The Counters

By: Dr. Ehab A. H. AL-Hialy
Electrical Engineering Department

The Counters

Upon completion of the chapter, students should be able to:

1.1 Understand the basic concepts of asynchronous counter and synchronous counters, and the difference between them.

- 1.1.1 Draw circuit and Timing Diagram of Asynchronous Counters
- 1.1.2 Interpret the Operation and Application of an asynchronous counter
- 1.1.3 Draw circuit and timing diagram of synchronous counters
- 1.1.4 Interpret the operation and application of synchronous up/down counters.
- 1.1.5 Describe how the counters in 1.1.1 and 1.1.3 can be connected in cascade to produce higher mod
- 1.1.6 Explain the application of counters in Digital Clock

Introduction – COUNTERS

- A *counter* is a register that goes through a predetermined sequence of states upon the application of clock pulses.
  - Asynchronous counters
  - Synchronous counters
- **Asynchronous Counters** (or *Ripple counters*)
  - the clock signal (CLK) is only used to clock the first FF.
  - Each FF (except the first FF) is clocked by the preceding FF.
- **Synchronous Counters**,
  - the clock signal (CLK) is applied to all FF, which means that all FF shares the same clock signal,
  - thus the output will change at the same time.

Introduction –COUNTERS

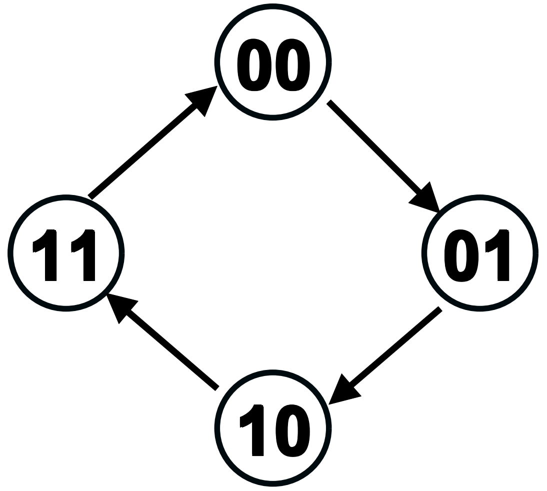
- Modulus (MOD) – the *number of states it counts* in a complete cycle before it goes back to the initial state.
- Thus, the number of flip-flops used depends on the MOD of the counter (ie; MOD-4 use 2 FF (2-bit), MOD-8 use 3 FF (3-bit), etc..)
- Example: MOD-4 Ripple/Asynchronous Up-Counter.

Asynchronous Counters

Asynchronous (Ripple) UP Counters

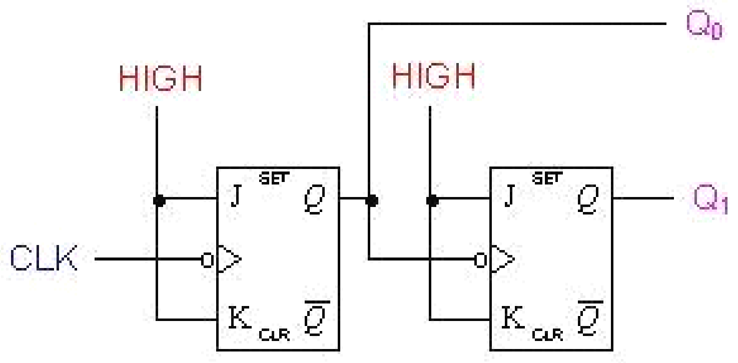
- The Asynchronous Counter that counts 4 number starts from 00→01→10→11 and back to 00 is called MOD-4 Ripple (Asynchronous) Up-Counter.
- **Next state table** and **state diagram**

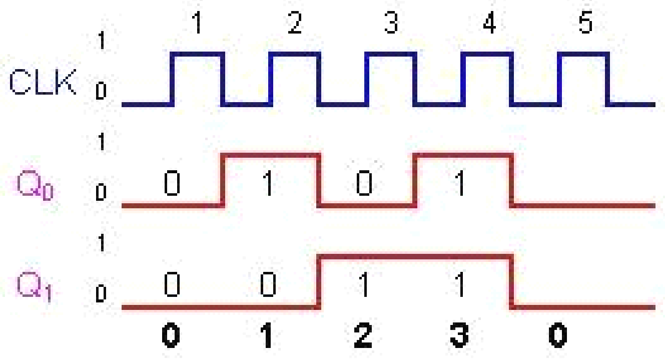
Present State	Next State
Q_1Q_0	Q_1Q_0
00	01
01	10
10	11
11	00


$$\begin{tikzcd} & 00 \arrow[dr] & \\ 11 \arrow[ur] & & 01 \arrow[dl] \\ & 10 \arrow[ul] & \end{tikzcd}$$

Asynchronous (Ripple) UP Counters

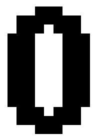
Figure 1.1 : MOD 4 Asynchronous Up Counter





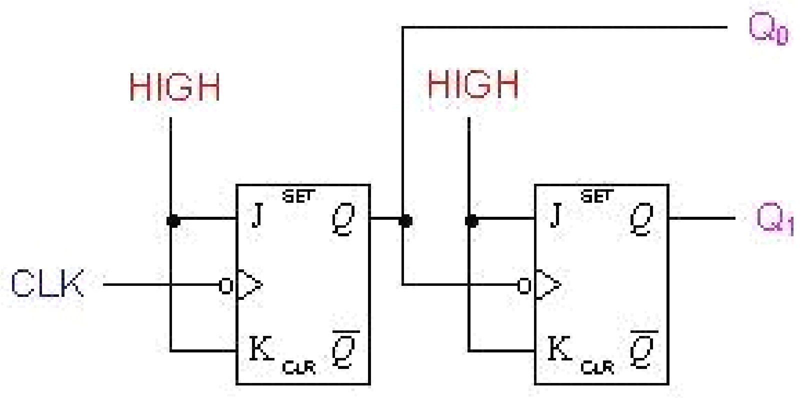
Waveform

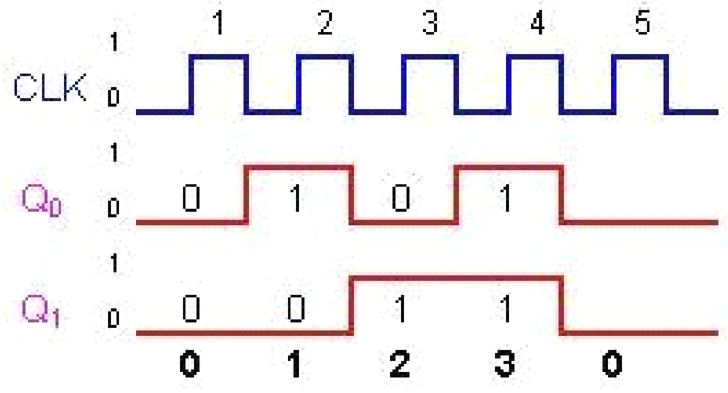
- **A two-bit asynchronous counter is shown on the left. The external clock is connected to the clock input of the first flip-flop (FF0) only. So, FF0 changes state at the falling edge of each clock pulse, but FF1 changes only when triggered by the falling edge of the Q output of FF0.**
- **Note that for simplicity, the transitions of Q0, Q1 and CLK in the timing diagram above are shown as simultaneous even though this is an asynchronous counter. Actually, there is some small delay between the CLK, Q0 and Q1 transitions.**



Asynchronous (Ripple) UP Counters

Figure 1.1 : MOD 4 Asynchronous Up Counter





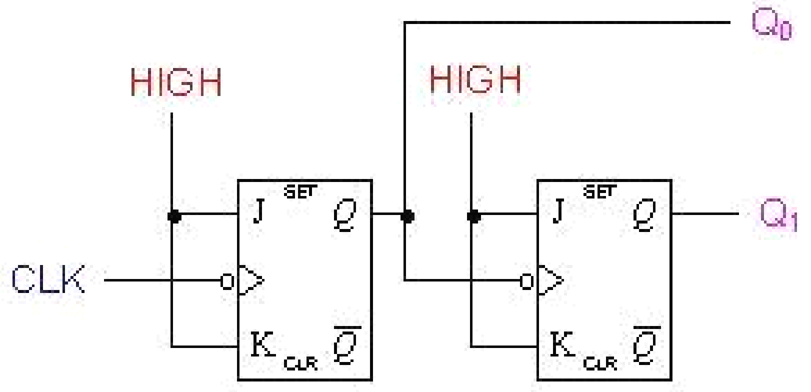
Waveform

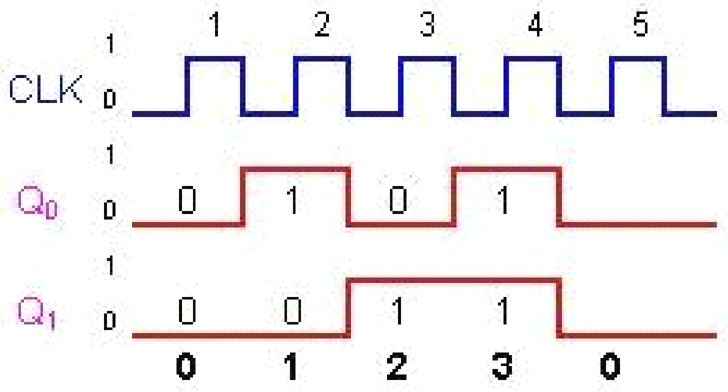
- **Because of the inherent propagation delay through a flip-flop, the transition of the input clock pulse and a transition of the Q output of FF0 can never occur at exactly the same time. Therefore, the flip-flops cannot be triggered simultaneously, producing an asynchronous operation.**
- **The 2-bit ripple counter circuit shown has four different states, each one corresponding to a count value. Similarly, a counter with *n* flip-flops can have *2 to the power n* states. (2^n) The number of states in a counter is known as its mod (modulo) number. Thus a 2-bit counter is a mod-4 counter.**

0

Asynchronous (Ripple) UP Counters

Figure 1.1 : MOD 4 Asynchronous Up Counter





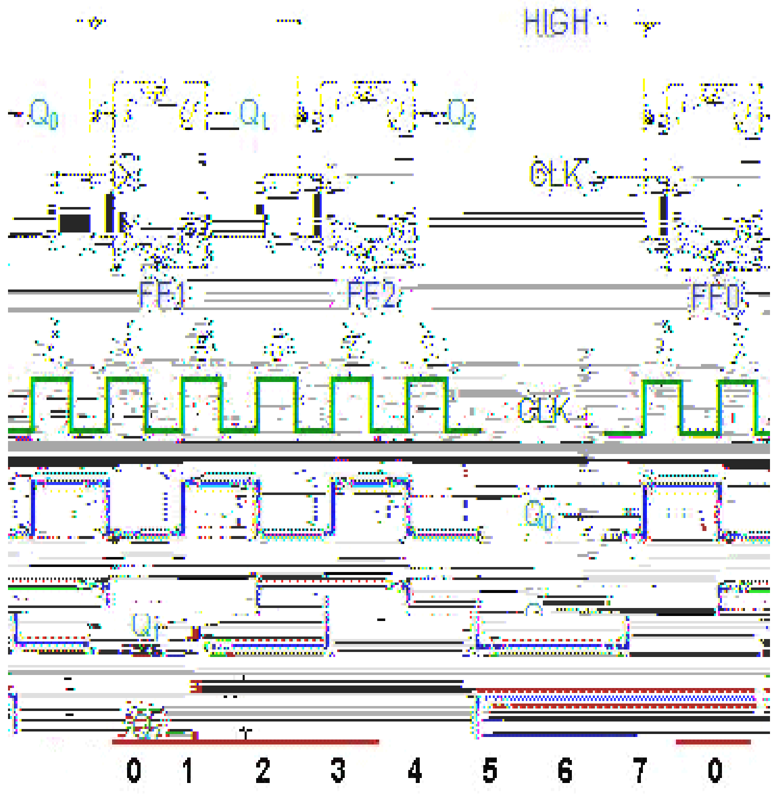
Waveform

- **Usually, all the CLEAR inputs are connected together, so that a single pulse can clear all the flip-flops before counting starts. The clock pulse fed into FF0 is rippled through the other counters after propagation delays, like a ripple on water, hence the name Ripple Counter**

- **A mod-n counter may also described as a divide-by-*n* counter. This is because the most significant flip-flop (the furthest flip-flop from the original clock pulse) produces one pulse for every *n* pulses at the clock input of the least significant flip-flop (the one triggers by the clock pulse).**

MOD 8 Asynchronous Up Counter

Figure 1.2 : MOD 8 Asynchronous Up Counter



Waveform

- **The following is a three-bit asynchronous binary counter and its timing diagram for one cycle.**
- **It works exactly the same way as a two-bit asynchronous binary counter mentioned above, except it has eight**

  states due to the third flip-flop.

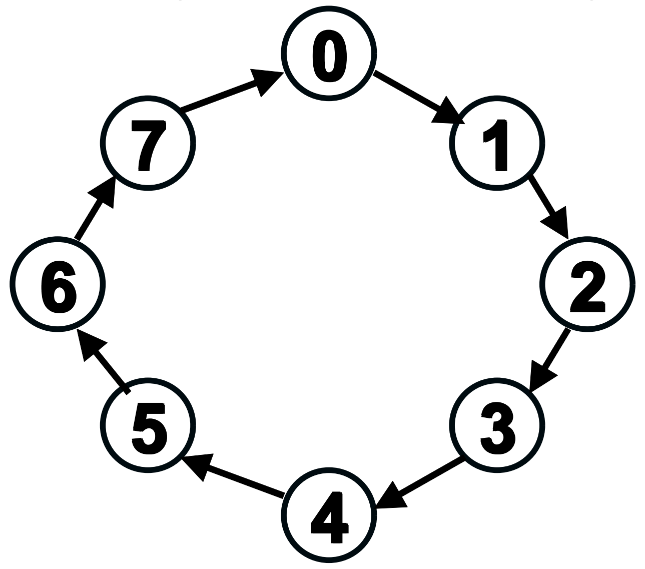
0

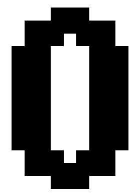
MOD 8 Asynchronous Up Counter

Figure 1.3a Next State Table

Present State	Next State
CBA	CBA
000	001
001	010
010	011
011	100
100	101
101	110
110	111
111	000

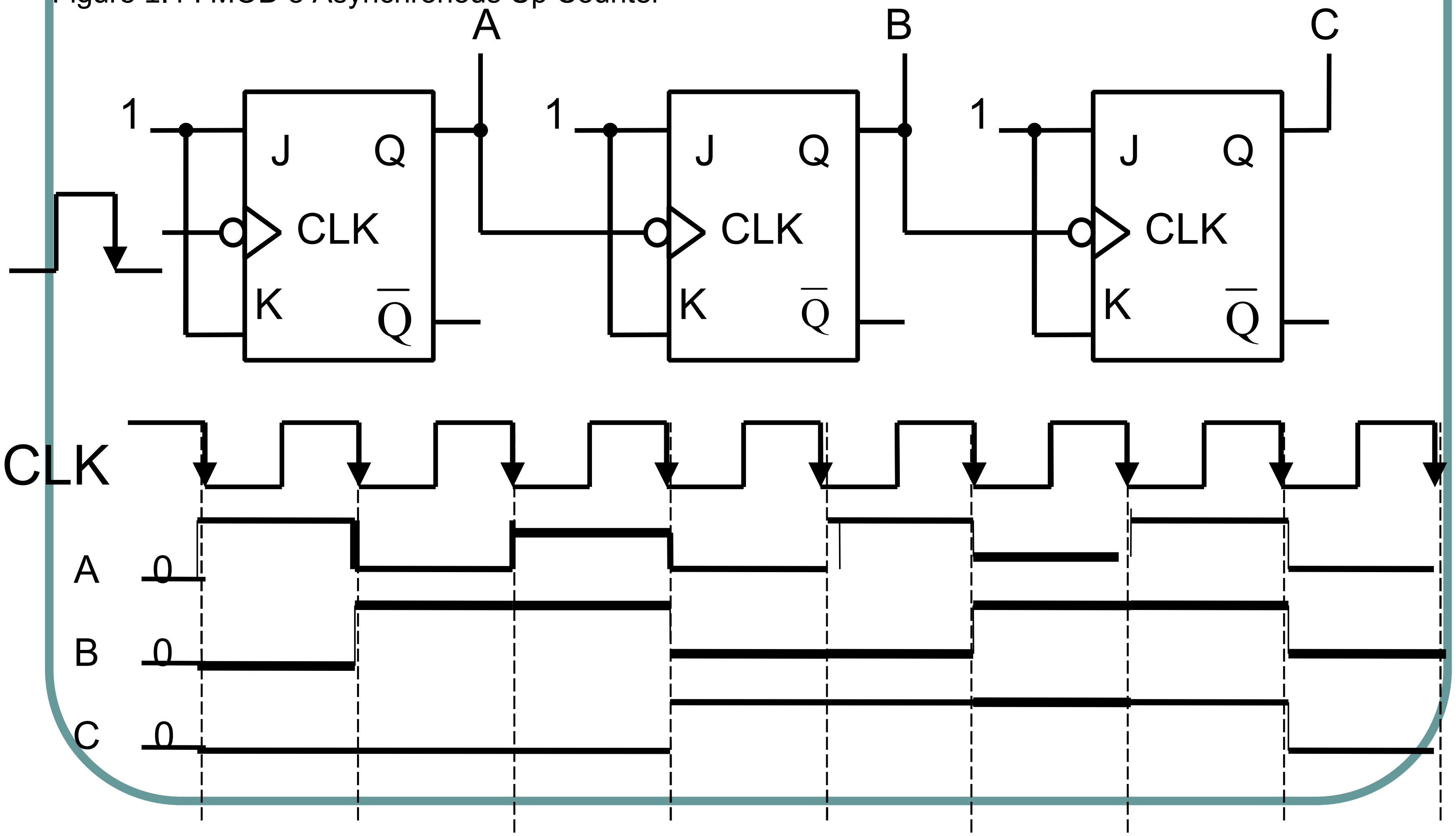
Figure 1.3b State Diagram





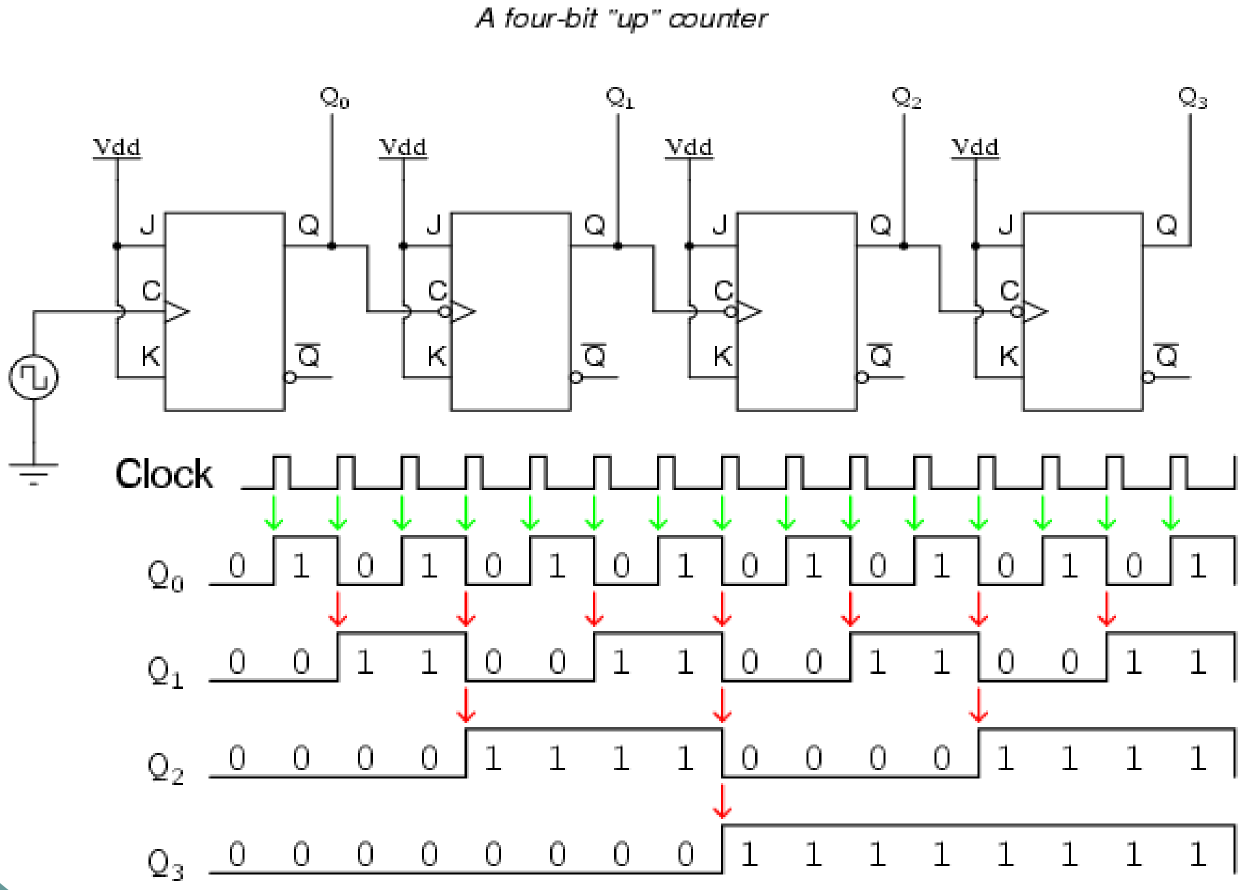
Exercise :

Figure 1.4 : MOD 8 Asynchronous Up Counter



MOD 16 Asynchronous Up counter – (Negative Triggered)

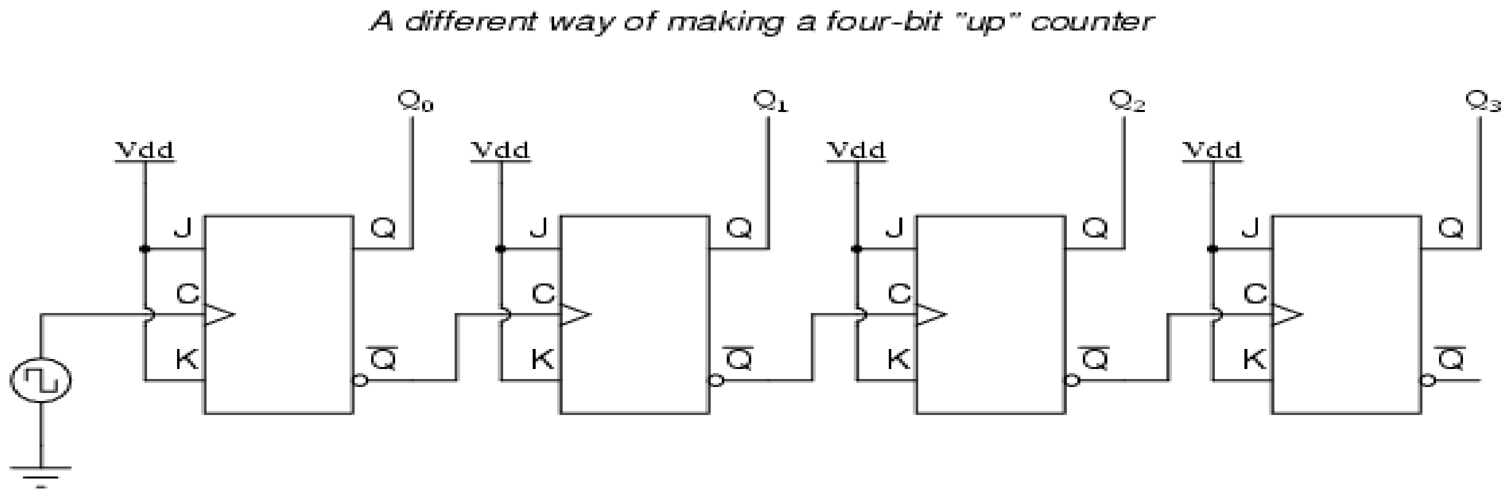
Figure 1.5 : MOD 16 Asynchronous Up Counter



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

MOD 16 Asynchronous Up counter (Positive Triggered)

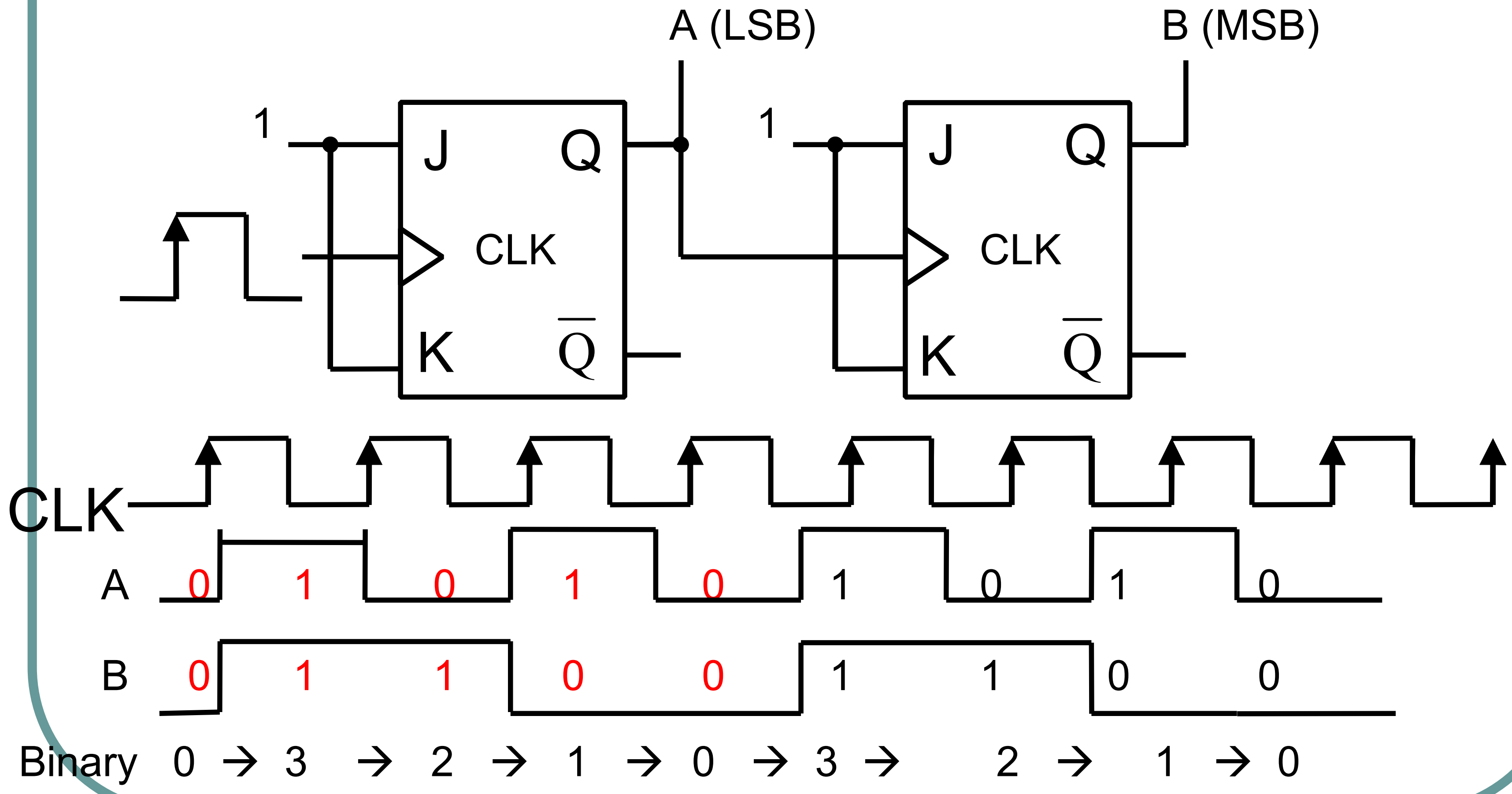
Figure 1.6 : MOD 16 Asynchronous Up Counter



- Exercise : Draw a MOD 16 Asynchronous DOWN Counter
  (Negative Triggered) :

Asynchronous DOWN Counter

Figure 1.7 : MOD 4 or 2-bit Asynchronous down counter



Asynchronous Counters

- Exercise:
  - Design a MOD-4 ripple down-counter
  - Design a MOD-8 ripple down counter using negative triggered.
  - Design a MOD-16 ripple down counter using positive triggered.

Asynchronous Counters (MOD $\neq 2^N$)

- So far, we have design the counters with MOD number equal to 2^N, where N is the number of bit (N = 1,2,3,4….) (also correspond to number of FF)
- Thus, the counters are limited on for counting MOD-2, MOD4, MOD-8, MOD-16 etc..
- The question is how to design a MOD-5, MOD-6, MOD-7, MOD-9 which is not a MOD-2^N **(MOD $\neq 2^N$)** ?
- MOD-6 counters will count from 0_{10} (000_2) to 5_{10}(101_2) and after that will recount back to 0_{10} (000_2) continuously.

Asynchronous Counters (MOD $\neq 2^N$)

MOD-6 ripple up-counter (MOD $\neq 2^N$)

Present St.	Next St.
CBA	CBA
000	001
001	010
010	011
011	100
100	101
101	000(110)

Figure 1.8a :Next State Table

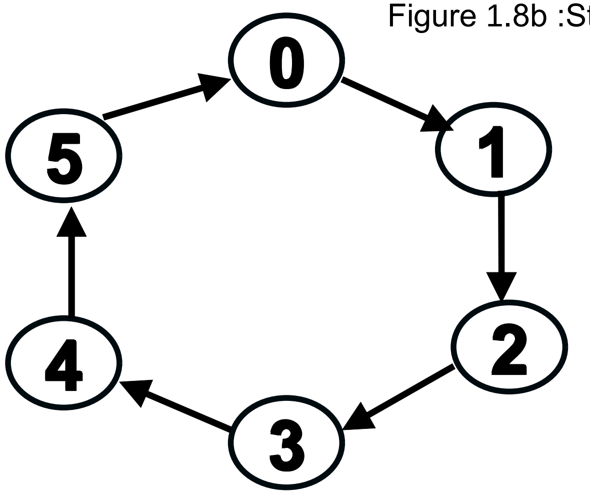


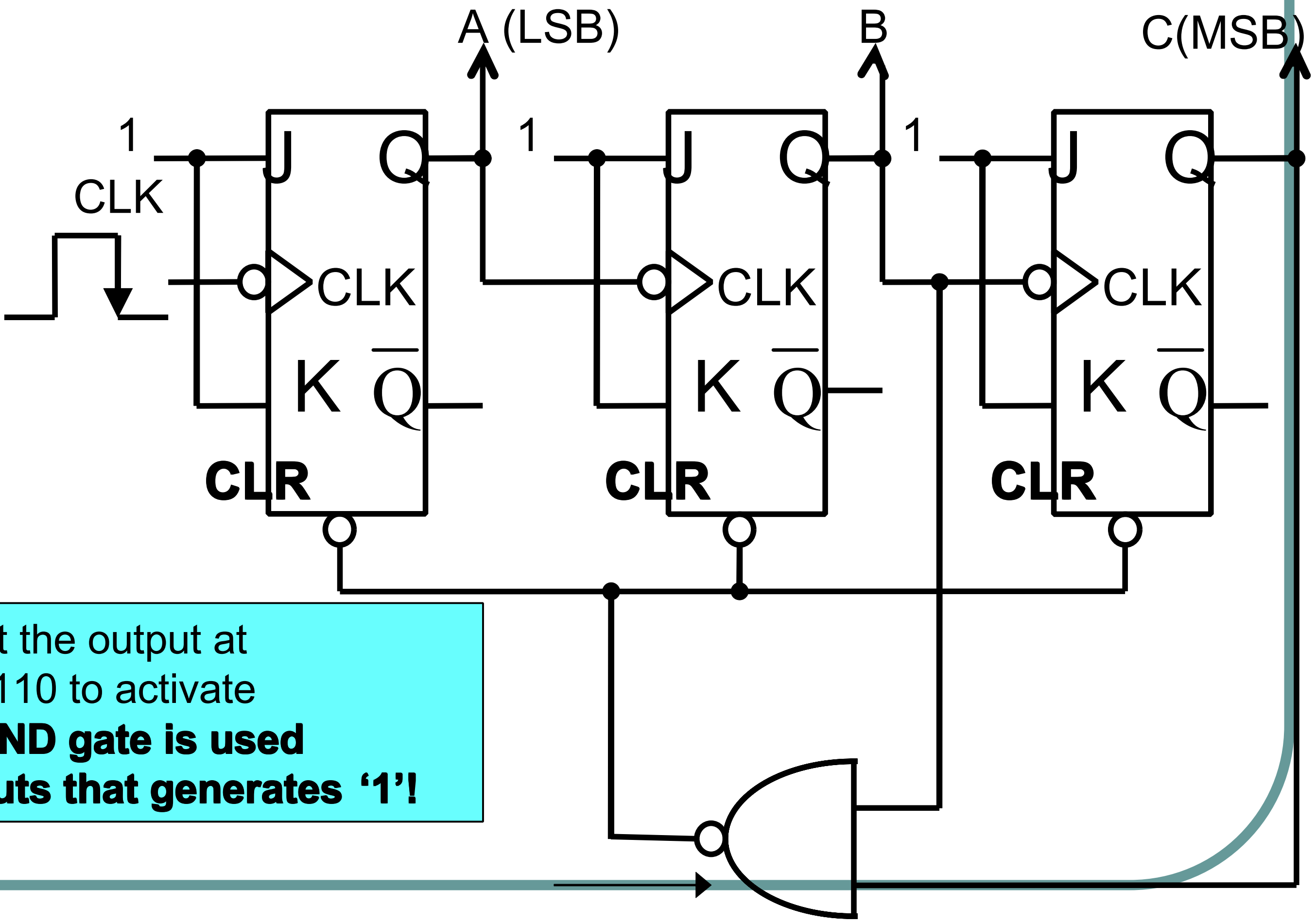
Figure 1.8b :State Diagram

Reset the state to 000_2 when 110_2 is detected

Asynchronous Counters (MOD $\neq 2^N$)

- Circuit diagram for MOD-6 ripple up-counter (MOD $\neq 2^N$)

Present St.	Next St.
CBA	CBA
000	001
001	010
010	011
011	100
100	101
101	000(110)



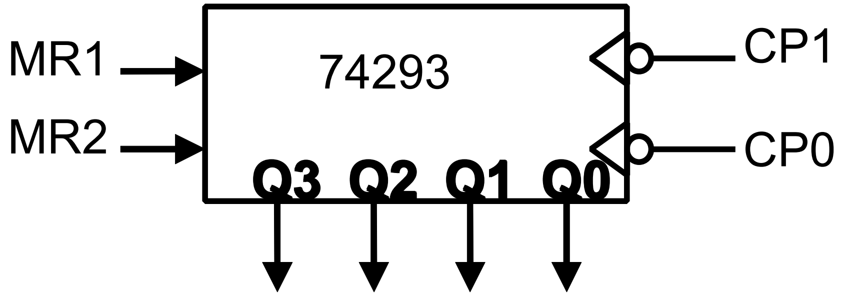
Detect the output at CBA=110 to activate CLR. **NAND gate is used to detect outputs that generates '1'!**

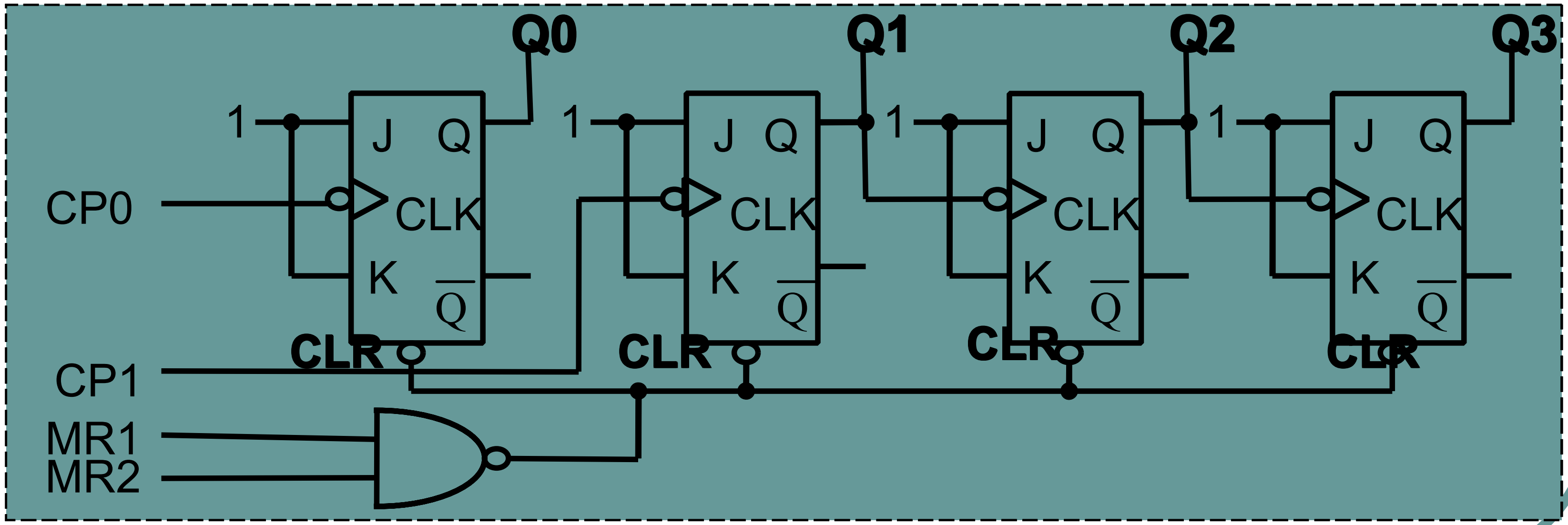
Asynchronous Counters (MOD $\neq 2^N$)

Exercise : Draw MOD-5 Ripple Down-counter and Up-counter (MOD $\neq 2^N$)

IC for Asynchronous counters (IC 74293)

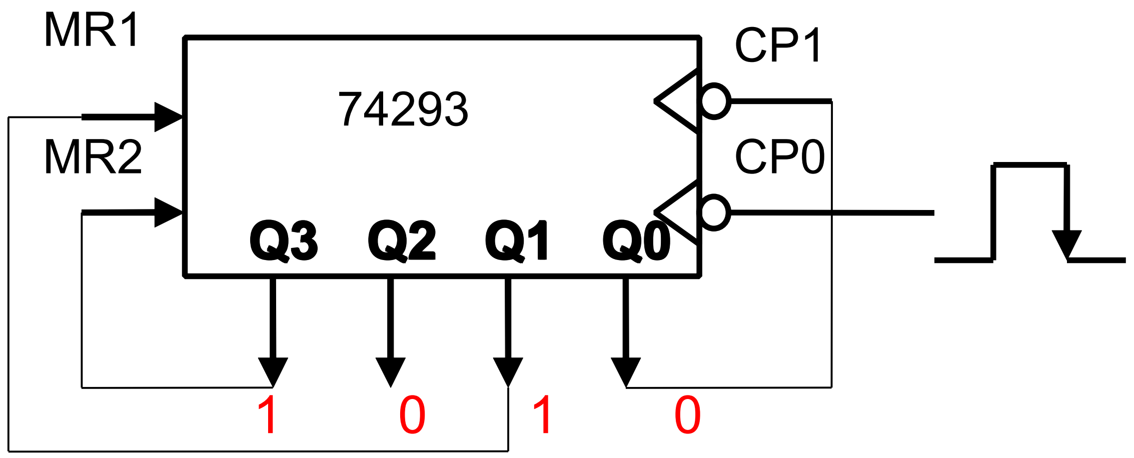
- 74293 IC for Asynchronous counter with Reset (MR1 and MR2)





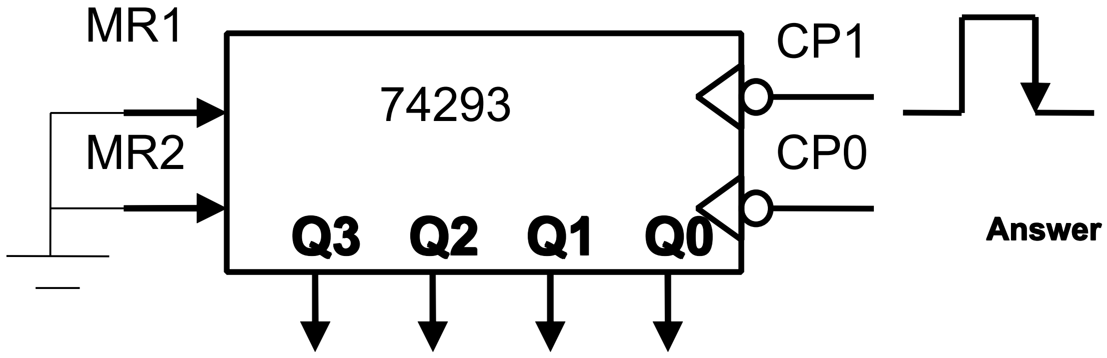
IC for Asynchronous counters (IC 74293)

- Using 74293 IC to design MOD ≤ 16 Asynchronous UP-Counter!
- Exercise:
  - Use 74293 IC to design MOD-10 ripple up-counter

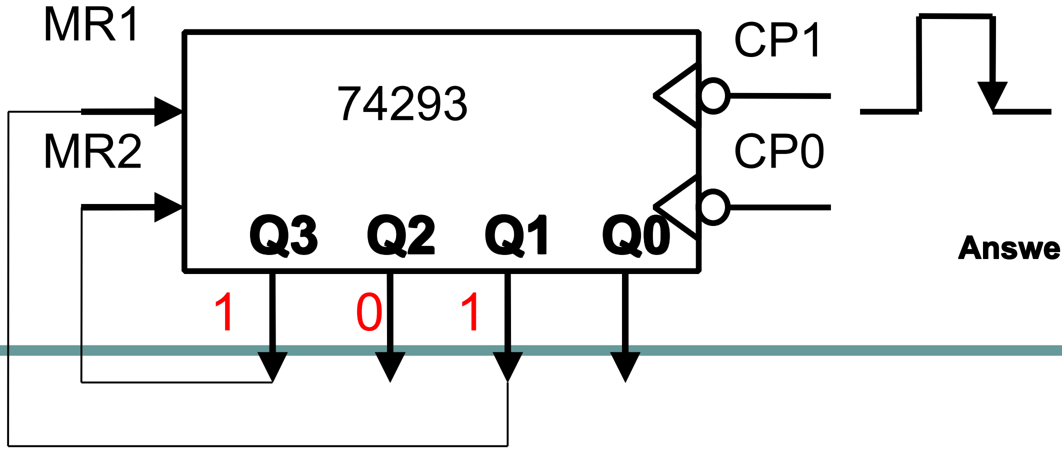


IC for Asynchronous counters (IC 74293)

- Exercise:
  - Determine the MOD for each configuration shown below?



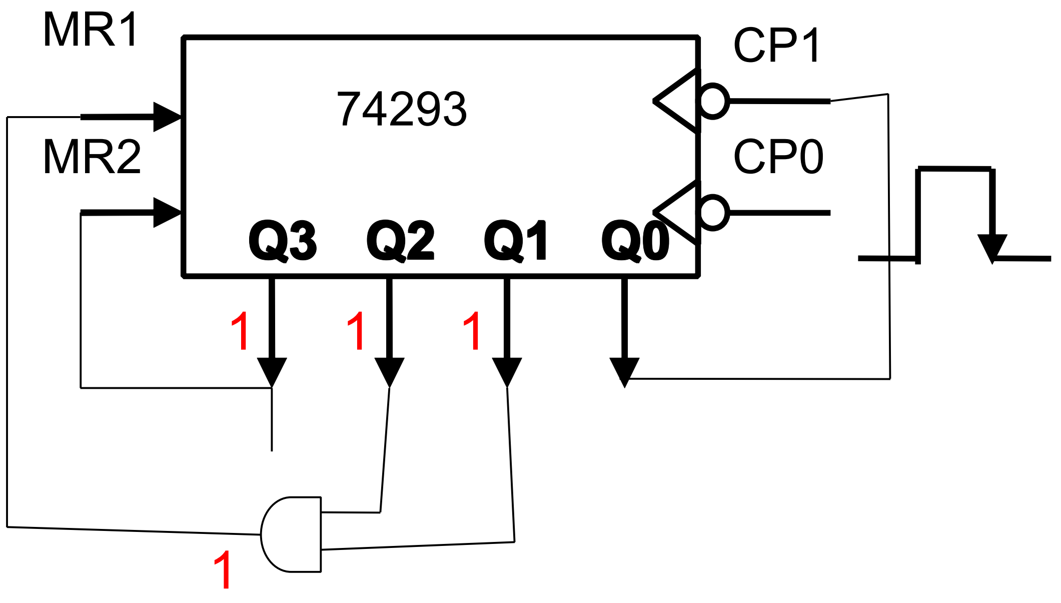
Answer : MOD 8



Answer : MOD 5

IC for Asynchronous counters (IC 74293)

- Determine the MOD for configuration shown below?



Answer : MOD 14

IC for Asynchronous counters (IC 74293)

CASCADE connection to produce Higher Mod

Exercise : Design Asynchronous counters MOD-60 using IC 74293.

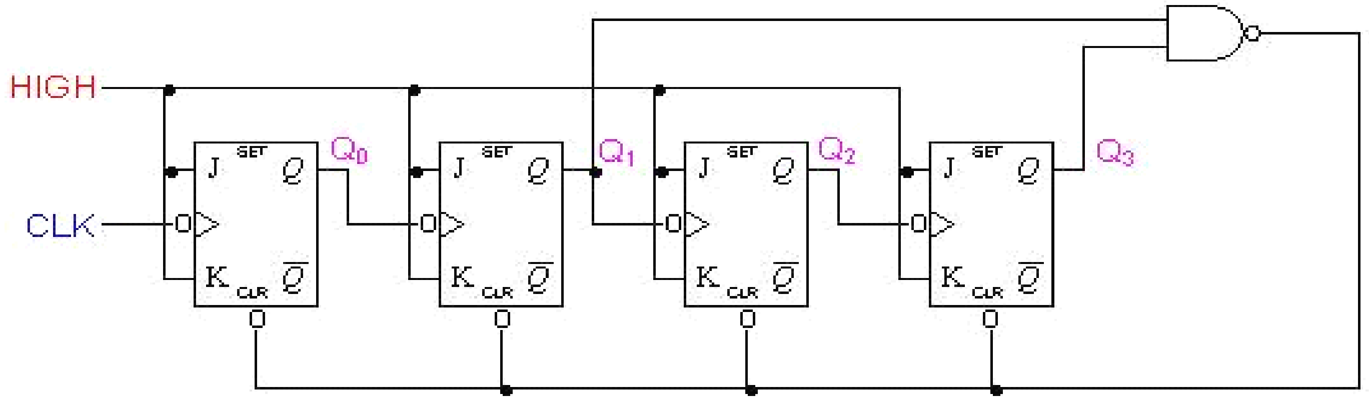
Solution : Discuss with your Lecturer in class.

Exercise : i. Design Asynchronous counters MOD-55 using IC 74293.

ii. Design Asynchronous counters MOD-1000 using IC 74293.

Asynchronous Decade Counters

Figure 1.3 : Asynchronous Decade Counter

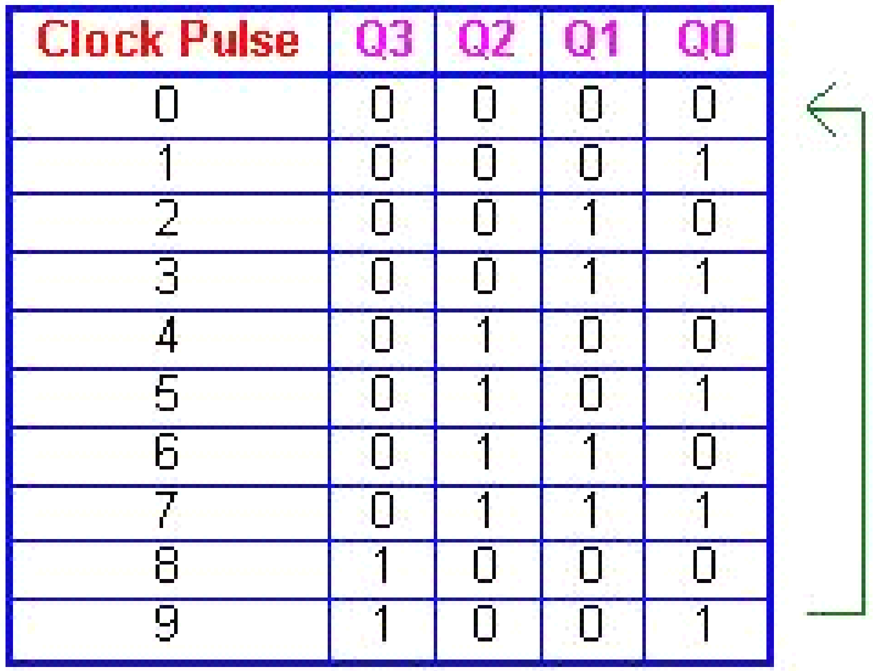


- **The binary counters previously introduced have two to the power *n* states. But counters with states less than this number are also possible. They are designed to have the number of states in their sequences, which are called truncated sequences. These sequences are achieved by forcing the counter to recycle before going through all of its normal states.**

- **A common modulus for counters with truncated sequences is ten. A counter with ten states in its sequence is called a *decade counter*. The circuit below is an implementation of a decade counter.**

Asynchronous Decade Counters

- The sequence of the decade counter is shown in the table below:



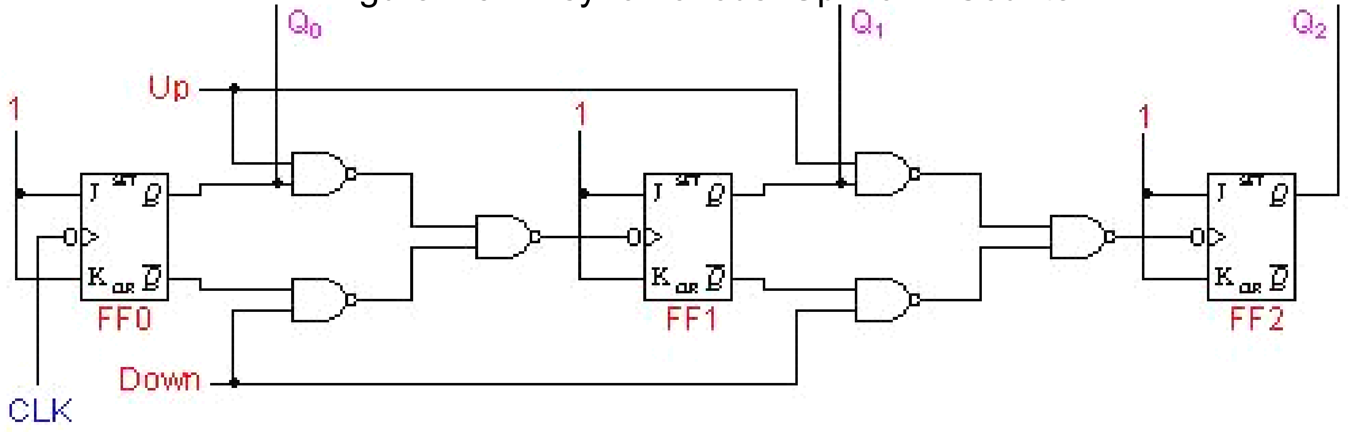
Clock Pulse	Q3	Q2	Q1	Q0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Figure 1.4 : True Table Asynchronous Decade Counter

- Once the counter counts to ten (1010), all the flip-flops are being cleared. Notice that only Q1 and Q3 are used to decode the count of ten. This is called partial decoding, as none of the other states (zero to nine) have both Q1 and Q3 HIGH at the same time.

Asynchronous Up-Down Counters

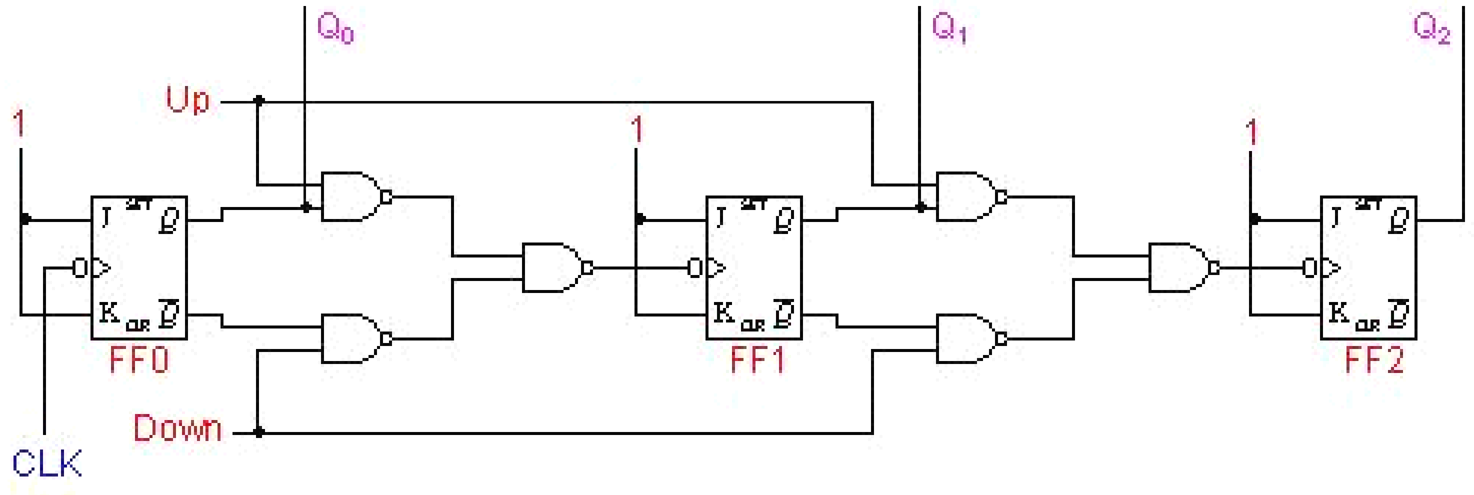
Figure 1.5 : Asynchronous Up-Down Counter



- In certain applications a counter must be able to count both up and down. The circuit below is a 3-bit up-down counter. It counts up or down depending on the status of the control signals UP and DOWN. When the UP input is at 1 and the DOWN input is at 0, the NAND network between FF0 and FF1 will gate the non-inverted output (Q) of FF0 into the clock input of FF1. Similarly, Q of FF1 will be gated through the other NAND network into the clock input of FF2. Thus the counter will count up.

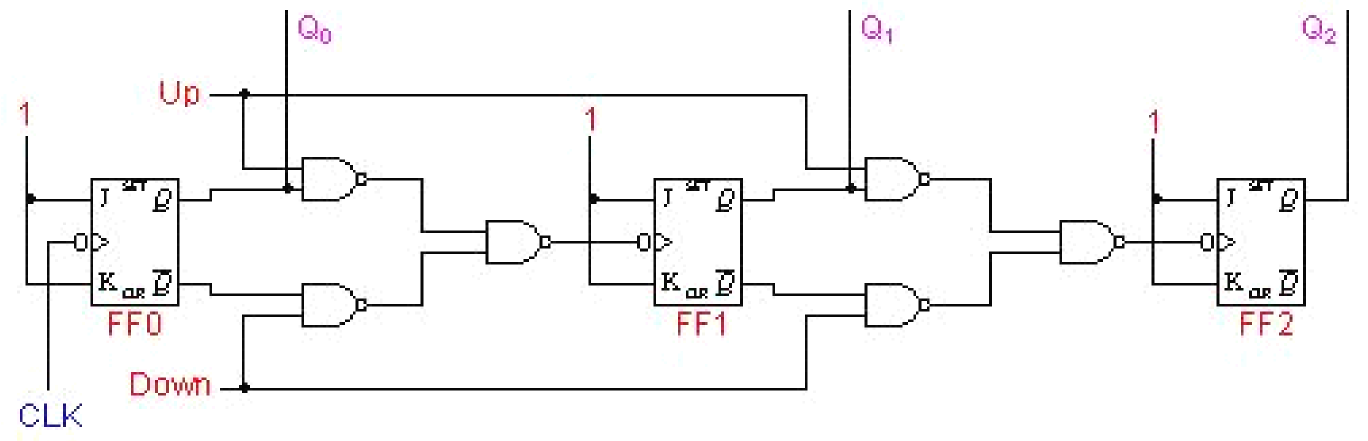
Asynchronous Up-Down Counters

Figure 1.5 : Asynchronous Up-Down Counters



- **When the control input UP is at 0 and DOWN is at 1, the inverted outputs of FF0 and FF1 are gated into the clock inputs of FF1 and FF2 respectively. If the flip-flops are initially reset to 0's, then the counter will go through the following sequence as input pulses are applied.**

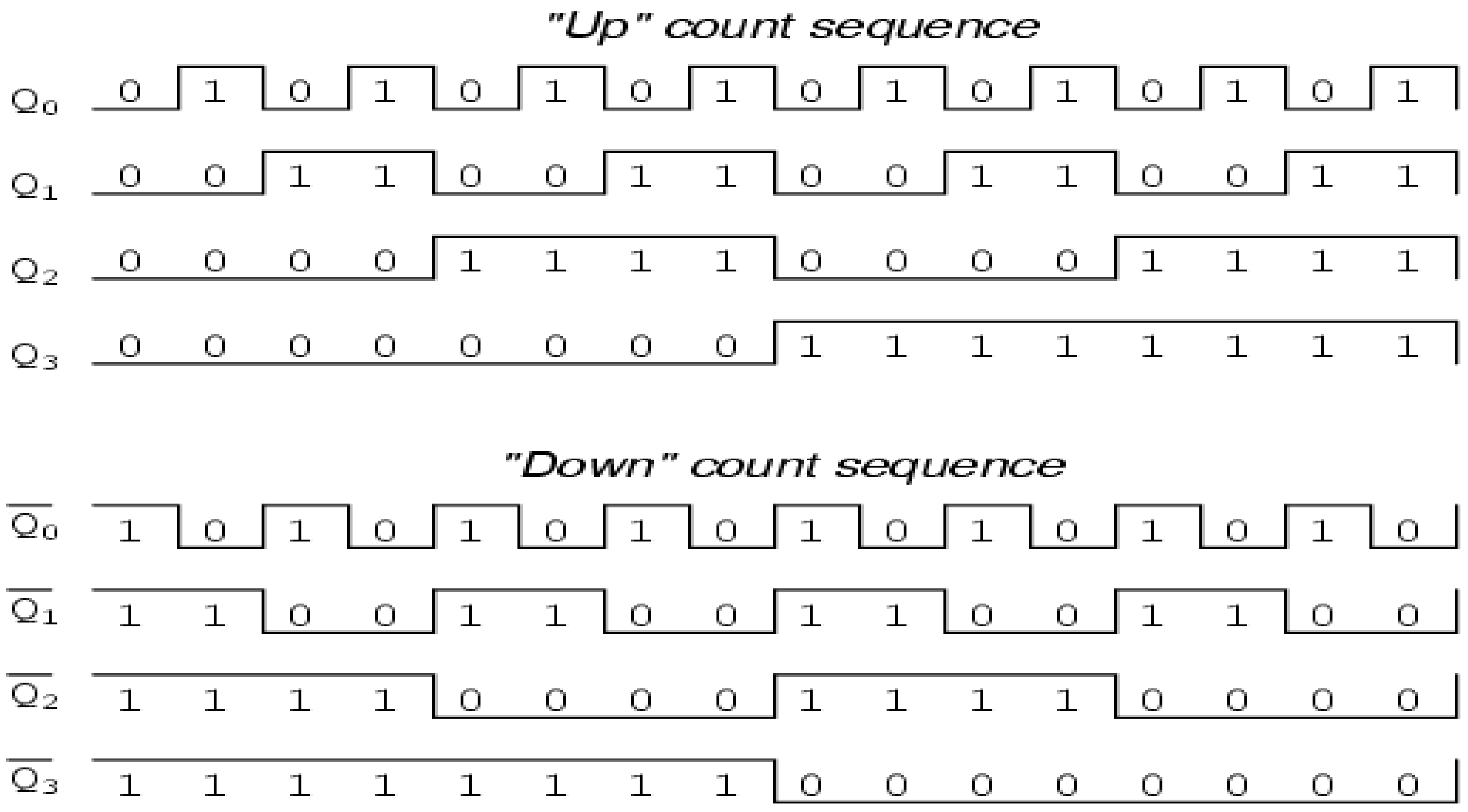
Asynchronous Up-Down Counters



- **Notice that an asynchronous up-down counter is slower than an up counter or a down counter because of the additional propagation delay introduced by the NAND networks.**

Asynchronous Up-Down Counters

Figure 1.3 : Asynchronous Up-Down Counters Waveform For 4 Bit Up-Down Counter



"Up" count sequence
Q0 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
Q1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1
Q2 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1
Q3 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
"Down" count sequence
Q̄0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
Q̄1 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0
Q̄2 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0
Q̄3 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0

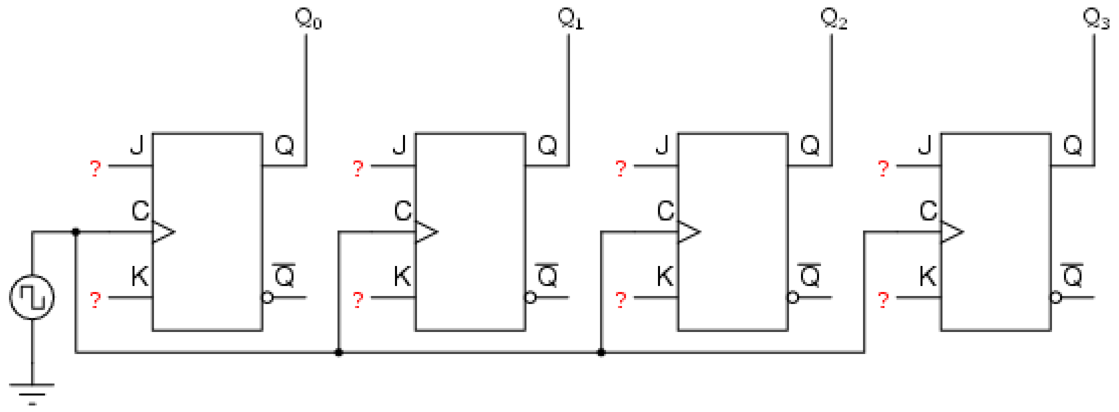
Asynchronous Counters

- Disadvantages of Asynchronous Counters:-
  - Propagation delay is severe for larger MOD of counters, especially at the MSB.
  - Existence of ‘glitch’ is inevitable for MOD $\neq 2^N$ counters.
  - Cannot design random counters (i.e:- to design circuit that counts numbers in these sequence 5→6→7→2→3→1→5→6→7→2→3→1→5→6….)
- Solution, use *SYNCHRONOUS COUNTERS*.

Synchronous Counters

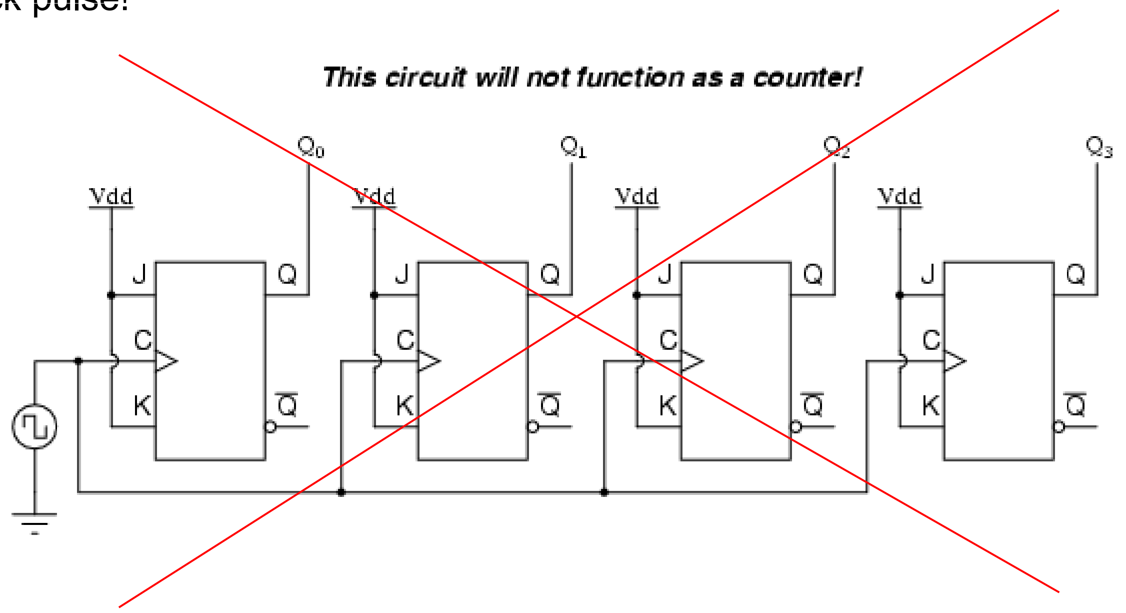
Synchronous Counters

- A *synchronous counter*, in contrast to an *asynchronous counter*, is one whose output bits change state simultaneously, with no ripple. The only way we can build such a counter circuit from J-K flip-flops is to connect all the clock inputs together, so that each and every flip-flop receives the exact same clock pulse at the exact same time:



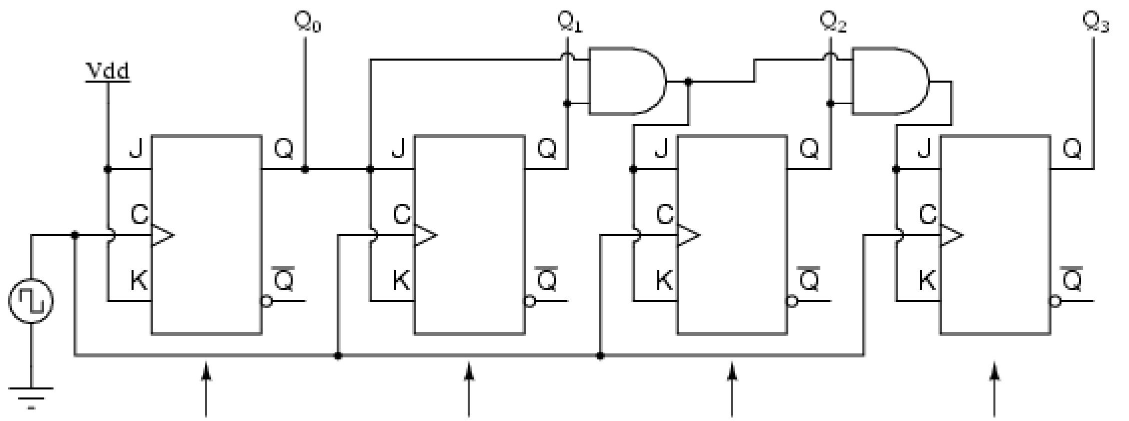
Synchronous Counters

- Now, the question is, what do we do with the J and K inputs? We know that we still have to maintain the same divide-by-two frequency pattern in order to count in a binary sequence, and that this pattern is best achieved utilizing the "toggle" mode of the flip-flop, so the fact that the J and K inputs must both be (at times) "high" is clear. However, if we simply connect all the J and K inputs to the positive rail of the power supply as we did in the asynchronous circuit, this would clearly not work because all the flip-flops would toggle at the same time: with each and every clock pulse!



Synchronous Counters

A four-bit synchronous "up" counter



This flip-flop toggles on every clock pulse

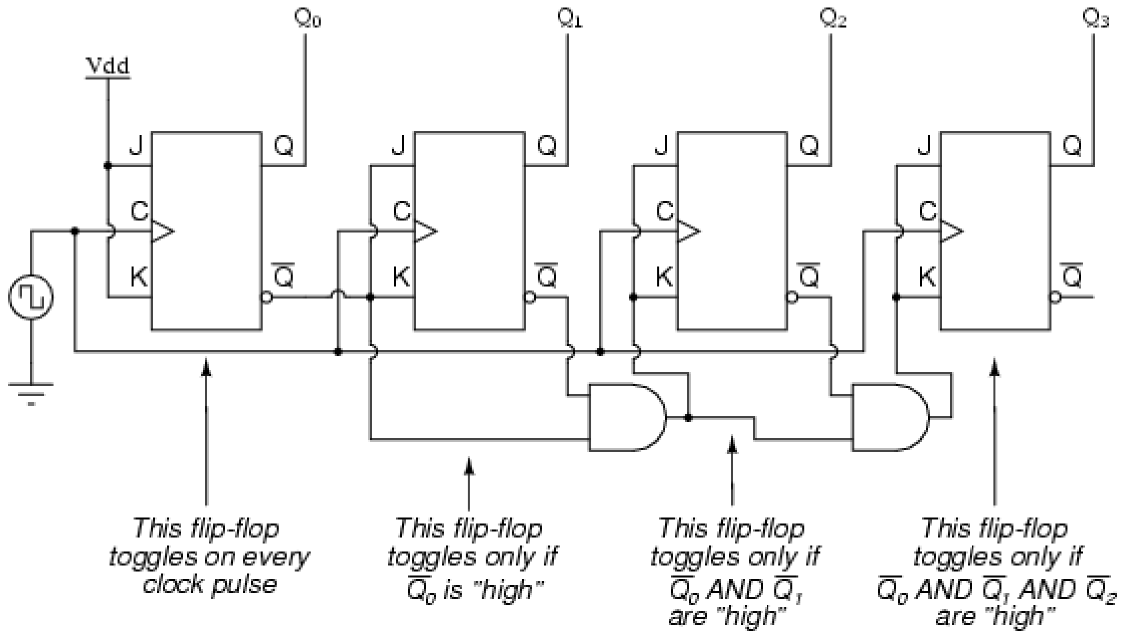
This flip-flop toggles only if Q_0 is "high"

This flip-flop toggles only if Q_0 AND Q_1 are "high"

This flip-flop toggles only if Q_0 AND Q_1 AND Q_2 are "high"

Synchronous Counters

A four-bit synchronous "down" counter


Q0
Q1
Q2
Q3
Vdd
J
C
K
Q
$\overline{Q}$
This flip-flop toggles on every clock pulse
This flip-flop toggles only if $\overline{Q}_0$ is "high"
This flip-flop toggles only if $\overline{Q}_0$ AND $\overline{Q}_1$ are "high"
This flip-flop toggles only if $\overline{Q}_0$ AND $\overline{Q}_1$ AND $\overline{Q}_2$ are "high"

How To Design Synchronous Counter

- For synchronous counters, all the flip-flops are using the same CLOCK signal. Thus, the output would change synchronously.
- Procedure to design synchronous counter are as follows:-

STEP 1: Obtain the State Diagram.

STEP 2: Obtain the Excitation Table using state transition table for any particular FF (JK or D). Determine number of FF used.

STEP 3: Obtain and simplify the function of each FF input using K-Map.

STEP 4: Draw the circuit.

How To Design Synchronous Counter

- Design a MOD-4 synchronous up-counter, using JK FF.

STEP 1: Obtain the State transition Diagram

$$\begin{tikzcd}
& 0 \arrow[dr] & & & & & 00 \arrow[dr] & \\
3 \arrow[ur] & & 1 \arrow[dl] & {} \arrow[r, "\text{Binary}"'] & {} & 11 \arrow[ur] & & 01 \arrow[dl] \\
& 2 \arrow[ul] & & & & & 10 \arrow[ul] &
\end{tikzcd}$$

How To Design Synchronous Counter

STEP 2: Obtain the Excitation table. Two JK FF are used.

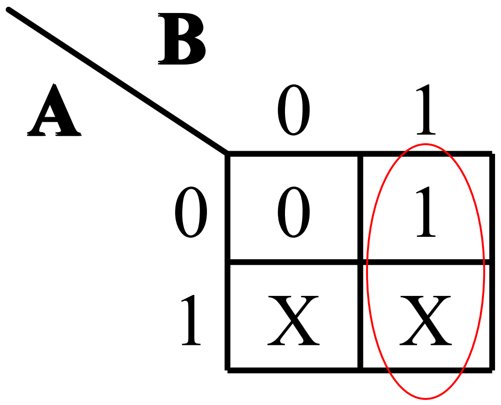
OUTPUT TRANSITION Q_N		Q_{N+1}	**FF INPUT** J	K
0	→	0	0	X
0	→	1	1	X
1	→	0	X	1
1	→	1	X	0

Excitation table

Present State	**Next State**	**Input, J K**	
B A	**B A**	J_B K_B	J_A K_A
0 0	0 1	0 X	1 X
0 1	1 0	1 X	X 1
1 0	1 1	X 0	1 X
1 1	0 0	X 1	X 1

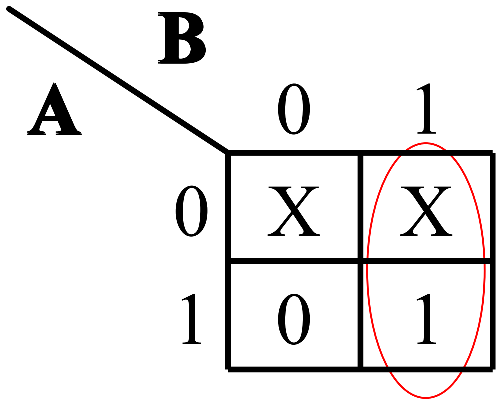
How To Design Synchronous Counter

STEP 3: Obtain the simplified function using K-Map



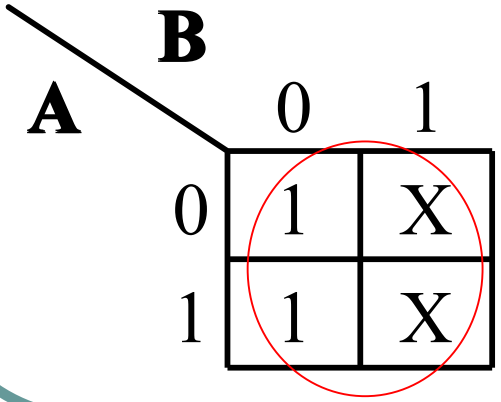
A \ B	0	1
0	0	1
1	X	X

$\boxed{J_B = A}$



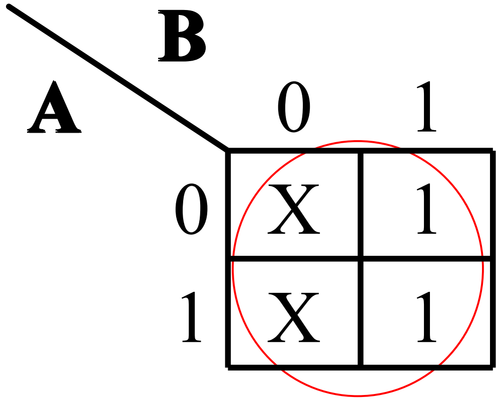
A \ B	0	1
0	X	X
1	0	1

$\boxed{K_B = A}$



A \ B	0	1
0	1	X
1	1	X

$\boxed{J_A = 1}$

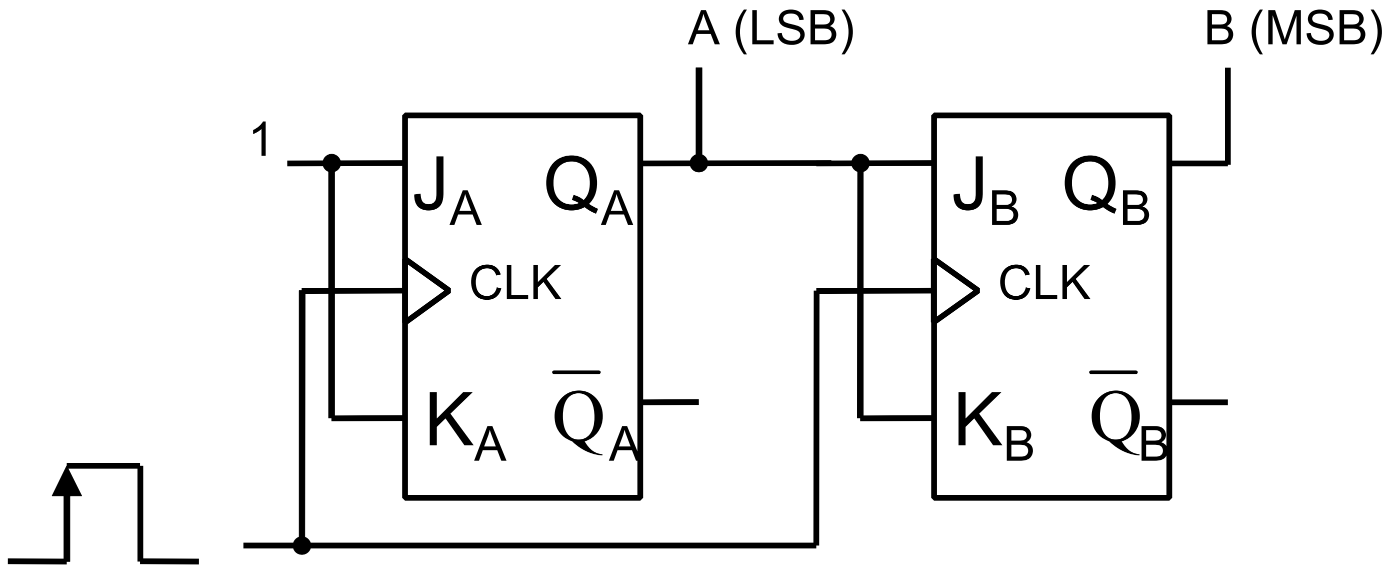


A \ B	0	1
0	X	1
1	X	1

$\boxed{K_A = 1}$

How To Design Synchronous Counter

STEP 4: Draw the circuit diagram.



(MOD-4 synchronous up-counter)

How To design Synchronous Counter

- **Let us employ these techniques to design a MOD-8 counter to count in the following sequence: 0, 1, 2, 3, 4, 5, 6, 7.**
- **Step1:** Determined Flip Flop Used and Creating **state transition diagram.** (*Rajah Keadaan*)

$$\boxed{N = 2^n}$$

$8 = 2^n$

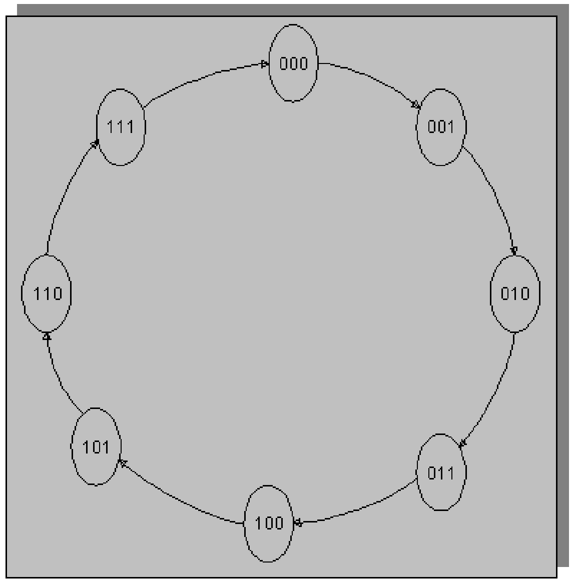
$n = \log 8 \ / \ \log 2$

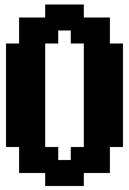
$= 3$ Flip Flop (3 Bit)

$$\boxed{M = 2^n - 1}$$

$= 2^3 - 1 = 8 - 1 = 7$

N = Modulo/MOD
n = Flip Flop Used
M = Maximum Number To Be Count





How To design Synchronous Counter

- **Step 2:** Creating present state-next state 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	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

$Q_0 = Q_A$
$Q_1 = Q_B$
$Q_2 = Q_C$

How To Design Synchronous Counter

- **Step 3:** Expand the present state-next state table to form the transition table.

Excitation Table (*Jadual Ujaan Flip Flop JK*)			
Q	**$\bar{Q}$**	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

'X' indicates a **"don't care"** condition.

Present State			**Next State**			**Present inputs**		
Q_C	Q_B	Q_A	Q_C	Q_B	Q_A	J_CK_C	J_BK_B	J_AK_A
0	0	0	0	0	1	0X	0X	1X
0	0	1	0	1	0	0X	1X	X1
0	1	0	0	1	1	0X	X0	1X
0	1	1	1	0	0	1X	X1	X1
1	0	0	1	0	1	X0	0X	1X
1	0	1	1	1	0	X0	1X	X1
1	1	0	1	1	1	X0	X0	1X
1	1	1	0	0	0	X1	X1	X1

How To Design Synchronous Counter

- **Step 4:** Use Karnaugh maps to identify the present state logic functions for each of the inputs.

E.g. for J_2 we get:

		$\overline{Q_C}\,\overline{Q_B}$			Q_CQ_B
		00	01	1 1	1 0
$\overline{Q_A}$	0	0	0	X	X
Q_A	1	0	1	X	X

$J_C = Q_BQ_A$

Using similar techniques for the other inputs we get:

$K_C = Q_BQ_A$

$J_B = Q_A$

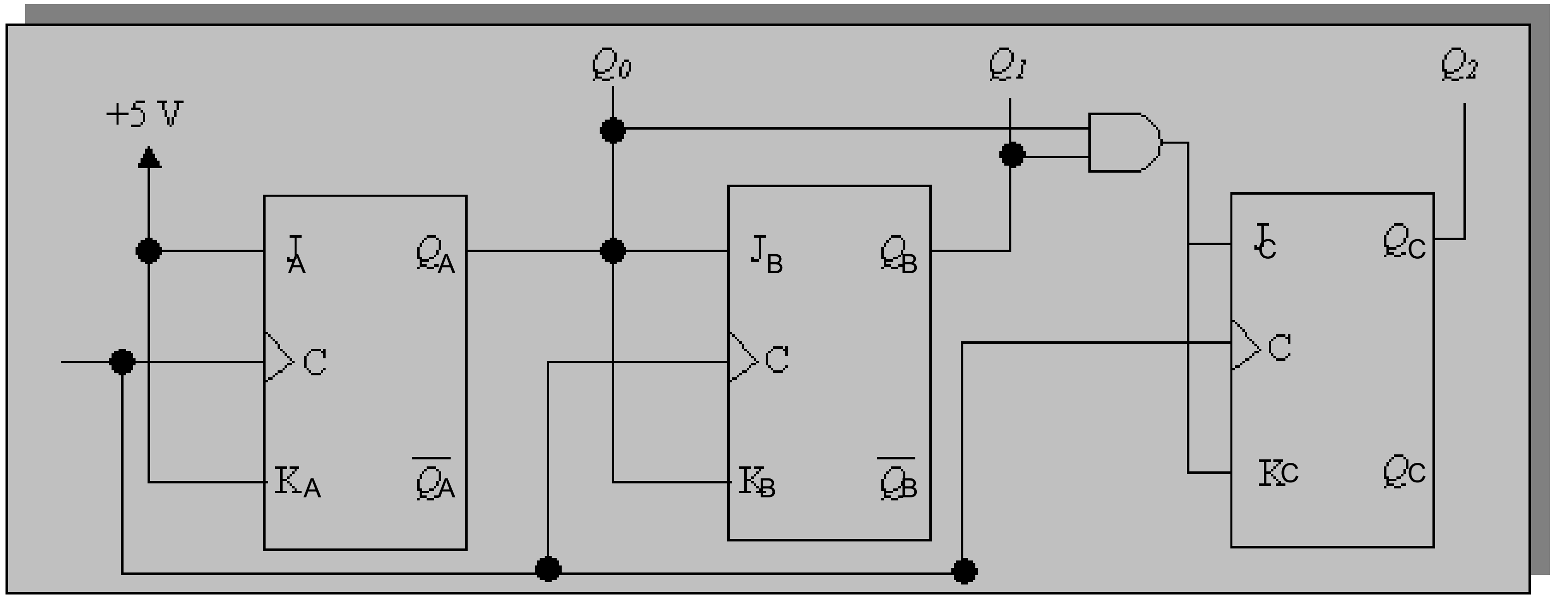
$K_B = Q_A$

$J_A = 1$

$K_A = 1$

How To Design Synchronous Counter

- **Step 5:** Constructing Circuit



How To Design Synchronous Counter that count Random number

Example :

Design a Synchronous Counter to Count 4,7,3,0 and 2 respectively using JKFlip Flop negative trigered by showing:

i. Flip Flop Used
ii. State Transition Diagram
iii. Exitation Table / Present state, next State
iv. Karnough Map & perform Simplified Function
v. The Synchronous Counter

Synchronous Counter to Count 4,7,3,0 and 2 respectively

Solution:

Step 1 : Flip Flop Used

Find Modulo, N= 2^n

M = 7, M = 2^n-1 = 7

2^n = N , so, N = 7+1 = 8, **MOD 8**

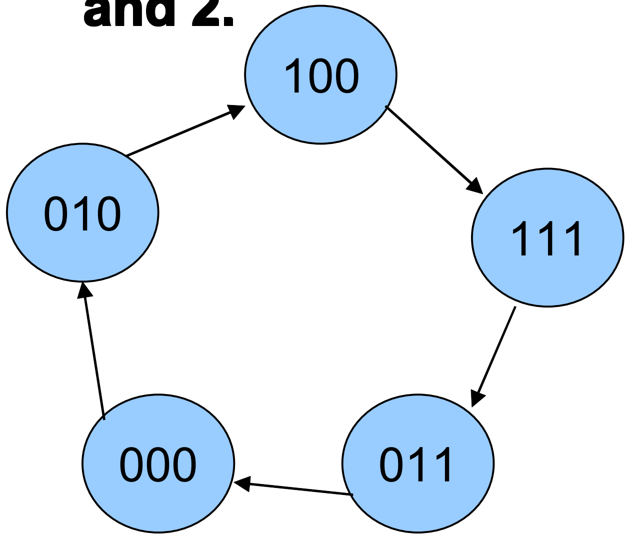
2^n = 8, n = log 8 / log 2

n = 3 bit = 3 Flip Flop.

Synchronous Counter to Count 4,7,3,0 and 2 respectively

Solution:

Step 2 : State Transation Diagram, to count 4, 7, 3, 0 and 2.



Exitation Truth Table For Counter using JK FlipFlop

Present Qn	Next State Qn+1	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

Synchronous Counter to Count 4,7,3,0 and 2 respectively

Step 3 : Exitation Table /present State, Next State

Decimal	Present State			Next State			JC	KC	JB	KB	JA	KA
	QC	QB	QA	QC	QB	QA						
4	1	0	0	1	1	1	x	0	1	x	1	x
7	1	1	1	0	1	1	x	1	x	0	x	0
3	0	1	1	0	0	0	0	x	x	1	x	1
0	0	0	0	0	1	0	0	x	1	x	0	x
2	0	1	0	1	0	0	1	x	x	1	0	x

Synchronous Counter to Count 4,7,3,0 and 2 respectively

Step 4: Karnough Map and Simplified Function

CB / A	**00**	**01**	**11**	**10**
0	**0** 0	**0** 2	**x** 6	**1** 4
1	**x** 1	**x** 3	**x** 7	**x** 5

K-Map For

JA = QC

Synchronous Counter to Count 4,7,3,0 and 2 respectively

K-Map For
JA = QC

CB / A	00	01	11	10
0	0 0	0 2	X 6	1 4
1	X 1	X 3	X 7	X 5

K-Map For
$KA = \overline{QC}$

CB / A	00	01	11	10
0	X 0	X 2	X 6	X 4
1	X 1	1 3	0 7	X 5

Synchronous Counter to Count 4,7,3,0 and 2 respectively

K-Map For

JB = 1

CB / A	00	01	11	10
0	**1** 0	**X** 2	**X** 6	**1** 4
1	**X** 1	**X** 3	**X** 7	**X** 5

K-Map For

KB = $\overline{QC}$

CB / A	00	01	11	10
0	**X** 0	**1** 2	**X** 6	**X** 4
1	**X** 1	**1** 3	**0** 7	**X** 5

Synchronous Counter to Count 4,7,3,0 and 2 respectively

K-Map For
JC = QA+QB

CB / A	00	01	11	10
0	**0** 0	**1** 2	**X** 6	**X** 4
1	**X** 1	**0** 3	**X** 7	**X** 5

K-Map For
KC = QB

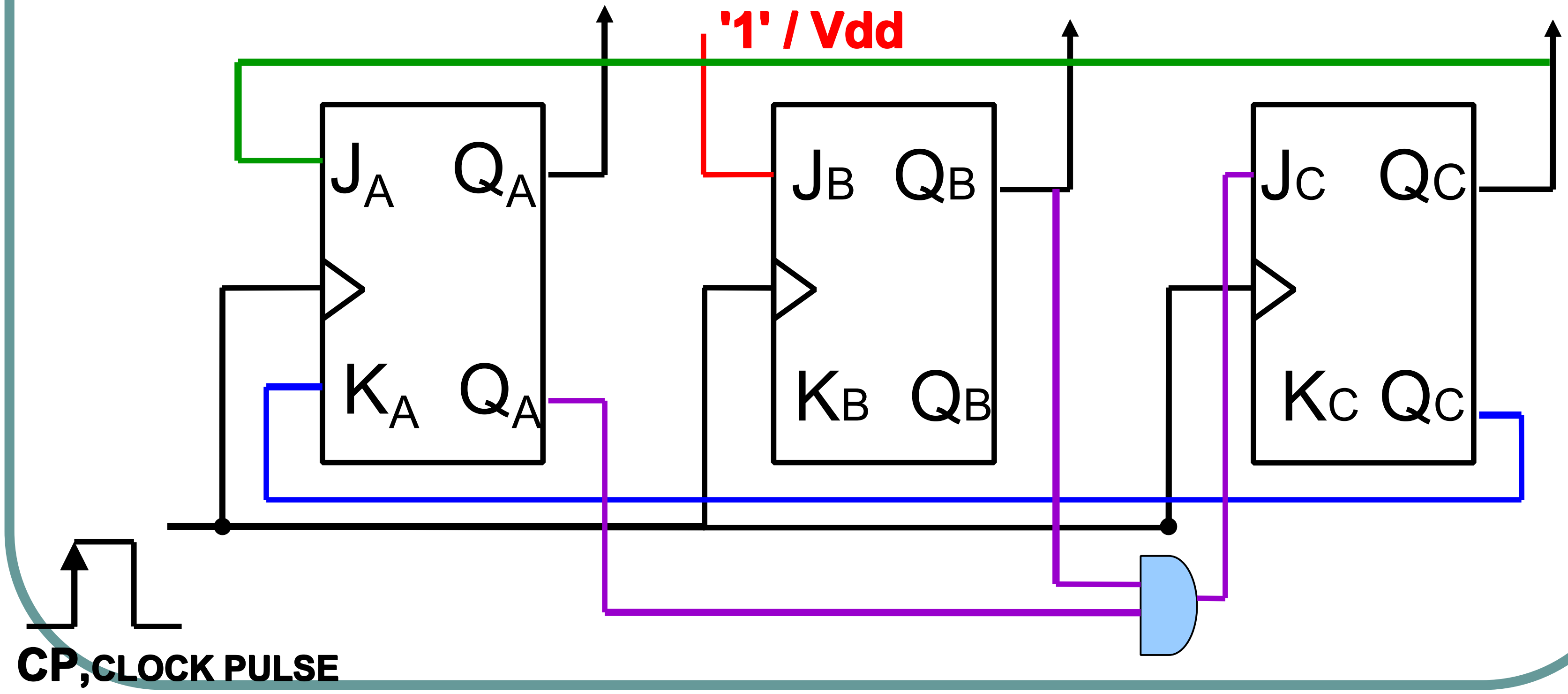
CB / A	00	01	11	10
0	**X** 0	**X** 2	**X** 6	**0** 4
1	**X** 1	**X** 3	**1** 7	**X** 5

Synchronous Counter to Count 4,7,3,0 and 2 respectively

Step 5 : Perform Counter Circuit

By using simplified function from K-Map, **$J_A = Q_C$, $K_A = \overline{Q_C}$,**

$J_B = 1$, $K_B = \overline{Q_C}$, $J_C = Q_A + Q_B$, $K_C = Q_B$



Synchronous Counter

Exercises:

- **Design a counter to count in the following sequence: 6, 4. 2, 3, 1.**
- **Design a counter to count in the following sequence: 15,9,11,5,2,13,1.**
- Do more exercises in Past Years Exam Paper.

End Of This Topic…..