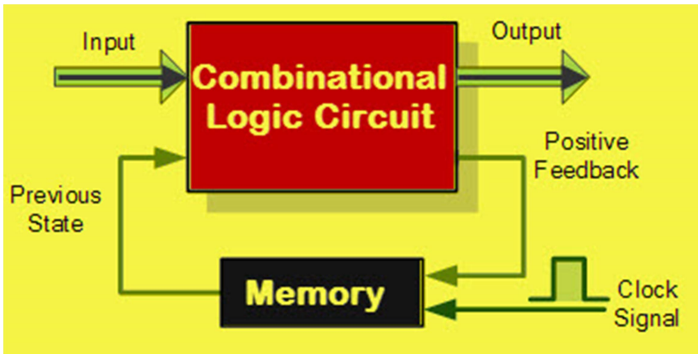
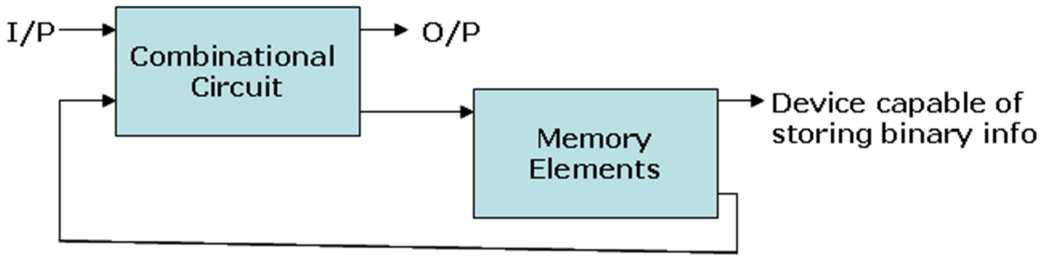
Sequential Circuits

Sequential circuit can be defined as a circuit whose output depends not only on the present inputs but also on the past history of the inputs. (Stored information).





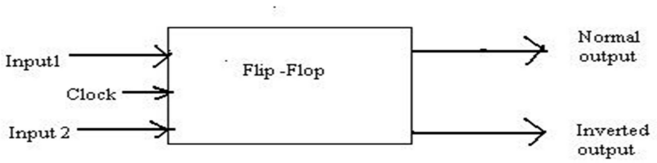
Combinational circuit	Sequential circuit
1. The circuit whose output at any instant depends only on the input present at that instant only is known as combinational circuit.	The circuit whose output at any instant depends not only on the input present but also on the past output a is known as sequential circuit
2. The combinational circuit has no memory unit.	The sequential circuit has memory unit for store past output
Examples of combinational circuits are half adder, full adder, magnitude comparator, multiplexer, demultiplexer e.t.c. half adder : used for add 2 bits. full adder : Used for add 3 bits. magnitude comparator: used for compare 2 binary data (< or > or =). multiplexer : It has n select line,2^n inputs and 1 output. demultiplexer : It has n select line,2^n outputs and 1 input.	Examples of sequential circuits are Flip flop, register, counter e.t.c. Flip flop: It used for store one bit information. Register: It is of 2 types. Buffer register: used for store information. Shift register: used for store and shift the information. Counter : Used for count the number of clock pulses .

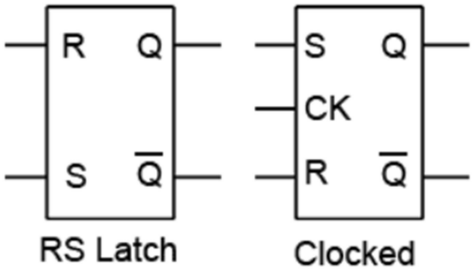
1. **What is a Flip-flop? Write the logical symbol of Flip-flop.**

A **flip-flop** is a bistable electronic circuit that has two stable states and can be used to store binary data (0 or 1).

- ✓ A flip-flop is a binary **storage device**.
- ✓ It has two stable states **HIGH** and **LOW** i.e **'1'** and **'0'**.
- ✓ It is also called bistable multivibrator.
- ✓ A flip-flop circuit has two outputs, one for the normal value and other for the complement value of the bit stored in it.

Logic Symbol:



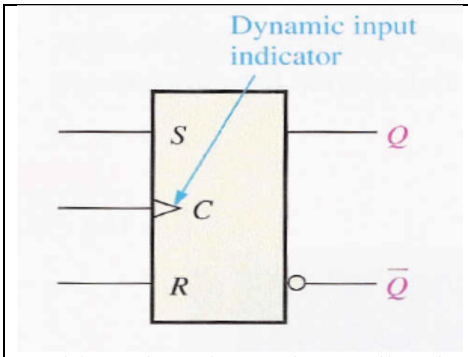


2. **List the different types of Flip-flops.**

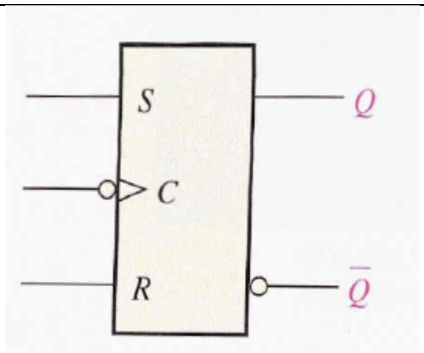
1. RS flip-flop
2. JK flip-flop
3. Master-slave flip-flop
4. D- flip-flop
5. T- flip-flop

3. Explain RS Flip-flop with truth table, logic symbol and logical circuit.

- ✓ The S and R in SR flip – flop means 'SET' and 'RESET' respectively.
- ✓ SR flip – flop has two stable states in which it can store data in the form of either 0 (LOW) or 1 (HIGH).

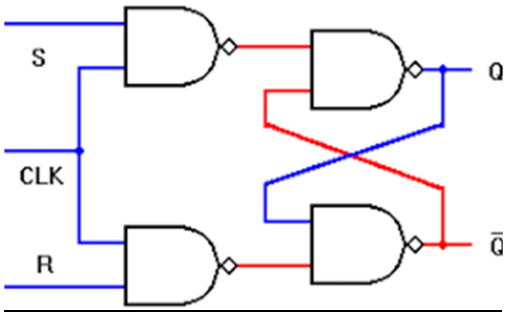


Positive edge-triggered S-R Flip-Flop



Negative edge-triggered S-R Flip-Flop

Logic circuit:



Operation of a positive edge-triggered S-R flip-flop.

INPUTS S	INPUTS R	INPUTS CLK	OUTPUTS Q	OUTPUTS $\overline{Q}$	COMMENTS
0	0	X	Q_0	$\overline{Q}_0$	No change
0	1	↑	0	1	RESET
1	0	↑	1	0	SET
1	1	↑	?	?	Invalid

↑ = clock transition LOW to HIGH

X = irrelevant ("don't care")

Q_0 = output level prior to clock transition

TABLE 7–2

Truth table for a positive edge-triggered S-R flip-flop.

Working;

1. When **S=0** and **R=0**, the output does not change from its prior state **or no change state**.
2. When **S=0** and **R=1**, the output **Q = 0**, and the flip-flop is in **RESET** state.
3. When **S=1** and **R=0**, the output **Q = 1**, and the flip-flop is in **SET** state.
4. When **S=1** and **R=1,** an **invalid** condition exists.

4. Explain JK Flip-flop with truth table, logic symbol and logical circuit.

- ✓ The J-K flip-flop is versatile and is a widely used type of flip-Hop.
- ✓ J-K Flip-flop has two inputs, labeled J and K (along with the CLK).
- ✓ The functioning of the J-K flip-flop is identical to that of the S-R flip-flop in the SET, RESET, and no-change conditions of operation. The difference is that the J-K flip-flop has no invalid state as does the S-R flip-flop.
- ✓ When both J and K = 1, the output changes states (toggles) on the rising clock edge.
- ✓ A J-K flip-flop connected for toggle operation is sometimes called a **T flip-flop**.

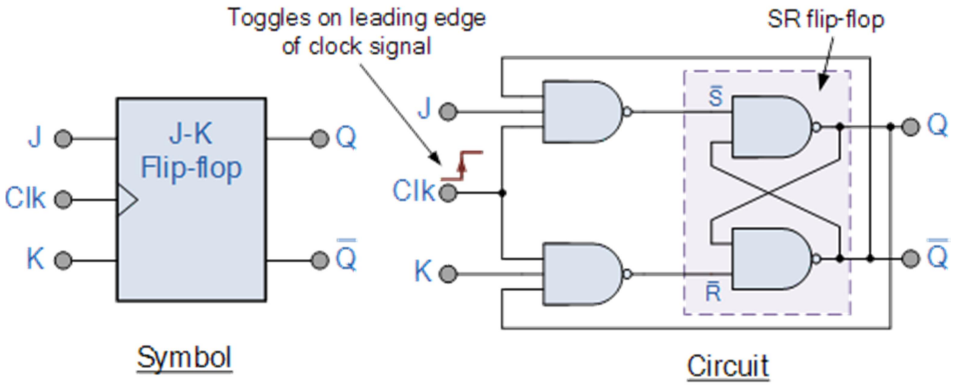


TABLE 7–4

Truth table for a positive edge-triggered J-K flip-flop.

INPUTS J	INPUTS K	INPUTS CLK	OUTPUTS Q	OUTPUTS $\overline{Q}$	COMMENTS
0	0	↑	Q_0	$\overline{Q}_0$	No change
0	1	↑	0	1	RESET
1	0	↑	1	0	SET
1	1	↑	$\overline{Q}_0$	Q_0	Toggle

↑ = clock transition LOW to HIGH

Q_0 = output level prior to clock transition

Working;

1. When **J=0** and **K=0**, the output does not change from its prior state **or no change state**.
2. When **J=0** and **K=1**, the output **Q = 0**, and the flip-flop is in **RESET** state.
3. When **J=1** and **K=0**, the output **Q = 1**, and the flip-flop is in **SET** state..
4. When **J=1** and **K=1** the output changes to the complement state called –**Toggle** or alternately blinking **on-off (1-0), off-on(0-1), on-off (1-0)** likewise.

5. What is race around condition? In which Flip-flop it is overcome.

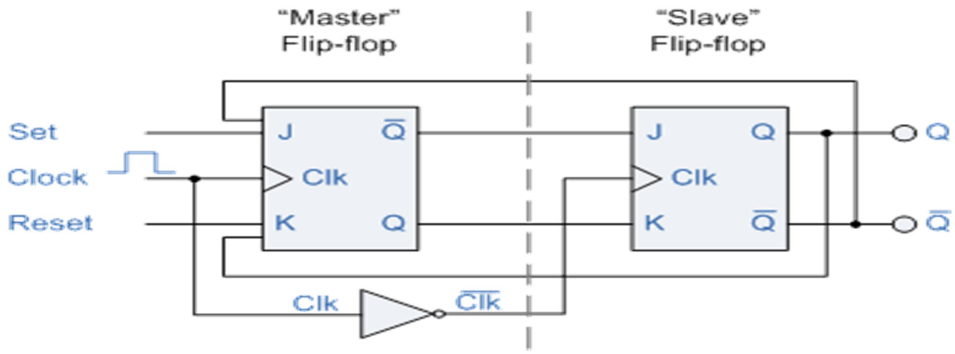
When we put **J=1** and **K=1** in **J-K** flip- flop, the output, **Q** toggles to **0 and 1** continuously; and it becomes uncertain to predict the output. This condition is known as **Race around condition**.

To overcome **race around condition** problem we should use **J -K Master Slave flip-flop.**

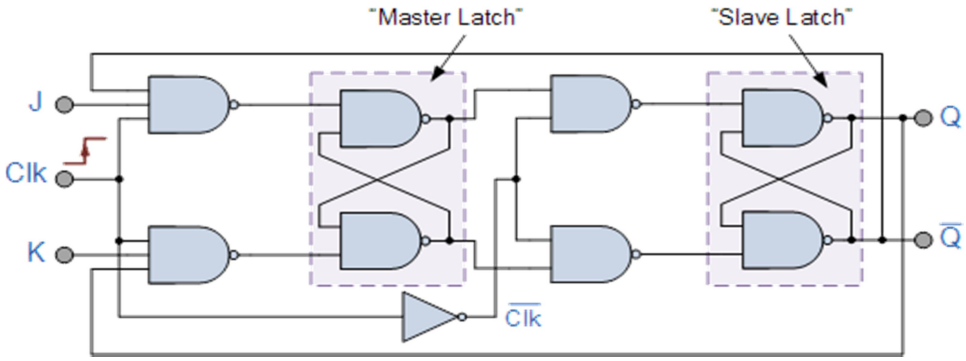
6. Explain J-K Master Slave Flip-flop with truth table, log symbol and logical circuit.

- ✓ A J-K Master Slave flip- flop consists of two clocked J-K flip-flops with a feedback from the output of the second flip-flop (Slave) to the input of the first (Master) flip-flop.
- ✓ When the clock is **HIGH**, the Master is active. The output of the Master is Set or Reset according to the state of the input. As the slave is inactive during this period, it's output remains in the previous state.
- ✓ When the clock becomes **LOW**, the output of the Slave flip flop changes because it becomes active during LOW clock period.

Logical symbol:



Logical circuit:



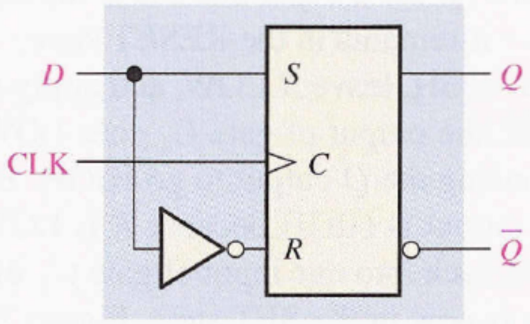
Truth table:

Inputs			Outputs		
J	K	CLK	Q	$\overline{Q}$	Comments
0	0	↑	Q_0	$\overline{Q}_0$	No change
0	1	↑	0	1	RESET
1	0	↑	1	0	SET
1	1	↑	$\overline{Q}_0$	Q_0	Toggle

7. **Explain D-Flip-flop with truth table, logic symbol and logical circuit**
   - ✓ The D-flip-flop is useful when a single data bit (1 or 0) is to be stored.
   - ✓ An Inverter is connected so that the R input is always the inverse of S.
   - ✓ The added inverter reduces the number of inputs from two to one.

FIGURE 7–20

A positive edge-triggered D flip-flop formed with an S-R flip-flop and an inverter.



Logical circuit:

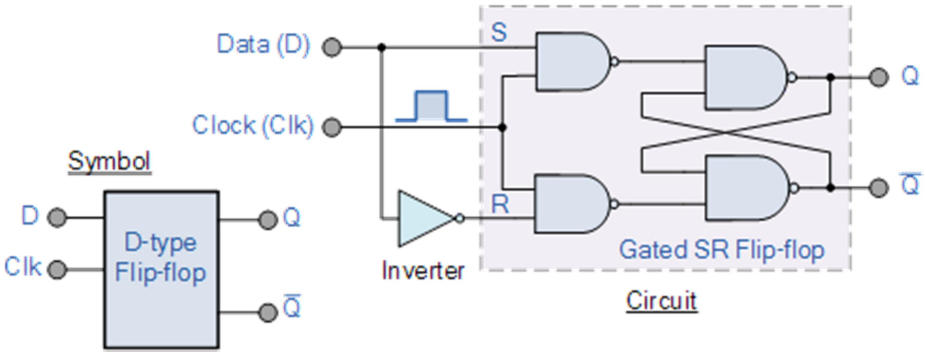


TABLE 7–3

Truth table for a positive edge-triggered D flip-flop.

INPUTS		OUTPUTS		COMMENTS
D	CLK	Q	$\overline{Q}$	
1	↑	1	0	SET (stores a 1)
0	↑	0	1	RESET (stores a 0)

↑ = clock transition LOW to HIGH

Working:

1. When **D=0**, the output **Q = 0**, and the flip-flop is in **RESET** state.
2. When **D=1**, the output **Q = 1**, and the flip-flop is in **SET** state.

8. List the applications of Flip-flop.

Flip Flops are used in the following applications:

1. Parallel Data Storage.
2. Counting.
3. Frequency Division
4. Register and Shift Register.
5. Latch.

9. What is Shift Registers? List its applications.

- ✓ A **register** is a digital circuit with two basic functions: data storage and data movement. It can consist of one or more flip-flops which are used to store and shift data.
- ✓ The **Shift Registers consists of flip-flops** that can be used for the storage and movement (transfer) of binary data in a digital system. **OR**
- ✓ Shift Registers are **sequential logic circuits**, capable of storage and transfer of data. They are made up of Flip Flops which are connected in such a way that the output of one flip flop could serve as the input of the other flip-flop, depending on the type of shift registers being created.

Applications of shift register:

- ✓ Storage and movement (transfer) of binary data in a digital system.
- ✓ Shift register is used as **Parallel to serial converter**, which converts the parallel data into serial data. It is utilized at the transmitter section after Analog to Digital Converter (ADC) block.
- ✓ Shift register is used as **Serial to parallel converter**, which converts the serial data into parallel data. It is utilized at the receiver section before Digital to Analog Converter (DAC) block.
- ✓ Shift registers are also used as **counters**. Arithmetic operations
- ✓ Time delays
- ✓ Keyboard Encoder
- ✓ URAT- Universal Asynchronous Receiver Transmitter

10. Write the functions of shift registers and list its types

The two basic functions of shift registers are:

1. Data storage and
2. Data movement.

- ✓ The ***storage capacity*** of a register is the total number of bits (1s and 0s) of digital data it can retain. Each stage (flip-flop) a shift register represents one bit of storage capacity; therefore, the number of stages in register determines its storage capacity.

- ✓ The ***shift capability*** of a register permits the movement of data from stage to stage within the register or into or out of the register upon application of clock pulses.
- ✓ Figure 8-2 illustrates the types of data movement in shift registers. The block represents any arbitrary 4-bit register, and the arrows indicate the direction of data movement.

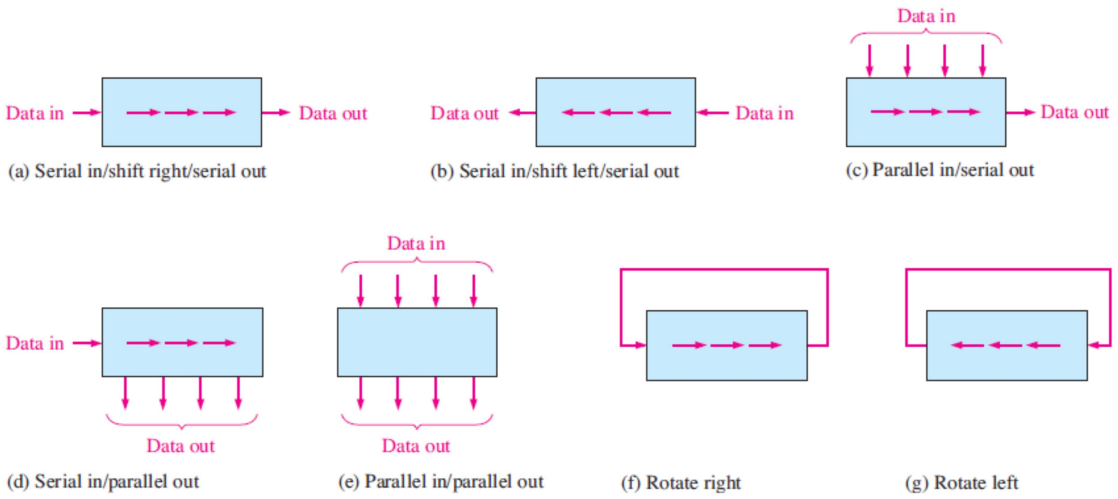


FIGURE 8–2 Basic data movement in shift registers. (Four bits are used for illustration. The bits move in the direction of the arrows.)

1. Serial-In/Serial-Out shift register (SISO)
2. Serial-In/Parallel-Out shift register (SIPO)
3. Parallel-In/Serial-Out shift register (PISO)
4. Parallel-In/Parallel-Out shift register (PIPO)

11. Explain the working of 4-bit serial in serial out (SISO) with logical circuit and truth table.

- ✓ The serial in/serial out shift register accept data serially -that is, one bit at a time on a single line. It produces the stored information on its output also in serial form.
- ✓ A basic 4-bit SISO shift register can be constructed using 4 D flip-flops, as shown in the below figure 8.3.
- ✓ The output of the one stage is connected to the input of the next stage and a common clock is used to activate all the flip-flops at a time. Input is applied at the first flip -flop (FF0) and output is taken at the last flip flop (FF3).

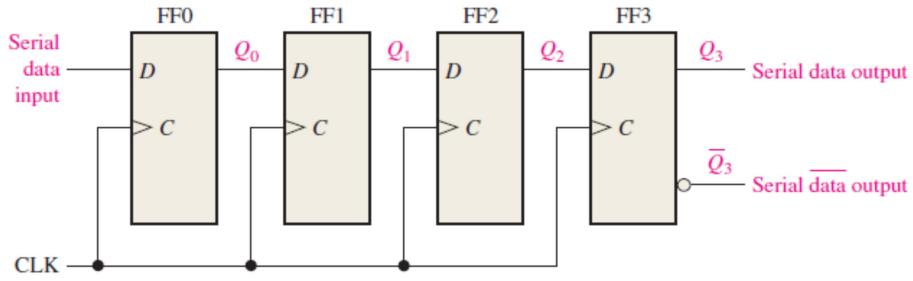


FIGURE 8–3 Serial in/serial out shift register.

Assuming for the 4-bit shift register above, we want to send the word "1010". After clearing the shift register, the output of all the flip flops becomes 0, so during the first clock cycle as we apply this data (1010) serially, the outputs of the flip flops look like the table below.

Clock Pulse	**Serial IN at D of FF0**	**FF0 (Q_0)**	**FF1 (Q_1)**	**FF2 (Q_2)**	**FF3 (Q_3)**	**Serial OUT at Q3 of FF3**
Register initially Clear	-	0	0	0	0	-
After CLK1 pulse	0 (LSB)	**0**	0	0	0	-
After CLK2 pulse	1	**1**	**0**	0	0	-
After CLK3 pulse	0	**0**	**1**	**0**	0	-
After CLK4 pulse	1 (MSB)	**1**	**0**	**1**	**0**	After CLK4, the 4 bit number 1010 is completely stored in register. 1st data bit out (0)
After CLK5 pulse	-	0	**1**	**0**	**1**	2nd data bit out (1)
After CLK6 pulse	-	0	0	**1**	**0**	3rd data bit out (0)
After CLK7 pulse	-	0	0	0	**1**	4th data bit out (1)
After CLK8 pulse	-	0	0	0	0	After CLK8, register is Clear

12. Explain the working of 4-bit serial in parallel out (SIPO) with logical circuit and truth table.

- ✓ The serial-in parallel out shift register accepts data serially (LSB first) – that is, one bit at a time on a single line. It produces the stored information on its output simultaneously in parallel form.
- ✓ A basic 4-bit SIPO shift register can be constructed using 4 D flip-flops, as shown in the below figure 8.6. The output of the one stage is connected to the input of the next stage and a common clock is used to activate all the flip lops at a time.
- ✓ Input is applied at the flip flops (FF0) and output is taken at the outputs of all the flips at a time represented as Q_0, Q_1, Q_2 and Q_3 (Q_0-MSB and Q_3-LSB).

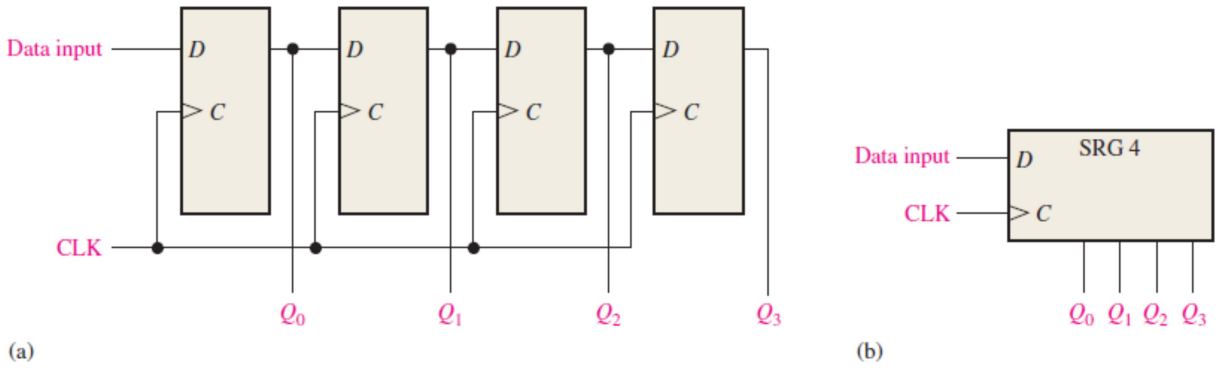
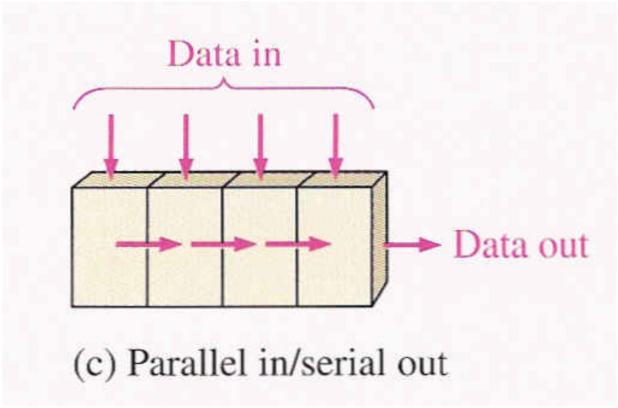


FIGURE 8–6 A serial in/parallel out shift register.

Clock Pulse	Serial IN at D of FF0	FF0	FF1	FF2	FF3	Parallel OUT at			
						Q_0	Q_1	Q_2	Q_3
Register initially Clear	-	0	0	0	0	-	-	-	-
After CLK1 pulse	0 (LSB)	**0**	0	0	0	**0**	-	-	-
After CLK2 pulse	1	**1**	**0**	0	0	**1**	**0**	1	-
After CLK3 pulse	0	**0**	**1**	**0**	0	**0**	**1**	**0**	-
After CLK4 pulse	1 (MSB)	**1**	**0**	**1**	**0**	**1**	**0**	**1**	**0**

13. Explain the working of 4-bit parallel in serial out (PISO) with logical circuit and truth table.

- ✓ The Parallel-In Serial-Out (PISO) shift register accepts data in parallel form -that is, all bits at a time on a multiple lines.
- ✓ It produces the stored information on its output in serial form as illustrated in the below figure 8.10.



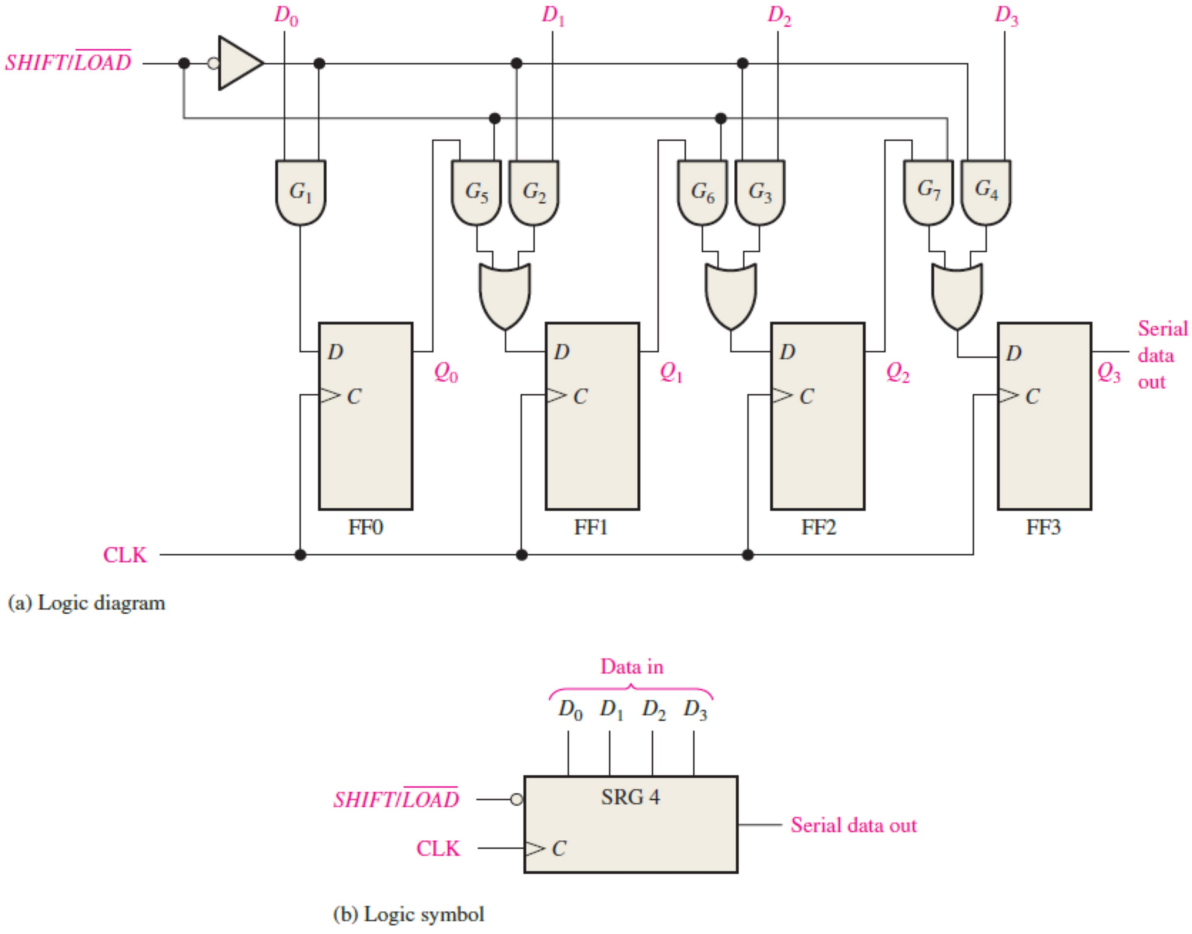


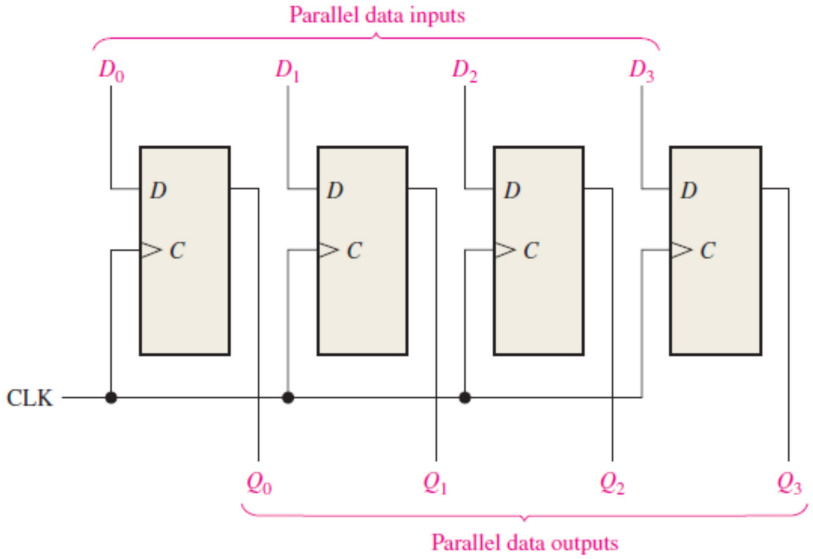
Figure 8–10 A 4-bit parallel in/serial out shift register.

- ✓ A basic 4-bit PISO shift register can be constructed using 4 D flip flops as shown in above figure.
- ✓ The four data input lines D_0 , D_1 , D_2 and D_3 and $SHIFT/\overline{LOAD}$ input, which allows four bits of data to **load** in parallel into the register.
- ✓ When $SHIFT/\overline{LOAD}$ is **LOW**, gates $\mathbf{G_1}$, $\mathbf{G_2}$, **G3** and $\mathbf{G_4}$ are enabled, allowing each data bit to be applied to the D input of its respective flip-flop.
- ✓ When a clock pulse is applied, the data is loaded and stored in flip-flop simultaneously.
- ✓ When $SHIFT/\overline{LOAD}$ is **HIGH**, gates $\mathbf{G_5}$, $\mathbf{G_6}$ and $\mathbf{G_7}$ are enabled, allowing the data bits to shift right from one flip-flop to the next flip-flop.
- ✓ The OR gates allow either the normal shifting operation or the parallel data-entry operation, depending on which AND gates are enabled by the level on the $SHIFT/\overline{LOAD}$ input.
- ✓ Notice that FF0 has a single AND to disable the parallel input, *D*0. It does not require an AND/OR arrangement because there is no serial data in.

Clock Pulse	**Parallel IN at D_0 , D_1 , D_2 , D_3**	**FF0**	**FF1**	**FF2**	**FF3**	**Serial OUT at Q_3**
Register initially Clear	-	0	0	0	0	
After CLK1 pulse	1010	1	0	1	**0**	First bit Out - **0**
After CLK2 pulse	-	-	1	0	**1**	Second bit Out - **1**
After CLK3 pulse	-	-	-	1	**0**	Third bit Out - **0**
After CLK4 pulse	-	-	-	-	**1**	First bit Out - **1**

14. Explain the working of 4-bit parallel in parallel out (PIPO) with logical circuit and truth table.

- ✓ In a parallel-in, parallel-out (PIPO) shift register, the data is entered into the register in parallel form, and also the data is taken out of the register in parallel form.
- ✓ A basic 4-bit PIPO shift register can be constructed using 4 D flip-flops, as shown in the below figure 8.14.
- ✓ A common clock is used to activate all the flip-flops at a time.
- ✓ Data is applied to the D input terminals of the FF's. When a clock pulse is applied, at the positive going edge of the pulse, the D inputs are shifted into the Q outputs of the FFs. The register now stores the data. The stored data is available instantaneously for shifting out in parallel form.
- ✓ The simultaneous entry of all data bits, the bits appear on the parallel outputs.



E 8–14 A parallel in/parallel out register.

Clock Pulse	D_0	D_1	D_2	D_3	Q_0	Q_1	Q_2	Q_3
Register initially Clear	0	0	0	0	0	0	0	0
After CLK1 pulse	**1**	**0**	**1**	**0**	**1**	**0**	**1**	**0**

15. Define Counter. Write its applications.

The **counter** is a digital sequential logic circuit in which a group of flip-flops are connected in cascade. The **basic function of the counter** is to count the number of input clock pulses applied. **OR**

Counter is a register which counts the sequence in binary form.

Applications of counter

1. Digital clock
2. Automobile parking control
3. Parallel to serial data conversion (multiplexing)
4. Traffic Signal Control.
5. A/D Converters
6. Frequency Meters/Counters
7. Signal Generators
8. Microprocessors

16. Compare Asynchronous and Synchronous counter.

Asynchronous counter	Synchronous counter
In Asynchronous counter, clock pulse is applied only to the initial flip flop whose value would be considered as LSB. The output of first flip-flop acts as a clock pulse to the next flip flop, whose output is used as a clock to the next in line flip-flop and so on.	In synchronous counter, clock pulse is applied to all Flip-Flops.
All flip- flops are not clocked simultaneously.	All flip -flops are clocked simultaneously.
Asynchronous Counter is slower than synchronous counter in operation.	Synchronous Counter is faster than asynchronous counter in operation.
Circuit is simple.	Circuit is complex or complicated as number of states increases.
Cost is less as compared to synchronous counters.	Cost is more as additional circuitry is required.
It is also known as a serial counter.	It is also known as a parallel counter.
It is slow in speed as compared to synchronous counter.	It is faster in speed as compared to asynchronous counter.
Speed is slow as compared to synchronous counters as clock signal is delayed for other flip-flops except the first one.	Speed is fast as no clock delay is provided to flip-flops.

Asynchronous counters

- ✓ The term ***asynchronous*** refers to events that do not have a fixed time relationship with each other.
- ✓ 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 counters commonly called **ripple counters**, the first flip-flop is clocked by the external clock pulse and then each successive flip-flop is clocked by the output of the preceding flip-flop.

A 2-Bit Asynchronous Binary Counter

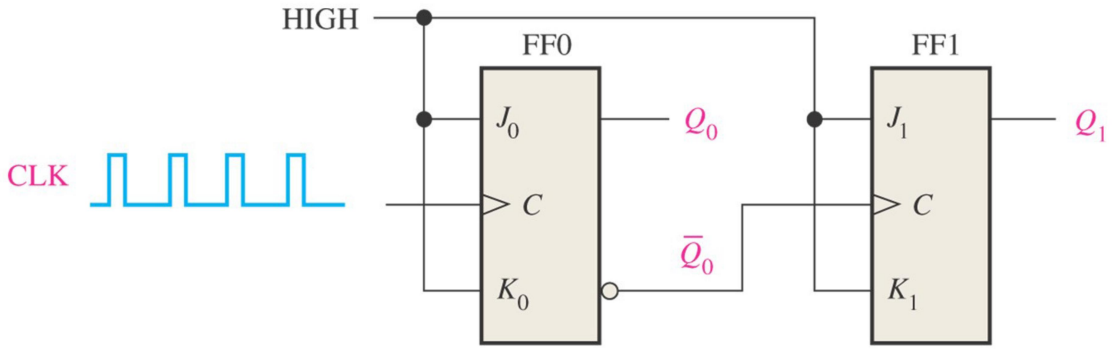


Fig1-1 2-bit asynchronous counter

- ✓ Fig1-1 shows a 2-bit counter connected for asynchronous operation.
- ✓ The clock (CLK) is applied to the clock input (C) of only the first flop-flop, **FF0**, which is always the least significant bit (LSB).
- ✓ The second flip-flop, FF1, is triggered by the $\overline{Q}_0$ out-put of FF0.
- ✓ FF0 changes state at the positive-going edge of each clock pulse. But FF1 changes only when triggered by a positive-going transition of the $\overline{Q}_0$ output of FF0. Because of the inherent propagation delay tie through a flip-flop, a transition of the input clock pulse (CLK) and a transition of the $\overline{Q}_0$ output of FF0 can never occur at exactly the same time. Therefore, the two flip-flops are never simultaneously triggered, so the counter operation is asynchronous.

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

Table 1-1Binary state sequence

The Timing Diagram

- ✓ Applying 4 clock pulses to FF0, Both flip-flops are connected for toggle operation (J=1, K=1) and initially RESET (Q LOW).
- ✓ The positive-going edge of CLK1 (clock pulse1) causes the $\mathbf{Q_0}$ output of FF0 to go HIGH. At the same time the $\mathbf{\bar{Q}_0}$ output goes LOW, but it has no effect on FF1 because a positive-going transition must occur to trigger the flip-flop.
- ✓ After the leading edge of CLK1, $\mathbf{Q_0=1}$ **&** $\mathbf{Q_1=0}$**.** The positive-going edge of CLK2 causes Q_0 to go LOW. $\bar{Q}_0$ goes HIGH and triggers FF1, causing Q1 to go HIGH.
- ✓ After the leading edge of CLK2, $\mathbf{Q_0=0}$ **&** $\mathbf{Q_1=1}$. The positive-going edge of CLK3 causes Q_0 to go HIGH again. Output $\bar{Q}_0$ goes LOW and has no effect on FF1.
- ✓ Thus, after the leading edge of CLK3, $\mathbf{\bar{Q}_0=1}$ **&** $\mathbf{Q_1=1.}$ The positive-going edge of CLK4 causes Q_0 to go LOW, while $\bar{Q}_0$ goes HIGH and triggers FF1, causing Q_1 to go LOW.
- ✓ After the leading edge of CLK4, $\mathbf{Q_0=0}$ **&** $\mathbf{Q_1=1}$. The 2-bit counter exhibits four different states, as you would expect with two flip-flops ($2^2 = 4$).

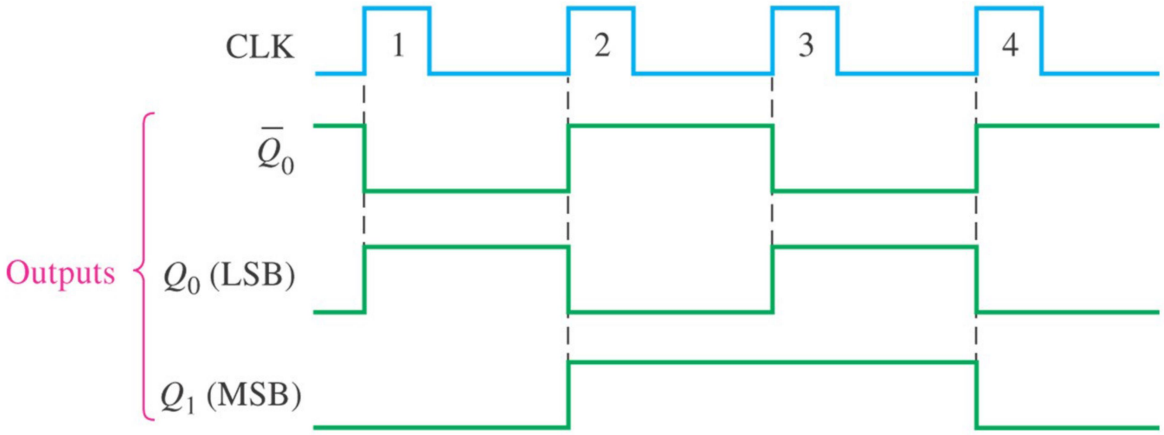


Fig 1-2 Timing diagram for the counter of Fig1-1

The fourth pulse it recycles to its original state (Q_0=0, Q_1=0). The term recycles; it refers to the transition of the counter from its final state back to its original state.

A 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

Table1-2 State sequence for a 3-bit binary counter

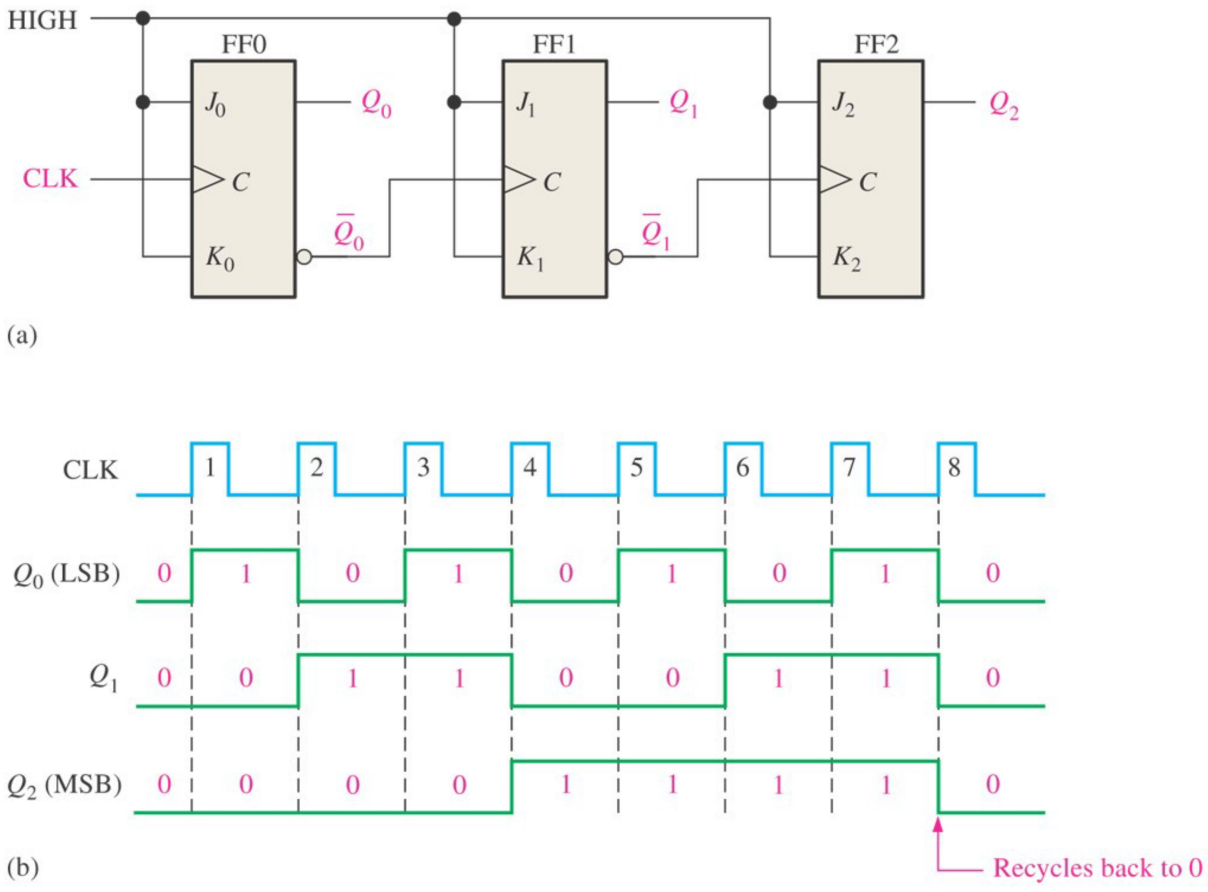


Fig 1-3 Three-bit asynchronous binary counter and its timing diagram for one cycle

16. Explain 4-bit asynchronous ripple counter.

- ✓ In 4-bit ripple counter, n value is 4 so, 4 JK flip flops are used and the counter can count up to 16 (2^4) pulses.
- ✓ Fig1-1 shows a 4-bit counter connected for asynchronous operation.
- ✓ Notice that the clock (CLK) is applied to the clock input (C) of only the first flop-flop, FF0, which is always the least significant bit (LSB).
- ✓ The second flip-flop, FF1, is triggered by the Q0 out-put of FF0.
- ✓ FF0 changes state at the negative-going edge of each clock pulse.
- ✓ But FF1 changes only when triggered by a positive-going transition of the Q0 output of FF0. Because of the inherent propagation delay tie through a flip-flop, a transition of the input clock pulse (CLK) and a transition of the Q0 output of FF0 can never occur at exactly the same time. Therefore, the two flip-flops are never simultaneously triggered, so the counter operation is asynchronous.

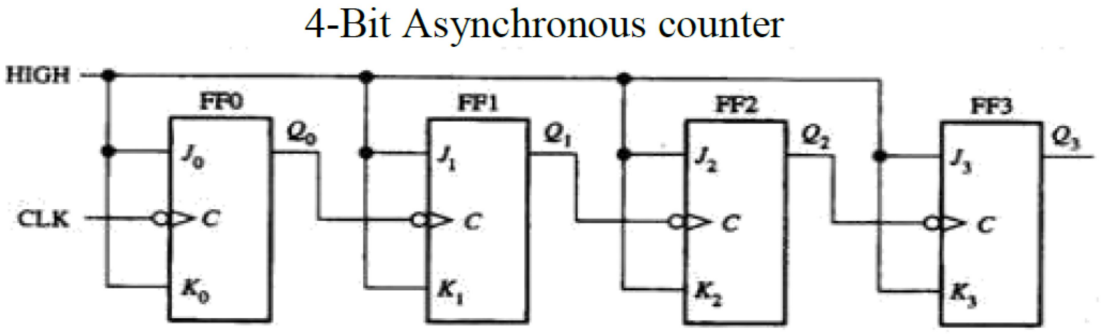


Fig1-1 a 4-bit counter

Clock Pulse	Q3	Q2	Q1	Q0
Initially	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
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

Table1-2 State sequence for a 4-bit binary counter

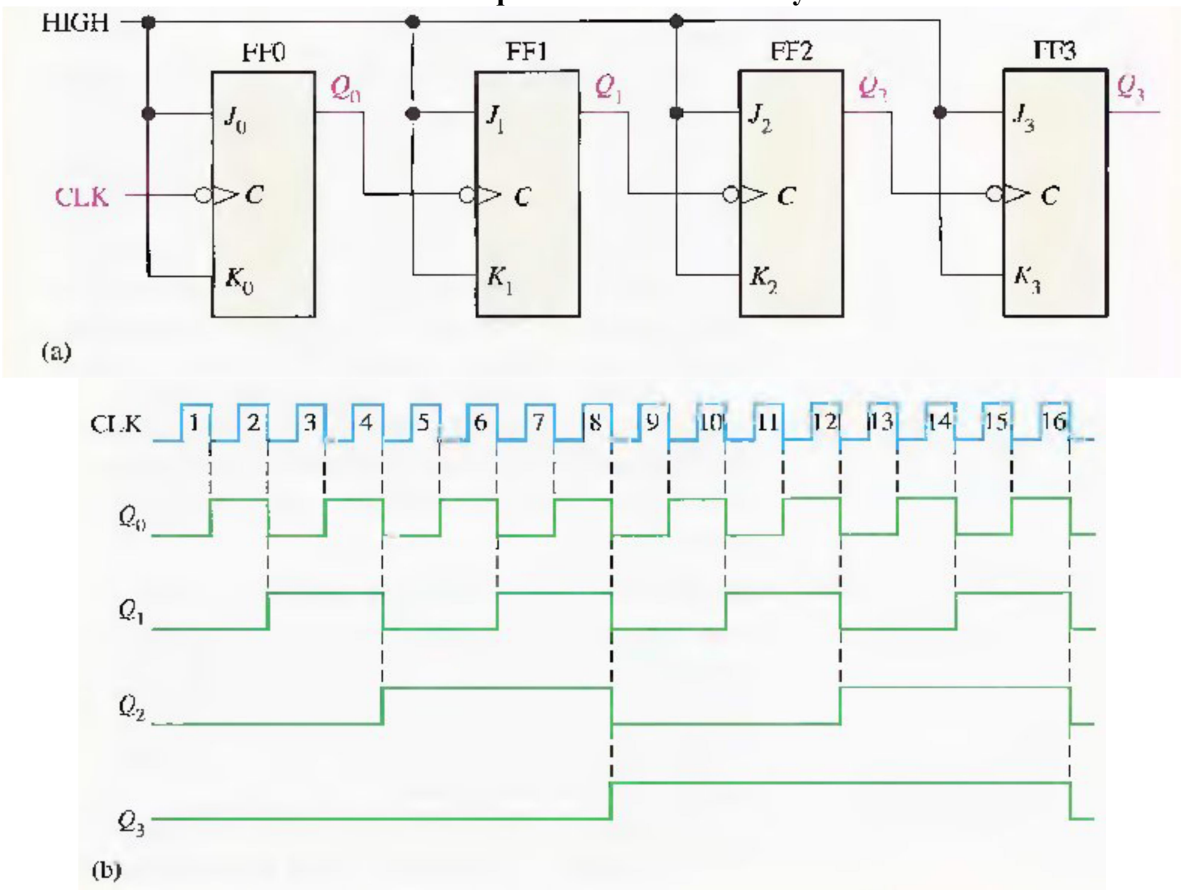
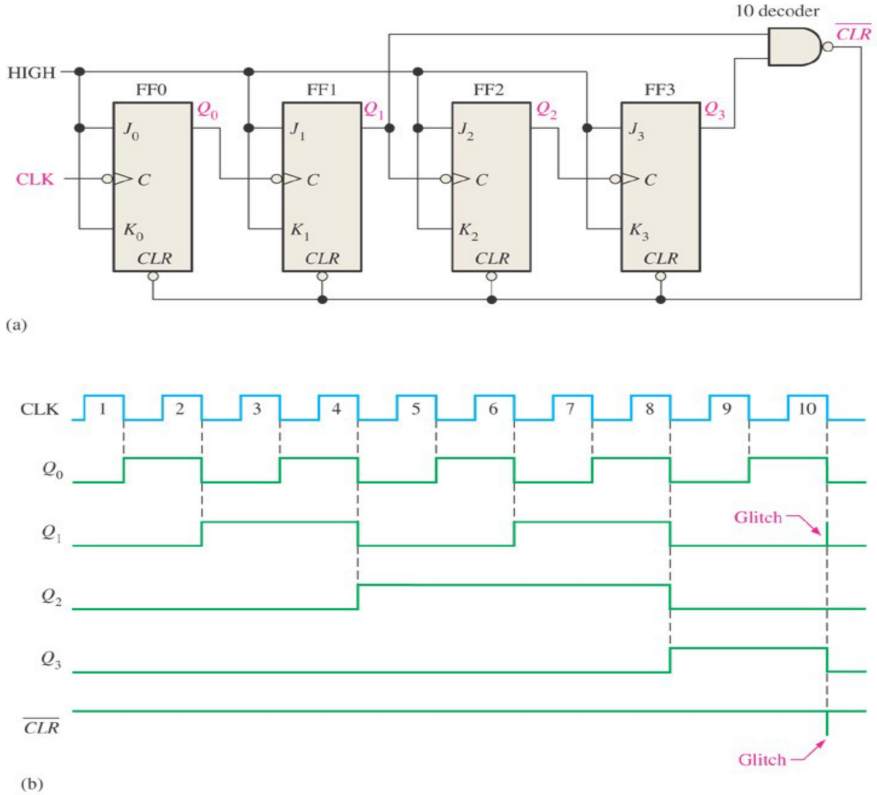


Fig 1-5 4-bit asynchronous binary counter and its timing diagram.

Asynchronous Decade Counters

- ✓ The modulus is the number of unique states through which the counter will sequence. The maximum possible number of states of a counter is 2^n where n is the number of flip-flops.
- ✓ 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. One common modulus for counters with truncated sequences is 10 (Modules10).
- ✓ A decade counter with a count sequence of zero (0000) through 9 (1001) is a BCD decade counter because its 10-state sequence produces the BCD code. To obtain a truncated sequence, it is necessary to force the counter to recycle before going through all of its possible states. A decade counter requires 4 flip-flops. One way to make the counter recycle after the count of 9 (1001) is to decode count 10 (1010) with a NAND gate and connect the output of the NAND gate to the clear (CLR) inputs of the flip-flops, as shown in Fig1-6(a).

Partial Decoding: in Fig1-6(a) only Q1 & Q3 are connected to the NAND gate inputs. This arrangement is an example of partial decoding, in which the two unique states (**Q_1=1 & Q_3=1**) are sufficient to decode the count of 10 because none of the other states (0 through 9) have both **Q_1 & Q_3** HIGH at the same time.



Clock Pulse	Q3	Q2	Q1	Q0
Initially	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
10 (recycles)	0	0	0	0

Fig 1-6 an asynchronously clocked decade counter with asynchronous recycling.

Synchronous Counters

- ✓ In synchronous counters, the clock input is connected to all of the flip-flops so that they are clocked simultaneously.
- ✓ The term synchronous refers to events that have a fixed time relationship with each other
- ✓ J-K flip-flops are used to illustrate most synchronous counters.

A 2-Bit Synchronous Binary Counter

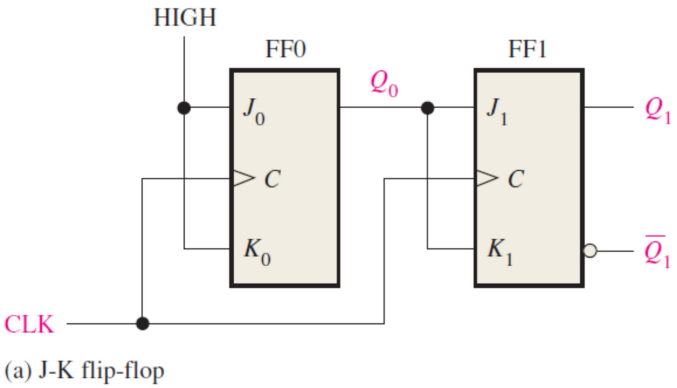


Fig 1-8 A 2-bit synchronous binary counter

- ✓ First, when the positive edge of the first clock pulse is applied, FF0 will toggle and Q_0 will therefore go HIGH. When FF1at the positive-going edge of CLK1 inputs J1 & K1 are both LOW because Q0 has not yet gone HIGH. So, J1=0 & K1=0. This is a no-change condition, and therefore FF1 does not change state.*(fig1-9a).*
- ✓ When the leading edge of CLK2 occurs, FF0 will toggle and Q_0 will go LOW and Q1 goes HIGH. Thus, after CLK2, Q_0=0 & Q_1=1 (Fig1-9b).
- ✓ When the leading edge of CLK3 occurs, FF0 again toggles to the SET state (Q_0 = 1), and FF1 remains SET (Q_1=1. After this triggering edge, Q_0 =1 & Q_1 =1 (Fig1-9c). At the leading edge of CLK4, Q_0 & Q_1 go LOW (Fig1-9d).

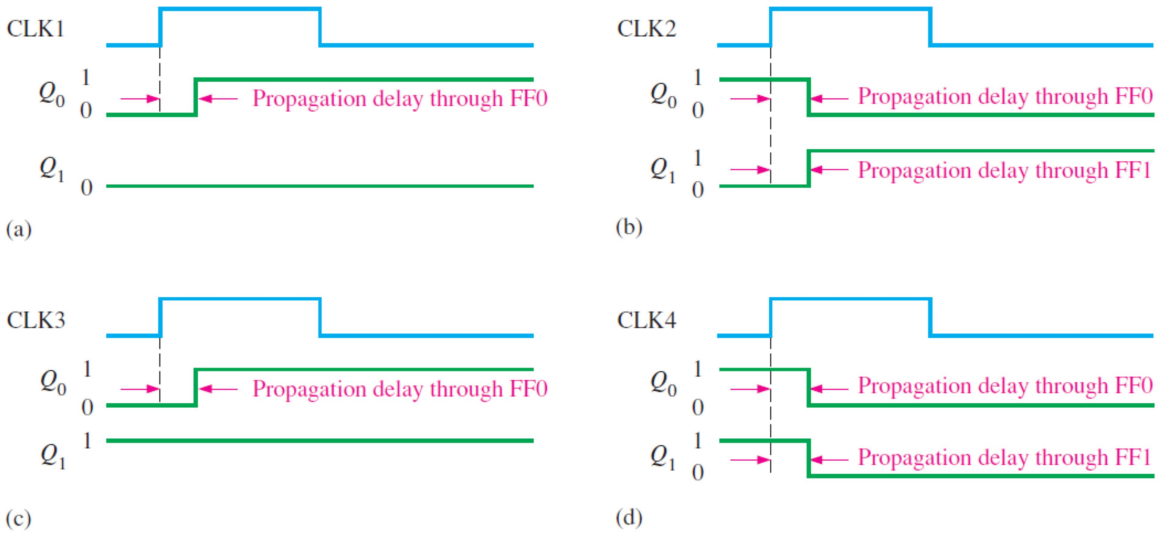


Fig 1-9 timing details for the1-bit synchronous counter operation

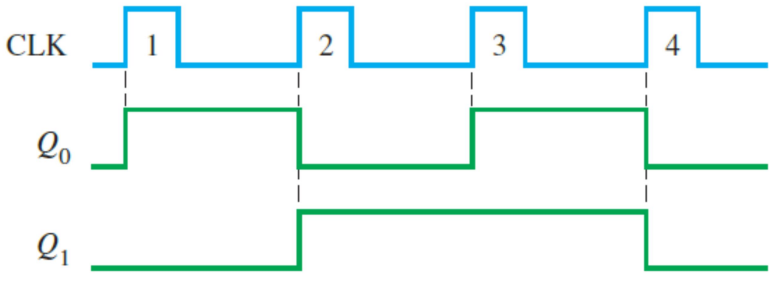


Fig 1-10 complete timing diagram for the counter

A 3-Bit Synchronous Binary Counter

In the 3-bit synchronous counter, we have used three j-k flip-flops. As in the diagram, The J and K inputs of FF0 are connected to HIGH. The inputs J and K of FF1 are connected to the output of FF0, and the J and K inputs of FF2 are connected to the output of an AND gate, which is fed by the outputs of FF0 and FF1.

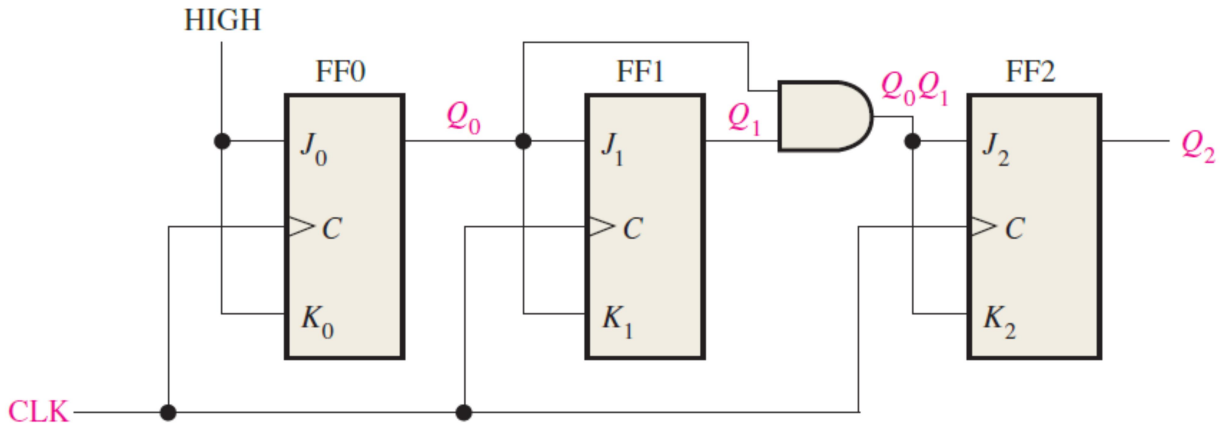


Fig1-11 3-bit synchronous counter

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

Table1-3 Binary state sequence

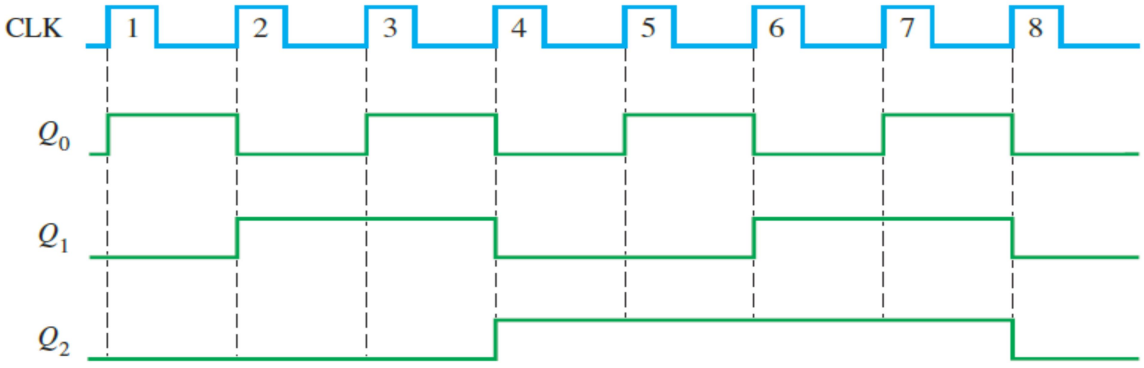


Fig1-12 the timing diagram

17. Explain 3-bit synchronous up/down counter.

- ✓ An **up/down (bidirectional)** counter is one that is capable of progressing in either direction through a certain sequence.
- ✓ A synchronous up-down counter has an **up-down** control input. It is used to control the direction of the counter through a certain sequence.

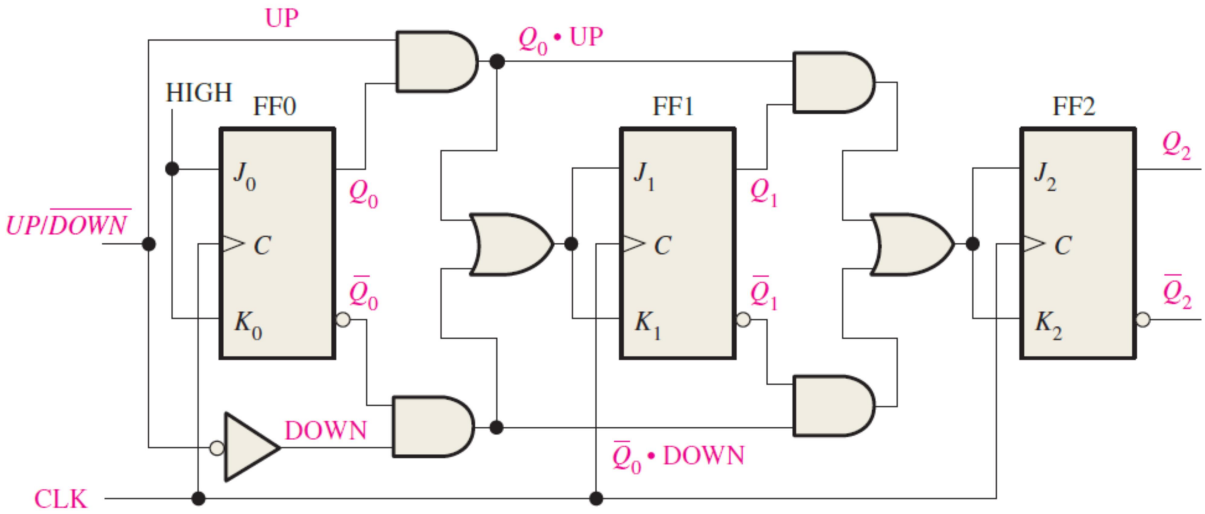


Figure 3.6 **A basic 3-bit up/down synchronous counter.**

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	

An examination of Q_0 for both the up and down sequences shows that FF0 toggles on each clock pulse. Thus, the J_0 and K_0 inputs of FF0 are

$J_0 = K_0 = 1$

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 FF1 must equal 1 under the conditions expressed by the following equation:

$$J_1 = K_1 = (Q_0 \cdot \text{UP}) + (\overline{Q}_0 \cdot \text{DOWN})$$

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:

$$J_2 = K_2 = (Q_0 \cdot Q_1 \cdot \text{UP}) + (\overline{Q}_0 \cdot \overline{Q}_1 \cdot \text{DOWN})$$

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 3.6 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

$UP/\overline{DOWN}$ control input is HIGH for UP and LOW for DOWN.

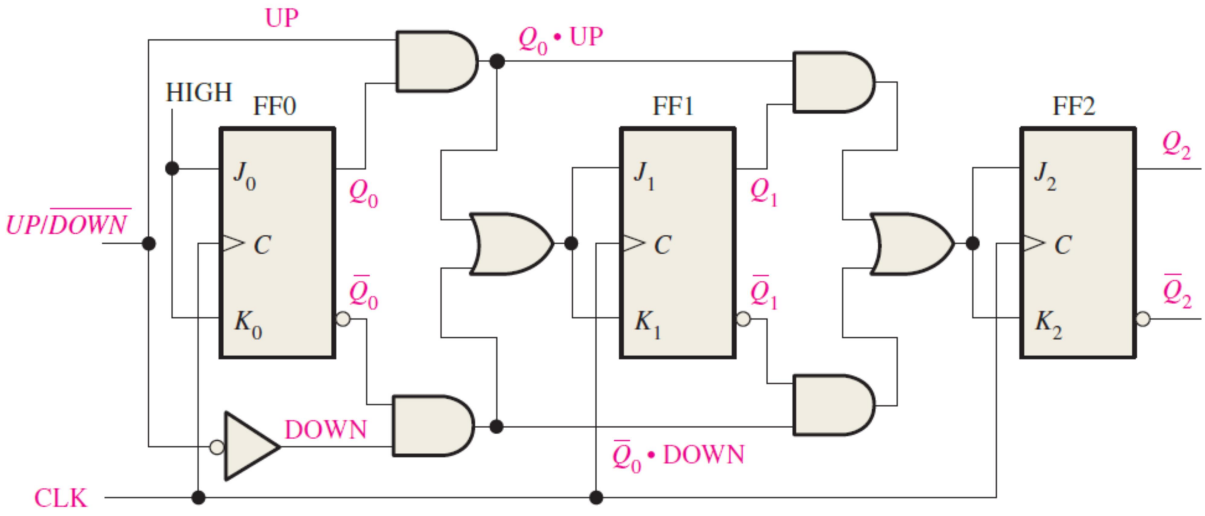
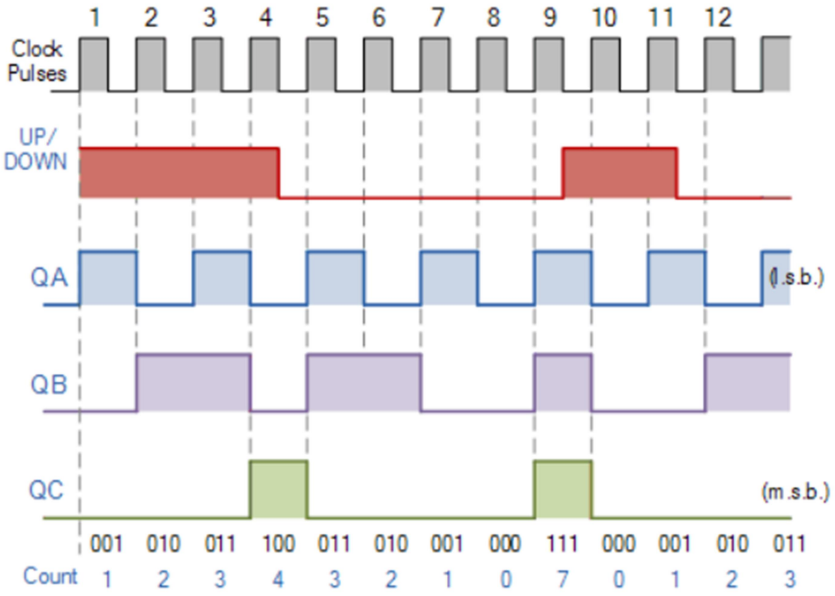
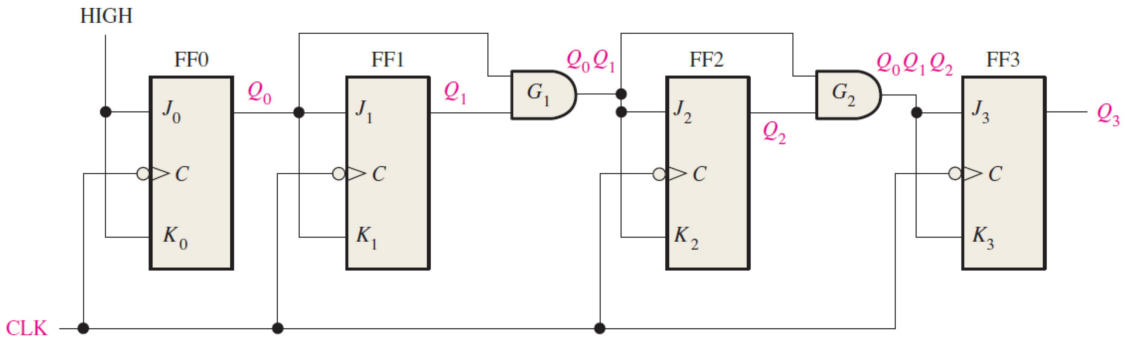


Figure 3.6 A basic 3-bit up/down synchronous counter.



A 4-Bit Synchronous Binary Counter



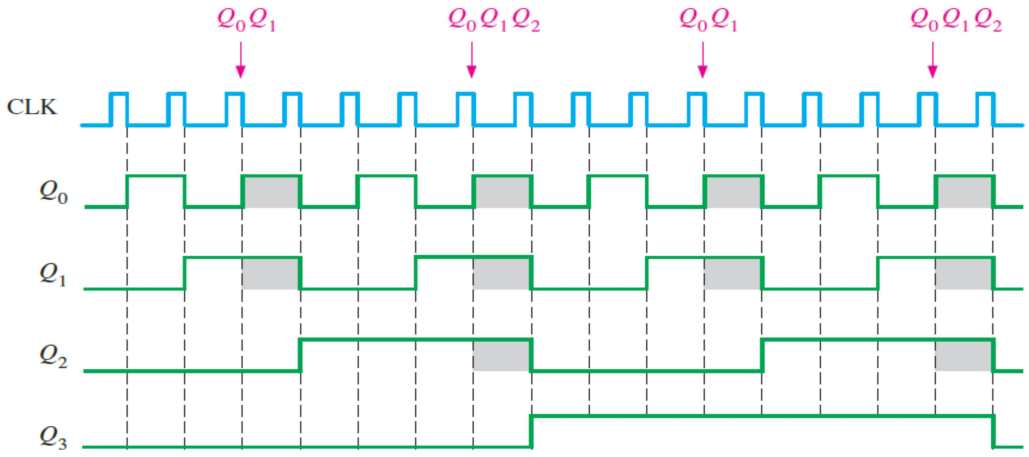
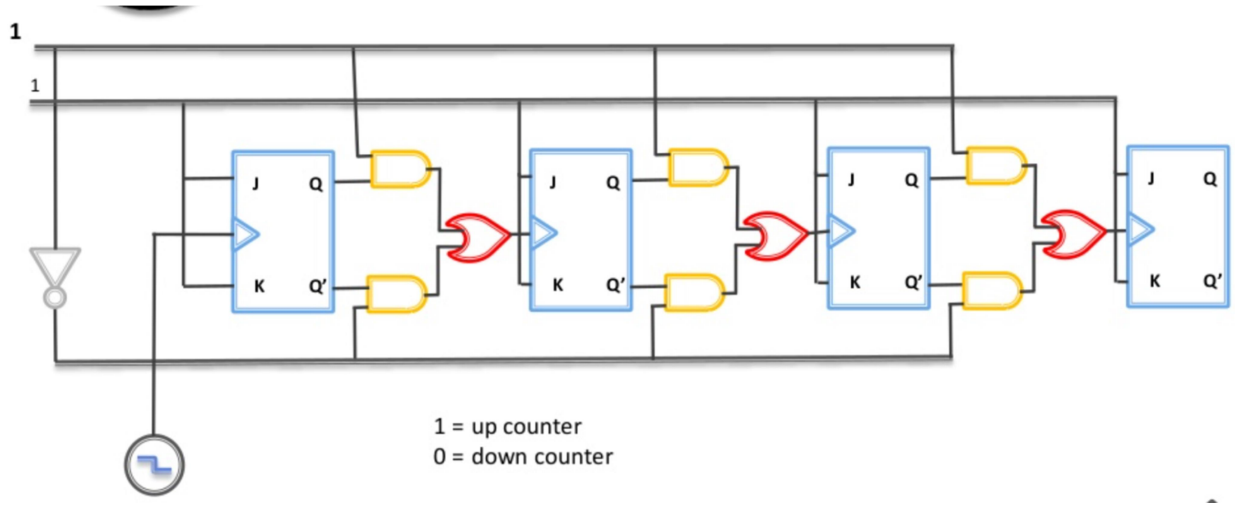
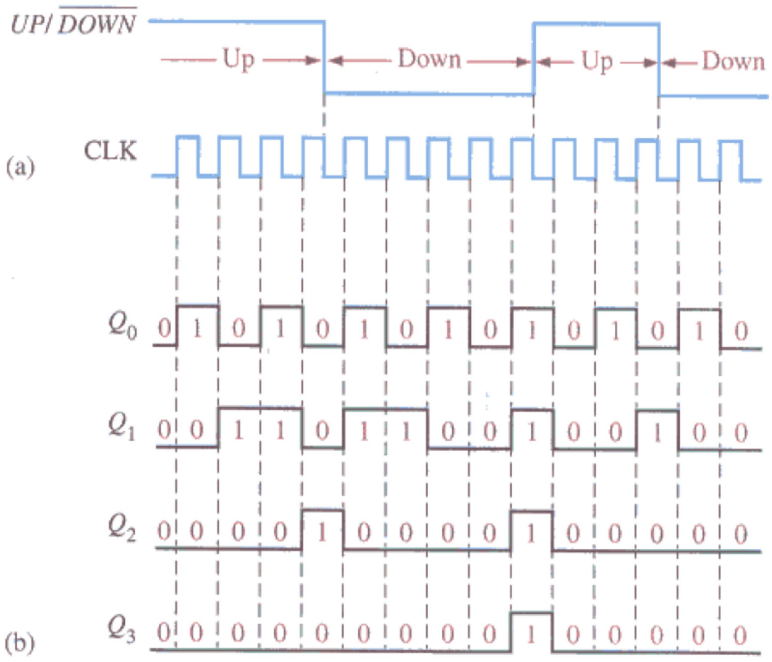


Fig1-13 a 4-bit synchronous binary counter and timing diagram. Points where the AND gate outputs are HIGH are indicated by the shaded areas



Example: 4-bit synchronous up-down counter [Floyd]



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	