Switching Circuits and Logic Design

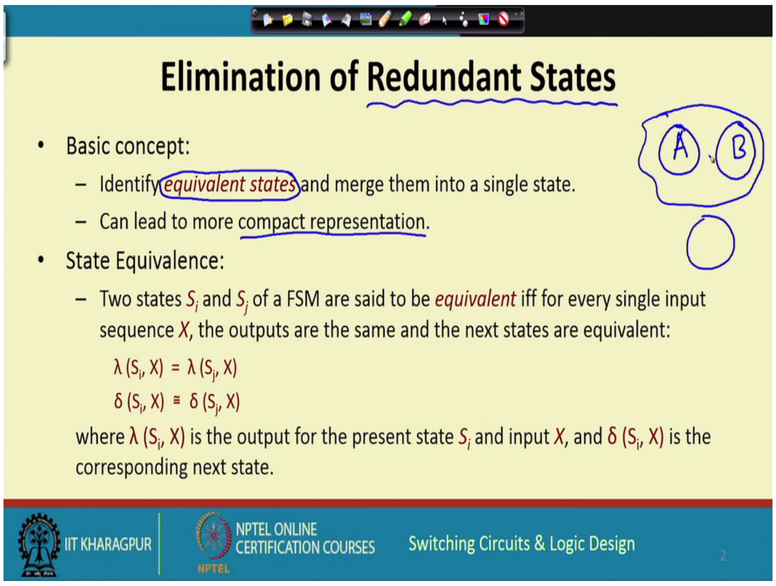
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 40
Minimization of Finite State Machines (Part- I)

If you recall in the last few lectures, we had discussed number of things regarding the design or synthesis of finite state machines. Now, we talked about the broad categories of finite state machines, the milli machine and the mode machine and we showed the different steps that we typically follow in the synthesis or the design of such finite state machines. Like the creation of the state table or the state transition diagram then creation of the so called excitation table then the transition and output table and finally, the functions are minimized and the combinational circuit part of the sequential circuits are synthesized.

Now, one thing we just mentioned, but did not discussed how to do it; given the specification of a finite state machine let say in the form of a state table; how to minimize it? There may be some states which are redundant or there may be states which are equivalent; we can possibly merge the 2 states into a single state that will make the state table smaller and in the process the final sequential circuit or FSM will also be more compact. So, the topic of our discussion today is Minimization of Finite State Machines the first part of it.

(Refer Slide Time: 01:49)



Elimination of Redundant States

- Basic concept:
  - Identify *equivalent states* and merge them into a single state.
  - Can lead to more compact representation.
- State Equivalence:
  - Two states S_i and S_j of a FSM are said to be *equivalent* iff for every single input sequence X, the outputs are the same and the next states are equivalent:

    $\lambda(S_i, X) = \lambda(S_j, X)$

    $\delta(S_i, X) \equiv \delta(S_j, X)$

  where $\lambda(S_i, X)$ is the output for the present state S_i and input X, and $\delta(S_i, X)$ is the corresponding next state.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | Switching Circuits & Logic Design

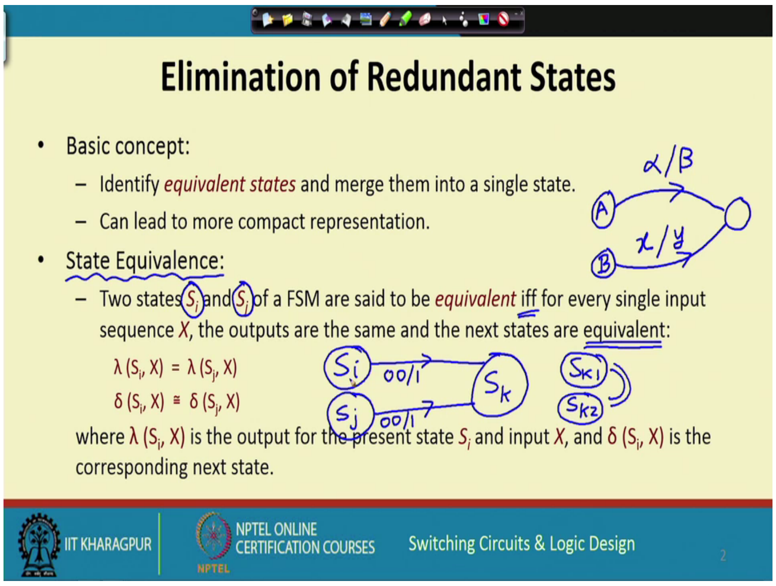
Now, let us see the basic idea the basic idea; one thing let me tell you that the kind of sequential circuits or sequential machines that we are discussed or discussing so far, they are the so called synchronous sequential circuits. Synchronous means there is the concept of a clock, the clocks are fed to all the flip flops that are used to implement the state variables and all state changes occurred in synchronism with the clock pulse.

So, when the clock pulse comes only then the states will change and it is expected that the inputs that are coming from outside; they will also be applied in synchronism with the clock. So, we are continuing with that assumption that our sequential circuits or FMS's are synchronous in nature ok. So, here we are talking about the issue of elimination of redundant states. Now the concept of elimination of redundant state is to identify something called equivalent states.

So, the basic concept is we want to identify equivalent states for example, in a finite state machine; we can see or we can find out the 2 states let us say A and B, they are equivalent. So, we can merge them together into a single state that will make our FSM smaller and that is expected to lead to a more compact representation.

Now, when you talk about equivalence of states let us try to understand what we exactly mean by that. 2 states we are saying that they are equivalent; now you recall in a state transition diagram for what we have?

(Refer Slide Time: 03:52)



Elimination of Redundant States

- Basic concept:
  - Identify *equivalent states* and merge them into a single state.
  - Can lead to more compact representation.
- State Equivalence:
  - Two states S_i and S_j of a FSM are said to be *equivalent* iff for every single input sequence X, the outputs are the same and the next states are equivalent:

$$\lambda(S_i, X) = \lambda(S_j, X)$$
$$\delta(S_i, X) \equiv \delta(S_j, X)$$

where $\lambda(S_i, X)$ is the output for the present state S_i and input X, and $\delta(S_i, X)$ is the corresponding next state.

α/β
A
x/y
B
Si
00/1
Sk
Sj
00/1
Sk1
Sk2
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
Switching Circuits & Logic Design

In a state transition diagram we have states, there are transitions and transitions are labeled by some inputs and they also specify some outputs. There may be some other state which is possibly going to the same state; they may be having some other inputs and some other outputs.

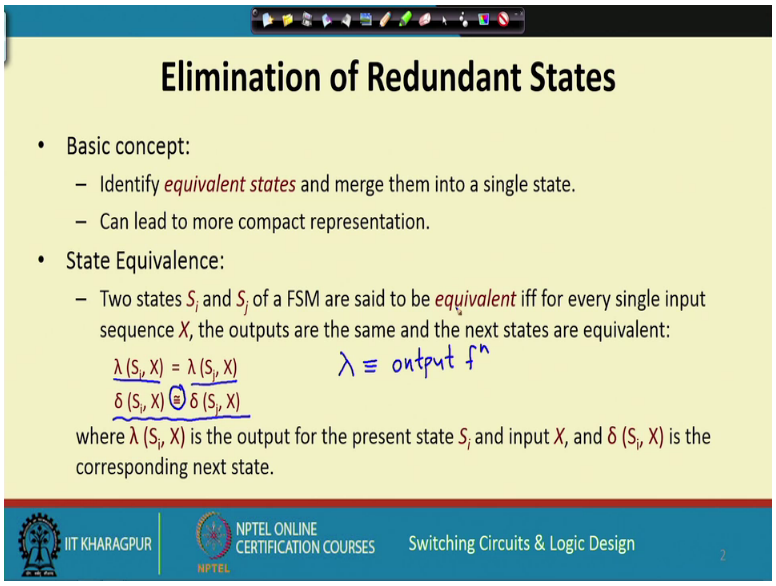
Now, we are trying to answer the question these 2 states A and B; now whether they are equivalent or not right? So, let us try to just answer this question in a slightly formal way; we talk about something called state equivalence. Here we are saying that 2 states S i and S j; they are said to be equivalent if and only if for every possible input sequence for example, this alpha and x; these are some input sequences what, I am saying is that for every possible input sequence.

Let us call it X the outputs will be the same and the next states are equivalent. So, what we are saying is that suppose I have a state S i here, I have another state S j here I am fine let us say there is a single transition only let us assume. So, here let us say for the input combination 0 0; the output is coming as 1 here also for the input combination 0 0; the output is coming as 1.

So, this is what is mentioned here for every input sequence X this is 0 0 is our X; the output is the same and let us say the next state is also some S k. So, if the next state is also the same then we say they are equivalent, but this definition says that the next state may also be equivalent which means say instead of S k. Let us say the first one is going into a state S k 1 and this S j is going to a state S K 2 and we have earlier found out that S k 1 and S k 2 are equivalent. So, even if they are going to 2 sates which are known to be equivalent then also we can say that S i and S j are equivalent right.

Now, mathematically speaking if lambda denote the output function. So, let us saying that the output when you are in state S i and input is X this output must be the same as the output when you are in state S j and the input is X.

(Refer Slide Time: 07:01)



Elimination of Redundant States

- Basic concept:
  - Identify *equivalent states* and merge them into a single state.
  - Can lead to more compact representation.
- State Equivalence:
  - Two states S_i and S_j of a FSM are said to be *equivalent* iff for every single input sequence X, the outputs are the same and the next states are equivalent:

    $\underline{\lambda(S_i, X)} = \underline{\lambda(S_j, X)}$ $\qquad \lambda \equiv$ output f^n

    $\underline{\delta(S_i, X) \equiv \delta(S_j, X)}$

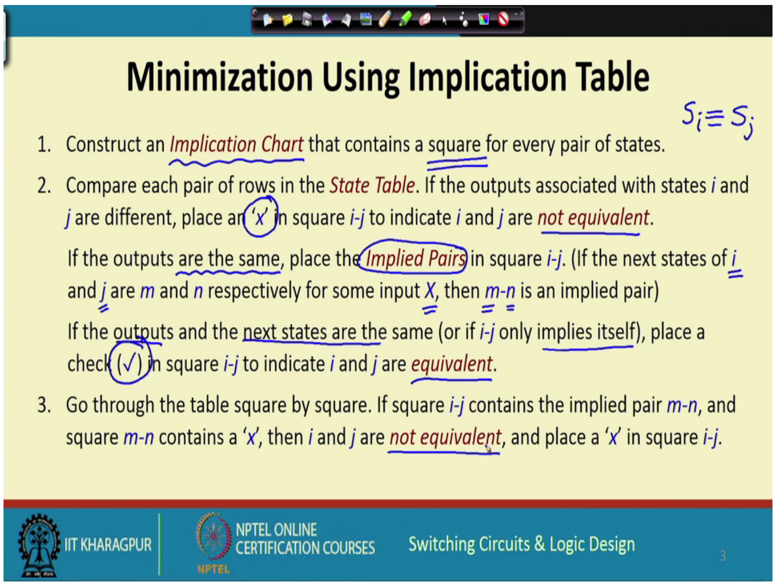
  where $\lambda(S_i, X)$ is the output for the present state S_i and input X, and $\delta(S_i, X)$ is the corresponding next state.

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
Switching Circuits & Logic Design

Here lambda is the output function. So, lambda omega we denote the output function in just using variable symbols. And delta is the next state function which says that the next state when you are in S i; the present state and input X and the next state when you are in S j and input is X that equivalent you say I am not shown it as equal that equivalent; these 2 states should be equivalent.

So, when you have this kind of a condition holding then we say that S i and S j of the machine are equivalent. Just you try to understand that if there are 2 states and if I find that all possible inputs that we apply from the 2 states; we are going to the same equivalent state and the outputs that are coming are also the same what does that mean? It means that we cannot distinguish between these 2 states, they are basically the same. So, we can merge S i and S j into a single state and you can make our FSM smaller this is the basic idea.

(Refer Slide Time: 08:29)



Minimization Using Implication Table

$S_i \equiv S_j$

1. Construct an *Implication Chart* that contains a square for every pair of states.
2. Compare each pair of rows in the *State Table*. If the outputs associated with states *i* and *j* are different, place an '*x*' in square *i-j* to indicate *i* and *j* are *not equivalent*.

   If the outputs are the same, place the *Implied Pairs* in square *i-j*. (If the next states of *i* and *j* are *m* and *n* respectively for some input *X*, then *m-n* is an implied pair)

   If the outputs and the next states are the same (or if *i-j* only implies itself), place a check (√) in square *i-j* to indicate *i* and *j* are *equivalent*.
3. Go through the table square by square. If square *i-j* contains the implied pair *m-n*, and square *m-n* contains a '*x*', then *i* and *j* are *not equivalent*, and place a '*x*' in square *i-j*.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | Switching Circuits & Logic Design

So, let us now show these steps of the minimization then we shall be illustrating with the help of an example. The first thing to notice that we use something called an implication table or an implication chart. Well implication chart is basically some kind of a 2 dimensional structure which shows the relationship between every pair of states.

Let say S i and S j, whether they are equivalent or not if they are not equivalent how they are differing? So, every entry of this 2 dimensional structure will capture the information regarding this similarity of every pair of states this table or chart is called an implication table or an implication chart.

Let say how this table looks like; now as I said this chart I should show you how it should looks like. This chart will have a square for every pair of states because we want to compare between every pair of states; let say S i and S j we want to compare. And what we put in this squares that is explained later; so, what we do first step is compare each pair of rows in this state table, pair of rows means actually we are talking about the pair of states.

We are comparing every pair of states S i and S j in the implication chart, but that is reflected from the state table; you are think of this state table 2 rows of this state table indicates two different states present states ok. That is why we are saying we consider two different rows of this state table and with respect to that we mark the entry in the implication chart how?

So, we compare each pair of rows in this state table suppose the rows corresponding to S i and S j. The condition is if the outputs in the state table associated with states i and j; in short I am saying I and J instead of writing S i and S j. If the outputs corresponding to i and j are different which means this states cannot be equivalent in the state table I find that if some input X is coming; the first state is giving an output 0, the second state is giving an output 1. So, they are not equivalent, so if they are not equivalent I put a cross in that entry in the implication chart indicating that the 2 states are not equivalent ok.

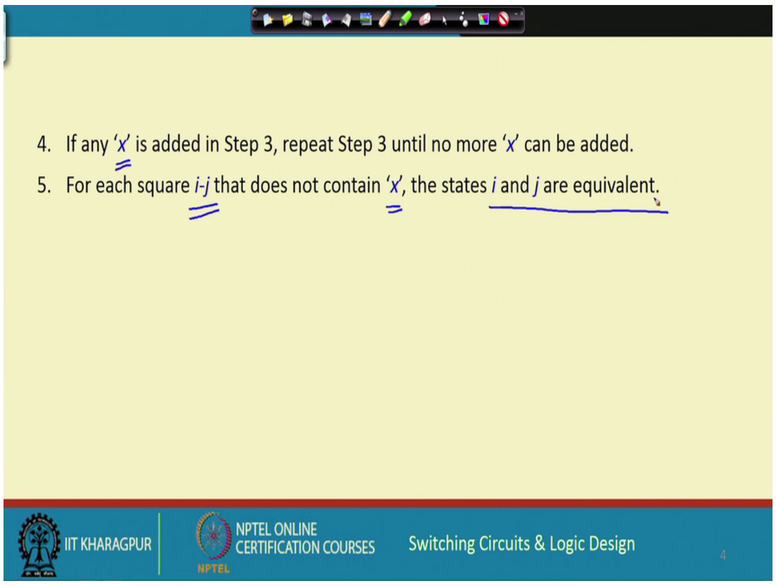
So, if the outputs are equivalent straight away we can conclude that this states are not equivalent and we place across in the corresponding square i-j, that this square corresponding to S i and S j i-j to indicate that they are not equivalent. But if we find that the outputs are the same; then we place something called implied pairs in the square i-j what is the meaning of implied square?

Suppose in this state table we find that corresponding to state i for a input, for particular input X particular input X; the next state is m and for this state j for the same input X the next state is n. So, what we are saying is that all though the outputs are the same, but the next states are different i is going to m, j is going to n. So, in the implication table in that square we note down m and n indicating that they are going to two different states m n, but we still do not know whether m and n are equivalent or not. So, we just note them down that they are going to the states m n; we note like this m dash n. We call these as an implied pair for a pair of states i-j and an input X m n is an implied pair, if the outputs are the same we put the implied pair in the corresponding square.

But if both the outputs and also the next states are the same which means this m and n are identical or this i implies i and j implies j; they imply itself ok, then we can say that definitely these 2 states are equivalent. There we put a tick mark in that square indicating that these 2 states are definitely equivalent. So, you see means we are tentatively marking 3 things first whether they are not equivalent at all we put a cross; they are definitely equivalent we put a tick, but we do not know yet we note down the implied pairs m n because will have to check m n again later to check whether m and n are equivalent or not.

If they are equivalent we will replace them by a tick, if they are not equivalent will replace them by a cross this is how it works ok. So, we repeat these for all this squares go through the table square by square; if a square i-j contains the implied pair m n. And the square m n in the implied table between m and n there is a X which means m and n are known to be not equivalent, then you can place an X in the square; you can mark them as not equivalent then after you have done it once will have to repeat it.

(Refer Slide Time: 14:58)



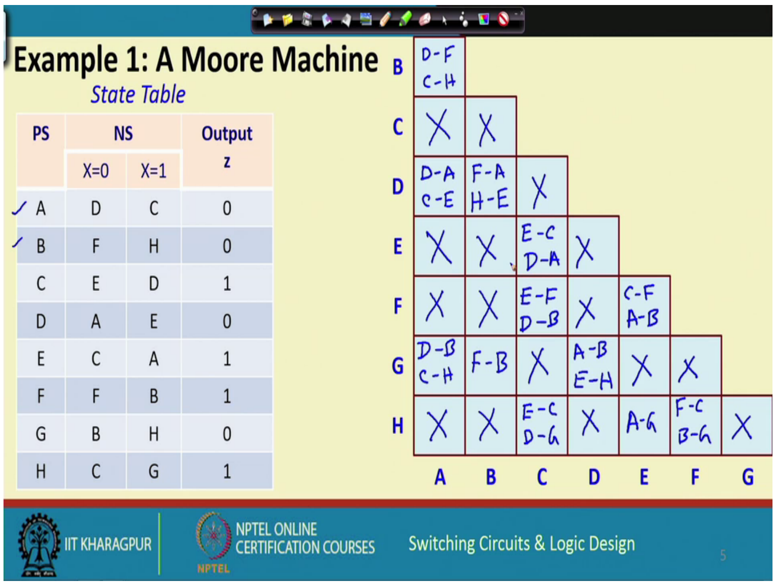
4. If any 'x' is added in Step 3, repeat Step 3 until no more 'x' can be added.
5. For each square *i-j* that does not contain 'x', the states *i* and *j* are equivalent.

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
Switching Circuits & Logic Design

If you find that in the previous step you have added some x; then will have to repeat the process until no more excess can be added.

So, you understand there is a iterative process; this not a onetime process you will have to repeat this several times, make changes in implication table until no more no further changes are possible. And finally, all those squares that does not contain x; you can say that the 2 states are equivalent; this is how the process works. Now let us illustrate this with the help of example it will be easiest.

(Refer Slide Time: 15:43)



Example 1: A Moore Machine

State Table

PS	NS		Output
	X=0	X=1	z
✓ A	D	C	0
✓ B	F	H	0
C	E	D	1
D	A	E	0
E	C	A	1
F	F	B	1
G	B	H	0
H	C	G	1

	A	B	C	D	E	F	G
B	D-F C-H						
C	X	X					
D	D-A C-E	F-A H-E	X				
E	X	X	E-C D-A	X			
F	X	X	E-F D-B	X	C-F A-B		
G	D-B C-H	F-B	X	A-B E-H	X	X	
H	X	X	E-C D-G	X	A-G	F-C B-G	X

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | Switching Circuits & Logic Design

Let us take the example of a Moore machine; this state table as we have shown on the left hand side. You see there are the 8 states A B C D E F G H; there is a single input capital X, there is a single output z ok. This table shows that for all input combination 0 and 1 what will be the next state and because it is a Moore machine the output will depend only on this state and not on the inputs. So, the output I am showing only once or else I could have written D; let us say D comma 0, C comma 0, F comma 0, H comma 0, E comma 1, D comma 1 that is also ok, but because it is a Moore machine I have shown the output as a separate column.

Now, you see the implication chart looks like this as I have shown here. So, I am not showing it as a square matrix because I need to keep 1 square cell for every pair. Let say I mean along the columns I label as A B C D E F G and along the rows I will label as B C D E F G H; So, this cell corresponds to A and B, A and C this A D, A E and so on. B C, B D, B because we do not have to check a salute itself a with A or B with B; that is why the number of columns get reduced as we go from left to right.

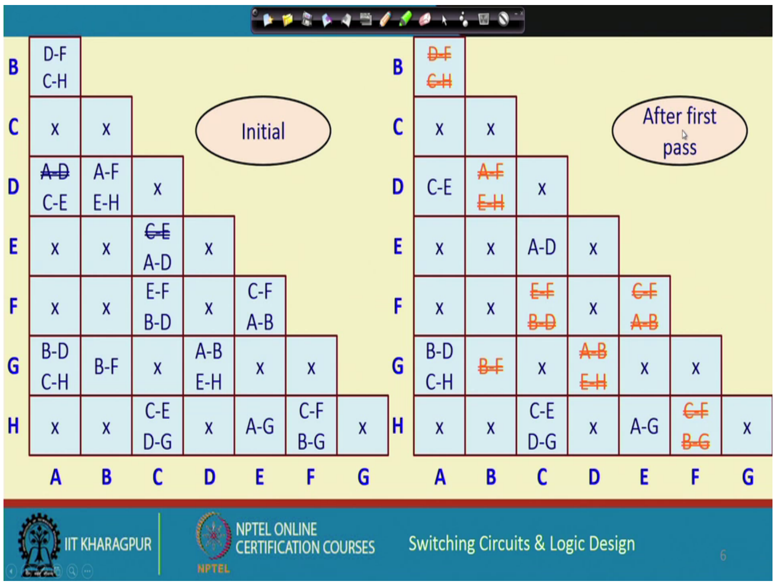
Now, this will be our implication chart; now from the table let us see because it is a Moore machine, let us look at it one by one between A and B. Look at the rows corresponding to A and B, this first 2 rows; the next states the output is a same 0 0 ok. So, the next states are for X equal to 0 D F, for X equal to 1; C H. So, we can write D F, C H between A and C; look similarly between A and C you say outputs are different 0 and 1. So, straight away you can write a cross here; they are not equivalent then A and D; look at A and look at D the outputs are same, but this states are D A, C E; D A, C E these are the implied pairs. A and E, A and E again the outputs are different 0 and 1; so, there will be a cross. A and F, A and F again outputs are different cross A and G output are the same D B, C H D B, C H then A and H again outputs are different 0 and 1; so, a cross.

So, like this you will be filling up all the cells let us say B and C; look at B and C outputs are different cross again, B and D B and D outputs are same. So, F A, H E F A, H E B and E; B and E outputs are different, B and F again outputs are different B and G output is same. So, F B; H while H maps to itself; so we are not writing H only F B the ones that are different those only will write F B; B and H B and H outputs are different.

Similarly, C D; C and D output different C and E; C and E output is the same E C D a E C D A; C and F C and F E F; D B, E F D B. C and G C and G outputs are different C and H; E C, D G; E C D G. Similarly D E output different D F outputs different D G; A B, E H, D and H output is different E and F same output C F; A B, C F A B, E and G different, E and H C is same A G; only A G. And F and G output different and F and H F C; B G F C B G and finally, G and H outputs are different.

So, you see this is the first iteration of this implication chart that we have found out. Now this process will be repeating let us say I am giving example let say the first cell contains D F and C H; we will look at D F. D and F you see D and F are not equivalent; so, you can say that this will also be not equivalent. Similarly D A and C E; D D and A, this we do not know D A is itself in fact, C E; C E is also something is that not X; so, this you cannot delete. So, wherever you see there is an X in the next step will be removing those. So, I am just showing this.

(Refer Slide Time: 22:27)



Initial

B	D-F C-H						
C	x	x					
D	~~A-D~~ C-E	A-F E-H	x				
E	x	x	~~C-E~~ A-D	x			
F	x	x	E-F B-D	x	C-F A-B		
G	B-D C-H	B-F	x	A-B E-H	x	x	
H	x	x	C-E D-G	x	A-G	C-F B-G	x
	A	**B**	**C**	**D**	**E**	**F**	**G**

After first pass

B	~~D-F~~ ~~C-H~~						
C	x	x					
D	C-E	~~A-F~~ ~~E-H~~	x				
E	x	x	A-D	x			
F	x	x	~~E-F~~ ~~B-D~~	x	~~C-F~~ ~~A-B~~		
G	B-D C-H	~~B-F~~	x	~~A-B~~ ~~E-H~~	x	x	
H	x	x	C-E D-G	x	A-G	~~C-F~~ ~~B-G~~	x
	A	**B**	**C**	**D**	**E**	**F**	**G**

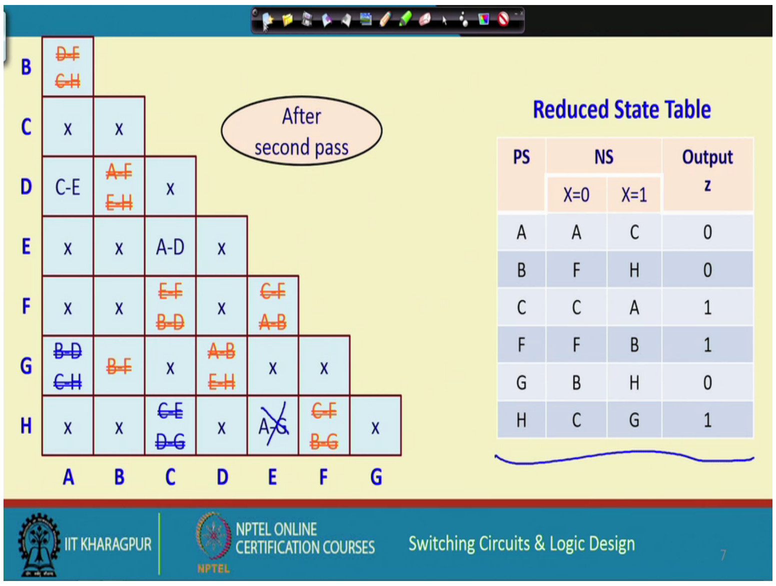
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | Switching Circuits & Logic Design

So, whatever the first step that we have shown here I am showing here and after that wherever there is a mismatch I am crossing them out. For example, this A D is the same; self A D mapped to A D, then C maps to C; I am just cutting them out; C E mapping to C E ok. This is my initial table; now after that I will have to check in the next step: what are the other things that are getting cancelled out.

Like as I said D F; you see D and F there is a cross mark here. So, this will also get cancelled out I am cutting them out meaning there by that I am putting cross here also. Similarly, we look at this cell A F and E H; A F there is a cross ok; so this also gets cancelled out. So, if there is at least one cross they will also get cancelled out.

Then look at this E F this also is E F; E F is E F is not cancelled then B D ah; this one right this one; E F; B D, E and F ok. Then B F B and F is cross this is getting cancelled out A B; E H, E and H is this A G; A G is already cancelled out. So, in this way you systematically cancel out ones which are not compatible pairs. So, this is the process you go on repeating ok. So, you will be repeating this over iterations; this will be after first pass and you continue this pass, wherever there has being some changes made there may be further changes.

(Refer Slide Time: 24:37)



B	~~D-F~~ ~~C-H~~						
C	x	x					
D	C-E	~~A-F~~ ~~E-H~~	x				
E	x	x	A-D	x			
F	x	x	~~E-F~~ ~~B-D~~	x	~~C-F~~ ~~A-B~~		
G	~~B-D~~ ~~C-H~~	~~B-F~~	x	~~A-B~~ ~~E-H~~	x	x	
H	x	x	~~C-E~~ ~~D-G~~	x	$\cancel{\text{A-G}}$	~~C-F~~ ~~B-G~~	x
	A	B	C	D	E	F	G

Reduced State Table

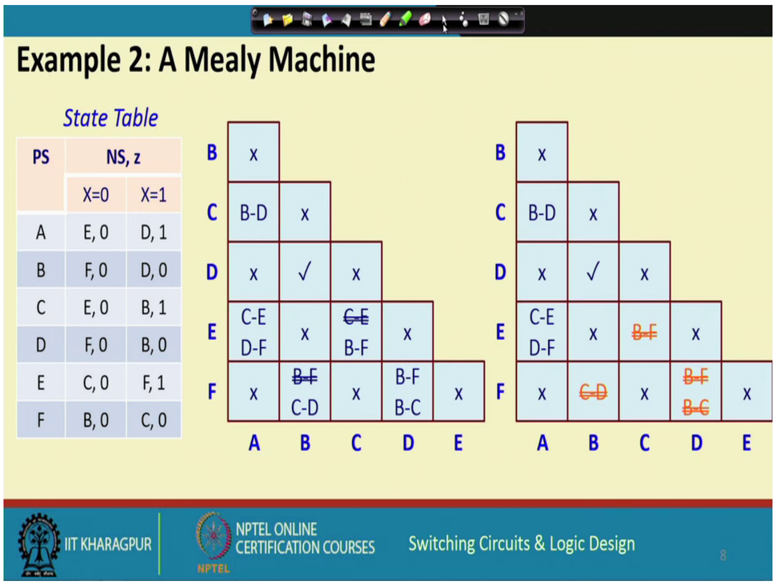
PS	NS		Output
	X=0	X=1	z
A	A	C	0
B	F	H	0
C	C	A	1
F	F	B	1
G	B	H	0
H	C	G	1

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
Switching Circuits & Logic Design
7

Like after another pass something more are getting cancelled out. So, what we see is that after all these cancellations have being made; you will find that there is still some entries which are not cancelled; like here we can find these right. So, here there are 8 states right A B C D E F G H; now you see that some states which you are identifying as equivalent C and E. So, we are merging these states C and E.

Then for example, this A and D; we are merging these states A and D; well this A G also has to be cancelled out this; this is a mistake here, this will also be cancelled out. So, there are there will be 2 equivalent states C E and A D; this you can check once C E and A D. So, C E you merge together, A and D you merge together; so from the initial table you arrive at this final table which you will see from 8 rows we have come down to 6 rows this is the basic idea behind state minimization.

(Refer Slide Time: 26:06)



Example 2: A Mealy Machine

State Table

PS	NS, z	
	X=0	X=1
A	E, 0	D, 1
B	F, 0	D, 0
C	E, 0	B, 1
D	F, 0	B, 0
E	C, 0	F, 1
F	B, 0	C, 0

B	x				
C	B-D	x			
D	x	✓	x		
E	C-E D-F	x	~~C-E~~ B-F	x	
F	x	~~B-F~~ C-D	x	B-F B-C	x
	A	B	C	D	E

B	x				
C	B-D	x			
D	x	✓	x		
E	C-E D-F	x	~~B-F~~	x	
F	x	~~C-D~~	x	~~B-F~~ ~~B-C~~	x
	A	B	C	D	E

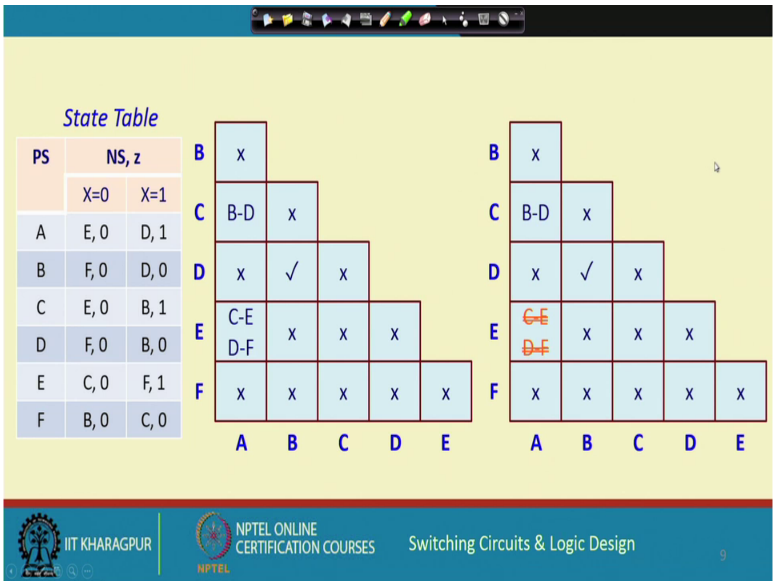
IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES Switching Circuits & Logic Design 8

Let us take another example a smaller example of a mealy machine where there are 6 states and there is one input X. So, the next state and the output are both shown; so, for this the implication chart is shown in a similar way. So, all this state pairs are shown; you see A and B they are not compatible because for this output 1 0, outputs are different; so, x. Similarly A and D are not compatible 1 0 then A and F are not compatible 1 and 0 similarly B and C are not compatible was 0 and 1. You see B D I have put at tick why because in B and D you see the next state is F output is 0, the next state is B and D which is 0; B D are the self the same pair.

So, I can definitely say they are equivalent I put a tick there. So, similarly the other things I have put and whichever are this self like B F; B F is a same pair I am cutting them out C E I am cutting them out. So, like this is the first step of the implication table; so, we can continue like this. In the next step will find the sum of these are getting cancelled out for example, B and F; C D, C D for example, you see between C and D there is a cross between C and D there is a cross.

So, C D gets cancelled out now this B F which was there between B and F you see between B and F there is C D which is already cancelled out. So, this will also get cancelled out and again B F has being cancelled out, B C has also being B C you see B C is already cancelled. So, this is also cancel; so, there will be 3 additional crosses that will be added. So, some B D C D F they remain; so, this is how you continue.

(Refer Slide Time: 28:31)



State Table

PS	NS, z	
	X=0	X=1
A	E, 0	D, 1
B	F, 0	D, 0
C	E, 0	B, 1
D	F, 0	B, 0
E	C, 0	F, 1
F	B, 0	C, 0

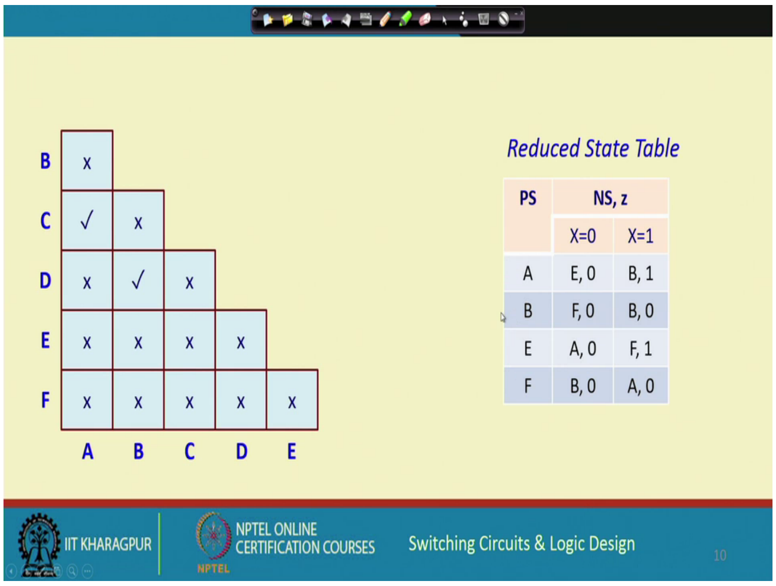
B	x				
C	B-D	x			
D	x	√	x		
E	C-E D-F	x	x	x	
F	x	x	x	x	x
	A	B	C	D	E

B	x				
C	B-D	x			
D	x	√	x		
E	~~C-E~~ ~~D-F~~	x	x	x	
F	x	x	x	x	x
	A	B	C	D	E

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | Switching Circuits & Logic Design

So, this is the final state table which shows that B D is equivalent, C E and D F, but if you check once more you see between C and E there is a cross. So, C E also goes out between D and F there is also a cross; so D F also goes out. So, only B D remains and no further changes can be made. So, in this table you can conclude that only the pair of states B and D are equivalent right; no other states are equivalent here.

(Refer Slide Time: 29:09)



B	x				
C	√	x			
D	x	√	x		
E	x	x	x	x	
F	x	x	x	x	x
	A	B	C	D	E

Reduced State Table

PS	NS, z	
	X=0	X=1
A	E, 0	B, 1
B	F, 0	B, 0
E	A, 0	F, 1
F	B, 0	A, 0

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | Switching Circuits & Logic Design

So, this is the final implication chart and B and D; you merge together and this is your means earlier of course, this A and C was also there A C and B D; this 2 are there. So, the

A C you merge together and B and D you merge together; so, you get the final reduced table. So, like this you can reduce the size of the table. So, in this lecture what we have actually discussed is given a state table; how we can reduce the size of the state table which is called state minimization.

Now, the process of state minimization is very important before you actually do sate assignment and then synthesize in terms of some flip flops; that you have seen earlier, but state minimization is an essential step which can drastically reduce the size of the FSM this specification. So, we will be continuing with our discussion in the next lecture.

Thank you.