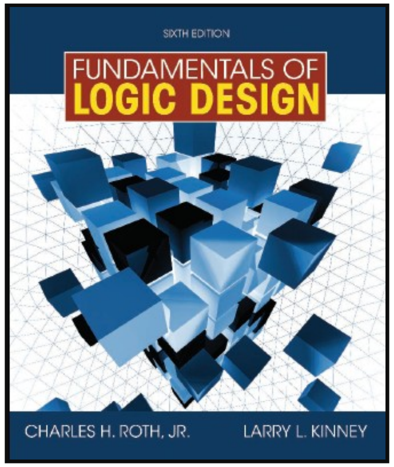
CHAPTER 14

DERIVATION OF STATE GRAPHS AND TABLES



This chapter in the book includes:

Objectives
Study Guide
14.1 Design of a Sequence Detector
14.2 More Complex Design Problems
14.3 Guidelines for Construction of State Graphs
14.4 Serial Data Code Conversion
14.5 Alphanumeric State Graph Notation
Programmed Exercises
Problems

Design of a Sequence Detector

To illustrate the design of a clocked Mealy sequential circuit, we will design a sequence detector. The circuit is of the form:

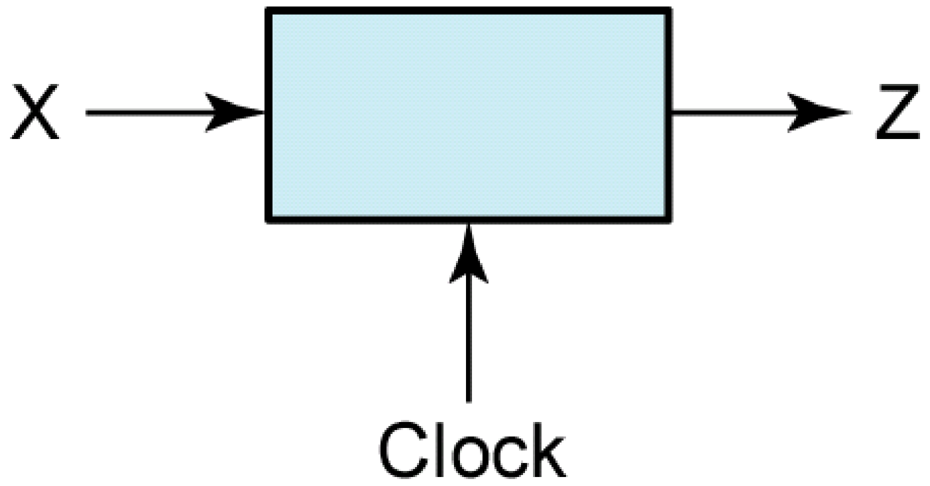


Figure 14-1: **Sequence Detector to be Designed**

Suppose we want to design the sequence detector so that any input sequence ending in 101 will produce an output $Z = 1$ coincident with the last 1. The circuit does not reset when a 1 output occurs. A typical input sequence and the corresponding output sequence are:

$X =$	0	0	1	1	0	1	1	0	0	1	0	1	0	1	0	0	
$Z =$	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	(14-1)
(time:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	

Initially, we do not know how many flip-flops will be required, so we will designate the circuit states as S_0, S_1, etc. We will start with a reset state designated S_0. If a 0 input is received, the circuit can stay in S_0 because the input sequence we are looking for does not start with a 0.

However, if a 1 is received, the circuit must go to a new state (S_1) to “remember” that the first input in the desired sequence has been received.

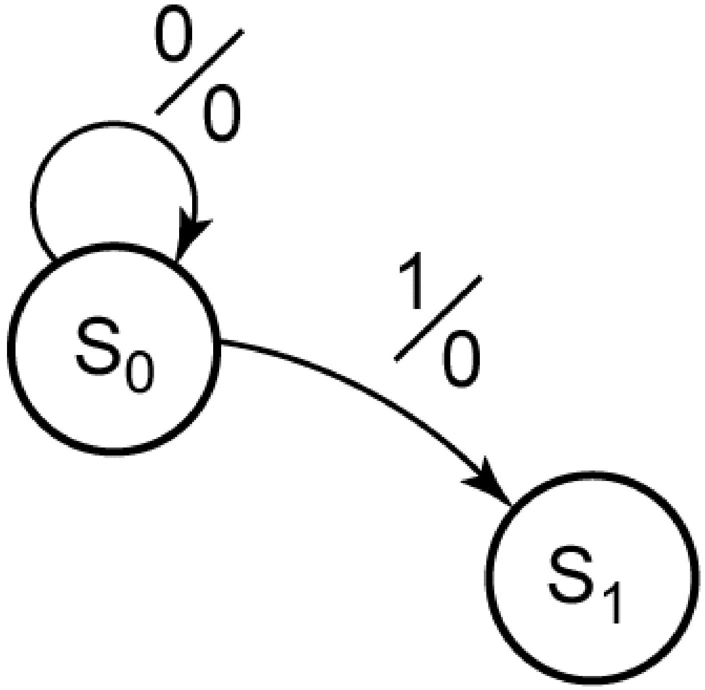


Figure 14-2

When in state S_1, if we receive a 0, the circuit must change to a new state (S_2) to remember that the first two inputs of the desired sequence (101) have been received.

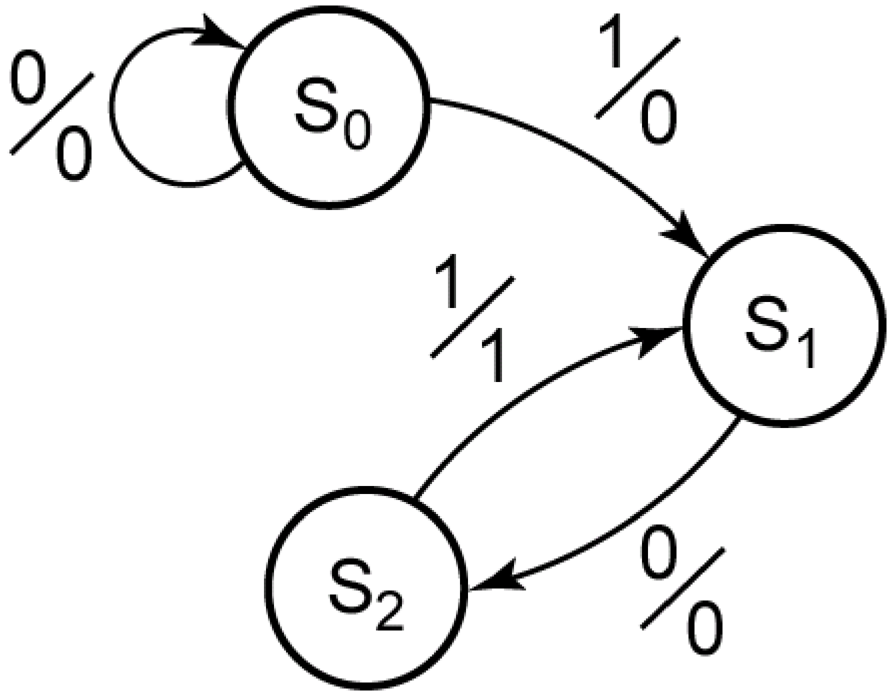


Figure 14-3

If a 1 is received in state S_2, the desired input sequence (101) is complete and the output should be 1. Since the last 1 in a sequence can also be the first 1 in a new sequence, we should return to S_1.

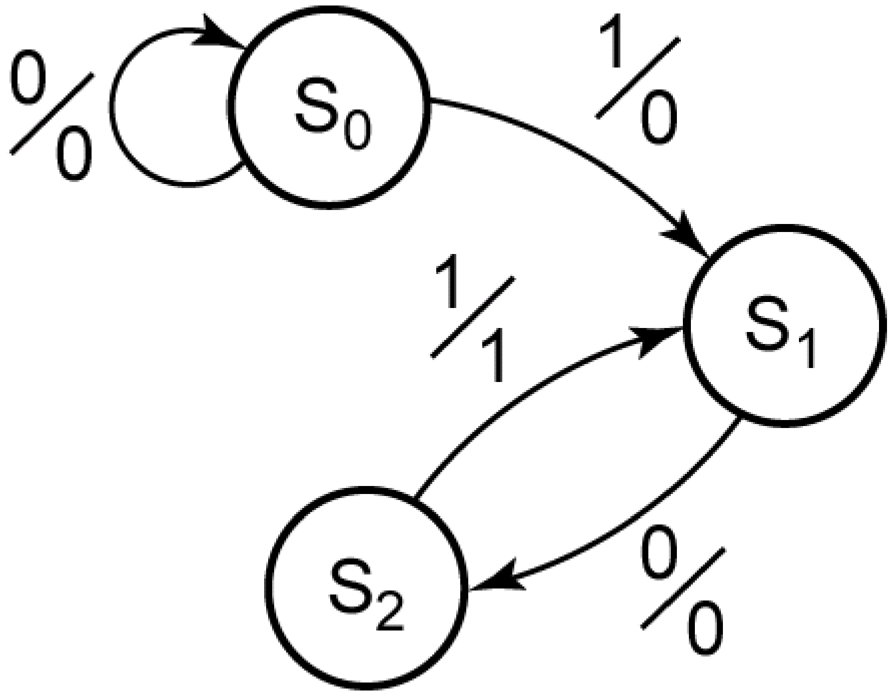


Figure 14-3

If a 1 input occurs when in state S_1, we can stay in S_1 because the sequence is simply restarted. If a 0 input occurs in state S_2, we have received two 0's in a row and must reset the circuit to S_0 because 00 is not part of the desired sequence.

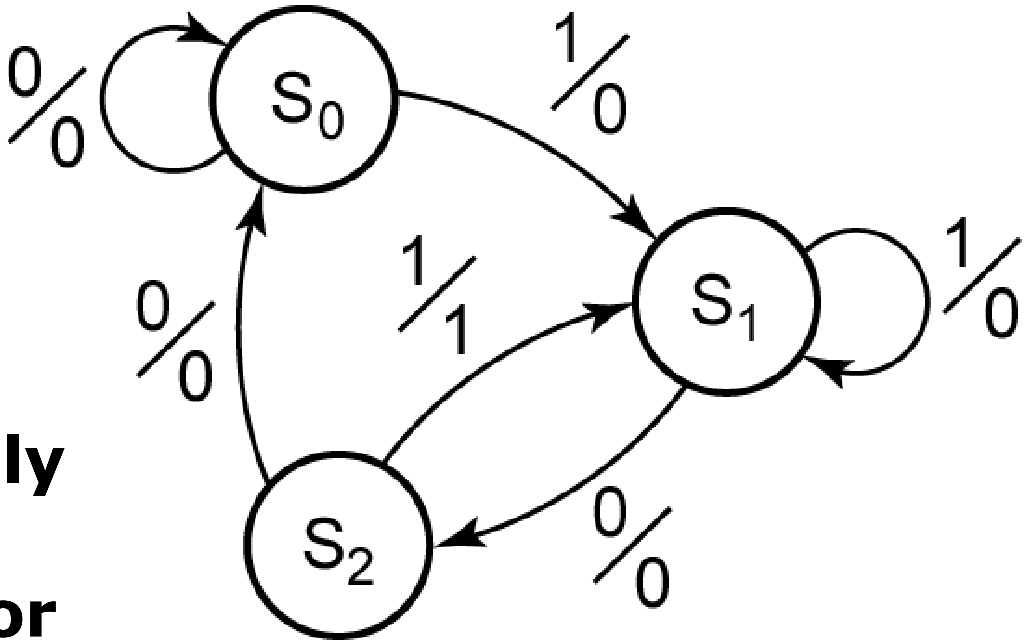


Figure 14-4: **Mealy State Graph for Sequence Detector**

We can then convert our state graph to a state table:

Table 14-1

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

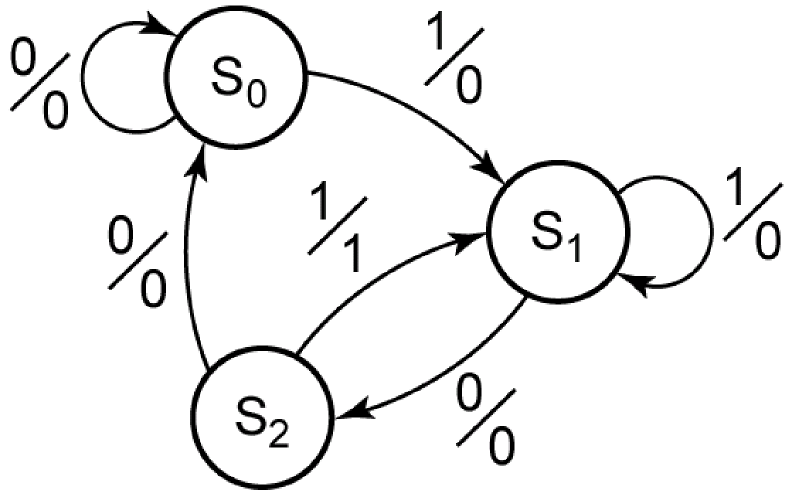


Figure 14-4

Since there are 3 states, we only need 2 flip-flops for the circuit (2 memory bits).

Now we can convert our state table into a transition table:

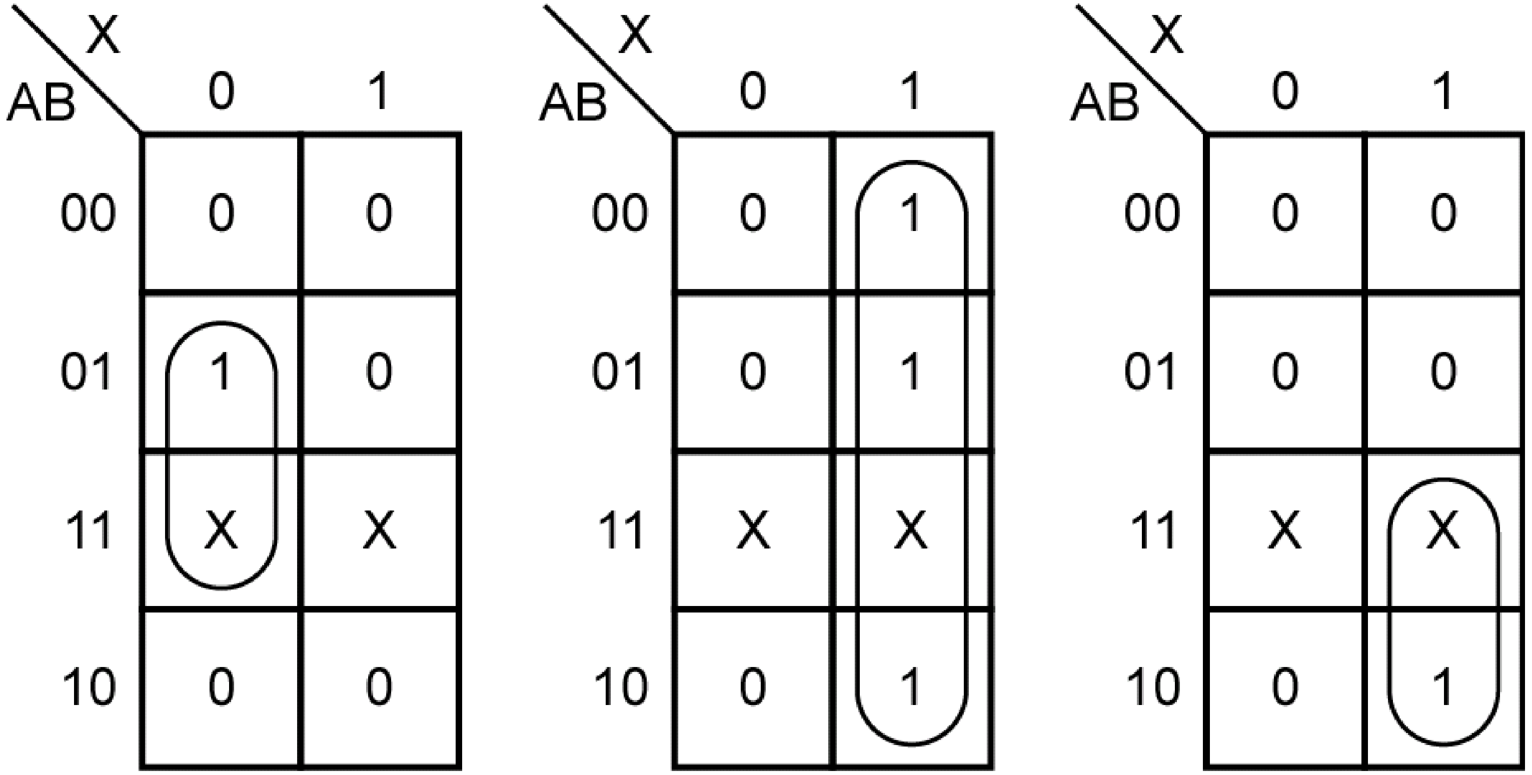
Table 14-1

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

Table 14-2

AB	A^+B^+		Z	
	$X = 0$	$X = 1$	$X = 0$	$X = 1$
00	00	01	0	0
01	10	01	0	0
10	00	01	0	1

From the transition table, we can plot the next-state maps for the flip-flops and the map for the output function *Z*:



AB \ X	0	1
00	0	0
01	1	0
11	X	X
10	0	0

$A^+ = X'B$

AB \ X	0	1
00	0	1
01	0	1
11	X	X
10	0	1

$B^+ = X$

AB \ X	0	1
00	0	0
01	0	0
11	X	X
10	0	1

$Z = XA$

Section 14.1 (p. 433)

Now we can draw the circuit corresponding to the equations we derived:

$A^+ = X'B$

$B^+ = X$

$Z = XA$

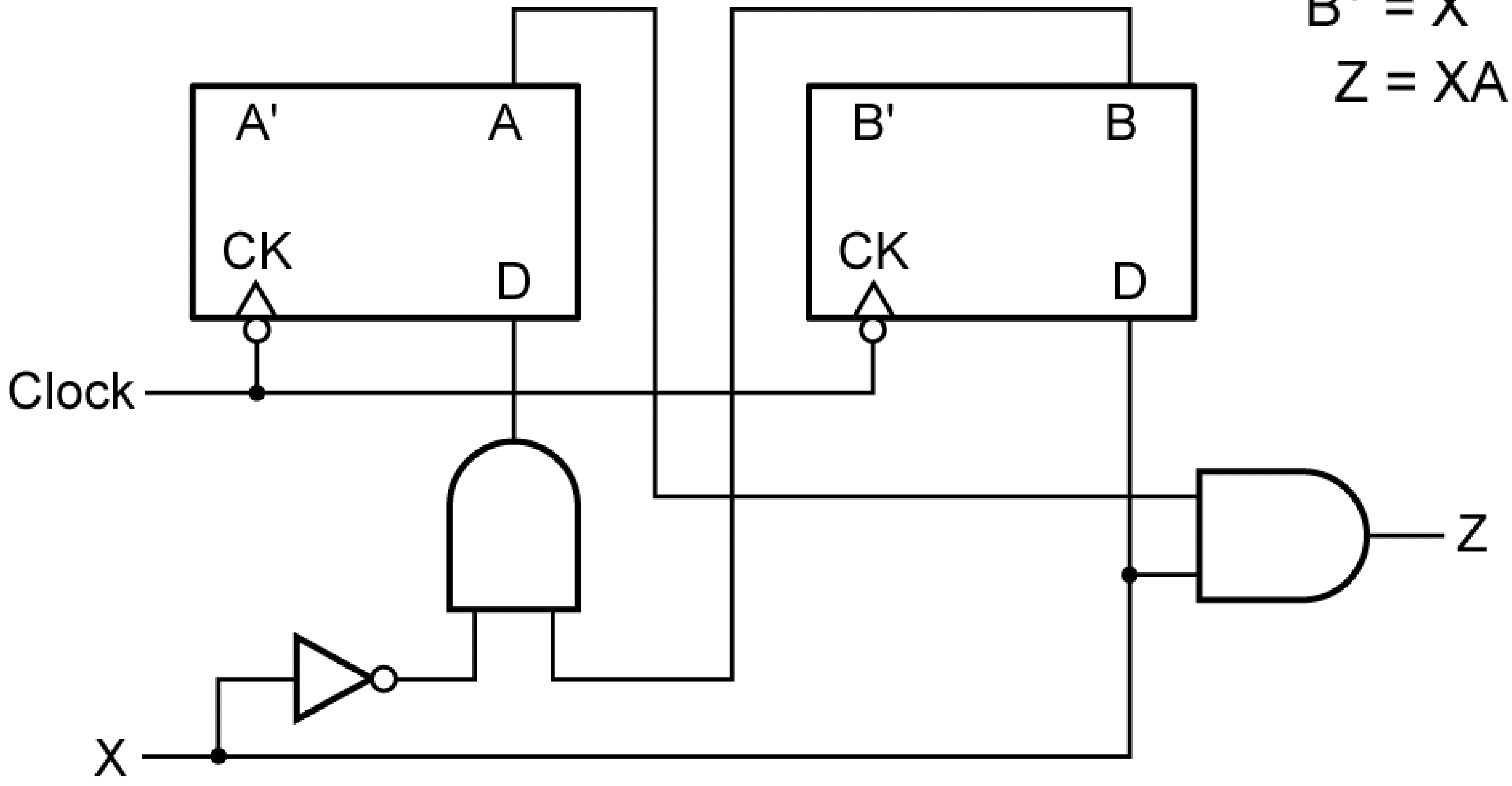
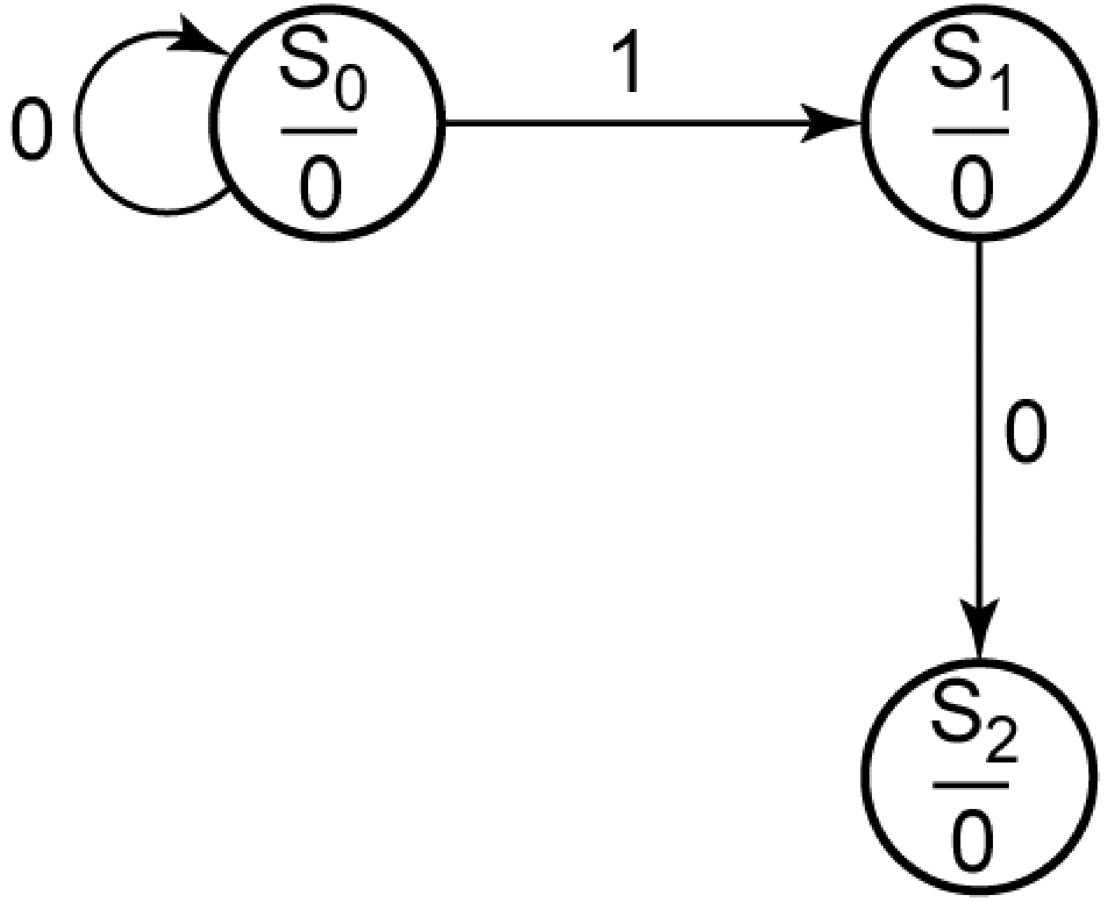


Figure 14-5

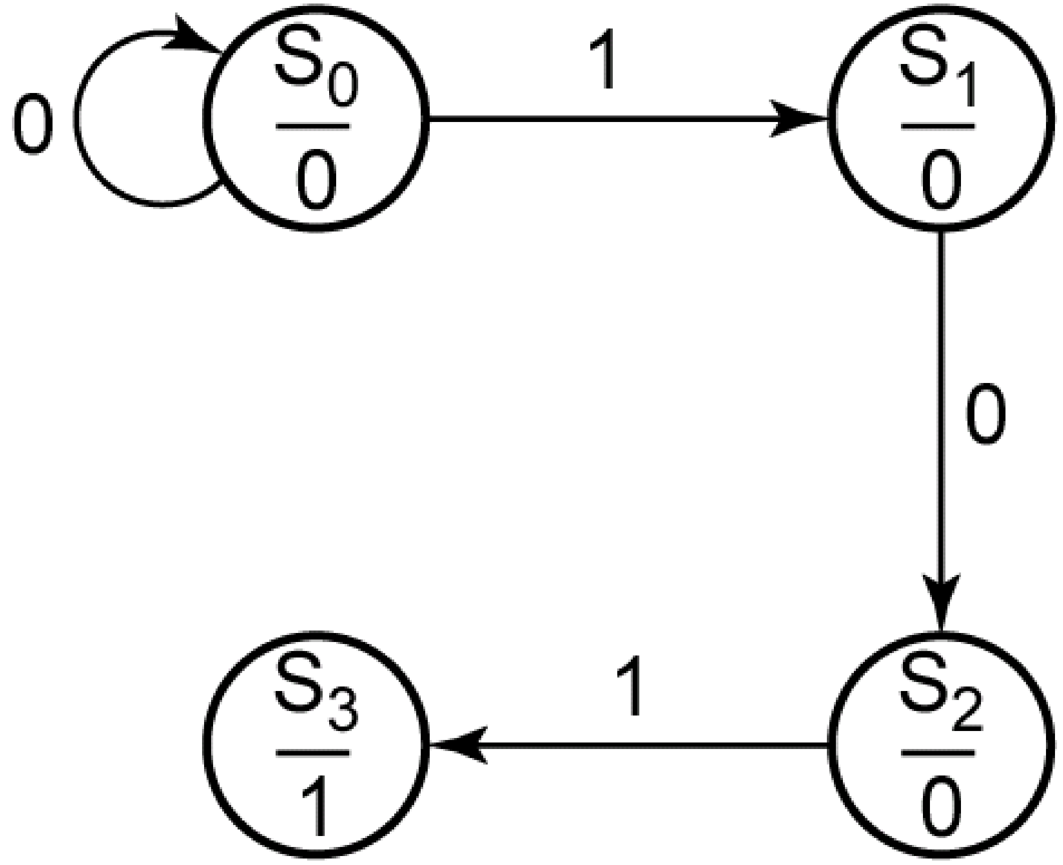
The procedure for finding the state graph for a Moore machine is similar to that used for a Mealy machine, except that the output is written with the state.

We will rework the previous example as a Moore machine: the circuit should produce an output of 1 only if an input sequence ending in 101 has occurred.

The design is similar to that for the Mealy machine up until the input sequence 10 has occurred, except that 0 output is associated with states S_0, S_1, and S_2:



Now, when a 1 input occurs to complete the 101 sequence, the output must become 1; therefore, we cannot go back to state S_1 and must create a new state S_3 with a 1 output:



We now complete the graph, as shown below. Note the sequence 100 resets the circuit to S_0. A sequence 1010 takes the circuit back to S_2 because another 1 input should cause Z to become 1 again:

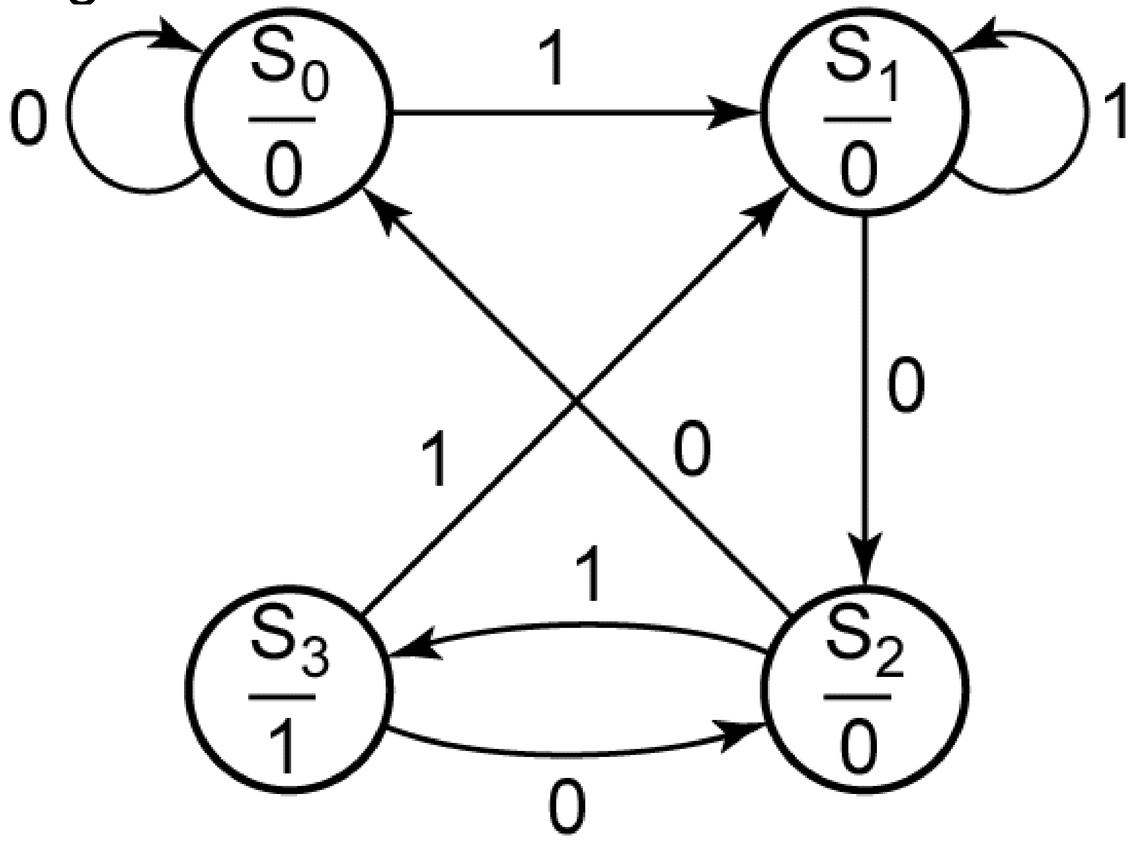


Table 14-3 and Table 14-4

Present State	Next State $X = 0$	Next State $X = 1$	Present Output(Z)
S_0	S_0	S_1	0
S_1	S_2	S_1	0
S_2	S_0	S_3	0
S_3	S_2	S_1	1

	A^+B^+		
AB	$X = 0$	$X = 1$	Z
00	00	01	0
01	11	01	0
11	00	10	0
10	11	01	1

As with the Mealy machine, we can derive the state and transition tables for the Moore machine from the state graph.

More Complex Design Problems

Now we will derive a state graph for a sequential circuit of somewhat greater complexity than the previous examples.

For this circuit, the output *Z* should be 1 if the input sequence ends in either 010 or 1001, and *Z* should be 0 otherwise.

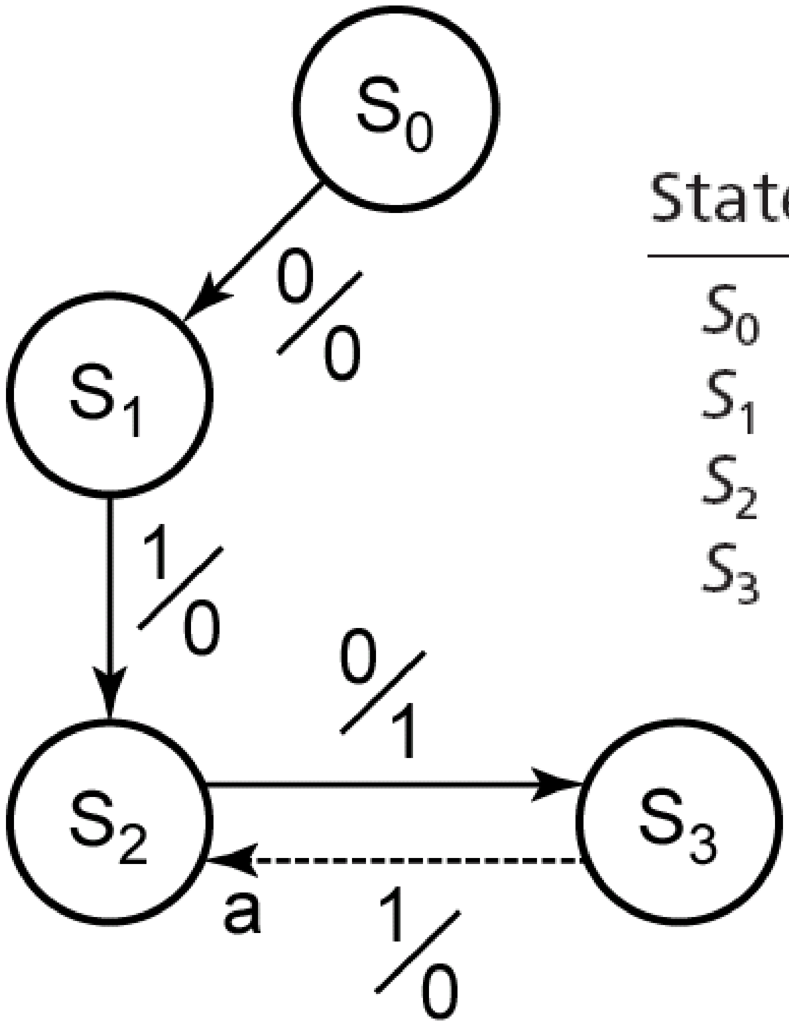
Ex:

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

Section 14.2 (p. 435)

Mealy Example

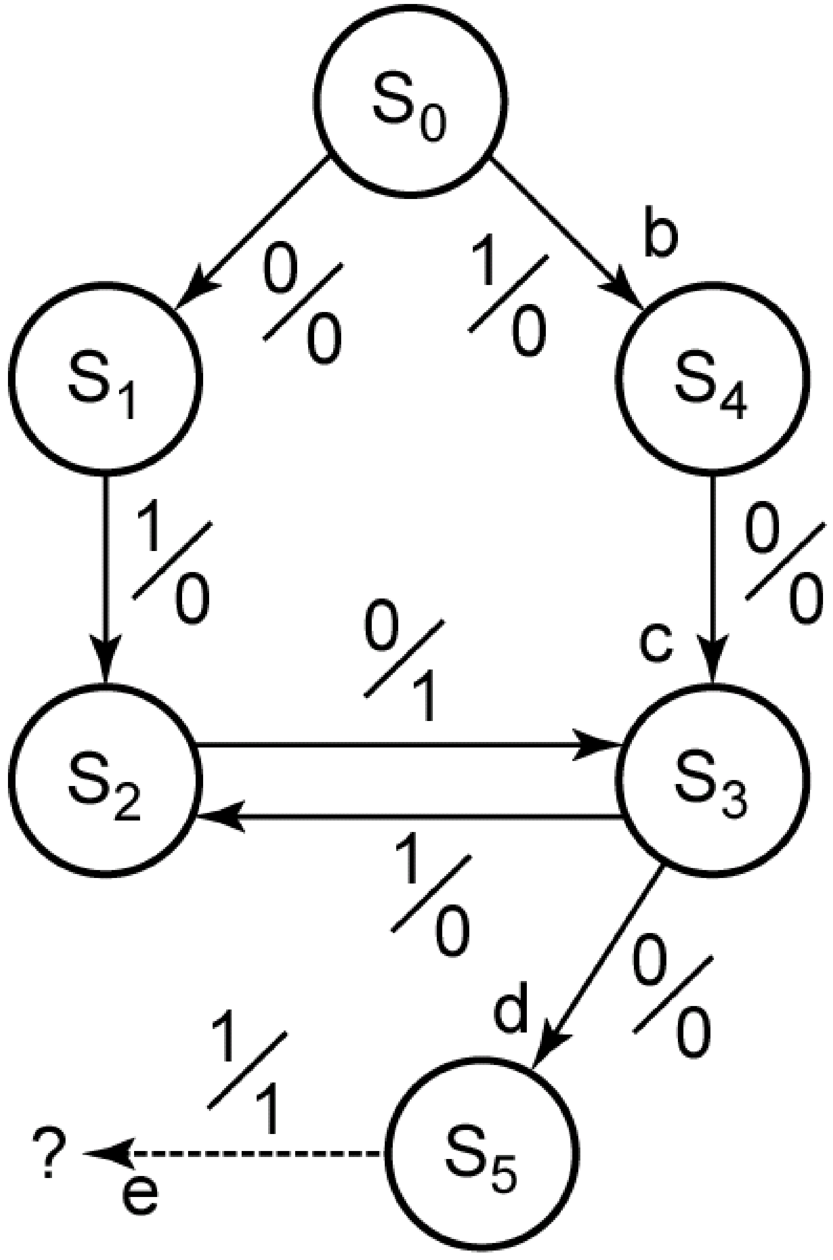
We will start construction of the state graph by working with the two sequences which lead to a 1 output. Then, we will later add arrows and states as required to make sure that the output is correct for other cases.



State	Sequence Received
S_0	Reset
S_1	0
S_2	01
S_3	010

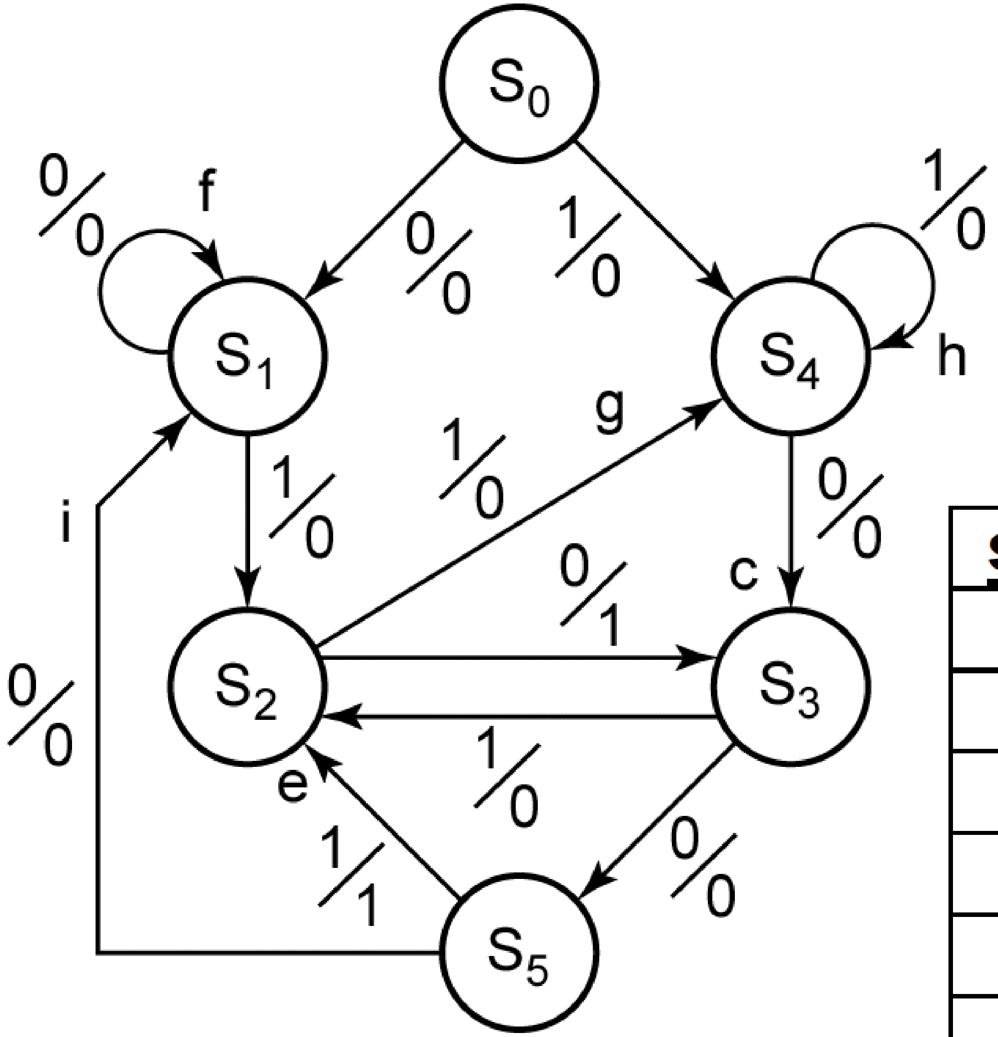
Figure 14-7

Next, we construct the part of the graph corresponding to the sequence 1001, starting from the reset state S_0.



state	sequence ends in
S_0	reset
S_1	0 (but not 10)
S_2	01
S_3	10
S_4	1 (but not 01)
S_5	100

Figure 14-8



Now we fill in the missing arcs. With each arc, we first ask if we can go back to one of the previous states or do we have to create a new state?

state	sequence ends in
S_0	reset
S_1	0 (but not 10)
S_2	01
S_3	10
S_4	1 (but not 01)
S_5	100

Figure 14-9

Moore Example

Design a Moore sequential circuit 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:

$$\begin{array}{llllllllll}
X = & & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
 & & \uparrow & & & \uparrow & & \uparrow & \uparrow & \uparrow \\
 & & a & & & b & & c & d & e \\
Z = & (0) & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1
\end{array}$$

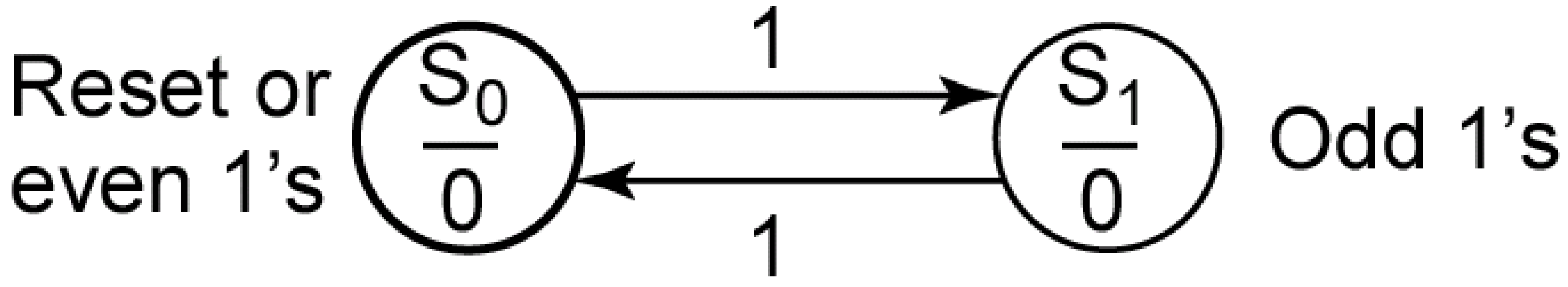
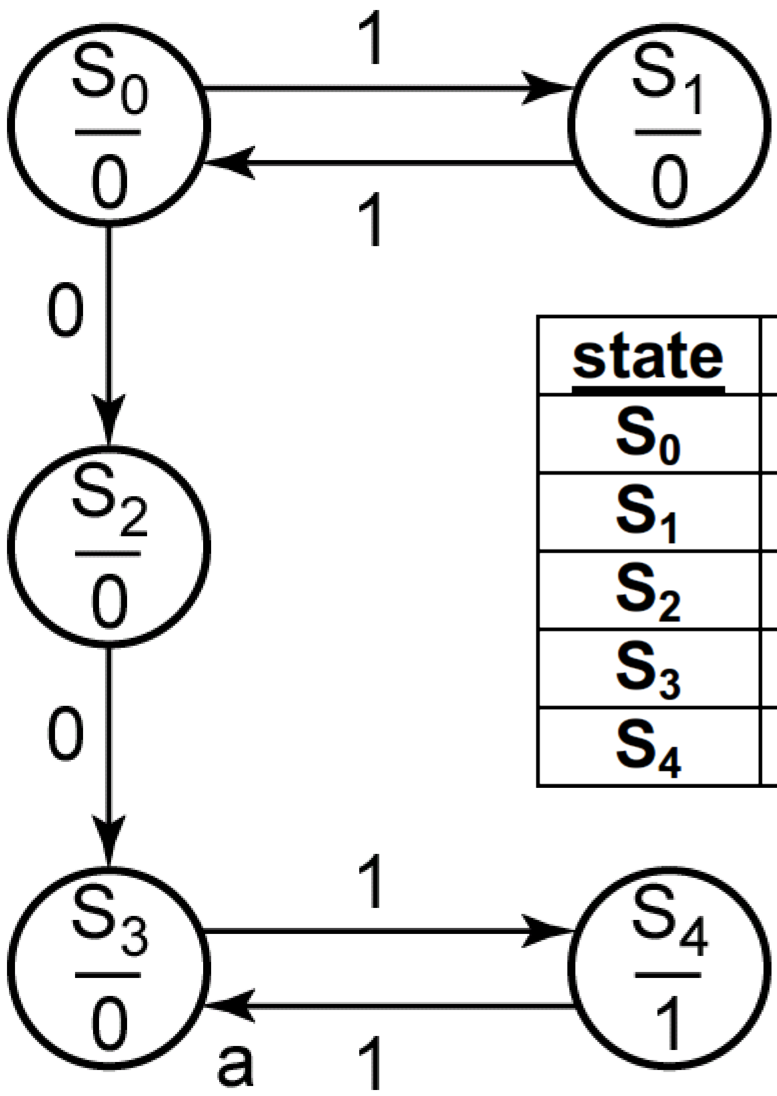
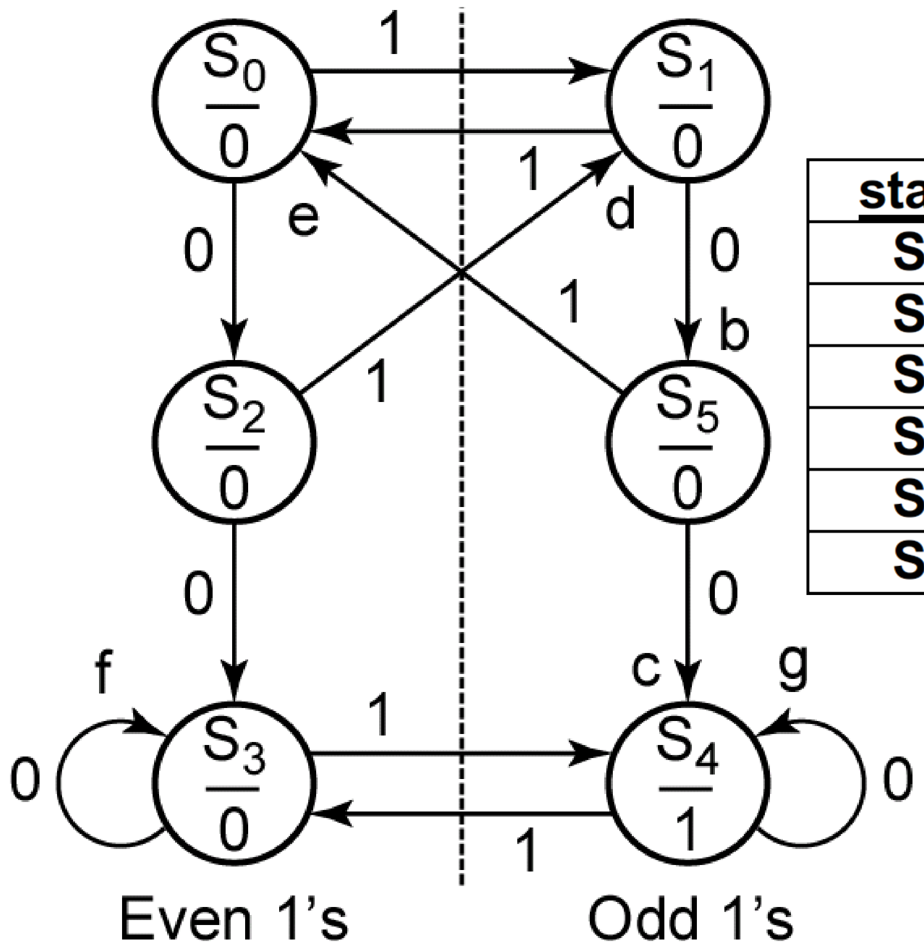


Figure 14-10



state	sequence received
S_0	reset or even 1's
S_1	odd 1's
S_2	even 1's and ends in 0
S_3	even 1's and 00 has occurred
S_4	00 has occurred and odd 1's

Figure 14-11



state	input sequences
S_0	reset or even 1's
S_1	odd 1's
S_2	even 1's and ends in 0
S_3	even 1's and 00 has occurred
S_4	odd 1's and 00 has occurred
S_5	odd 1's and ends in 0

Figure 14-12

Guidelines for Construction of State Graphs

Although there is no one specific procedure which can be used to derive state graphs or tables for every problem, the following guidelines should prove helpful:

1. First, construct some sample input and output sequences to make sure that you understand the problem statement.
2. Determine under what conditions, if any, the circuit should reset to its initial state.
3. If only one or two sequences lead to a nonzero output, a good way to start is to construct a partial state graph for those sequences.

Section 14.3 (p. 439)

4. Another way to get started is to determine what sequences or groups of sequences must be remembered by the circuit and set up states accordingly.
5. Each time you add an arrow to the state graph, determine whether it can go to one of the previously defined states or whether a new state must be 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.

Example 1

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

1. First, construct some sample input and output sequences to make sure that you understand the problem statement.

A typical sequence of inputs and outputs is

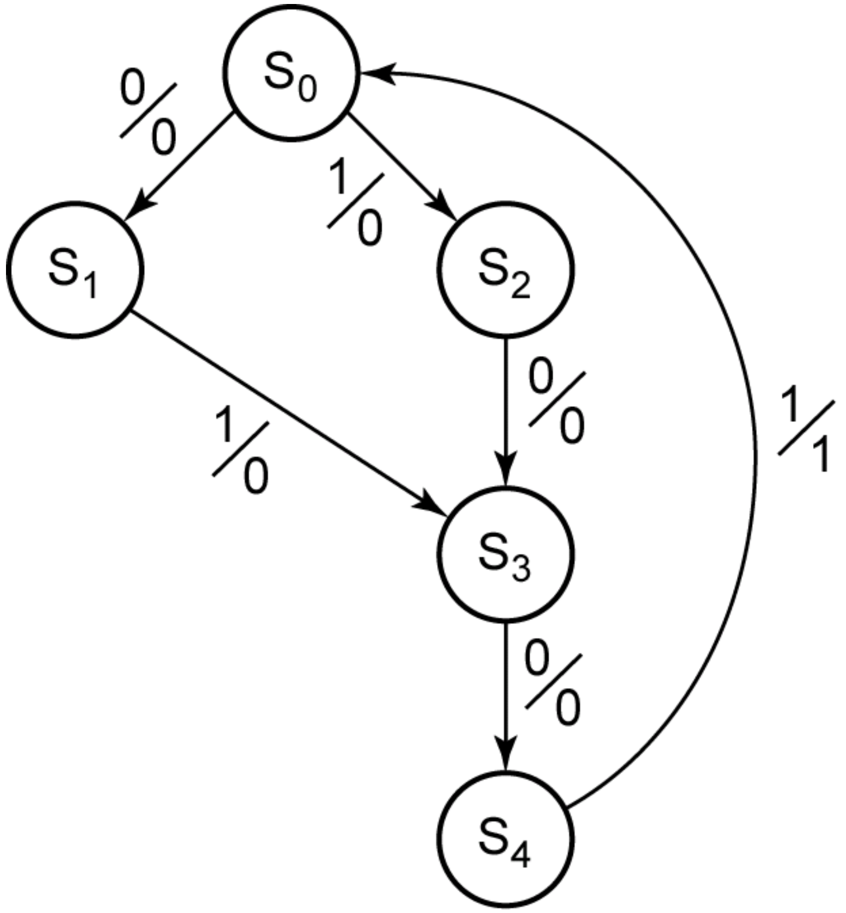
$$\begin{array}{l|l|l|l} X = 0101 & 0010 & 1001 & 0100 \\ Z = 0001 & 0000 & 0001 & 0000 \end{array}$$

Section 14.3 (p. 439)

2. Determine under what conditions, if any, the circuit should reset to its initial state.

Since the circuit examines groups of four consecutive inputs and resets after each group of four, the circuit should reset to S_0 after every fourth input is received.

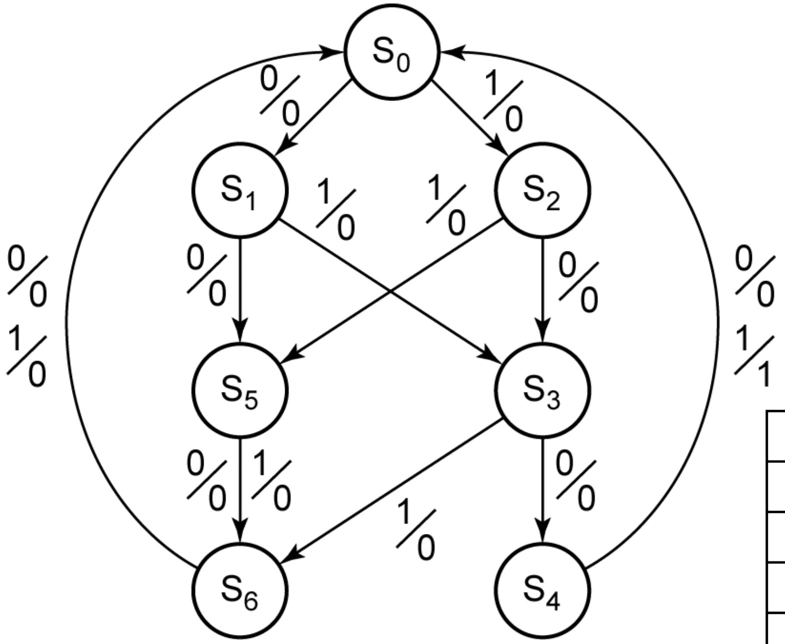
4. Another way to get started is to determine what sequences or groups of sequences must be remembered by the circuit and set up states accordingly.



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**

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.



state	sequence received
S_0	reset
S_1	0
S_2	1
S_3	01 or 10
S_4	010 or 100
S_5	2 inputs received, no 1 output is possible
S_6	3 inputs received, no 1 output is possible

Figure 14-14: **Complete State Graph for Example 1**

Example 2

A sequential circuit has one input (X) and two outputs(Z_1 and Z_2). An output $Z_1 = 1$ occurs every time the input sequence 100 is completed, provided that the sequence 010 has never occurred. An output $Z_2 = 1$ occurs every time the input sequence 010 is completed. Note that once a $Z_2 = 1$ output has occurred, $Z_1 = 1$ can never occur but not vice versa. Find a Mealy state graph and table.

A typical sequence of inputs and outputs is:

$$\begin{array}{rcccccccccc|ccccccccccccc}
X = & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\
Z_1 = & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
Z_2 = & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0
\end{array}$$

Section 14.3 (p. 440-441)

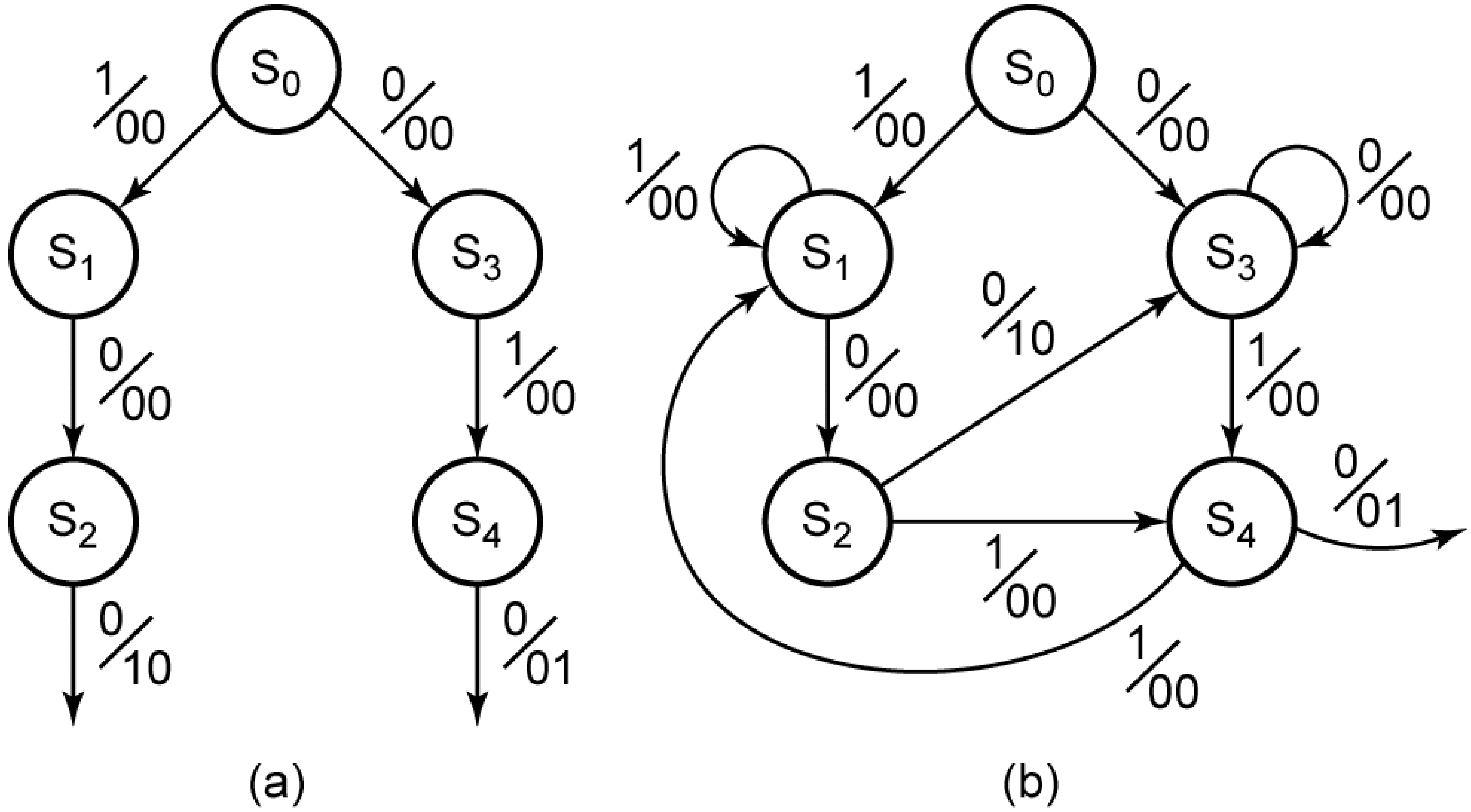
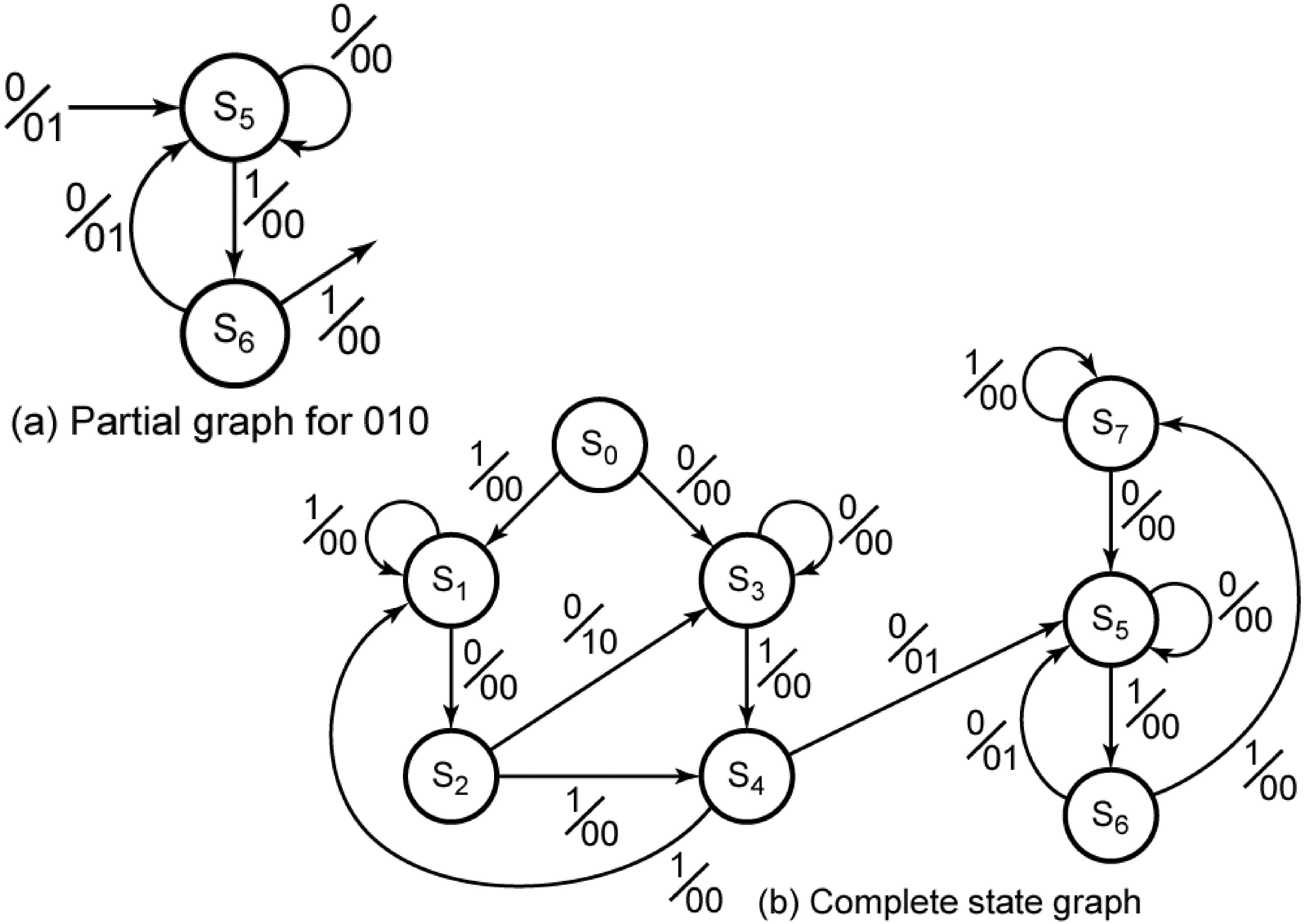


Figure 14-15: **Partial Graphs for Example 2**

Keeping track of what is remembered by each state will help us make the correct state graph.

Table 14-5 State Descriptions for Example 2

State	Description		
S_0	No progress on 100	No progress on 010	
S_1	Progress of 1 on 100	No progress on 010	
S_2	Progress of 10 on 100	Progress of 0 on 010	010 has never occurred
S_3	No progress on 100	Progress of 0 on 010	
S_4	Progress of 1 on 100	Progress of 01 on 010	
S_5		Progress of 0 on 010	
S_6		Progress of 01 on 010	010 has occurred
S_7		No progress on 010	



(a) Partial graph for 010

(b) Complete state graph

Figure 14-16: **State Graphs for Example 2**

Table 14-6.

Present State	Next State X = 0	Next State X = 1	Output (Z_1Z_2) X = 0	Output (Z_1Z_2) X = 1
S_0	S_3	S_1	00	00
S_1	S_2	S_1	00	00
S_2	S_3	S_4	10	00
S_3	S_3	S_4	00	00
S_4	S_5	S_1	01	00
S_5	S_5	S_6	00	00
S_6	S_5	S_7	01	00
S_7	S_5	S_7	00	00

Example 3

A sequential circuit has two inputs (X_1, X_2) and one output (Z). The output remains a constant value unless one of the following input sequences occurs:
(a) The input sequence X_1X_2 = 01, 11 causes the output to become 0.
(b) The input sequence X_1X_2 = 10, 11 causes the output to become 1.
(c) The input sequence X_1X_2 = 10, 01 causes the output to change value.

(The notation X_1X_2 = 01, 11 means X_1 = 0, X_2 = 1 followed by X_1 = 1, X_2 = 1.)
Derive a Moore state graph for the circuit.

Section 14.3 (p. 443)

Previous Input (X_1X_2)	Output (Z)	State Designation
00 or 11	0	S_0
00 or 11	1	S_1
01	0	S_2
01	1	S_3
10	0	S_4
10	1	S_5

Table 14-7

Present State	Z	Next State $X_1X_2 = 00$	01	11	10
S_0	0	S_0	S_2	S_0	S_4
S_1	1	S_1	S_3	S_1	S_5
S_2	0	S_0	S_2	S_0	S_4
S_3	1	S_1	S_3	S_0	S_5
S_4	0	S_0	S_3	S_1	S_4
S_5	1	S_1	S_2	S_1	S_5

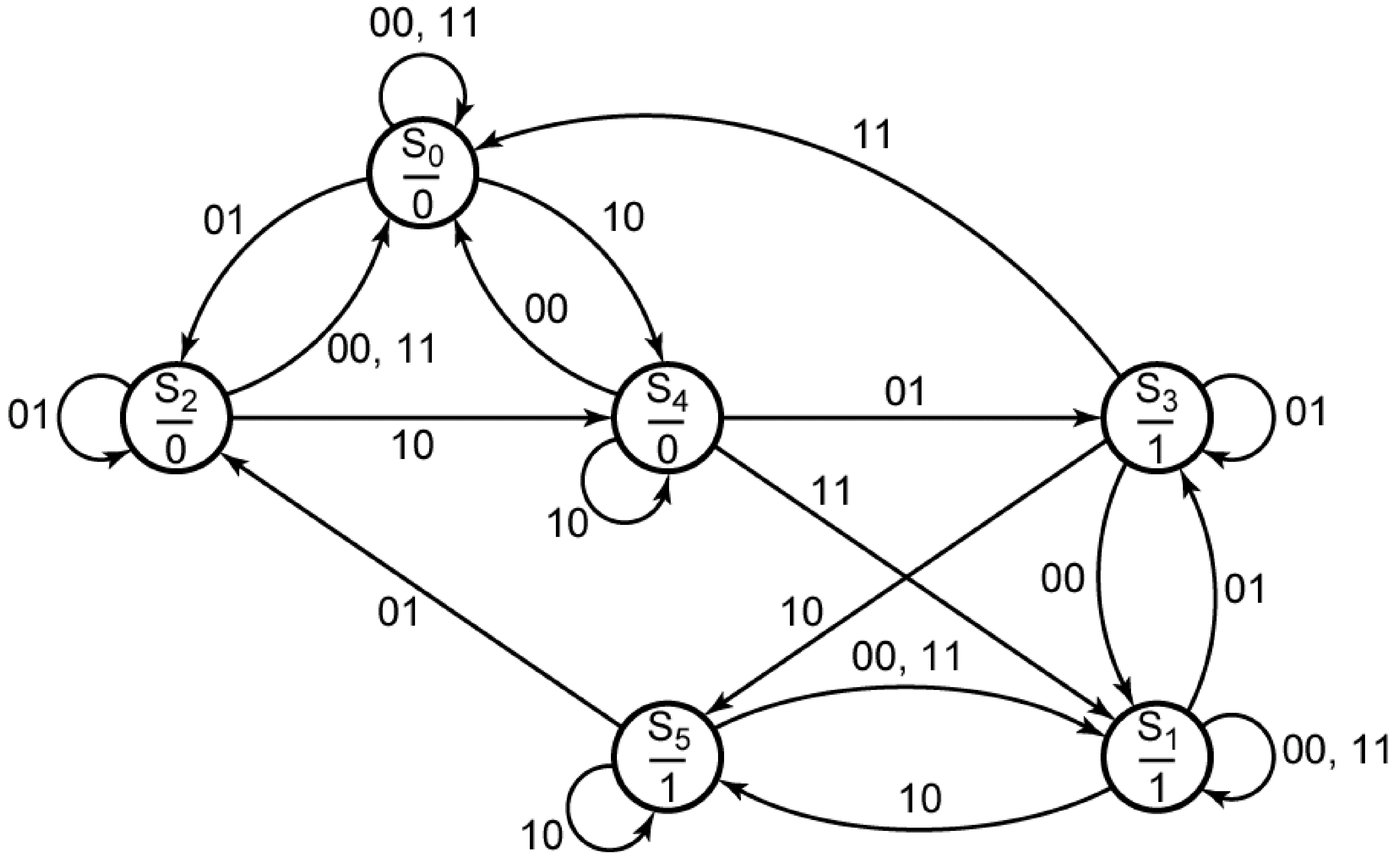


Figure 14-17: **State Graph for Example 3**

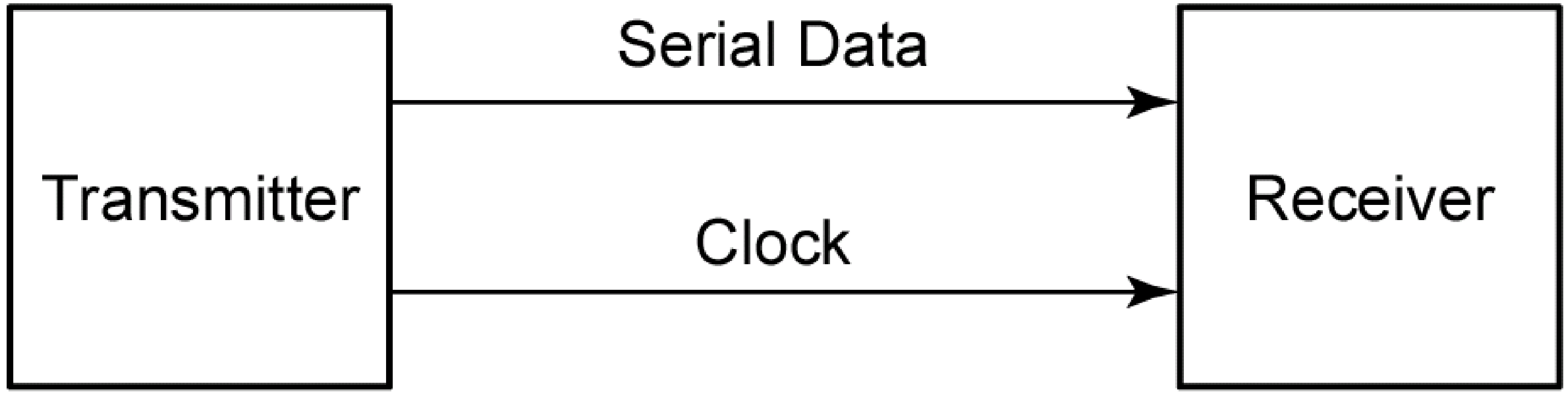
Serial Data Code Conversion

As a final example of state graph construction, we will design a converter for serial data. Binary data is frequently transmitted between computers as a serial stream of bits.

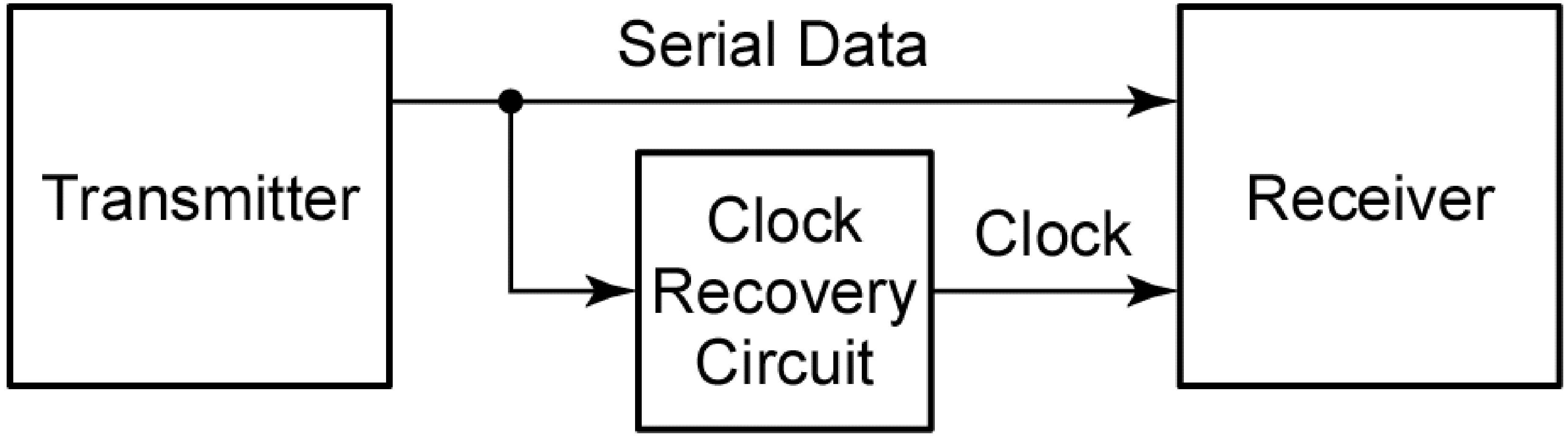
A clock signal is often transmitted along with the data so the receiver can read the data at the proper time.

Alternatively, only the serial data is transmitted, and a clock recovery circuit (called a digital phase-locked loop) is used to regenerate the clock signal at the receiver.

Section 14.4 (p. 444)



(a)



(b)

***Figure 14-18:* Serial Data Transmission**

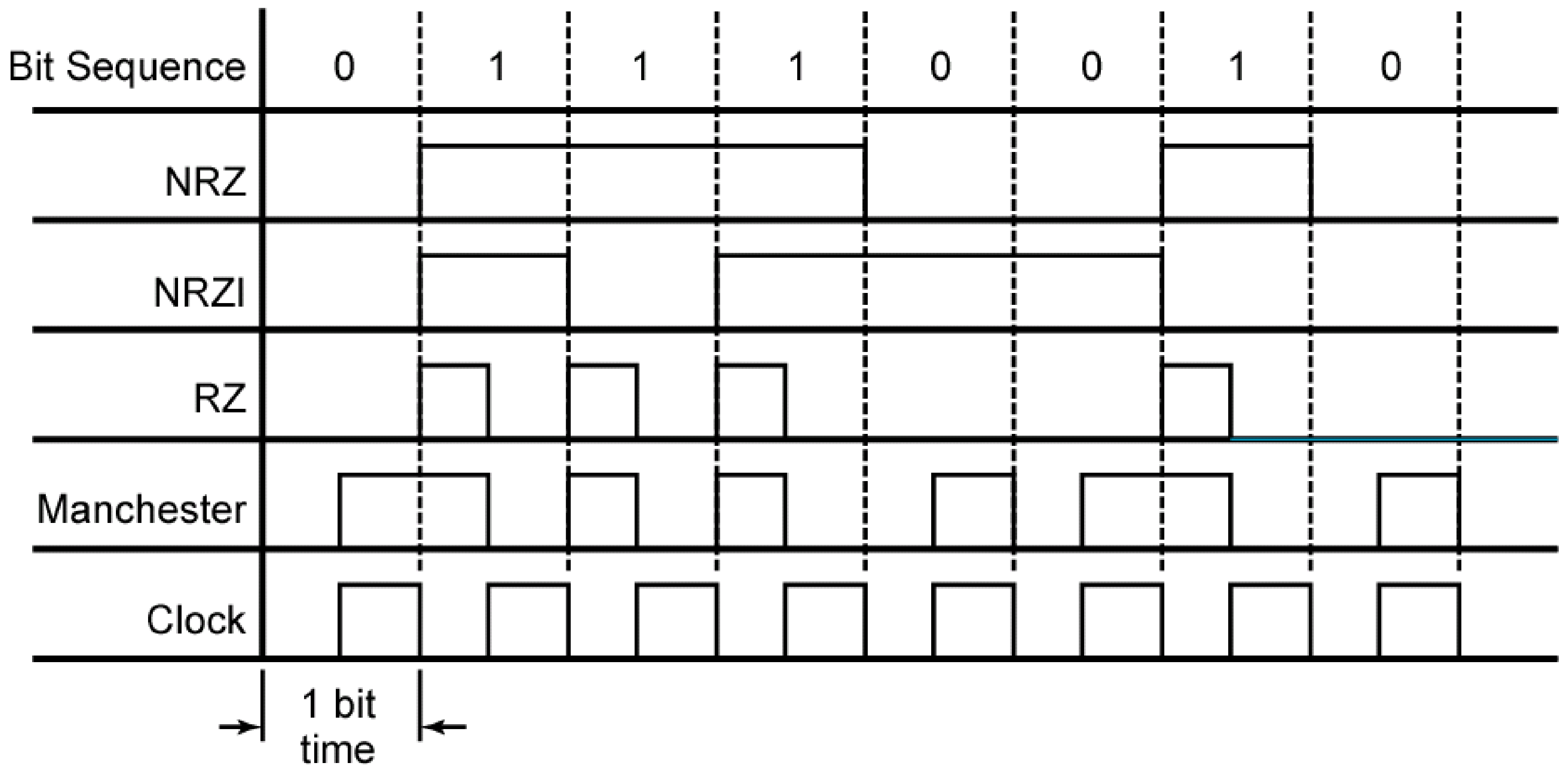


Figure 14-19: **Coding Schemes for Serial Data Transmission**

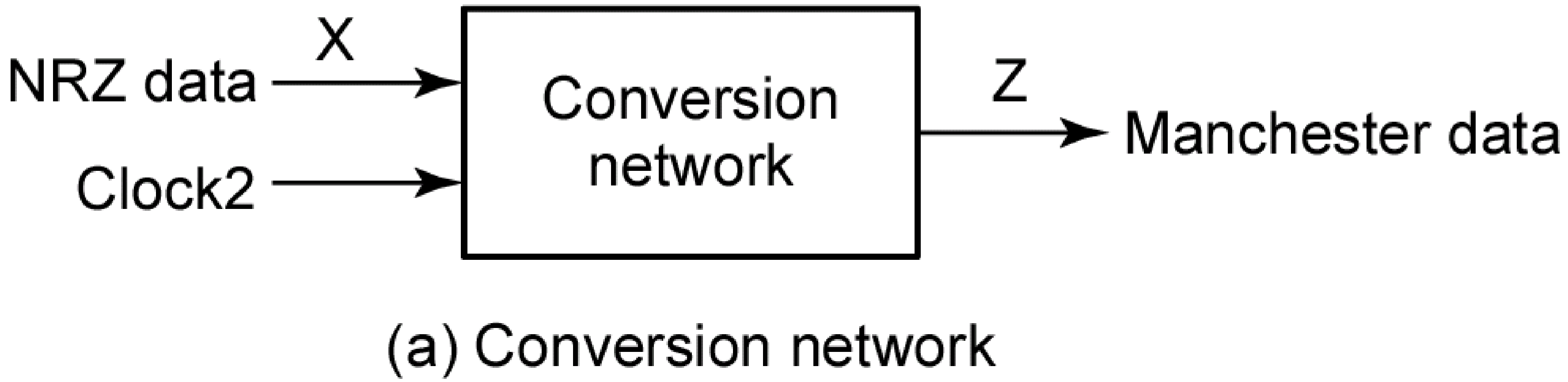


Figure 14-20a:

Mealy Circuit for NRZ-to-Manchester Conversion

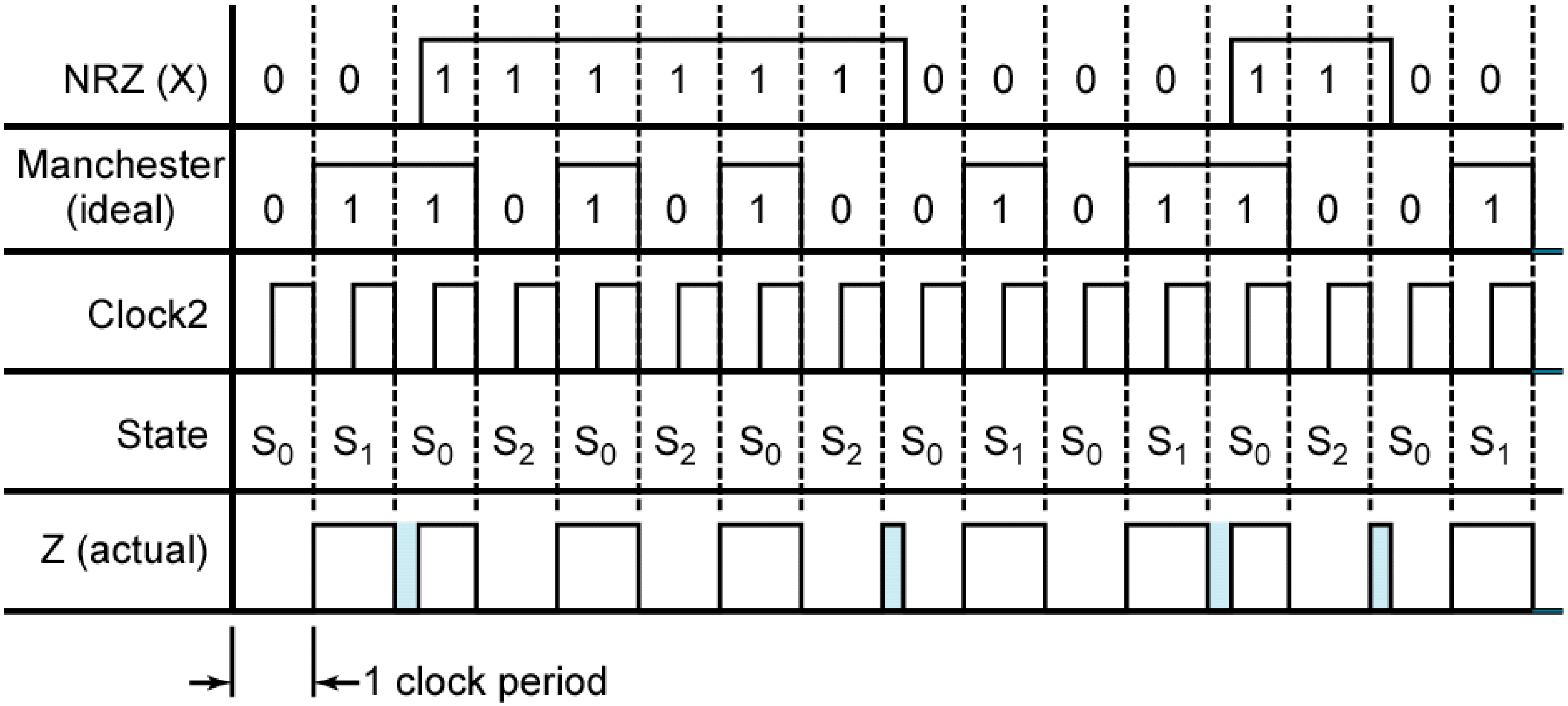
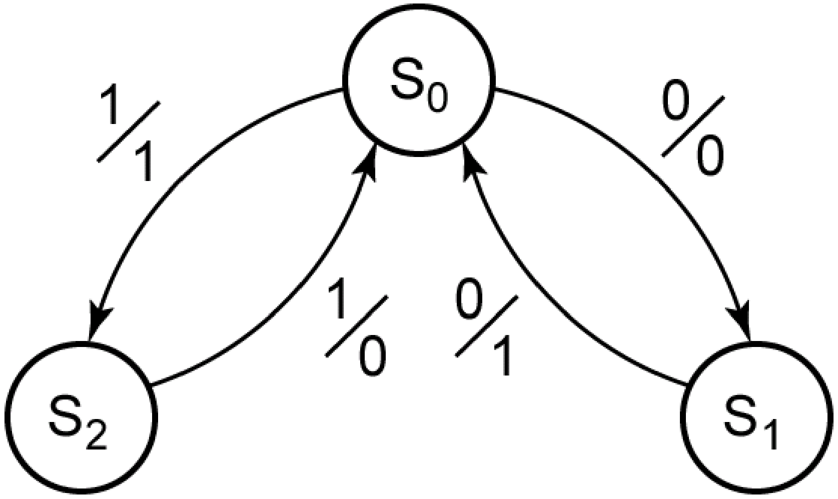


Figure 14-20b:

Mealy Circuit for NRZ-to-Manchester Conversion

Present State	Next State $X = 0$	Next State $X = 1$	Output (Z) $X = 0$	Output (Z) $X = 1$
S_0	S_1	S_2	0	1
S_1	S_0	–	1	–
S_2	–	S_0	–	0

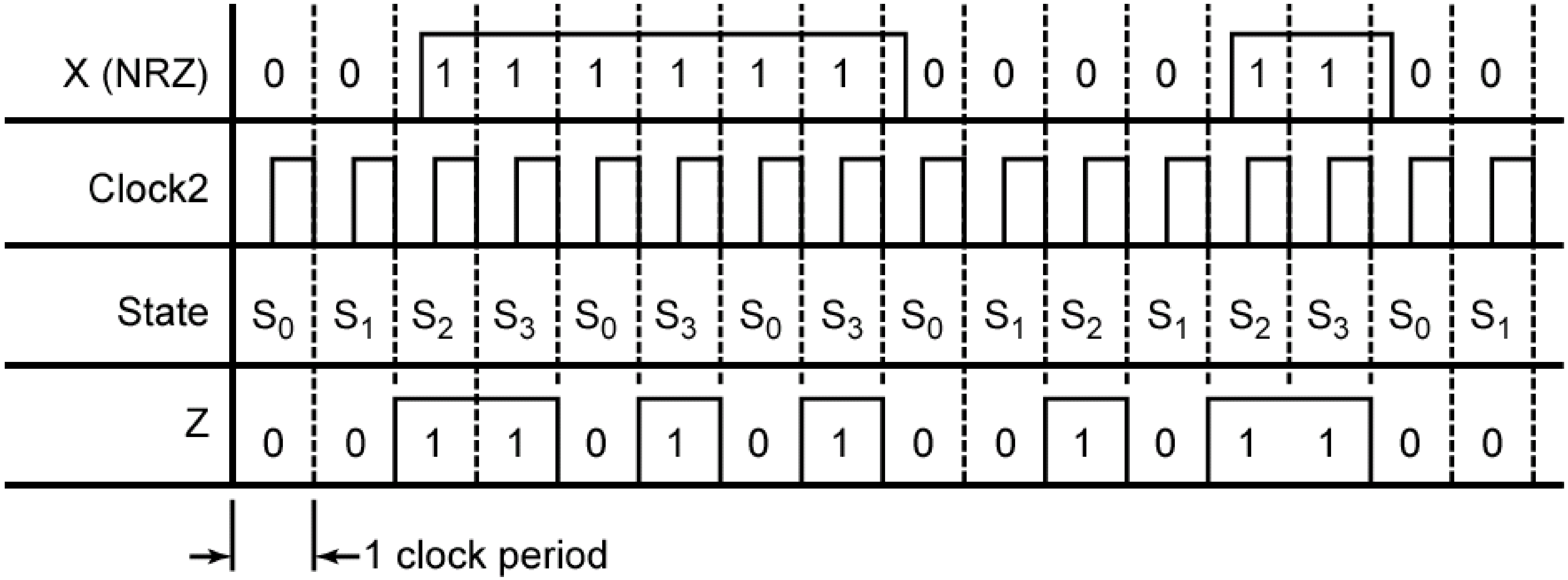
(d) State table



(c) State graph

Figure 14-20cd:

Mealy Circuit for NRZ-to-Manchester Conversion

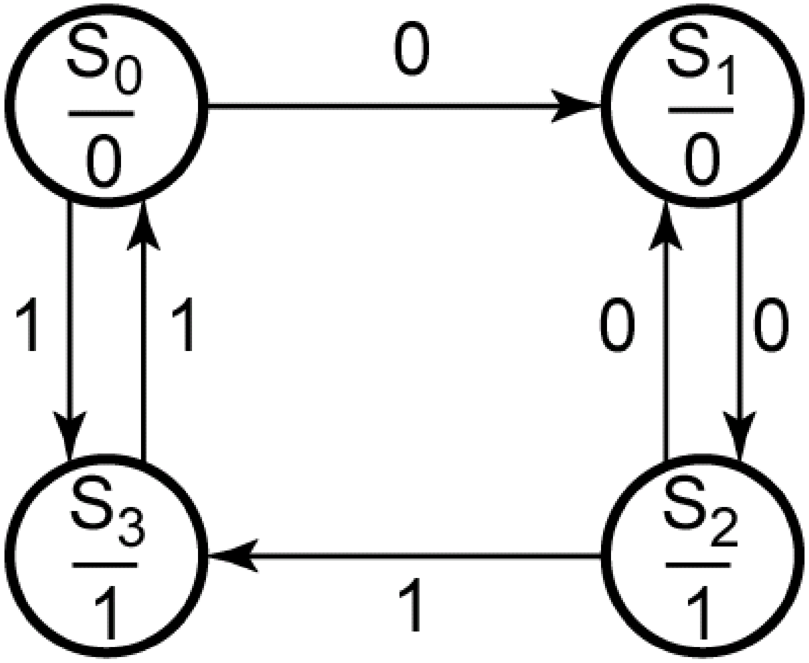


(a) Timing chart

Figure 14-21a:

Moore Circuit for NRZ-to-Manchester Conversion

Figure 14-21bc: Moore Circuit for NRZ-to-Manchester Conversion



(b) State graph

(c) State table

Present State	Next State $X = 0$	Next State $X = 1$	Present Output (Z)
S_0	S_1	S_3	0
S_1	S_2	–	0
S_2	S_1	S_3	1
S_3	–	S_0	1

Alphanumeric State Graph Notation

When a state sequential circuit has several inputs, it is often convenient to label the state graph arcs with alphanumeric input variable names instead of 0’s and 1’s.

Section 14.5 (p. 448)

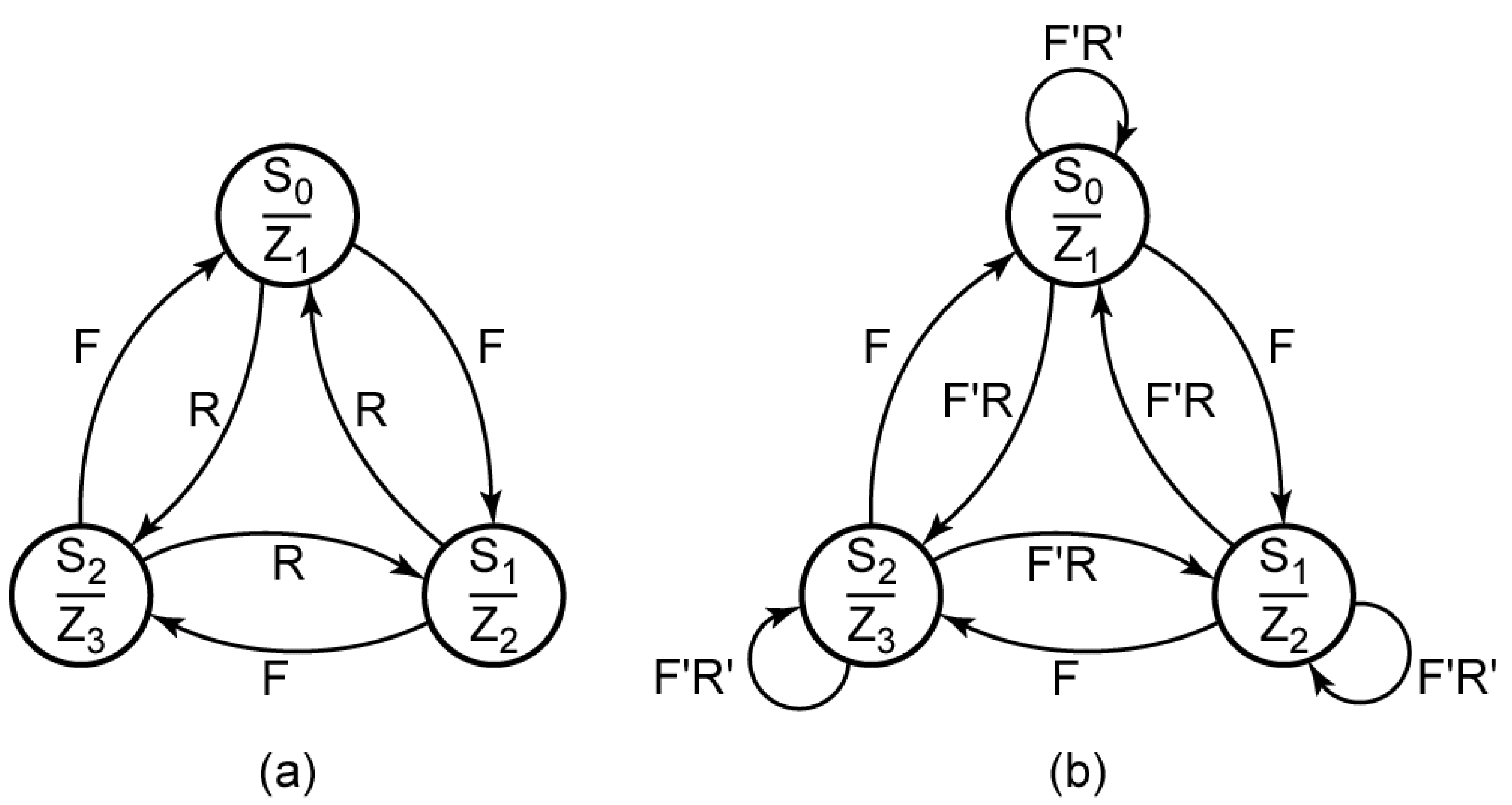


Figure 14-22: **State Graphs with Variable Names on Arc Labels**

Table 14-8. State Table for Figure 14-22

PS	NS FR=00	01	10	11	Output $Z_1Z_2Z_3$
S_0	S_0	S_2	S_1	S_1	1 0 0
S_1	S_1	S_0	S_2	S_2	0 1 0
S_2	S_2	S_1	S_0	S_0	0 0 1

Properly Specified State Graphs

In general, a completely specified state graph has the following properties:

1. When we OR together all input labels on arcs emanating from a state, the result reduces to 1.
2. When we AND together any pair of input labels on arcs emanating from a state, the result is 0.

Section 14.5 (p. 449)

Alphanumeric Notation for Mealy State Graphs

X_iX_j / Z_pZ_q means if inputs X_i and X_j are 1 (we don't care what the other input values are), the outputs Z_p and Z_q are 1 (and the other outputs are 0). That is, for a circuit with four inputs (X_1, X_2, X_3, and X_4) and four outputs (Z_1, Z_2, Z_3, and Z_4), X_1X_4' / Z_2Z_3 is equivalent to 1--0 / 0110.

This type of notation is very useful for large sequential circuits where there are many inputs and outputs.

Section 14.5 (p. 449)