8

Design of Advanced Sequential Machines

Chapter Outline

8.1 State diagram of an up/down synchronous decade counter
8.2 State diagram for sequence detectors
8.3 Design of counters

Development of state diagram from problem description and assignment of states are challenges in the design of an optimal performance sequential machine. As we have seen in the previous chapters, a state diagram is a graphical representation of a state transition table. It depicts the sequence of states and the transition conditions, and is the first step towards the design of a sequential machine. This chapter deals with the development of state diagrams for some classical cases and addresses the design of some advanced counters.

8.1 State Diagram of an Up/down Synchronous Decode Counter

Let us look at the problem of writing a state diagram of an up/down counter. Obviously, there should be a control input x which decides whether the counter is to count up or down from any

present state. Such a system would have ten states which require four state variables and one input variable to act as an up/down control input.

Let the states sequence from 0000 to 1001 in the natural binary sequence. Let $x = 0$ indicate count up and $x = 1$ indicate count down.

As in commercial up/down counters let there be two additional outputs, u to indicate terminal count while counting up and d to indicate terminal count when counting down.

The state diagram can now be drawn as shown in Fig. 8.1.

Let the input/output representation be x/ud.

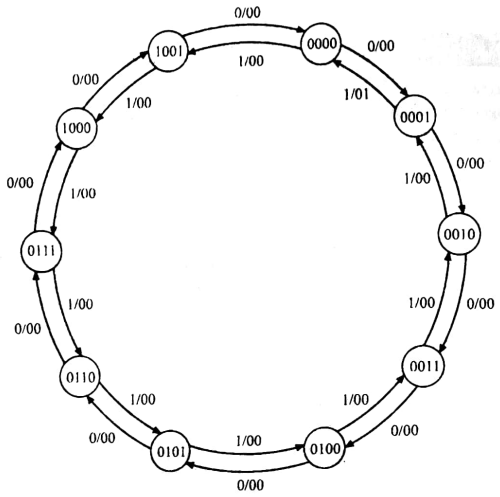


Fig. 8.1 State diagram of an up / down decade counter

Observe that for this system,

$$Q^+ = f_1(X, Q)$$

and $$Y = f_2(X, Q)$$

Hence it is a Mealy machine.

8.2 State Diagram for Sequence Detectors

Development of sequence detectors again involves the design of sequential circuits with a desired relationship between the input and output sequences. Several interesting examples are now discussed.

Draw the state diagram of a Moore machine to output a 1 if the input has been 1 for three consecutive clock cycles.

Solution: Let us first write an example which is described by the problem. Let the input be x and the output be z.

A sample input/output sequence is shown in Fig. 8.2

x	0	1	1	1	1	1	1	1	0	1	1	0	1	1	1	0	0
z	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	1	0

Fig. 8.2 Sample 1/0 sequence for Example 8.1

Let us proceed step by step.

Step 1: Start at state A. If $x = 0$, remain at A implying zero 1s and if $x = 1$, go to state B, implying the occurrence of one 1, as shown in Fig. 8.3.

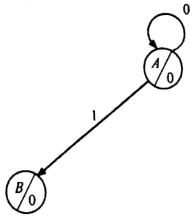


Fig. 8.3 State design after step 1

Step 2: At B, if $x = 0$, go back to state A and if $x = 1$, go to state C implying occurrence of two 1s as shown in Fig. 8.4 (a).

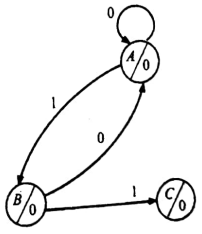


Fig. 8.4 (a) State diagram after step 2

Step 3: At C, if $x = 0$, go back to A and if $x = 1$, go to state D implying the occurrence of three 1s at which point a 1 is output, since 1s have been encountered in three consecutive clocks as shown in Fig. 8.4 (b).

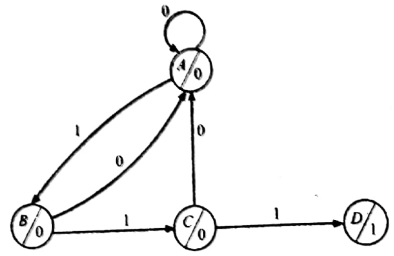


Fig. 8.4 (b) State diagram after step 3

Step 4: At D, if $x = 0$, go back to state A and if $x = 1$, remain in state D implying more consecutive 1s as shown in Fig. 8.4 (c).

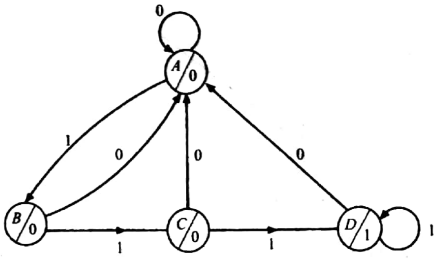


Fig. 8.4 (c) Complete state diagram for Example 8.1

Draw the state diagram of a Mealy machine whose output is 1 for every third input being 1, not necessarily consecutive but non-overlapping.

Solution: Let the input be x and the output z. A sample sequence for the problem is shown in Fig. 8.5.

x	0	1	0	0	1	1	0	0	0	1	0	0	1	1	0	0	1
z	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0

Fig. 8.5 Sample I/O sequence for Example 8.2

Step 1: Start at state A. If $x = 0$, remain at A, else go to state B implying occurrence of one 1 as shown in Fig. 8.6 (a).

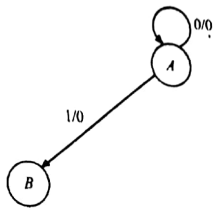


Fig. 8.6 (a) State diagram after step 1

Step 2: At state B, if $x = 0$, remain at B, else go to state C as shown in Fig. 8.6 (b).

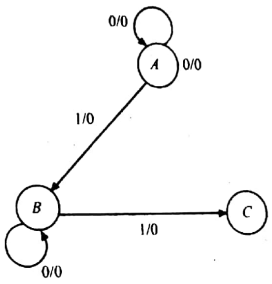


Fig. 8.6 (b) State diagram after step 2

Step 3: At state C, if $x = 0$, remain at C else go to A implying occurrence of three 1s and output one 1 as shown in Fig. 8.6 (c).

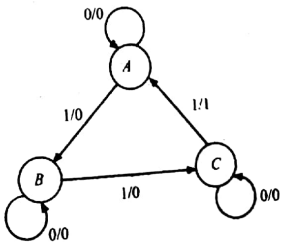


Fig. 8.6 (c) Complete state diagram for Example 8.2

[redacted]

Draw the state diagram of a Mealy machine whose output is 1 iff the input has been 1 for three consecutive clock cycles, but non-overlapping.

Solution: Let the input be x and the output be z. A sample sequence for the problem is shown in Fig. 8.7.

x	0	0	1	1	1	1	0	0	1	1	0	0	1	1	1	1	1	1	0	0
z	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0

Fig. 8.7 Sample I/O sequence for Example 8.3

Note that the input sequence of Example 8.2 will produce an output uniformly zero over the entire sequence for the problem description of Example 8.3 since three consecutive 1s are not present.

Step 1: Start at state A. if $x = 0$, remain at A, else go to B, implying occurence of one 1 as shown in Fig. 8.8 (a).

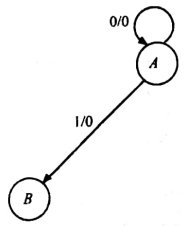


Fig. 8.8 (a) State diagram after step 1

Step 2: At state B, if $x = 0$, go back to A, else go to state C, implying the occurrence of two 1s, as shown in Fig. 8.8 (b).

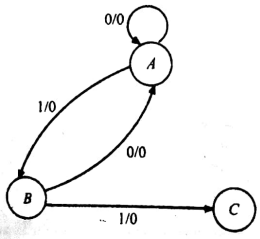


Fig. 8.8 (b) State diagram after step 2

Step 3: At state C, if $x = 0$, go back to state A with zero output else go back to state A with 1 output indicating the occurrence of three consecutive 1s as shown in Fig. 8.8 (c).

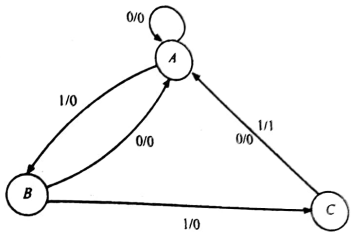


Fig. 8.8 (c) Complete state diagram for Example 8.3

[redacted]

Draw the state diagram of a Mealy machine whose output is 1 iff the last three inputs were 010 assuming that the sequence could overlap.

Solution: Assuming input to be x and output to be z, a sample input/output sequence is shown in Fig. 8.9.

x	1	1	0	1	0	1	0	1	0	0	1	0	0	1	0	1	0	1	0	0
z	0	0	0	0	1	0	1	0	1	0	0	1	0	0	1	0	1	0	1	0

Fig .8.9 Sample I/O sequence for Example 8.4

Step 1: Start at state A. If x is 0, go to state B indicating the possible start of a valid sequence, else remain at A as shown in Fig. 8.10 (a).

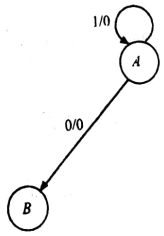


Fig. 8.10 (a) State diagram after step 1

Step 2: At state B, if $x = 0$, remain at B, else go to state C indicating the occurrence of the second valid bit in the sequence to be detected as shown in Fig. 8.10 (b).

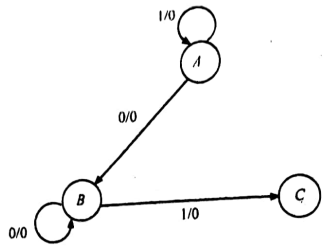


Fig. 8.10 (b) State diagram after step 2

Step 3: At state C, if $x = 0$, return to state A with output 1, indicating the detection of the complete sequence 010, else return to state A with output 0 as shown in Fig. 8.10 (c).

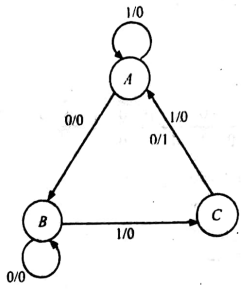


Fig. 8.10 (c) Complete state diagram for Example 8.4

Example 8.5

Draw the state diagram of a Moore machine whose output toggles whenever the 110 sequence is detected.

Solution: Let the input be x and the output z. A sample input/output sequence is shown in Fig. 8.11.

x	0	0	1	0	1	1	1	0	1	1	0	0	1	1	0	1	0	1
z	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	1	1	1

Fig. 8.11 Sample I/O sequence for Example 8.5

Step 1: Start at state A, if $x = 0$, remain at A, else go to state B as shown in Fig. 8.12 (a).

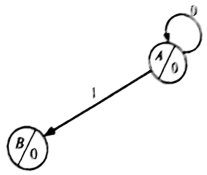


Fig. 8.12 (a) State diagram at the end of step 1.

Step 2: At state B, if $x = 0$, go back to A, else go to C indicating that a sequence 11 has occurred as shown in Fig. 8.12 (b).

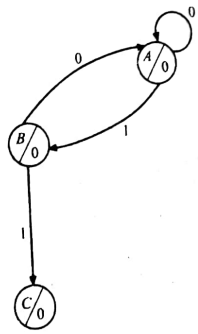


Fig. 8.12 (b) State diagram at end of step 2

Step 3: At state C, if $x = 1$, remain at C, waiting for a valid sequence (0) else go to state D, at which state 1 is output indicating the occurrence of the sequence 110 as shown in Fig. 8.12 (c).

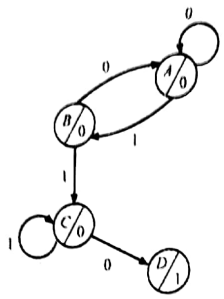


Fig. 8.12 (c) State diagram at the end of step 3

Step 4: At state D, if $x = 0$, remain at state D waiting for the next occurrence of 1, else go to state E as shown in Fig. 8.12 (d). State E generates a 1 since the output must toggle to 0 only when the next valid sequence 110 is encountered.

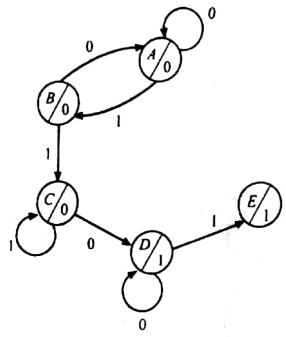


Fig. 8.12 (d) State diagram at the end of step 3

Step 5: At state E, if $x = 0$, go back to state D to start afresh only when 1 is again encountered, else go to F again with output 0 as shown in Fig. 8.12 (e).

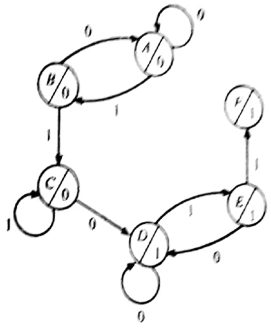


Fig. 8.12 (e) State diagram at the end of step 5.

Step 6: At state F, if $x = 1$, remain at F else go to A while a 0 is output (toggle) indicating a valid sequence 110 has once again been detected as shown in Fig. 8.12 (f).

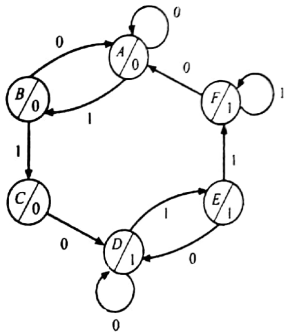


Fig. 8.12 (f) Complete state diagram for Example 8.5

Draw the state diagram of a Mealy machine to detect as input sequence 10110 which could overlap. An output 1 is to be generated when the sequence is detected.

Solution: Assuming input as x and output as z, a sample input sequence is shown in Fig. 8.13.

x	1	0	1	1	0	1	1	0	1	1	0
z	0	0	0	0	1	0	0	1	0	0	1

Fig. 8.13 Sample sequence for Example 8.6

Step 1: Start at A. If $x = 0$, remain at A else go to B, as shown in Fig. 8.14 (a).

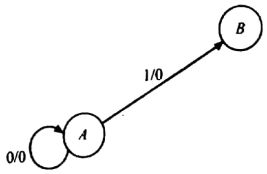


Fig. 8.14 (a) State diagram after step 1

Step 2: At state B if $x = 1$, remain at B else go to C(1-0-) as shown in Fig. 8.14 (b).

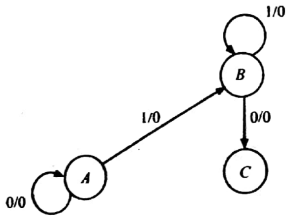


Fig. 8.14 (b) State diagram after step 2

Step 3: At state C, if x is 0, go back to state A as the sequence has failed, else go to state D (1-0-1-) as shown in Fig. 8.14 (c).

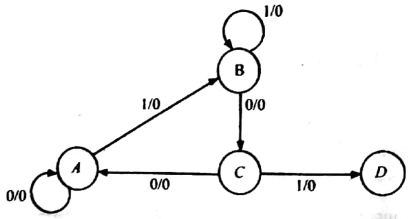


Fig. 8.14 (c) State diagram after step 3

Step 4: At state D if x is 0, go back to C, else go to E(1-0-1-1-) as shown in Fig. 8.14 (d).

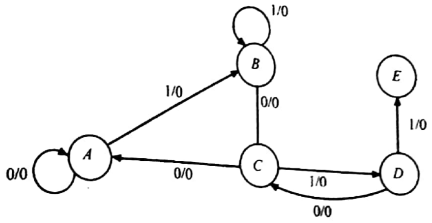


Fig. 8.14 (d) State diagram after step 4

Step 5: At state E if x is 1 go to state B, else go to C (1-0-1-1-0) with output at 1 as shown in Fig. 8.14 (e).

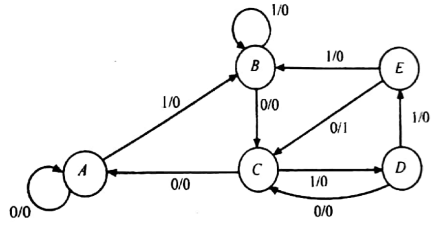


Fig. 8.14 (e) Complete State diagram for Example 8.6

Draw the state diagram for an electronic combination lock Mealy machine which will open when the following 4-bit input combinations are punched in the sequence 1010, 0110, 0011, 1001.

Solution: Let the input combination be $PQRS$ and let the output be Z, which must be set to 1 for the lock to open.

Step:1 Let the machine be in some initial state A. If the combinations punched is 1010, it goes to state B, else it goes to state E as shown in Fig. 8.15 (a).

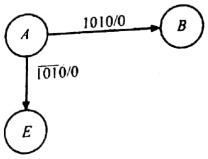


Fig. 8.15 (a) State diagram after step 1

Step 2:

(i) At state B if the input code is 0110 it goes to state C else it goes to state F.

(ii) At state E any state called 'ϕ' will take the machine to state F.

These transitions are shown in Fig. 8.15 (b).

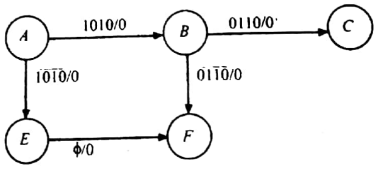


Fig. 8.15 (b) State diagram after step 2

Step 3:

(i) At state C, if the input code is 0011, it goes to state D, else to state G.

(ii) At state F, any state 'ϕ' takes the system to state G.

These transitions are shown in Fig. 8.15 (c).

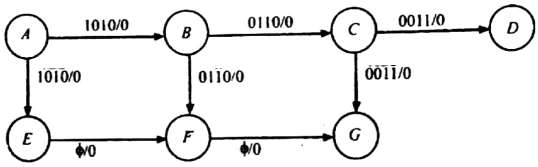


Fig. 8.15 (c) State diagram after step 3

Step 4:

(i) At state C input code 1001 takes it back to A with output 1, which any other code will take it back to A with output 0.

(ii) At state G, any state will take the system back to A with output 0. These transitions are shown in Fig. 8.15 (d).

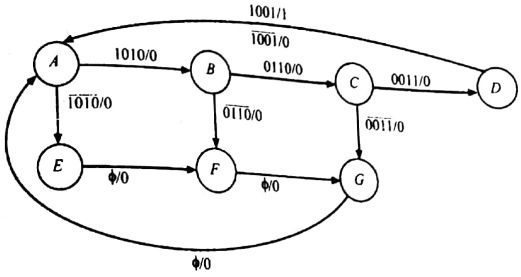


Fig. 8.15 (d) Complete state diagram of Example 8.7

Draw the state diagram of a Mealy machine to output a 1 on detection of the input sequence 010110.

Solution: Let the input be x and the output be z. The sample input and output sequence is shown in Fig. 8.16.

x	0	1	0	1	1	0	1	0	1	1	0	1	0	1	1	0	0	1
z	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0

Fig. 8.16 Sample I/O sequence for Example 8.8

Step 1: Start with state A. If $x = 0$ go to B, else remain at A, as shown in Fig. 8.17 (a).

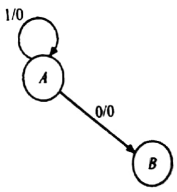


Fig. 8.17 (a) State diagram after step 1

Step 2: At B, if $x = 1$ go to C, else go back to A as shown in Fig. 8.17 (b).

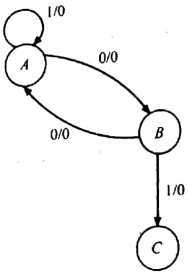


Fig. 8.17 (b) State diagram after step 2

This process repeats. If the input corresponds to the next valid bit in the sequence to be detected, the machine goes to a new next state, else it goes back to state A.

The state diagram for the complete sequence detection is shown in Fig. 8.17 (c).

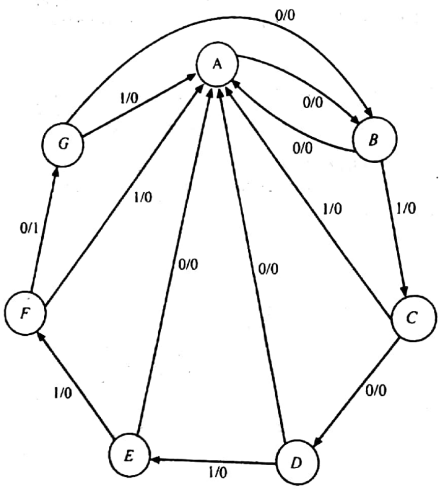


Fig. 8.17 (c) Complete state diagram for Example 8.8

Note that when at state G, $x = 0$ will take it to state B else the machine is back at state A.

♦ **Exercise 8.1**

Draw the state diagram of a Mealy machine to detect the sequence 1101 assuming
(a) over lapping is allowed
(b) over lapping is not allowed

♦ **Exercise 8.2**

Draw the state diagram of a Moore machine that produces a 1 output iff there have been four or more consecutive 1 inputs or two or more consecutive 0 inputs.

♦ **Exercise 8.3**

Draw the state diagram of a Mealy machine that produces a 1 output if the sequence 0011 is detected at the input.

♦ **Exercise 8.4**

Draw the state diagram of a Mealy machine that produces a 1 output if the sequence 010 or 101 appears at the input with overlapping being allowed.

♦ **Exercise 8.5**

Draw the state diagram of a Mealy machine that produces a 1 output if a 00 or 11 sequence appears at the input assuming that overlapping is (a) allowed (b) not allowed.

♦ **Exercise 8.6**

Draw the state diagram of a Moore machine that produces a 1 output if a sequence 10101 is detected.

♦ **Exercise 8.7**

Draw the state diagrams of a Moore machine to output a 1 if two or more consecutive 1s or three or more consecutive 0s are detected at the input.

8.3 Design of Counters

We now look at the design procedures for a type of synchronous sequential machine referred to as counters. Most counters do not have any inputs but go through a fixed sequence of states on successive clock pulses. However, there are counters with inputs for count enable or up/down control and so on. Such inputs can be incorporated in the design of the sequential machine.

Let us look at the example of the design of a synchronous eight bit counter using JK flip flops to count the number of occurrence of an input.

[redacted]

Design a synchronous eight bit binary counter using JK flip flops to count the number instances that an input coincident with the clock is high.

Solution: A three bit binary counter will have 2^3 or 8 states. Let X be the input which is to be counted whenever its value is 1 at the instances of the clock.

The state diagram of the system is shown in Fig. 8.18.

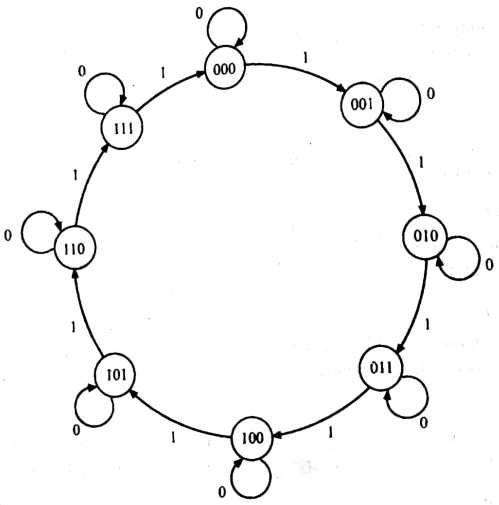


Fig. 8.18 State diagram of 8 bit counter of Example 8.9

The state transition table is shown in Fig. 8.19.

Cell no.	A	B	C	X	A^+	B^+	C^+
0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	1
2	0	0	1	0	0	0	1
3	0	0	1	1	0	1	0
4	0	1	0	0	0	1	0
5	0	1	0	1	0	1	1
6	0	1	1	0	0	1	1
7	0	1	1	1	1	0	0
8	1	0	0	0	1	0	0
9	1	0	0	1	1	0	1
10	1	0	1	0	1	0	1
11	1	0	1	1	1	1	0
12	1	1	0	0	1	1	0
13	1	1	0	1	1	1	1
14	1	1	1	0	1	1	1
15	1	1	1	1	0	0	0

Fig. 8.19 State transition table for Example 8.9

The excitation table for *JK* flip flop is as follows:

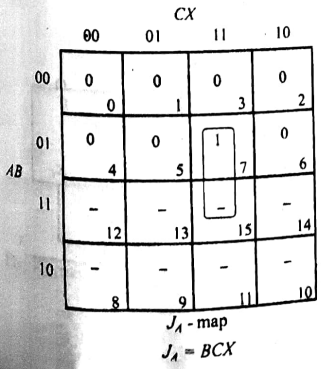
$Q \rightarrow Q^+$	J	K
0 0	0	–
0 1	1	–
1 0	–	1
1 1	–	0

This is used to obtain the excitation table for Example 8.9 as shown in Fig. 8.20.

Cell no.	A	B	C	X	A^+	B^+	C^+	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	0	0	0	0	–	0	–	0	–
1	0	0	0	1	0	0	1	0	–	0	–	1	–
2	0	0	1	0	0	0	1	0	–	0	–	–	0
3	0	0	1	1	0	1	0	0	–	1	–	–	1
4	0	1	0	0	0	1	0	0	–	–	0	0	–
5	0	1	0	1	0	1	1	0	–	–	0	1	–
6	0	1	1	0	0	1	1	0	–	–	0	–	0
7	0	1	1	1	1	0	0	1	–	–	1	–	1
8	1	0	0	0	1	0	0	–	0	0	–	0	–
9	1	0	0	1	1	0	1	–	0	0	–	1	–
10	1	0	1	0	1	0	1	–	0	0	–	–	0
11	1	0	1	1	1	1	0	–	0	1	–	–	1
12	1	1	0	0	1	1	0	–	0	–	0	0	–
13	1	1	0	1	1	1	1	–	0	–	0	1	–
14	1	1	1	0	1	1	1	–	0	–	0	–	0
15	1	1	1	1	0	0	0	–	1	–	1	–	1

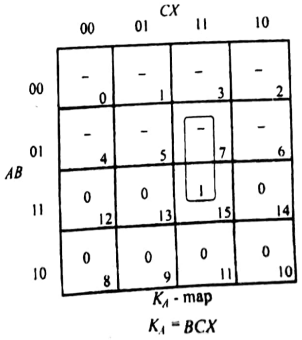
Fig. 8.20 Excitation Table for the flip flop inputs

The Karnaugh maps are now generated for each flip flop input.



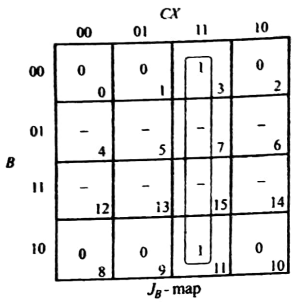
J_A - map

$J_A = BCX$



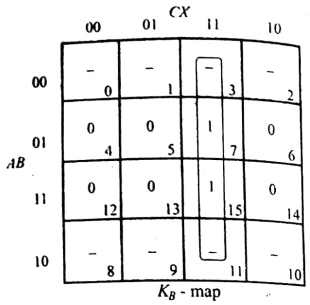
K_A - map

$K_A = BCX$



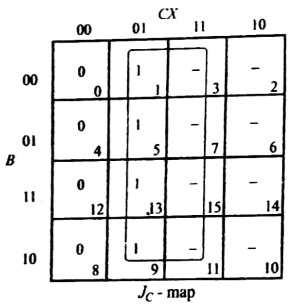
J_B - map

$J_B = CX$



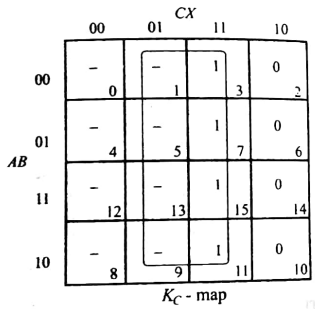
K_B - map

$K_B = CX$



J_C - map

$J_C = X$



K_C - map

$K_C = X$

The circuit is shown in Fig. 8.21.

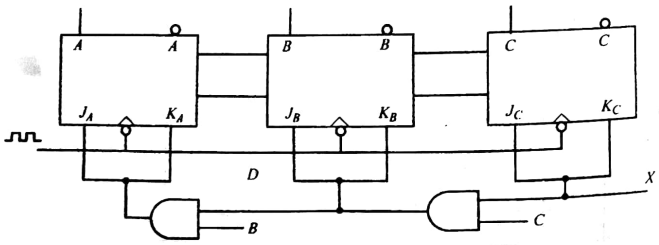
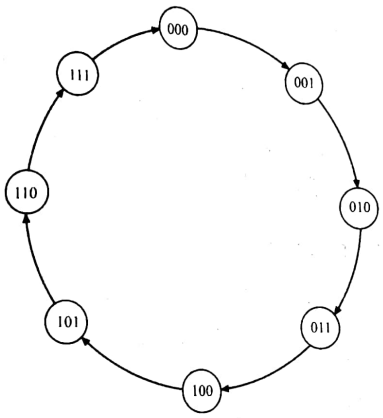


Fig. 8.21 Circuit for Example 8.9

Design a synchronous modulo-8 binary counter using *D*-flip flops starting from the state diagram.

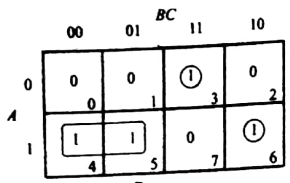
Solutions: **The state diagram of a synchronous modulo-8 binary counter is shown below.**



The excitation table for implementation using *D* flip flops is written:

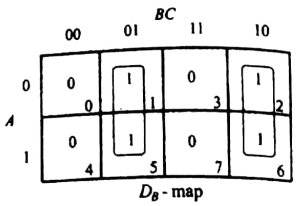
Row no.	*A*	*B*	*C*	A^+	B^+	C^+	D_A	D_B	D_C
0	0	0	0	0	0	1	0	0	1
1	0	0	1	0	1	0	0	1	0
2	0	1	0	0	1	1	0	1	1
3	0	1	1	1	0	0	1	0	0
4	1	0	0	1	0	1	1	0	1
5	1	0	1	1	1	0	1	1	0
6	1	1	0	1	1	1	1	1	1
7	1	1	1	0	0	0	0	0	0

The Karnaugh maps for the *D*-flip flop inputs is now drawn.



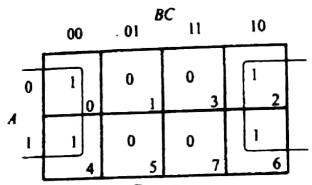
D_A - map

$D_A = \bar{A}BC + AB\bar{C} + A\bar{B}$



D_B - map

$D_B = \bar{B}C + B\bar{C}$

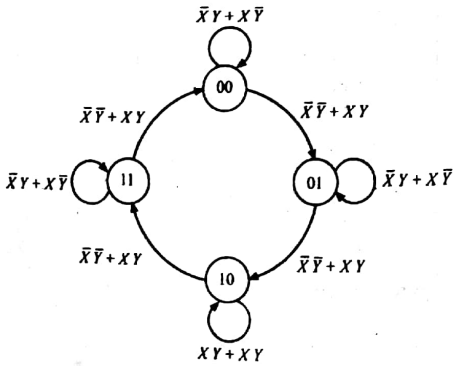


D_C - map

$D_C = \bar{C}$

Design a modulo-4 counter using *JK*- flip flops which will increment when two external inputs are coincident.

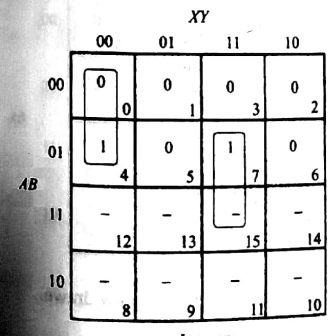
Solution: Let the external inputs be x and y, the counter increments when $x = y$ and remains in the present state other wise. The state diagram is shown below.



The state transition and excitation table for implementation using *JK*-flip flops is shown below.

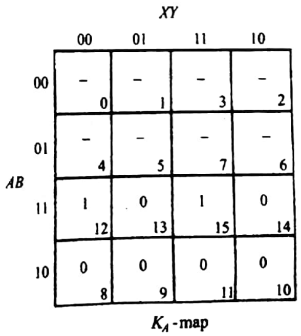
Cell no.	A	B	X	Y	A^+	B^+	J_A	K_A	J_B	K_B
0	0	0	0	0	0	1	0	–	1	–
1	0	0	0	1	0	0	0	–	0	–
2	0	0	1	0	0	0	0	–	0	–
3	0	0	1	1	0	1	0	–	1	–
4	0	1	0	0	1	0	1	–	–	1
5	0	1	0	1	0	1	0	–	–	0
6	0	1	1	0	0	1	0	–	–	0
7	0	1	1	1	1	0	1	–	–	1
8	1	0	0	0	1	1	–	0	1	–
9	1	0	0	1	1	0	–	0	0	–
10	1	0	1	0	1	0	–	0	0	–
11	1	0	1	1	1	1	–	0	1	–
12	1	1	0	0	0	0	–	1	–	1
13	1	1	0	1	1	1	–	0	–	0
14	1	1	1	0	1	1	–	0	–	0
15	1	1	1	1	0	0	–	1	–	1

The Karnaugh maps for the flip flop inputs are now written.



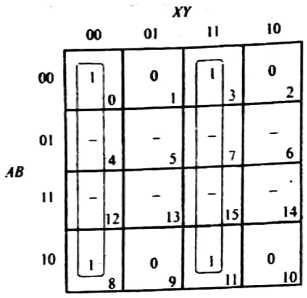
J_A - map

$J_A = B\bar{X}\bar{Y} + BXY$



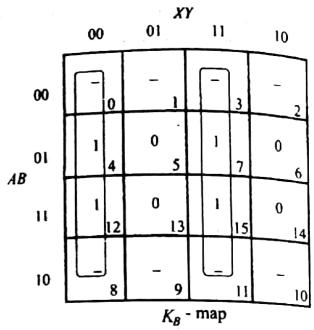
K_A - map

$K_A = B\bar{Y}$



J_B - map

$J_B = \bar{X}\bar{Y} + XY$

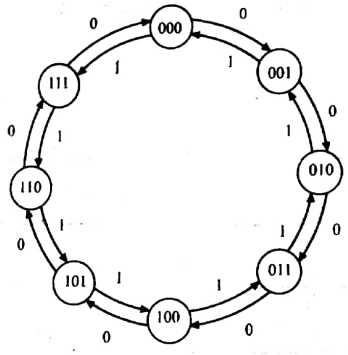


K_B - map

$K_B = \bar{X}\bar{Y} + XY$

Design a synchronous up/down modulo-8 counter using *SR*-flip flops.

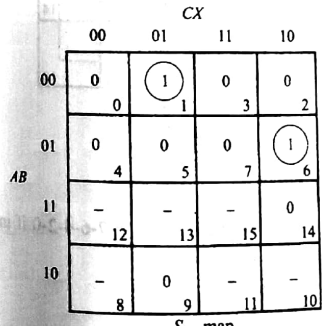
Solution: Let us define an input X such that $X = 0$ counts up and $X = 1$ counts down.
The state diagram is as follows:



The state transition table and the excitation table using *SR*-flip flops is now drawn.

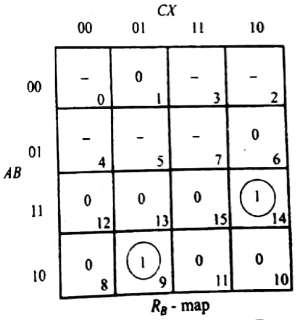
Cell no.	A	B	C	X	A^+	B^+	C^+	S_A	R_A	S_B	R_B	S_C	R_C
0	0	0	0	0	0	0	1	0	–	0	–	1	0
1	0	0	0	1	1	1	1	1	0	1	0	1	0
2	0	0	1	0	0	1	0	0	–	1	0	0	1
3	0	0	1	1	0	0	0	0	–	0	–	0	1
4	0	1	0	0	0	1	1	0	–	–	0	1	0
5	0	1	0	1	0	0	1	0	–	0	1	1	0
6	0	1	1	0	1	0	0	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0	–	–	0	0	1
8	1	0	0	0	1	0	1	–	0	0	–	1	0
9	1	0	0	1	0	1	1	0	1	1	0	1	0
10	1	0	1	0	1	1	0	–	0	1	0	0	1
11	1	0	1	1	1	0	0	–	0	0	–	0	1
12	1	1	0	0	1	1	1	–	0	–	0	1	0
13	1	1	0	1	1	0	1	–	0	0	1	1	0
14	1	1	1	0	0	0	0	0	1	0	1	0	1
15	1	1	1	1	1	1	0	–	0	–	0	0	1

The Karnaugh maps for the *SR* – flip flop inputs using table in Fig 6.42 can now be written.



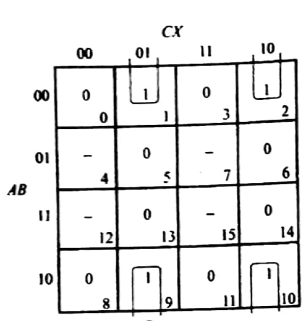
S_A - map

$S_A = \bar{A}\bar{B}\bar{C}X + \bar{A}\bar{B}C\bar{X}$



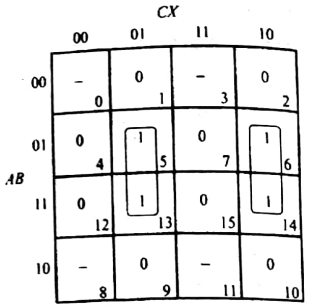
R_B - map

$R_B = A\bar{B}\bar{C}X + ABC\bar{X}$



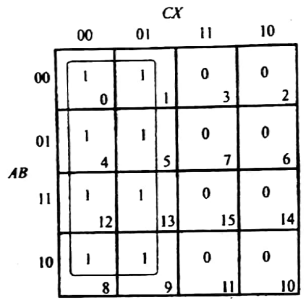
S_B - map

$S_B = \bar{B}\,\bar{C}\,X + \bar{B}\,C\,\bar{X}$



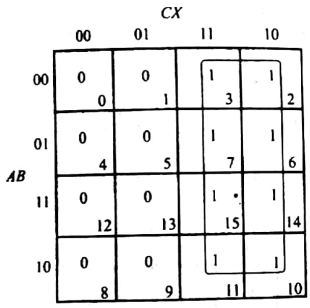
R_B - map

$S_B = B\,\bar{C}\,X + B\,C\,\bar{X}$



S_C - map

$S_C = \bar{C}$



R_C - map

$R_C = C$

◆ Exercise 8.8

Design a counter using *JK*-flip flops that goes through the sequence 0-1-3-5-7-6-4-2-0 if input $x = 1$ and restarts from 5 when input $x = 0$.

◆ Exercise 8.9

Design a decade up/down counter using *D*-flip flops with two inputs x and y, where x is assigned for direction of count and y for count enable.

◆ Exercise 8.10

Design a modulo-12 up/down counter using *JK*-flip flops.

◆ Exercise 8.11

Design a modulo-8 counter using *JK*-flip flops with two control inputs x, y such that

(i) When $x\,y = 00$, count up even sequence
(ii) When $x\,y = 01$, count down even sequence
(iii) When $x\,y = 10$, count up odd sequence
(iv) When $x\,y = 11$, count down even sequence

◆ Exercise 8.12

Design a modulo-8 counter using *SR*-flip flop with the counter inputs X and Y to perform the following function:

X	Y	*Function*
0	0	reset to 000
0	1	count up
1	0	count down
1	1	hold