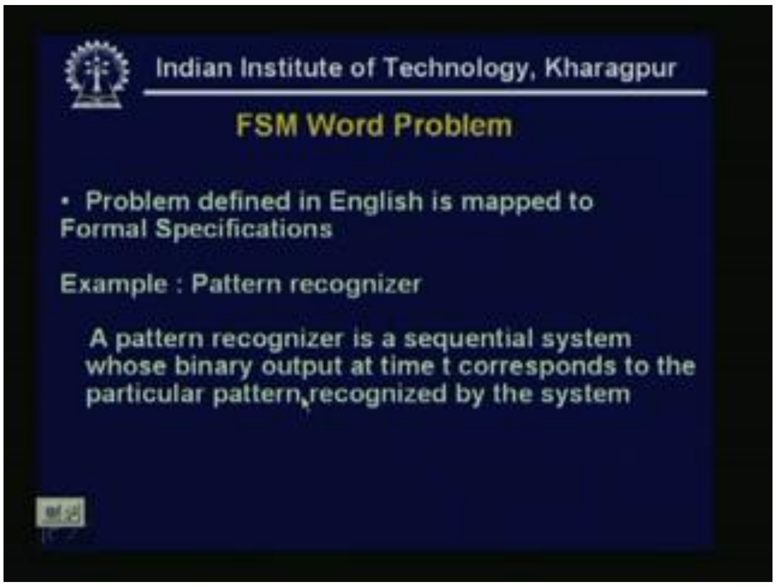
Digital Systems Design

Prof. D. Roychoudhury
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 25
Finite State Machine Design and Optimization

We are discussing on finite state machine design, actually the last day. We are in the half way of, explaining one example, the word problem, the pattern recognizer. So, today we learn the State Machine Design and Optimization.

(Refer Slide Time 01:06)



Indian Institute of Technology, Kharagpur

FSM Word Problem

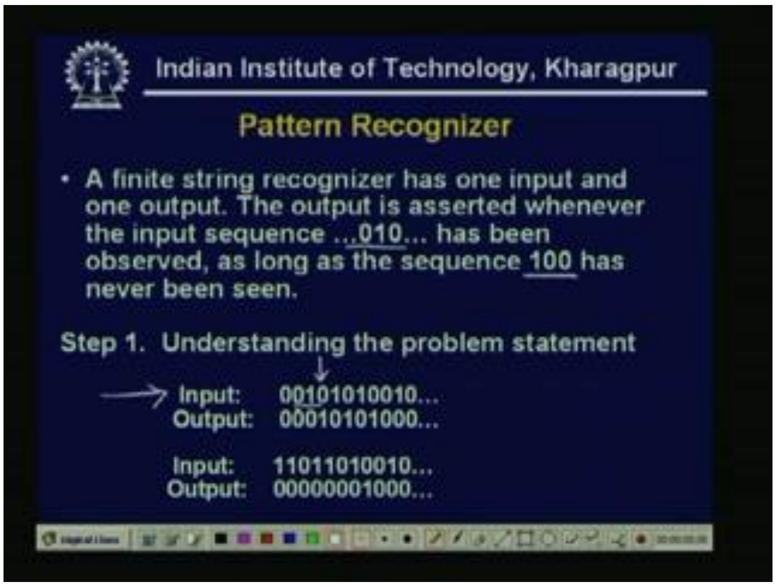
- Problem defined in English is mapped to Formal Specifications

Example : Pattern recognizer

A pattern recognizer is a sequential system whose binary output at time t corresponds to the particular pattern recognized by the system

Now, the last day the problem we are discussing, that was, how the FSM can be designed, for a word problem. See the problem is, a problem defined in some high level language, say English language, and first we map it to formal specification. The example we took, is a pattern recognizer, we define that pattern recognizer is sequential system, whose binary output at time t, corresponds to the particular pattern, recognized by the system.

(Refer Slide Time: 01:46)



Indian Institute of Technology, Kharagpur

Pattern Recognizer

- A finite string recognizer has one input and one output. The output is asserted whenever the input sequence ...010... has been observed, as long as the sequence 100 has never been seen.

Step 1. Understanding the problem statement

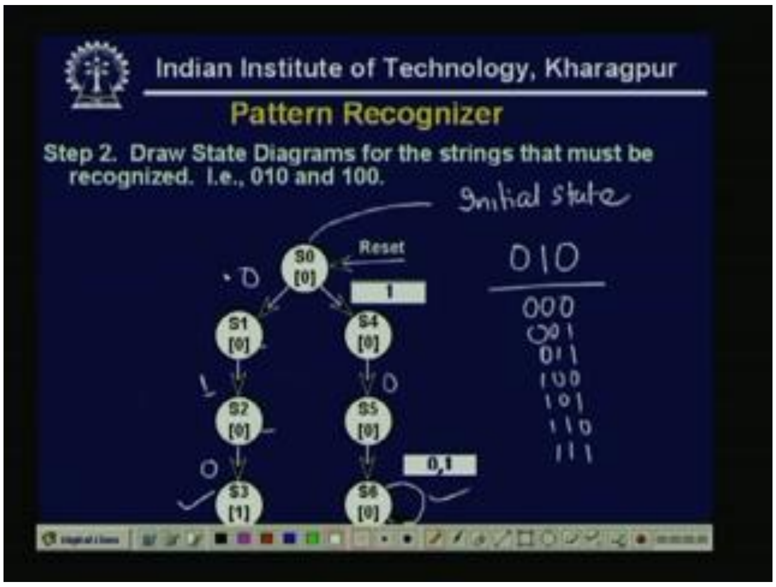
```
Input:   00101010010...
Output:  00010101000...

Input:   11011010010...
Output:  00000001000...
```

Now, we take one example, say if finite string recognizer has 1 input and 1 output, the output is asserted, that means output will be 1, whenever the input sequence, says 0 1 0, as, been identified, so my pattern recognized is 0 1 0. And, another condition we have imposed, that as long as the sequence 1 0 0, has never been seen means, if 1 string, 1 0 0 is reached, then this condition is no more valid. So, if we take one example, see this is my input string, so 0 0 if 0 comes, then output is 0, input 0, output 0, 1 that means, string is 0 0 1, output 0.

Now, when 1 0 arrives, that means, 1 0 1 0 string has identified, so the output is 1. Now again, input is 1, so 1 0 1 output is 0. 1 0 has reached again 0 1 0 is identified. So output is 1, so in this way, again 0 1 0 output is 1, in this way it will proceed. So, we understand the problems statement.

(Refer Slide Time: 03:30)


Indian Institute of Technology, Kharagpur

Pattern Recognizer

Step 2. Draw State Diagrams for the strings that must be recognized. I.e., 010 and 100.

Now, we first, draw the state diagram, so first this is the initial state, as last I mention the design process, then we need one starting state or initial state, it is initial state. Then, if if a 0 has reached, then it will be a new state, if a, ones has come that means, always, because it is a 1 input missing. So either 0 or 1 can come. If it 0 comes it is this 1, if a 1 comes then it is S 4. Similarly, that again it is a 0 comes, then it is a another line 0 0, and see this is the output, means when, one 0 comes output will be 0, another 0 comes output will be 0.

Now, if a 0 1 0, so 0 say this output is 1, means these, this should be a 0 input, 1 input, then 0 input, so if a 0 1 0 is identified, then only the output is 1. But, if it is a 1, and say 1 0, we know that for any other string, other than, other than 0 1 0, any other string 0 0 0, 0 0 1, 0 0 1 any other string, 1 0 0, 1 0 1, 1 1 0, 1 1 1, for remaining 7, it will be a 0 output. So, mainly this is the 0 output state, so this is the first stage or the first step of the state diagram.

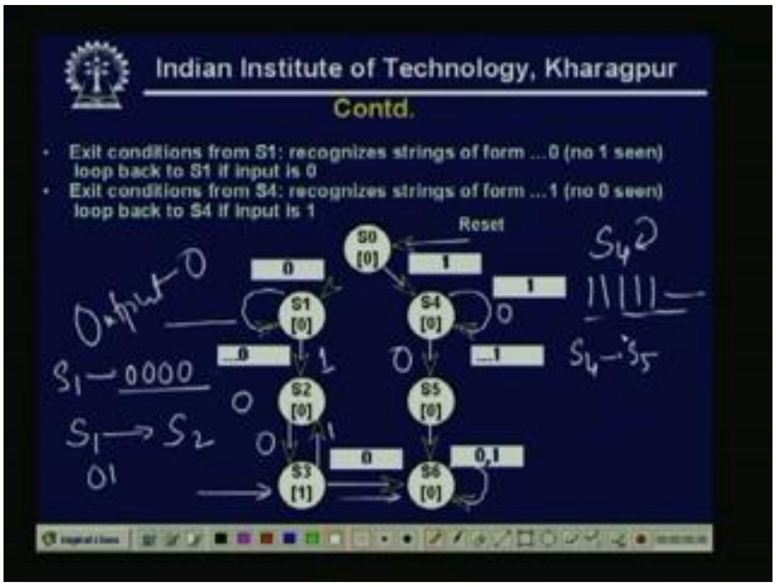
Now, another condition was there, that if a 1 0 0 is recognized then, this condition is no more valid, so that is why, that 1 0 0 as, been seen. So, what will be the next step, now draw state diagrams for the strings, that must be recognized, that is 0 1 0 and 1 0 0. So already we have seen, that 0, that means, this is my 0, this is my 0 1 0, and this is a 1 0 0 state, this can be 0 or 1. So, it will be, in the some gravier state, it will be there. Now, exit condition from state S 3, have recognized 0 1 0, if next input is 0.

So, already we have seen, that if it is a 0 1 0, then it will be a, output, this is the output 1, output 1, output will be 1. Now, if another 0 has come, say 0 1 0 and 1 0, so 0 1 0, now if it a 0, if it is a, say first we take 1 say 1 0 1 0 has come, then it is output will be, output will be 1, this is the output. Now say, if 1 again 1 has come, then there are chances, that the next can be 0, that means, here 1 0 1 0 string is identified, here can be 1 0 1 0 string can be identified, so here it can be 1.

So, if it is a 1, then it again returns to S 2 state, because this 0 state, so actually this 0 state is, the this 1, this 0 1, so, 0 1 0 it will be looping again, because here, it can be, like that 0 1 0, again 1 0, so that S 2 and S 3, it will be S 2 and S 3, this is the situation. Now, if a, if a 0 has reached, because if 1 is, 1 comes, then every time it will be, giving a one output, and 0 1 0 1 these loops, will be in between S 2 and S 3. Now, if a 0 has come, then it should be, it should be 0 output, because if my, input string is 0 1 0.

Now, if 1 0 has come, then here it is 1 output, but here, here it should be 0 output. So, then it, it goes to S 6 state, because S 6 state we are seeing that, this is a 0 output, so it goes to 0 output.

(Refer Slide Time: 09:49)



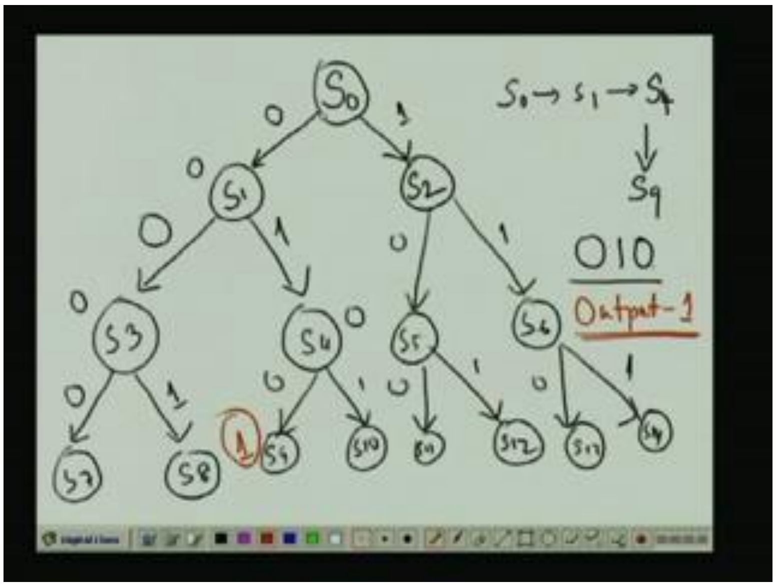
Indian Institute of Technology, Kharagpur

Contd.

- Exit conditions from S1: recognizes strings of form ...0 (no 1 seen) loop back to S1 if input is 0
- Exit conditions from S4: recognizes strings of form ...1 (no 0 seen) loop back to S4 if input is 1

Now, we see the exit condition from S 1. It recognizes strings of form 0, and no 1 seen, means, see this the S 0 was my initial state, S 0 is my initial state. So, initially, that some, one 0 can come, or one 1 can come.

(Refer Slide Time: 10:23)



So, actually, if we see, that see, this is my, this is my S 0 state, now, if 1 0 input as reached, then it will be going to S 1, and if a 1 as come, see this is S 2. Now, similarly here also, some 0 can come, here also, can 1 can come, then if it is 0, every time it is giving, it is giving 0 output, 0. Then if it is a S 3, then because 0 0 string is identified, again 0 output, here see 0 1, see this is S 4, again it is a 0 output. Now, here it is, say again, it should be, it should be again 2 children S 5 and S 6, again it can be 0, it can be 1.

So, this can be S 7, S 8 again 0 1, say in this way, it will S 11, S 12, S 13, say S 14, so 0 then 1, then 1 0 input as come, then 1, then 0 input as come, then 1. Now see, that here the input string is, if a, consider S 0, S 1, S 3, S 7, or S 0, S 1 this transition. If we consider S 0, S 1, S 3 S 7 then actually, 0 0 0 input string is identified, 0 0 pattern is identified. So, the output will be 0, now if S 0, S 1, S 3 S 8, so this is S 0, S 1, S 3, S 8, if, if, this is a S 8, this is a S 8, then 0, S 0, S 1, S 3, S 8, so 0 0 1. So, the pattern identified is 0 0 1, output will be 0.

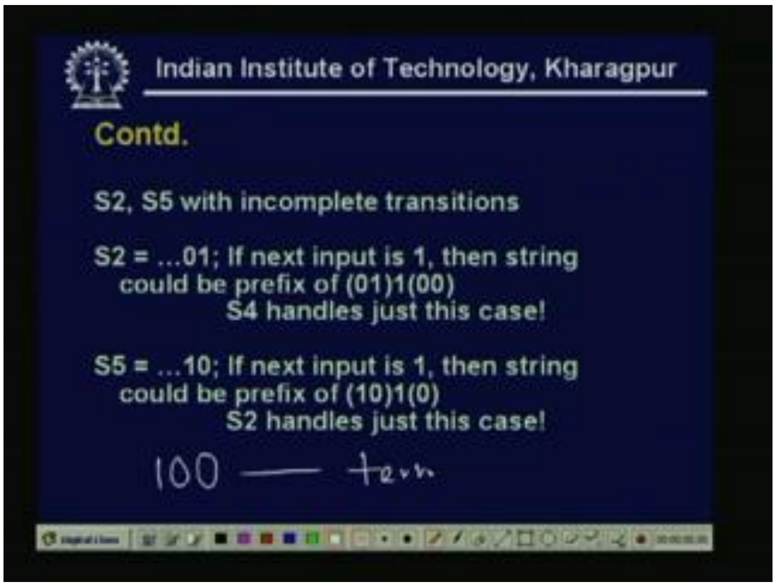
Now, if it is, similarly if it is S 0, S 1, S 4, S 9, now see 1 0 has come, then 1, 1 then again 1 0. So, if it is, S 0, S 1, S 4 S 9, if it is a S 0, S 1, S 4, S 9 this transition, then 1 0 1 0, that string will be identified. So, the string identified is 0 1 0, and our problem, tells that if, this pattern is recognized, 0 1 0 is reached in the input line, then output will be 0. So, then my output, output will be 1, if 0 1 0 is identified, then output will be 1, remaining cases output will be 0, so here 1, 1 output, this is 0 1 0, this is my 1 output. See now, similarly, if we see the other branches, then S 0, S 1, S 4, S 10 means, 0 1 1, then, if in this branch, it should be 1 0 0, output will be 0, 1 0 1 output will be 0, 1 1 0 output will be 0, 1 1 1 output will be 0. So, if we consider this, if we consider only, the current 3 inputs, then only this is the branch, this the, branch, where I can get a 1, this is the branch, where I can get a 1 output. And the remaining branches will give the 0 output, so the same thing we have identified here, that see, the S 0 is initial state, if 1 0 as come, then it is S 1.

Now, again 1 0 as come, 0 may come, may reach, then again it will be in the same state, because that if it 0 comes, then output will be, this output will be 0, output 0. So, S 1 recognizes, S 1 recognizes 0 0 0 this type of inputs, input string, now only one shift will be there, when 1 comes. So, now it will be, a transition will occur from S 1 to S 2. So, this is a, but the string identified is 0 1. So, again output is 0, so here output is 0, now again, if the 0 comes, then only the output is 1.

Now, if a 1 comes, then again, it will go back to returns back to S 2 state, if it is a 0, then it will go to, if the transition from S 3 to S 6. If we now, see the other branches, that means, when the transition occurs from S 0 to S 4, if 1, 1 input arrives, then, this 1 comes output will be 0. Now, similarly just like the S 1, S 4 identifies, that all 1 string, that means, if a 1 comes, then it, it will recognize a 1 string, so S 4, will S 4 state is the all 1 string, similarly, that is 1 recognizes all 0 string.

Now, if it is a, if a 0 comes, so output will be 0, now if a 0 comes, then 1 state transition occurs from S 4 to S 5. Now see, there can be another 1, and we know that, if it is a 1, then, then also 1 0 1, that a, then it is a 1 0 1, output will be 0, and if it is a 0 also, then also output will be 0. So, if it is 0 or 1, that it will be in the S 6 state, and the output is 0, the output is 0.

(Refer Slide Time: 18:55)


Indian Institute of Technology, Kharagpur

Contd.

S2, S5 with incomplete transitions

S2 = ...01; If next input is 1, then string could be prefix of (01)1(00)
S4 handles just this case!

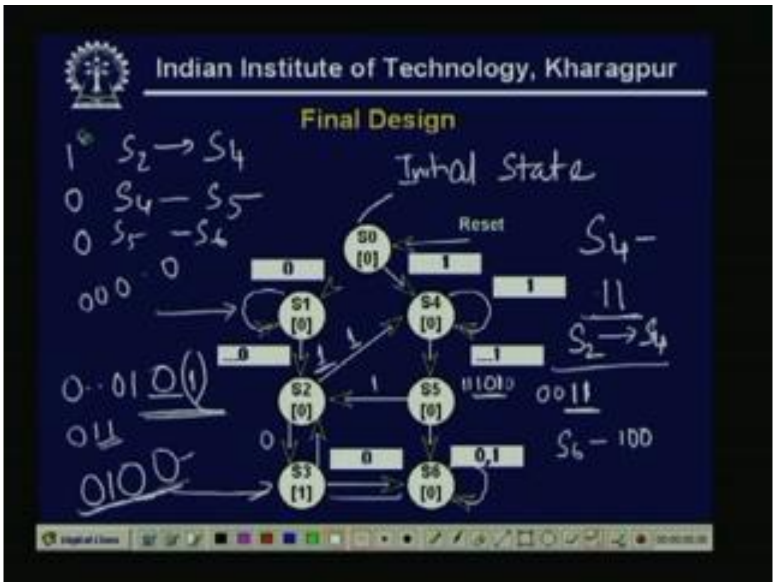
S5 = ...10; If next input is 1, then string could be prefix of (10)1(0)
S2 handles just this case!

100 —— term

So, S 2, S 5 with incomplete transitions, why, that S 2 is 0 1 state, we have seen that. Again if we, see the previous state ((Refer Time: 19:12)), that actually, 0 1 again 0. That means here, if it is a input string 0, then if 1 comes then, it will come go to S 2, then 0, so that 0 1 0, now again 1. So, S 2, S 3, this transition is actually 0 1, it identifies the string 0 1, and see, if it is S 5, if it is S 5, then actually 1 0. So, that is written, that S 2 0 1, if next input is 1, then string could be, prefix 0 1 1, so output will be 0 0, and S 4 handles, just this case.

And, S 5 is the reverse 1, because it is a 1 branch, and then if 0 comes, then only the state transition occurs, then if it is 1 0, then it can be 1 0 1 or 1 0 0. So, if it is a 1 0 0, we know that is our, if it is a 1 0 0, that is our terminating condition, terminating means that, the pattern recognition case, that will no more valid, so this is the termination condition. So, 1 0 0 can be identified.

(Refer Slide Time: 20:53)



So, if we consider now the final design, see, this our initial state, S 0 0, S 0. So, we start from there, see now if a 0 comes, it will go to the left branch S 1. Now, already we have seen, that if, now after reaching into S 1, the output will be 0. And, number of strings that contain 0 that means, for every 0, it will be in the same state S 1, and output will be 0, so S 1 recognizes all 0 strings. Now, if a 1 comes, so now, if a 1 comes, then only, it will be a transition S 1 to 2.

If a 0 comes, then it is a, now if a 1 come, then what type of, so because we have already seen that every state, there can be two input either 0 or 1. So, now the present input is, present state is S 2, and that is, some a strings of 0's and then 1, now it can be, it can be either 0, or it can be 1, so there are two branches. Now, if it is a, if it is a 0, then already we have seen, that is our recognized pattern 0 1 0, but if it is a 1, then I want, that should be a 0 1 1 case.

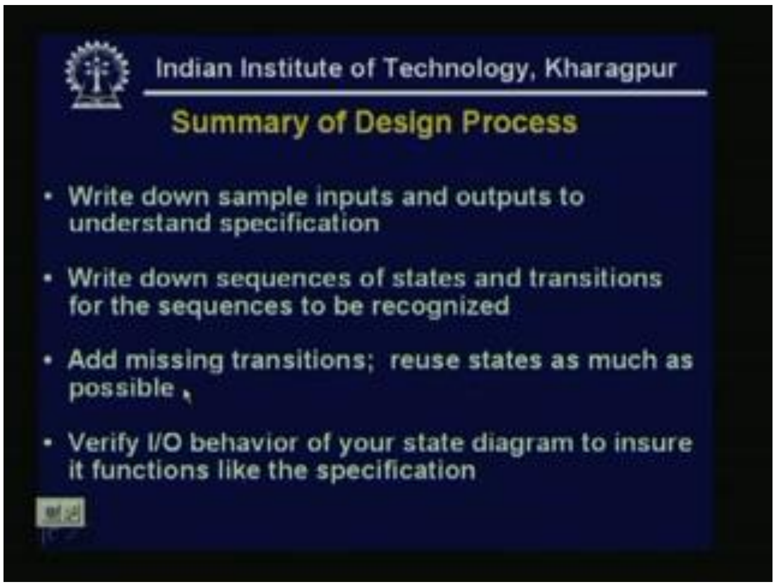
And we know, as it is 1 1, already we have seen, that actually our S 4 recognizes all 1 1 string, so that is why, we redirect, when 1 1 comes, that state transition will be from S 2 to S 4, because in the S 2 state as, been reached, after getting a 1, after getting a 1 input. So, it was a some 0 0 0 string 1, now if a 1 comes, then it will be actually, 1 1 stream, and that is, already we have identified that, that state is S 4, S 4 recognizes all 1 or more than 1 input string, so that is why, it transitions, or it goes to S 2 to S 4.

Now, this is the S 3 is the condition, so the 0 branches, or if it is now, if it is in S 3, so already we have seen S 3 is the our recognized input. Now, if a 0 comes, so output will be 0, so it transitions from S 3 to S 6, because the string is 0 1 0 0, so it is a 1 0 0, actually it is a 1 0 0 state. So, we see, whether it is a 1 0 0 state or not, so from the initial state again we start, 1 means, it goes to S 2 to S 4, 1 1 input means, S 2 to S 4, then any number of 1, we have seen that S 4, so if a 0 comes, so actually, that is S 4 to S 5.

Now, it is again, if a 0 comes, then it is a S 5 to S 6, so actually our S 6 state, recognizes if 3 input we consider, 3 consecutive input bits, then actually, S 6 is 1 0 0, that is why, from S 3, when the present state is S 3 means, S 3 recognizes 0 1 0. Now, if a 0 comes, so if we consider 3 consecutive input bits, then it will be 1 0 0, which is recognized, by the state S 6, that is why S 3 to S 3 transition as been occurred, now if S 5, is in the state S 5 1 1 as come.

Now, see S 5 means, this is a 1 1, say S 4 means any number of 1, now it is a 0, so if it is 1 comes, then it is a 1 0 1 state, it is a 1 0 1 state. So, it will go to the S 2 state, because it is a, again if we start this is a 0 1 state, so that is why, it a, because again, it can 1 0 can come, so S 5 to S 2 transition occurs. So, these are my final, this is the final design of the pattern recognizer.

(Refer Slide Time: 26:58)


Indian Institute of Technology, Kharagpur

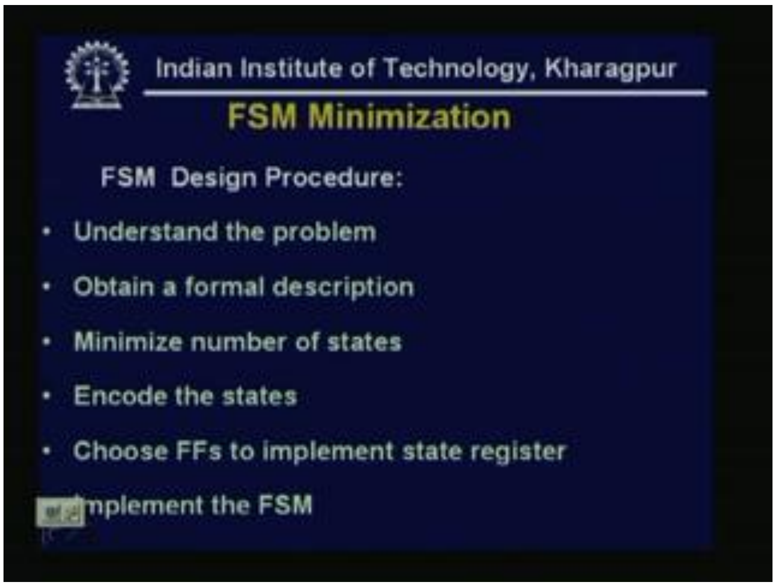
Summary of Design Process

- Write down sample inputs and outputs to understand specification
- Write down sequences of states and transitions for the sequences to be recognized
- Add missing transitions; reuse states as much as possible
- Verify I/O behavior of your state diagram to insure it functions like the specification

So, now if we summarize, the full design process, what we have done, first we write down, the sample inputs outputs, to understand the specification. Therefore, the pattern recognizer, we have written that for, what input patterns, it will give a, output 1, and remaining inputs, it output will be 0. Then we have done the state transitions, of the sequences to be recognized, say for this particular example, it is giving, one output, for only particular input patterns.

Now, there can be, the cases there can be some problem, for which say more than one input bit pattern the output will be 1, say for 0 1 0, as well as for 1 1 1, the machine should give a 1 output. So, we do the state transitions, now add missing transitions. So, that reuse states, as much as possible, and then verify the, IO behavior of your state diagram to insure it functions, is actually are for the specifications.

(Refer Slide Time: 28:23)



Indian Institute of Technology, Kharagpur

FSM Minimization

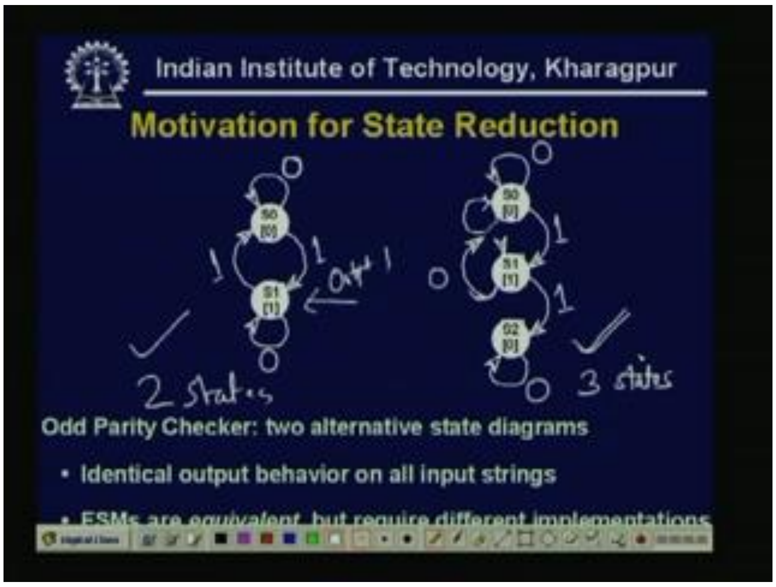
FSM Design Procedure:

- Understand the problem
- Obtain a formal description
- Minimize number of states
- Encode the states
- Choose FFs to implement state register
- Implement the FSM

So, now what we have seen that last example, we have seen that, there are 7 states we have drawn, because it is a 3 input pattern should be recognized 0 1 0, that is why, we have given, the 7 states. Now, if we see the, full efficient design procedure, what we will be doing, first we should understand the problem, then a formal description, then minimize the number of states, because ultimately, our aim is to design and, or is to make a efficient design.

So, we will not keep too many states, because if a number of states are more, then more flip flops will be needed, because already we have seen, that if it is a, N number of states, then log n number of flip flops, will be needed for that. And then, encode the states, this is for that, encoding state, state encoding, then choose flip flops, which flip flop we want to, utilize for particular application, and, then implement the FSM, so now we will see the minimization.

(Refer Slide Time: 29:52)



Indian Institute of Technology, Kharagpur

Motivation for State Reduction

Odd Parity Checker: two alternative state diagrams

- Identical output behavior on all input strings
- FSMs are *equivalent*, but require different implementations

Again, we consider the, example the last day we discussed, the odd parity checker, so that means, one input pattern, if the number of 1's are odd, then it will give a 1 output, and otherwise it will be 0. So, there are, what we have seen there are two, alternative state diagram. See that, if, a initial state is S 0, then if 0, input as come any number of 0, means, it will be a even parity. Because no 1 is there, now if 1 comes, then actually, it will be a odd parity, giving output 1.

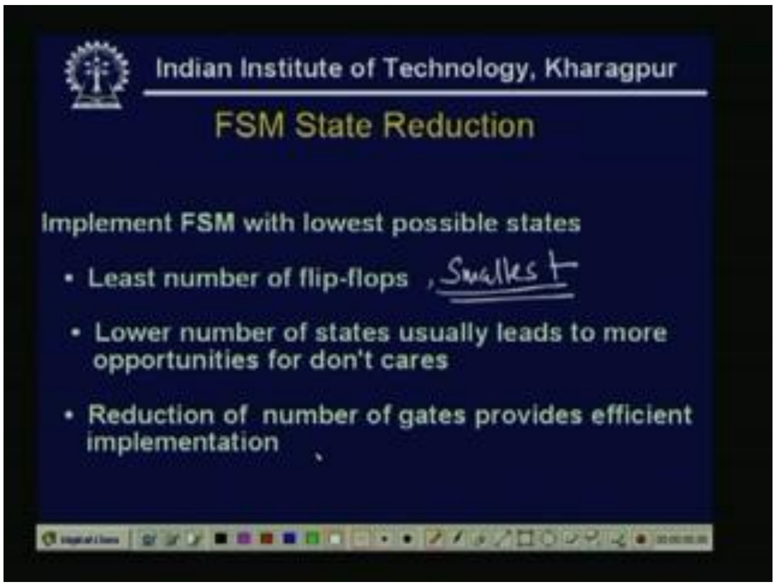
Then again, if it is 0 comes, any number of 0, actually it will be a odd parity, it so it remains in the same state, if it is 1 comes, then 1 plus 1, again it will be even parity. Now, if we see, that if we use 3 states, then again S 0 is the initial state, any number of 0 it will be it will remain in S 0. Now if a 1 comes, then it will be in S 1, now again, if a 1 come, it will be should, it should be in S 2, output should be 0, because even parity. Now, again any number of 0 will create that even parity, so it will be in the S 2, giving 0, output.

And, so if it is in the 0, now if it is in 0 comes, then it will, it will go back to the 0 state, now see, that identical output behavior on all input strings. So, here, there are 2 states, here the same problem, we have specified using 3 states. So, the transition diagram, we have made using 2 states, using 2 states, and using 3 states, and they are behaving likely.

So, which one will be choosing, see now the implementation will be different, because here it is a, 2 states are there, so we need only 1 flip flop 0 and 0 and 1 state.

Here, it is a 3 flip flops, at least we need 2 flip flop, 2 3 states we need 2 flip flop, that, if we do the state encoding, that then, 0 0, 0 1, 1 0, or 1 1 any1, of this, we will be taking. See, so the implementation of this 2 state transitions, will be different, but the, they are solving, both are solving, the same problem, giving the same output. Now, which one you will choose, so this is the main problem of the state reduction.

(Refer Slide Time: 33:36)



Indian Institute of Technology, Kharagpur

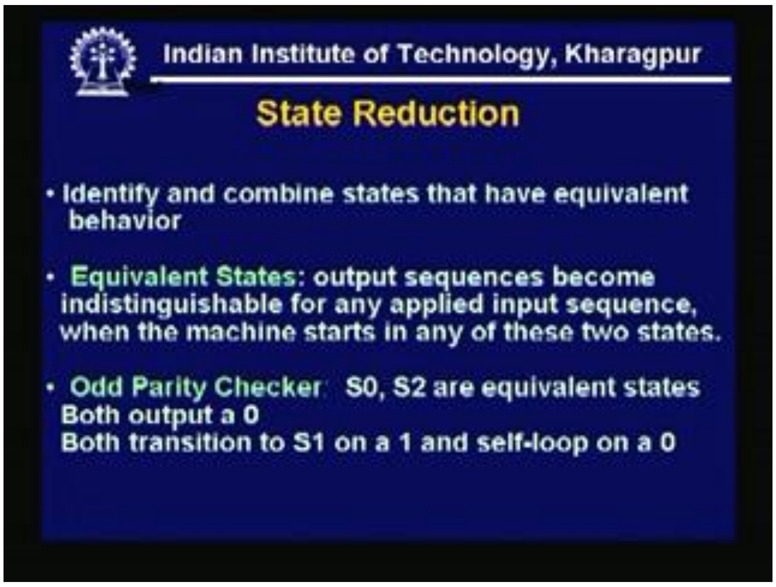
FSM State Reduction

Implement FSM with lowest possible states

- Least number of flip-flops , Smallest
- Lower number of states usually leads to more opportunities for don't cares
- Reduction of number of gates provides efficient implementation

So, main motivation is the implement FSM with lowest possible states, so that, we get, the lowest number or least number of flip flops. So, as much as possible, or smallest number of smallest number of possible, now lower number if states, usually leads to, more opportunities for do not cares. When we will be doing the design, we will take care, this thing, and reduction of number of gates, provides efficient implementation. So, not only the state, but also, if we can reduce the number of gates, needed to implement the design, then it will also provide a efficient design. So, these are the main motivation, for the FSM state reduction.

(Refer Slide Time: 34:42)



Indian Institute of Technology, Kharagpur

State Reduction

- Identify and combine states that have equivalent behavior
- Equivalent States: output sequences become indistinguishable for any applied input sequence, when the machine starts in any of these two states.
- Odd Parity Checker: S0, S2 are equivalent states
  Both output a 0
  Both transition to S1 on a 1 and self-loop on a 0

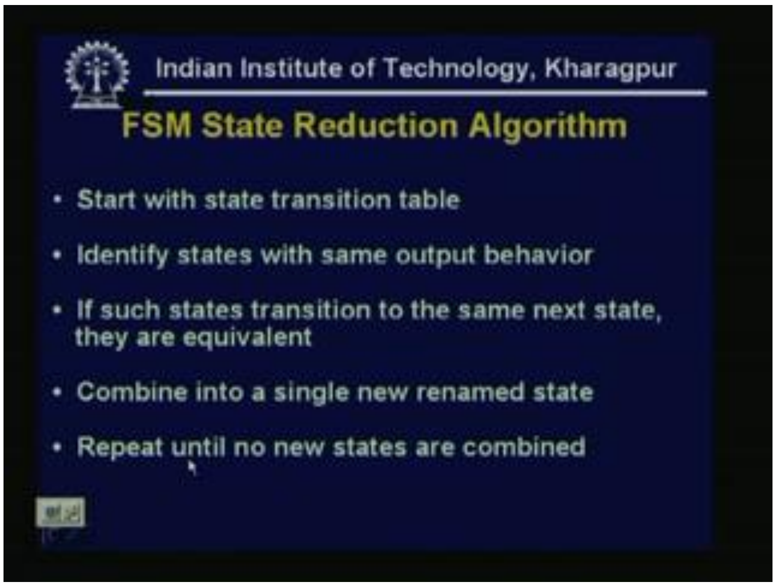
So, identify and combine states, that have equivalent behavior, see first, what we are doing, we are doing, we are considering all possible cases, if all possible input string come, then what will be the output or how the machine behaves, means for all possible inputs, first we define, all possible output. Then, we try to identify and combine states that have equivalent behavior. See in the last example, we have seen that, whenever the S 0 was all 0 strings, then from other present state, if it is a all 0 type of string arrives, immediately the state transition occurs from other state to the S 1 state.

We have seen, that in the, our pattern recognition, pattern a pattern recognizer FSMs. So, now we, that what do you mean by equivalent states, see for, all input combination states, input combinations, state transitions to the same equivalent states. That means, that, if the state transitions, occurs to the same equivalent state or same state, for all input combinations. Then we have, defining this or we are naming these are the, equivalent states. So, say for odd parity checker, S 0, S 2 are equivalent states, because both output S 0, both transition to S 1 on a 1 and self-loop on a 0.

To check, so ((Refer Time: 37:11)) see here, S 0 and S 2, so here, S 0, if 1 0 input as come, that means, means my present state, present state is S 0, present state it is S 0. Now if a 0 comes, if a 0 arrives, then again it will be in the S 0, that means S 0 to S 0. Now, if a 1 comes, ((Refer Time: 37:53)) then it is a S 0 to S 1. ((Refer time: 38:04)) So if it is a 0 comes, then it is say it is S 0, and if it is a 1 comes, it is S 0 to S 1, for S 0 state. If we consider the S 2 state, S 2, say if a 0 comes, then again, it is S 2 to S 2, if a 0 comes.

If a 1 comes, then it is S 2 to S 1, now see, for all possible input combinations means 0 and 1 that S 0 remains in S 0, S 2 remains in S 2. And for 1, for 1, S 0 to S 1, S 2 to S 1. So, that means, for all possible input combinations, the state transitions of S 0 and S 2 are same, so S 0 and S 2 are equivalent states. So, for both output, is 0, both transition is S 1, and self-loop on a 0, so S 0 and S 2 are equivalent state.

(Refer Slide Time: 39:21)



Indian Institute of Technology, Kharagpur

FSM State Reduction Algorithm

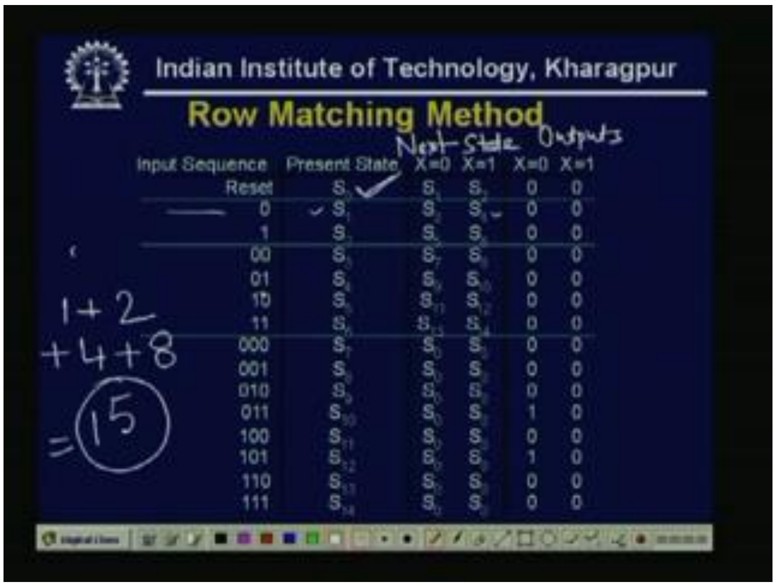
- Start with state transition table
- Identify states with same output behavior
- If such states transition to the same next state, they are equivalent
- Combine into a single new renamed state
- Repeat until no new states are combined

Now, there are several algorithms, exist for the state reduction, so what should be the algorithmic steps. So, if we start with state transition table, fist we already, we have defined for all possible inputs, what should be the output of the machine, then identify states, with same output behavior. So, first what we are doing, that the states we are identifying, which are giving the same output, now if such state transition to the same next state, they are equivalent.

Just now we have seen, that if the state transitions are also same that means, for all input combinations, if the state transitions are also same, giving the same output, then they are equivalent state, so we identify here, thing is, we identify the equivalent states. So, that what we, intuitively what, we understand, that the equivalent states can be merged, they are equivalent states. Now, combine into a single new renamed state, that means, this equivalent state when merged, they are giving a new state.

So repeat, until no new states are combined, now we will, follow this rules, until we get any new, any 2 states can be combined. That means, we cannot, identify any equivalent states no more, then we will stop.

(Refer Slide Time: 41:27)


Indian Institute of Technology, Kharagpur

Row Matching Method

		Next State		Outputs	
Input Sequence	Present State	X=0	X=1	X=0	X=1
Reset	S_0 ✓	S_1	S_2	0	0
0	✓ S_1	S_3	S_4	0	0
1	S_2	S_5	S_6	0	0
00	S_3	S_7	S_8	0	0
01	S_4	S_9	S_{10}	0	0
10	S_5	S_{11}	S_{12}	0	0
11	S_6	S_{13}	S_{14}	0	0
000	S_7	S_0	S_0	0	0
001	S_8	S_0	S_0	0	0
010	S_9	S_0	S_0	0	0
011	S_{10}	S_0	S_0	1	0
100	S_{11}	S_0	S_0	0	0
101	S_{12}	S_0	S_0	1	0
110	S_{13}	S_0	S_0	0	0
111	S_{14}	S_0	S_0	0	0

$$1+2+4+8=\boxed{15}$$

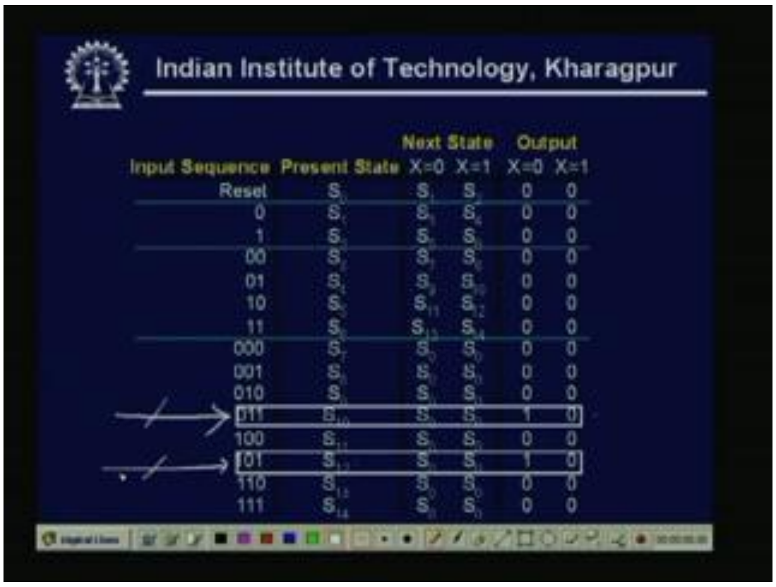
So, one very simple algorithm, just from common sense it appears, that is the row matching method, so if we see that, again that, odd parity checker. See that, this is the input sequences, and present state, and the all possible inputs we get, then first is that input sequence. If reset, then the present state is S 0, as always we are seeing. Now, if it is X is equal to 0, it goes to S 1, if X is equal to 1, it goes to S 2. So, actually these are the inputs, and this are my outputs, outputs, are 0.

Now, if it is, if it is now, input sequence sorry, these are next state, for these inputs, so I should write, these are the, these are the next states, for these inputs. So now, if the present state is S 1, and input sequence 0 comes, then it will goes to S 1 to S 3. Similarly, if a, 1 comes then it will be S 4. Now, if it is S 2, X equal to 0, it will go to X 5, then it will be S 2 to S 6, and in each of the cases, it will give a 0 output. Now in this way, if we consider the all 3 input patterns, then we will be seeing that, from 0 to, that for 0 state, or only for 1 input, 1 bits input sequence, there are 2 states.

So, if we see, that for reset 1 state, for 2 inputs, 1 input sequence there are 2 states 0 or 1. For if we consider 2 bit input sequence, then 2 square means 4 state. If we consider 3 input bit sequence, then 2 to the power 3, 8. So, it will be total 15 such, states will be there, that is why, it is, that 0 1 0 0, so here it is 1 state, here it is 2 state, here it is 4 state and these are my 8 states.

So, 1 plus 2 plus 4 plus 8, for all these the table are completed. That means, these are this is the table, that for all possible input sequences, what will be the present state, what will be the next state. For this particular input, if this, this particular input arrives, and then what will be the output, so this is my table.

(Refer Slide Time: 46:03)



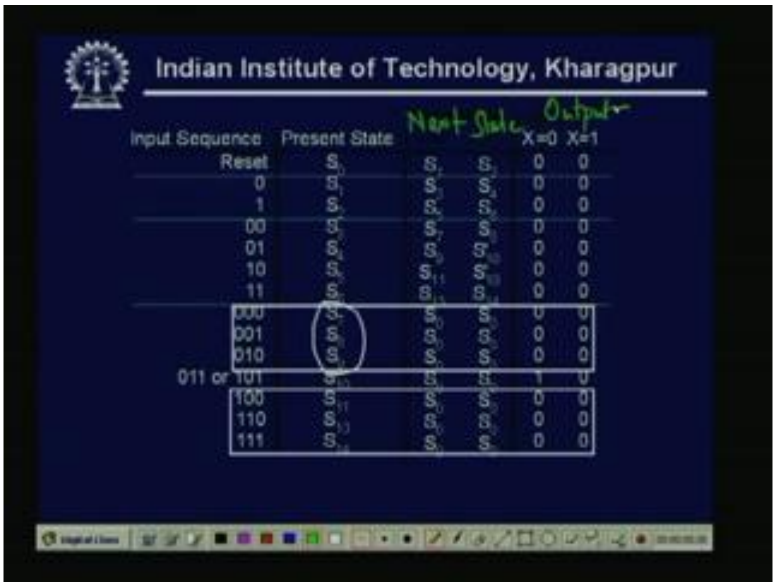
Indian Institute of Technology, Kharagpur

		Next State		Output	
Input Sequence	Present State	X=0	X=1	X=0	X=1
Reset	S_0	S_1	S_2	0	0
0	S_1	S_3	S_4	0	0
1	S_2	S_5	S_6	0	0
00	S_3	S_7	S_8	0	0
01	S_4	S_9	S_{10}	0	0
10	S_5	S_{11}	S_{12}	0	0
11	S_6	S_{13}	S_{14}	0	0
000	S_7	S_0	S_0	0	0
001	S_8	S_0	S_0	0	0
010	S_9	S_0	S_0	0	0
011	S_{10}	S_0	S_0	1	0
100	S_{11}	S_0	S_0	0	0
101	S_{12}	S_0	S_0	1	0
110	S_{13}	S_0	S_0	0	0
111	S_{14}	S_0	S_0	0	0

Now, we try to find out, whether any row matches, that is why it is a row matching algorithm. So, see that if we consider that row, now what we want to match, we want to match the present state or what are the different present states, which gives a same output, and same state transitions. So, here, if we see, you check all the rows, then we will be seeing, that, here it is a, see this is, for this state, that means, when the input sequence is 0 1 0, 0 1 1, present is S 10.

And, for the next state X equal to 0, the next state is S 0 or z equal to 1, and next state is S 0, output is 1 and 0. Similarly, for 1 0 1, when the present state is S 12, then also the next will be S 0, S 0 giving the output 1 0. So, these two rows, are giving the same state transitions. So, these row two rows can be combined, now see these two rows these two rows 0 1 1 or 1 0 1 can be combined, and as the next states are S 0 S 0. So it is, and this is a new present state S 10 dash, so this is S 10 dash, the new rename state. Now, similarly, again if we check, now see that, if the, these are the all possible inputs sequences, 0 0 0, 0 0 1, 0 1 0.

(Refer Slide Time: 48:26)

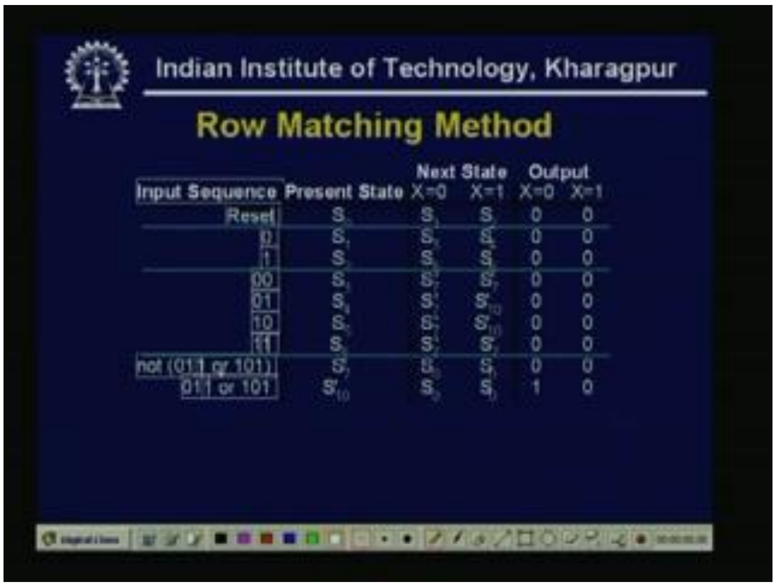


Indian Institute of Technology, Kharagpur

Input Sequence	Present State	Next State		Output X=0	X=1
Reset	S_0	S_1	S_2	0	0
0	S_1	S_3	S_4	0	0
1	S_2	S_5	S_6	0	0
00	S_3	S_7	S_8	0	0
01	S_4	S_9	S'_{10}	0	0
10	S_5	S_{11}	S'_{10}	0	0
11	S_6	S_{13}	S_{14}	0	0
000	S_7	S_0	S_0	0	0
001	S_8	S_0	S_0	0	0
010	S_9	S_0	S_0	0	0
011 or 101	S'_{10}	S_0	S_0	1	0
100	S_{11}	S_0	S_0	0	0
110	S_{13}	S_0	S_0	0	0
111	S_{14}	S_0	S_0	0	0

Then the present state, a group of states, say S 7, S 8, S 9, the next states are, these are my, these are actually the next states, so and these are the output. So, see next states are, S 0 S 0, S 0 S 0, S 0 S 0. Now for this group of present state S 11, S 13, S 14, obviously the, input combinations are different. Then again the same next states, are same 0 0 outputs, have been observed.

(Refer Slide Time: 49:21)



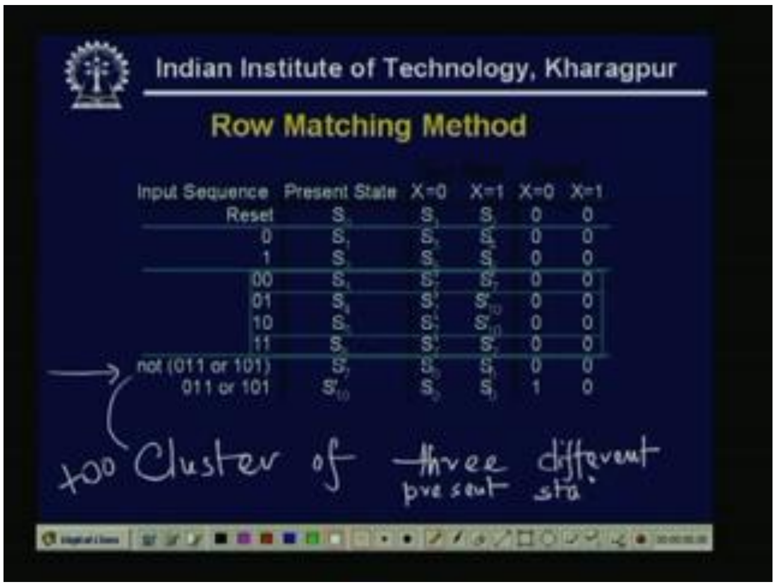
Indian Institute of Technology, Kharagpur

Row Matching Method

		Next State		Output	
Input Sequence	Present State	X=0	X=1	X=0	X=1
Reset	S_0	S_1	S_2	0	0
0	S_1	S_3	S_4	0	0
1	S_2	S_5	S_6	0	0
00	S_3	S_7	S_7	0	0
01	S_4	S_7	S'_{10}	0	0
10	S_5	S_7	S'_{10}	0	0
11	S_6	S_7	S_7	0	0
not (011 or 101)	S_7	S_0	S_0	0	0
011 or 101	S'_{10}	S_0	S_0	1	0

So, if we combine, then we can write in this way, that not 0 1 1 or 1 0 1, because already we have seen, see, we have already seen ((Refer Time: 49:35)), that 0 1, if it is input sequence is 0 1 1 or 1 0 1, then it is S 10 and next state is S 0. For others, that actually this, it is giving the or the output behavior is same, for other cluster of three present states S 7, S 8, S 9 and S 11, S 13, S 14. So, this is the 2 classes identified, this can be combined, this can be combined, this is the 2 classes identified.

(Refer Slide Time: 50:38)



Indian Institute of Technology, Kharagpur

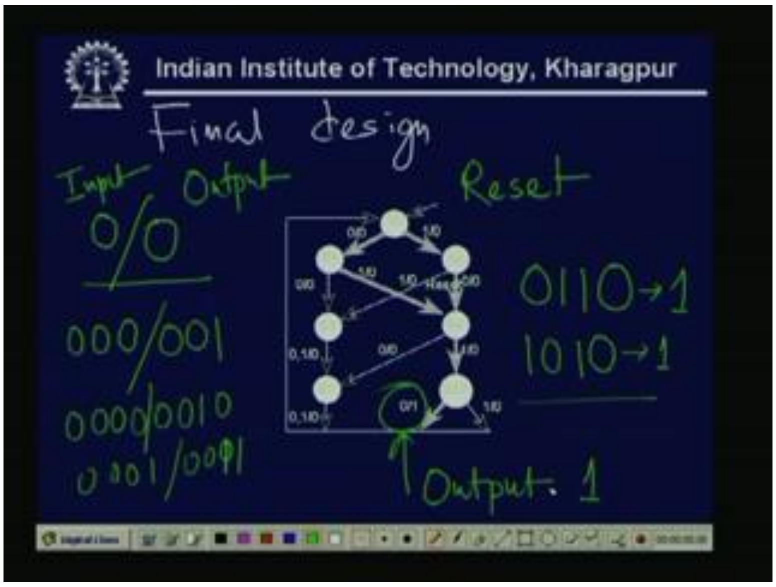
Row Matching Method

Input Sequence	Present State	X=0	X=1	X=0	X=1
Reset	S_0	S_1	S_2	0	0
0	S_1	S_3	S_4	0	0
1	S_2	S_5	S_6	0	0
00	S_3	S'_7	S'_7	0	0
01	S_4	S'_7	S'_{10}	0	0
10	S_5	S'_7	S'_{10}	0	0
11	S_6	S'_7	S'_7	0	0
not (011 or 101)	S'_7	S_0	S_0	0	0
011 or 101	S'_{10}	S_0	S_0	1	0

two Cluster of three different
present sta

So, that is why, it is written that, it is written that, not 0 1 1 or 1 0 1, this is not 0 1 1 or 1 0 1, means that, cluster of three or or better two cluster of three different present states. And, one new rename states S 7 dash, S 7 dash, as been given for the present state, output is S 0, next state S 0 output is 0 0. So, there are two combines states we get, one is S 7 dash, one is S 10 dash.

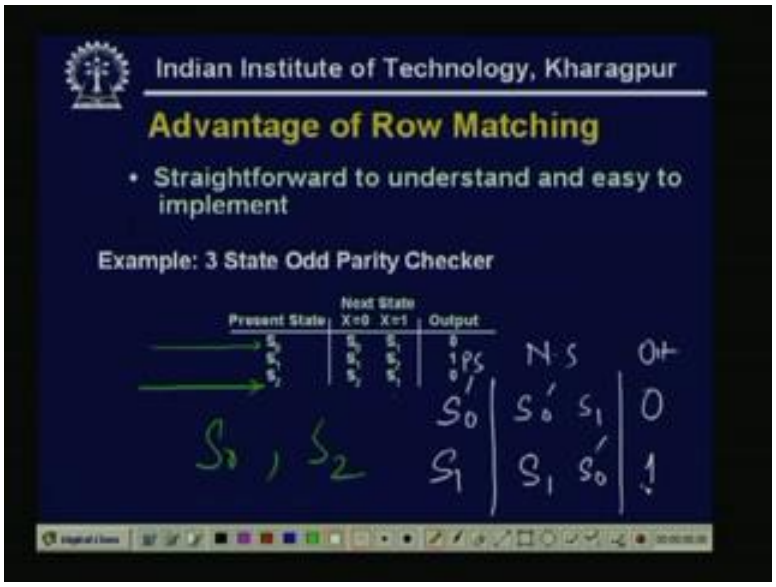
(Refer Slide Time: 51:55)



So, if we draw the final diagram, see, this is my reset state or the initial state. Now say that, we if S 0, so it is a 0 input, 0 slash 0 means, this is 0 if input bit 0, then it is generating, a output bit 0. So, 0 as come, this is 0, then again 1 0, so bit pattern is coming 0 0, then either 0 or 1 that means, 0 0 0 or 0 0 1, in both the cases, it is given output 0. So it will be 0 0, 0 0 1 and 0 or 1, that means, this branch S is either 0 0 0 or 0 0 1 or 0 0 0 0 0 like that, 0 0 0 0 0 or 0 0 1 0 all these patterns, all these patterns will come, 0 0 0 1, 0 0 1 1, this type of all these things.

Now, if it is a 0, then 1, then 0, so see this is a, see this is a 0 as come, Now, 1 1, but 0 1, so 0, now it is a 0 1 1 0, so this is a 0 1 1 0. Then it will give a 1 output, so this is a 0 1 1 0, and then it is a giving a 1 output, and if you consider that 1 0, so we consider 1 0 1 0, again this will be a, 1 output, 1 0 1 0 this will be a 1 output. So, these are the two branches, that, we are getting a output, output 1, and remaining branches, that it will be, so this is a 1 0 1 1, it will be 0, then if it is a 0 1, 1 then it will be a 0, like that, so it will be giving a, this type of pattern matching.

(Refer Slide Time: 55:45)



Indian Institute of Technology, Kharagpur

Advantage of Row Matching

- Straightforward to understand and easy to implement

Example: 3 State Odd Parity Checker

Present State	Next State X=0	Next State X=1	Output
S_0	S_0	S_1	0
S_1	S_1	S_2	1
S_2	S_2	S_1	0

S_0 , S_2

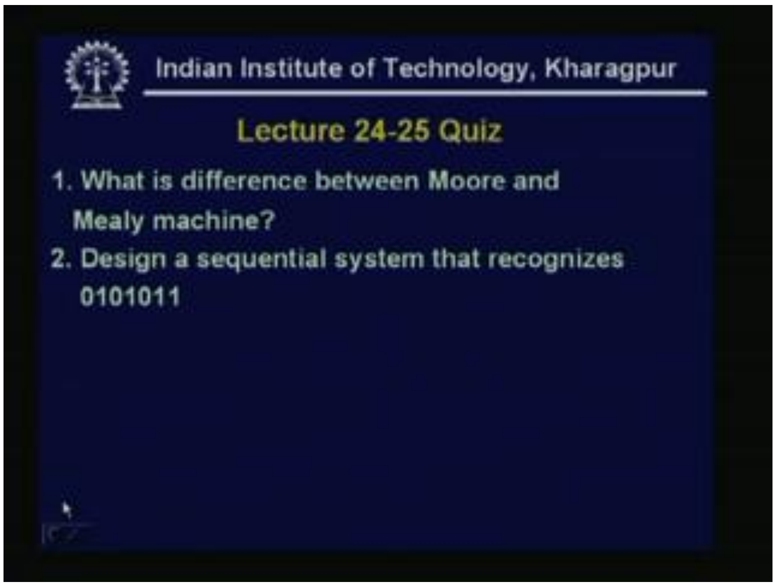
PS	N.S		O/P
S_0'	S_0'	S_1	0
S_1	S_1	S_0'	1

So, mainly this is a very straightforward algorithm, so what we have done, that the all possible inputs, the input output behavior, and the state transition table, we have created. And, then we are trying to check, whether any two row, at least any two row exist for that, that for any input combinations the output behavior is same. Then what we are doing, that this two, we are identifying these two rows, that they can be combined, and then the present states, are a given present state is renamed.

And, that can be a, that can be a state, when we will be doing the state encoding we will be doing the rename state, as encoded state. So, the state odd, parity checker, this will be the S 0, S 1, S 2, and these are the next states, and these outputs 0 1 0. So, these are if we check, this if we check, this table the parity checker, then we will be seeing, that this table are no more combined, that S 0. Then again next state is S 0, S 1 output 0, for the S 0, S 2 say S 2, that it is S 2, S 1 output is 0, so these, output behaviors are same.

So, this is S 0 and S 2, S 0 and S 2 are combined and giving a new name S 0 dash. So, there can be 2 states, S 0 dash and S 1, and for S 0 dash, the next states are same, S 0 dash, S 1, 0. So, what will be the thing, see if it is these two are mapped, then if S 0 and S 2 are mapped, then the table will be, that, that S 0 dash, this is the present state, and next states will be, S 0 dash, S 1 output will be 0, and this is S 1. So this is my present state, and this is S 1, and now S 2 becomes S 0 dash, and output becomes 1. So see far this, this can be easily combined, and can be minimized.

(Refer Slide Time: 58:54)



Indian Institute of Technology, Kharagpur

Lecture 24-25 Quiz

1. What is difference between Moore and Mealy machine?
2. Design a sequential system that recognizes 0101011

So, we consider two quiz questions, for last 2 lectures, the lecture 24 and 25, that what is the difference between Moore and Mealy machine. Because mainly we are designing some machines, and design is sequential system, that recognizes 0 1 0 1 0 1 1. So, just like the, today we have seen 1 0 1 0 pattern recognizer instead of 0 1 0, we want 0 1 0 1 0 1 1, this thing should be recognized.

Thank you.