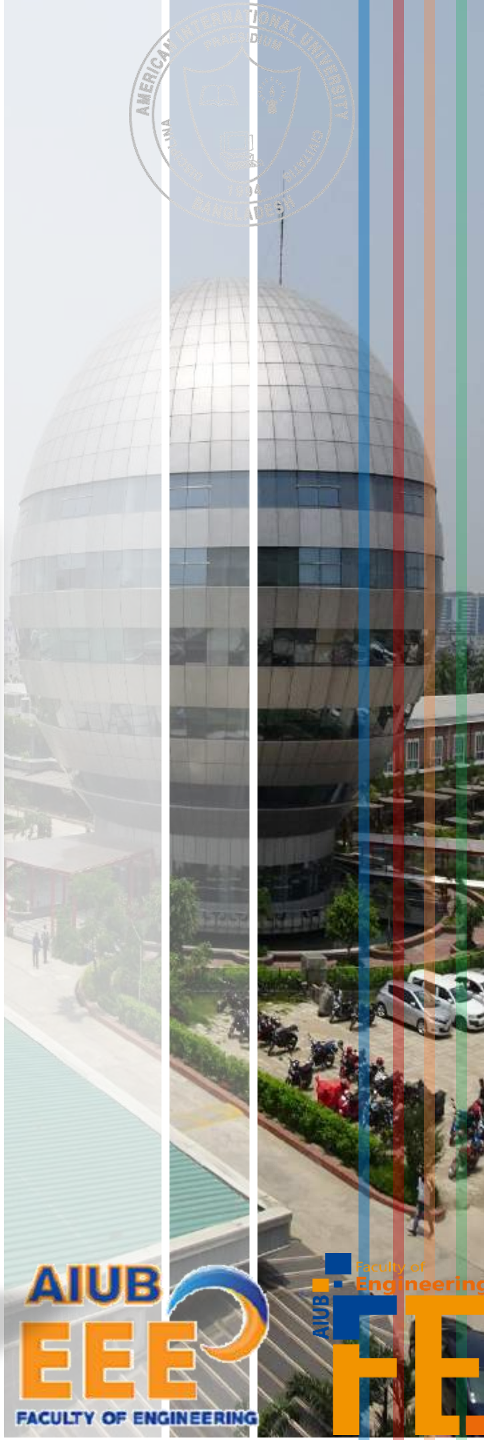
EEE 3101: Digital Logic and Circuits

Counters

Course Teacher: Nafiz Ahmed Chisty

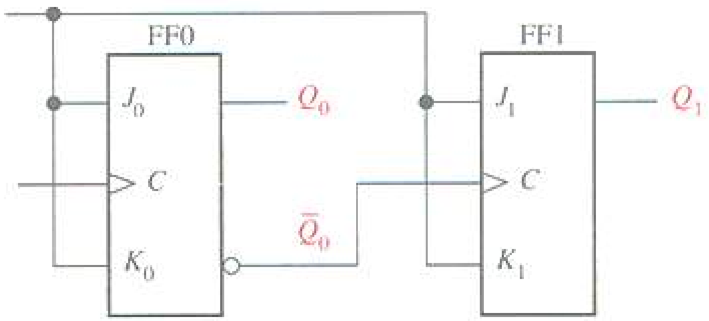
Head, Department of EEE
Associate Professor, Department of EEE & CoE
Faculty of Engineering
Room# D0105, D Building
Email: chisty@aiub.edu
Website: http://engg.aiub.edu/faculties/nafiz
Website: www.nachisty.com



Counters

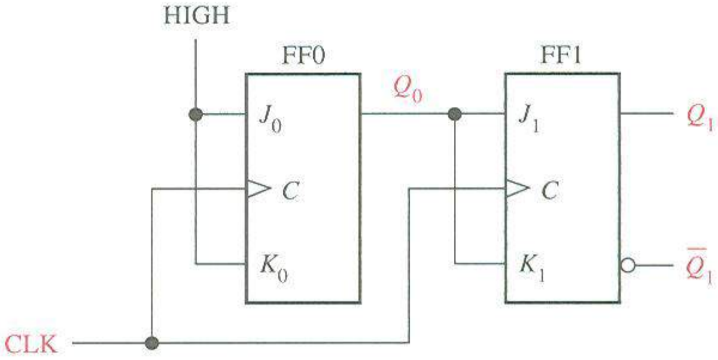
Asynchronous Counter

Flip-flops(FF) **do not** have a **common clock pulse**



Synchronous Counter

Flip-flops (FF) have a **common clock pulse**



Asynchronous Counter

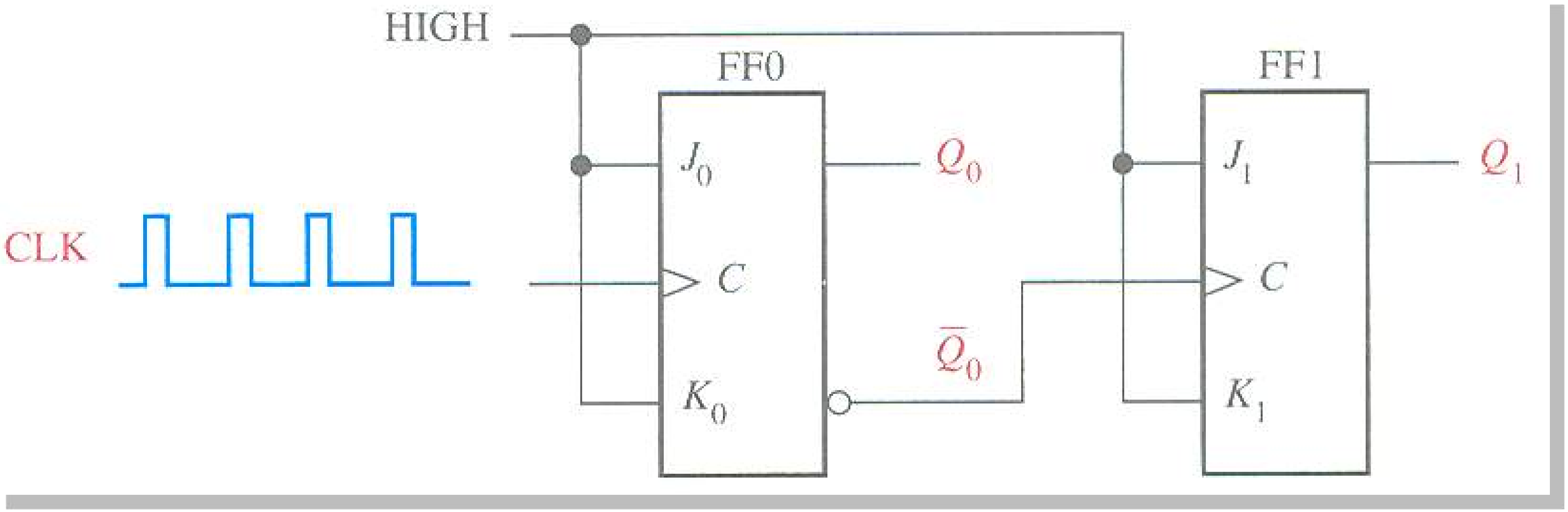
An asynchronous counter is one in which the flip-flops(FF) within the counter do not change states at exactly the same time because they do not have a **common clock pulse**

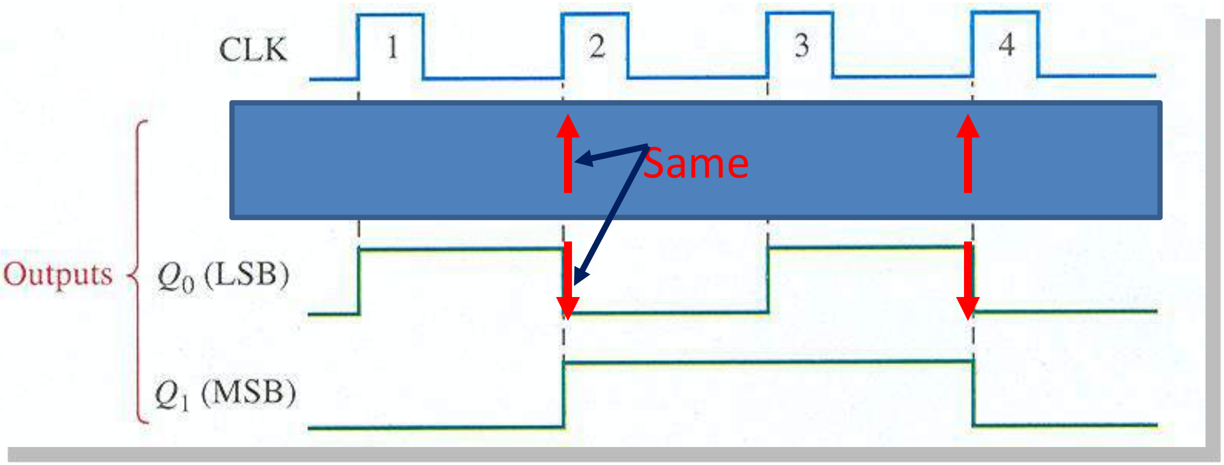
Asynchronous Counter Operation

- Asynchronous binary counter
  - 2-bit asynchronous binary counter
  - 3-bit asynchronous binary counter
  - 4-bit asynchronous binary counter

- Asynchronous decade counter
- Asynchronous Modulus Twelve counter

2-bit asynchronous binary counter

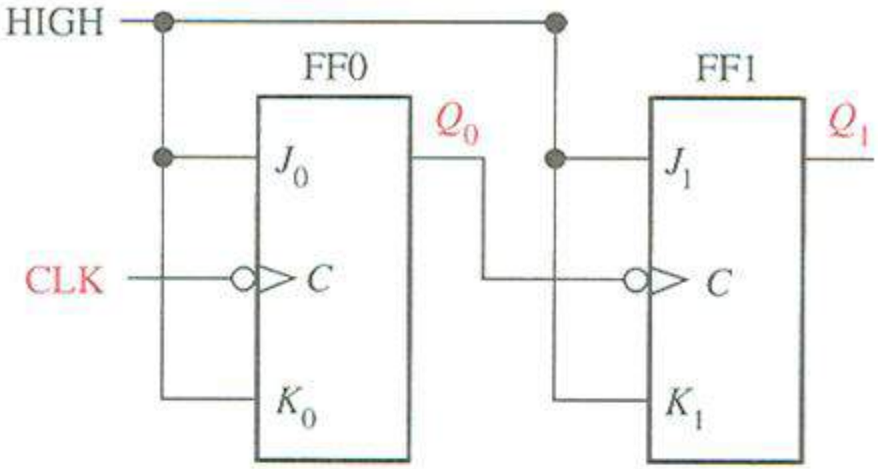


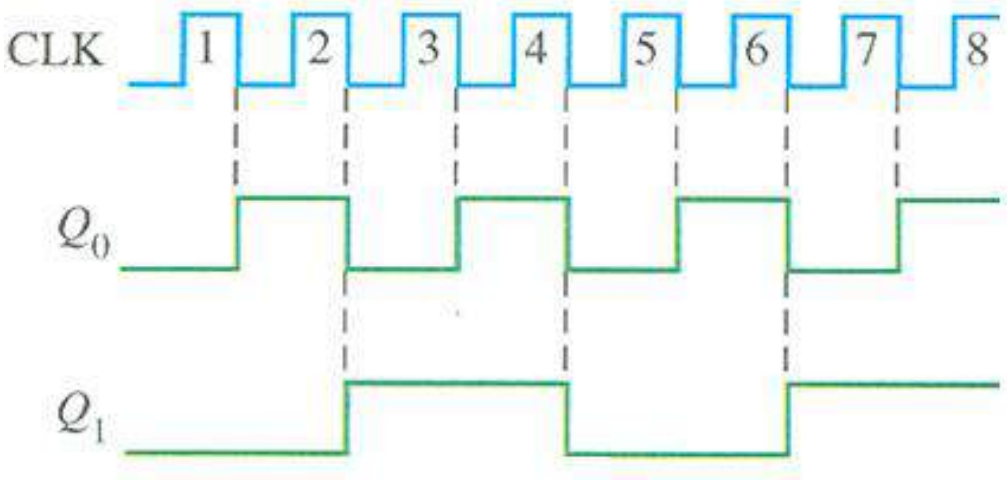


Same

CLOCK PULSE	Q_1	Q_0
Initially	0	0
1	0	1
2	1	0
3	1	1
4 (recycles)	0	0

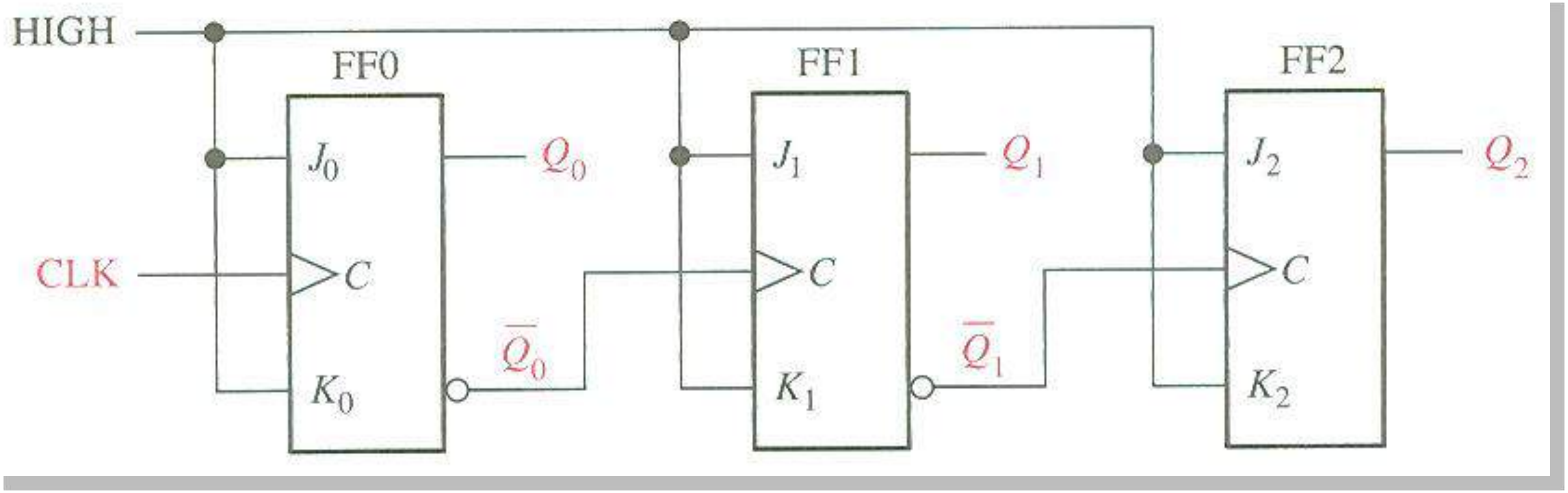
2-bit asynchronous binary counter

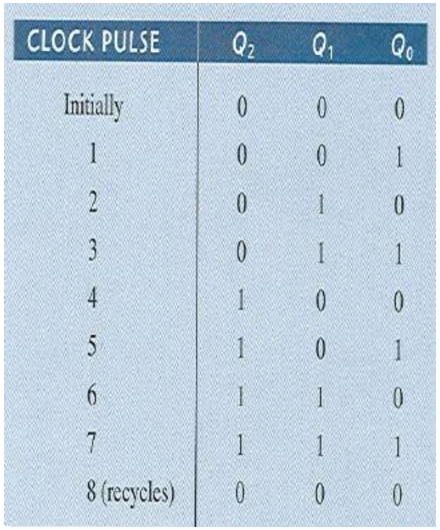




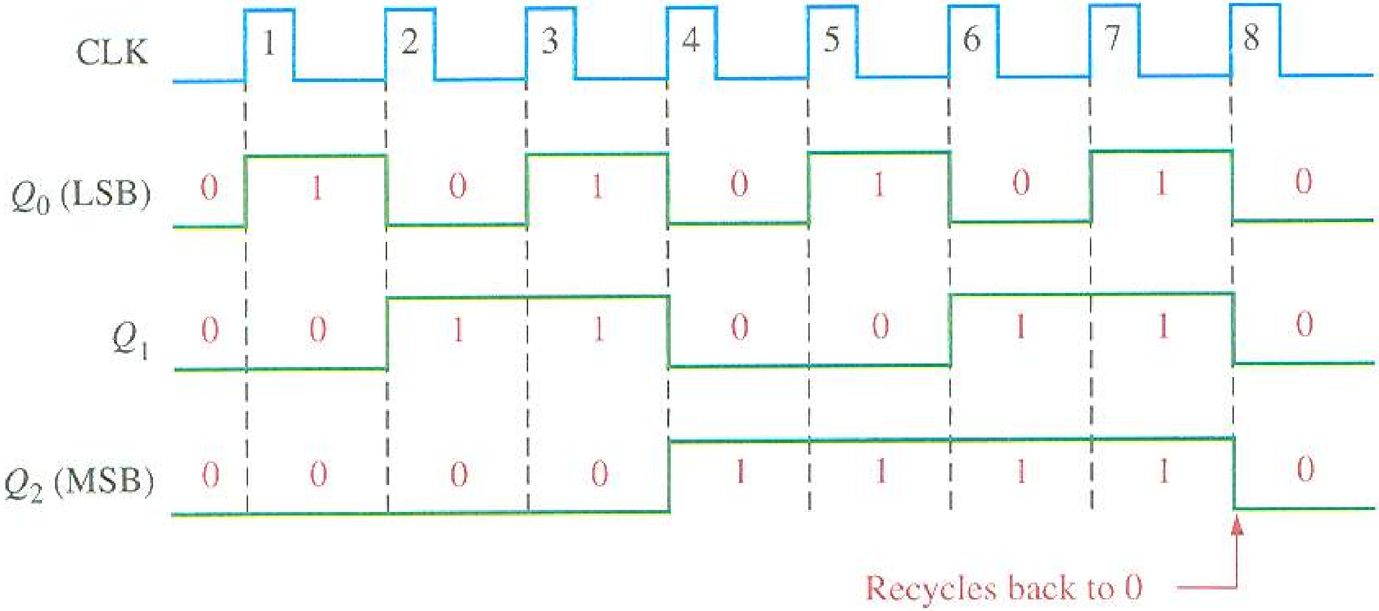
CLOCK PULSE	Q_1	Q_0
Initially	0	0
1	0	1
2	1	0
3	1	1
4 (recycles)	0	0

3-bit asynchronous binary counter

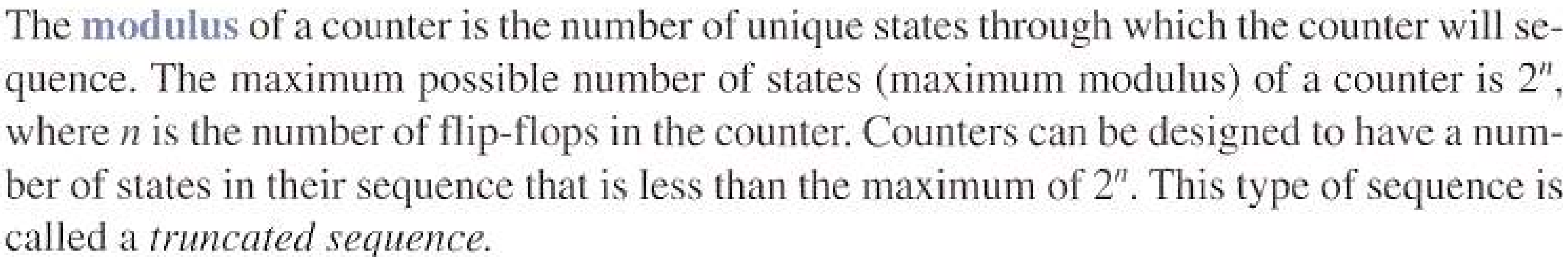




CLOCK PULSE	Q_2	Q_1	Q_0
Initially	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8 (recycles)	0	0	0

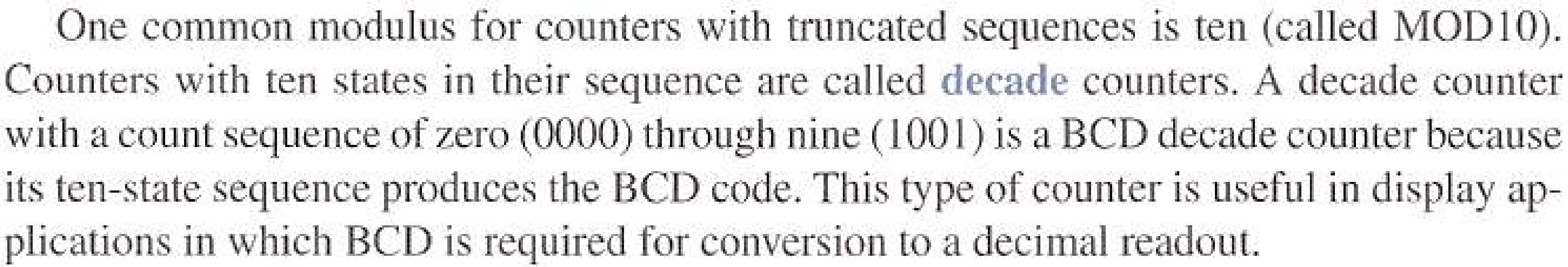


Modulus of a Counter

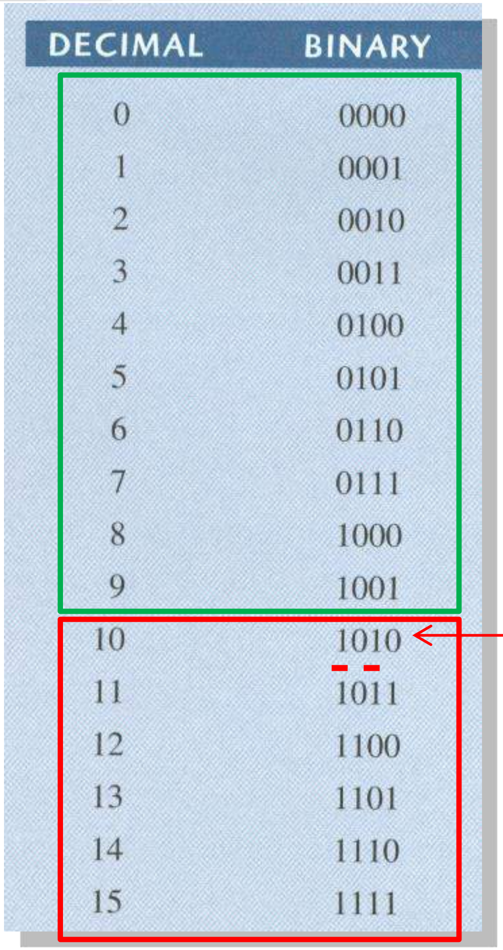


The **modulus** of a counter is the number of unique states through which the counter will sequence. The maximum possible number of states (maximum modulus) of a counter is 2^n, where n is the number of flip-flops in the counter. Counters can be designed to have a number of states in their sequence that is less than the maximum of 2^n. This type of sequence is called a *truncated sequence*.

Asynchronous decade (MOD-10) counter



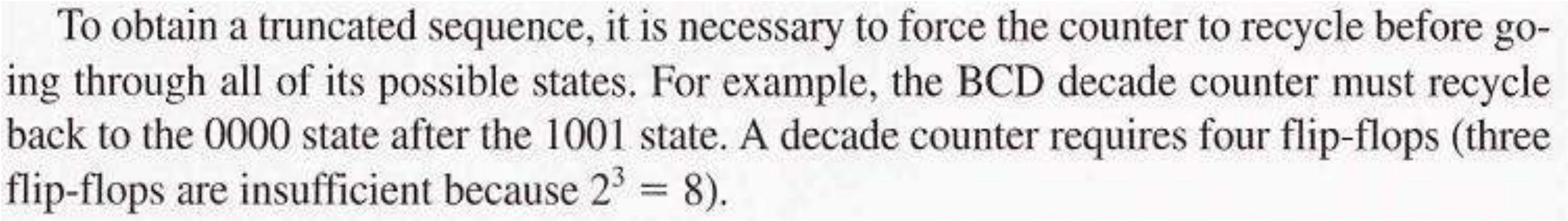
One common modulus for counters with truncated sequences is ten (called MOD10). Counters with ten states in their sequence are called **decade** counters. A decade counter with a count sequence of zero (0000) through nine (1001) is a BCD decade counter because its ten-state sequence produces the BCD code. This type of counter is useful in display applications in which BCD is required for conversion to a decimal readout.



DECIMAL	BINARY
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

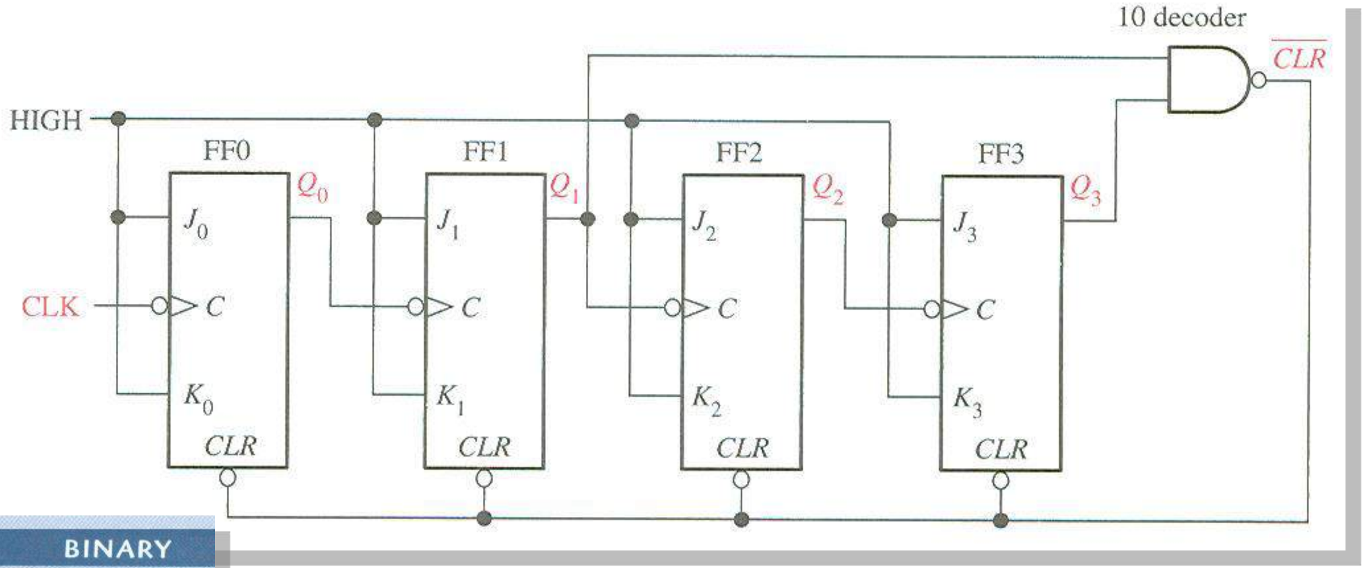
Not wanted. We want 0000.

So Decode this state.

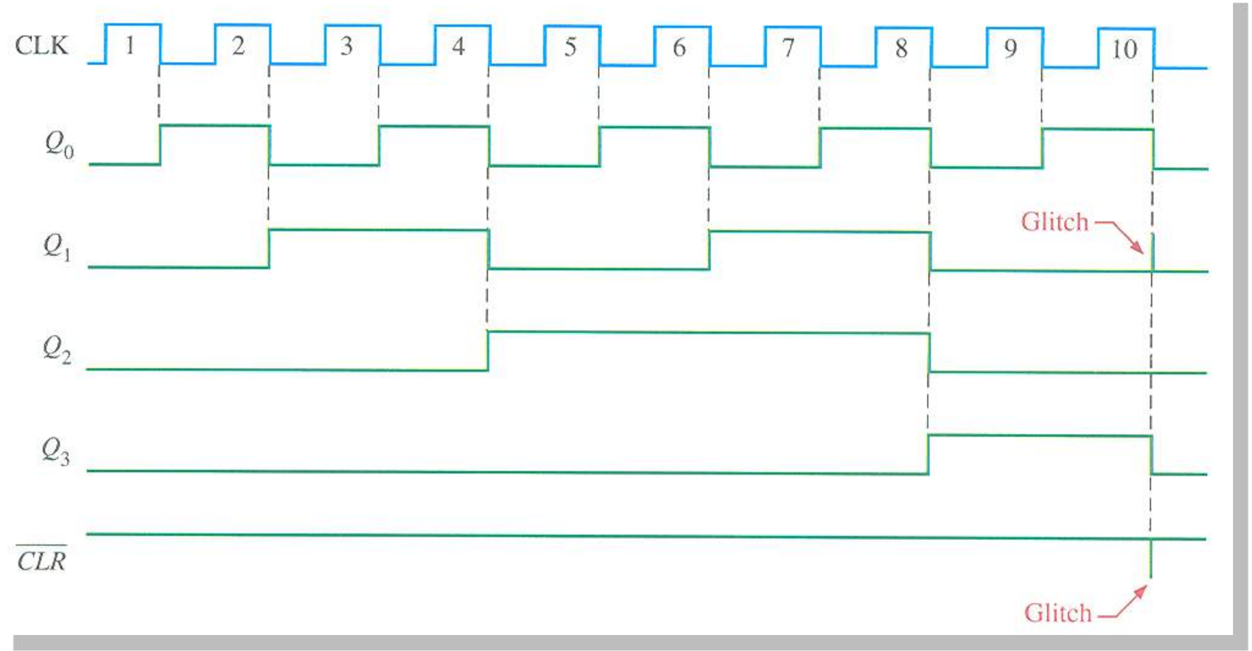


To obtain a truncated sequence, it is necessary to force the counter to recycle before going through all of its possible states. For example, the BCD decade counter must recycle back to the 0000 state after the 1001 state. A decade counter requires four flip-flops (three flip-flops are insufficient because $2^3 = 8$).

Asynchronous decade counter



DECIMAL	BINARY
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111



Nafiz Ahmed Chisty| Head (UG) & Associate Professor, Dept. of EEE, FE, AIUB|

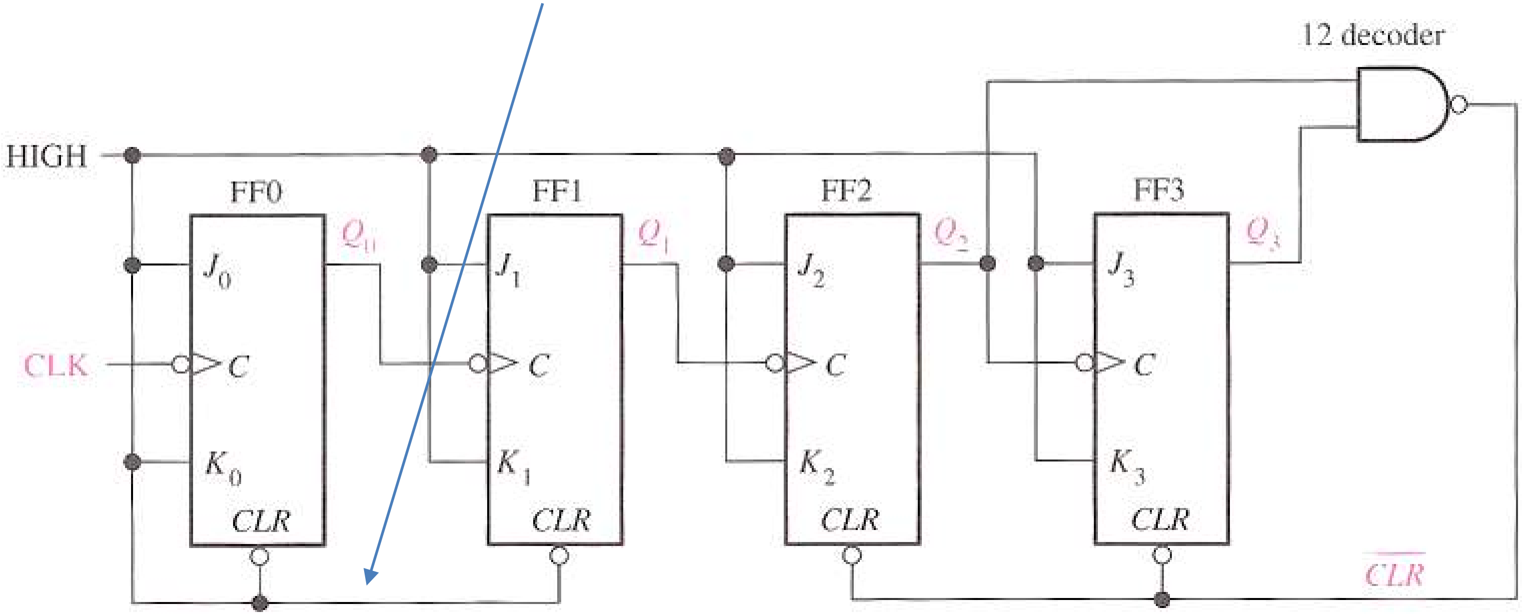
Asynchronous Modulus Twelve counter

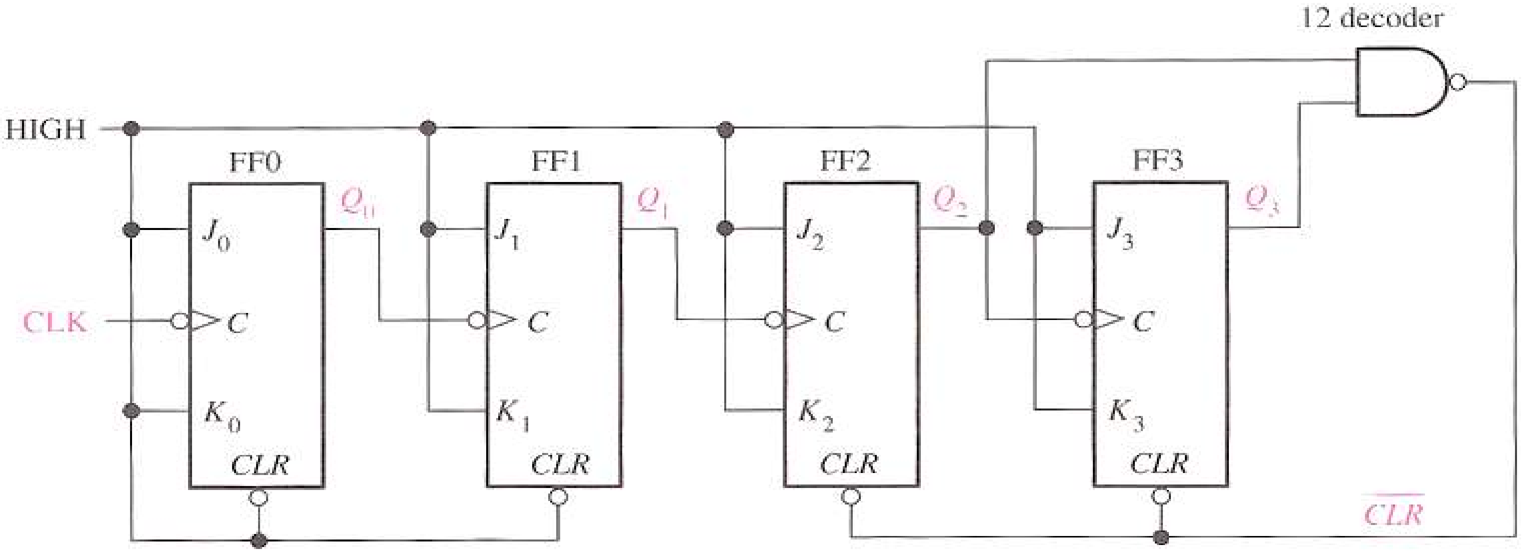
Q_3	Q_2	Q_1	Q_0
0	0	0	0
.	.	.	.
.	.	.	.
.	.	.	.
1	0	1	1
1	1	0	0

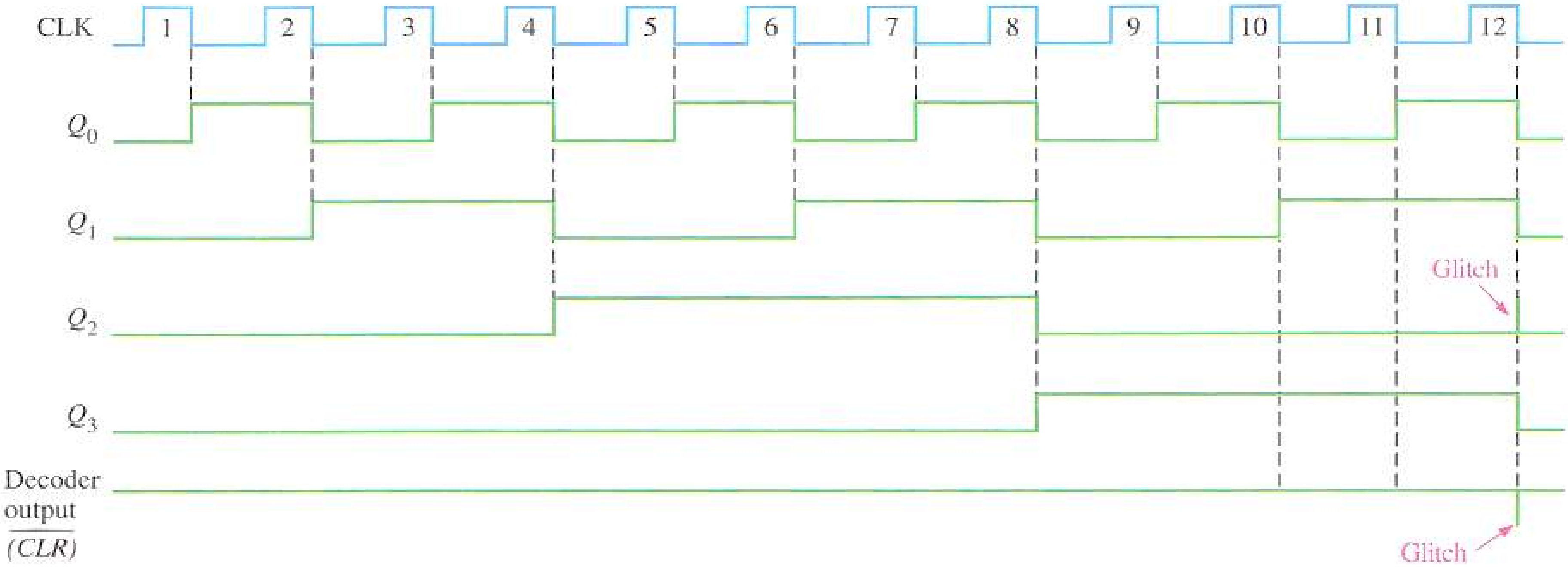
Recycles

Normal next state So Decode this state.

Already ZERO, so no need to ACTIVAE Clear




12 decoder
HIGH
CLK
FF0
FF1
FF2
FF3
J_0
K_0
C
CLR
Q_0
J_1
K_1
Q_1
J_2
K_2
Q_2
J_3
K_3
Q_3
$\overline{CLR}$


CLK
1
2
3
4
5
6
7
8
9
10
11
12
Q_0
Q_1
Q_2
Q_3
Decoder output $(\overline{CLR})$
Glitch
Glitch

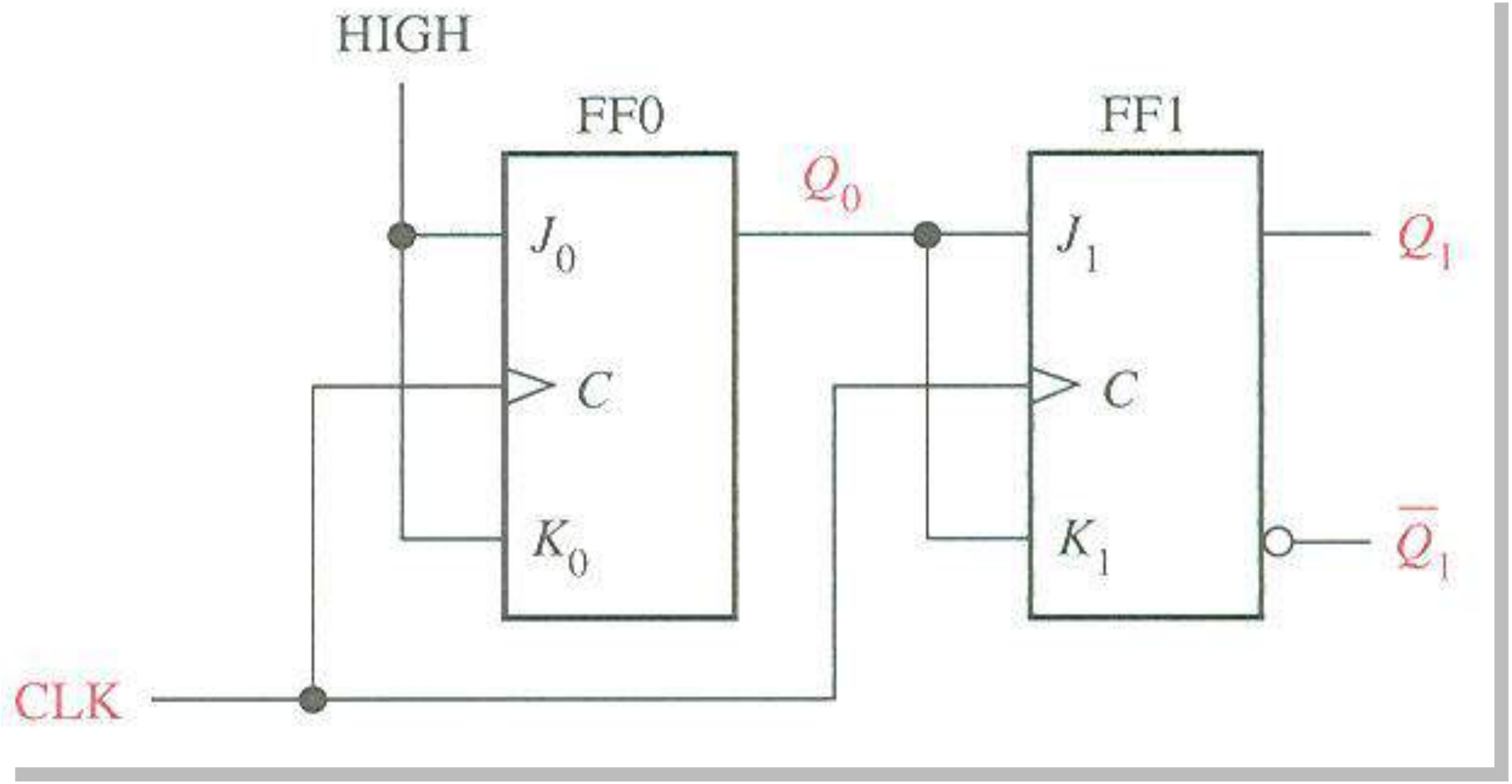
Synchronous Counter

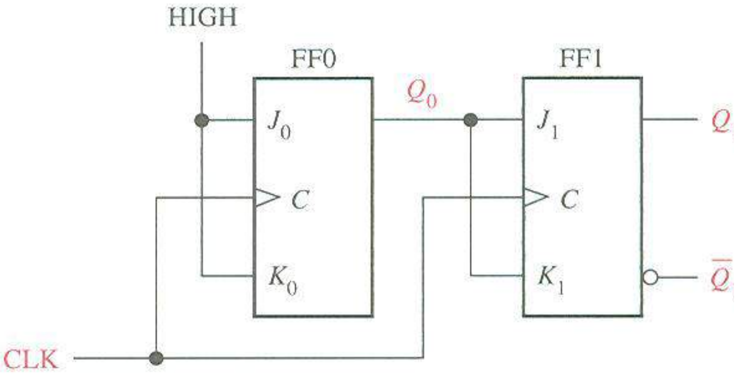
A Synchronous counter is one in which the flip-flops(FF) within the counter are clocked at the same time by a common clock pulse.

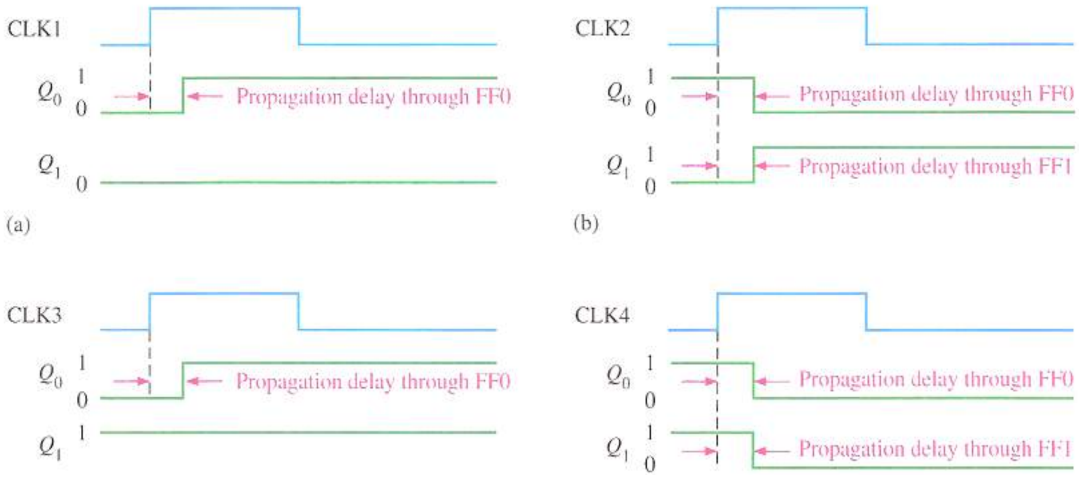
Synchronous Counter Operation

- Synchronous binary counters
  - 2-bit counter
  - 3-bit counter
  - 4-bit counter
- Synchronous BCD Decade counter
- Irregular sequence counter

2-bit synchronous binary counter



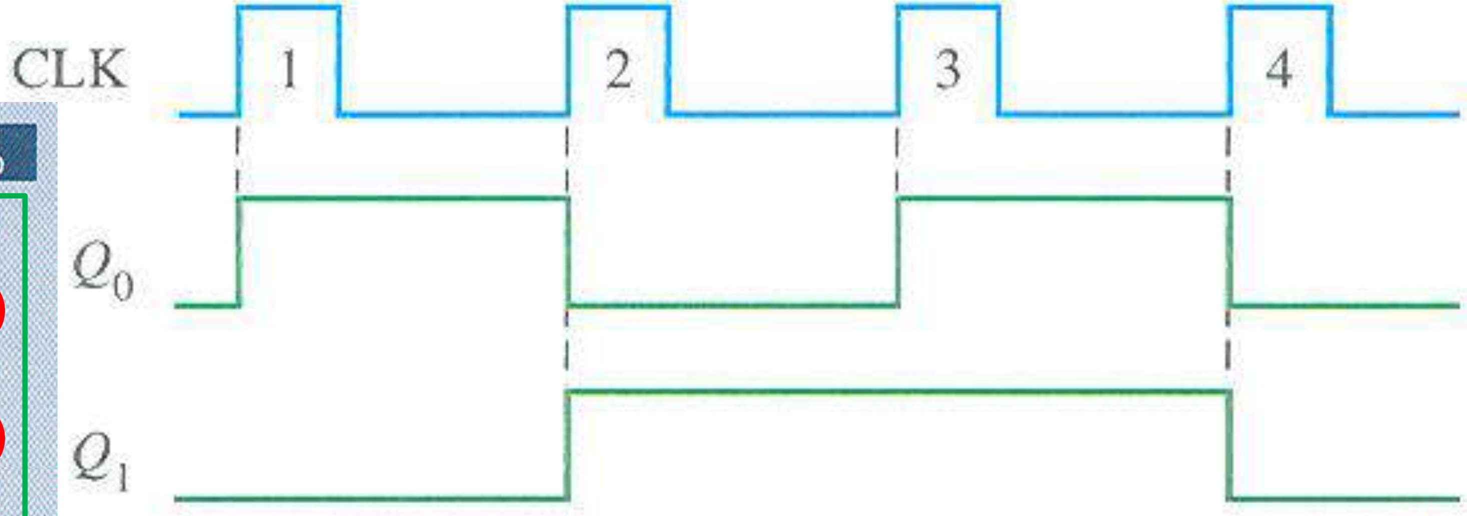



CLK1
Q_0 1 0
Propagation delay through FF0
Q_1 0
(a)
CLK2
Q_0 1 0
Propagation delay through FF0
Q_1 1 0
Propagation delay through FF1
(b)
CLK3
Q_0 1 0
Propagation delay through FF0
Q_1 1
CLK4
Q_0 1 0
Propagation delay through FF0
Q_1 1 0
Propagation delay through FF1

$$J_0 = K_0 = 1$$

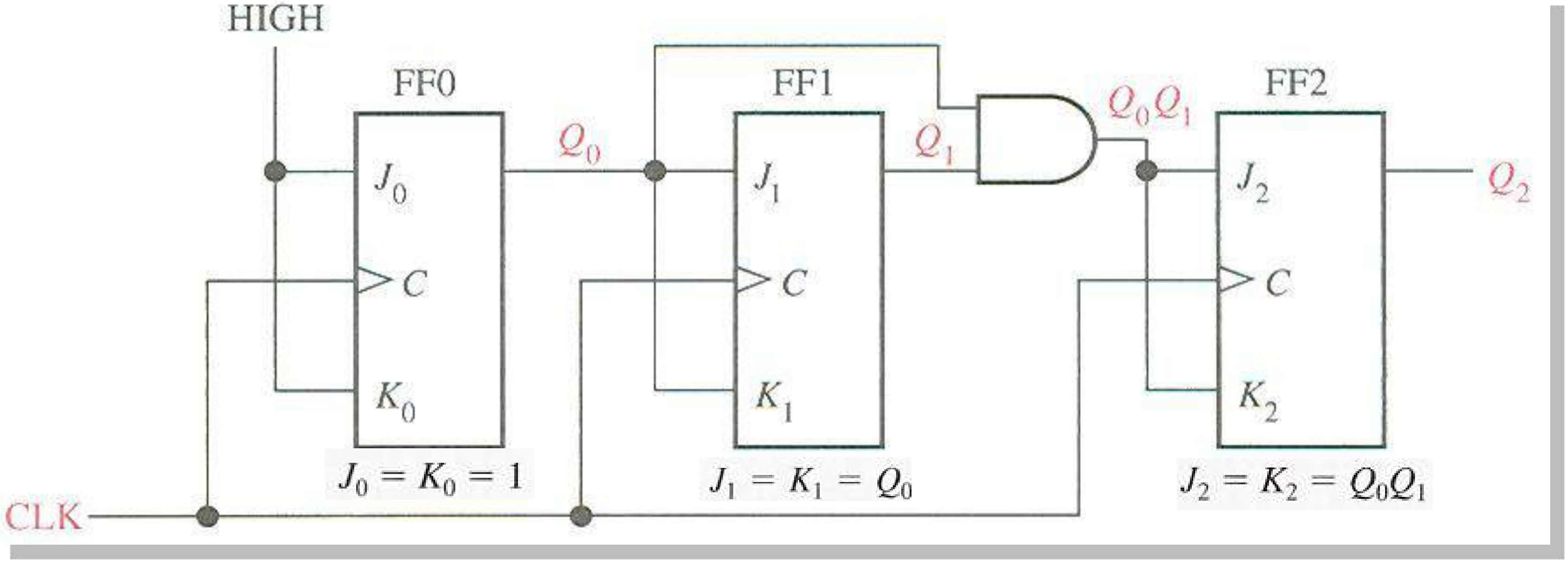
$$J_1 = K_1 = Q_0$$

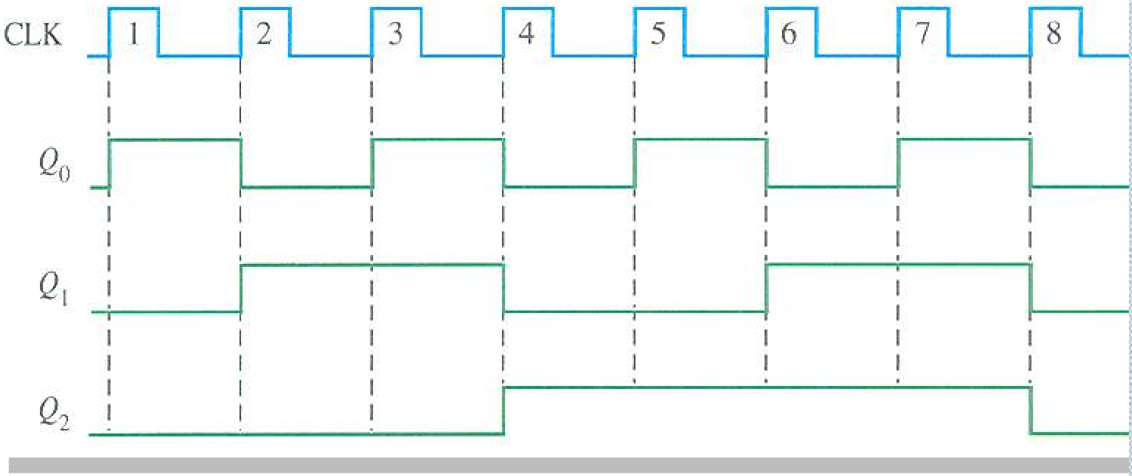
Q_1	Q_0
0	0
0	1
1	0
1	1
0	0



Nafiz Ahmed Chisty| Head (UG) & Associate Professor, Dept. of EEE, FE, AIUB|

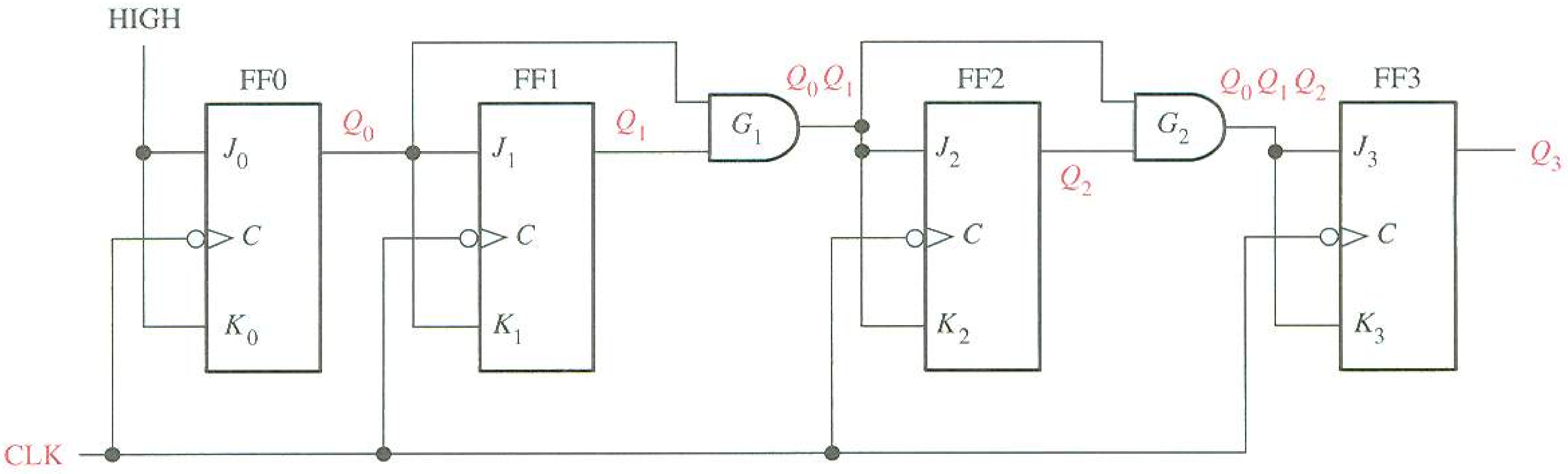
3-bit synchronous binary counter

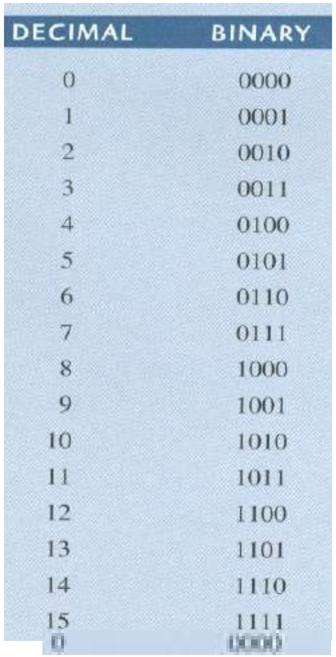




CLOCK PULSE	Q_2	Q_1	Q_0
Initially	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8 (recycles)	0	0	0

4-bit synchronous binary counter





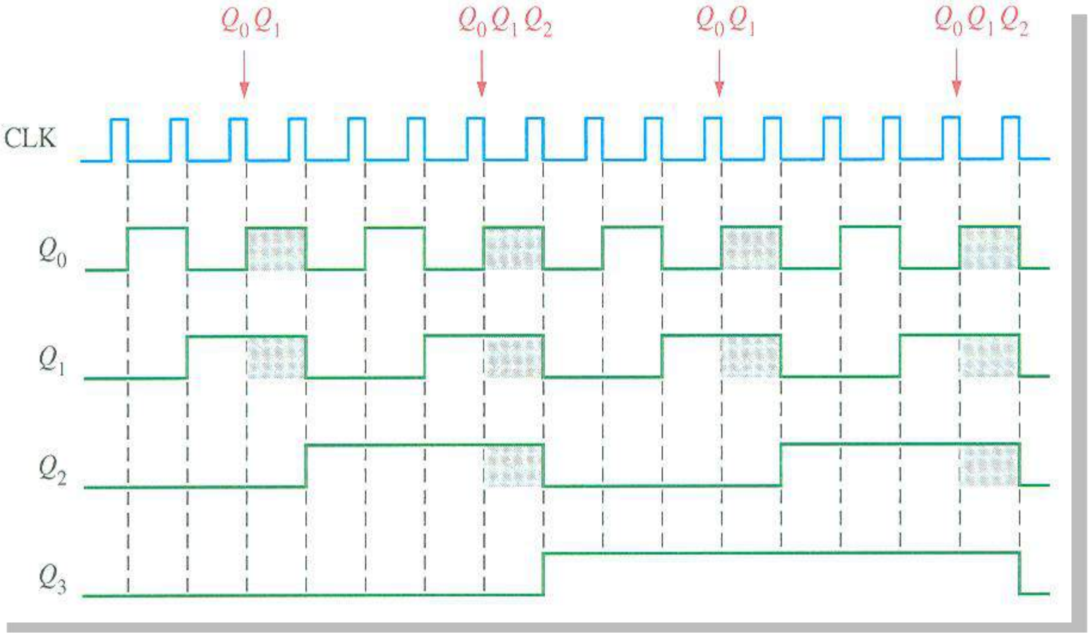
DECIMAL	BINARY
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111
0	0000

$$J_0 = K_0 = 1$$

$$J_1 = K_1 = Q_0$$

$$J_2 = K_2 = Q_0Q_1$$

$$J_3 = K_3 = Q_0Q_1Q_2$$



Synchronous Decade counter (MOD 10)

4-bit synchronous binary counter

$J_0 = K_0 = 1$

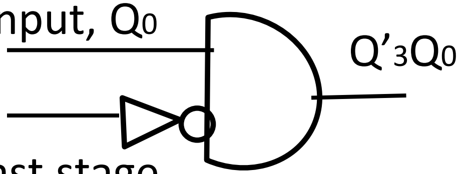
$J_1 = K_1 = Q_0$

$J_2 = K_2 = Q_0Q_1$

$J_3 = K_3 = Q_0Q_1Q_2$

Forcefully Stable

Normal input, Q_0

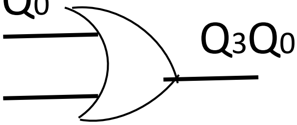


Q'_3Q_0

Decode last stage before recycle, Q_3Q_0

Forcefully Toggle

Normal input, $Q_2Q_1Q_0$



$Q_3Q_0+Q_2Q_1Q_0$

Decode last stage before recycle, Q_3Q_0

Naturally Zero (unchanged)

	$Q_3Q_2Q_1Q_0$
9	1001
10	1010
0	0000

Forcefully Toggle

Forcefully Stable

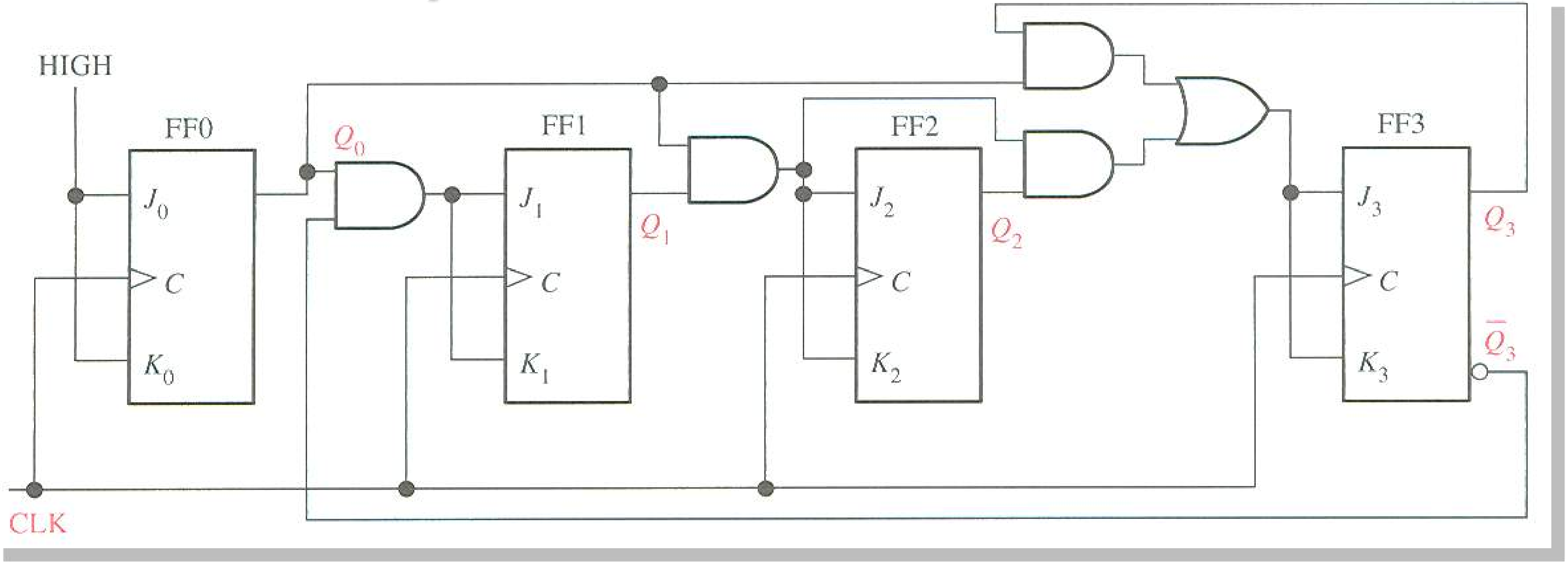
$J_0 = K_0 = 1$

$J_1 = K_1 = Q_0\overline{Q}_3$

$J_2 = K_2 = Q_0Q_1$

$J_3 = K_3 = Q_0Q_1Q_2 + Q_0Q_3$

Synchronous BCD decade counter

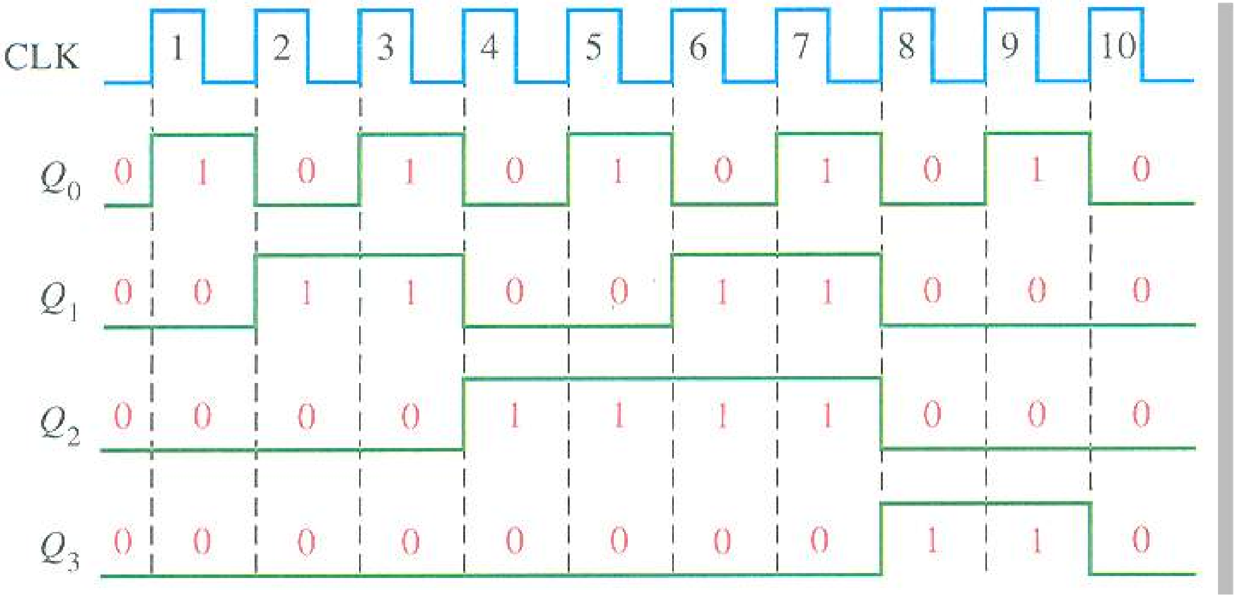


$$J_0 = K_0 = 1$$

$$J_1 = K_1 = Q_0\overline{Q}_3$$

$$J_2 = K_2 = Q_0Q_1$$

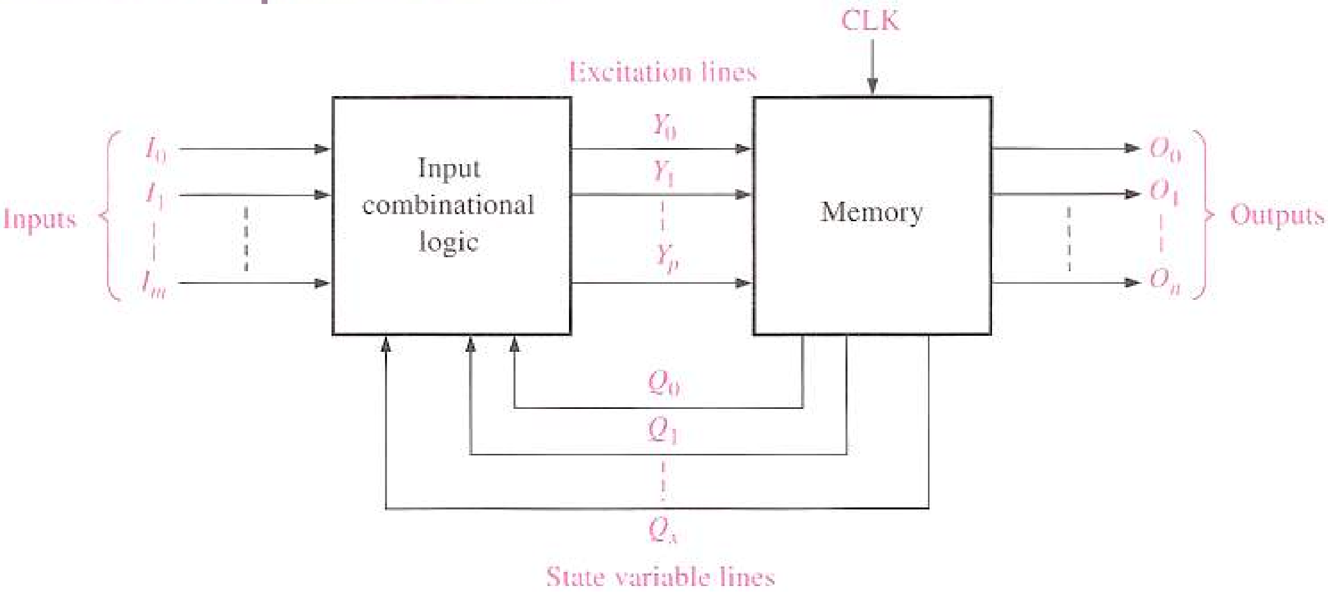
$$J_3 = K_3 = Q_0Q_1Q_2 + Q_0Q_3$$



DESIGN OF SYNCHRONOUS COUNTERS

In this section, you will see how sequential circuit design techniques can be applied specifically to counter design. In general, sequential circuits can be classified into two types: (1) those in which the output or outputs depend only on the present internal state (called *Moore circuits*) and (2) those in which the output or outputs depend on both the present state and the input or inputs (called *Mealy circuits*). This section is

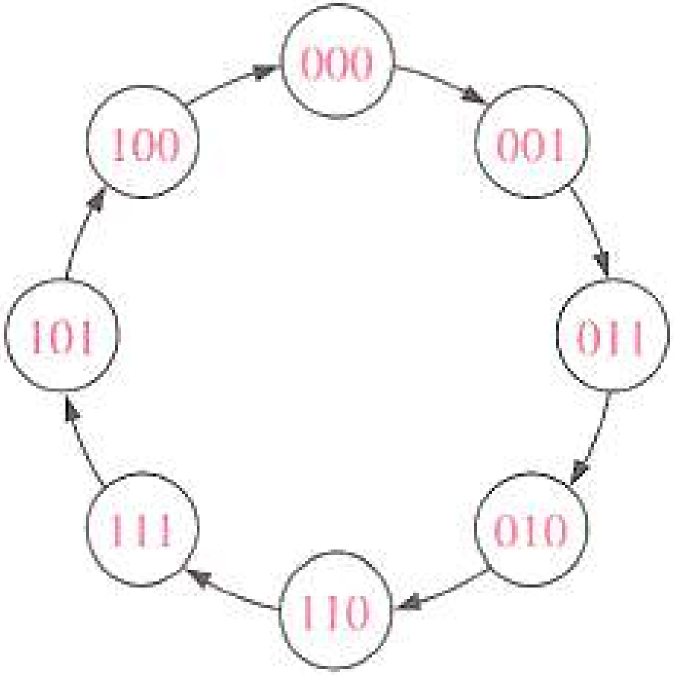
General Model of a Sequential Circuit (sequential circuit or state machine)



Not all sequential circuits have input and output variables as in the general model just discussed. However, all have excitation variables and state variables. Counters are a special case of clocked sequential circuits. In this section, a general design procedure for sequential circuits is applied to synchronous counters in a series of steps.

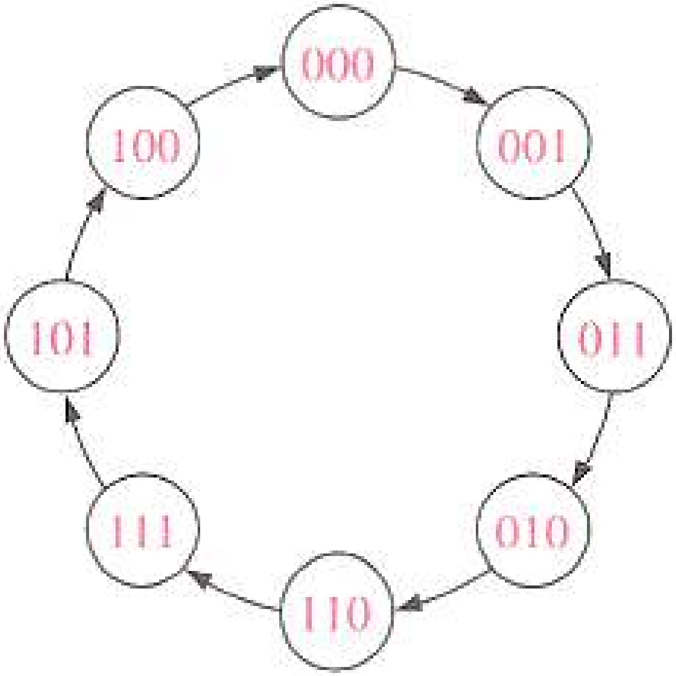
Step 1: State Diagram

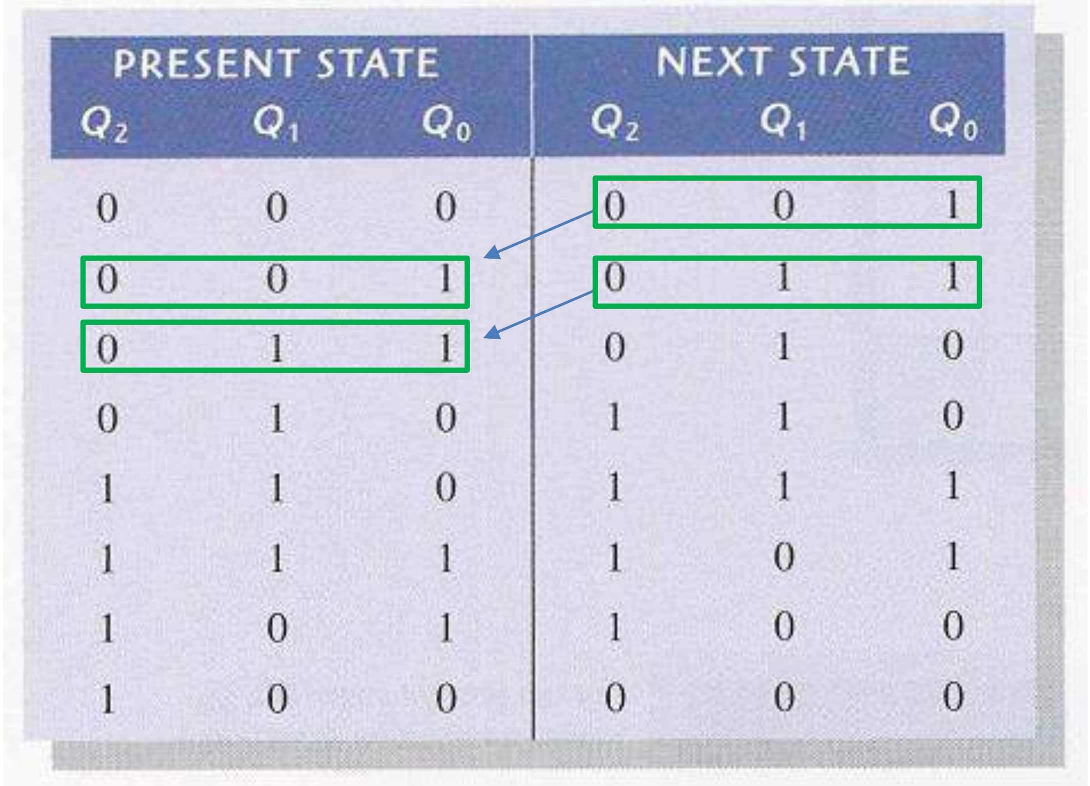
The first step in the design of a counter is to create a state diagram. A **state diagram** shows the progression of states through which the counter advances when it is clocked. As an example, Figure 8–28 is a state diagram for a basic 3-bit Gray code counter. This particular circuit has no inputs other than the clock and no outputs other than the outputs taken off each flip-flop in



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 8–7 for the 3-bit Gray code counter. Q_0 is the least significant bit.

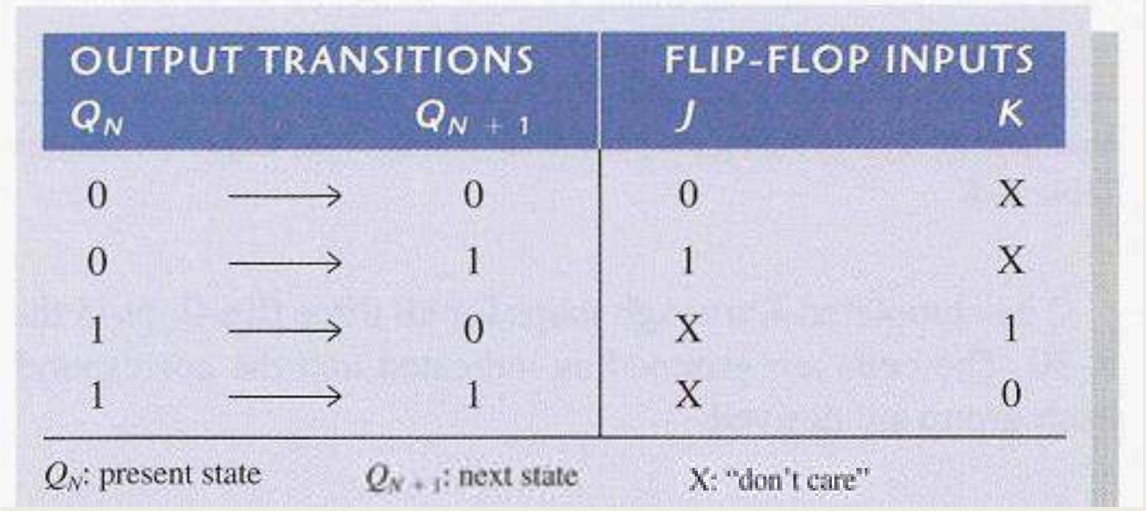




PRESENT STATE			NEXT STATE		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0
0	0	0	0	0	1
0	0	1	0	1	1
0	1	1	0	1	0
0	1	0	1	1	0
1	1	0	1	1	1
1	1	1	1	0	1
1	0	1	1	0	0
1	0	0	0	0	0

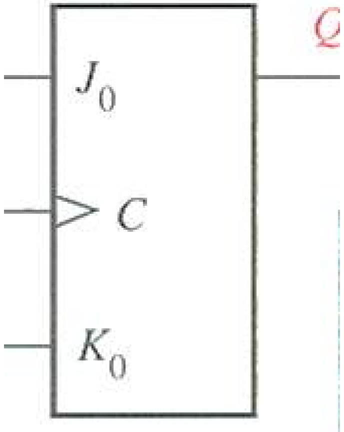
Step 3: Flip-Flop Transition Table

Table 8–8 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. Q_N is the present state of the flip-flop (before a clock pulse) and Q_{N+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).



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"



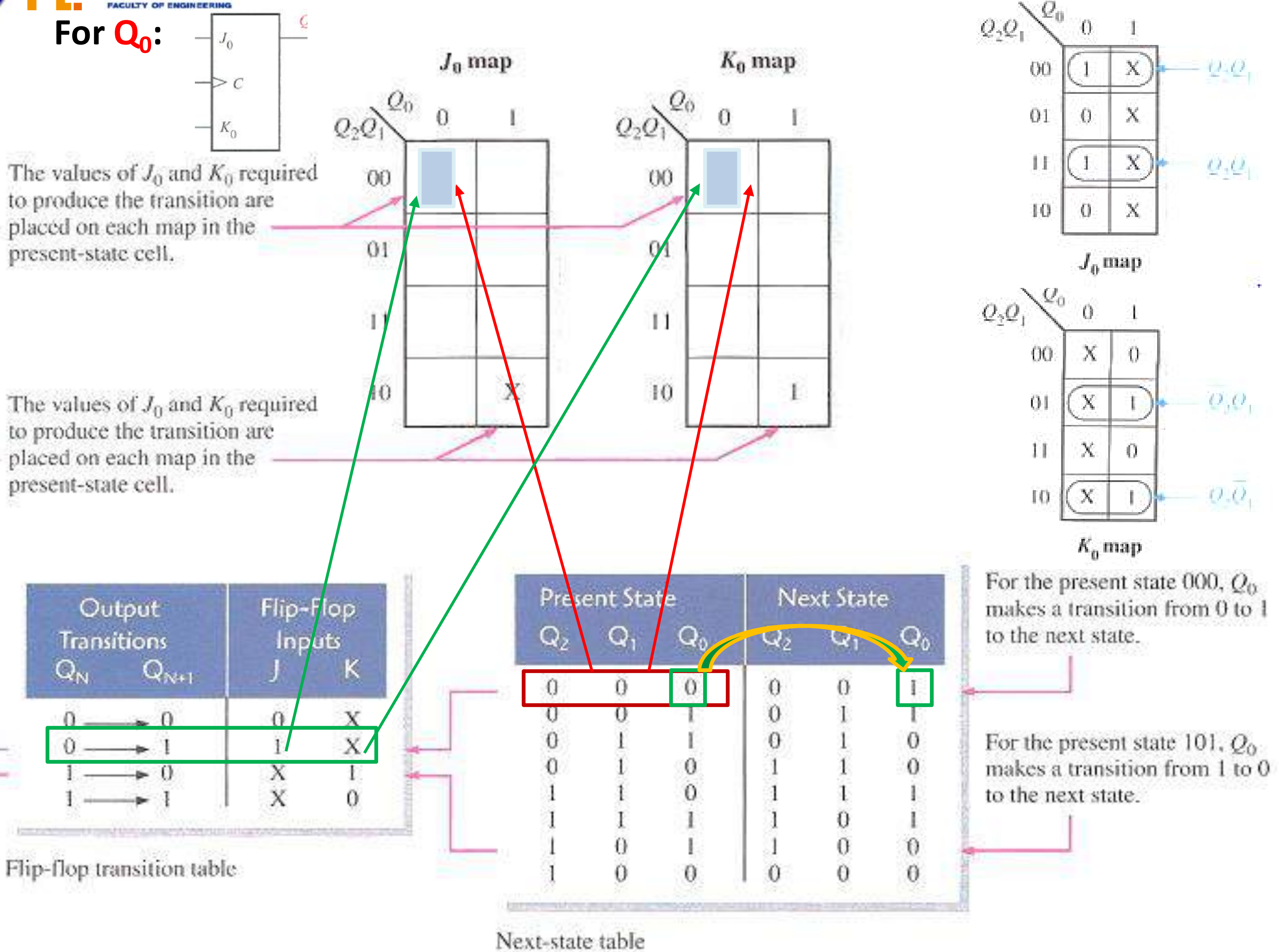
To design the counter, the transition table is applied to each of the flip-flops in the counter, based on the next-state table (Table 8–7). For example, for the present state 000, Q_0 goes from a present state of 0 to a next state of 1. To make this happen, J_0 must be a 1 and you don't care what K_0 is ($J_0 = 1$, $K_0 = X$), as you can see in the transition table (Table 8–8). Next, Q_1 is 0 in the present state and remains a 0 in the next state. For this transition, $J_1 = 0$ and $K_1 = X$. Finally, Q_2 is 0 in the present state and remains a 0 in the next state. Therefore, $J_2 = 0$ and $K_2 = X$. This analysis is repeated for each present state in Table 8–7.

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 8–7.

From the *J* and *K* states in the transition table (Table 8–8) 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 J_0 and the K_0 inputs to the least significant flip-flop (Q_0) in Figure 8–29.

For Q_0:



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.

J_0 map

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

K_0 map

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

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

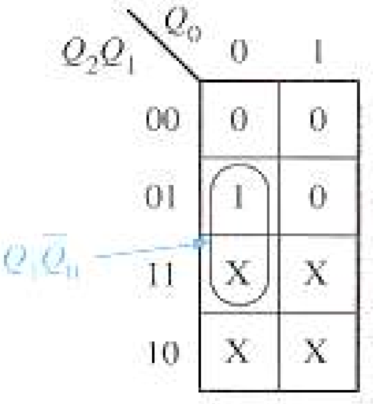
Next-state table

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.

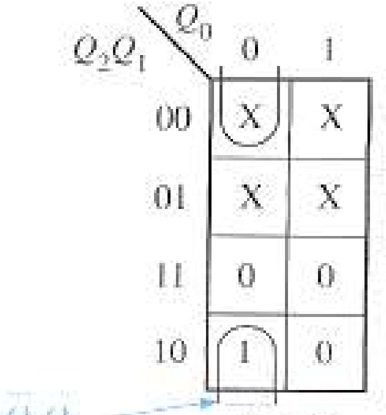
The completed Karnaugh maps for all three flip-flops in the counter are shown in Figure 8–30. The cells are grouped as indicated and the corresponding Boolean expressions for each group are derived.

For Q_2:



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

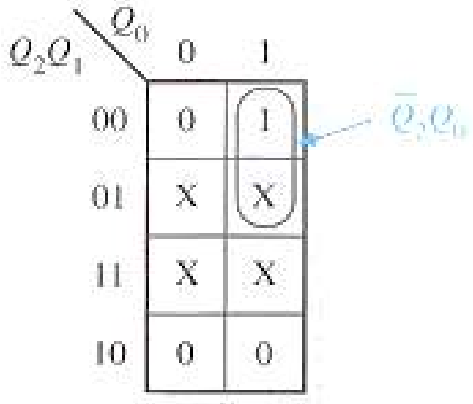
J_2 map



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

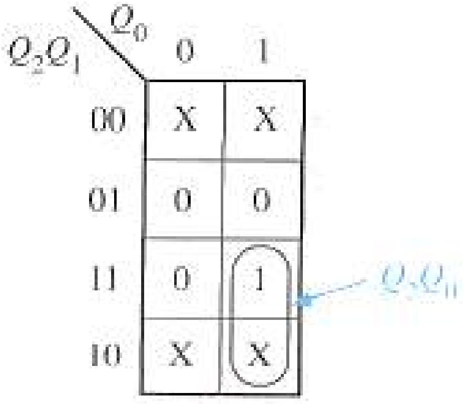
K_2 map

For Q_1:



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

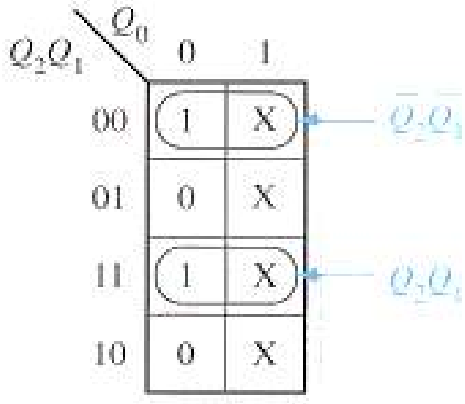
J_1 map



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

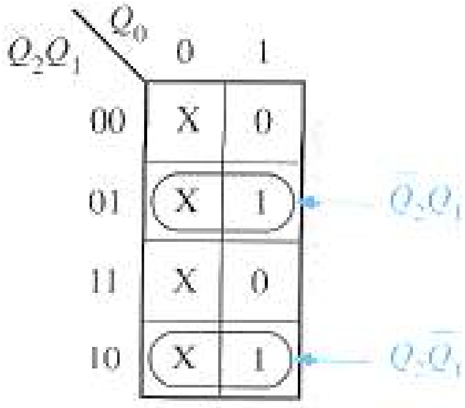
K_1 map

For Q_0:



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	0
01	X	1
11	X	0
10	X	1

K_0 map

Step 5: Logic Expressions for Flip-Flop Inputs

From the Karnaugh maps of Figure 8–30 you obtain the following expressions for the *J* and *K* inputs of each flip-flop:

$$J_0 = Q_2Q_1 + \overline{Q}_2\overline{Q}_1 = \overline{Q_2 \oplus Q_1}$$

$$K_0 = Q_2\overline{Q}_1 + \overline{Q}_2Q_1 = Q_2 \oplus Q_1$$

$$J_1 = \overline{Q}_2Q_0$$

$$K_1 = Q_2Q_0$$

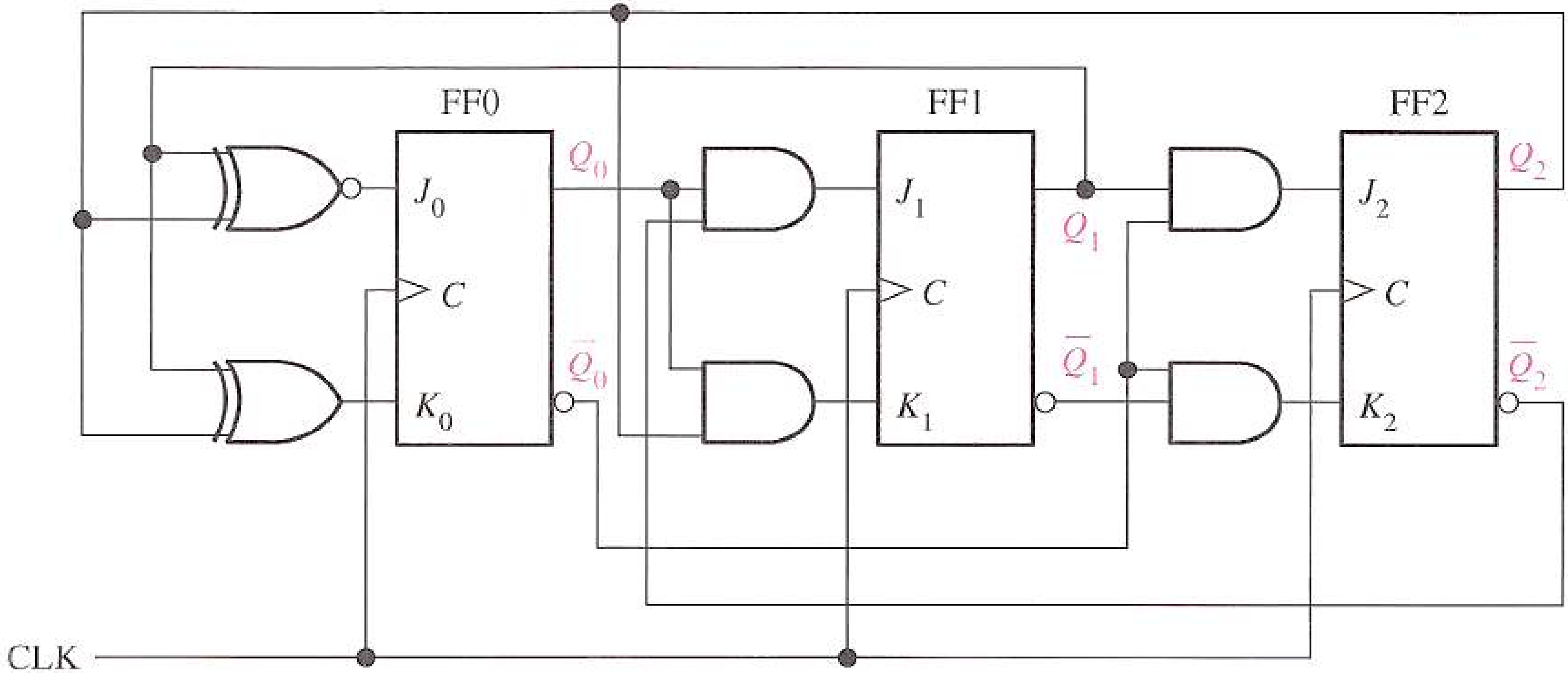
$$J_2 = Q_1\overline{Q}_0$$

$$K_2 = \overline{Q}_1\overline{Q}_0$$

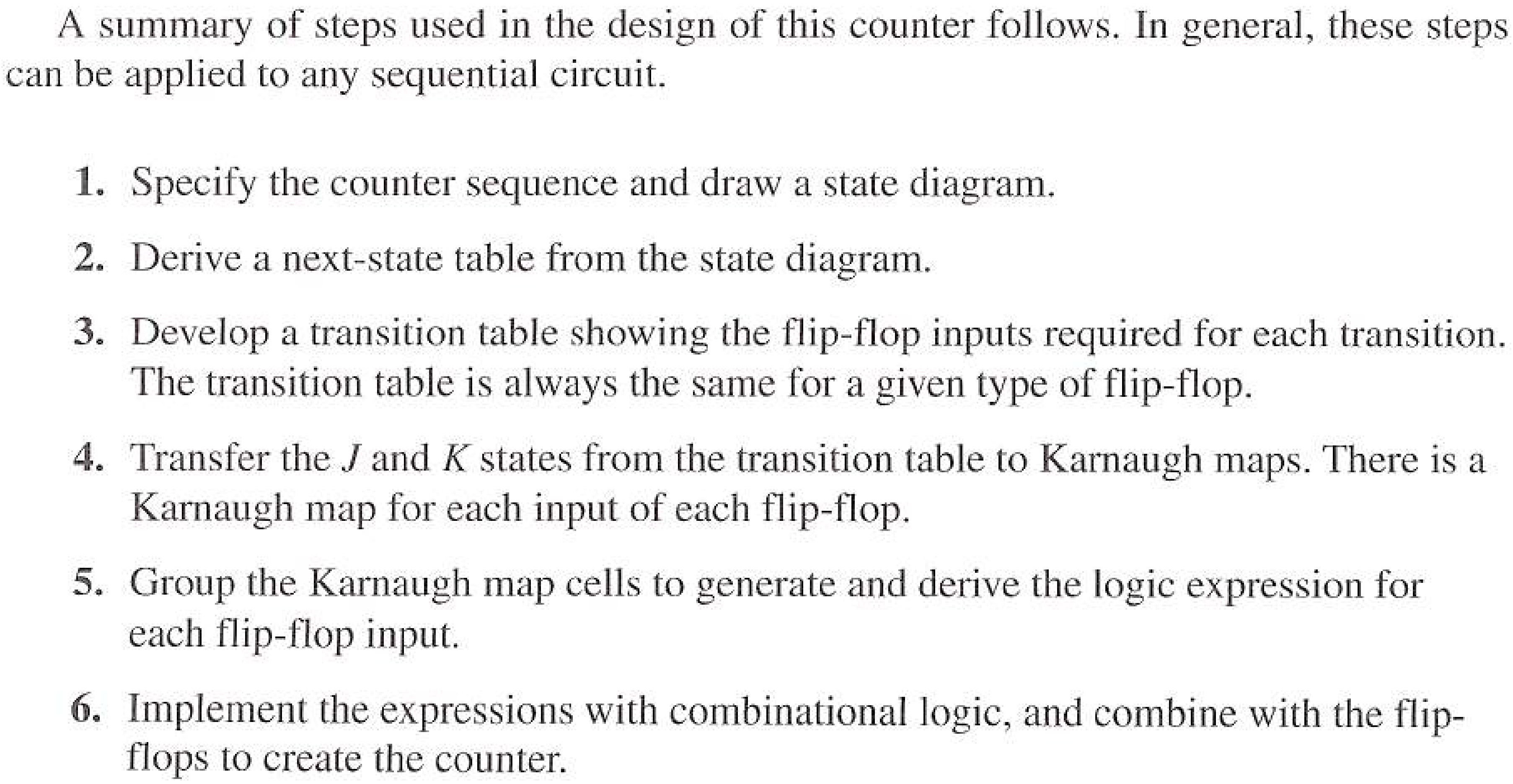
Implement in front of the flip-flops

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 8–31.



Three-bit Gray code counter.

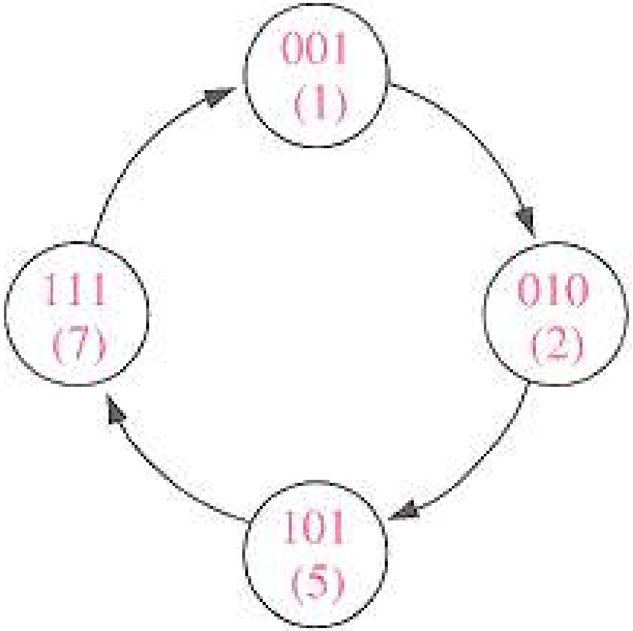


A summary of steps used in the design of this counter follows. In general, these steps can be applied to any sequential circuit.

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.

EXAMPLE

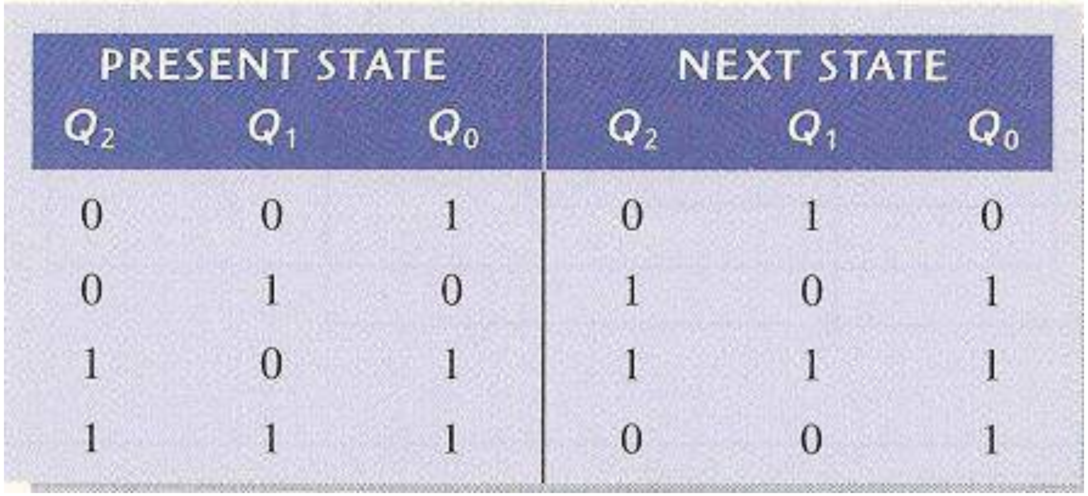
Design a counter with the irregular binary count sequence shown in the state diagram of Figure 8–32. Use J-K flip-flops.



Solution

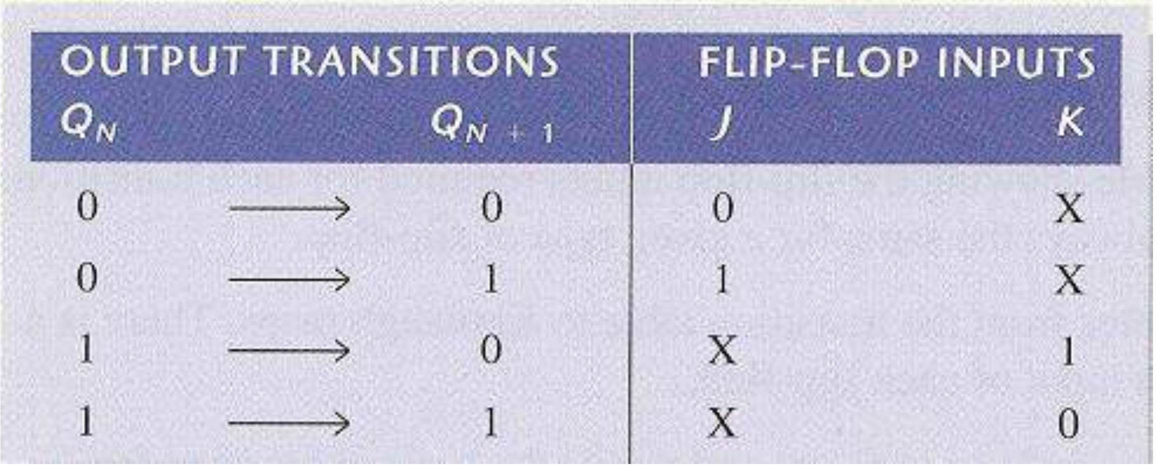
Step 1: The state diagram is as shown. Although there are only four states, a 3-bit counter is required to implement this sequence because the maximum binary count is seven. Since the required sequence does not include all the possible binary states, the invalid states (0, 3, 4, and 6) can be treated 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 8–9.



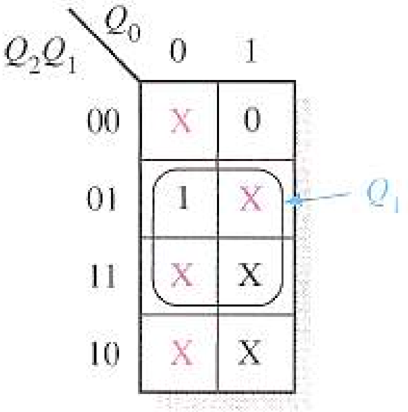
PRESENT STATE			NEXT STATE		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0
0	0	1	0	1	0
0	1	0	1	0	1
1	0	1	1	1	1
1	1	1	0	0	1

Step 3: The transition table for the J-K flip-flop is repeated in Table 8–10.



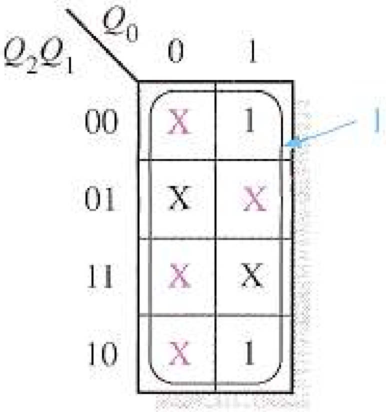
OUTPUT TRANSITIONS			FLIP-FLOP INPUTS	
Q_N		Q_{N+1}	J	K
0	→	0	0	X
0	→	1	1	X
1	→	0	X	1
1	→	1	X	0

Step 4: The *J* and *K* inputs are plotted on the present-state Karnaugh maps in Figure 8–33. 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.



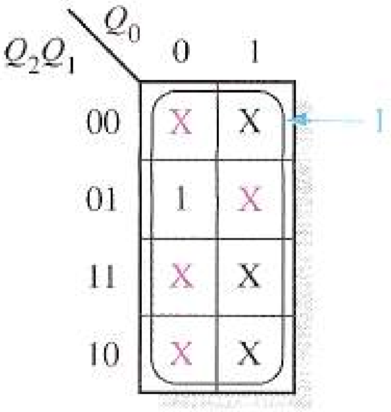
Q_2Q_1 \ Q_0	0	1
00	X	0
01	1	X
11	X	X
10	X	X

J_2 map



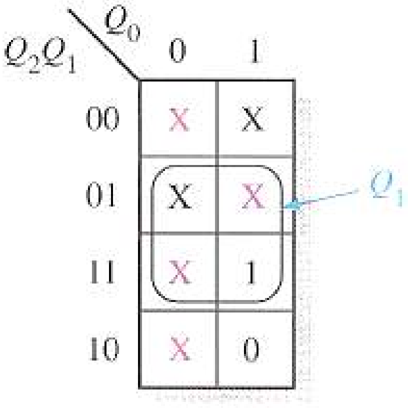
Q_2Q_1 \ Q_0	0	1
00	X	1
01	X	X
11	X	X
10	X	1

J_1 map



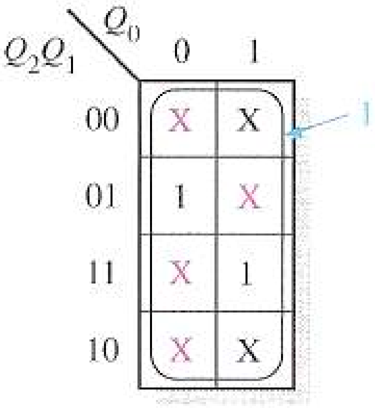
Q_2Q_1 \ Q_0	0	1
00	X	X
01	1	X
11	X	X
10	X	X

J_0 map



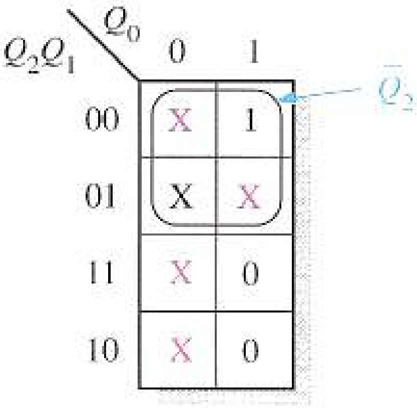
Q_2Q_1 \ Q_0	0	1
00	X	X
01	X	X
11	X	1
10	X	0

K_2 map



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

K_1 map



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

K_0 map

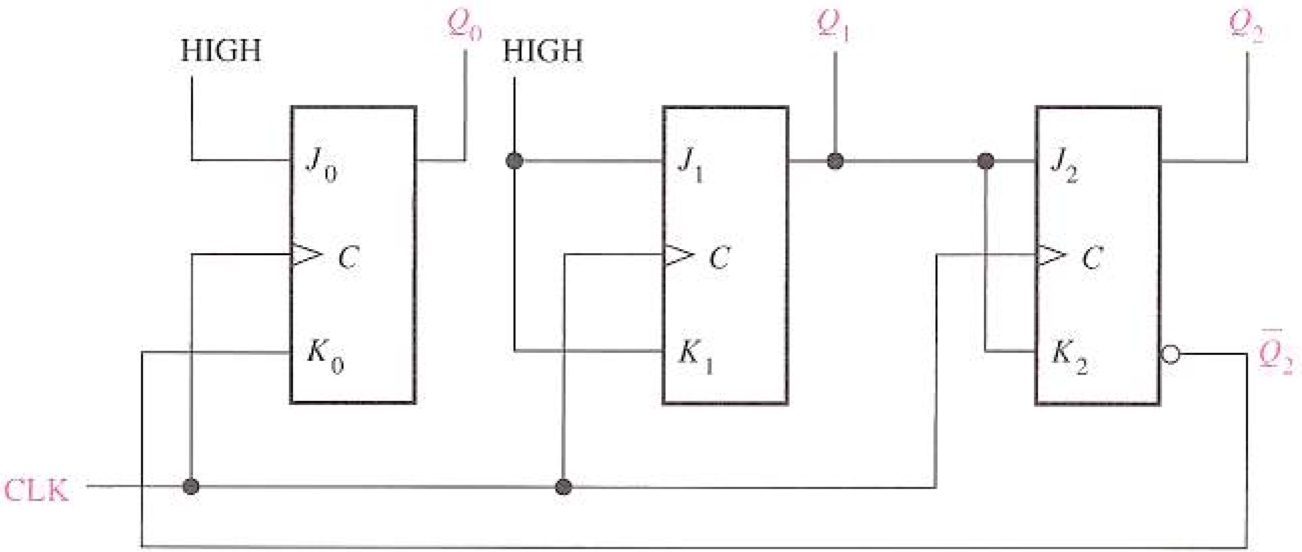
PRESENT STATE			NEXT STATE		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0
0	0	1	0	1	0
0	1	0	1	0	1
1	0	1	1	1	1
1	1	1	0	0	1

The *J* and *K* inputs are plotted on the present-state Karnaugh maps in Figure 8–33. 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.

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 8–33. Notice that when *all* cells in a map are grouped, the expression is simply equal to 1. The expression for each *J* and *K* input taken from the maps is as follows:

$$J_0 = 1, K_0 = \overline{Q}_2$$
$$J_1 = K_1 = 1$$
$$J_2 = K_2 = Q_1$$

Step 6: The implementation of the counter is shown in Figure 8–34.

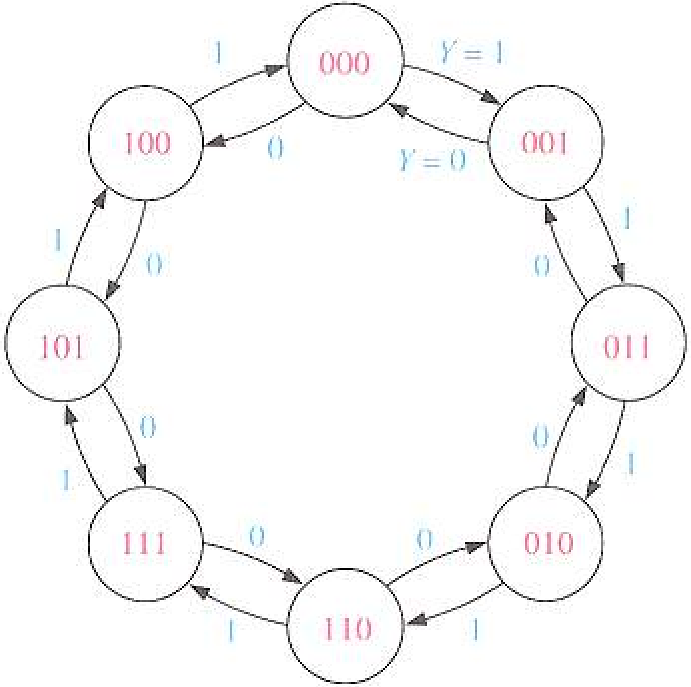


EXAMPLE

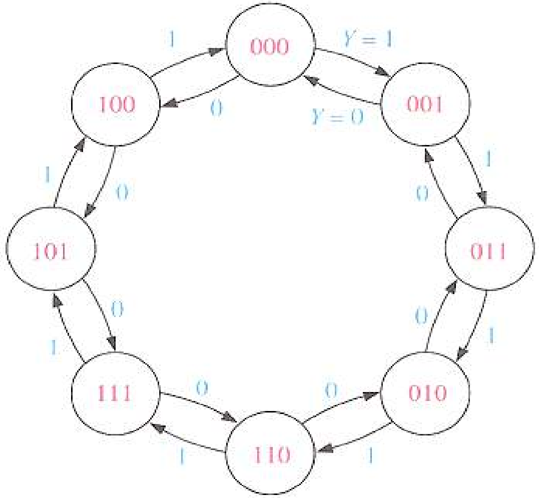
Develop a synchronous 3-bit up/down counter with a Gray code sequence. The counter should count up when an $\text{UP}/\overline{\text{DOWN}}$ control input is 1 and count down when the control input is 0.

Solution

Step 1: The state diagram is shown in Figure 8–35. The 1 or 0 beside each arrow indicates the state of the $\text{UP}/\overline{\text{DOWN}}$ control input, *Y*.



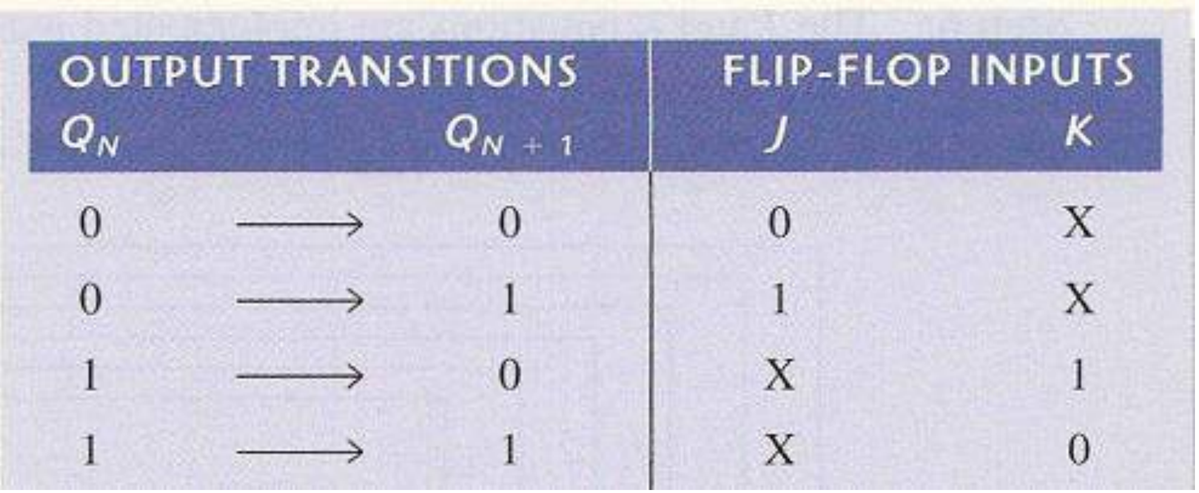
Next-State Table



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

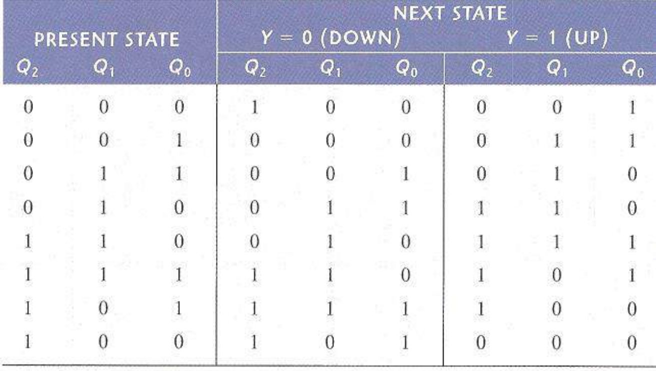
Y = UP/$\overline{\text{DOWN}}$ control input.

Step 3: The transition table for the J-K flip-flops is repeated in Table 8–12.



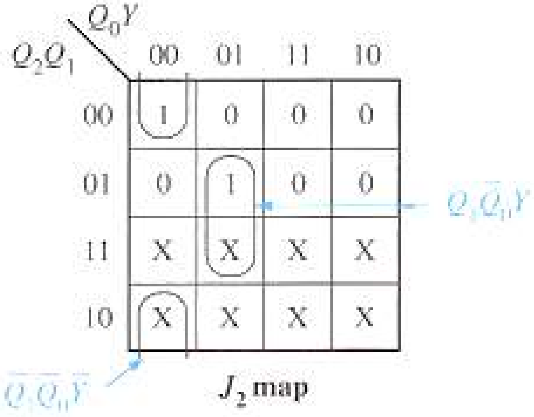
OUTPUT TRANSITIONS			FLIP-FLOP INPUTS	
Q_N		Q_{N+1}	J	K
0	⟶	0	0	X
0	⟶	1	1	X
1	⟶	0	X	1
1	⟶	1	X	0

Step 4: The Karnaugh maps for the *J* and *K* inputs of the flip-flops are shown in Figure 8–36. The $\text{UP}/\overline{\text{DOWN}}$ 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 8–12 is transferred onto the maps as indicated for each present state of the counter.



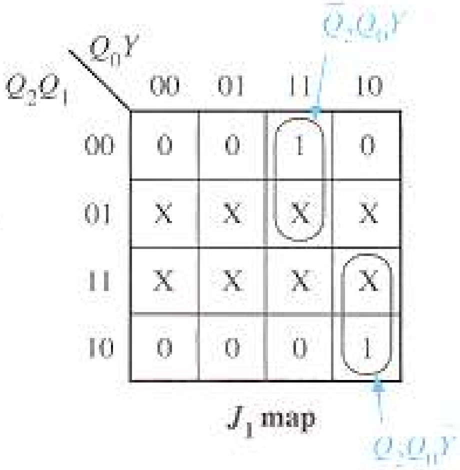
PRESENT STATE			NEXT STATE					
			Y = 0 (DOWN)			Y = 1 (UP)		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0	Q_2	Q_1	Q_0
0	0	0	1	0	0	0	0	1
0	0	1	0	0	0	0	1	1
0	1	1	0	0	1	0	1	0
0	1	0	0	1	1	1	1	0
1	1	0	0	1	0	1	1	1
1	1	1	1	1	0	1	0	1
1	0	1	1	1	1	1	0	0
1	0	0	1	0	1	0	0	0

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



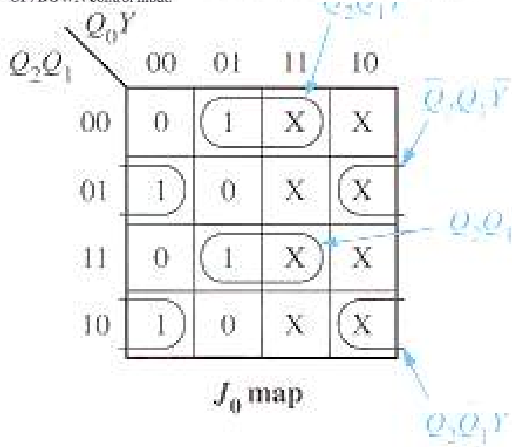
Q_2Q_1 Q_0Y	00	01	11	10
00	1	0	0	0
01	0	1	0	0
11	X	X	X	X
10	X	X	X	X

J_2 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

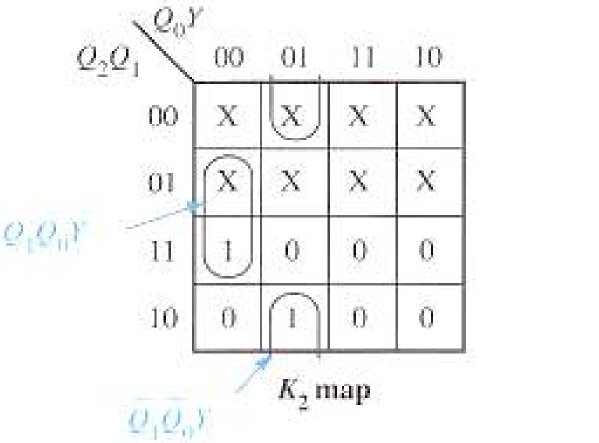
J_1 map



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

$\overline{Q}_2\overline{Q}_1Y$
$\overline{Q}_2Q_1\overline{Y}$
Q_2Q_1Y
$Q_2\overline{Q}_1\overline{Y}$

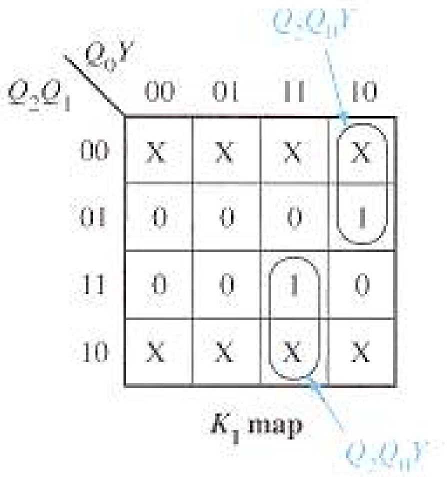
J_0 map



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

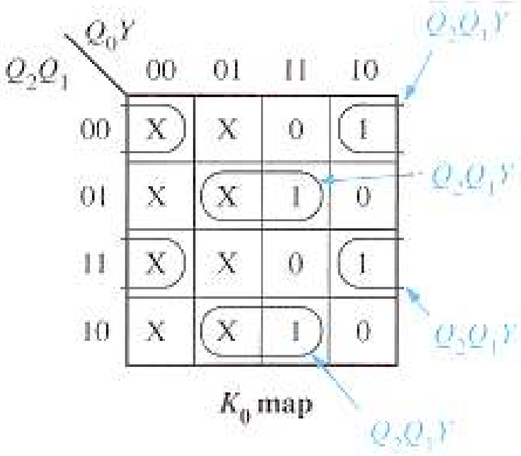
$Q_1\overline{Q}_0\overline{Y}$
$\overline{Q}_1\overline{Q}_0Y$

K_2 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_1 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

K_0 map

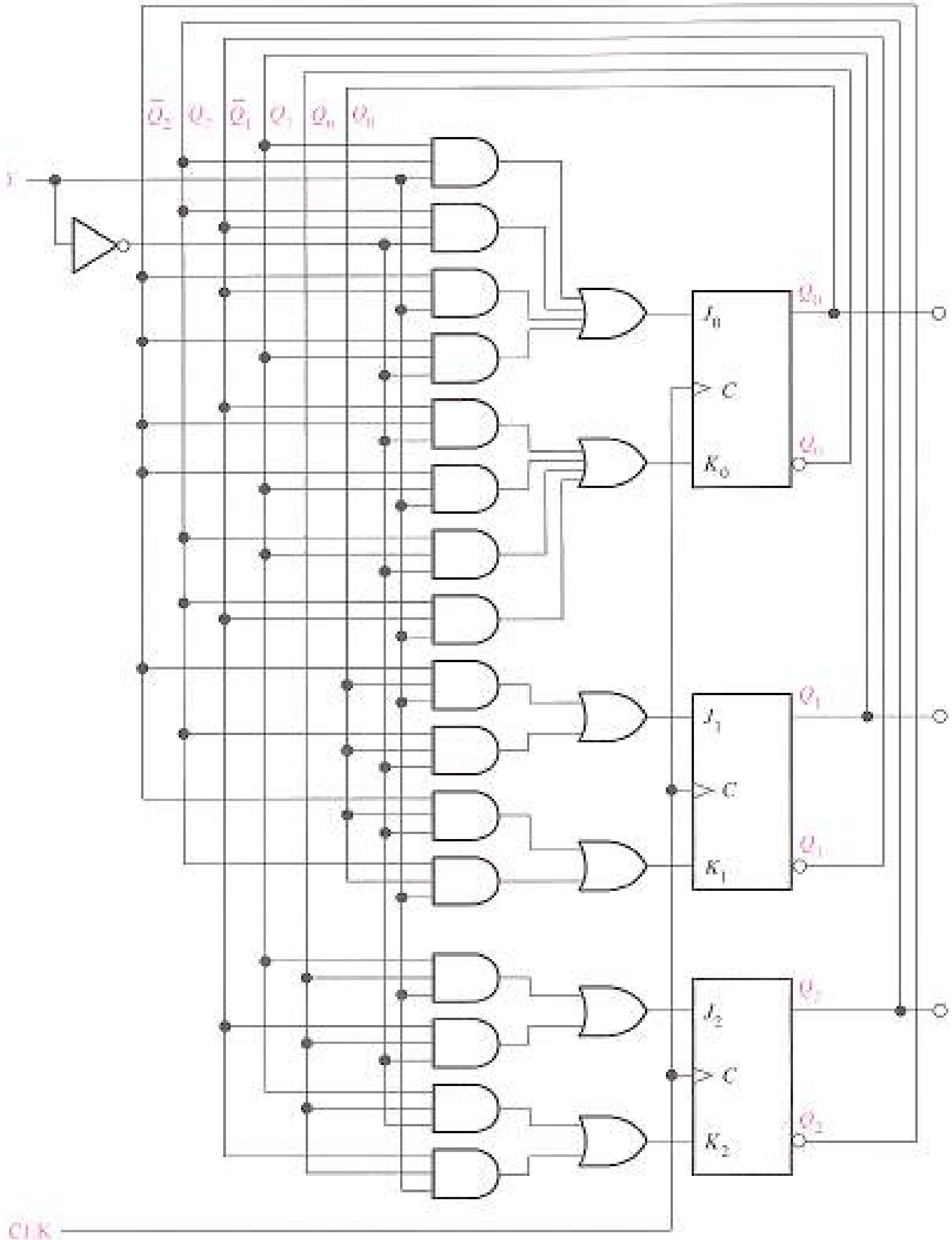
Step 5: The 1s are combined in the largest possible groupings, with "don't cares" (Xs) used where possible. The groups are factored, and the expressions for the *J* and *K* inputs are as follows:

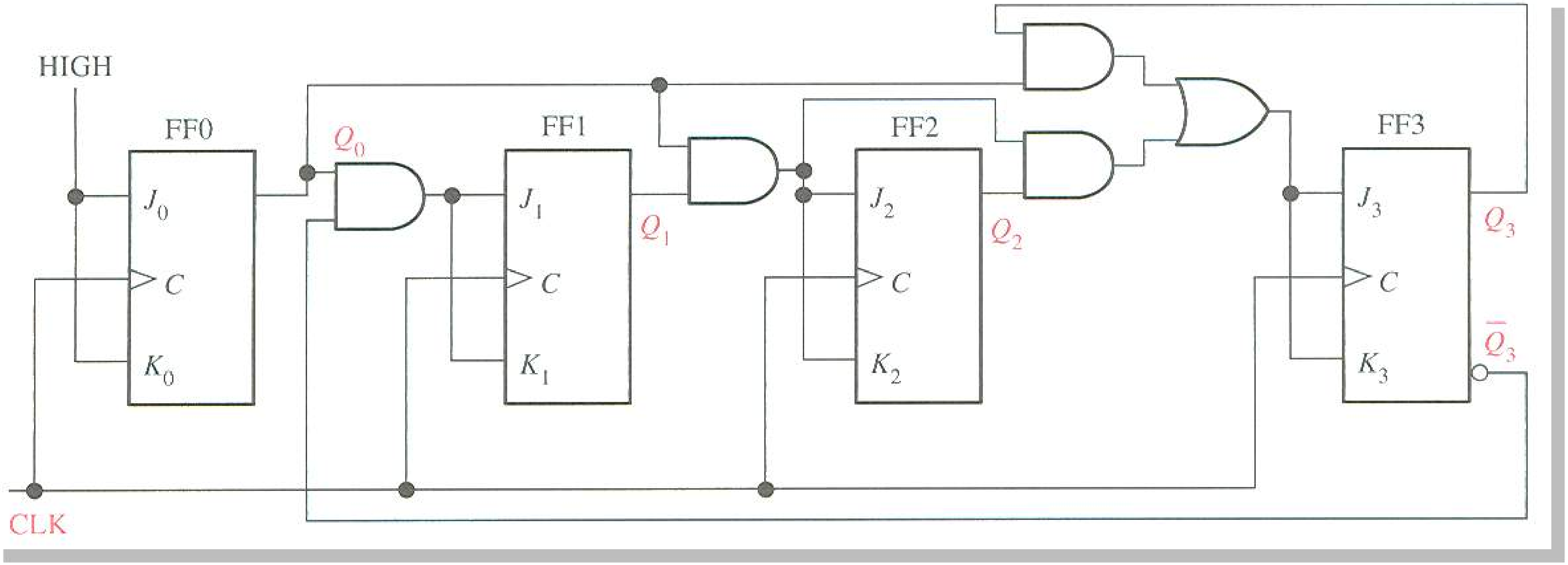
$$J_0 = Q_2Q_1Y + Q_2\overline{Q}_1\overline{Y} + \overline{Q}_2\overline{Q}_1Y + \overline{Q}_2Q_1\overline{Y} \qquad K_0 = \overline{Q}_2\overline{Q}_1\overline{Y} + \overline{Q}_2Q_1Y + Q_2\overline{Q}_1Y + Q_2Q_1\overline{Y}$$

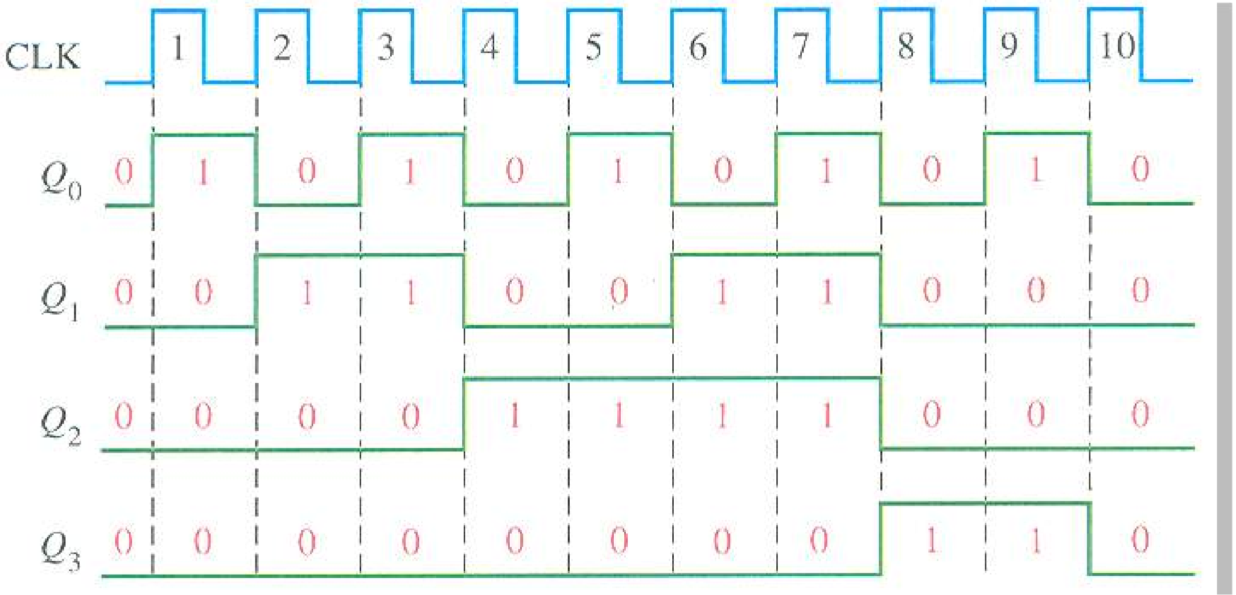
$$J_1 = \overline{Q}_2Q_0Y + Q_2Q_0\overline{Y} \qquad K_1 = \overline{Q}_2Q_0\overline{Y} + Q_2Q_0Y$$

$$J_2 = Q_1\overline{Q}_0Y + \overline{Q}_1\overline{Q}_0\overline{Y} \qquad K_2 = Q_1\overline{Q}_0\overline{Y} + \overline{Q}_1\overline{Q}_0Y$$

Step 6: The *J* and *K* equations are implemented with combinational logic, and the complete counter is shown in Figure 8–37.


$\overline{Q}_2$
Q_2
$\overline{Q}_1$
Q_1
$\overline{Q}_0$
Q_0
X
J_0
C
K_0
Q_0
$\overline{Q}_0$
J_1
C
K_1
Q_1
$\overline{Q}_1$
J_2
C
K_2
Q_2
$\overline{Q}_2$
CLK


HIGH
FF0
J_0
C
K_0
Q_0
FF1
J_1
C
K_1
Q_1
FF2
J_2
C
K_2
Q_2
FF3
J_3
C
K_3
Q_3
$\overline{Q}_3$
CLK


CLK 1 2 3 4 5 6 7 8 9 10
Q_0 0 1 0 1 0 1 0 1 0 1 0
Q_1 0 0 1 1 0 0 1 1 0 0 0
Q_2 0 0 0 0 1 1 1 1 0 0 0
Q_3 0 0 0 0 0 0 0 0 1 1 0

Reference:

[1] Thomas L. Floyd, “Digital Fundamentals” 11th edition, Prentice Hall.
[2] M. Morris Mano, “Digital Logic & Computer Design” Prentice Hall.
[3] Mixed contents from Vahid And Howard.

Thanks