Logic and Computer Design Fundamentals

Chapter 5 – Counters

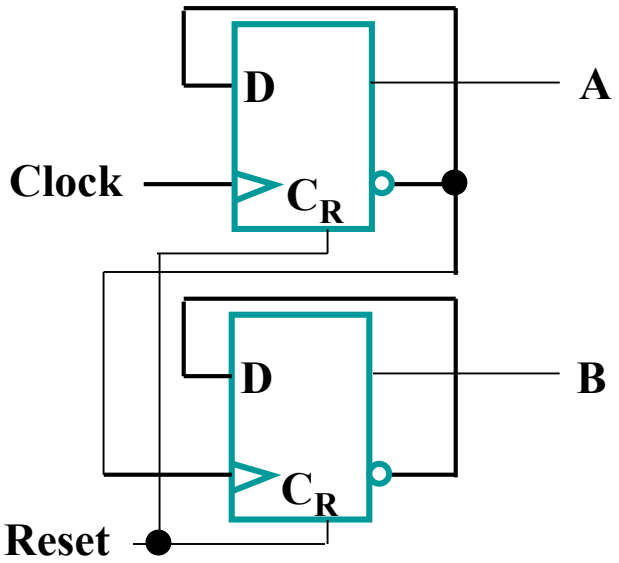
Charles Kime & Thomas Kaminski

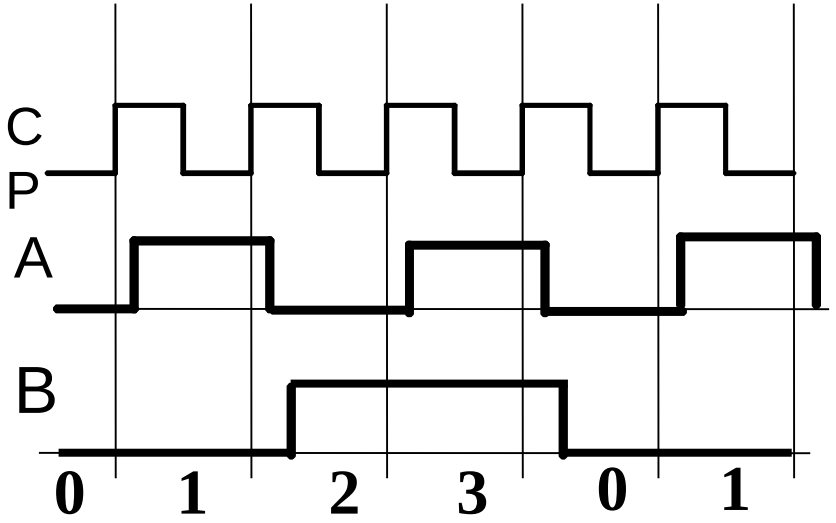
Counters

- Counters are sequential circuits which "count" through a specific state sequence. They can count up, count down, or count through other fixed sequences. Two distinct types are in common usage:
- Ripple Counters
  - Clock is connected to the flip-flop clock input on the LSB bit flip-flop
  - For all other bits, a flip-flop output is connected to the clock input, thus circuit is not truly synchronous
  - Output change is delayed more for each bit toward the MSB.
  - Resurgent because of low power consumption
- Synchronous Counters
  - Clock is directly connected to the flip-flop clock inputs
  - Logic is used to implement the desired state sequencing

Ripple Counter

- **How does it work?**
  - **When there is a positive edge on the clock input of A, A complements**
  - **The clock input for flip-flop B is the complemented output of flip-flop A**
  - **When flip A changes from 1 to 0, there is a positive edge on the clock input of B causing B to complement**



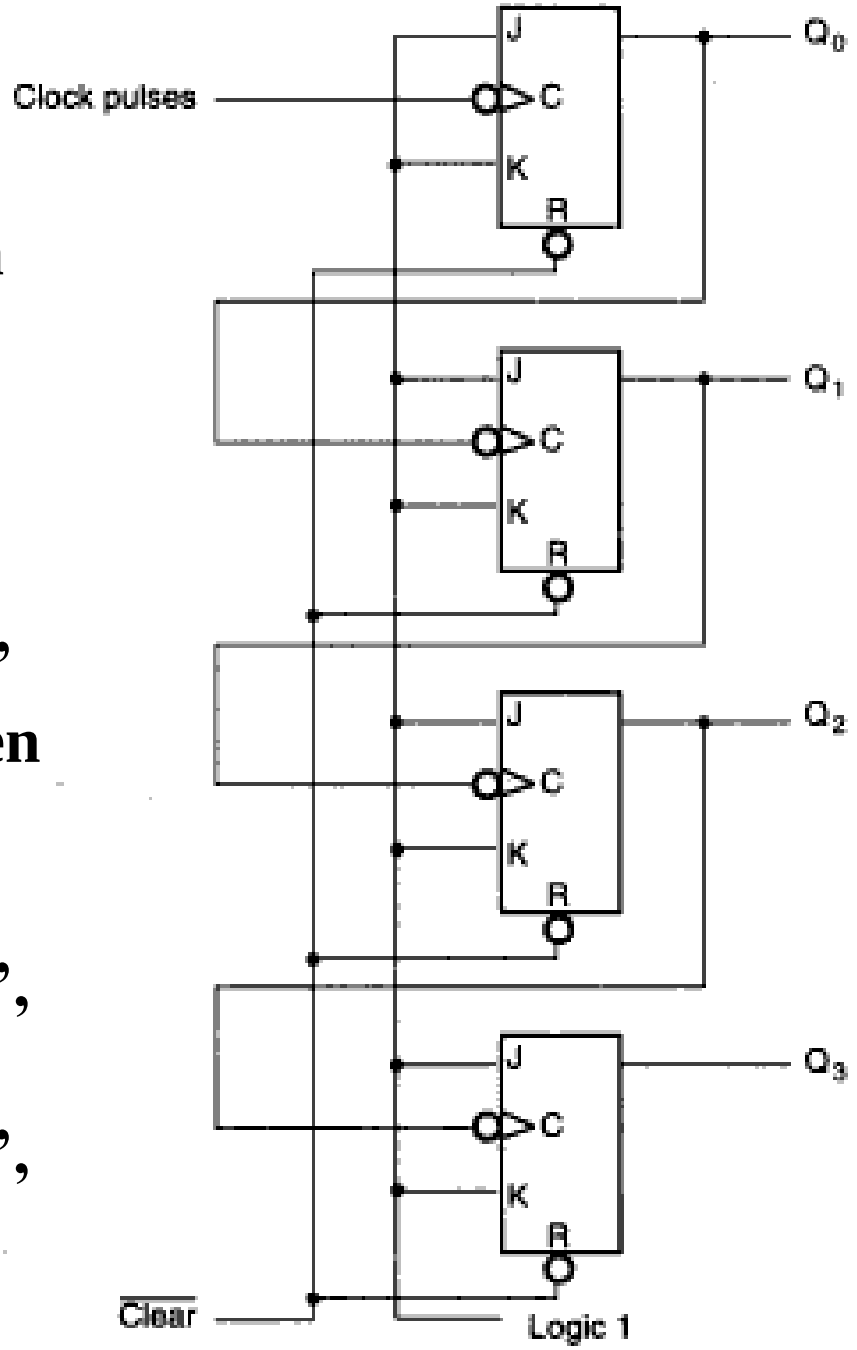


Ripple Counter (continued)

- **These circuits are called *ripple counters* because each edge sensitive transition (positive in the example) causes a change in the next flip-flop's state.**
- **The changes "ripple" upward through the chain of flip-flops, i. e., each transition occurs after a clock-to-output delay from the stage before.**

Ripple Counter: 4-bit Upward Counter

- Remember J-K toggles when both J and K are ‘1’
- The bubble on the clock means negative-edge triggered
- When Q0 goes from ‘1’ to ‘0’ (as if a negative-edge has been triggered to clock of Q1), Q1 toggles
- When Q1 goes from ‘1’ to ‘0’, Q2 toggles
- When Q2 goes from ‘1’ to ‘0’, Q3 toggles



Ripple Counter: 4-bit Upward Counter

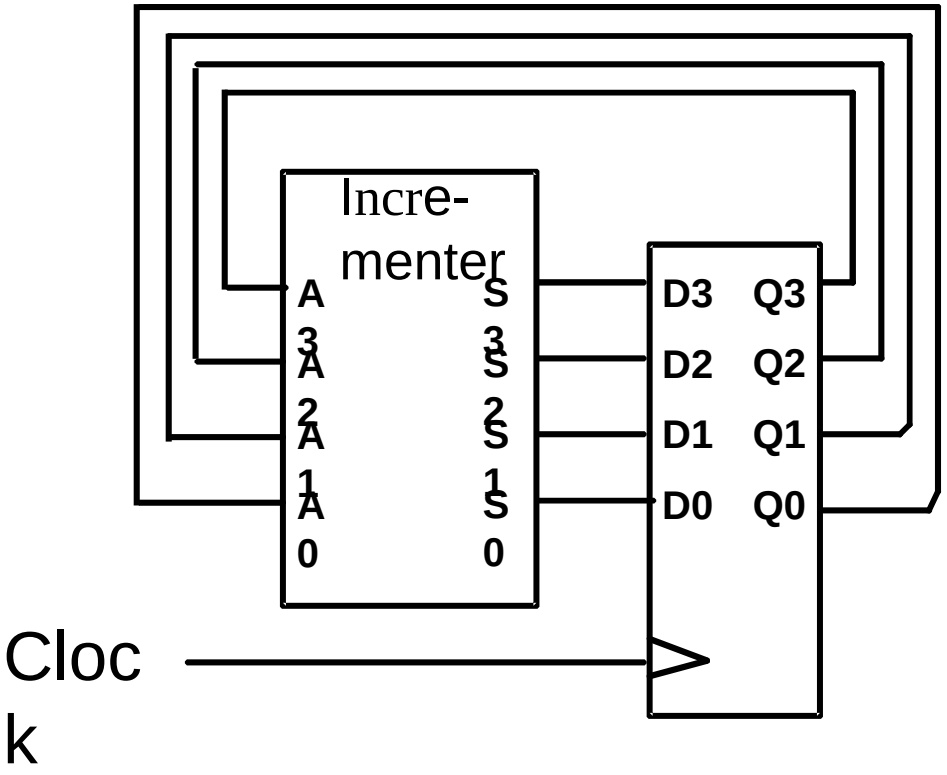
- When Q0 goes from ‘1’ to ‘0’ (as if a negative-edge has been triggered to clock of Q1), Q1 toggles
- When Q1 goes from ‘1’ to ‘0’, Q2 toggles
- When Q2 goes from ‘1’ to ‘0’, Q3 toggles
- What about the last sequence: how it goes from ‘1111’ to ‘0000’?

Upward Counting Sequence

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

Synchronous Counters

- **To eliminate the "ripple" effects, use a common clock for each flip-flop and a combinational circuit to generate the next state.**
- **For an up-counter, use an incrementer =>**



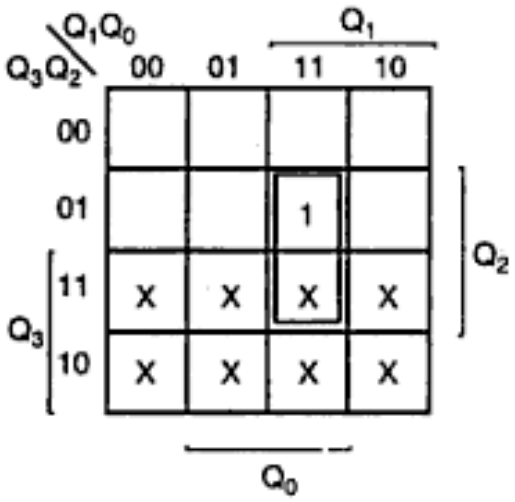
4-bit Synchronous Counter with J-K

(a) JK Flip-Flop

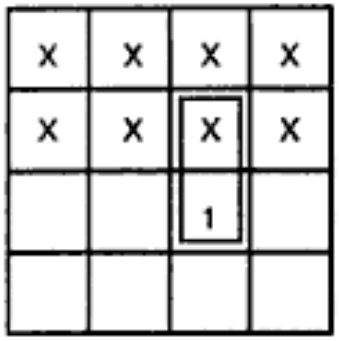
Q(t)	Q(t + 1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Present state				Next state				Flip-flop inputs							
Q_3	Q_2	Q_1	Q_0	Q_3	Q_2	Q_1	Q_0	J_{Q3}	K_{Q3}	J_{Q2}	K_{Q2}	J_{Q1}	K_{Q1}	J_{Q0}	K_{Q0}
0	0	0	0	0	0	0	1	0	×	0	×	0	×	1	×
0	0	0	1	0	0	1	0	0	×	0	×	1	×	×	1
0	0	1	0	0	0	1	1	0	×	0	×	×	0	1	×
0	0	1	1	0	1	0	0	0	×	1	×	×	1	×	1
0	1	0	0	0	1	0	1	0	×	×	0	0	×	1	×
0	1	0	1	0	1	1	0	0	×	×	0	1	×	×	1
0	1	1	0	0	1	1	1	0	×	×	0	×	0	1	×
0	1	1	1	1	0	0	0	1	×	×	1	×	1	×	1
1	0	0	0	1	0	0	1	×	0	0	×	0	×	1	×
1	0	0	1	1	0	1	0	×	0	0	×	1	×	×	1
1	0	1	0	1	0	1	1	×	0	0	×	×	0	1	×
1	0	1	1	1	1	0	0	×	0	1	×	×	1	×	1
1	1	0	0	1	1	0	1	×	0	×	0	0	×	1	×
1	1	0	1	1	1	1	0	×	0	×	0	1	×	×	1
1	1	1	0	1	1	1	1	×	0	×	0	×	0	1	×
1	1	1	1	0	0	0	0	×	1	×	1	×	1	×	1

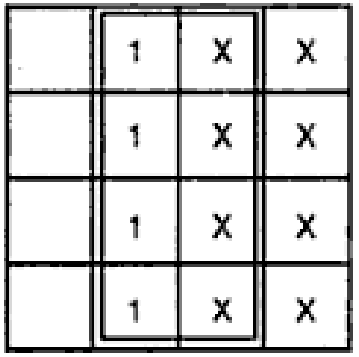
K-Maps



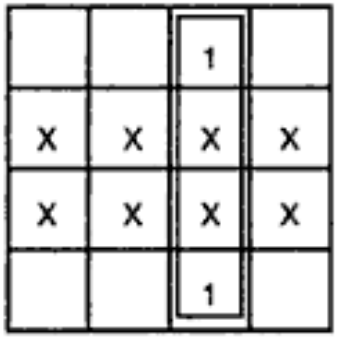
$J_{Q3} = Q_0Q_1Q_2$



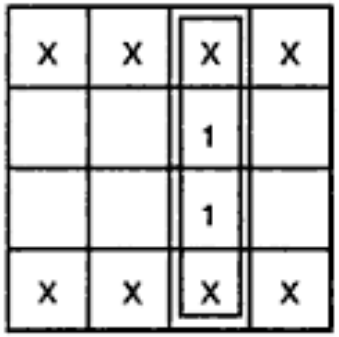
$K_{Q3} = Q_0Q_1Q_2$



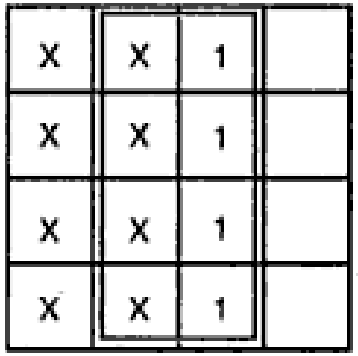
$J_{Q1} = Q_0$



$J_{Q2} = Q_0Q_1$



$K_{Q2} = Q_0Q_1$



$K_{Q1} = Q_0$

Equations

- **EN has been added to control the operation of the counter**

$$J_{Q0} = K_{Q0} = EN$$

$$J_{Q1} = K_{Q1} = Q_0 \cdot EN$$

$$J_{Q2} = K_{Q2} = Q_0 \cdot Q_1 \cdot EN$$

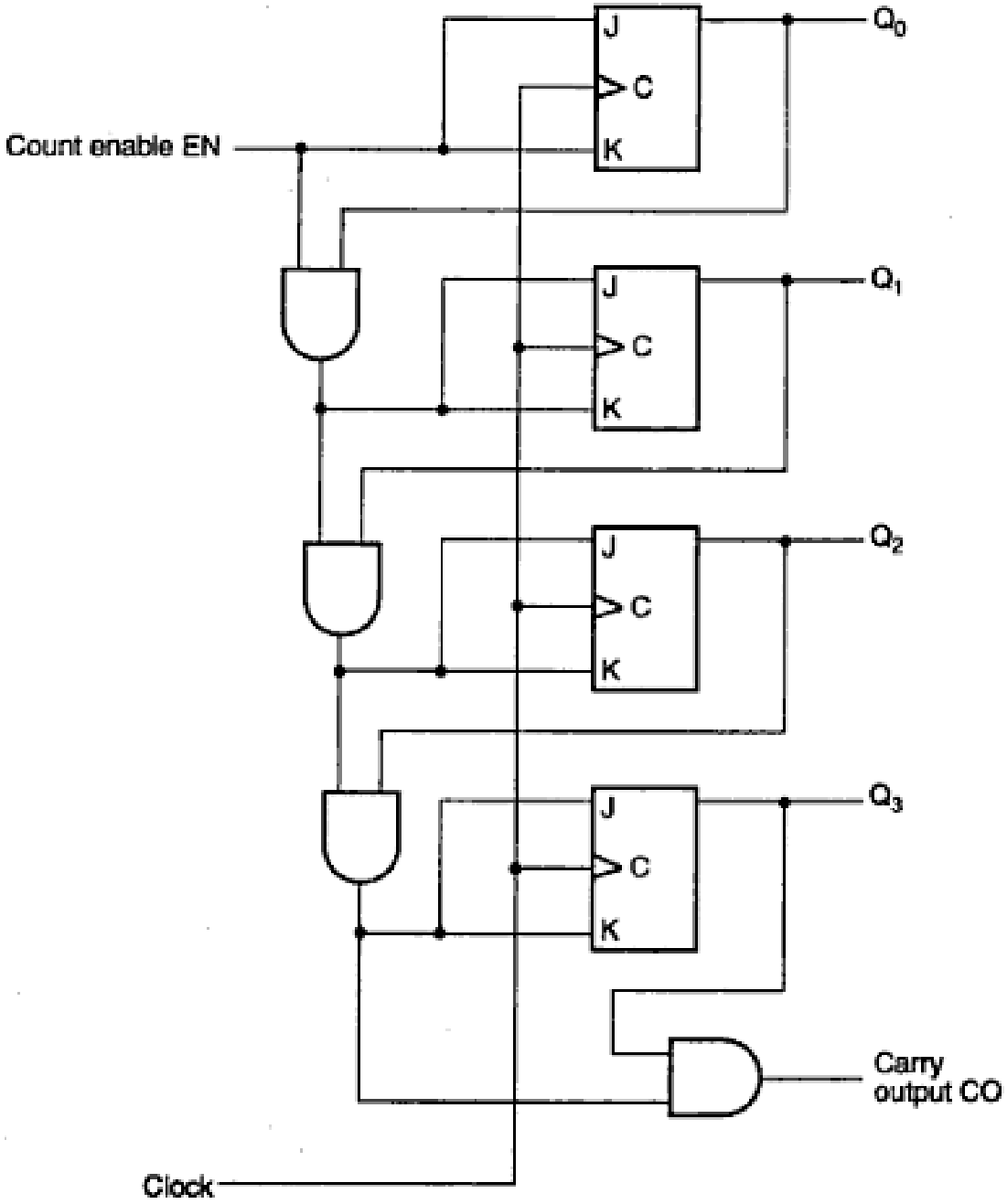
$$J_{Q3} = K_{Q3} = Q_0 \cdot Q_1 \cdot Q_2 \cdot EN$$

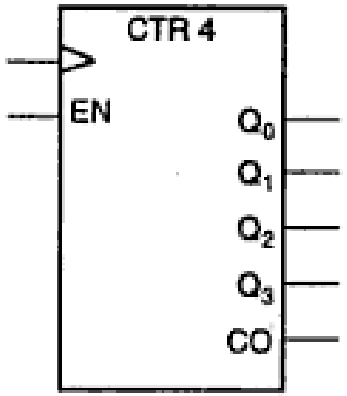
- **This relations could be generalized**

$$J_{Qi} = K_{Qi} = Q_0 \cdot Q_1 \cdot Q_2 \cdot \ldots \cdot Q_{i-1} \cdot EN$$

- **A Carry Output (CO) has been added so that the counter could be extended to more stages**
- **The implementation might be positive/negative edge triggered**

Logic Diagram





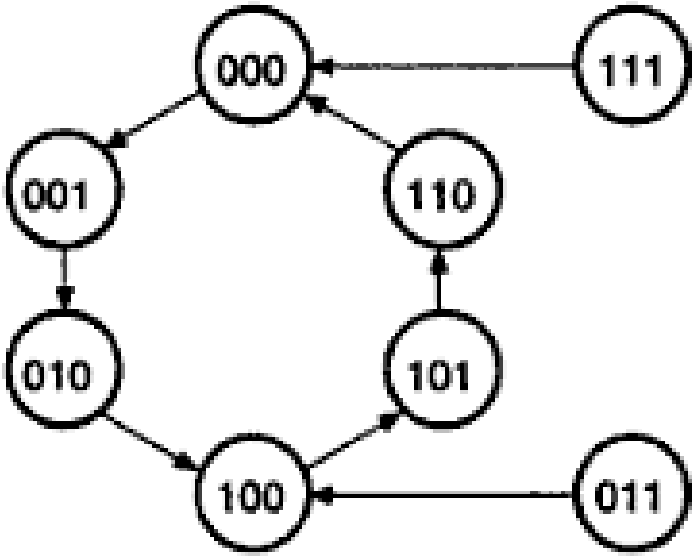
(b) Symbol

Arbitrary Counter: Problem Statement

- **Design an arbitrary counter that repeatedly follows the following sequence:**

  0, 1, 2, 4, 5, 6, 0, 1, 2, 4, 5, 6, 0. . .

Arbitrary Counter: State Diagram



(b) State diagram

Arbitrary Counter: State Table

TABLE 5-8
State Table and Flip-Flop Inputs for Counter

Present State			Next State			Flip-Flop Inputs					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	×	0	×	1	×
0	0	1	0	1	0	0	×	1	×	×	1
0	1	0	1	0	0	1	×	×	1	0	×
1	0	0	1	0	1	×	0	0	×	1	×
1	0	1	1	1	0	×	0	1	×	×	1
1	1	0	0	0	0	×	1	×	1	0	×

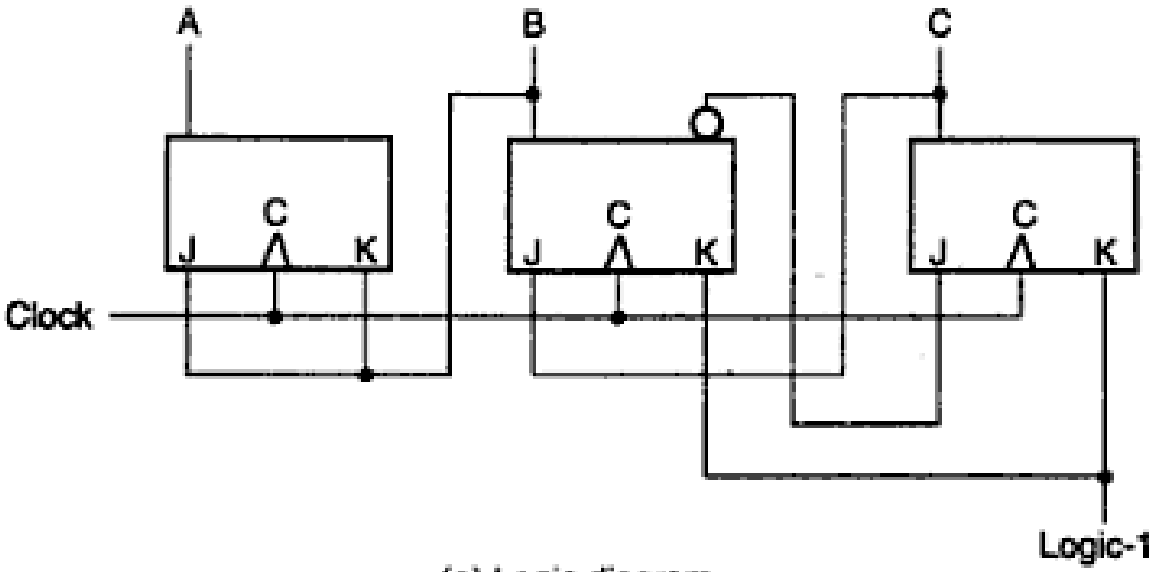
$$J_A = B \qquad K_A = B$$
$$J_B = C \qquad K_B = 1$$
$$J_C = \overline{B} \qquad K_C = 1$$

Flip-Flop Excitation Tables

(a) *JK* Flip-Flop

Q(t)	Q(t + 1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Arbitrary Counter: Logic Diagram



(a) Logic diagram

$$J_A = B \qquad K_A = B$$
$$J_B = C \qquad K_B = 1$$
$$J_C = \overline{B} \qquad K_C = 1$$