

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 - 29
State Machines 1

Welcome to week 6 of this course Digital Circuits and Systems. And in week 5, we saw lot of things and so there was quite a bit of contents about Seemas vlc circuits, what the delay of a Seema circuits and so on. And also believe that the homework that is given in week 5, itself is quite hard. So, what I want to do is takes some time to solve the problem, so that you understand all the concepts and anyway the deadline is pass the usual deadline.

So, the deadline is Sunday for a home work 5, so I suggest that you take a time and solve the problem and understand all the little concepts, because they will be useful for you if not in this course somewhere else later. So, in this week what we are going to do is, we are going to look at state machines. So, we are going to look at finite state machines and how sequential circuits, what are the special class of circuits called state machines and how to design them and so on.

So, it is a very fundamental thing to many digital design courses as well as digital systems itself. So, it is important that you pay attention to the details and I am going to show you a lot of different tricks of the trade, so called tricks of the trade. So, I suggest that you follow the material carefully and the homework assignments that you see this week will also reflect the kind of problem that are there in the videos.

So, in the last week, we left off with this applies, where you saw that if I want us a sequence to be detected. Then we can put in a shift register and we can do a few things about the shift register itself. And however, what would happen is, if the patterns become complex or if there are many, many patterns that you have to look for, then the approach that you have using the shift registers will not work, and shift registers also solve only a small class of problem called the detection problem.

So, we will see a several examples now, and then we can go back and think about where the approach that we used in the last class fails. Let us start with a design problems, so

before getting in to this week design problem, there is one class of flip flops that I did not of talk about so far.

(Refer Slide Time: 02:41)

CLK	J	K	Q*
↓	0	0	Q
↓	0	1	0
↓	1	0	1
↓	1	1	Q

J: Set
K: Reset
J=K=0: Keep old value
J=K=1: toggle

This is called the JK flip flops. So, the J and K does not really mean anything, it does not stands for any ones name or it does not stands for any particular thing. This is a flip flop that was in use for many, many decades, but JK flip flop is not something that is commonly used in designs now a days. So however, it is good to know what the flip flop is. So, as any flip flop is you have either a positive edge trigger or a negative edge trigger flip flop.

In this case, let us assume that we have a positive edge trigger flip flop and the characteristic table is as follows. If both J and K are 0, then the output q, so the output q star or the next state of the flip flop is the same as the current state. If K is 1 and if J is 0, the output is 0. If J is 1 and K is 0, the output is 1; however, if both are 1, then the output is q bar. So, this is q bar, this should have been q bar or complement of q, so this should have been q bar.

Now, in summary we can say that J is equivalent to set, K is equivalent to reset, if both K and J are 0, keep the old value, if J equal to K equal to 1, then toggle the old value. So, this is the set up behind the JK flip flop. So, this table has to have q bar here, instead of q and this is the standard flip flop that you will see in many, many text books. So, I just want you to be introduced to it, so you do not go back after the course and not know, what the flip flops is.

(Refer Slide Time: 04:33)

Example:

1. **Input/Output logic equations:**

$$\begin{aligned} J_0 &= Q_1 \bar{X}; & K_0 &= Q_1 X; \\ J_1 &= X; & K_1 &= \bar{X} \oplus Q_0 = XQ_0 + \bar{X}\bar{Q}_0 \end{aligned}$$

MPTEL Analysis and Design of Sequential Logic Circuits 3

So, what I am going to do in the next few minutes is, take this particular circuit here, which is designed using a JK flip flop. So, it is a flip flop and it has a common clock connected to it, so it is a synchronous circuit and to this synchronous circuit, there are two inputs and two outputs, the J and K input are there and Q and Q bar are the outputs of the flip flops. There is a single input called X to the whole circuits itself.

The sequential circuit takes X as the input and the outputs are Q naught and Q 1 itself. So, let us say this is the circuit and you are asked to find out the state machine that this circuit describes. So, I already described what a state machine in the previous video. So, let us see how to approach this problem of given a circuit, how to analyze the video, analyze the circuit, so that you can get a state machine. So, this is one part of the problem.

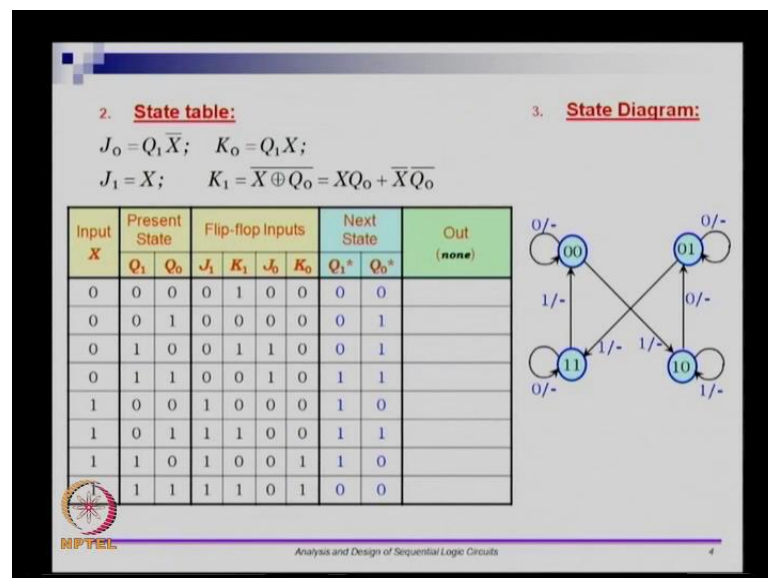
How to figure out what somebody else has done, this is something that you may have to do in your work also, there is a circuits that is already given, you are trying to figure out what it is doing. So, let us see this as a circuit, the first thing you have to do is you have the J and K inputs for both the flip flop Q naught and Q 1. You can now go and write equations, so can you write the inputs and output logic equations. So, if you look at J naught, it is Q 1 AND X complement, then K naught is Q 1 AND X, J 1 is X itself and K 1 is X, X NOR Q naught or X XOR Q naught complement. So, this is, these are the inputs, this is something that you can fairly quickly derive based on the circuit itself.

If you look at this circuit, this resembles the canonical circuits in some sense. You have

the state registers which are given by Q 1 and Q naught and the input forming logic or the next state logic is here. So, based on the current state and the current input, you drive J and K inputs, so these J and K inputs will get register. Based on J and K, Q naught and Q 1 will change in the next state. So, if you supply current state here, then this logic takes care of forming the next state and when the clock pulse comes in, the next state gets register.

There is no output forming logic here, Q naught and Q 1 are directly given us output, so these are just wires, there is no gates there, there is just wires there. So, it is a valid synchronous sequential circuit and from the canonical module, the only thing that has degenerated here is, there is no output logic explicitly. And to be more specific, since the output is Q 1, Q naught explicitly, it is only the flip flop that are controlling the output. So, this is actually a Moore machine, I mention this in the last video, this is actually a Moore machine, where the output depends only on the states, but not on the inputs directly. So, this is a Moore machine. Now, I want to find out what the state diagram for this is.

(Refer Slide Time: 07:43)



So, once you have the equations, so let us say I have this presents state and the inputs. So, you have the presents states could be 0 0, 0 1, 1 0 or 1 1, because there are two flip flops, they can be in four different combinations and X equal to 0 and X equal to 1 are two combinations of X. So, over all from the input side, we have X Q 1 and Q naught, you will... There are three inputs, which mean there are eight combinations.

So, these are the eight combinations and if you go and look at, what the output of the next state logic is, the next state logic actually drives J_{next} , K_{next} , J_1 , K_1 based on Q_1 , Q_{next} and X . So, based on these 3 inputs, you derive J_1 , K_1 , J_{next} , K_{next} . So, for instance if I want to drive, if you want to derive this column J_1 , J_1 is X itself, so I take the value of X put it there, K_1 happens to be $X \oplus Q_{next}$.

So, it is this column vector XOR with this column vector and that is XNOR with the third column vector, so that is K_1 , similarly you can derive J_{next} and K_{next} . Now, that you have J_{next} and K_{next} and J_1 and K_1 , you can actually determine what the next state is going to be. So, let us look at Q_1^* and Q_{next}^* , these are the next states. So, if you place J_1 , K_1 , J_{next} , K_{next} before the clock gets arrives, then when the clock gets arrives the J and K values are sample.

So, your clock is a triggering signal, J and K values are sample and based on the semantic of J and K , whatever the values of J and K are the flip flop values Q_1 and Q_{next} will change. So, let us see how this is going to happen, so let us take the very first one. So, J_1 is 0 and K_1 is 0, so if I look at this combination, K_1 is a 1, which means it is a reset. So, whatever the value of Q_1 is, you do not care about it, you have to reset and if you look at J_{next} , K_{next} , it is 0 0.

So, 0 0 is keep the old value, you go and look at the old value is 0 itself, so we have Q_1^* is 0 and Q_{next}^* is 0. Let us look at the next row, the next row says J_1 , K_1 is both 0, if both are 0, then keep the old value, the old value is 0 itself. So, this is 0 and for J_{next} and K_{next} , it says keep the value of the flip flop Q_{next} at the present, whatever the present state is, continue in the same state. So, 0 0 will take the present state, which is 1 and you get 0 1 here.

So, remember I am not doing this is a sequence, so these are combinational inputs, for each of the combinational inputs for X , Q_1 and Q_{next} , the combinational circuit that drives J and K will drive these values. But, when these values are put in as input to the JK flip flop, the JK flip flop in the next clock cycle will have this. So, the next clock cycle when I have 0 1 here, for instance this is just saying that flip flop 1 will have a 0 and flip flop 0 will have a 1.

Let us see the next one, so J_1 , K_1 is 0 1 which means, whatever the values of Q_1 is do not care, but go and reset. So, 0 1 means K is equivalent for the reset, so reset Q_1 flip flop and set Q_{next} flip flop. So, you have when you reset Q_1 flip flop, it is 0, when

you set Q naught flip flop, it is 1, so you get the state 0 1. If you continue this way, you can get all the other values of Q 1, Q naught. So, that is what you have here so far, so let us look at this particular row.

So, in this particular row, if X is 1, Q 1 is 0 and Q naught is 1, J 1 K 1 are both 1. So, when both are 1 toggle 0, that gives you 1 and J naught K naught happened to be a 0 for the equations. This keeps the old value itself which is 1, so that is what it is and in this circuit there is no output, so we just leave this as a blank column. Now, what we have is, if you forget the inputs for a while, based on the present state and the input X, you can derive the next state.

So, forget the flip flop inputs now, you do not need it any more to draw the state machine, because you already derive the vectors Q 1 star and Q naught star. Once you have that, you can forget these 4 columns in the middle. So, based on the input and the present state, you can get the next state and output we know is just Q 1 star and Q naught star itself, we are not explicitly writing that. So, this is what you want for a state machine.

You want, if this is the input, this is the present state, this should be the output and so on, let us do this. So, there are 4 states 0 0 zero 1 1 zero 1 1 these are the 4 states here, 0 0, 0 1, 1 0 and 1 1, let us see you start with those circles. Then, you go and looked at non input 0, where do you go? So, you go to 0 0 itself. So, you draw an arc on 0, go to 0 0 itself, then 0 0 on 1 goes to 1 0 state. So, 0 0 on a 1 goes to 1 0 state. So, you draw this arc and this arc by looking at just the 0 0 state.

So, if I look at each one of these things, these are all giving me states and these are all giving me the inputs on which I have to make a transition and these two columns will tell me, to which circle or to which state I should be going. So, let us look at this row for an instance. What this rows says is, in the state 0 1, if you get 1 as an input, you should go to state 1 1. So, in the state 0 1, if you get 1 as an input, you go to the state 1 1.

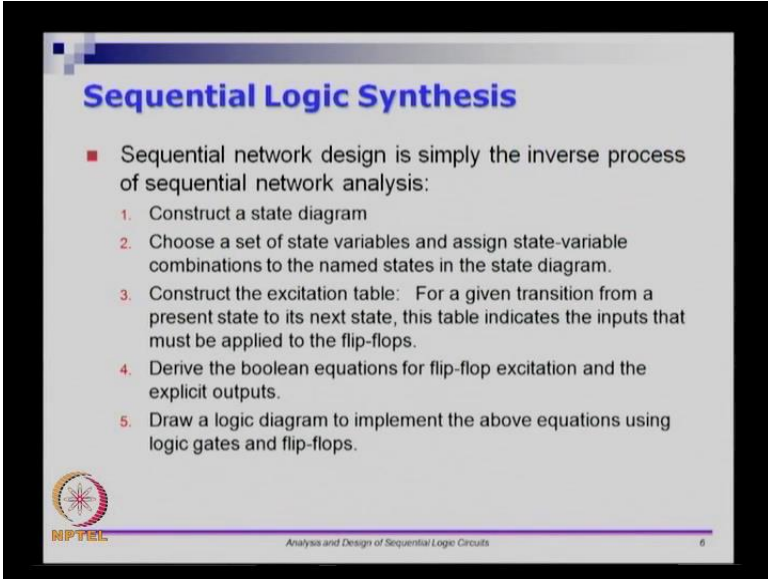
So, there are four different states and there are two transitions possible from each state, the transition from, transition using 0 and transition using 1, you will see that there are 8 arcs here. So, an all the input combinations, you go to 1 state or the other, this is the state machine that describes the circuit that is given earlier. So, there is a fairly straight forward and simple example, it was easy to get different equations and since, so on.

So, in a real world the state machines are not just two bits, it gets lot more complicated.

And if you have to analyze the circuit, you have to do this process of getting all the equations, finding out what the flip flops are and based on the flip flops, you have to derive the next state. So, let us move on to the next part of the different set up with the state machines namely called the sequential logic synthesis. So, what we did so far is sequential logic analysis, given a circuit what it does.


Now, the question is given a problem, how you design a circuit. This is something that is very common. So, the problem is specify to you in English and you are supposed to derive a circuit for it. So, this is called the synthesis problem.

(Refer Slide Time: 15:25)



Sequential Logic Synthesis

- Sequential network design is simply the inverse process of sequential network analysis:
 1. Construct a state diagram
 2. Choose a set of state variables and assign state-variable combinations to the named states in the state diagram.
 3. Construct the excitation table: For a given transition from a present state to its next state, this table indicates the inputs that must be applied to the flip-flops.
 4. Derive the boolean equations for flip-flop excitation and the explicit outputs.
 5. Draw a logic diagram to implement the above equations using logic gates and flip-flops.

 Analysis and Design of Sequential Logic Circuits 6

And the steps are actually straight forward, but it is the inverse process of analysis that we did earlier. So, in analysis you end up with the state machine, in synthesis you start with the state diagram. So, in the analysis you ended with a state diagram, in the synthesis you start with a state diagram. You construct the state diagram based on the English description that is given to you. Once you have the state diagram, you go and choose a set of state variables and assign combinations to them.

So, we will see an example in a little while, this sentence will makes sense. Then, whenever you are told to construct a circuit, if you are going to synthesis the state machine, you should be also given constrains on what flip flops you can use. So, use the flip flops excitation table and derive the transitions and get the Boolean equations for it and draw the logic diagram. So, there are five different steps, but we will see them, one after the other in a little while.

So, this slide is there, so that you can go back and look at the steps and see, how to do it for a problem by yourself. So, before we going to the actual problem, we need to derive, we need to know what an excitation table is.

(Refer Slide Time: 16:37)


Flip-flop Excitation Tables

- Excitation tables for D and T flip-flops:

Q	Q*	D	T
0	0	0	0
0	1	1	1
1	0	0	1
1	1	1	0

- Excitation table for JK flip-flop:

Q	Q*	J	K	J	K
0	0	0	0 or 1	0	X
0	1	1	0 or 1	1	X
1	0	0 or 1	1	X	1
1	1	0 or 1	0	X	0



HPTCL

Analysis and Design of Sequential Logic Circuits

7

So, an excitation table in some sense is the inverse of the analysis or the characteristic table. So, in excitation table what you get is, if you are given Q and Q star, you are asked to guess, what the input of the flip flop must have been. So, for instance if I tell you that the present state of the flip flop is 0 and the next state is 0 and if I tell you that it is a D flip flop, then the question is what should have been at the input, so that you get 0 in the next state.

So, for Q star, for a D flip flop, so the previous state is 0, the current state is 0 also, which means you should have put D at the 0 input in the previous state, so that the flip flop output will be a 0. So, this is called the characteristic, the excitation table. So, let us look at this row, if I, if the current state 0 and the next state is 1, I should place 1 at the end of the current cycle. So, that the next state will register a 1, so that is this value here.

Similarly, so since D is just delay flip flop or what you have to do is, whatever is present in the input will come as the next state. So, you can just copy the next state that should be the input, that is one way to look at D flip flop. T flip flop on the other hand is interesting. So, 0 to 0 means you are not toggling, so T should have been 0, 0 to 1 means you are toggling, which means T should have been 1, 1 to 0 means you are toggling.

So, T should have been 1 to toggle, 1 to 1 means you stay at the same place, so it is 0.

So, if I give you current state and next state, in the excitation table you are actually writing down, what should have been the value of the inputs to the flip flop, so that the flip flop will go to the corresponding next state. J K flip flop is slightly more involved. If I give 0 0 as input, so either, what is essentially means is, remember J is equivalent to set and K is equivalent to reset.

So, definitely you should not be setting the flip flop, because if I, if J input, if the current state is 0 and if J is 1, then this might actually go and change 0 to 1. So, you do not want that, so you want J input to be 0 and K input can be 0 or 1. The reason is, if K input is 0, you are going to keep the old value itself, so this is a 0 0. So, if I want to go from 0 to 0, if I put J K inputs as 0 0, it will keep the same state or if I put a 1, K is equivalent to reset.

So, it is okay to go and reset the flip flop which is already at 0 to be at 0, itself. So, equivalently J K is 0 X, so J should be 0 and K is a do not care. Whether K is 0 or 1, the flip flop will go to 0 itself. So, if I want to go from 0 to 1, you want to set the flip flop and reset should be, reset is this K flip flop, K input is a do not care. So, let us see, how this works. If J is 1 and K is 0, then that is equivalent to just saying set, so Q will go to 1.

If J is 1 and K is 1 that is equivalent to saying toggle the flip flop. So, when you toggle from 0, you go to 1, so that is equivalent to 1 X. Similarly, going from 1 to 0 requires X 1 and 1 to 1 requires X 0. So, you can forget these two columns in the middle, Q, Q star, J, K the excitation table is this table without this J and K columns in the middle. So, that is the excitation table for a JK flip flop.

(Refer Slide Time: 20:26)

Example:

- Design a 2-bit binary counter using D flip-flops

□ **State diagram:**

MPTEL Analysis and Design of Sequential Logic Circuits 8

So, let us take a small example, we let us design a 2 bit binary counter using D flip flops, the state diagram is very simple here. So, 0 0 in the next clock cycle should go to 1, in the next clock cycle it should go to 1 0 and the next clock cycle it should go to 1 1 and finally, it should go to 0 0. So, there is no input here and the transition means on every clock pulse, you go to the next one. Let us see how to go about doing this.

(Refer Slide Time: 20:54)

■ **Excitation table:**

Input (none)	Present State		Next State		Flip-flop Input	
	Q_1	Q_0	Q_1^*	Q_0^*	D_1	D_0
	0	0	0	1	0	1
	0	1	1	0	1	0
	1	0	1	1	1	1
	1	1	0	0	0	0

MPTEL Analysis and Design of Sequential Logic Circuits 9

So, let us look at, how to use this information and derive a circuit. So, we have the present state Q_1 and Q_0 and we have the next state Q_1^* and Q_0^* . So, from 0 0 you want to go to 0 1, from 0 1 you want to go to 1 0, from 1 0 you want to go to 1 1 and from 1 1 you want to go to 0 0. Now, we have to do the inverse process of

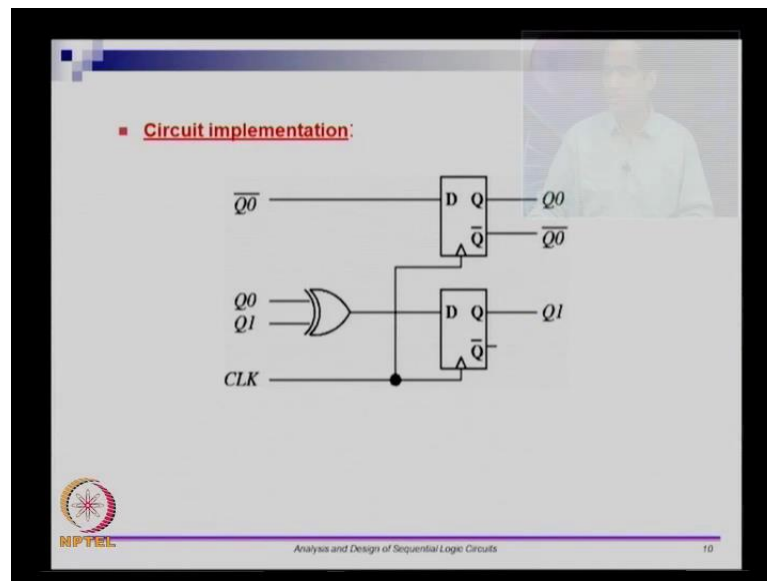
using the excitation table. If I want to go from 0 to 0 for this flip flop Q 1, then I should have place D 1 at the input. So, that is what the excitation table gives us.

Excitation table for D flip flop could tell us, if I want to go from 0 to 0, I should place a 0, if I want to go from 0 to 1, I should place a 1. So, let us look at this row, this one is suggesting that if the... In this Q 1 flip flop should go from Q naught to 1, which means you should place 1 at the input, so that it can go from 0 to 1 and for Q naught it says, it should go from 1 to 0 and if you want to go from 1 to 0, so D naught must be 0.

So, you can derive this and this using the 4 columns here. So, once you get these 4 entries, so remember this actually happens in combination. So, Q 1 Q naught is going from is 0 0 to 0 1, it would go from to 0 0 to 0 1, if you place 0 1 in D input and 1 at D naught input. Similarly, Q 1 Q naught will go from 1 1 to 0 0, if you place D 1 at 0 and D naught at 0, so that is the meaning of this table. Once you have D 1 and D naught, you can forget this, these 2 columns about the next state, because we want the relationship between present state and the flip flop inputs.

Automatically, flip flops inputs will dictate what the next state is. So, you look at the four combinations in terms of Q 1, Q naught and you looked at D 1, you get karnaugh map for it, D naught you get a karnaugh map for it. So, the karnaugh map entries are shown here and the groupings are also shown here. So, it looks like D naught is Q naught complement and D 1 is Q 1 XOR Q naught. So, that is derived from these two vectors here. Once you have these two vectors, then now we are ready to do a circuit. We know how to do combinational circuits, so we will go and do a combination circuit.

(Refer Slide Time: 23:22)



So, I will put the circuit here. So, there is a flip flop, there are 2 flip flops $Q0$ and $Q1$ and the D input of $Q0$ flip flop should be $Q0$ complement and the D input of $Q1$ flip flop should be $Q0$ XOR $Q1$. So, that is what you have here and if your flip flops $Q1$ and $Q0$ can both be tapped, then you can connect this directly here. If $Q0$ cannot be tapped, then you need to take this, put an inverter and that will give you $Q0$ complement and in this circuit, $Q0$ and $Q1$ are directly going to the external world.

So, you can see that, now this is resembling the canonical circuit. Except that, so you have a state register, you have the input forming logic, there is no output forming logic, it is just the wires. So, it resembles the canonical circuit, except that you do not have the output forming logic and the input forming logic on this line is, either the line itself or it should be an inverter. So, the combination is the input forming logic and the combination for this is the output forming logic.

So, what we did is something very simple, we started with analysis and now what we did is synthesis. So, given a design problem, you draw the state machine for it, once you draw the state machine, you know the state machine has these labels. You put the present state and next state, use the current whatever flip flop you have, derive the inputs, the D inputs in this case. And you take the D inputs and then you do the Karnaugh map, you get this.

So, what I suggest is, you try and do the same thing with T flip flop, so do the same counter with the T flip flop. So, the only thing that will change is, ((Refer Time: 25:11))

this flip flop inputs, T_1 and T_{naught} you should look at the excitation table for the T flip flop. I will suggest that as a home work and so this bring me to the end of this module. So, in the next module what we will do is, we will look at the state machines in a lot more gory detail. So, this is just a sample for state machines. In the next few videos, we will see that these state machines are discussed in a lot more detail and in a lot more rigger. But, do this little home work, before you move to the next video.

Thank you and I will see you soon.