have no circuit in mind I don't know what it's going to lead to in terms of a realization. And when circuit is S_2 whatever is the value of x it goes to S_0 but I may say when x is 0 output may be 0 and when x is equal to 1 the output may be 1. So in other words, for each arrow not only I should say what is input condition which takes the circuit from a given state to the next state but I should also mark the output one or more outputs. of course I have taken a simple example of one output it's nothing like we have we have only one output.

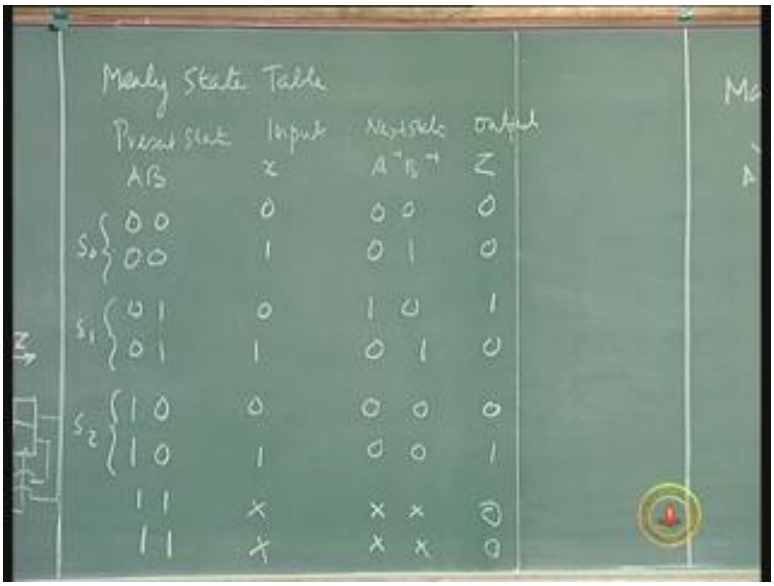
In a generalized state machine there may be several external outputs Z_1 Z_2 Z_3 m outputs will be there so my state graph will be written such that, for a given state what is the present state, what input condition will take it to the next state, under different input conditions what are the next states, and then the outputs for each of those input conditions. So my state table corresponding to this would be Mealy state table would be present state input, next state output. I will have to now give the state variables some labels so I will call them A and B I have already indicated a and b here so it's a b 0 0 0 1 1 1 or 1 0 if you want it doesn't matter. So the present state value input x, next state value output Z, the value of present state variables a and b are 0 0 here, input x is 0 what will be the next state? It is also 0 0 state, x is 0 leads you back to this state and output would be 0 for that condition. And when present state 0 0 x is equal to 1 circuit is taken to S_1 which is 0 1 state and the output is still 0.

(Refer Slide Time: 16:30)



Take state S_1 which is 0 1 state, this is S_0 state (Refer Slide Time: 16:30) 1 state if x is 0 it goes to S_2 which is 1 0 with an output of 1, output is 1 under this state, in state S_1 if the input is 0 the output is 1, in state S_1 the input is 1 it stays in the same state and the output is 0. Finally the state is S_2 the third state in which circuit can exist the state variables are 1 and 0, in 1 0 whether or not x is 0 or 1 so I will have to put separately 0 1 both goes to 0 0 as the next state but if the value of x is 0 output is 0 input is 1 and the output is 1 so I should put 0 for the output corresponding to x is equal to 0 and 1 for the output corresponding x is equal to 1, this is my S_2 state.

(Refer Slide Time: 20:37)



So this is my table from which I can draw the transition table based on the type of flip-flops I use, it's D flip-flop which is same as the transition table you know all that and then I need one extra Karnaugh Map for output Z. So I can only draw that because you know how to do the maps for A and B either it is D flip-flop in which case you draw the map for A and B or A power plus and B power plus in fact. There are other flip-flops other than D flip-flops J-K or S-R you have to go through the excitation table, draw the inputs J_A K_A J_B K_B and then draw the Karnaugh Maps for each one of those inputs and then you will do the steering logic.

Let us assume you have done all that because you have done it earlier and you know how to do it because to the output only we will draw the Z map output map, map for Z, (Refer Slide Time: 19:10) this is A B x 0 0 input 0, output is 0, 0 0 input 1 is also output 0, 0 1 input 0 output is 1, 0 1 input 1 output is 0, 1 0 0 output is 0, 1 0 1 output is 1.

(Refer Slide Time: 27:43)

The 1 1 state is not defined. Next state is not defined because input is not defined, 1 1 state is not defined so the next state is not defined. But you have to remember to put the output as 0s not don't cares and output don't care can be 1. And, I don't want an output of one when I don't need it. When you try to put 1 and simplify the map I may land up in a situation where there may be an output when it's not required. The output should be generated only when it is required and all other times an output should be 0. So do not use don't cares for output conditions and output maps.

For output maps whenever the state is not defined use 0. Whenever a condition of state is not defined make an output of 0 so that if by chance that circuit happens to be in that state because of the transients when you switch on the power for the first time the circuit may find itself in a state I don't want any output in that state because I don't want a glitch in the output because the output may be a condition of a lamp the traffic light may be a green light, I don't want a green glitch, suddenly you have a green patch and someone

tries to pass then it might end up in an accident or it could be some other mechanism of aircraft or something where you don't want an unnecessary signal there. So it is always advisable it's a good design practice to make all undefined outputs as 0s not don't cares and when you put a don't care you put it as a don't care in the map and sometimes the don't cares will be treated as 1 when you draw the map to simplify it which may result in a glitch which is not required, so a glitch itself is not required.

There is nothing like a glitch not required. So I put 0 0 (Refer Slide Time: 21:55) so my output will be A bar B C x bar or A B bar x. So this will be an additional hardware. So whatever is the steering logic I am going to get for A power plus and B power plus D_A and D_B in addition I need to have one more combinational block or a gate combination to produce outputs. So, output has to be taken into account when you are specifying the system.

When somebody gives you a system you talk about the various states the circuit is going to be in and different input conditions under which the circuit is going to change from state to state should also ask what is the output required for each of those states or each of those simple conditions and then specifically mark them in the state graph and carry it on to the state table until the transition table until the final design, this is called Mealy as I said. Mealy graph is one in which an output is defined for a state and an input combination.

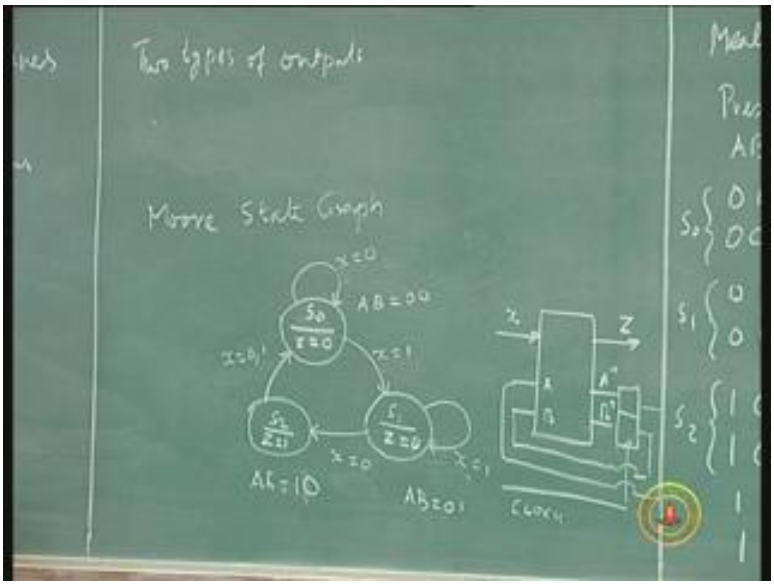
If the output is defined purely for a state and it is independent of the input the output is same for that state such a machine is called a Moore machine. If I now modify this into a Moore state machine I will have to draw Zs as arrows along with the arrows. I will define Zs inside the state circles because in a Moore machine output is defined for each state which this is similar to a counter.

For example in a counter each state is a count. A counter is a typical example of a Moore machine. But sometimes there may be outputs other than the count. Supposing for all counts less than 5 I want a light to be on, for all the counts more than 5 I want a light to be off that's an extra output which can take only two values 0 or 1 depending on the count. So the state variable itself can be an output which can also be an output in the case of Mealy machine. But in addition there may be external outputs same as the state value but dependent on the state variable. Of course when I say the output is dependent on the state the values of the state variables decide the output of course but you have to define it into a graph for each state.

So since I have a model of x as input and Z as the output so Z has to be shown separately in the state graph as well as the state table but instead of showing it along with the arrows of the inputs we will draw it along inside the circle. So now this is how you will find a Moore graph with Z is equal to 0, in S_0 state the output is 0, in S_1 state also output is 0, in S_2 state maybe the output is 1 just an arbitrary case. I am having an output called Z which is 0 here, 0 here, 1 here that should be reflected so you cant say it's a state variables so whatever is the state variable, for example in this case state variable 0 0 gives an output

of 0, 0 1 gives an output of 0, 1 0 should give an output of 1 I should indicate that condition in the state graph also design for it in the output.

(Refer Slide Time: 25:55)



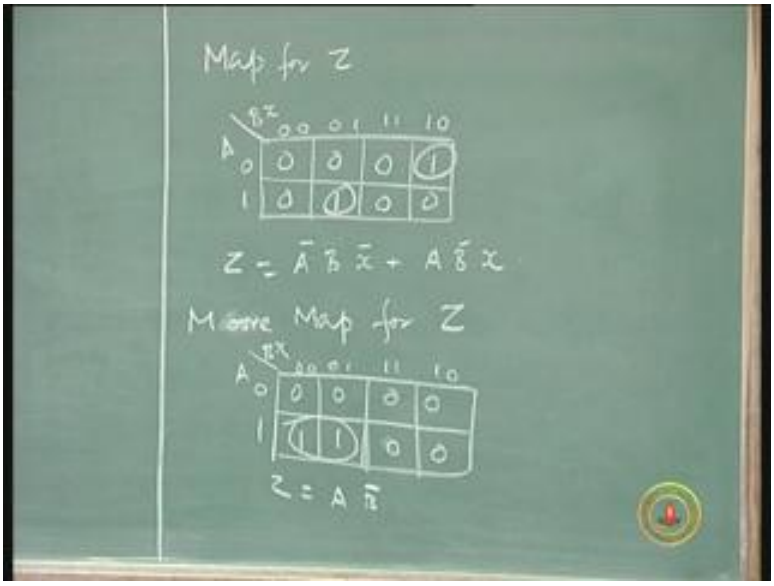
Hence now Mealy state graph has to be modified into Moore state graph which is very easy. Moore state table corresponding to Moore state graph, output is now defined for the state. for S0 state the output is 0 whether or not input is present or not, whether input is 0 or input is 1 the output is 0 that's what we mean by putting the output within the circle, it is not affected by the arrow, it is not affected by the value of input x.

Similarly S_1 state the output is 0 whether x is equal to 0 or x is equal to 1 so I will have to change this into 0. And for S_2 state in the S_2 state output is 1 whether x is equal to 0 or x is equal to 1 independent of the value of x output in state S_2 is equal to 1. But it is easier to design the circuit because you know that output is 1 for S_2 or 0 otherwise, what is S_2? S_2 is A B bar A is equal to 1 B is equal to 0 so if you take a b bar that is the output. And here also in the undefined state 1 1 even though the next state is not defined for 1 1 we will still make output 0 as I said you don't want to have glitches in a state which might come into being because of a false counter, because of a transient, because of a power supply being switched on and all that type of thing you want to make sure that no such output occurs at that time when undefined state occurs.

Moore Map for Z:

There is no need to draw this but I will do it for the sake of completeness which is A B bar. We know it's A B bar because when circuit is in S_2 the output is 1, what is state S_2? S_2 is; A is equal to 1 B is equal to 0 that is A B bar you can directly write it also from the graph. Sometimes it may not be possible because more than one condition may happen and there will be more than one output.

(Refer Slide Time: 29:29)



So the output condition is also important in the design of a state machine, outputs are important because outputs need not always be the state of inputs. State variables will tell you the state in which the circuit exists at that time but may not give you the indication of the outputs. The outputs may be independent or different from the state variables. There are two types of outputs possible one is the Mealy output where the output is defined in a particular state and in an input condition like the output in state S_0 when x is 0, the output in x S_0 when x is equal to 1 and so forth. On the other hand if an output is defined for the state alone in states S_0 the output is 0, in state S_1 the output is 1 so if it is defined like that then it's called a Moore machine.

In both cases you should indicate the outputs in the state graph carried into the state table and the transition table and into the graphs Karnaugh Maps and in the final logic that we design for the hardware.

Now with this background we will now consider systems. We will have outputs which are other than the state outputs or state variables. We will take some examples. We will start with a simple example of a state graph. So you should also know how to draw because so far I have been assuming state graphs I never told you how to draw the state graph. Of course in the state of counters you know because if I give you the count sequence you know how to draw the state graph because each count sequence is a state so put one circle and then put the count value inside and then go to the next count put an arrow so that is one way of doing it but that is a very simple way. Independently you should know how to draw a state graph.

Suppose I give you a specification of the problem the circuit will have these characteristics this behavior, this is the type of input and this is the type of outputs and this is how it should behave then will you be able to come up with this state graph that is the first thing. Once you know the state graph you can implement it. How to draw the

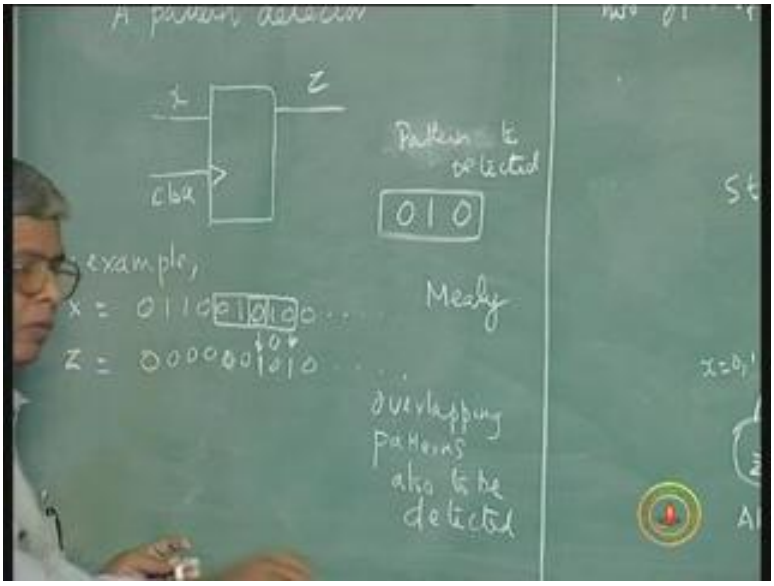
state graph? You cannot always expect somebody to draw the state graph for you, if you are a system designer the problem is given to you in word problem they call it which is in the form of a statement, they will tell you what are the inputs and what are the outputs and what input conditions will lead to what output conditions. Using this statement of the problem you should draw the state graph which is the first thing, of course you should ask a question whether you want a Mealy graph or a Moore graph and then once you have that state graph then you know the rest of the story, state graph to state table and state table to transition table based on the type of flip-flops and once you have the transition table it's a question of drawing Karnaugh Maps and drawing a logic for that.

So the exercise is now to start with how to even draw the state graph. So we will have to do some examples. We will try to take a simple example, see how it works. suppose I want a detector pattern suppose I am seeing whether data is coming in serially one bit at a time and when a particular pattern occurs I want to know that that could be a password, you may send a bit stream and embedded in the bit stream may be a pattern you are looking for so when that pattern comes you know that you have identified the correct sender or recipient and from then on you do whatever you want to do with that.

So a simple pattern detector could be….(Refer Slide Time: 34:25) so I am going to assume that we are having a single input x and a single output Z, I don't know the number of states now because I need to draw the state graph and only when I draw the state graph I know how many states are there and then only I know how many state variables will be there but I know it's a state machine because the input is coming one at a time one bit at a time it is a clocked input. so I know there is a clock here so the input occurs like this; for each clock pulse input may be like this (Refer Slide Time: 35:05) you don't know in what way it's going to come.

Let us simply say whenever a state 0 1 a pattern 0 1 0 in this sequence occurs. Supposing a sequence 0 1 0 occurs I should know it and then indicate it by an output high and for the rest of them output is 0. so output will be 0 normally but when a pattern of 0 1 0 is detected by this circuit that you are designing for one clock period coinciding with the third 0 the output should be 1 and then again the output will go back to 0. So my output will be 0 0 0 0 I don't want I have not even put one I should probably try to put one 0 here otherwise we are never going to get it. now 0 1 0 this is the pattern I am looking for which has been identified and I said along with the third bit of the input pattern it's a time so the first 0 comes then 1 comes then 0 comes and along the three bit of the input I want an output 1 and as long as the third bit of the input exists output will be one and then when the next bit output has to fall back to 0.

(Refer Slide Time: 40:23)



So along with this this should be an output or 1. This is only a typical sequence, this is not the sequence. I may get the sequence in any order. I may get 0s and ones in any pattern but wherever there is an embedded 0 1 0 I should be able to detect it. The circuit is to be general. This is only an example sequence not the specification. So, for example this is an input sequence may be like this for which I need an output sequence like this. This is the sequence to be detected or pattern to be detected (Refer Slide Time: 37:40). Now again 1 0 1 now the question is after 1 0 1 what happens? Again should it reset or…?

If I take this 0 1 0 as one output and then there is 0 1 0 so there is one more 0 1 0 here. Whether this pattern also is considered as another pattern or is it because it's overlapping with the previous pattern should it, so that also you should decide. All these are very clear in this specification. You can't just jump to a design board or the bread board as soon as somebody tells you a problem. If the problem is vague you should ask questions, everything should be clear. Do you want a Mealy machine or Moore machine, do you want an overlapping pattern also to be detected or only clear pattern should be detected, all those questions you should ask before hand because your circuit will be vague otherwise. You should always understand what the person wants before you go and design the circuit.

So let us now assume that the overlapping pattern also should be detected. Whenever there is 0 1 0 whether or not as separate pattern even if it's a part of another detected pattern 1 0 1 I should still be able to detect it, if that is the answer then I should put 1 here again and there may be any number of 0 1 0s. Wherever there is 0 1 0 whether as an independent pattern or a overlapping pattern corresponding to the third 0 of that 0 1 0 I need an output 1 for one clock period that is the problem statement and a sample sequence of input and output. With this I should go to the drawing board and then draw the state graph, how do I do that. The circuit as to exist in a state and we have to assume that the circuit is existing in a state and then the input comes and then whether it looks like a 0 and if it is 0 how is it treated and if it is 1 how is it treated and all that.

So the next step is to draw the state graph for this. Let us also put a Mealy machine to start with. So these are my input specifications (refer Slide Time: 40:00), this is the pattern detector, one input and one output with a clock, pattern to be detected is this and I want it as a Mealy machine and I want overlapping patterns also to be detected.

With this I go to the next step of the state graph drawing. I may not draw the best state graph but I should start somewhere and I get going. So the state graph for this problem is this. Any design problem you will have more than one solution and one of them may be better than the other an optimal solution and all that. You don't worry about whether what I do is the best solution for it at least to start with. Later on you should improve and then probably give the best possible solution. So let us say the circuit exists in a state called S_0. It's a Mealy machine so there is no output within the state and this S_0 is a state in which the circuit normally exists so it is a reset state. The power starts and when you power up the circuit goes to S_0 state.

Let us assume that you have to start somewhere and then input comes input can be any one bit, any one bit coming at a time so input is x the input can be either 0 or 1. If it is 0 I should be careful because it could be a part of 0 1 0 so one I have no use for it so discard it.

So what I will say is if 1 occurs the circuit can remain in same state as reset state, it doesn't matter to me even if any number of 1s occur so any number of 1s occurring has no meaning because the pattern has not started and you can continue to be in S_0 state and no output is also required so this is the way to put this. Usually they put x stroke Z, the input and the corresponding output is written instead of putting x is equal to 0 is equal to 0 this is the normal way of representing this in a state graph in text books you will see the value binary value with a slash and another binary value, the first binary value is input value and the second binary value is the output value it's a convention so first is the input and second is the output.

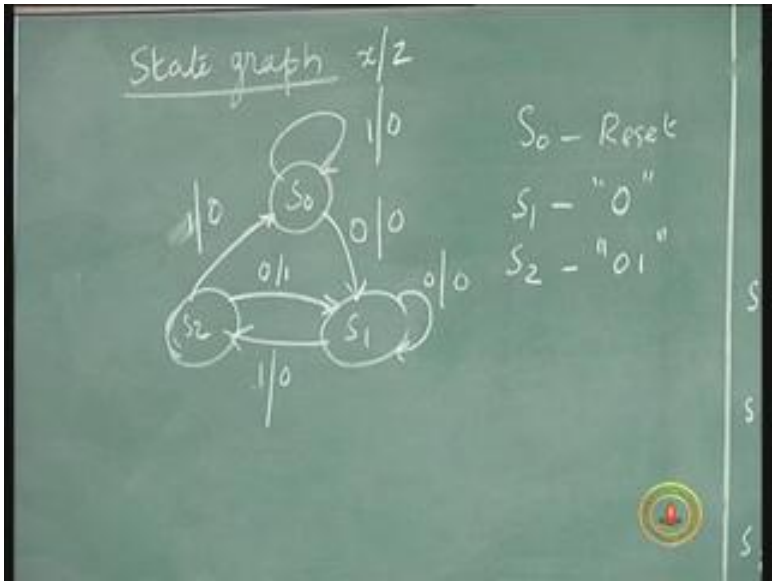
On the other hand if x is equal to 1 I mean 0 I will have to be careful because this could be part of 0 1 0 so I am going to remember this state as a state in which 0 has already occurred, these are little notes I am keeping. And even now the output is 0 because output will be 1 only when 0 1 0 is…., so first 0 is not going to give you any output. But I am sure in a state where I have detected 1 detected 0 so to that extent I have progressed.

In S_1 again a next bit comes and it could be 0 or 1. If it is 1 then I am going back to square one, 0 1 is part of a sequence. if 0 occurs in this any number of 0s has no meaning only when the last 0 is of interest to us, 0 0 0 0 I have already detected the first 0 so a second 0 has no meaning because the latest 0s will be stored and rest of the 0s will be discarded so if 0 occurs I will continue to be in same state with same output whereas if 1 occurs this could be the 0 1 of 0 1 0 so I will have to go to a new state called S_2 so 1 is the input but again output is still 0 because 0 1 itself cannot give you the output but only 0 1 0 can give you the output so 1 is a new state but output is still 0 so S_2 state will be a state in which 0 1 will be remembered. Now in S_2 again a third bit can be 0 or 1. The third bit

is 0 you have got the 0 1 0 pattern completed so output should be 1 but where should I go, I should go to S_1 which stores the first 0 because this 0 could be the starting of a next pattern. So at this time when x is 0 I will go here but it will give an output of 1 because I have completed 1 0 1and it will go back to S_1 because S_1 is going to remember the latest 0 and this latest 0 could be the start of the next pattern as in this case.

On the other hand if there is a 1 there then there is no point because 0 1 1 has no meaning so I have to start all over again so I can as well go to the reset state without any output, this is the state graph.

(Refer Slide Time: 45:56)



Does it work? It may not be the best state graph, it may be possible to do it simpler. later on we will see whether there is anyway like the Karnaugh Map reduction or Boolean algebra where you try to simplify and finally you land up with something and you may or may not know whether it is right so then you go for Karnaugh Maps. Similarly in the state graph also there are some techniques to see whether this is the best possible state graph the minimum possible state graph.

We may or may not have time to do that but that doesn't matter. We are only starting to design our own circuits. We are now able to translate a word statement into a state graph and then into a Karnaugh Map into a hardware diagram which hopefully when you are given those parts, components and a bread board you will able to wire it up and even make it work which is what you will go and do in a lab, next semester. So the idea is, from somebody's description of the problem to the statement of the final realization of the circuit is that it has to go through all the stages and this is the most critical stage because understanding the problem and translate it into a graph so from here on it is a routine procedure from state graph to state table and state table to transition table and transition table to Karnaugh Map, Karnaugh Map to gates, gate to buying them and putting them

into bread board and testing it is all routine. The most critical part of this design is to understand the specs and translate it into a workable state graph.

So this is the state graph and from here you know how to proceed because you know from here you have to assign states for example in can say S_0 is equal to 0 0 A B are the state variables, S_1 is 0 1, S_2 1 0 and then we go to state table and then I can use D flip-flop or I can use J-K flip-flops and I know how to draw the transition table, how to draw the Karnaugh Maps, how to get the output map, how to put them together and show it as a drawing on a design sheet and then how to translate it into a hardware when you get these components.

But this is the thing the state graph formation. The circuit can exist in three states. In the first state it is waiting for the bit to come, the bit can be 1 or 0, if it is 1 there is no use for it, if it is 0 it goes and remembers that, if it is 0 there again the same 1 0 has to be remembered so there is no need to go to the new state it continues to be in same state, if 1 comes it goes to another state and will remember 0 1 in this if 0 comes you have completed the pattern but remember the last 0 and on the other hand if 1 comes there is no use for any of them because now 0 1 1 has no meaning so you have to start all over again and you have to go here and no output except in this case when 0 1 0 is completed and in all other state the output is 0.

And I want to take it as an assignment to complete this state graph into hardware design. Take this assignment and draw the state table and use J-K flip-flops and get a transition table and draw the logic diagram for the steering logic of the flip-flops as well as the output Z. We will see more examples in the next classes.