جامعة تكريت
كلية هندسة الشرقاط
TU - ENGINEERING COLLEGE - ALSHIRQAT

جامعة تكريت

Digital System Design

Electrical Department-Fourth Stage

Lecture Five

By

Assistant Lecturer: Adnan Ali Abdullah

2025-2024

Finite State Machines and Algorithm State machine (ASM) with applications.

Lecture Outlines

- ❑Up/Down Synchronous Counters
- ❑Design of Synchronous Counters

Up/Down Synchronous Counters

- An **up/down counter** is one that is capable of progressing in either direction through a certain sequence.
- An up/down counter, sometimes called a **bidirectional counter**, can have any specified sequence of states..
- A 3-bit binary counter that advances upward through its sequence (0, 1, 2, 3, 4, 5, 6, 7) and then can be reversed so that it goes through the sequence in the opposite direction (7, 6, 5, 4, 3, 2, 1, 0) is an illustration of up/down sequential operation.
- In general, most up/down counters can be reversed at any point in their sequence. For instance, the 3-bit binary counter can be made to go through the following sequence:

$$\overbrace{0, 1, 2, 3, 4, 5,}^{\text{UP}} \underbrace{4, 3, 2,}_{\text{DOWN}} \overbrace{3, 4, 5, 6, 7,}^{\text{UP}} \underbrace{6, 5,}_{\text{DOWN}} \text{etc.}$$

- Table 1 shows the complete up/down sequence for a 3-bit binary counter.

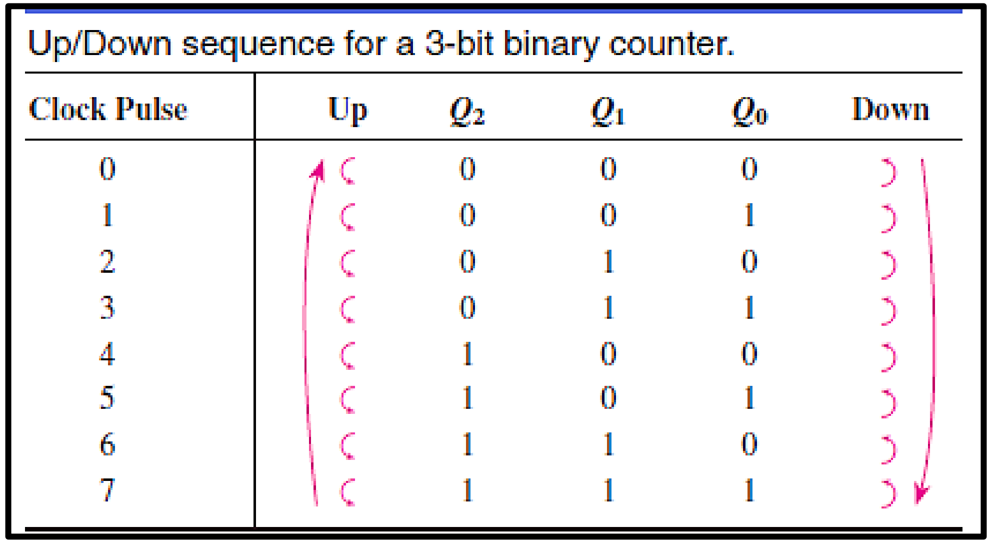
- The arrows indicate the state-to-state movement of the counter for both its UP and its DOWN modes of operation. An examination of Q0 for both the up an down sequences shows that FF$_0$ toggles on each clock pulse. Thus, the J$_0$ and K$_0$ inputs of FF$_0$ are

$$\boxed{J_0 = K_0 = 1}$$

- For **the up sequence**, Q$_1$ changes state on the next clock pulse when Q$_0$ = 1. For the down sequence, Q$_1$ changes on the next clock pulse when Q$_0$ = 0. Thus, the J$_1$ and K$_1$ inputs of FF$_1$ must equal 1 under the conditions expressed by the following equation:

$$\boxed{J_1 = K_1 = (Q_0 \,.\, UP) + (\overline{Q_0} \,.\, DOWN)}$$

Table 1



Up/Down sequence for a 3-bit binary counter.

Clock Pulse	Up	Q_2	Q_1	Q_0	Down
0		0	0	0	
1		0	0	1	
2		0	1	0	
3		0	1	1	
4		1	0	0	
5		1	0	1	
6		1	1	0	
7		1	1	1	

- For **the up sequence**, Q_2 changes state on the next clock pulse when $Q_0 = Q_1 = 1$. For the down sequence, Q_2 changes on the next clock pulse when $Q_0 = Q_1 = 0$. Thus, the J_2 and K_2 inputs of FF2 must equal 1 under the conditions expressed by the following equation:

$$\boxed{J_2 = K_2 = (Q_0 \cdot Q_1 \cdot \text{UP}) + (\overline{Q_0} \cdot \overline{Q_1} \cdot \text{DOWN})}$$

- Each of the conditions for the J and K inputs of each flip-flop produces a toggle at the appropriate point in the counter sequence.

- Figure 1 shows a basic implementation of a 3-bit up/down binary counter using the logic equations just developed for the J and K inputs of each flip-flop. Notice that the UP/$\overline{\text{DOWN}}$ control input is HIGH for UP and LOW for DOWN.

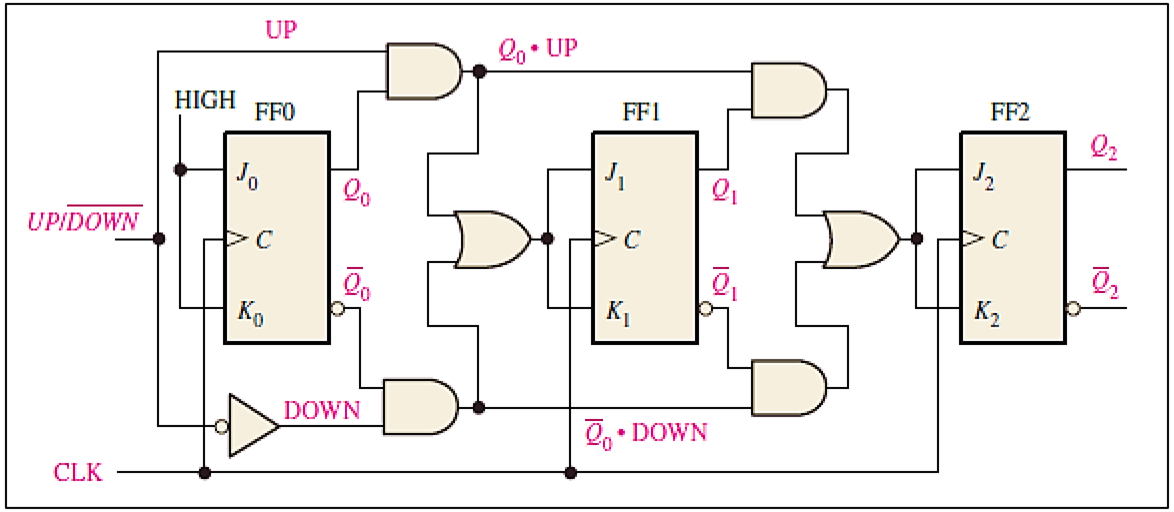


FIGURE 1 A basic 3-bit up/down synchronous counter

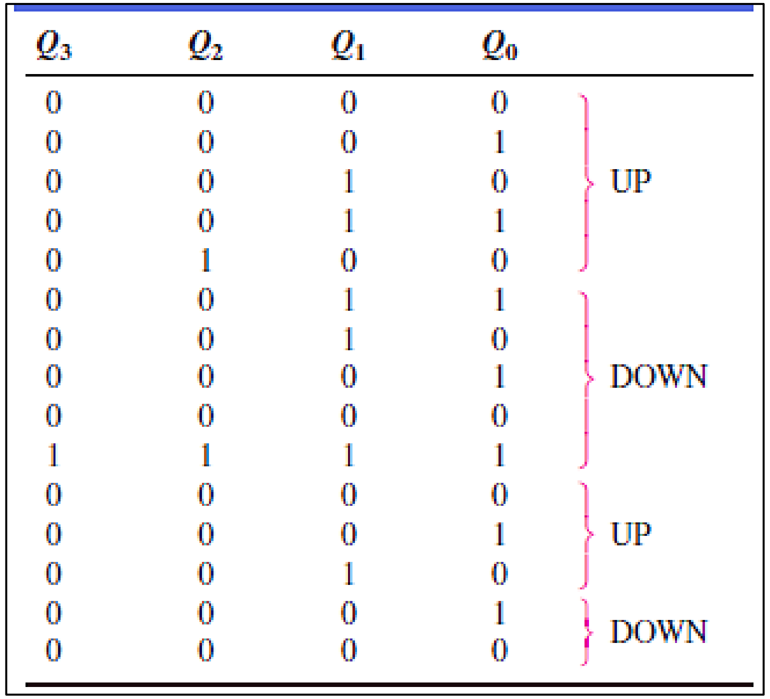
EXAMPLE 1

Show the timing diagram and determine the sequence of a 4-bit synchronous binary up/ down counter if the clock and $UP/\overline{DOWN}$ control inputs have waveforms as shown in Figure 2 (a). The counter starts in the all-0s state and is positive edge-triggered.

Solution

The timing diagram showing the Q outputs is shown in Figure 2 (b). From these waveforms, the counter sequence is as shown in Table 2.

Table 2



Q_3	Q_2	Q_1	Q_0	
0	0	0	0	UP
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	0	1	1	DOWN
0	0	1	0	
0	0	0	1	
0	0	0	0	
1	1	1	1	
0	0	0	0	UP
0	0	0	1	
0	0	1	0	
0	0	0	1	DOWN
0	0	0	0	

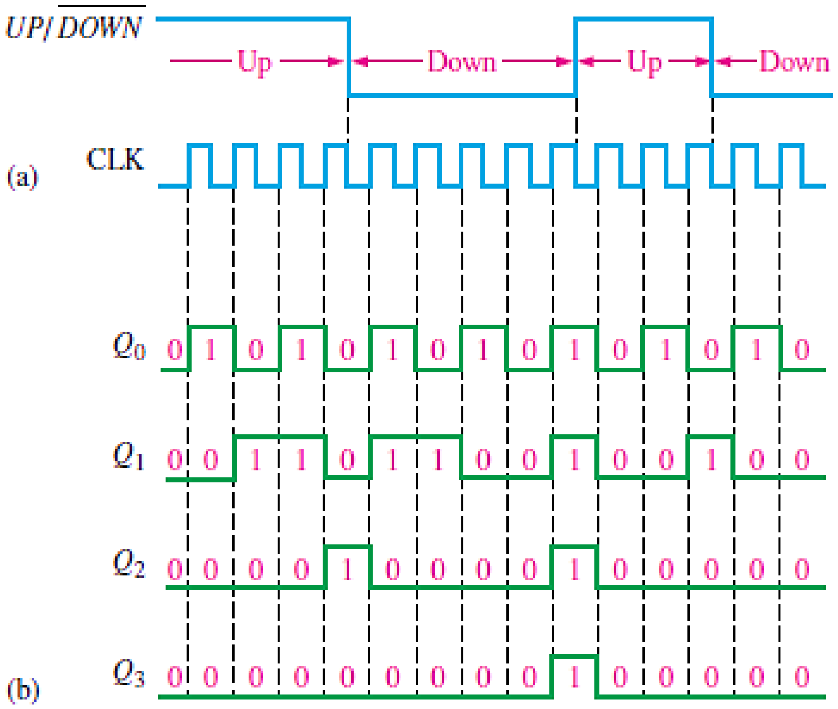


FIGURE 2

Design of Synchronous Counters

❑ There are the six steps to design a counter (state machine).

Step 1: State Diagram

- The first step in the design of a state machine (counter) is to create a state diagram. A **state diagram** shows the progression of states through which the counter advances when it is clocked.

- As an example, Figure 3 is a state diagram for a basic 3-bit Gray code counter.

- This particular circuit has no inputs other than the clock and no outputs other than the outputs taken off each flip-flop in the counter.

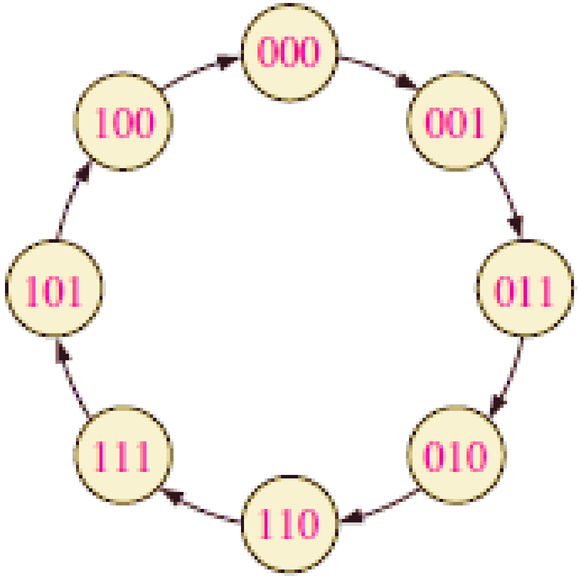


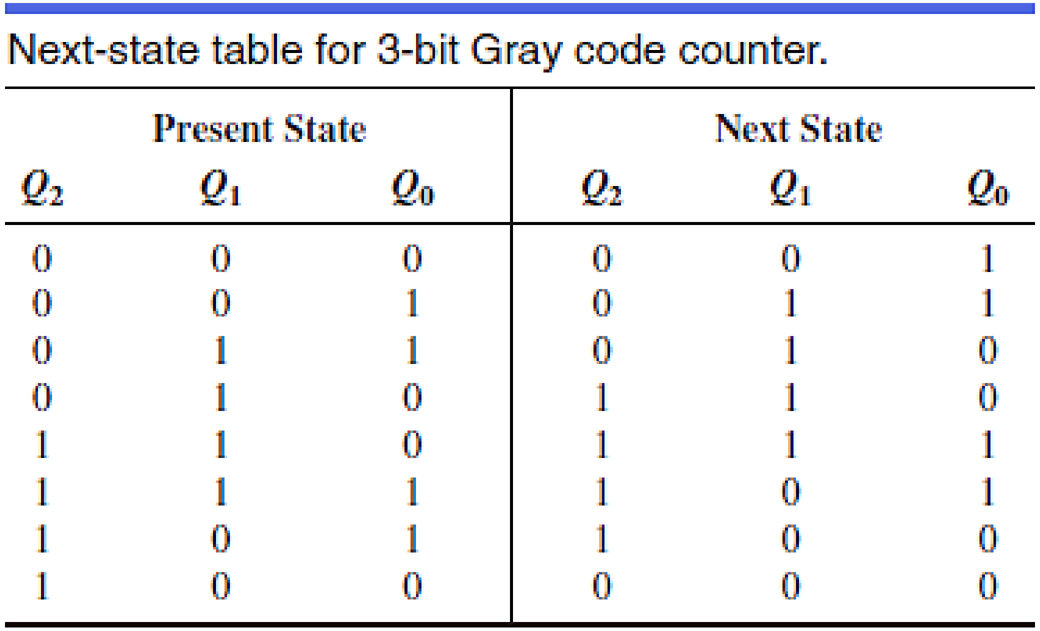
FIGURE 3 State diagram for a 3-bit Gray code counter.

Step 2: Next-State Table

- Once the sequential circuit is defined by a state diagram, the second step is to derive a next-state table, which lists each state of the counter (present state) along with the corresponding next state.
- The next state is the state that the counter goes to from its present state upon application of a clock pulse.

- The next-state table is derived from the state diagram and is shown in Table 3 for the 3-bit Gray code counter. Q0 is the least significant bit

Table 3



Next-state table for 3-bit Gray code counter.

Present State			Next State		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0
0	0	0	0	0	1
0	0	1	0	1	1
0	1	1	0	1	0
0	1	0	1	1	0
1	1	0	1	1	1
1	1	1	1	0	1
1	0	1	1	0	0
1	0	0	0	0	0

Step 3: Flip-Flop Transition Table

- Table 4 is a transition table for the J-K flip-flop. All possible output transitions are listed by showing the Q output of the flip-flop going from present states to next states.
- QN is the present state of the flip-flop (before a clock pulse) and QN + 1 is the next state (after a clock pulse).
- For each output transition, the J and K inputs that will cause the transition to occur are listed. An X indicates a “don’t care” (the input can be either a 1 or a 0).
- To design the counter, the transition table is applied to each of the flip-flops in the counter, based on the next-state table (Table 3).

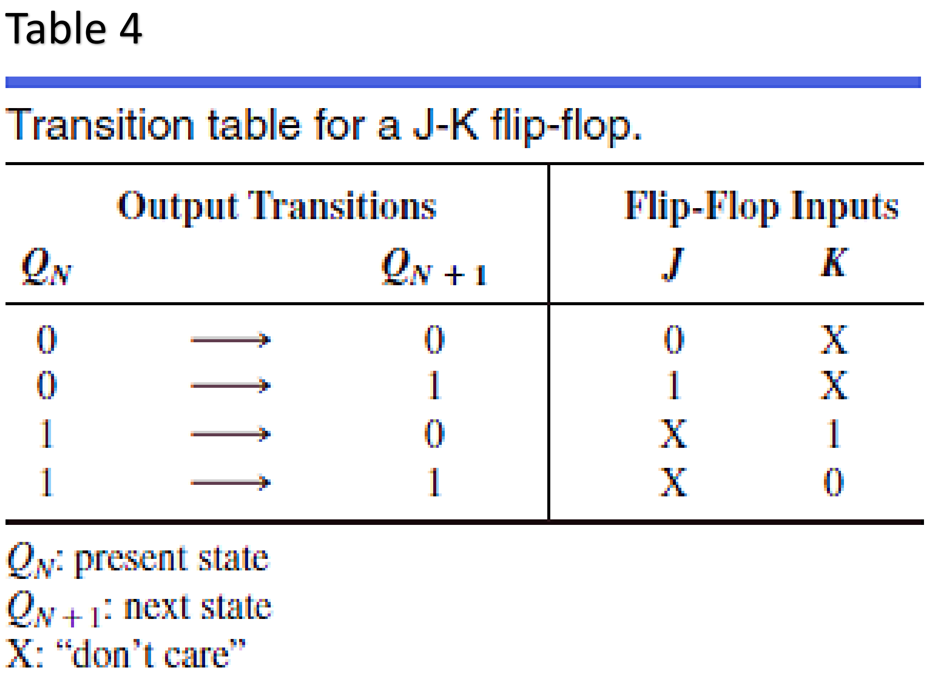


Table 4

Transition table for a J-K flip-flop.

Output Transitions			Flip-Flop Inputs	
Q_N		Q_{N+1}	J	K
0	⟶	0	0	X
0	⟶	1	1	X
1	⟶	0	X	1
1	⟶	1	X	0

Q_N: present state
Q_{N+1}: next state
X: “don’t care”

- For example, for the present state 000, Q_0 goes from a present state of 0 to a next state of 1.
- To make this happen, J0 must be a 1 and you don’t care what K0 is (J0 = 1, K0 = X), as you can see in the transition table (Table 4).
- Next, Q1 is 0 in the present state and remains a 0 in the next state. For this transition, J1 = 0 and K1 = X. Finally, Q2 is 0 in the present state and remains a 0 in the next state. Therefore, J2 = 0 and K2 = X.

Step 4: Karnaugh Maps

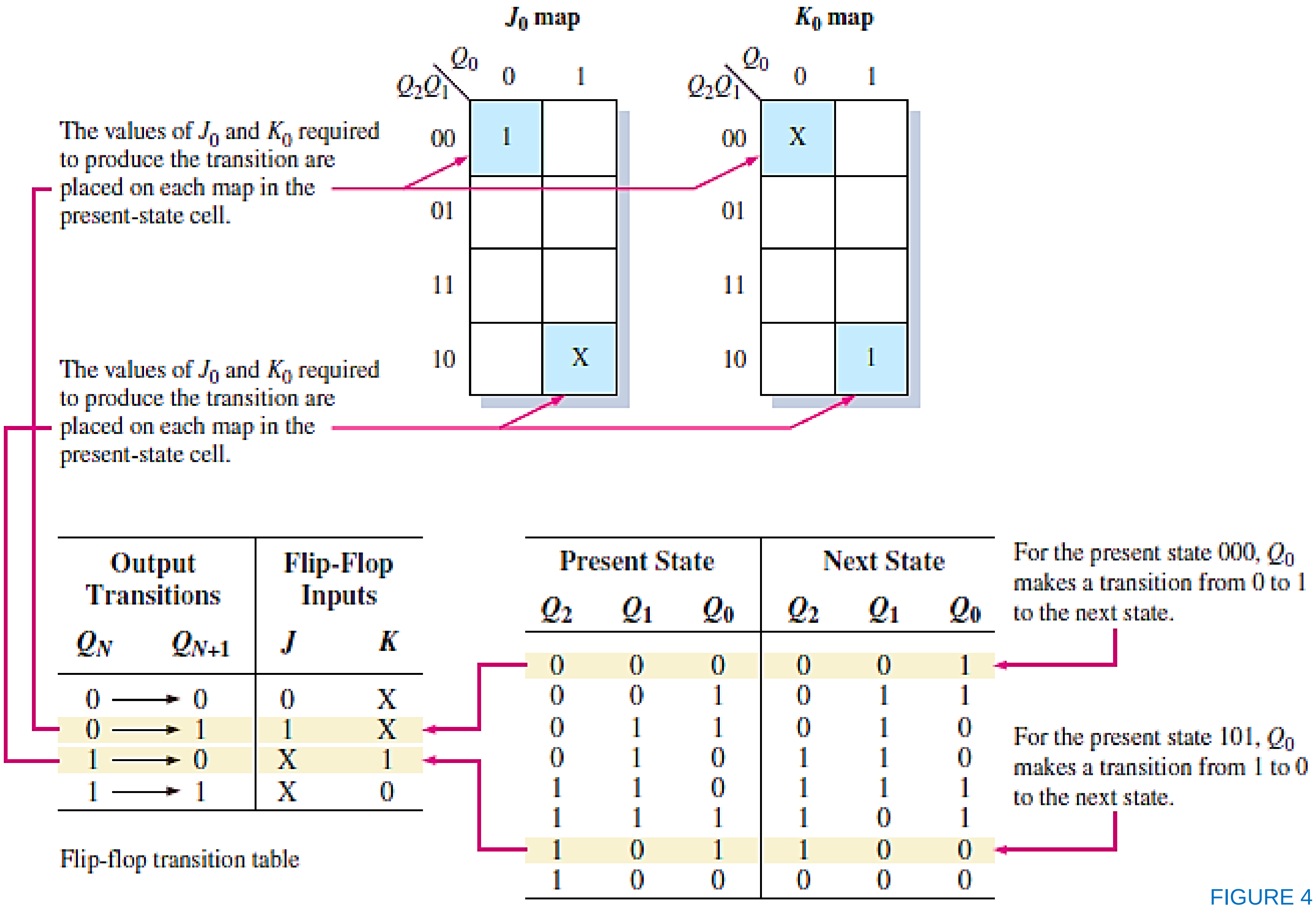
- Karnaugh maps can be used to determine the logic required for the *J* and *K* inputs of each flip-flop in the counter.

- There is a Karnaugh map for the J input and a Karnaugh map for the K input of each flip-flop.

- In this design procedure, each cell in a Karnaugh map represents one of the present states in the counter sequence listed in Table 3.

- From the J and K states in the transition table (Table 4) a 1, 0, or X is entered into each present-state cell on the maps depending on the transition of the Q output for a particular flip-flop.

- To illustrate this procedure, two sample entries are shown for the J0 and the K0 inputs to the least significant flip-flop (Q0) in Figure 4.



J_0 map

Q_2Q_1 \ Q_0	0	1
00	1	
01		
11		
10		X

K_0 map

Q_2Q_1 \ Q_0	0	1
00	X	
01		
11		
10		1

The values of J_0 and K_0 required to produce the transition are placed on each map in the present-state cell.

The values of J_0 and K_0 required to produce the transition are placed on each map in the present-state cell.

Output Transitions		Flip-Flop Inputs	
Q_N	Q_{N+1}	J	K
0 ⟶	0	0	X
0 ⟶	1	1	X
1 ⟶	0	X	1
1 ⟶	1	X	0

Flip-flop transition table

Present State			Next State		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0
0	0	0	0	0	1
0	0	1	0	1	1
0	1	1	0	1	0
0	1	0	1	1	0
1	1	0	1	1	1
1	1	1	1	0	1
1	0	1	1	0	0
1	0	0	0	0	0

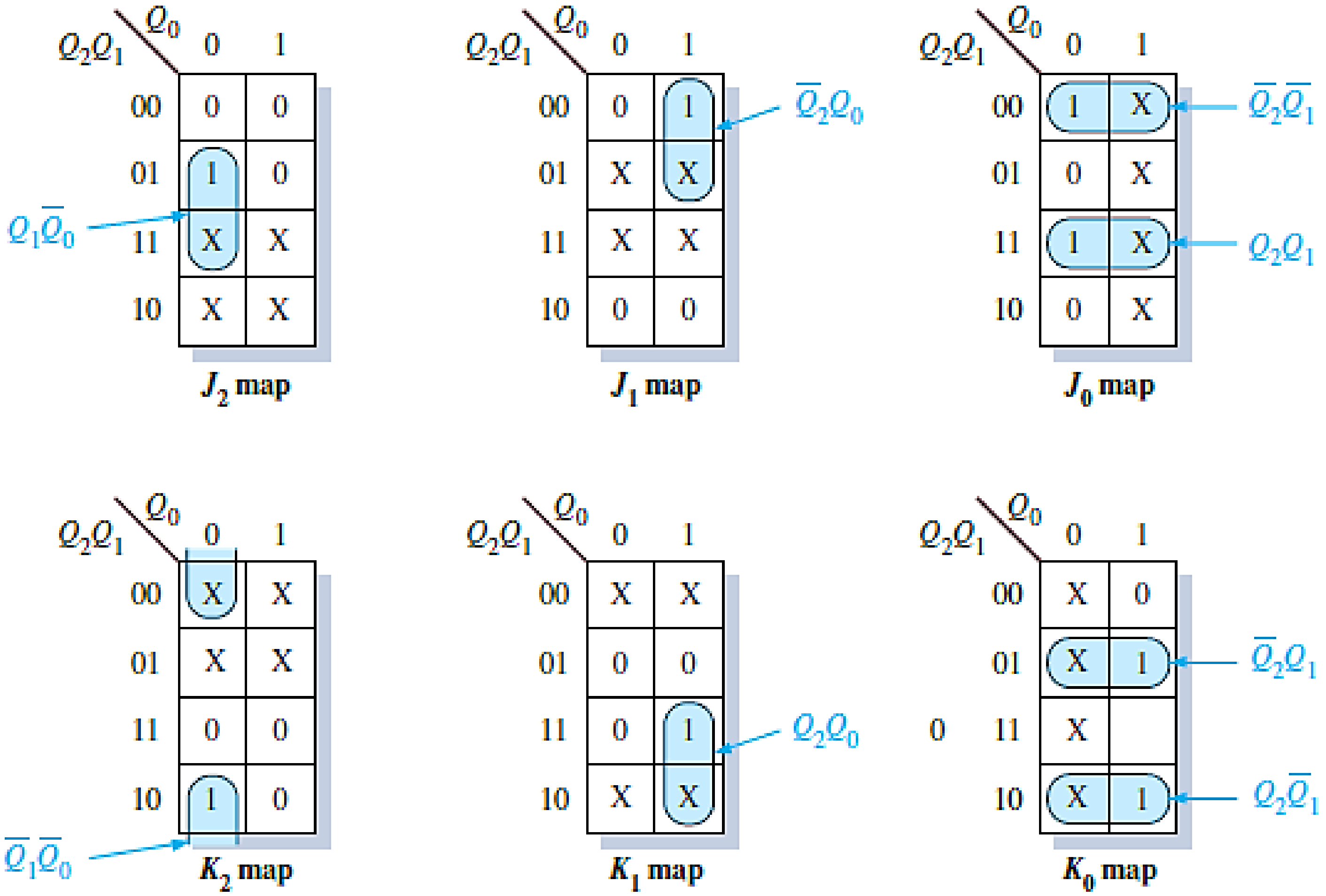
Next-state table

For the present state 000, Q_0 makes a transition from 0 to 1 to the next state.

For the present state 101, Q_0 makes a transition from 1 to 0 to the next state.

FIGURE 4 Examples of the mapping procedure for the counter sequence represented in Table 3 and Table 4.

- The completed Karnaugh maps for all three flip-flops in the counter are shown in Figure 5 . The cells are grouped as indicated and the corresponding Boolean expressions for each group are derived.



Q_2Q_1 \ Q_0	0	1
00	0	0
01	1	0
11	X	X
10	X	X

J_2 **map**

Q_2Q_1 \ Q_0	0	1
00	0	1
01	X	X
11	X	X
10	0	0

J_1 **map**

Q_2Q_1 \ Q_0	0	1
00	1	X
01	0	X
11	1	X
10	0	X

J_0 **map**

Q_2Q_1 \ Q_0	0	1
00	X	X
01	X	X
11	0	0
10	1	0

K_2 **map**

Q_2Q_1 \ Q_0	0	1
00	X	X
01	0	0
11	0	1
10	X	X

K_1 **map**

Q_2Q_1 \ Q_0	0	1
00	X	0
01	X	1
11	X	
10	X	1

K_0 **map**

FIGURE 5 Karnaugh maps for present-state *J* and *K* inputs.

Step 5: Logic Expressions for Flip-Flop Inputs

- From the Karnaugh maps of Figure 5 you obtain the following expressions for the *J* and *K* inputs of each flip-flop:

$$
\begin{aligned}
J_0 &= Q_2Q_1 + \overline{Q}_2\overline{Q}_1 = \overline{Q_2 \oplus Q_1}\\
K_0 &= Q_2\overline{Q}_1 + \overline{Q}_2Q_1 = Q_2 \oplus Q_1\\
J_1 &= \overline{Q}_2Q_0\\
K_1 &= Q_2Q_0\\
J_2 &= Q_1\overline{Q}_0\\
K_2 &= \overline{Q}_1\overline{Q}_0
\end{aligned}
$$

Step 6: Counter Implementation

- The final step is to implement the combinational logic from the expressions for the J and K inputs and connect the flip-flops to form the complete 3-bit Gray code counter as shown in Figure 6.

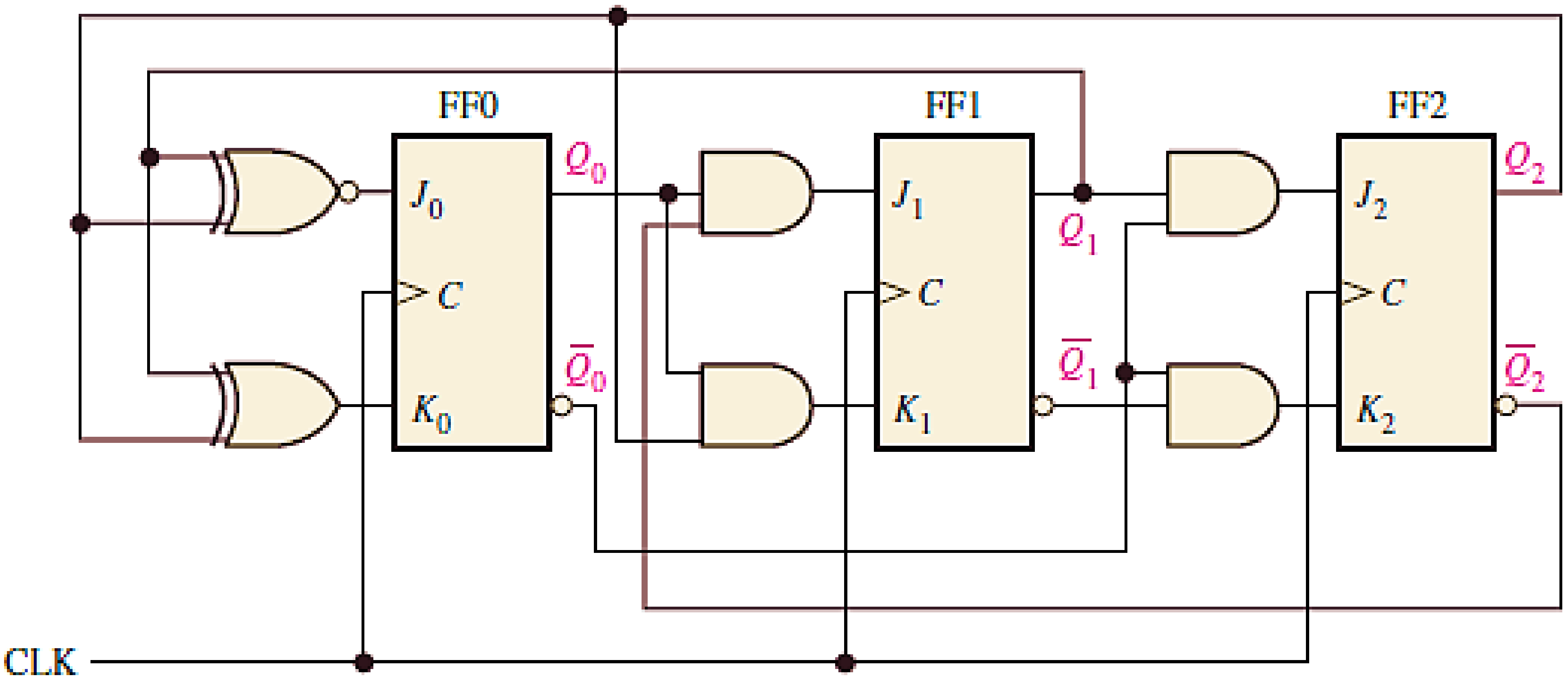


FIGURE 6 Three-bit Gray code counter.

A summary of steps used in the design of the 3-bit Gray code counter follows. In general, these steps can be applied to any state machine.

1. Specify the counter sequence and draw a state diagram.
2. Derive a next-state table from the state diagram.
3. Develop a transition table showing the flip-flop inputs required for each transition. The transition table is always the same for a given type of flip-flop.
4. Transfer the *J* and *K* states from the transition table to Karnaugh maps. There is a Karnaugh map for each input of each flip-flop.
5. Group the Karnaugh map cells to generate and derive the logic expression for each flip-flop input.
6. Implement the expressions with combinational logic, and combine with the flip-flops to create the counter.

This procedure is now applied to the design of other synchronous counters in next Examples.

EXAMPLE 2

Design a counter with the irregular binary count sequence shown in the state diagram of Figure 7. Use D flip-flops.

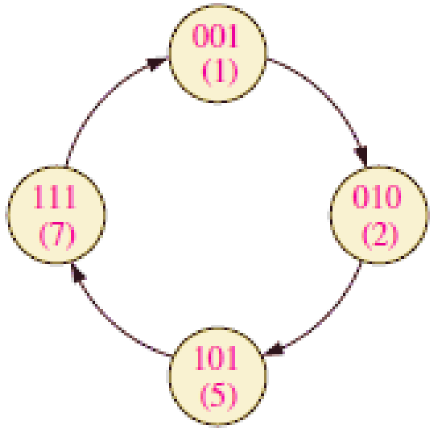


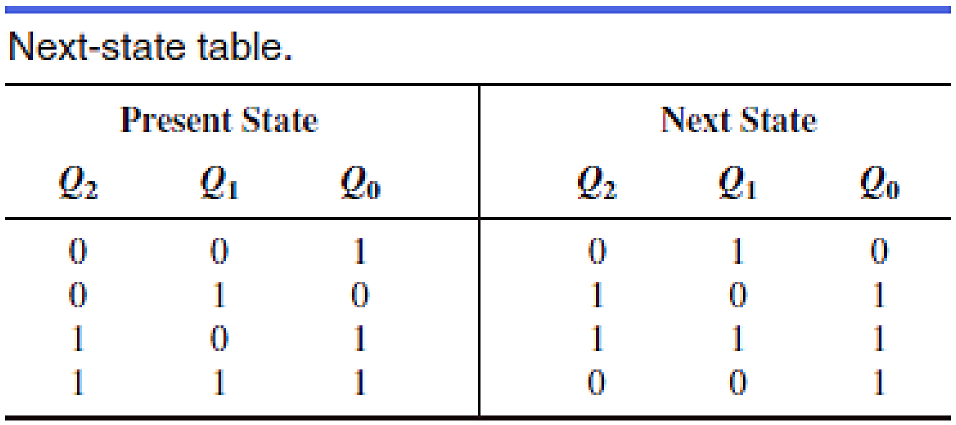
FIGURE 7

Solution

Step 1: The state diagram is as shown. Although there are only four states, a 3-bit counter is required to implement this sequence because the maximum binary count is seven.
Since the required sequence does not include all the possible binary states, the invalid states (0, 3, 4, and 6) can be treated as "don't cares" in the design. However, if the counter should erroneously get into an invalid state, you must make sure that it goes back to a valid state.

Step 2: The next-state table is developed from the state diagram and is given in Table 5.

Table 5

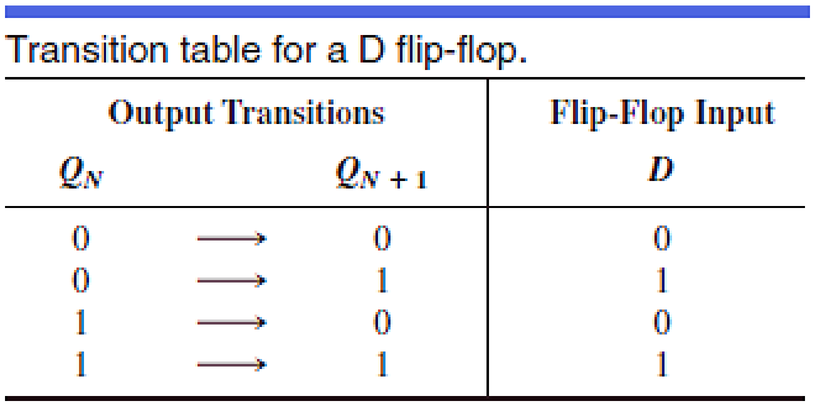


Next-state table.

Present State			Next State		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0
0	0	1	0	1	0
0	1	0	1	0	1
1	0	1	1	1	1
1	1	1	0	0	1

Step 3: The transition table for the D flip-flop is shown in Table 6.

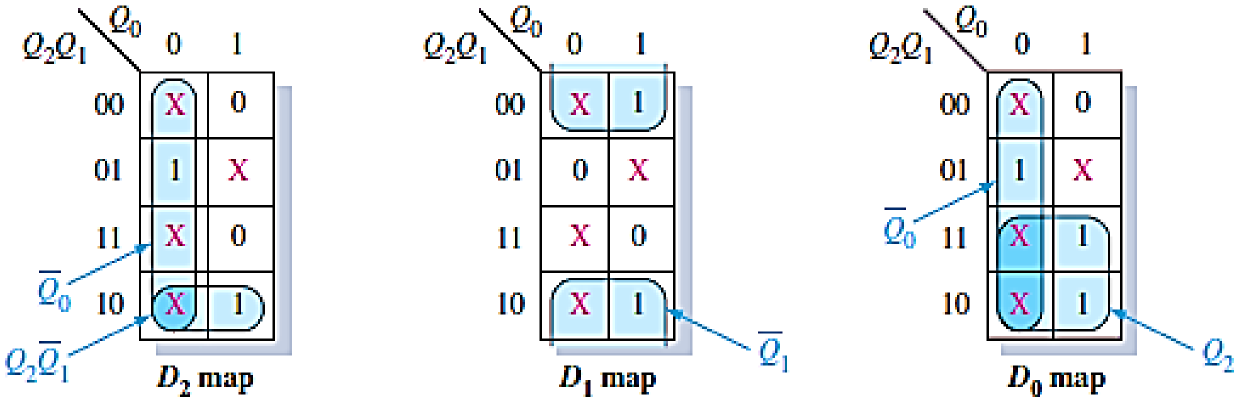
Table 6



Transition table for a D flip-flop.

Output Transitions			Flip-Flop Input
Q_N		Q_{N+1}	D
0	⟶	0	0
0	⟶	1	1
1	⟶	0	0
1	⟶	1	1

Step 4: The *D* inputs are plotted on the present-state Karnaugh maps in Figure 8. Also “don’t cares” can be placed in the cells corresponding to the invalid states of 000, 011, 100, and 110, as indicated by the red Xs.



$Q_2Q_1 \backslash Q_0$	0	1
00	X	0
01	1	X
11	X	0
10	X	1

D_2 map

$Q_2Q_1 \backslash Q_0$	0	1
00	X	1
01	0	X
11	X	0
10	X	1

D_1 map

$Q_2Q_1 \backslash Q_0$	0	1
00	X	0
01	1	X
11	X	1
10	X	1

D_0 map

Figure 8

Step 5: Group the 1s, taking advantage of as many of the “don’t care” states as possible for maximum simplification, as shown in Figure 8. The expression for each D input taken from the maps is as follows:

$$D_0 = \overline{Q}_0 + Q_2$$
$$D_1 = \overline{Q}_1$$
$$D_2 = \overline{Q}_0 + Q_2\overline{Q}_1$$

Step 6: The implementation of the counter is shown in Figure 9.

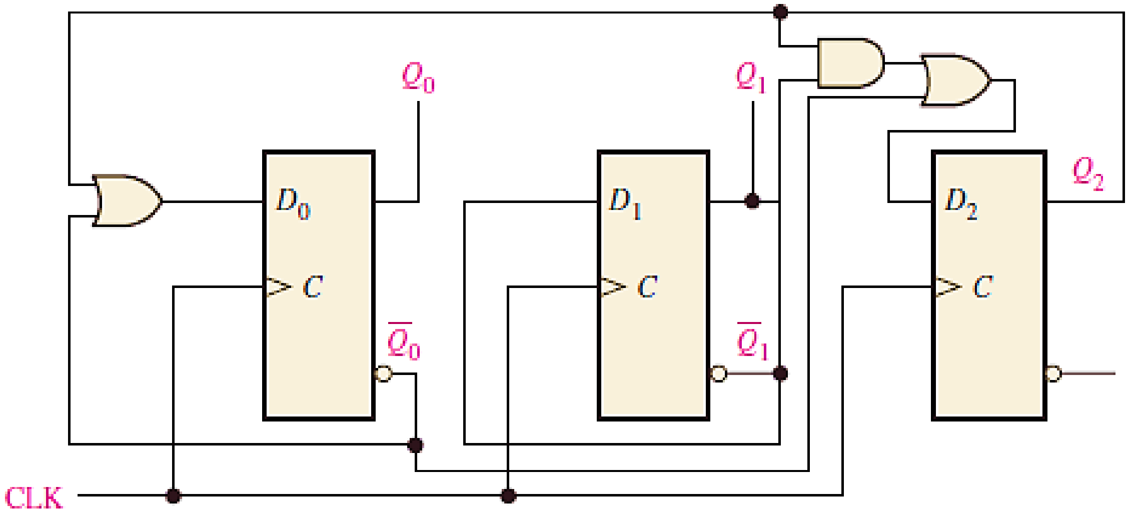


Figure 9

An analysis shows that if the counter, by accident, gets into one of the invalid states (0, 3, 4, 6), it will always return to a valid state according to the following sequences:
0 → 3 → 4 → 7, and 6 → 1.

EXAMPLE 3

Develop a synchronous 3-bit up/down counter with a Gray code sequence using J-K flip-flops. The counter should count up when an UP/$\overline{\text{DOWN}}$ control input is 1 and count down when the control input is 0.

Solution

Step 1: The state diagram is shown in Figure 10. The 1 or 0 beside each arrow indicates the state of the UP/$\overline{\text{DOWN}}$ control input, *Y.*

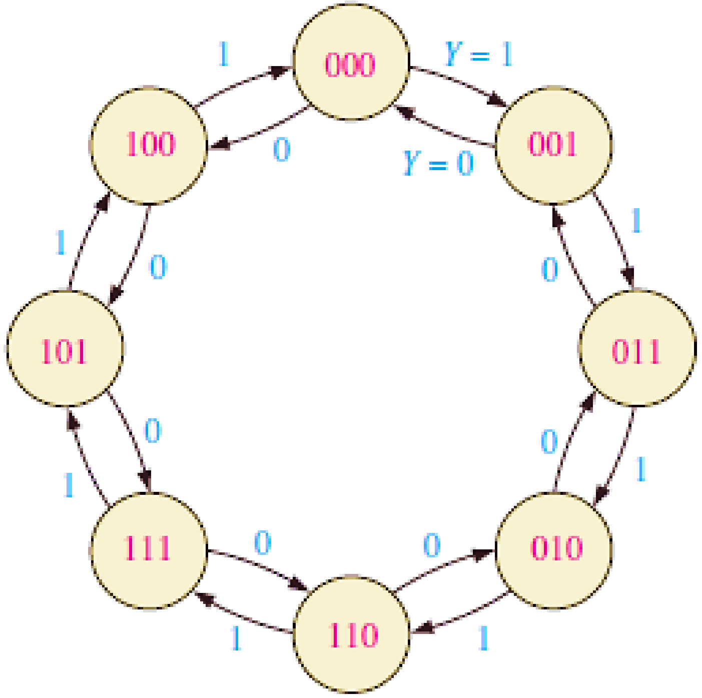
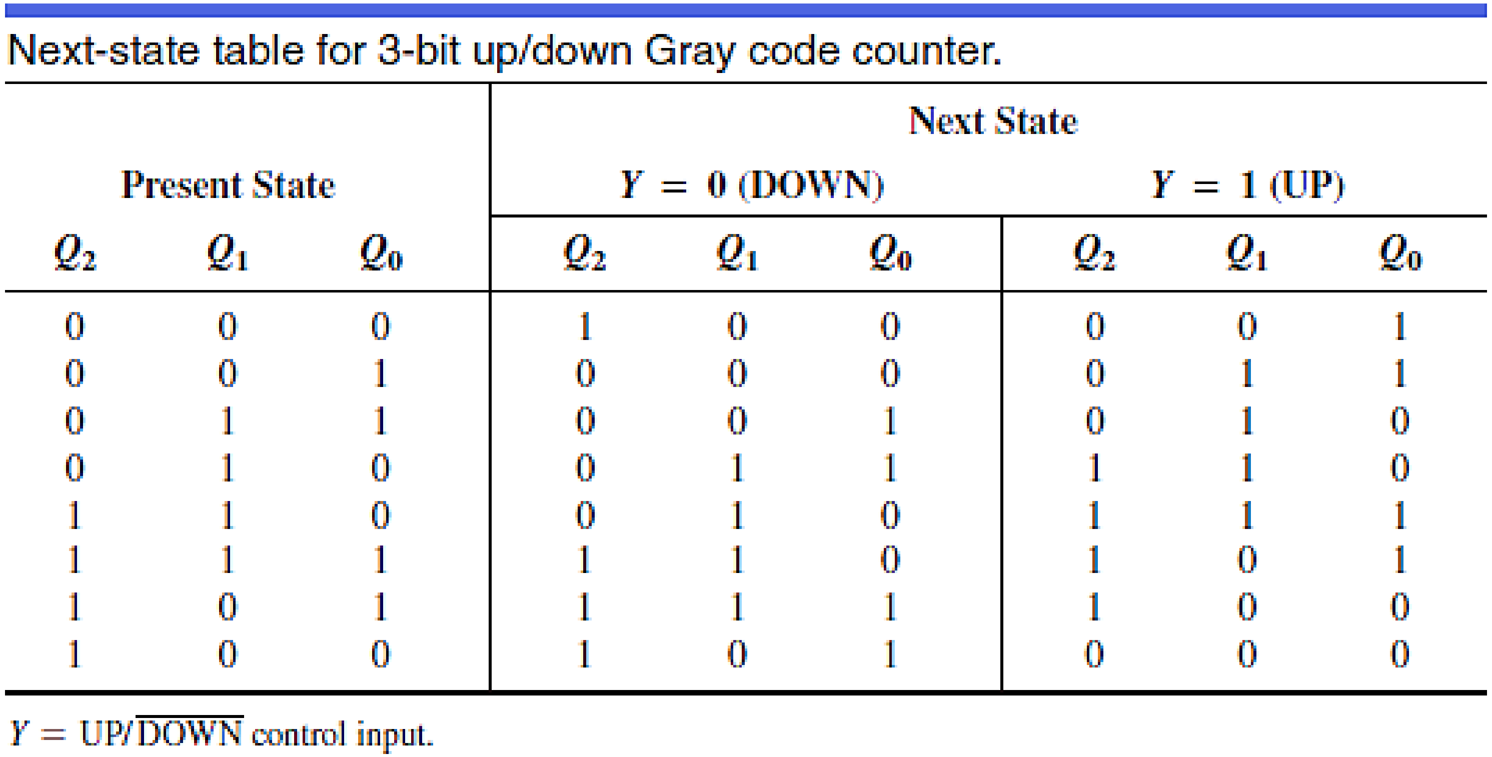


FIGURE 10 State diagram for a 3-bit up/down Gray code counter.

Step 2: The next-state table is derived from the state diagram and is shown in Table 7. Notice that for each present state there are two possible next states, depending on the UP/$\overline{\text{DOWN}}$ control variable, *Y.*

Table 7



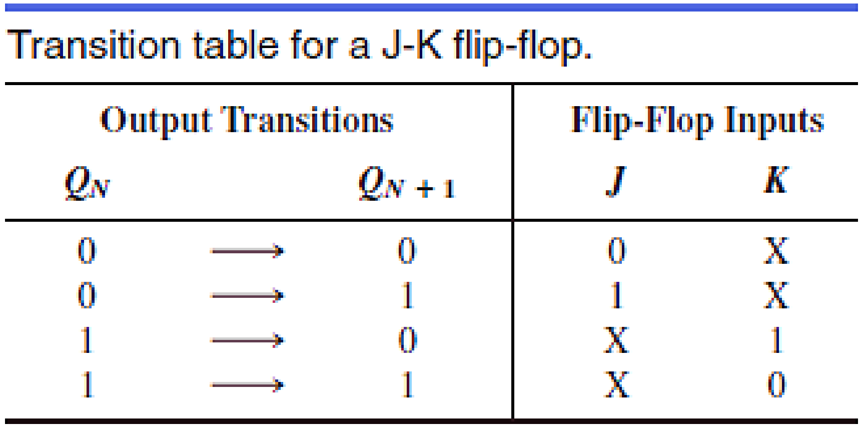
Next-state table for 3-bit up/down Gray code counter.

Present State			Next State					
			$Y = 0$ (DOWN)			$Y = 1$ (UP)		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0	Q_2	Q_1	Q_0
0	0	0	1	0	0	0	0	1
0	0	1	0	0	0	0	1	1
0	1	1	0	0	1	0	1	0
0	1	0	0	1	1	1	1	0
1	1	0	0	1	0	1	1	1
1	1	1	1	1	0	1	0	1
1	0	1	1	1	1	1	0	0
1	0	0	1	0	1	0	0	0

Y = UP/$\overline{\text{DOWN}}$ control input.

Step 3: The transition table for the J-K flip-flops is repeated in Table 8.

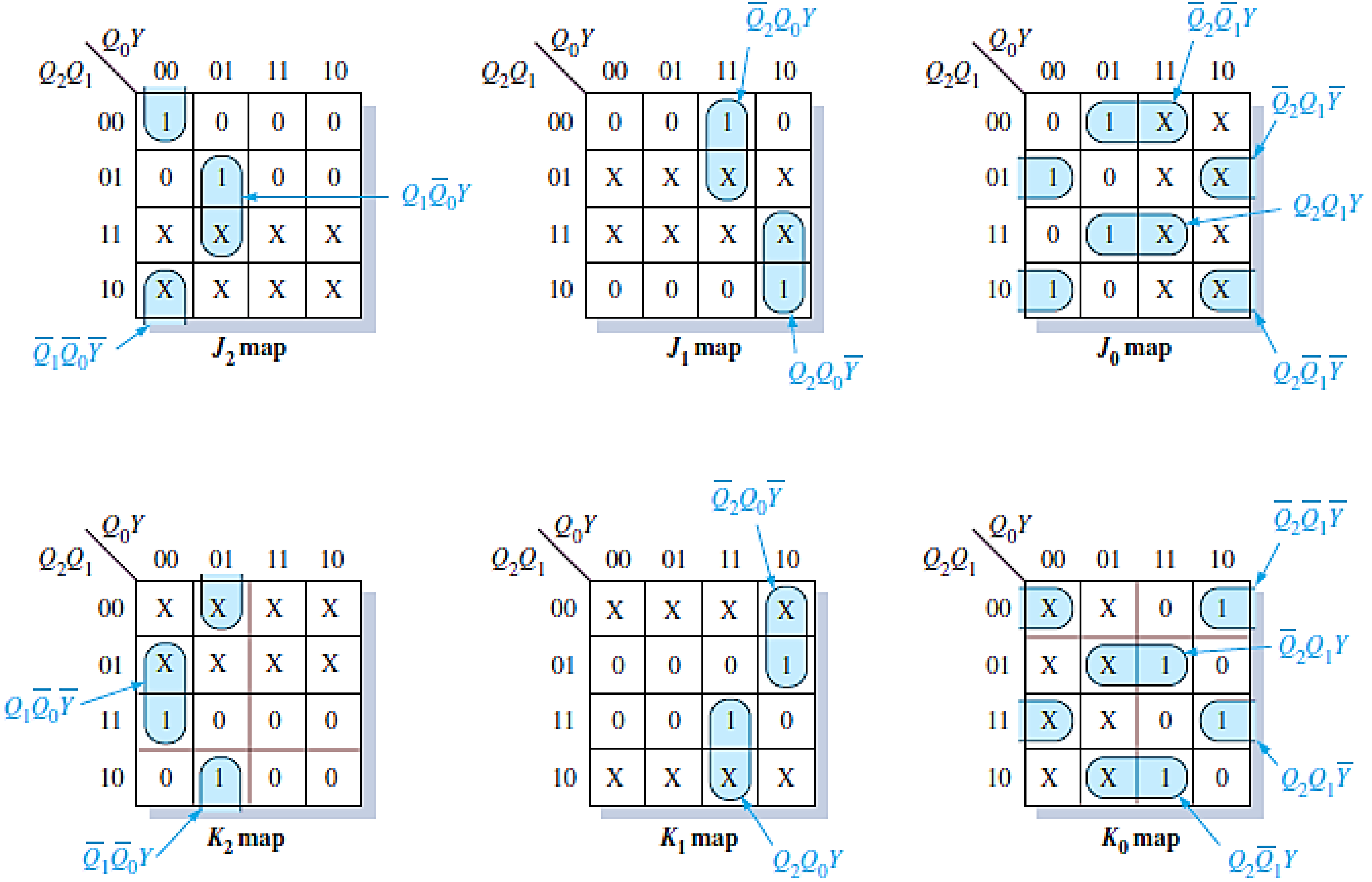
Table 8



Transition table for a J-K flip-flop.

Output Transitions			Flip-Flop Inputs	
Q_N		Q_{N+1}	J	K
0	⟶	0	0	X
0	⟶	1	1	X
1	⟶	0	X	1
1	⟶	1	X	0

Step 4: The Karnaugh maps for the *J* and *K* inputs of the flip-flops are shown in Figure 11. The UP/$\overline{\text{DOWN}}$ control input, *Y*, is considered one of the state variables along with Q_0, Q_1, and Q_2. Using the next-state table, the information in the "Flip-Flop Inputs" column of Table 8 is transferred onto the maps as indicated for each present state of the counter.


$\overline{Q}_1\overline{Q}_0\overline{Y}$
$Q_1\overline{Q}_0Y$
$\overline{Q}_2Q_0Y$
$Q_2Q_0\overline{Y}$
$\overline{Q}_2\overline{Q}_1Y$
$\overline{Q}_2Q_1\overline{Y}$
Q_2Q_1Y
$Q_2\overline{Q}_1\overline{Y}$
$Q_1\overline{Q}_0\overline{Y}$
$\overline{Q}_1\overline{Q}_0Y$
$\overline{Q}_2Q_0\overline{Y}$
Q_2Q_0Y
$\overline{Q}_2\overline{Q}_1\overline{Y}$
$\overline{Q}_2Q_1Y$
$Q_2Q_1\overline{Y}$
$Q_2\overline{Q}_1Y$

J_2 map

Q_2Q_1 \ Q_0Y	00	01	11	10
00	1	0	0	0
01	0	1	0	0
11	X	X	X	X
10	X	X	X	X

J_1 map

Q_2Q_1 \ Q_0Y	00	01	11	10
00	0	0	1	0
01	X	X	X	X
11	X	X	X	X
10	0	0	0	1

J_0 map

Q_2Q_1 \ Q_0Y	00	01	11	10
00	0	1	X	X
01	1	0	X	X
11	0	1	X	X
10	1	0	X	X

K_2 map

Q_2Q_1 \ Q_0Y	00	01	11	10
00	X	X	X	X
01	X	X	X	X
11	1	0	0	0
10	0	1	0	0

K_1 map

Q_2Q_1 \ Q_0Y	00	01	11	10
00	X	X	X	X
01	0	0	0	1
11	0	0	1	0
10	X	X	X	X

K_0 map

Q_2Q_1 \ Q_0Y	00	01	11	10
00	X	X	0	1
01	X	X	1	0
11	X	X	0	1
10	X	X	1	0

Figure 11 *J* and *K* maps for Table 8. The UP/$\overline{\text{DOWN}}$ control input, *Y*, is treated as a fourth variable.

Step 5: The 1s are combined in the largest possible groupings, with “don’t cares” (Xs) used where possible. The groups are factored, and the expressions for the *J* and *K* inputs are as follows:

$$
\begin{aligned}
J_0 &= Q_2Q_1Y + Q_2\overline{Q}_1\overline{Y} + \overline{Q}_2\overline{Q}_1Y + \overline{Q}_2Q_1\overline{Y} & \qquad K_0 &= \overline{Q}_2\overline{Q}_1\overline{Y} + \overline{Q}_2Q_1Y + Q_2\overline{Q}_1Y + Q_2Q_1\overline{Y} \\
J_1 &= \overline{Q}_2Q_0Y + Q_2Q_0\overline{Y} & K_1 &= \overline{Q}_2Q_0\overline{Y} + Q_2Q_0Y \\
J_2 &= Q_1\overline{Q}_0Y + \overline{Q}_1\overline{Q}_0\overline{Y} & K_2 &= Q_1\overline{Q}_0\overline{Y} + \overline{Q}_1\overline{Q}_0Y
\end{aligned}
$$

Step 6: The *J* and *K* equations are implemented with combinational logic.