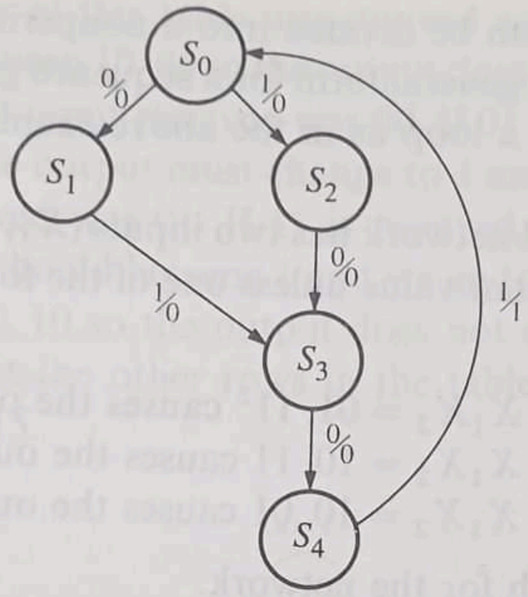


state	sequence received
S_0	reset
S_1	0
S_2	1
S_3	01 or 10
S_4	010 or 100

Figure 14–13
Partial State Graph for Example 1

Note that the network resets to S_0 when the fourth input is received. Next we add arrows and labels to the graph to take care of sequences which do not give a 1 output as shown in Fig. 14–14.

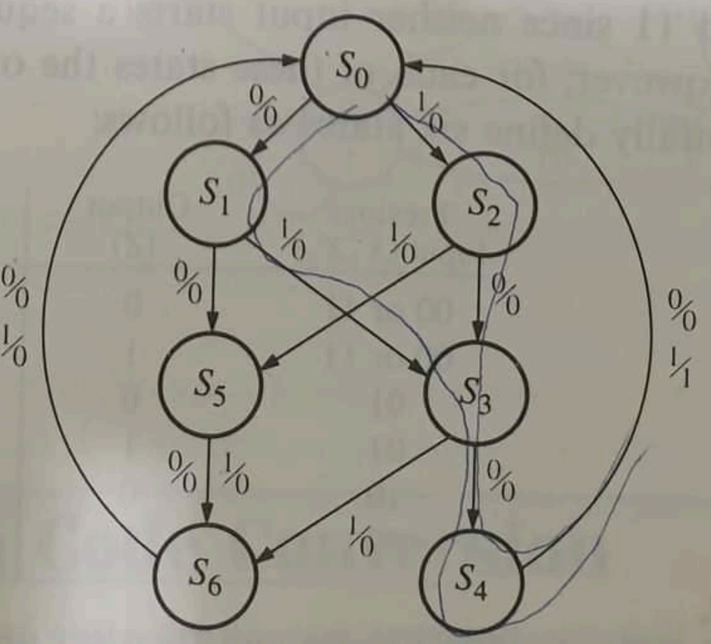


Figure 14–14

Addition of states S_5 and S_6 was necessary so that the network would

added.

6. Check your graph to make sure there is one and only one path leaving each state for each combination of values of the input variables.
7. When your graph is complete, test it by applying the input sequences formulated in part 1 and making sure the output sequences are correct.

Several examples of deriving state graphs or tables follow.

EXAMPLE 1 A sequential network has one input (X) and one output (Z). The network examines groups of four consecutive inputs and produces an output $Z = 1$ if the input sequence 0101 or 1001 occurs. The network resets after every four inputs. Find the Mealy state graph.

Solution: A typical sequence of inputs and outputs is

$$\begin{array}{l|l|l|l} X = 0101 & 0010 & 1001 & 0100 \\ Z = 0001 & 0000 & 0001 & 0000 \end{array}$$

The vertical bars indicate the points at which the network resets to the initial state. Note that an input sequence of either 01 or 10 followed by 01 will produce an output of $Z = 1$. Therefore, the network can go to the same state if either 01 or 10 is received. The partial state graph for the two sequences leading to a 1 output is shown in Fig. 14–13.

14.2 More Complex Design Problems

In this section we will derive a state graph for a sequential network of somewhat greater complexity than the previous examples. The network to be designed again has the form shown in Fig. 14–1. The output Z should be 1 if the input sequence ends in either 010 *or* 1001, and Z should be 0 otherwise. Before attempting to draw the state graph, we will work out some typical input-output sequences to make sure that we have a clear understanding of the problem statement. We will determine the desired output sequence for the following input sequence:

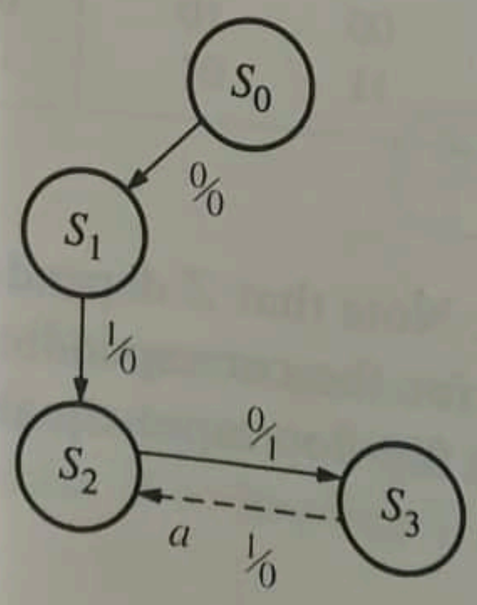
$X = 0\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0$

$a\quad b\quad c\ d\quad e\quad f$

$Z = 0\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0$

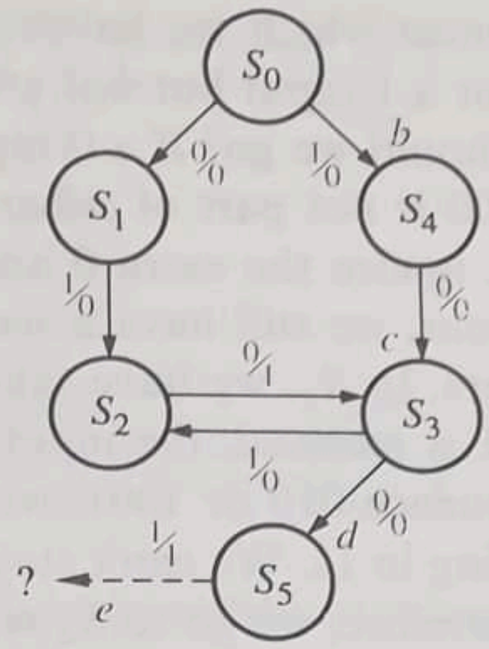
At point a, the input sequence ends in 010, one of the sequences we are looking for, so the output is $Z = 1$. At point b, the input again ends in 010 so $Z = 1$. Note that overlapping sequences are allowed since the problem statement does not say anything about resetting the network when a 1 output occurs. At point c, the input sequence ends in 1001, so Z is again 1. Why do we have a 1 output at points d, e, and f?

We will start construction of the state graph by working with the two sequences which lead to a 1 output. Then we will later add additional arrows and states as required to make sure that the output is correct for other cases. We start off with a reset state S_0 which corresponds to having received no inputs. Whenever an input is received which corresponds to part of one of the sequences we are looking for, the network should go to a new state to "remember" having received this input. Figure 14–7 shows a partial state graph which gives a 1 output for the sequence 010. In this graph S_1 corresponds to having received a sequence ending in 0, S_2 to a sequence ending in 01, and S_3 to a sequence ending in 010. Now, if a 1 input is received in state S_3, we again have a sequence ending in 01, which is part of the input sequence we are looking for. Therefore we can go back to state S_2 (arrow a) since S_2 corresponds to having received a sequence ending in 01. Then if we get another 0 in state S_2, we go to S_3 with a 1 output. This is correct since the sequence again ends in 010.



state	sequence received
S_0	reset
S_1	0
S_2	01
S_3	010

Figure 14–7



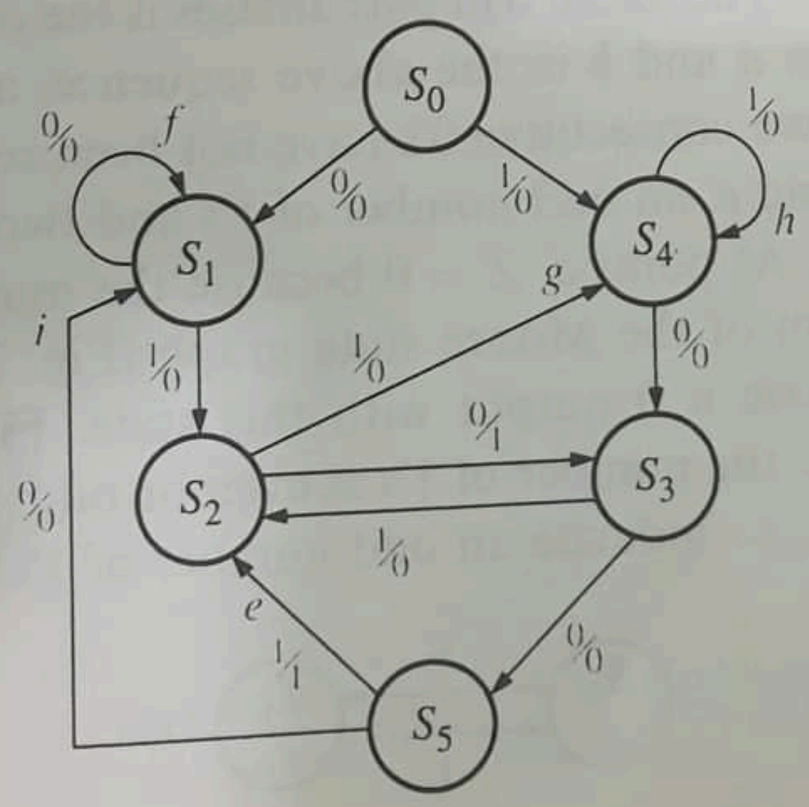
state	sequence ends in
S_0	reset
S_1	0
S_2	01
S_3	010 or 10
S_4	1
S_5	100

Figure 14–8

Next we will construct the part of the graph corresponding to the sequence 1001. Again we start in the reset state S_0, and when we receive a 1 input, we go to S_4 (Fig. 14–8, arrow *b*) to remember that we have received the first 1 in the sequence 1001. The next input in the sequence is 0, and when this 0 is received we should ask the question: Should we create a new state to correspond to a sequence ending in 10, or can we go to one of the previous states on the state graph? Since S_3 corresponds to a sequence ending in 10, we can go to S_3 (arrow *c*). The fact that we didn't have an initial 0 this time does not matter since 10 starts off the sequence we are looking for. If we get a 0 input when in S_3, the input sequence received will end in 100 regardless of the path we took to get to S_3. Since there is no state so far corresponding to the sequence 100, we create a new state S_5 to indicate having received a sequence ending in 100.

If we get a 1 input when in state S_5, this completes the sequence 1001 and gives a 1 output as indicated by arrow *e*. Again we ask the question, can we go back to one of the previous states, or do we have to create a new state. Since the end of the sequence 1001 is 01, and S_2 corresponds to a sequence ending in 01, we can go back to S_2 (Fig. 14–9). If we get another 001, we have again completed the sequence 1001 and get another 1 output.

We have now taken care of putting out a 1 when either the sequence 010 or 1001 is completed. Next, we will go back and complete the state graph to take



state	sequence ends in
S_0	reset
S_1	0
S_2	01
S_3	010 or 10
S_4	1
S_5	100

Figure 14–9

care of the other input sequences which we haven't already accounted for. In state S_1, we have accounted for a 1 input but not a 0 input. If we are in S_1 and get a 0 input, to which state should we go? If a 0 input occurs in S_1, we have a sequence ending in 00. Since 00 is not part of either of the input sequences for which we are looking, we can ignore the extra 0 and stay in S_1 (arrow f). No matter how many extra 0's occur, we still have a sequence ending in 0, and we stay in S_1 until a 1 input occurs. In S_2, we have taken care of the 0 input case, but not the 1 input case. If a 1 is received, the input sequence ends in 11. Since 11 is not part of either the sequence 010 or 1001, we do not need a state which corresponds to a sequence ending in 11. We can't stay in S_2 since S_2 corresponds to a sequence ending in 01. Therefore, we go to S_4 which corresponds to having received a sequence ending in 1 (arrow g). S_3 already has arrows corresponding to 0 and 1 inputs, so we examine S_4 next. If a 1 is received in S_4, the input sequence ends in 11. We can stay in S_4 and ignore the extra 1 (arrow h) since 11 is not part of either sequence for which we are looking. In S_5, if we get a 0 input, the sequence ends in 000. Since 000 is not contained in either 010 or 1001, we can go back to S_1 since S_1 corresponds to having received a sequence ending in one (or more) 0's. This completes the state graph since every state has arrows leaving it which correspond to both 0 and 1 inputs. We should now go back and check the state graph against the original input sequences to make sure that a 1 output is always obtained for a sequence ending in 010 or 1001 and that a 1 output does not occur for any other sequence.

Next we will derive the state graph for a Moore sequential network with one input X and one output Z. The output Z is to be 1 if the total number of 1's received is odd and at least two consecutive 0's have been received. A typical input and output sequence is

$$X = \begin{array}{ccccccccc} 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ \uparrow & & & \uparrow & & \uparrow & \uparrow & \uparrow \\ a & & & b & & c & d & e \end{array}$$

$$Z = (0)\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1$$

We have shifted the Z sequence to the right to emphasize that for a Moore network an input change does not affect Z immediately, but Z can change only after the next clock pulse. The initial 0 in parentheses is the output associated with the reset state. At points a and b in the above sequence, an odd number of 1's has been received, but two consecutive 0's have not been received, so the output remains 0. At points c and e, an odd number of 1's and two consecutive 0's have