Counters in Digital Logic

According to Wikipedia, in digital logic and computing, a **Counter** is a device which stores (and sometimes displays) the number of times a particular event or process has occurred, often in relationship to a clock signal. Counters are used in digital electronics for counting purpose, they can count specific event happening in the circuit. For example, in UP counter a counter increases count for every rising edge of clock. Not only counting, a counter can follow the certain sequence based on our design like any random sequence 0,1,3,2… .They can also be designed with the help of flip flops.

Counter Applications:

1,Digital clock
2, Automobile Parking control
3, Delay generation.

Counter Classification

Counters are broadly divided into two categories

1. Asynchronous counter
2. Synchronous counter

1. **Asynchronous Counter(Ripple counters)**

In asynchronous counter we don't use universal clock, only first flip flop is driven by main clock and the clock input of rest of the following flip flop is driven by output of previous flip flops. We can understand it by following diagram-

2. **Synchronous counter(Parallel counters)**

In synchronous counter clock pulse directly applied to all flip-flops .So it also known as Parallel counters.

Modulus Counter (MOD-N Counter)

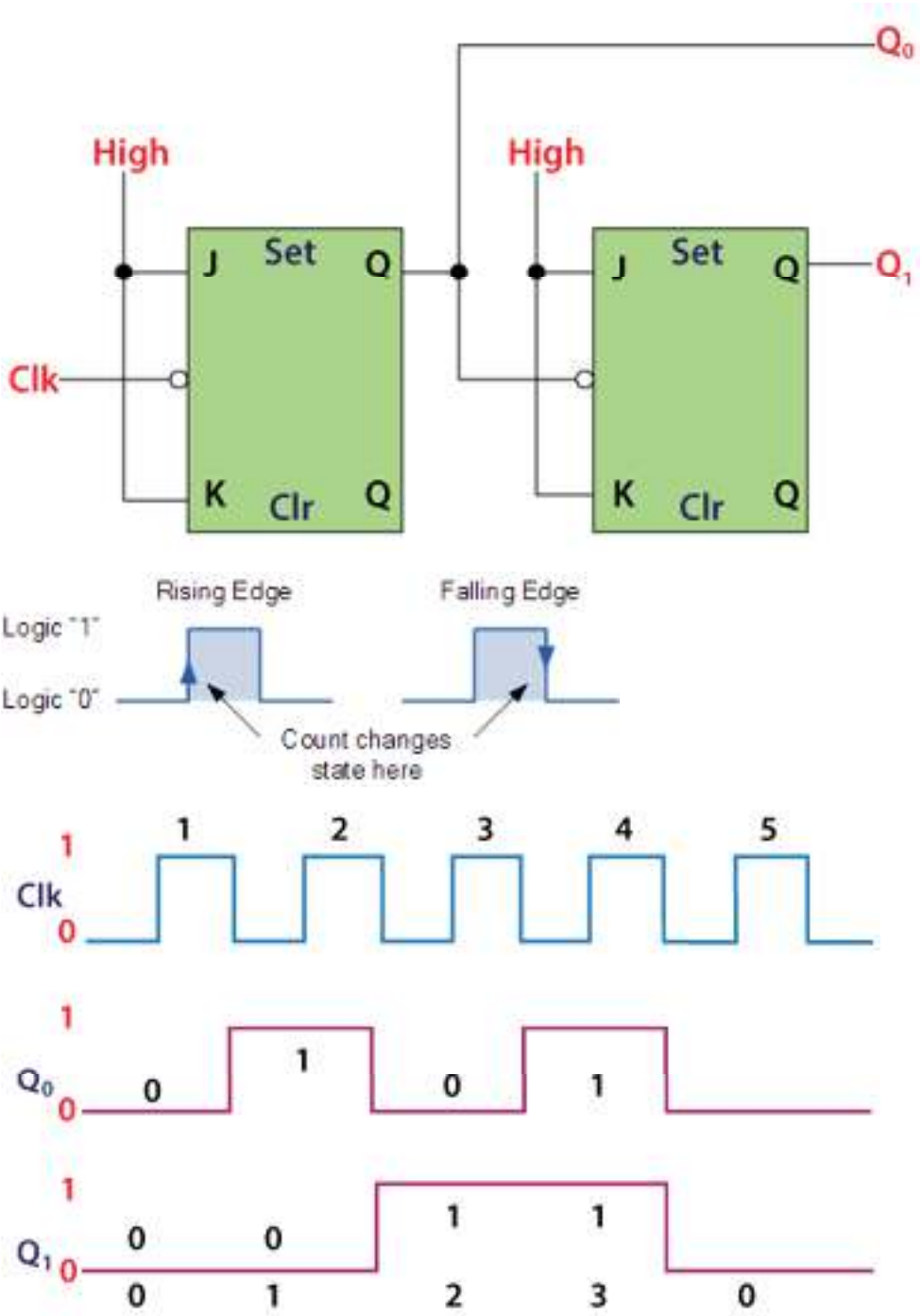
The 2-bit ripple counter is called as MOD-4 counter and 3-bit ripple counter is called as MOD-8 counter. So in general, an n-bit ripple counter is called as modulo-N counter. Where, MOD number = 2^n.

Type of modulus

- 2-bit up or down (MOD-4)

- 3-bit up or down (MOD-8)
- 4-bit up or down (MOD-16)

2 bit counter or modulus 4 counter



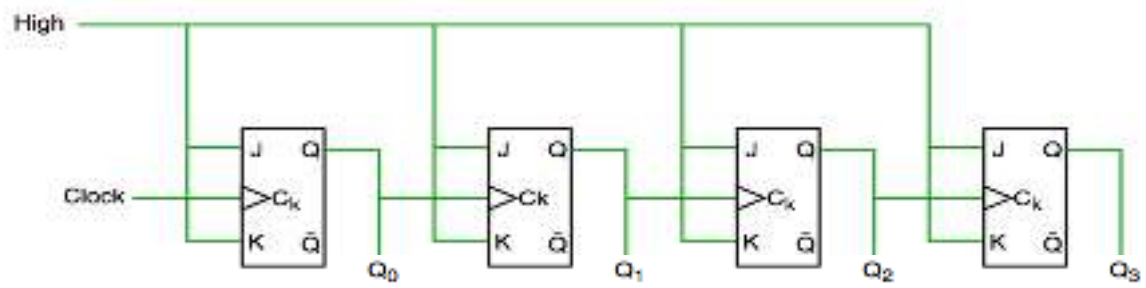
Operation

1. **Condition 1:** When both the flip flops are in reset condition.
   Operation: The outputs of both flip flops, i.e., Q_A Q_B, will be 0.
2. **Condition 2:** When the first negative clock edge passes.
   Operation: The first flip flop will toggle, and the output of this flip flop will change from 0 to 1. The output of this flip flop will be taken by the clock input of the next flip flop. This output will be taken as a positive edge clock by the second flip flop. This input will not change the second flip flop's output state because it is the negative edge triggered flip flop.
   So, $Q_A = 1$ and $Q_B = 0$
3. **Condition 3:** When the second negative clock edge is applied.
   Operation: The first flip flop will toggle again, and the output of this flip flop will change from 1 to 0. This output will be taken as a negative edge clock by the second flip flop. This input will change the second flip flop's output state because it is the negative edge triggered flip flop.
   So, $Q_A = 0$ and $Q_B = 1$.
4. **Condition 4:** When the third negative clock edge is applied.
   Operation: The first flip flop will toggle again, and the output of this flip flop will change from 0 to 1. This output will be taken as a positive edge clock by the second flip flop. This input will not change the second flip flop's output state because it is the negative edge triggered flip flop.
   So, $Q_A = 1$ and $Q_B = 1$
5. **Condition 5:** When the fourth negative clock edge is applied.
   Operation: The first flip flop will toggle again, and the output of this flip flop will change from 1 to 0. This output will be taken as a negative edge clock by the second flip flop. This input will change the output state of the second flip flop.
   So, $Q_A = 0$ and $Q_B = 0$

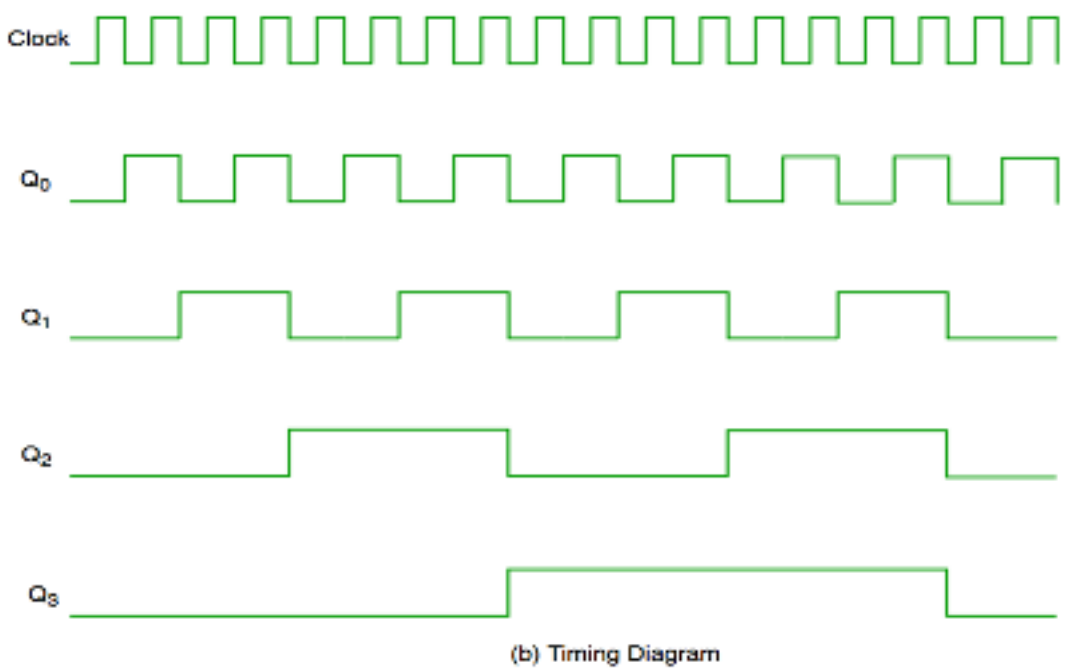
Truth Table

Clock	Counter output		State number	Deciimal Counter output
	Q_B	Q_A		
Initially	0	0	—	0
1st	0	1	1	1
2nd	1	0	2	2
3rd	1	1	3	3
4th	0	0	4	0

4 bit counter or Modulus 16 counter



(a) Asynchronous counter



(b) Timing Diagram

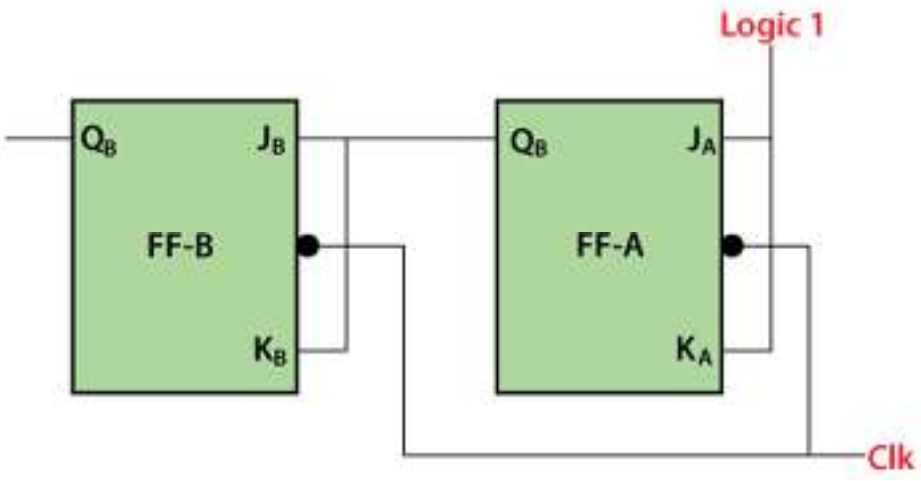
It is evident from timing diagram that Q0 is changing as soon as the rising edge of clock pulse is encountered, Q1 is changing when rising edge of Q0 is encountered(because Q0 is like clock pulse for second flip flop) and so on. In this way ripples are generated through Q0,Q1,Q2,Q3 hence it is also called **RIPPLE counter.**

Synchronous counters

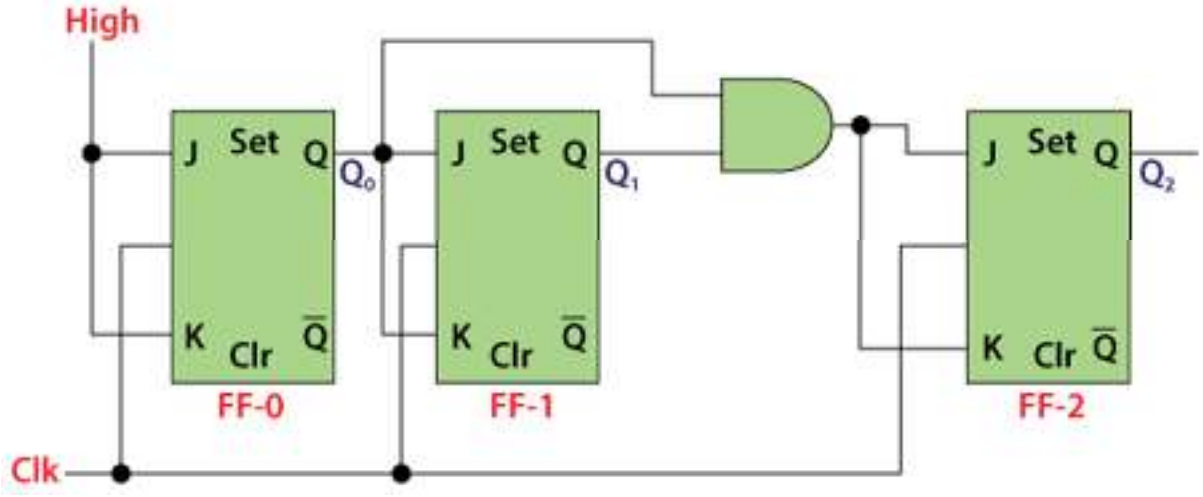
In the **Asynchronous counter**, the present counter's output passes to the input of the next counter. So, the counters are connected like a chain. The drawback of this system is that it creates the counting delay, and the propagation delay also occurs during the counting stage. The **synchronous counter** is designed to remove this drawback.

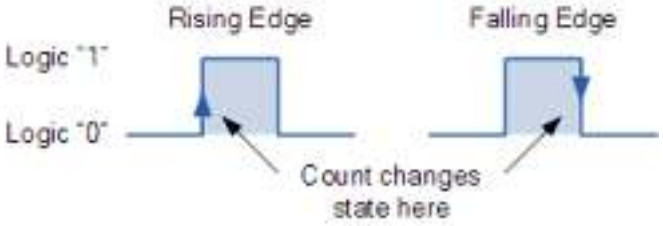
In the **synchronous counter**, the same clock pulse is passed to the clock input of all the flip flops. The clock signals produced by all the flip flops are the same as each other. Below is the diagram of a 2-bit synchronous counter in which the inputs of the first flip flop, i.e., FF-A, are set to 1. So, the first flip flop will work as a toggle flip-flop. The output of the first flip flop is passed to both the inputs of the next JK flip flop.

Logical Diagram

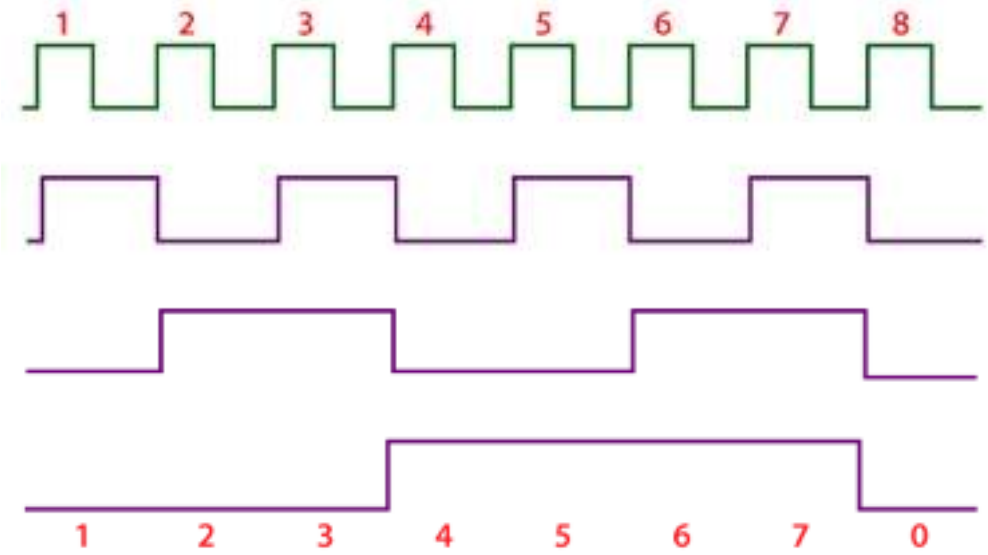


3 bit counter modulus 8 counter





Signal Diagram (here u draw the output Q0 at negative edge of clock pulse)



Operation

1. **Condition 1:** When both the flip flops are in reset condition.
   Operation: The outputs of both flip flops, i.e., Q_A Q_B, will be 0.
   So, **$Q_A = 0$** and **$Q_B = 0$**
2. **Condition 2:** When the first positive clock edge passes.
   Operation: The first flip flop will be toggled, and the output of this flip flop will be changed from 0 to 1. When the first positive clock edge is passed, the output of the first flip flop will be 0. The clock input of the first flip flop and both of its inputs will set to 0. In this way, the state of the second flip flop will remain the same.
   So, **$Q_A = 1$** and **$Q_B = 0$**
3. **Condition 2:** When the second negative clock edge is passed.
   Operation: The first flip flop will be toggled again, and the output of this flip flop will be changed from 1 to 0. When the second negative clock edge is passed, the output of the first flip flop will be 1. The clock input of the first flip flop and both of its inputs will set to 1. In this way, the state of the second flip flop will change from 0 to 1.
   So, **$Q_A = 0$** and **$Q_B = 1$**
4. **Condition 2:** When the third negative clock edge passes.
   Operation: The first flip flop will toggle from 0 to 1, but at this instance, both the inputs and the clock input set to 0. Hence, the outputs will remain the same as before.
   So, **$Q_A = 1$** and **$Q_B = 1$**
5. **Condition 2:** When the fourth negative clock edge passes.
   Operation: The first flip flop will toggle from 1 to 0. At this instance, the inputs and the clock input of the second flip flop set to 1. Hence, the outputs will change from 1 to 0.
   So, **$Q_A = 0$** and **$Q_B = 0$**

State	Q_2	Q_1	Q_0
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

Applications of counters

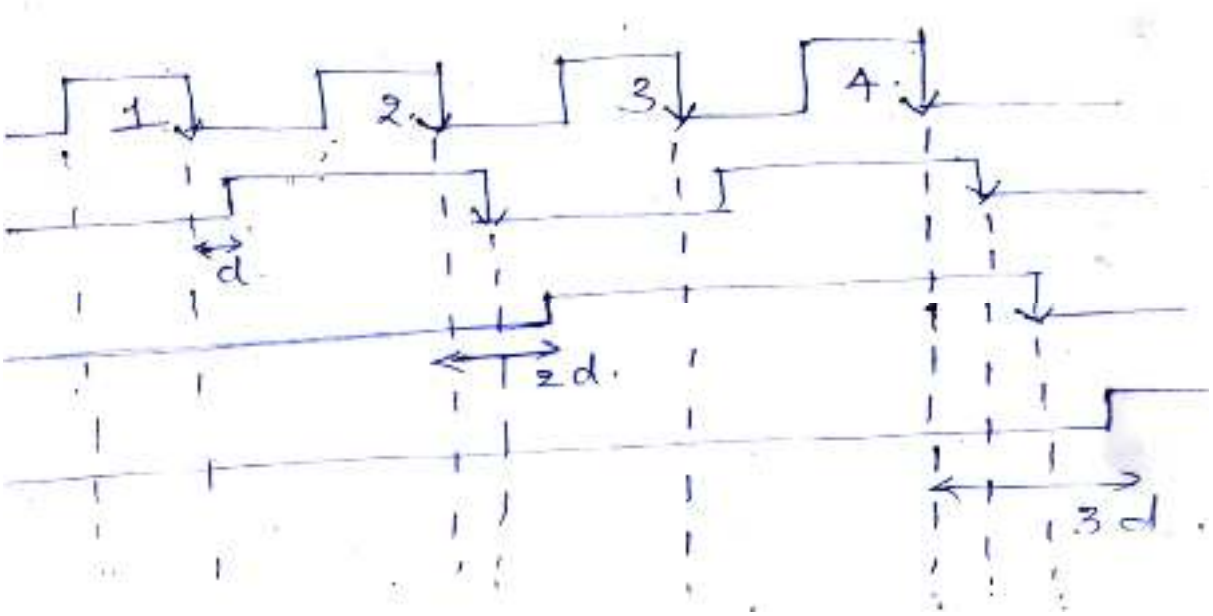
Counter found their applications in many digital electronic devices. Some of their applications are listed below.

- Frequency counters
- Digital clocks
- Analog to digital convertors.
- With some changes in their design, counters can be used as frequency divider circuits. The frequency divider circuit is that which divides the input frequency exactly by '2'.
- In time measurement. That means calculating time in timers such as electronic devices like ovens and washing machines.
- We can design digital triangular wave generator by using counters.

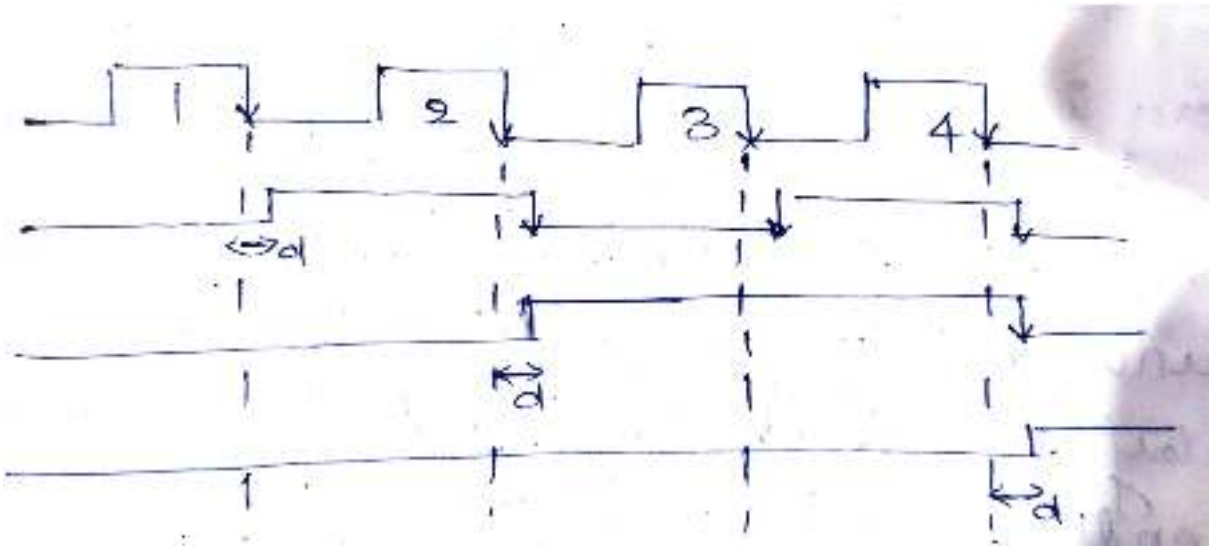
There are many other type of counters rather than synchronous and asynchronous counters, such as Decade counter, Binary counter, Ring

counter, Johnson counter, Up / Down counter etc. , which we will discuss about them in our upcoming sessions.

Actual timing diagram of asynchronous counters.



Actual Timing diagram of Synchronous Counters.

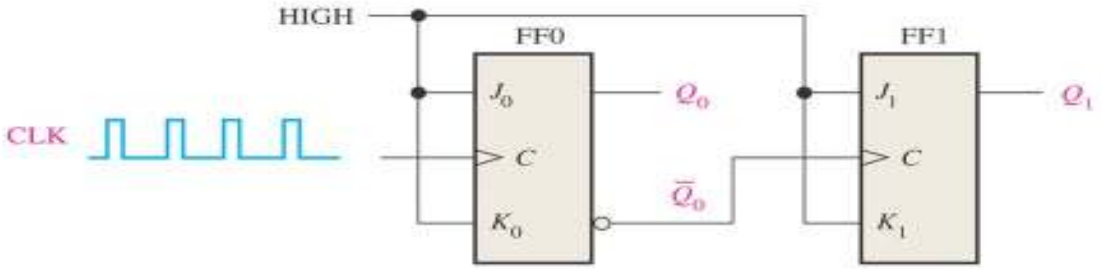


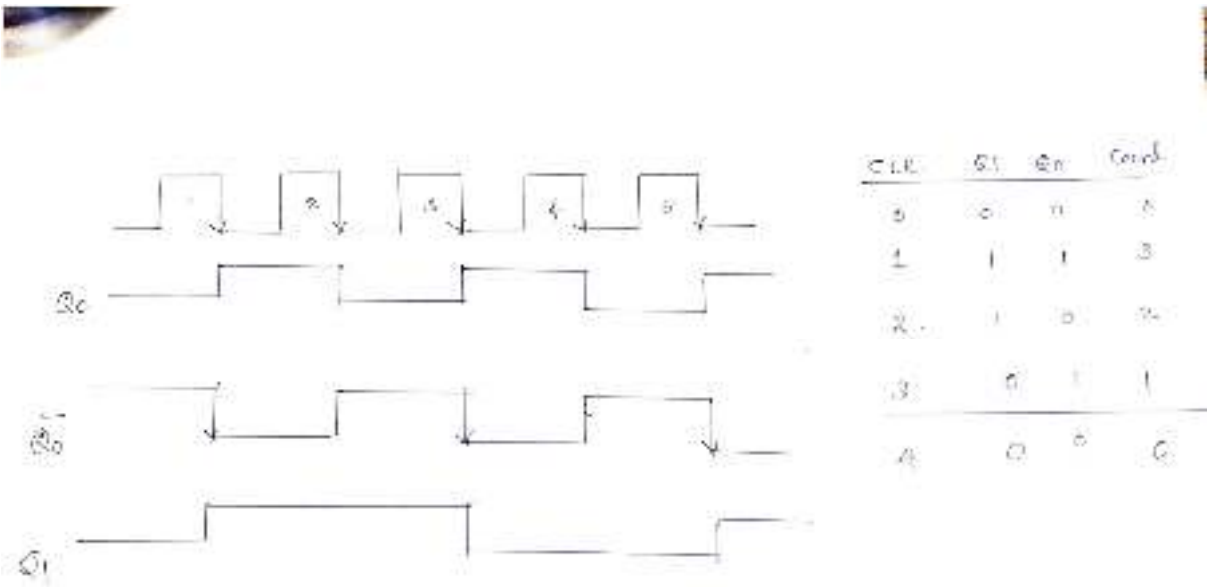
SYNCHRONOUS COUNTERS	ASYNCHRONOUS COUNTERS
The propagation delay is very low.	Propagation delay is higher than that of synchronous counters.
Its operational frequency is very high.	The maximum frequency of operation is very low.
These are faster than that of ripple counters.	These are slow in operation.
Large number of logic gates are required to design	Less number of logic gates required.
High cost.	Low cost.
Synchronous circuits are easy to design.	Complex to design.
Standard logic packages available for synchronous.	For asynchronous counters, Standard logic packages are not available.

SYNCHRONOUS COUNTERS ASYNCHRONOUS COUNTERS

Down Counters

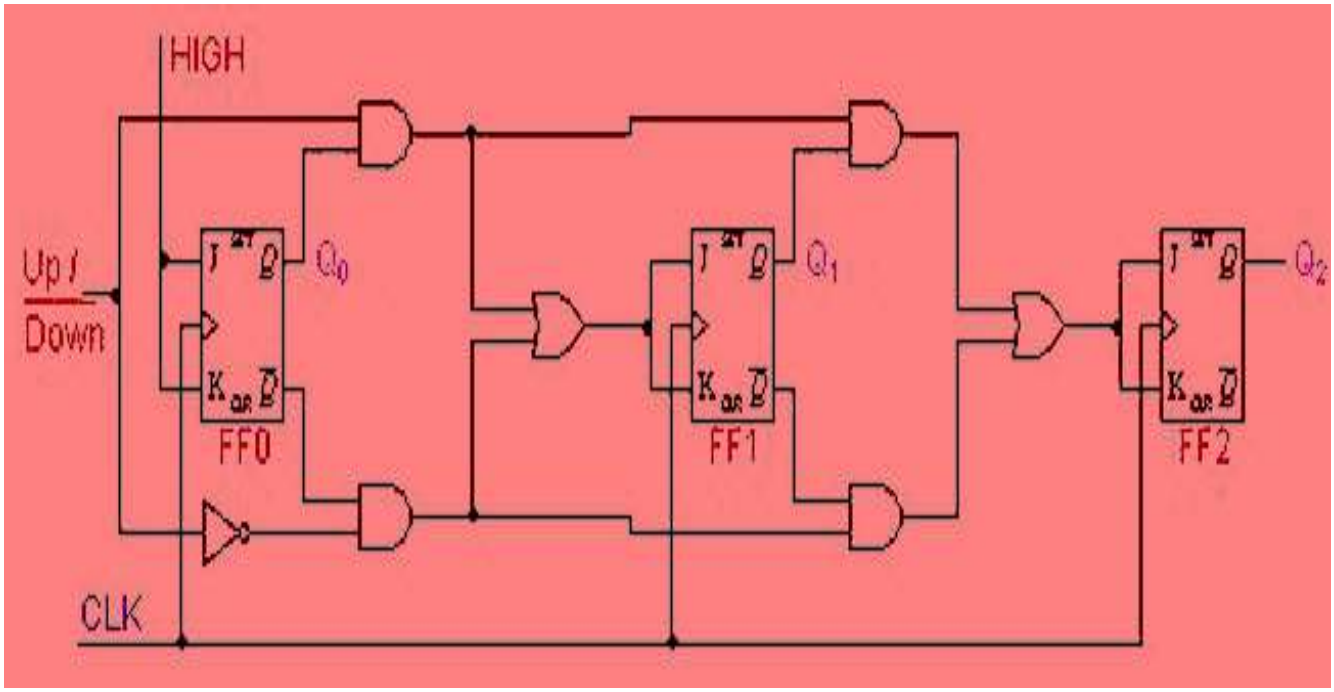
Figure 8–1 A 2-bit asynchronous binary counter. Open file F08-01 to verify operation.





Synchronous Up-Down Counters

A three bit synchronous Up-Down counter, tabular form and series are given below. This type of counter has an up-down control i/p similar to asynchronous up-down counter, that is used to control the counter's direction through a certain series.



Synchronous Up-Down Counters Circuit Diagram

The series of the table shows

- Q0 ties on each CLK pulse for both up & down series
- When Q0=1 for the up series, then the state of the Q1 changes on the next CLK pulse.
- When Q0=0 for the down series, then the state of the Q1 changes on the next CLK pulse.
- When Q0=Q1=1 for the up series, then the state of the Q2 changes on the next CLK pulse.
- When Q0=Q1=0 for the down series, then the state of the Q2 changes on the next CLK pulse.

COUNT-UP Mode				COUNT-DOWN Mode			
States	Q_C	Q_B	Q_A	States	Q_C	Q_B	Q_A
0	0	0	0	7	1	1	1
1	0	0	1	6	1	1	0
2	0	1	0	5	1	0	1
3	0	1	1	4	1	0	0
4	1	0	0	3	0	1	1
5	1	0	1	2	0	1	0
6	1	1	0	1	0	0	1
7	1	1	1	0	0	0	0

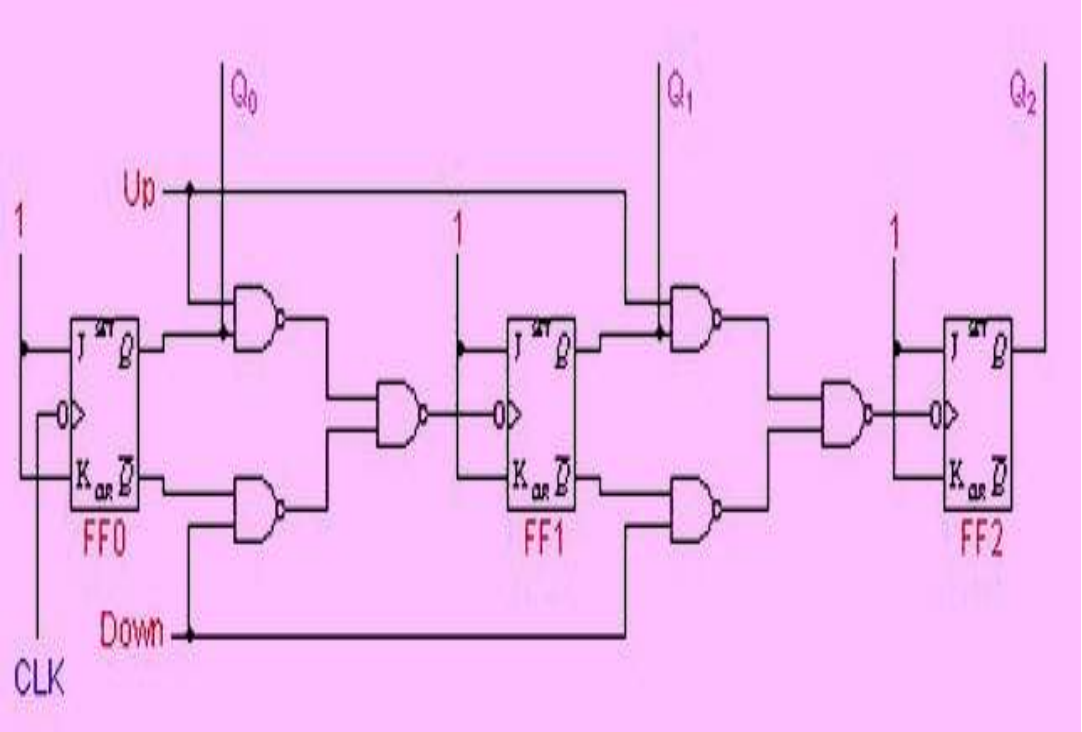
Sequence of the Synchronous Decade Counters

The above characteristics are employed with the AND gate, OR gate and NOT gate. The logic diagram of this is shown in the above diagram.

Asynchronous Up-Down Counters

In particular applications, a counter must be capable to count both up & down. The below circuit is a three bit up & down counter, that counts UP or DOWN based on the control signal status. When the UP i/p is at 1 & the DOWN i/p is at 0, the NAND gate between FF0 & FF1 will gate the non-inverted o/p (Q) of flip flop (FF0) into the clock i/p of flip flop (FF1). Likewise, the non-inverted o/p of Flip Flop1 will be gated through the other NAND gate into the clock i/p of flip-flop2. Therefore the counter will count up.

((here in diagram instead of using NAND gate we can use AND and OR gate as like used in previous diagram)



Asynchronous Up-Down Counter Circuit Diagram

Once the control i/p (UP) is at 0 & DOWN is at 1, the inverted o/ps of flip-flop0 (FF0) and flip-flop1 (FF) are gated into the clock i/ps of FF1 & FF2 separately. If the FFs are initially changed to 0's, then the counter will go through the below series as i/p pulses are applied. Notice that an asynchronous up-down counter is slower than an UP counter/down counter because of an extra propagation delay introduced by the NAND gates.

COUNT-UP Mode				*COUNT-DOWN Mode*			
States	Q_C	Q_B	Q_A	*States*	Q_C	Q_B	Q_A
0	0	0	0	7	1	1	1
1	0	0	1	6	1	1	0
2	0	1	0	5	1	0	1
3	0	1	1	4	1	0	0
4	1	0	0	3	0	1	1
5	1	0	1	2	0	1	0
6	1	1	0	1	0	0	1
7	1	1	1	0	0	0	0