Finite State Machine 4

Introduction

Chapter 3 already discusses the Mealy and Moore machines. These machines fall in the category of automata with output. Finite state machine, in short FSM, is a machine with a finite number of states. As all finite automata contain finite number of states, they are FSMs also. In this chapter, we shall discuss elaborately the different features of an FSM, some application such as sequence detector, incompletely specified machine, where some states and/or some outputs are not mentioned, definite machine, finite machine, information lossless machine, minimization of machines, etc.

4.1 Sequence Detector

The sequence detector detects and counts a particular sequence from a long input. Let us assume that, in a circuit, a long input of '0' and '1' is fed. Let us also assume that, from there, it needs to count the occurring of a particular sequence where overlapping sequences are also counted. The circuit generates '1' if the particular output pattern is generated; otherwise, '0' is generated.

The sequence detector can be designed either in a Mealy or Moore machine format.

The following examples describe the design of a sequence detector elaborately.

Example 4.1 `Design a two-input two-output sequence detector which generates an output '1' every time the sequence 1001 is detected. And for all other cases, output '0' is generated. Overlapping sequences are also counted.`

Solution: Before designing this circuit, some clarification regarding overlapping sequence is needed. Let the input string be 1001001.

We have to design the circuit in such a way that it will take one input at a time. The input can be either '0' or '1' (two types of input). The output will also be of two types, either '0' or '1'. The circuit can store (memorize) the input string up to four clock pulses from $t_i - 3$ to t_i.

If the input string is placed according to clock pulses, the output will become

	t_1	t_2	t_3	t_4	t_5	t_6	t_7
I/P	1	0	0	1	0	0	1
O/P	0	0	0	1	0	0	1

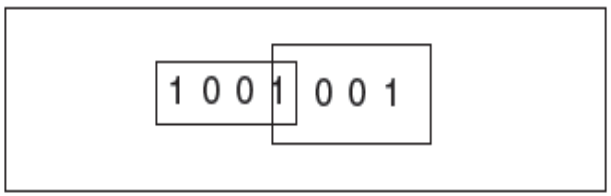

Fig. 4.1 *Overlapping Sequence*

The first input at t_1 is 1, and as there is no input from $t_i - 3$ to ti, the input sequence does not equal 1001. So, the output will be 0. The same cases occur for the inputs up to t_3.

But at time t_4, the input from $t_i - 3$ to t_i becomes 1001, and so the output '1' is produced at time t_4. At time t_5 and t_6, the input string from $t_i - 3$ to t_i are 0010 and 0100, respectively. So, the output '0' is produced. But at t_7 clock pulse, the input string is 1001, and so the output '1' is produced. As the '1' at t_4 is overlapped from t_1 to t_4 and from t_4 to t_7, this is called overlapping condition as shown in Fig. 4.1.

For this case, we have to fi rst draw the state diagram given in Fig. 4.2.

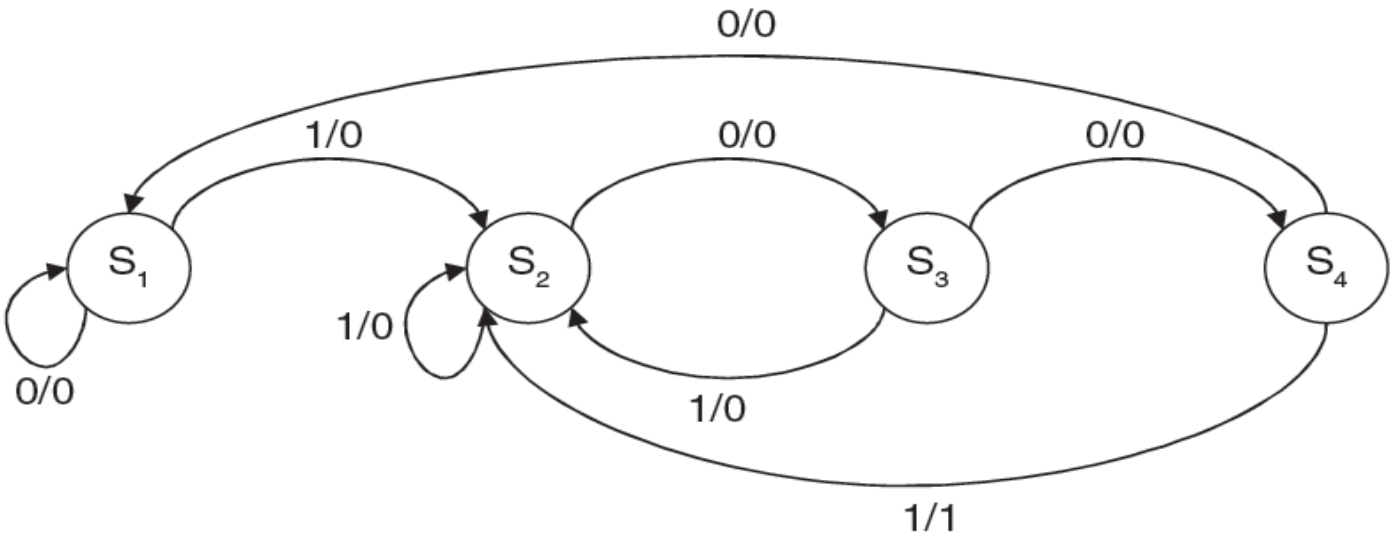

Fig. 4.2 *State Diagram of the Sequence Detector*

In state S_1, the input may be either '0' or '1'. If the input is given '0', there is no chance to get 1001. The machine has to wait for the next input. So, '0' becomes a loop on S1 with output '0'.

If the input is '1', there is a chance to get 1001 by considering that input, and so by getting input '1' the machine moves to S_2 producing output '0'.

In S_2, again the input may be either '0' or '1'. If it is '1', the input becomes 11. There is no chance to get 1001 by taking the previous inputs from $t_i - 1$ to ti. But there is a chance to get 1001 by considering the given input '1' at ti. So, it will be in state S_2. (If it goes to S1, then there will be a loss of clock pulse, which means again from S_1 by taking input '1', it has to come to S_2, i.e., one extra input, which means one clock pulse is needed, and for this the output will not be in right pattern.) If the input is '0', the input becomes 10, by considering the previous input. As there is a chance to get 1001, it will move to S_3.

to get 1001, it shifts to S_4. But if it gets '0', it has no chance to get 1001 considering the previous input, but there is a chance to get 1001 by considering the given input '1'. So, it will shift to S_2 as we know by getting '1' in S_1 the machine comes to S_2.

In S_4, if it gets '0', the input will become 1000, but it does not match with 1001. So it has to start from the beginning, i.e., S1. Getting '1', the string becomes the desired 1001. The overlapping part is the last '1' as given in Fig. 4.1. From the last '1' of 1001, if it gets 001 only, it will give an output '1'. So, it will go to S_2.

State Table: From the previous state diagram, a state graph can be easily constructed.

	$NextState, O/P$	
$State$	$X = 0$	$= 1$
S_1	$S_1, 0$	$S_2, 0$
S_2	$S_3, 0$	$S_2, 0$
S_3	$S_4, 0$	$S_2, 0$
S_4	$S_1, 0$	$S_2, 1$

State Assignment: For making a digital circuit, the states must be assigned to some binary numbers. This is called state assignment. As the number of states is 4, only two-digit is sufficient to represent the four states ($2^2 = 4$).

Let S_1 be assigned to 00, S_2 be assigned to 0_1, S_3 be assigned to 11, S_4 be assigned to 10.

After doing this state assignment, the state table becomes

	Next State, $(Y_1 Y_2)$		$O/P(z)$	
$PresentState(y_1 y_2)$	$X = 0$	$= 1$	$= 0$	$= 1$
00	00	01	0	0
01	11	01	0	0
11	10	01	0	0
10	00	01	0	1

From this state assignment table, the digital function can easily be derived.

Y_1

X / y_1y_2	0	1
00	0	0
01	1	0
11	1	0
10	0	0

Y_2

X / y_1y_2	0	1
00	0	1
01	1	1
11	0	1
10	0	1

z

X / y_1y_2	0	1
00	0	0
01	0	0
11	0	0
10	0	1

$$Y_1 = X'y_2$$
$$Y_2 = X + y_1'y_2$$
$$z = Xy_1y'$$

Y_1 and Y_2 are the next states, which are the memory elements. These will be feedbacked to the input as states y_1 and y_2 with some delay by D flip flop.

The circuit diagram for this sequence detector is given in Fig. 4.3.

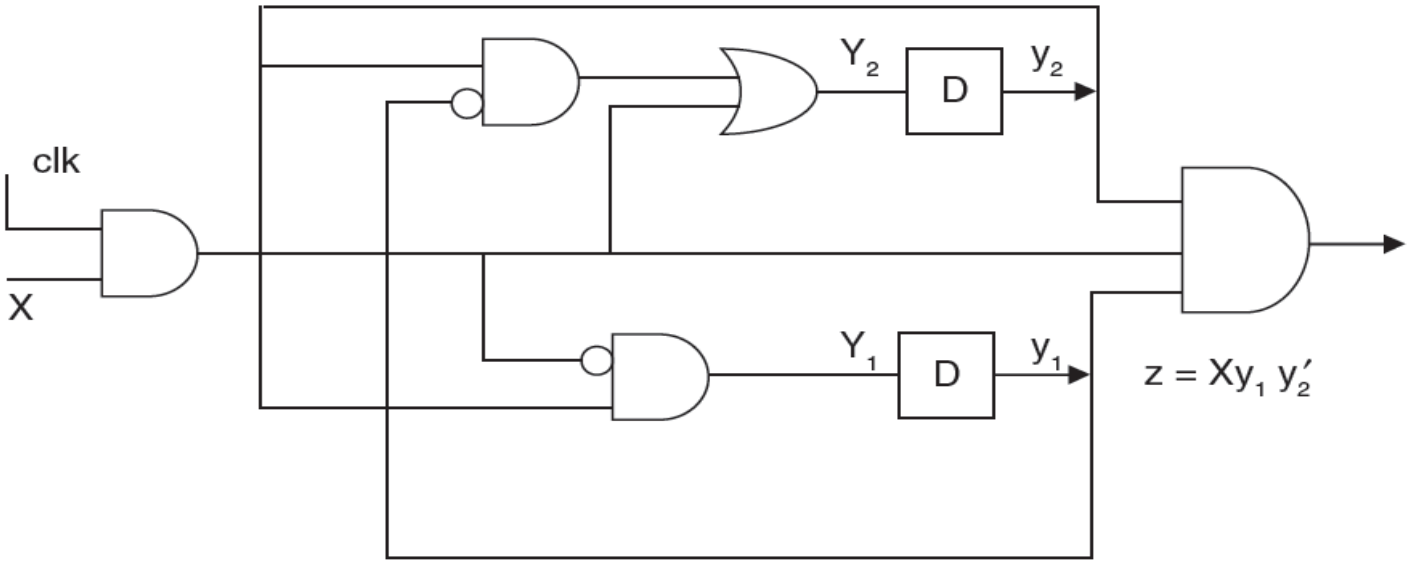

Fig. 4.3 *Digital Circuit for the Sequence Detector*

Example 4.2 Design a two-input two-output sequence detector which generates an output '1' every time the sequence 1010 is detected. And for all other cases, output '0' is generated. Overlapping sequences are also counted.

Solution: The input string 1010100 is placed according to the clock pulses, and it looks like the following.

	t_1	t_2	t_3	t_4	t_5	t_6	t_7
I/P	1	0	1	0	1	0	0
O/P	0	0	0	1	0	1	0

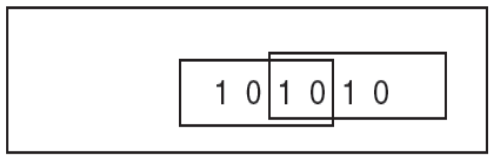

Fig. 4.4 *Overlapping Sequence*

And the output becomes as given earlier. Overlapping portion is 10, as shown in Fig. 4.4.

The state diagram is given in Fig. 4.5.

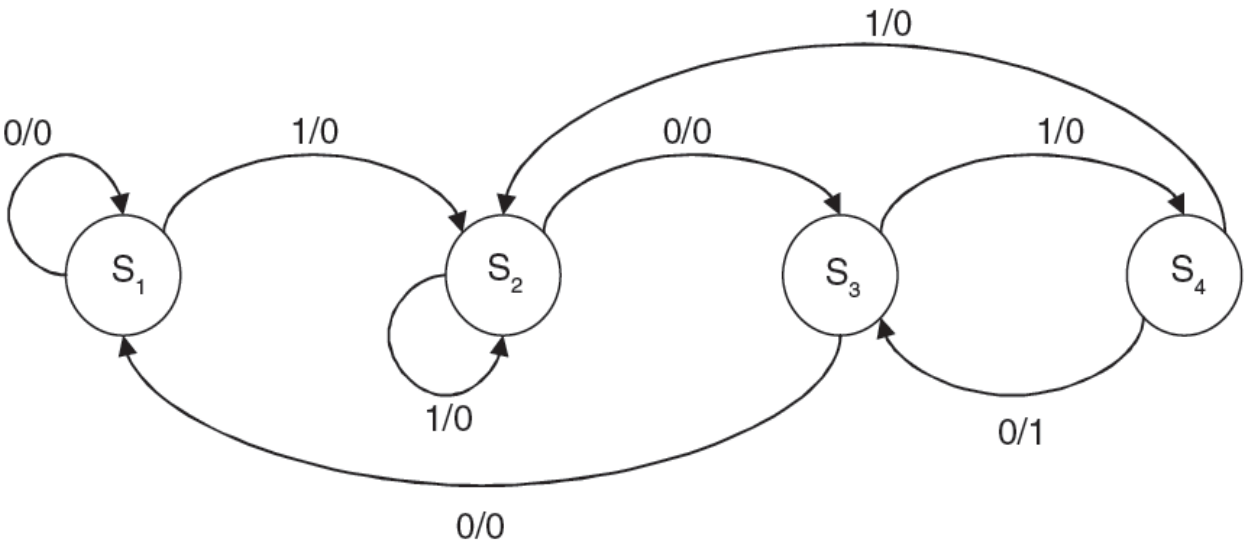

Fig. 4.5 *State Diagram*

State Table: From the previous state diagram, a state table as follows can easily be constructed.

	$NextState, O/P$	
$State$	$X = 0$	$= 1$
S_1	$S_1, 0$	$S_2, 0$
S_2	$S_3, 0$	$S_2, 0$
S_3	$S_1, 0$	$S_4, 0$
S_4	$S_3, 1$	$S_2, 0$

State Assignments: For making a digital circuit, the states must be assigned to some binary numbers to make a digital circuit. This is called state assignment. As the number of states is 4, two-digit is sufficient to represent the four states ($2^2 = 4$).

Let S_1 be assigned to 00, S_2 be assigned to 01, S_3 be assigned to 11, S_4 be assigned to 10.