Present state c

When X is 0, the next state is d and Y is 0, because the third bit of the sequence is correct. When X is 1, the third bit is not correctly detected, but there is no need to go to the initial state a. We assume that the first bit is wrong and the third bit is correct. ~~101~~ is considered as 1 and hence, the next state is b and Y is 0.

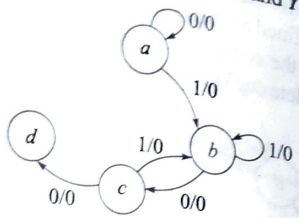


Fig. 5.158

Present state d

When X is 1, the next state is e and Y is 0, because the fourth bit of the sequence is correctly detected. When X is 0, the fourth bit is wrong and the next state must be a and Y is 0, because the whole sequence is wrong.

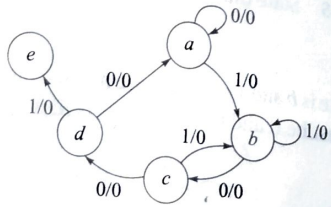


Fig. 5.159

Present state e

When X is 0, the next state is f and Y is 0, because the fifth bit is correctly detected. When X is 1, the next state is b. The sequence ~~11001~~ will be treated as 1 and Y is 0.

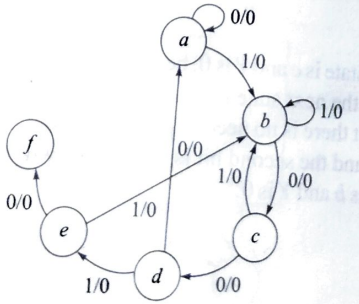


Fig. 5.160

Present state f

When X is 1, the next state is g and Y is 1, because the sixth bit of the sequence is correctly detected. When X is 0, the sixth bit of the sequence is wrong. The

next state must be d and Y is 0. The sequence 001~~001~~ will be treated as 001 and Y is 0.

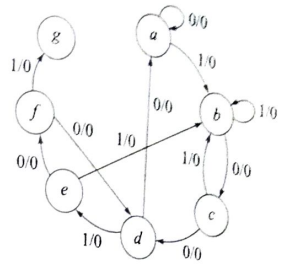


Fig. 5.161

Present state g

When X is 1, the next state is a and Y is 1, because the last bit of the sequence is correctly detected. When X is 0, the last bit is wrong and the next state must be c and Y is 0. The sequence 010~~1001~~ will be treated as 01 and Y is 0.

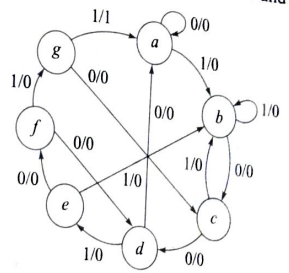


Fig. 5.162

State table The state table for the state diagram of Fig. 5.158 is shown in Table 5.99.

Table 5.99 State table for state diagram (Fig. 5.162)

Present state	Next state		Output	
	X=0	X=1	X=0	X=1
a	a	b	0	0
b	c	b	0	0
c	d	b	0	0
d	a	e	0	0
e	f	b	0	0
f	d	g	0	0
g	c	a	0	1

State reduction From the state table, it is observed that all the states are different and there is no possibility to reduce the state table.

State assignment The state table shown in Table 5.99 includes seven alphabets and hence, the number of bits required to represent the alphabets in binary is three. The assigned values for the alphabets using 3-bits are given in the following table.

Variable	Assigned value
a	000
b	001
c	010
d	011
e	100
f	101
g	111

The state table which shows the present state and the next state for $X = 0$ and $X = 1$, and the output for $X = 0$ and $X = 1$ is given in Table 5.100.

Table 5.100 State table for state diagram (Fig. 5.159) in binary

Present state			Next state						Output	
			$X=0$			$X=1$			$X=0$	$X=1$
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0	Q_2	Q_1	Q_0	Y	Y
0	0	0	0	0	0	0	0	1	0	0
0	0	1	0	1	0	0	0	1	0	0
0	1	0	0	1	1	0	0	1	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	1	0	1	0	0	1	0	0
1	0	1	0	1	1	1	1	1	0	0
1	1	0	0	1	0	0	0	0	0	1
1	1	1	0	0	0	0	0	0	0	0

Excitation table The excitation table for the given state diagram using D flip-flop is given in Table 5.101.

Table 5.101 Excitation table for state table (Table 5.100) using D flip-flop

Present state			Input	Next state			Output	Input of the flip-flops		
Q_2	Q_1	Q_0	X	Q_2	Q_1	Q_0	Y	D_0	D_1	D_2
0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0	1	0	0
0	0	1	0	0	1	0	0	0	1	0
0	0	1	1	0	1	1	0	1	1	0
0	1	0	0	0	0	0	0	0	0	0
0	1	0	1	0	0	1	0	1	0	0
0	1	1	0	0	1	0	0	0	1	0
0	1	1	1	0	1	1	0	1	1	0
1	0	0	0	1	0	0	0	0	0	0
1	0	0	1	0	0	1	0	1	0	0
1	0	1	0	1	0	0	0	0	0	0
1	0	1	1	1	0	1	0	1	0	0
1	1	0	0	0	0	0	0	0	0	0
1	1	0	1	0	0	1	0	1	0	0
1	1	1	0	0	0	0	0	0	0	0
1	1	1	1	0	0	1	0	1	0	0

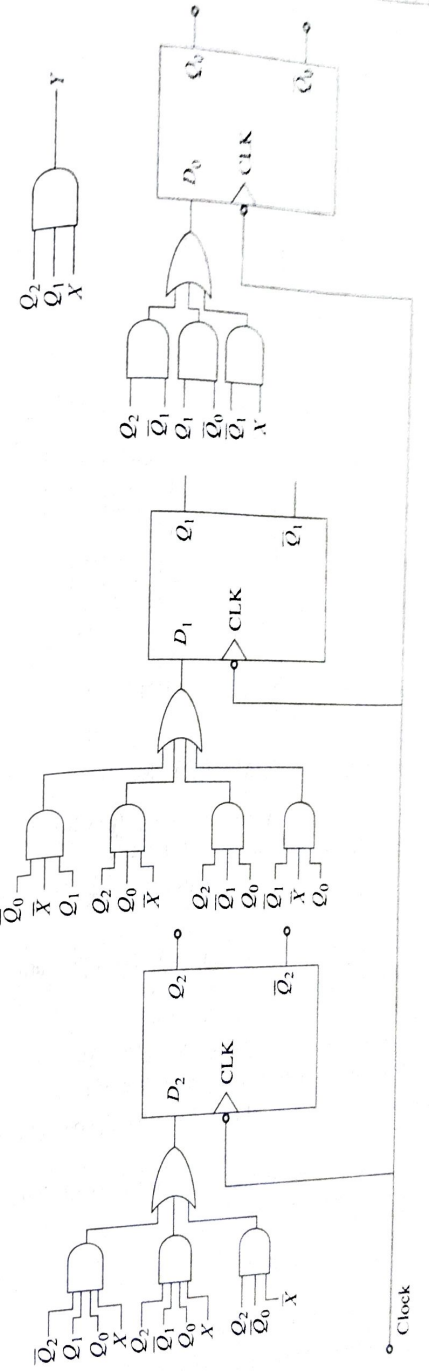


Fig. 5.165 Logic diagram for the sequence (1101001) detector

K-map for D_0

Q_0X \ Q_2Q_1	00	01	11	10
00	0	1	X	1
01	1	1	X	1
11	1	0	0	1
10	0	0	0	1

$$D_0 = Q_2\overline{Q}_1 + Q_1\overline{Q}_0 + \overline{Q}_1X$$

K-map for D_1

Q_0X \ Q_2Q_1	00	01	11	10
00	0	1	X	0
01	0	0	X	0
11	0	0	0	1
10	1	0	1	1

$$D_1 = \overline{Q}_0\overline{X}Q_1 + Q_2Q_0\overline{X} + Q_2\overline{Q}_1Q_0 + \overline{Q}_1\overline{X}Q_0$$

K-map for D_2

Q_0X \ Q_2Q_1	00	01	11	10
00	0	0	X	1
01	0	0	X	0
11	0	1	0	1
10	0	0	0	0

$$D_2 = \overline{Q}_2Q_1Q_0X + Q_2\overline{Q}_1Q_0X + Q_2\overline{Q}_0\overline{X}$$

K-map for Y

Q_0X \ Q_2Q_1	00	01	11	10
00	0	0	X	0
01	0	0	X	0
11	0	0	1	0
10	0	0	0	0

$$Y = Q_2Q_1X$$

The logic diagram for the given state table using D flip-flop is shown in Fig. 5.163.

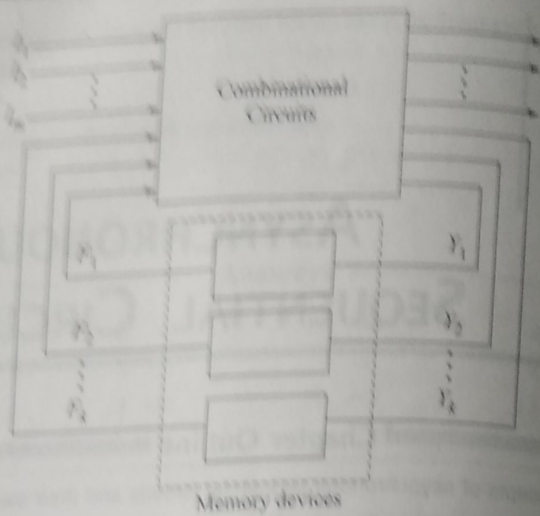


Fig. 6.1 Block diagram of asynchronous sequential circuit

In an asynchronous sequential circuit, the circuit is said to be in stable state if the outputs of memory devices become equal to their inputs ($F_i = Y_i$ for $i = 1, 2, \ldots, k$ for a given set of inputs). If there is a change in the inputs, then the combinational circuit produces a new set of excitation variables, $Y_1, Y_2, \ldots, Y_k$, at the inputs of memory devices and the circuit enters into an unstable state. When the outputs of memory devices become equal to their inputs, the circuit is said to enter the next stable state.

6.2 FUNDAMENTAL MODE ASYNCHRONOUS SEQUENTIAL CIRCUITS

In fundamental mode asynchronous sequential circuits, the inputs and outputs are represented by their levels rather than pulses. Designing a fundamental mode asynchronous sequential circuit requires the following steps:

- Formulate what the circuit has to do, from the problem definition.
- Develop a primitive state table and specify the outputs that are associated with the stable state.
- Minimize the primitive state table by merge process. It is the process of identifying those primitive states that are related by input sequences and can be combined together into a single stable state, i.e. finding the states that can be put together in the same row of the excitation map. At the end of this merge process, the number of rows will be reduced.
- Assign state variables (i.e. secondary variables) to the rows of merged primitive state table and obtain the present state/next state and output table. The outputs assigned to the unstable states vary according to the design requirements.
- Decide the memory element to be used and obtain the excitation and output table. Obtain the simplified expression of the excitation and output function.
- Draw the schematic circuit diagram.

Let us consider the design example of fundamental mode asynchronous sequential circuits.

Example 6.1 Design an asynchronous sequential circuit with two input, I_1 and I_2, and one output, Z. Initially, both inputs are equal to zero. When I_1 or I_2 becomes 1, Z becomes 1. When the second input goes to 1, the output changes from 1 to 0. The output stays at 0 until the circuit goes back to (0, 0).

Solution

The primitive state table for the given requirement can be constructed as shown in Table 6.1.

Table 6.1 Primitive state table

Row	Next state, Output			
	$I_1I_2 = 0$	$I_1I_2 = 0\ 1$	$I_1I_2 = 1\ 11$	$I_1I_2 = 1\ 0$
1	①,0	2		3
2		②,1	4	
3			4	③,1
4			④,0	
5	1	⑤,0	4	
6	1		4	⑥,0

Initially, both the inputs are 0 (i.e. $I_1 = I_2 = 0$), and the circuit is said to be in a stable state 1, represented as ①, 0, where the circle represents the stable state and the output is 0.

When I_1 remains same at 0 and I_2 becomes 1, the circuit enters a stable state 2 and its output is 1. This is represented as ②, 1 in the second row. In the first row, an uncircled 2 is entered to indicate that a transition to ② state will occur as a result of the input change from 00 to 01. The uncircled 2 represents an unstable state.

Similarly, when I_1 becomes 1 and I_2 remains same at 0, the circuit enters a stable state 3 and its output is 1. This is represented as ③, 1 in the third row. In the first row, an uncircled 3 is entered to indicate that a transition to ③ state will occur as a result of the input change from 00 to 10. The uncircled 3 represents an unstable state.

When the circuit is in stable state ② or ③ (i.e. $I_1I_2 = 01$ or $I_1I_2 = 10$) and the other input also becomes 1 ($I_1I_2 = 11$), then the circuit is in stable state ④ with output 0. This is represented as ④, 0 in the fourth row. In second and third rows, an uncircled 4 is entered to indicate that a transition to ④ state will occur as a result of the input change from 01 to 11 or from 10 to 11. The uncircled 4 represents an unstable state. In an asynchronous circuit, both inputs cannot change simultaneously, hence stable states 5 and 6 are shown in Table 6.1.

When the circuit is in ④, and I_1 becomes 0, then the circuit enters ⑤ with output 0. When I_2 becomes 0 too, it will go to the initial state. Similarly, if the circuit is in ④, and I_2 becomes 0, the circuit enters ⑥ with output 0. From here, when I_1 becomes 0, it will go to ①.

To complete the partial primitive state table, the transitions from every stable state for each change in the input should be specified. Since two inputs are not allowed to change simultaneously, a dash is entered at such places.

A circled entry and an uncircled entry can be combined to minimize the primitive state table by merge process. From Table 6.1, it is clear that the first three rows can be combined and similarly, the last three rows can be combined. Whenever a circled entry and an uncircled entry are combined as a circled entry, as shown in Table 6.2, the outputs associated with unstable states correspond to their stable states.

Table 6.2 Minimized state table

Row number	Next state, Output			
	$I_1 I_2 = 0\,0$	$I_1 I_2 = 0\,1$	$I_1 I_2 = 1\,1$	$I_1 I_2 = 1\,0$
1	①,0	②,1	4,0	③,1
2	1 0	⑤,0	④,0	⑥,0

Since there are only two rows in the merged state table, a 1-bit variable 0 and 1 can be assigned to row 1 and row 2, respectively. The present state/next state and output table is given in Table 6.3.

Table 6.3 Present state /next state and output table

Row number	Next state, Output			
	$I_1 I_2 = 0\,0$	$I_1 I_2 = 0\,1$	$I_1 I_2 = 1\,1$	$I_1 I_2 = 1\,0$
0	0 0	0 1	1,0	0,1
1	0 0	1 0	1,0	1,0

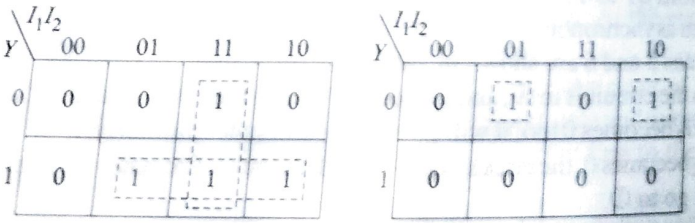
The present state/next state and output table can be obtained by replacing every circled entry in the first row by 0 and in the second row by 1. The unstable state 4 in the first row is assigned 1, since it must go to stable state 4 which belongs to second row. Similarly, the unstable state 1 in the second row is assigned 0.

If a delay flip-flop is used, then an excitation and output table is generated, as given in Table 6.4.

Table 6.4 Excitation and output table

Y	D, Z			
	$I_1 I_2 = 0\,0$	$I_1 I_2 = 0\,1$	$I_1 I_2 = 1\,1$	$I_1 I_2 = 1\,0$
0	0 0	0 1	1,0	0,1
1	0 0	1 0	1,0	1,0

The K-maps for the function (D) and output (Z) with inputs $I_1 I_2$ and Y are as shown in Fig. 6.2.



(a) K-map for D (b) K-map for Z

Fig. 6.2 K-map simplification for excitation and output functions

The simplified expressions for D and Z are:

$$D = I_1 I_2 + I_2 Y + Y I_1 \tag{6.1}$$

$$\begin{aligned} Z &= \bar{I}_1 I_2 \bar{Y} + I_1 \bar{I}_2 \bar{Y} \\ &= \bar{Y}(\bar{I}_1 I_2 + I_1 \bar{I}_2) \\ &= \bar{Y}(I_1 \oplus I_2) \end{aligned} \tag{6.2}$$

The circuit diagram for the given asynchronous sequential circuit is shown in Fig. 6.3.

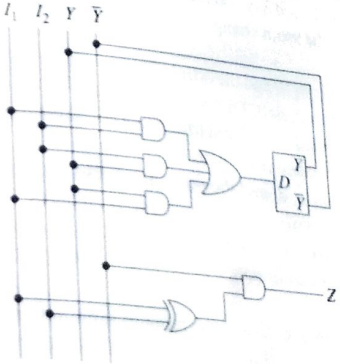


Fig. 6.3 Circuit diagram of Example 6.1

Example 6.2 Design an asynchronous sequential circuit with two inputs, I_1 and I_2 and one output, Z. Initially, both inputs are equal to 0. When I_1 changes from 0 to 1, Z becomes 1. When I_2 changes from 0 to 1, Z becomes 0. Otherwise, Z is 0. Realize the circuit using J-K flip flop.

Solution

The primitive state table for the given requirement is given in Tables 6.5.

Table 6.5 Primitive state table

Row number	Next state, Output			
	$I_1 I_2 = 0\,0$	$I_1 I_2 = 0\,1$	$I_1 I_2 = 1\,1$	$I_1 I_2 = 1\,0$
1	①,0	3		2
2	1		4	②,1
3	1	③,0	5	
4		3	④,0	6
5		3	⑤,1	6
6	1		4	⑥,0

Initially, both inputs are equal to 0 (i.e. $I_1 = I_2 = 0$), the circuit is said to be in a stable state 1 and the output is 0, represented as ①, 0.

When I_2 remains same at 0 and I_1 becomes 1, the circuit enters a stable state 2 and its output is 0. This is represented as ②, 1 in the second row. In the first row, an uncircled 2 is entered to indicate that a transition to ② state will occur as a result of the input change from 00 to 10. The uncircled 2 represents an unstable state.

Similarly, when I_1 remains same at 0 and I_2 becomes 1, the circuit enters a stable state 3 and its output is 0. This is represented as ③, 0 in the fourth row. In the first row, an uncircled 3 is entered to indicate that a transition to ③ state will occur as a result of the input change from 00 to 10. The uncircled 3 represents an unstable state.

When the circuit is in ② and the other input also becomes 1(I_1I_2=11), then the circuit enters ④ with output 0. This is represented as ④, 0 in the fourth row. In the second row, an uncircled 4 is entered to indicate that a transition to ④ will occur as a result of the input change from 10 to 11.

When the circuit is in ③ and the other input also becomes 1(I_1I_2=11), then the circuit enters ④ with output 1 because I_1 changes from 0 to 1. This is represented as ④, 1 in the fifth row. In the third row, an uncircled 5 is entered to indicate that a transition to ④ state will occur as a result of the input change from 01 to 11.

When the circuit is in ④ or ⑤ and if I_1 remains same at 1 and I_2 becomes 0, then the circuit enters a stable state 6 and its output is 0. This is represented as ⑥, 0 in the sixth row. In the fourth and fifth rows, an uncircled 6 is entered to indicate that a transition to ⑥ state will occur as a result of the input change from 11 to 10.

A circled entry and an uncircled entry can be combined to minimize the primitive state table by merge process. The outputs associated with unstable states correspond to their stable states. The minimized state table is given in Table 6.6.

Table 6.6 Minimized state table

Row	Next state, Output			
	$I_1I_2 = 00$	$I_1I_2 = 01$	$I_1I_2 = 11$	$I_1I_2 = 10$
1	①,0	3,0	4,0	②,1
2	1,0	③,0	⑤,1	6,0
3	1,0	3,0	④,0	⑥,0

Since there are three rows in the merged state table, a 2-bit variable can be used; 00 assigned to row 1, 01 assigned to row 2, and 10 assigned to row 3. The present state/next state and output table is given in Table 6.7. The unstable states are assigned 00/01/10 according to their stable state.

Table. 6.7 Present state/next state and output table

Y_1 Y_2	Next state, Output			
	$I_1I_2 = 00$	$I_1I_2 = 01$	$I_1I_2 = 11$	$I_1I_2 = 10$
0 0	00, 0	01, 0	10, 0	00, 1
0 1	00, 0	01, 0	01, 1	10, 0
1 0	00, 0	01, 0	10, 0	10, 0

If JK flip-flop is used, then an excitation and output table is required, as given in Table 6.8.

Table. 6.8 Excitation and output table

Y_1 Y_2	J_1K_1, J_2K_2, Z			
	$I_1I_2 = 00$	$I_1I_2 = 01$	$I_1I_2 = 11$	$I_1I_2 = 10$
0 0	0×, 0×, 0	0×, 1×, 0	1×, 0×, 0	0×, 0×, 1
0 1	0×, ×1, 0	0×, ×0, 0	0×, ×0, 1	1×, ×1, 0
1 0	×1, 0×, 0	×1, 1×, 0	×0, 0×, 0	×0, 0×, 0

The excitation function (J_1K_1), (J_2K_2), and Z can be simplified using K-maps, as shown in Fig. 6.4.

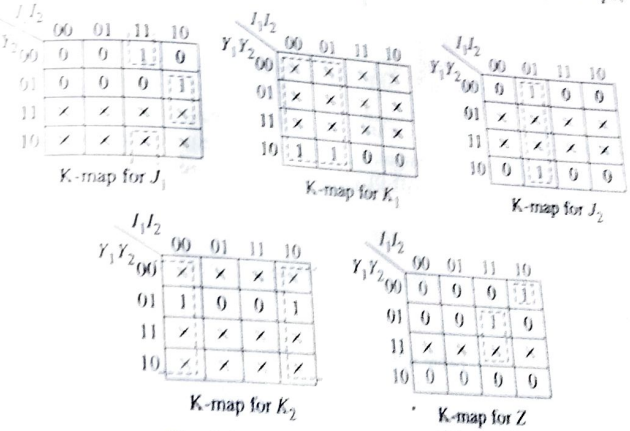


Fig. 6.4 K-map for J_1, K_1, J_2, K_2, and Z

$$J_1 = I_1I_2\bar{Y}_2 + I_1\bar{I}_2Y_2 \tag{6.3}$$

$$K_1 = \bar{I}_1 \tag{6.4}$$

$$J_2 = \bar{I}_1I_2 \tag{6.5}$$

$$K_2 = I_2 \tag{6.6}$$

$$Z = I_1I_2\bar{Y}_1\bar{Y}_2 + I_1I_2Y_2 \tag{6.7}$$

Using the above equations, the circuit diagram for the given asynchronous circuit can be drawn, as shown in Fig. 6.5.

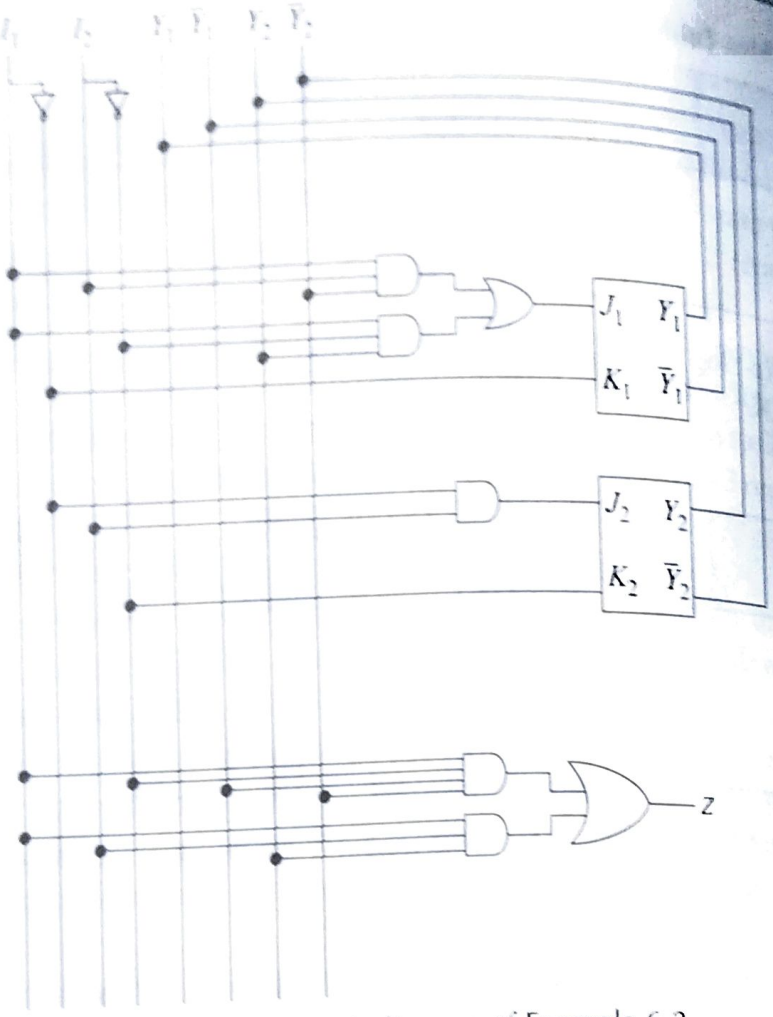


Fig 6.5 Circuit diagram of Example 6.2

Example 6.3 Design an asynchronous circuit using J-K flip-flop that has two inputs, I_1 and I_2, and one output, Z. The circuit is required to give an output whenever the input sequence (0, 0), (1, 0), and (1, 1) is received but only in that order.

Solution

The primitive state table for the given design requirement can be constructed as shown in Table 6.9.

Table 6.9 Primitive state table

Row	*Next state, Output*			
	$I_1 I_2 = 0\ 0$	$I_1 I_2 = 0\ 1$	$I_1 I_2 = 1\ 1$	$I_1 I_2 = 1\ 0$
1	①, 0	3		2
2	1		4	②, 0
3	1	③, 0	5	
4		3	④, 1	6
5		3	⑤, 0	6
6	1		5	⑥, 0

minimized by merging the

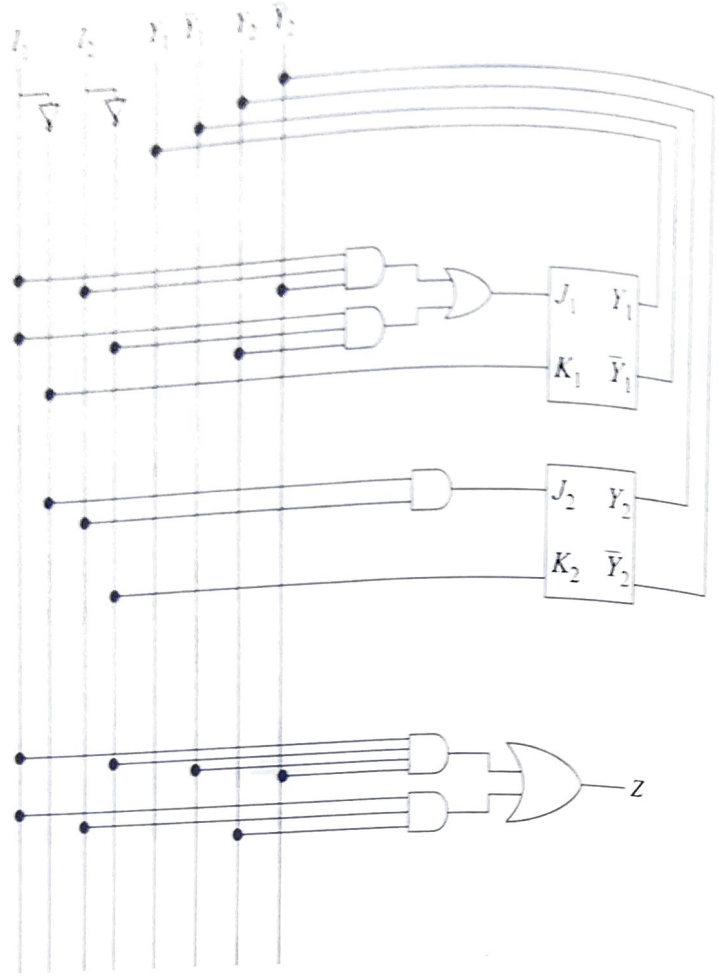


Fig 6.5 Circuit diagram of Example 6.2

Example 6.3 Design an asynchronous circuit using J-K flip-flop that has two inputs, I_1 and I_2, and one output, Z. The circuit is required to give an output whenever the input sequence (0, 0), (1, 0), and (1, 1) is received but only in that order.

Solution

The primitive state table for the given design requirement can be constructed as shown in Table 6.9.

Table 6.9 Primitive state table

Row	*Next state, Output*			
	$I_1 I_2 = 0\ 0$	$I_1 I_2 = 0\ 1$	$I_1 I_2 = 1\ 1$	$I_1 I_2 = 1\ 0$
①	①, 0	3, 0	4, 1	2, 0
②	1, 0	③, 0	⑤, 0	⑥, 0
③		3, 0	④, 1	6, 0
④				
⑤				
⑥				

Table 6.10 Minimized state table

Next state, Output			
$I_1 I_2 = 0\ 0$	$I_1 I_2 = 0\ 1$	$I_1 I_2 = 1\ 1$	$I_1 I_2 = 1\ 0$
①, 0	3, 0	4, 1	2, 0
1, 0	③, 0	②, 0	③, 0
	3, 0	③, 1	6, 0

Since there are three rows in the merged state table, a 2-bit variable can be used. (0 0) assigned to row 1, (0 1) assigned to row 2, and (1 0) assigned to row 3, and the present state/next state and output table is given in Table 6.11. The unstable state is assigned 00/01/10 according to their stable state.

Table. 6.11 Present state/next state and output table

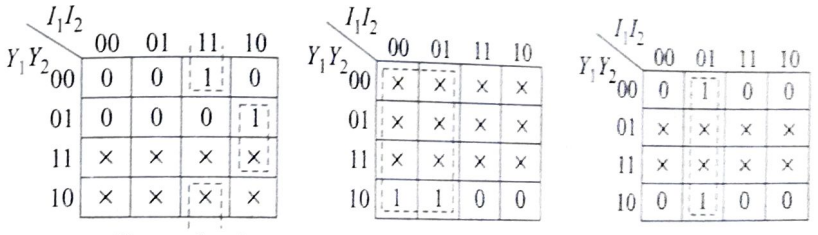
$Y_1\ Y_2$	$Y_1\ Y_2, Z$			
	$I_1 I_2 = 0\ 0$	$I_1 I_2 = 0\ 1$	$I_1 I_2 = 1\ 1$	$I_1 I_2 = 1\ 0$
0 0	00, 0	01, 0	10, 1	00, 0
0 1	00, 0	01, 0	01, 0	10, 0
1 0	00, 0	01, 0	10, 1	10, 0

If J-K flip-flop is used, then an excitation and output table is generated, as shown in Table 6.12.

Table. 6.12 Excitation and output table

$Y_1\ Y_2$	$J_1 K_1, J_2 K_2, Z$			
	$I_1 I_2 = 0\ 0$	$I_1 I_2 = 0\ 1$	$I_1 I_2 = 1\ 1$	$I_1 I_2 = 1\ 0$
0 0	0×, 0×, 0	0×, 1×, 0	1×, 0×, 0	0×, 0×, 1
0 1	0×, ×1, 0	0×, 0×, 0	0×, ×0, 1	1×, ×1, 0
1 0	×1, 0×, 0	×1, 1×, 0	×0, 0×, 0	×0, 0×, 0

The excitation function J_1K_1, J_2K_2, and Z are simplified using K-maps, as shown in Fig. 6.6.



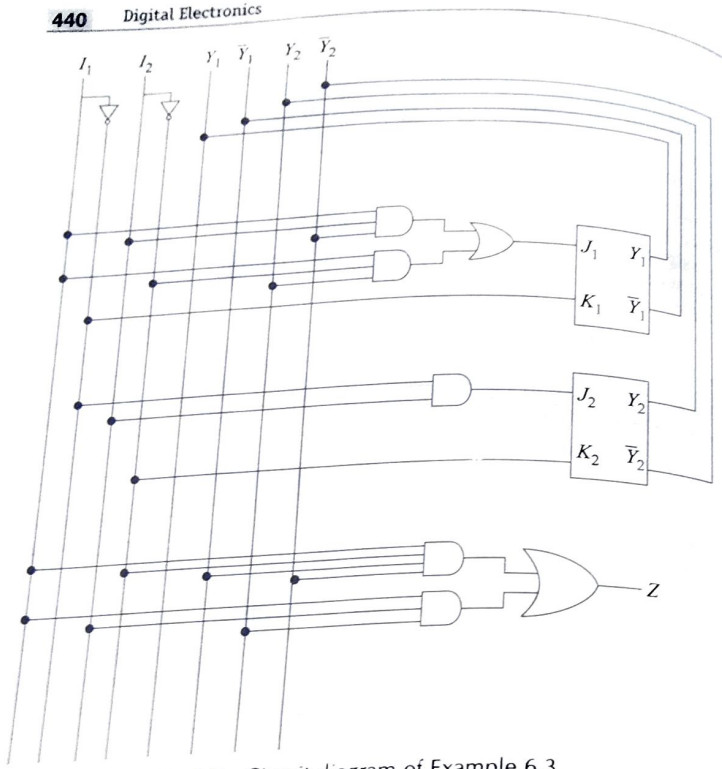


Fig. 6.7 Circuit diagram of Example 6.3

$$J_1 = I_1 I_2 \bar{Y}_2 + I_1 \bar{I}_2 Y_2 \tag{6.8}$$

$$K_1 = \bar{I}_1 \tag{6.9}$$

$$J_2 = \bar{I}_1 I_2 \tag{6.10}$$

$$K_2 = \bar{I}_2 \tag{6.11}$$

$$Z = I_1 \bar{I}_2 \bar{Y}_1 \bar{Y}_2 + I_1 I_2 Y_2 \tag{6.12}$$

Using the above equations, the circuit diagram of the given asynchronous circuit can be drawn, as shown in Fig. 6.7.

6.3 PULSE MODE ASYNCHRONOUS SEQUENTIAL CIRCUITS

In pulse mode asynchronous sequential circuits, the inputs and output are pulses. In this circuit, it is assumed that more than one pulse will not arrive at the input at the same time, and the duration of the pulses is long enough to cause state transition, and it is short enough so that there will not be more than one state transition for a single pulse.

The design procedure used for fundamental mode asynchronous sequential circuits is also applicable to pulse mode asynchronous sequential circuits. Designing a pulse mode asynchronous sequential circuits requires the following steps:

- Formulate what the circuit has to do, from the problem definition.
- Develop a primitive state table and specify the outputs that are associated with the stable state.
- Minimize the primitive state table by merge process. It is the process of identifying those primitive states that are related by input sequences and can be combined together into a single stable state, i.e. finding the states that can be put together in the same row of the excitation map. At the end of this merge process, the number of rows will be reduced.
- Assign state variables (i.e. secondary variables) to the rows of merged primitive state table and obtain present state/next state and output table. The outputs assigned to the unstable states vary according to the design requirements.
- Decide the memory element to be used and obtain the excitation and output table. Obtain the simplified expression for the excitation and output function.
- Draw the schematic circuit diagram.

Let us consider the design example of pulse mode asynchronous circuits.

Example 6.4 Design an asynchronous circuit that will output only the second pulse received and ignore other pulses.

Solution

The block diagram of the asynchronous circuit is shown in Fig. 6.8. It has a pulse input, and one output. As per the given design requirement, only the second pulse received in the pulse input should be available at the output, as shown in Fig. 6.9.

Fig. 6.8 Block diagram of asynchronous circuit

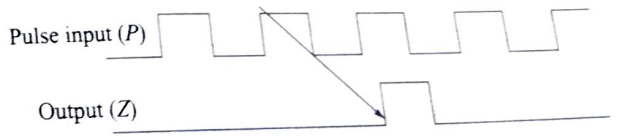


Fig. 6.9 Timing diagram

The primitive state table for the given requirement can be constructed, as shown in Table 6.13.

Table 6.13 Primitive state table

Row	Next state, Output	
	$P = 0$	$P = 1$
1	①,0	2
2	3	②,0
3	③,0	4
4		④,1
5	⑤,0	6
6	5	⑥,0

Initially, the input is 0 (i.e. $P = 0$), the circuit is in a stable state 1 and the output is 0, represented as ①, 0.

When P becomes 1, the circuit enters a stable state 2 and its output is 0. This is represented as ②, 0 in the second row. In the first row, an uncircled 2 is entered to indicate that a transition to ② state will occur as a result of the input change from 0 to 1. The uncircled 2 represents an unstable state.

When P becomes 0 and the circuit enters a stable state 3, its output is 0. This is represented as ③, 0 in the third row. In the second row, an uncircled 3 is entered to indicate that a transition to ③ state will occur as a result of the input change from 1 to 0. The uncircled 3 represents an unstable state.

When the circuit is in stable state ③ and the pulse input becomes 1, then the circuit enters a stable state ④ with output 1. This is represented as ④, 1 in the fourth row. It indicates that the second pulse is available at the output.

A circled entry and an uncircled entry can be combined to minimize the primitive state table by merge process. The primitive state table shown in Table 6.13 can be minimized by merging row 3 and row 4, as shown in Table 6.14. The stable state and the corresponding unstable states are combined as stable states, and the outputs associated with the unstable states correspond to their stable states.

Table 6.14 Minimized primitive state table

Table 6.15 Present state/next state and output table

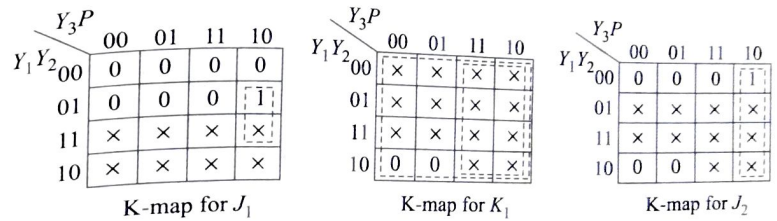
$Y_1Y_2Y_3$	$Y_1Y_2Y_3, Z$	
	$P = 0$	$P = 1$
000	000, 0	001, 0
001	010, 0	001, 0
010	010, 0	011, 1
011	100, 0	011, 1
100	100, 0	100, 0

If J-K flip-flop is used, then an excitation and output table is generated, as shown in Table 6.16.

Table 6.16 Excitation and output table

$Y_1Y_2Y_3$	$J_1K_1, J_2K_2, J_3K_3, Z$	
	$P = 0$	$P = 1$
000	0×, 0×, 0×, 0	0×, 0×, 1×, 0
001	0×, 1×, ×1, 0	0×, 0×, ×0, 0
010	0×, ×0, 0×, 0	0×, ×0, 1×, 1
011	1×, ×1, ×1, 0	0×, ×0, ×0, 1
100	×0, 0×, 0×, 0	×0, 0×, 0×, 0

The excitation function J_1K_1, J_2K_2, J_3K_3, and Z can be simplified using the K-maps, as shown in Fig. 6.10.



$$J_1 = Y_2 Y_3 P \quad (6.13)$$
$$K_1 = Y_3 \quad (6.14)$$
$$J_2 = Y_3 P \quad (6.15)$$
$$K_2 = Y_3 P \quad (6.16)$$
$$J_3 = \bar{Y}_1 P \quad (6.17)$$
$$K_3 = \bar{P} \quad (6.18)$$
$$Z = Y_2 P \quad (6.19)$$

Using the above equations, the circuit diagram of the given asynchronous circuit can be drawn, as shown in Fig. 6.11.

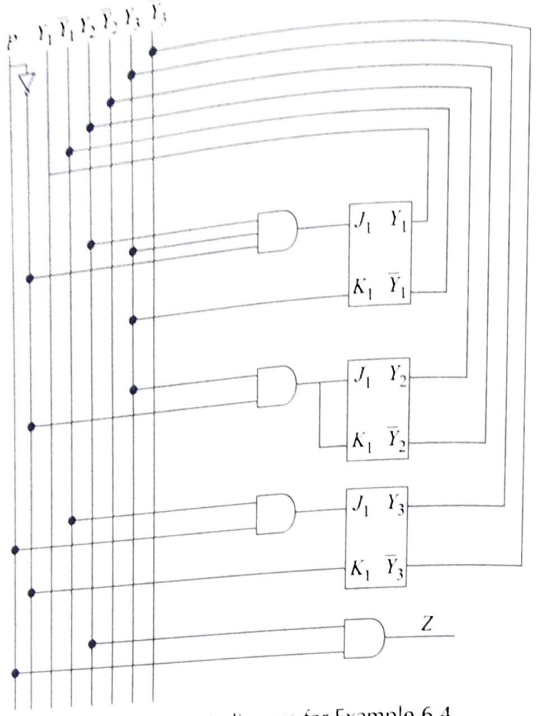


Fig. 6.11 Circuit diagram for Example 6.4

Example 6.5 Design as asynchronous circuit that will output only the first pulse received whenever a control input is asserted from LOW to HIGH state. Any further pulses will be ignored.

Solution

The block diagram of the asynchronous circuit is shown in Fig. 6.12. It has two inputs: a pulse input, and a control input, and one output. As per the given design requirement, only the first pulse received in pulse input (P) is available at the output (Z), after the control input is asserted form LOW to HIGH state, and further input pulses are ignored.

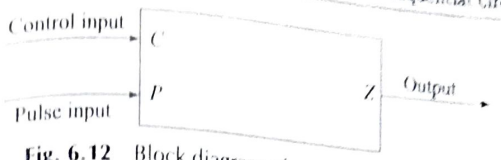


Fig. 6.12 Block diagram of asynchronous circuit

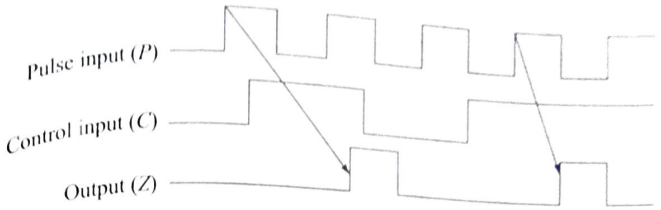


Fig. 6.13 Timing diagram

The primitive state table for the given requirement can be constructed, as shown in Table 6.17.

Table 6.17 Primitive state table

Row	*Next state, Output*			
	CP = 0 0	***CP = 0 1***	***CP = 1 1***	***CP = 1 0***
1	①, 0	2		3
2	1	②, 0	4	
3	1		5	③, 0
4		2	④, 1	3
5		6	⑤, 0	7
6	1	⑥, 0	5	
7	1		8	⑦, 0
8		2	⑧, 0	7

Initially both inputs are equal to zero (i.e. $C = P = 0$), the circuit is said to be in a stable state 1, represented as ①, 0 .

When C remains same at 0 and P becomes 1, the circuit enters a stable state 2 and its output is 0. This is represented as ②, 0 in the second row. In the first row, an uncircled 2 is entered to indicate that a transition to ② state will occur as a result of the input change from 00 to 01. The uncircled 2 represents an unstable state.

Similarly, when C becomes 1 and P remains same at 0, the circuit enters a stable state 3 and its output is 0. This is represented as ③, 0 in the third row. In the first row, an uncircled 3 is entered to indicate that a transition to ③ state will occur as a result of the input change from 00 to 10. The uncircled 3 represents an unstable state.

When the circuit is in stable state ② and the control input (C) becomes 1, then the circuit enters a stable state ④ with output 1. This is represented as ④, 1 in the fourth row. It indicates that the first pulse is available at the output. In the second row, an uncircled 4 is entered to indicate that a transition to ④ state will occur as a result of the input change from 01 to 11. The uncircled 4 represents an unstable state.

When the circuit is in stable state ④, and C becomes 0, the circuit enters a stable state ⑤ with output 0. When P also becomes 0, it will go to the initial state. Similarly, if the circuit is in stable state ④, and P becomes 0, the circuit enters a stable state ⑥ with output 0. From here, when C also becomes 0, it will go to stable state 1.

Table 6.18 Minimized primitive state table

	Next state, Output		
CP = 00	***CP = 01***	***CP = 11***	***CP = 10***
①, 0	②, 0	④, 1	3, 0
1, 0	⑥, 0	5, 0	③, 0
	6, 0	⑤, 0	7, 0
	2, 0	⑧, 0	⑦, 0
1, 0			

The primitive state table shown in Table 6.17 can be minimized by merging rows 1, 2, and 4; rows 3, 6; and rows 5, 7, and 8, as shown in Table 6.18. The stable state and the corresponding unstable states are combined as the stable state, and the outputs associated with the unstable states correspond to their stable states.

Table 6.19 PS/NS and output table

	$Y_1 Y_2$, Z			
Y_1Y_2	***CP = 00***	***CP = 01***	***CP = 11***	***CP = 10***
0 0	00, 0	00, 0	00, 1	10, 0
0 1	00, 0	01, 0	10, 0	10, 0
1 0	00, 0	01, 0	10, 0	11, 0
1 1	00, 0	00,0	11,0	11,0

If J-K flip-flop is used, then an excitation and output table is generated, as shown in Table 6.20.

Table. 6.20 Excitation and output table

	$J_1 K_1, J_2 K_2$, Z			
Y_1Y_2	***CP = 00***	***CP = 01***	***CP = 11***	***CP = 10***
0 0	0 ×, 0 ×, 0	0 ×, 0 ×, 0	0 ×, 0 ×, 1	1 ×, 0 ×, 0
0 1	0 ×, × 1, 0	0 ×, × 0, 0	1 ×, × 1, 0	1 ×, × 1, 0
1 0	× 1, 0 ×, 0	× 1, 1 ×, 0	× 0, 0 ×, 0	× 0, 1 ×, 0
1 1	× 1, × 1, 0	× 1, × 1, 0	× 0, × 0, 0	× 0, × 0, 0

The excitation function J_1K_1, J_2K_2, and Z can be simplified using K-maps, as shown in Fig. 6.14.

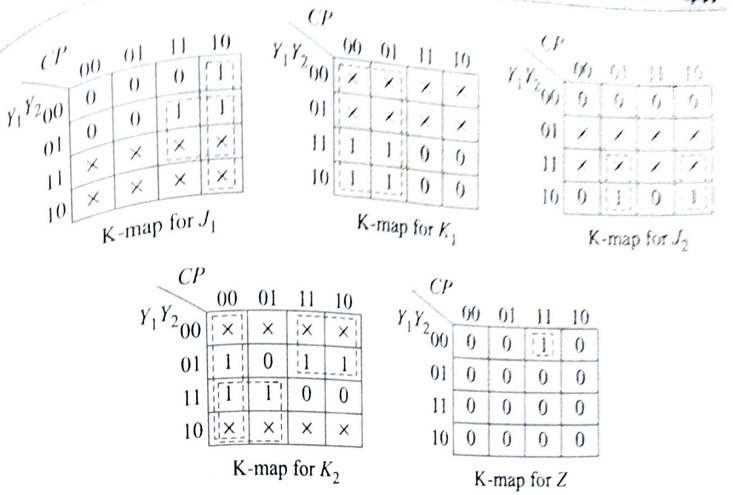


Fig. 6.14 K-map for J_1, K_1, J_2, K_2, and Z

$$J_1 = C\overline{P} + CY_2 \tag{6.20}$$

$$K_1 = \overline{C} \tag{6.21}$$

$$J_2 = C\overline{P}\,Y_1 + \overline{C}\,PY_1 \tag{6.22}$$

$$K_2 = C\overline{Y}_1 + \overline{C}\,Y_1 + \overline{C}\,\overline{P} \tag{6.23}$$

$$Z = CP\overline{Y}_1\overline{Y}_2 \tag{6.24}$$

Using the above equations, the circuit diagram of the given asynchronous circuit can be drawn, as shown in Fig. 6.15.

6.4 INCOMPLETELY SPECIFIED STATE MACHINES

In sequential circuits, some of the states are not specified. Such sequential circuits are known as incompletely specified state machines. In sequential circuits, not all combinations of states and inputs are possible. For example, when a machine is in a state C, it will never receive a '1' as input and consequently, the corresponding transition and output map may be left unspecified, as shown in Table 6.21, by a dash in row C and column $I = 1$.

Table 6.21 State table

Present state	*Next state and output*	
	I=0	*I=1*
A	*B*,0	*D*,1
B	*C*, 0	*A*,0
C	A,0	—
D	*C*,0	*B*,0

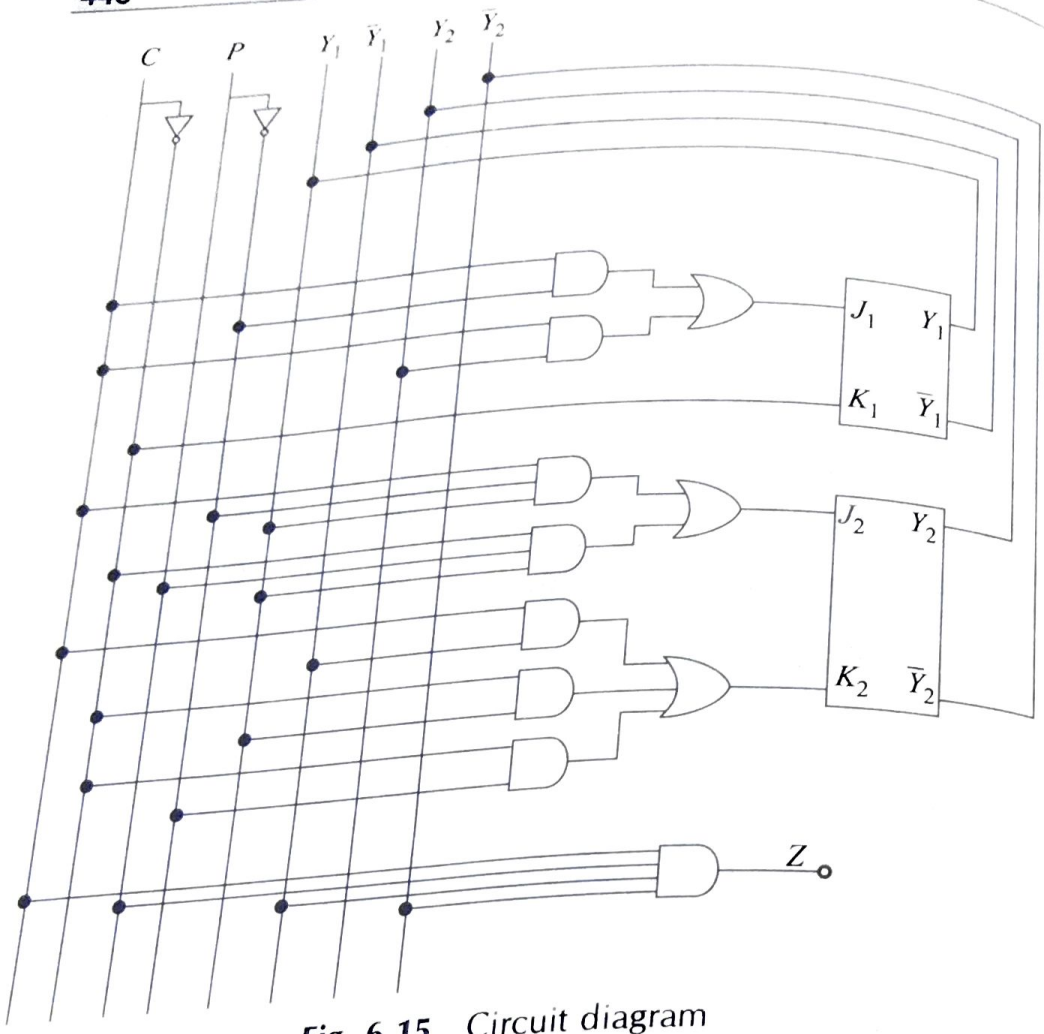


Fig. 6.15 Circuit diagram

In some sequential circuits, the state transitions are completely defined but for some combinations of states and inputs, the output values may not be critical and thus, left unspecified. For example, when a machine is in a state B, it will receive a '0' as input and consequently, the corresponding transition C occurs, but the output values are not critical and thus, left unspecified, as shown in Table 6.22, by a dash in row B and column I = 0.

Table 6.22 State table

Present state	*Next state and output*	
	I=0	*I=1*
A	*B*, 0	*D*, 1
B	*C*, –	*B*, 0
C	*A*, 0	*D*, 0
D	*C*, 0	*B*, 0

When a state transition is not specified, the behaviour of the machine may become unpredictable. To avoid such a situation, it is assumed that when the machine is in any of its possible starting states, the input sequences are applied in such a way that

Table 6.24 PS/NS state table

Present state	Next state (Y_1Y_2) Z			
(Y_1Y_2)	$I_1I_2 = 00$	$I_1I_2 = 01$	$I_1I_2 = 11$	$I_1I_2 = 10$
00	11(1)	00(3)	11(9)	11(13)
01	01(4)	11(6)	11(10)	00(15)
10	10(2)	00(7)	11(11)	10(16)
11	10(5)	11(8)	11(12)	10(14)

Assume that the circuit is in stable state 3 (i.e. $Y_1Y_2 = 00$ corresponding to $X_1X_2 = 01$) and input I_2 changes from 1 to 0. Now the next state variable Y_1Y_2 begins to switch from 00 to 11. But due to an unequal propagation delay, one of the next state variables becomes 1 and other variables do not change from the value 0, and hence, $Y_1Y_2 = 01$. Then the circuit goes to stable state 4. If $Y_1 = Y_2 = 10$, then the circuit goes to stable state 2. Therefore, for an input change, i.e. from $I_1I_2 = 01$ to 00, depending on the relative switching speed of the next state variables Y_1Y_2, the final state will be either stable state 2 or 4. But both final states, 2 and 4, are wrong for the given input change. This situation is called a *critical race*, which must be avoided in an asynchronous circuit.

Non-critical races

A non-critical race problem may always go to a correct final stable state, after its transition through unstable states. It is also because of different propagation delays. Consider Table 6.24 and assume that the circuit is in stable state 3 (i.e. $Y_1Y_2 = 00$). Now, if the input changes from $I_1I_2 = 01$ to 11, then the next state variables (Y_1Y_2) of the circuit are supposed to switch from 00 to 11. Again, due to propagation delay, Y_1Y_2 becomes either 01 or 10. Therefore, the circuit will either go to state (10) or (11) and finally reach the stable state 12. Such a situation where the correct final stable state is reached after a transition through unstable states is called a *non-critical race*. Non-critical races can be tolerated in an asynchronous circuit.

6.5.3 Hazards

Combinational circuits used in asynchronous sequential circuits may have unequal propagation delays. Hazard is an unwanted transient, i.e. spike or glitch that occurs due to unequal propagation delays through a combinational circuit. There are two types of hazards: *static hazard* and *dynamic hazard*, as shown Fig. 6.16.

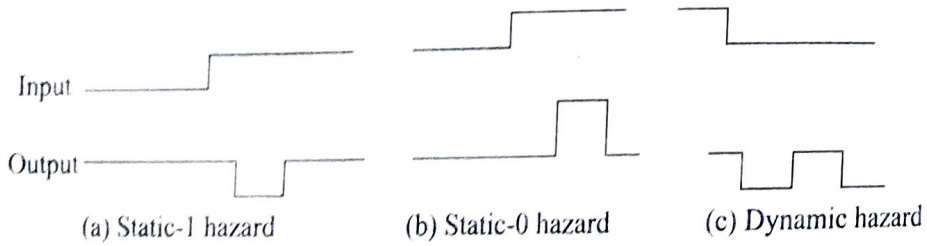


Fig. 6.16 Types of hazards

Static hazard is a condition which results in a single momentary incorrect output due to the change in an input variable when the output is expected to remain in the same state. Static hazard is of two types: *Static-1 hazard* and *Static-0* hazard.

Dynamic hazard

Multiple glitch situations at the output due a change from 0 to 1 (or form 1 to 0) is known as *dynamic hazard*. It can be removed by forming a group of adjacent cells, as in static hazard.

$$y = \overline{A}C + B\overline{C} + AB \tag{6.26}$$

Essential hazard

It is a type of hazard that exists only in asynchronous sequential circuits with two or more feedbacks. Essential hazard occurs normally in toggling type circuits. It is an operational error generally caused by an excessive delay to a feedback variable in response to an input change, leading to a transition to an improper state. Essential hazards cannot be eliminated by adding redundant gates, same as static hazards.

6.6 DESIGN OF HAZARD-FREE SWITCHING CIRCUITS

Static, dynamic, and essential hazards discussed in the previous section can be eliminated by using different procedures as discussed below.

6.6.1 Static Hazards Elimination

As discussed in the previous section, hazard is an unwanted transient, i.e. spike or glitch that occurs due to unequal propagation delays through a combinational circuit. Static hazard is a condition which results in a single momentary incorrect output due to change in an input variable, when the output is expected to remain in the same state. Such hazards occur whenever there exists a pair of adjacent input combinations that produces the same output.

Therefore, to design a static hazard-free switching circuit, we need to consider all adjacent input combinations of every pair of adjacent 1 cells and every pair of adjacent 0 cells in the K-map of a switching function at least one sub-cube.

Consider the K-map for the function, $Y(A, B, C) = \Sigma(0, 2, 4, 5)$, as shown in Fig. 6.21. The corresponding circuits for the simplified sum of products (SOP) and the product of sums (POS) expression are shown in Figs 6.22 and 6.23, respectively.

The simplified sum of product (SOP) expression is

$$F = \overline{A}\,\overline{C} + A\overline{B} \tag{6.27}$$

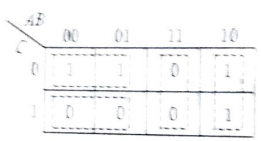


Fig. 6.21 K-map with hazard

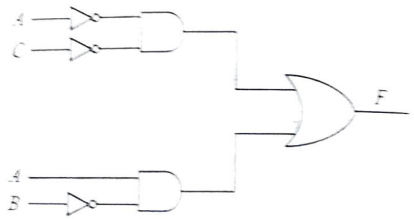


Fig. 6.22 Implementation of SOP expression

In the circuit (Fig. 6.23), when the input (ABC) changes from 100 to 000 or from 000 to 111, the output may momentarily go to 0 state due to the unequal propagation delay in the AND gates, rather than remaining constant at 1. As a result, static-1 hazard is present.

The simplified product of sums (POS) expression is

$$F = (A + \overline{C})(\overline{A} + \overline{B}) \tag{6.28}$$

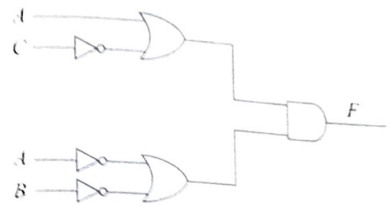


Fig. 6.23 Implementation of POS expression

In the circuit (Fig. 6.23), when the input changes from 111 to 011 or from 011 to 111, the output may momentarily go to 1 state due to the unequal propagation delay in the OR gates, rather than remaining constant at 0. As a result, static-0 hazard is present.

To make the switching circuits shown in Figs 6.22 and 6.23 hazard-free, form at least one sub-cube pair of adjacent 1 cells or adjacent 0 cells. To remove the static-1 hazard present in Fig. 6.22, form at least one sub-cube pair of adjacent 1 cells, as shown in Fig. 6.24 by dotted sub-cubes. To remove the static-0 hazard present in Fig. 6.23, form at least one sub-cube pair of adjacent 0 cells, as shown in Fig. 6.24 by dotted sub-cubes.

The simplified sum of product (SOP) expression is

$$F = \overline{A}\,\overline{C} + A\overline{B} + \overline{B}\,\overline{C} \tag{6.29}$$

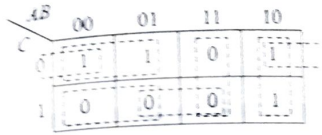


Fig. 6.24 K-map for hazard-free circuit

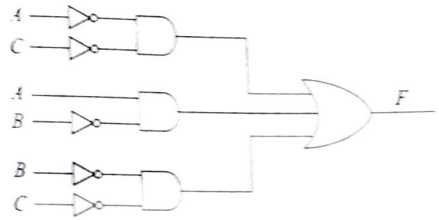


Fig. 6.25 Implementation of SOP expression

The simplified product of sum (POS) expression is

$$F = (A + \overline{C})(\overline{A} + \overline{B})(\overline{C} + \overline{B}) \tag{6.30}$$

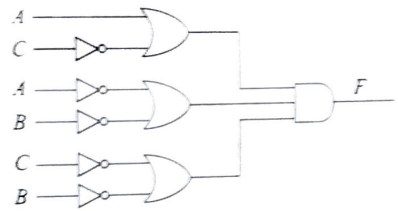


Fig. 6.26 Implementation of POS expression

PLAs can also be used for implementing sequential circuits like counters or synchronous machines. The design methodology for sequential circuits remains the same as discussed in Chapter 5. In the complete design, PLA is used for implementing the combinational circuits required to derive the flip-flop inputs. Hence, the block diagram of a sequential circuit using PLA can be represented as shown in Fig. 8.15.

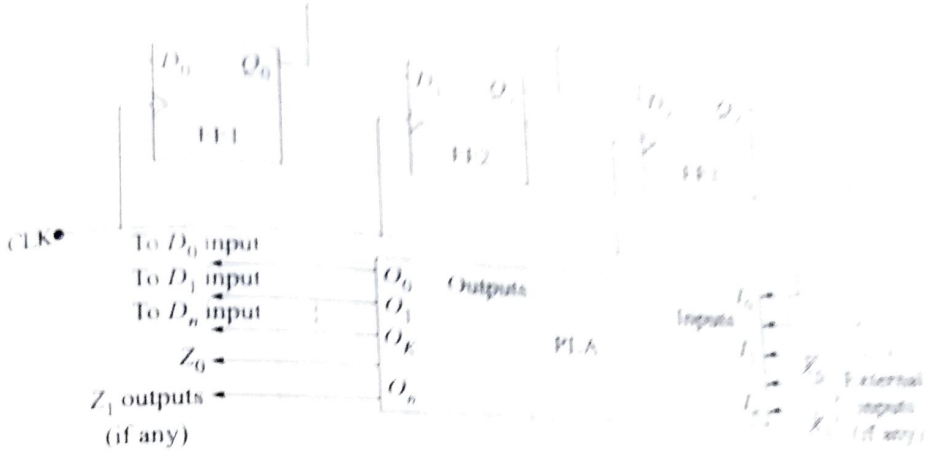


Fig. 8.15 Block diagram of sequential circuit design using PLA and DFFs

Example 8.3 Design a synchronous sequential circuit for the state diagram shown in Fig. 8.16 and implement it using D flip-flops and a suitable PLA.

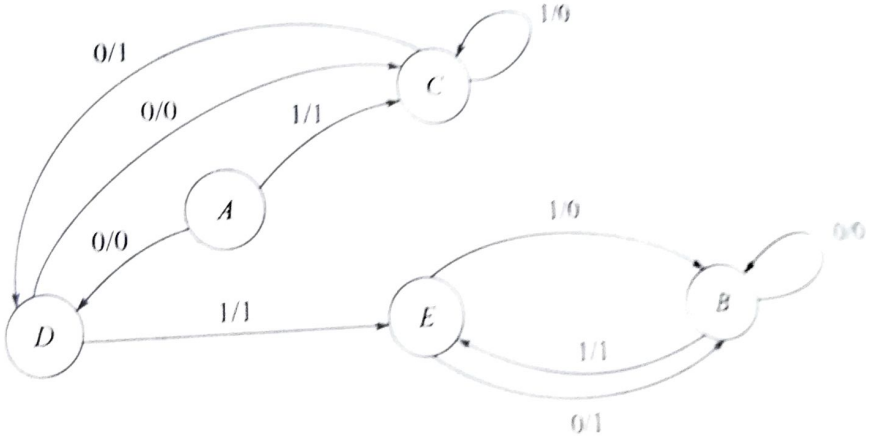


Fig. 8.16 State diagram for Example 8.3

Solution

The state table for Fig. 8.16 is shown in Table 8.4.

Since there are five distinct states, we need to use three flip-flops. The flip-flop inputs will be D_A, D_B, D_C and the outputs will be Q_A, Q_B, Q_C.

Let the state assignments be

$A = 000$, $B = 001$, $C = 010$, $D = 011$, $E = 100$.

Table 8.5 shows the excitation table for all the three D flip-flops for the desired state changes by considering the next state for all unassigned state as state A (000).

Table 8.4 State table for Example 8.3

Present state	Next state		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
A	D	C	0	1
B	B	F	0	1
C	D	C	1	0
D	C	F	0	1
F	B	B	1	0

Table 8.5 Excitation table for Example 8.3

Present state			Input	Next state			FF inputs			o/p
Q_A	Q_B	Q_C	x	Q_A	Q_B	Q_C	D_A	D_B	D_C	Z
0	0	0	0	0	1	1	0	1	1	0
0	0	0	1	0	1	0	0	1	0	0
0	0	1	0	0	0	1	0	0	1	1
0	0	1	1	1	0	0	1	0	0	0
0	1	0	0	0	1	1	0	1	1	1
0	1	0	1	0	1	0	0	1	0	1
0	1	1	0	0	1	0	0	1	0	1
0	1	1	1	1	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	1	0
1	0	0	1	0	0	1	0	0	1	1
1	0	1	X	0	0	0	0	0	0	1
1	1	0	X	0	0	0	0	0	0	0
1	0	1	X	0	0	0	0	0	0	0

Figure 8.17 shows the K-maps for the inputs D_A, D_B, D_C and the output Z, and gives their SOP forms having a minimum number of distinct product terms.

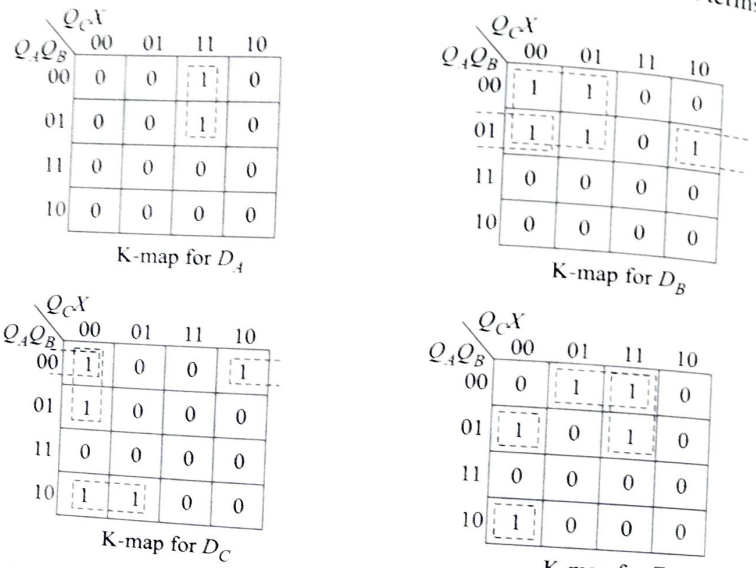


Fig. 8.17 K-maps for D_A, D_B, D_C

The design equations for the inputs D_A, D_B, D_C and the output Z are

$$D_A = \overline{Q}_A Q_C X \tag{8.10}$$

$$D_B = \overline{Q}_A \overline{Q}_C + \overline{Q}_A Q_B \overline{X} \tag{8.11}$$

$$D_C = \overline{Q}_A \overline{Q}_C \overline{X} + Q_A \overline{Q}_B \overline{Q}_C + \overline{Q}_A Q_B \overline{X} \tag{8.12}$$

$$Z = \overline{Q}_A \overline{Q}_B X + \overline{Q}_A Q_C X + \overline{Q}_A Q_B \overline{Q}_C \overline{X} + Q_A \overline{Q}_B \overline{Q}_C \overline{X} \tag{8.13}$$

Equations 8.10, 8.11, 8.12, and 8.13 indicate that there are nine distinct products and one being shared amongst D_A and Z. Hence, we need to use a (4 × 4) PLA with nine product terms. Table 8.6 shows the program table for a (4 × 4) PLA with nine product terms.

Table 8.6 PLA program table for Example 8.3

Product terms		Inputs				Outputs			
		Q_A	Q_B	Q_C	x	D_A	D_B	D_C	Z
$\overline{Q}_A Q_C X$	1	0	–	1	1	1	–	–	1
$\overline{Q}_A \overline{Q}_C$	2	0	–	0	–	–	1	–	–
$\overline{Q}_A Q_B \overline{X}$	3	0	1	–	0	–	1	–	–
$\overline{Q}_A \overline{Q}_C \overline{X}$	4	0	–	0	0	–	–	1	–
$Q_A \overline{Q}_B \overline{Q}_C$	5	1	0	0	0	–	–	1	–
$\overline{Q}_A \overline{Q}_B \overline{X}$	6	0	0	0	–	–	–	–	1
$\overline{Q}_A \overline{Q}_B X$	7	0	0	–	0	–	–	–	1
$\overline{Q}_A Q_B \overline{Q}_C \overline{X}$	8	0	1	0	0	–	–	–	1
$Q_A \overline{Q}_B \overline{Q}_C \overline{X}$	9	1	0	0	0	–	–	–	1
						T	T	T	T

Figure 8.18 shows the diagram of a (4 × 4) PLA with nine product terms programmed according to Table 8.6 with D flip-flops.

8.3 PROGRAMMABLE ARRAY LOGIC

Programming array logic (PAL) devices are those PLDs which are very widely used and are readily available. PALs, like PLAs, consist of an array of AND gates and an array of OR gates. The major difference between PLA and PAL devices is that, in case of PLA, both the arrays are programmable, whereas in PAL, OR array is fixed and AND array is programmable. Because only the AND gates are programmable, the PAL is easier to program, but is not as flexible as the PLA

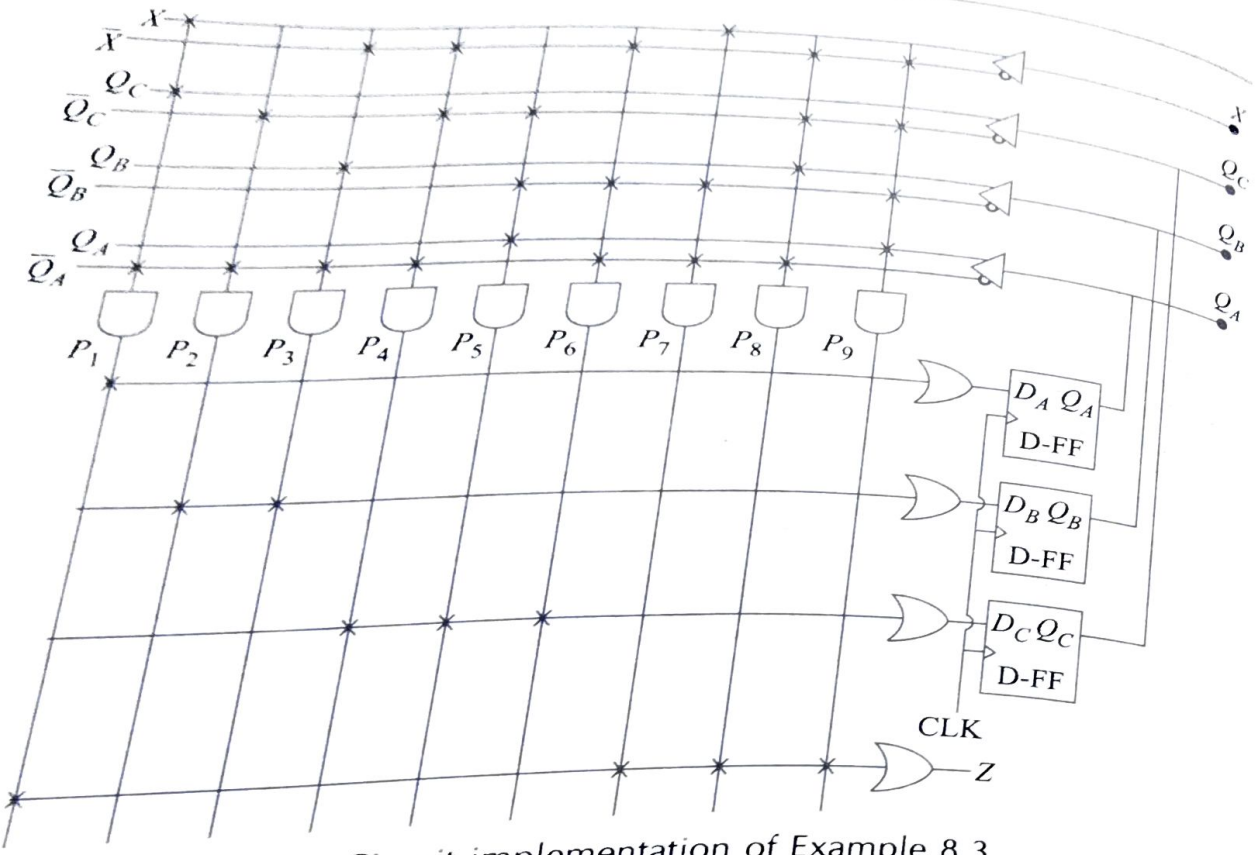
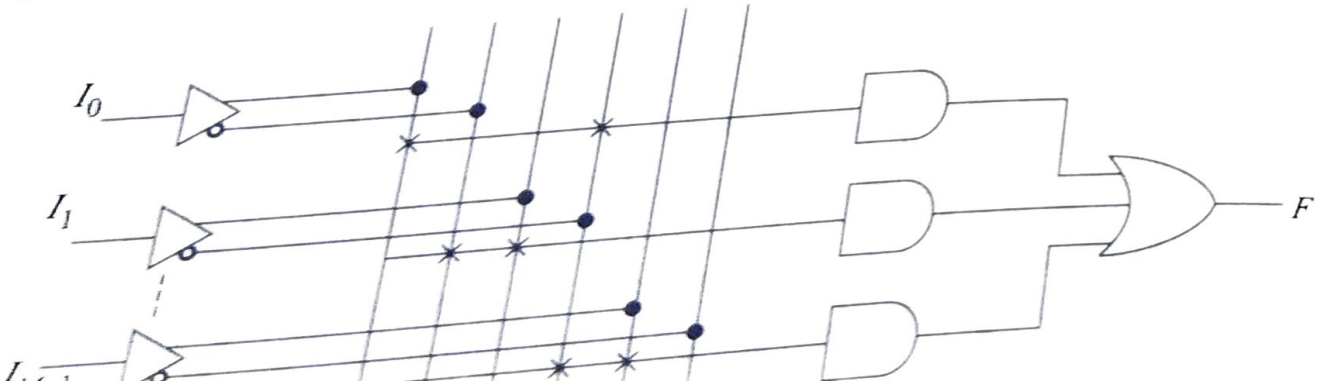


Fig. 8.18 *Circuit implementation of Example 8.3*

8.3.1 Block Diagram of PAL

PAL is a type of fixed architecture logic device with programmable AND gates and fixed OR gates. The block diagram of PAL is as shown in Fig. 8.19.



any function is more, then it may be necessary for the sections to implement Boolean function. In PALs, unlike the PLAs, a product term cannot be shared among two or more OR gates.

Example 8.4 A 3-input, 4-output combinational circuit has the following output functions. Implement the circuit using a suitable PAL.

$$A(x, y, z) = \Sigma m(1, 2, 4, 6)$$

$$B(x, y, z) = \Sigma m(0, 1, 3, 6, 7)$$

$$C(x, y, z) = \Sigma m(1, 2, 4, 6, 7)$$

$$D(x, y, z) = \Sigma m(1, 2, 3, 5, 7)$$

Solution

The K-maps for outputs A, B, C and D are given in Fig. 8.22.

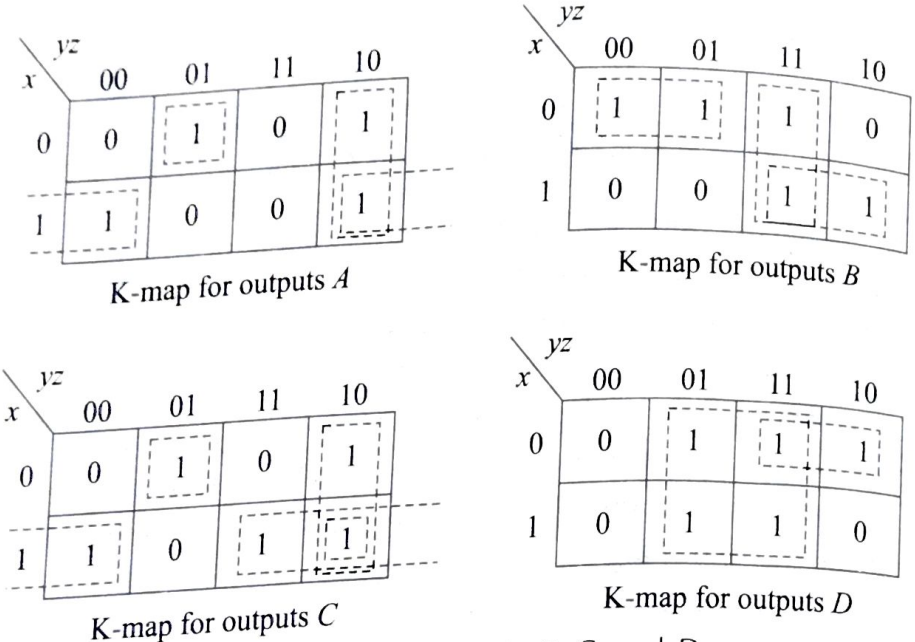


Fig. 8.22 K-maps for A, B, C, and D

$$A = x\bar{z} + y\bar{z} + \bar{x}\,\bar{y}\,z \tag{8.14}$$

$$B = \bar{x}\,\bar{y} + yz + xy \tag{8.15}$$

$$C = \bar{x}\,\bar{y}\,z + x\,\bar{z} + xy + y\,\bar{z} \tag{8.16}$$

$$D = z + \bar{x}\,y \tag{8.17}$$

From the expression of output C (8.16) and output A (8.14),

$$C = xy + A$$

From equations 8.14, 8.15, 8.16, and 8.17, it can be seen that there are 3 inputs, 4 outputs, and a maximum of 3 product terms per OR gate. Hence, a 4×4 PAL is required with programmable input/output so as to allow us the use of one output function (A) in the other output function (C) with feedback.

The PAL program table shown in Table 8.8 contains three columns specifying [illegible] inputs and outputs. In this example, the input column also

Mealy and Moore type of sequential circuits. The PALs which are designed for sequential circuits have bi-directional capacity. As shown in Fig. 8.20(d), registered outputs have tri-state control. If they are disabled, these output pins can be used for giving additional external inputs and if the tri-state control is enabled, the pins give feedback to the AND gates. Since these PALs have flip-flops as well as AND-OR arrays, a complete sequential circuit can be fit into it without any external hardware.

***Example* 8.5** Design a 4-bit Gray code counter using a suitable PAL device. Show the coded logic diagram of the device.

Solution

The state diagram for a 4-bit gray counter is shown in Fig. 8.24.

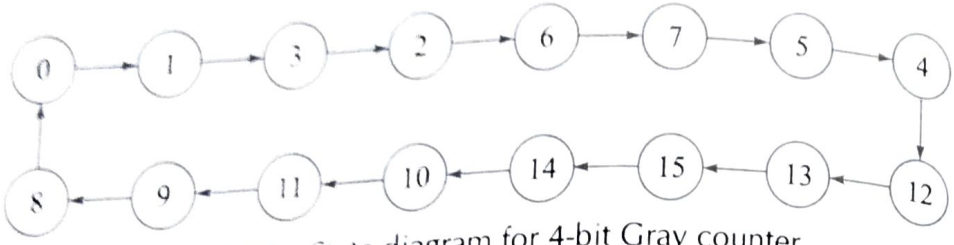


Fig. 8.24 State diagram for 4-bit Gray counter

Table 8.9 Excitation table for Example 8.5

CLK	*Present State*				*Next State*				*Flip-flop i/p*			
	Q_0	Q_1	Q_2	Q_3	Q_0	Q_1	Q_2	Q_3	D_0	D_1	D_2	D_3
1	0	0	0	0	0	0	0	1	0	0	0	1
2	0	0	0	1	0	0	1	1	0	0	1	1
3	0	0	1	1	0	0	1	0	0	0	1	0
4	0	0	1	0	0	1	1	0	0	1	1	0
5	0	1	1	0	0	1	1	1	0	1	1	1
6	0	1	1	1	0	1	0	1	0	1	0	1
7	0	1	0	1	0	1	0	0	0	1	0	0
8	0	1	0	0	1	1	0	0	1	1	0	0
9	1	1	0	0	1	1	0	1	1	1	0	1
10	1	1	0	1	1	1	1	1	1	1	1	1
11	1	1	1	1	1	1	1	0	1	1	1	0
12	1	1	1	0	1	0	1	0	1	0	1	0
13	1	0	1	0	1	0	1	1	1	0	1	1
14	1	0	1	1	1	0	0	1	1	0	0	1
15	1	0	0	1	1	0	0	0	1	0	0	0
16	1	0	0	0	0	0	0	0	0	0	0	0

The K-maps for D_0, D_1, D_2, and D_3 are shown in Fig. 8.25.

$$D_0 = Q_0Q_3 + Q_0Q_2 + Q_1\overline{Q}_2\overline{Q}_3 \tag{8.18}$$

$$D_1 = Q_1\overline{Q}_2 + Q_1Q_3 + \overline{Q}_0Q_2\overline{Q}_3 \tag{8.19}$$

- When both the inputs are low, Q_2 and Q_3 are OFF. The current cannot flow through the drain terminal and the output is high (V_{DD}).
- When any one of the inputs is high (0 V or –ve), then the corresponding MOSFET is ON. The current flows through the circuit and the output is low.
- When inputs are high (+ve voltage), Q_2 and Q_3 are ON. The current flows through the drain terminal and the output is low.

The operation of the circuit is summarized in Table 1.13(a).

Table 1.13(a) NMOS NOR gate

A	B	Q_2	Q_3	V_o
LOW	LOW	OFF	OFF	HIGH
LOW	HIGH	OFF	ON	HIGH
HIGH	LOW	ON	OFF	HIGH
HIGH	HIGH	ON	ON	LOW

In terms of 0 and 1, Table 1.13(a) can be written as in Table 1.13(b).

Table 1.13(b) NMOS NOR gate

A	B	V_o
0	0	1
0	1	0
1	0	0
1	1	0

The circuit shown in Fig. 1.43 acts as a two-inputs NAND gate and its truth table is given in Table 1.13(b).

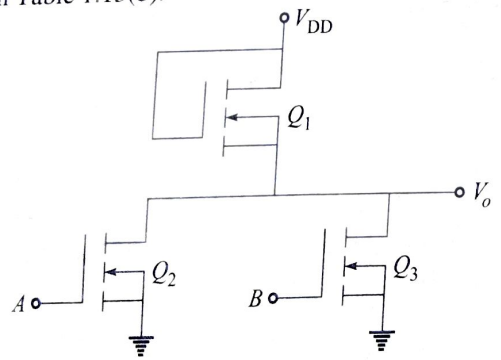


Fig. 1.43 NMOS NOR gate

1.17.4 Fan-out

We have seen that the fan-out of a logic family is the function of its input and output impedance. For high input impedance, the fan-out is large. MOSFET devices have very high input impedance, therefore fan-out is large. Since the number of driving gates is more, the capacitor at the driving gate output considerably reduces the speed of MOSFET gate.

1.17.5 Propagation Delay Time

It is a function of the capacitor of loaded gate and the charging resistor. In case of MOS devices, the large capacitor is present at input and output and the resistor through which the capacitor gets charged or discharged is also high. Hence, the propagation delay is large and the speed of operation is low.

1.17.6 Power Dissipation

It is a function of current supply by the source and resistance of the load. The power supply by the source in MOS logic family is small and hence the power dissipation is low.

1.17.7 Characteristics of NMOS

Table 1.14 summarizes the characteristics of NMOS.

Table 1.14 Characteristics of NMOS

Parameter	*Value*	*Parameter*	*Value*	*Parameter*	*Value*
V_{IH}	2.0 V	I_{OH}	– 400 μA	t_{PHL}	60 ns
V_{IL}	0.8 V	I_{OL}	2 mA	t_{PLH}	45 ns
V_{OH}	2.4 V	Fan-out	30	P.D.	0.1 mW
V_{OL}	0.45 V	Noise margin	1.5 V		

1.18 CMOS

CMOS family uses *n*-channel and *p*-channel MOSFETs. In CMOS, *p*-channel and *n*-channel MOS devices are fabricated on the same chip, which makes its fabrication complicated but it reduces the packaging density, and has small power consumption. Hence, CMOS is ideally suited for battery-operated systems.

1.18.1 CMOS Inverter

Figure 1.44 shows a CMOS logic inverter. For the circuit, the logic levels are 0 V and V_{CC}. It is important to note that the *p*-channel MOSFET is ON, when the input is 0 V and the *n*-channel MOSFET is ON, when the input is V_{CC}. Q_1 is *p*-channel and Q_2 is *n*-channel. When Q_1 is ON, the output voltage is equal to V_{CC} and when Q_2 is ON, the output voltage is equal to 0 V.

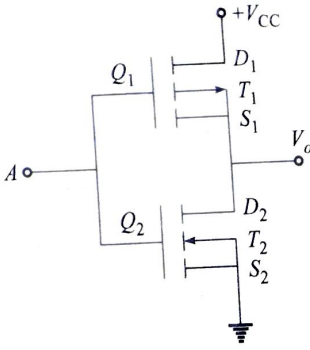


Fig. 1.44 CMOS inverter

Operation

- When the input is low, Q_1 is ON and Q_2 is OFF, output is high.
- When the input is high, Q_1 is OFF and Q_2 is ON, output is low.