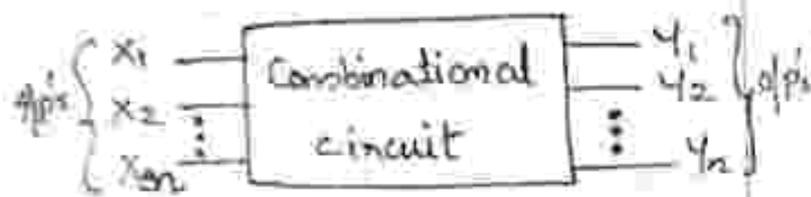


Switching circuits may be split into two categories

Combinational Logic Circuits

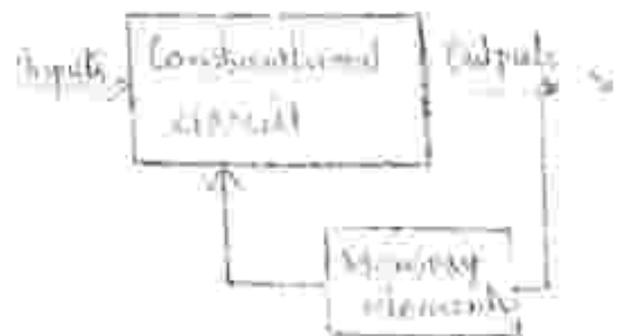
In combinational circuits, the output variables at any instant of time are dependent only on the present input variables.



Block diagram of combinational logic circuit

Sequential Logic

In sequential circuits, the output variables at any instant of time are dependent not only on the present input variables, but also on the present state, i.e. on the past history of the system.



Block diagram of sequential circuit

① Memory unit is not required.

② They are faster because the delay between the input & the output is due to the propagation delay of gates only.

③ Easy to design.

① Memory unit is required to store the past history.

② Sequential circuits are slower than combinational circuits.

③ Comparatively harder to design.

Classification of Sequential Circuits.

They are classified as ① Synchronous Sequential Circuit
② Asynchronous Sequential Circuit.

Table 10.2 Comparison between synchronous and asynchronous sequential circuits

Synchronous sequential circuits	Asynchronous sequential circuits
1. In synchronous circuits, memory elements are clocked FFs.	1. In asynchronous circuits, memory elements are either unclocked FFs or time delay elements.
2. In synchronous circuits, the change in input signals can affect memory elements upon activation of clock signal.	2. In asynchronous circuits, change in input signals can affect memory elements at any instant of time.
3. The maximum operating speed of the clock depends on time delays involved.	3. Because of the absence of the clock, asynchronous circuits can operate faster than synchronous circuits.
4. Easier to design.	4. More difficult to design.

or

Latches & Flipflops.

Sequential circuits are made up of combinational circuits & memory elements. The most important memory element is flipflop which is made up of an assembly of logic gates. Logic gate by itself has no storage capacity. Flip-flops are the basic building blocks of sequential circuits.

Flip flops

Logic gates can be connected together in a way that permit information to be stored.

There are different gate arrangements that are used to construct flip-flops in a wide variety of ways.

A flip flop, known as a bistable multivibrator, has two stable states. It can remain in either of the states indefinitely. Its states can be changed by applying the proper triggering signal.

General type of Symbol used for a Flip-flop



Flip-flop has 2 o/p's :- Q & \bar{Q} . FF are have one or more inputs.

Normal o/p is Q output \bar{Q} is the inverted output.

O/p as well as complement are available for each FF.

Flip-flop

State of flip-flop always refers to the state of the normal output Q .

The inverted output \bar{Q} is in the opposite state.

A flip-flop is said to be in HIGH state or logic 1 state or SET state when $\bar{Q} = 1$. Flip-flop stores a 1 when its Q output is a 1.

FF in low state or logic 0 state or RESET state when $Q = 0$. FF stores a 0 when its Q output is a 0.

The input signals which command the flip-flop to change state are called excitations.

Latch

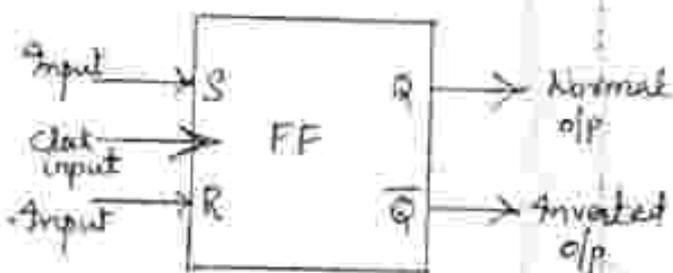
The term 'latch' is used for certain flipflops

It refers to non-clocked flipflops, because these flipflops 'latch on' to a 1 or a 0 immediately upon receiving the input pulse called SET or RESET.

They are not dependent on the clock signal.

Flip-flop

Flipflop is a sequential device that normally samples its inputs and changes its outputs only at times determined by clock pulses.

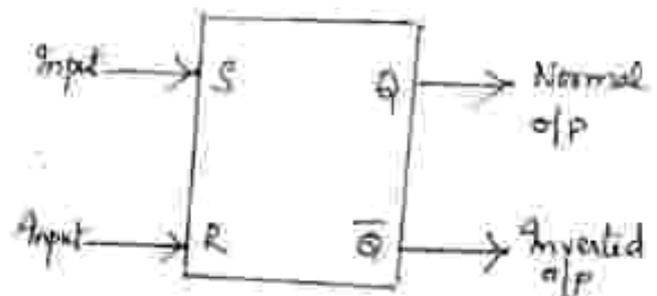


No symbol for FF

Latch

Latch is a sequential device that checks all its inputs continuously & changes its outputs accordingly at any time independent of a clock signal.

Latch refers to non-clocked flipflop.

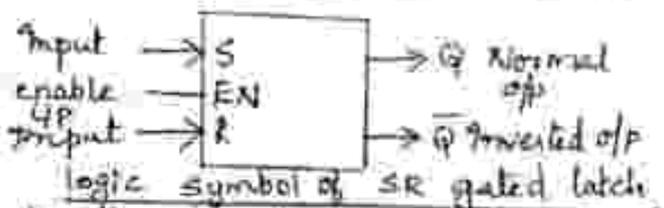


Gated Latches

Also called Clocked Flipflops

They are latches which respond to the inputs and latch on to a 1 or a 0 only when they are enabled, i.e. only when the input ENABLE or gating signal is HIGH.

In the absence of Enable or gating signal, the latch does not



logic symbol of SR gated latch

respond to the changes in its inputs.

(The gating signal may be a clock pulse).

Active High Input latch

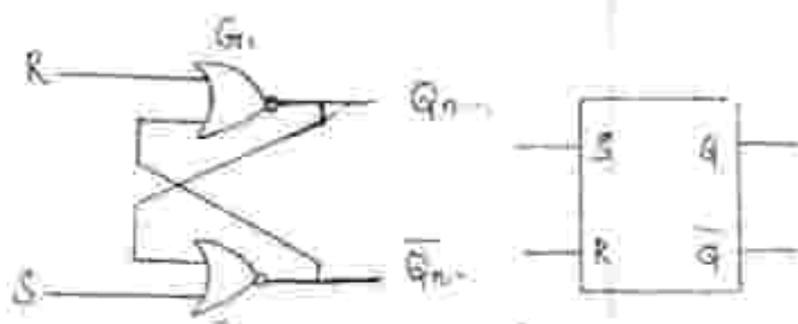
Means that the SET & RESET inputs are normally resting in the LOW state & one of them will be pulsed HIGH whenever we want to change the latch outputs.

Active Low Input latch

Means that the SET & RESET inputs are normally resting in the HIGH state & one of them will be pulsed low whenever we want to change the latch outputs.

A Latch may be an active - HIGH input latch or an active - Low input latch.

SR Latch With NOR Gate (Active High SR Latch)



(a) Logic diagrams

Inputs		Outputs	
S	R	Qn	Qn-bar
0	0	No Change	
0	1	0	1
1	0	1	0
1	1	FORBIDDEN	

(c) Truth table

The SR latch is composed of two cross-coupled NOR gates. The latch works as per the truth table.

Analysis of operation of Active High NOR latch

Case I [S=0 R=0]

⇒ Assume initially the latch is in SET state i.e. $Q_n = 1, \bar{Q}_n = 0$

If $S=0, R=0$ is the input conditions.
For gate G_1 the inputs are 'R' & 'feed back from \bar{Q}_n '

ie $G_1 \rightarrow R=0, \bar{Q}_n=0$. Then the o/p of $G_1=1$ ie $Q_{n+1}=1$

(set state) - output)

For gate G_2 the inputs are 'S' & 'feed back from \bar{Q}_n '
ie $G_2 \rightarrow S=0, \bar{Q}_n=1$. Then the o/p of $G_2=0$ ie $Q_{n+1}=0$

\Rightarrow If assume latch is initially in RESET state ie $Q_n=0$ & $\bar{Q}_n=1$

For $G_1 \rightarrow R=0, \bar{Q}_n=1$; then the o/p of $G_1=0$ ie $Q_{n+1}=0$

For $G_2 \rightarrow S=0, \bar{Q}_n=0$, then the o/p of $G_2=1$ ie $Q_{n+1}=1$

Hence we can conclude the latch remains in the same state when $S=R=0$

Case II ($S=1, R=0$)

$\Rightarrow Q_n=1, \bar{Q}_n=0$ (latch in SET state)

For $G_1 \rightarrow R=0, \bar{Q}_n=0$, o/p of $G_1=1$ ie $Q_{n+1}=1$

For $G_2 \rightarrow S=1, \bar{Q}_n=1$, o/p of $G_2=0$ ie $Q_{n+1}=0$

$\Rightarrow Q_n=0, \bar{Q}_n=1$ (latch in RESET state)

For $G_2 \rightarrow S=1, \bar{Q}_n=0$ o/p of $G_2=0$ ie $Q_{n+1}=0$

For $G_1 \rightarrow R=0, \bar{Q}_n=1$, o/p of $G_1=1$ ie $Q_{n+1}=1$

Hence we can conclude the latch remains in the SET state when $S=1$ & $R=0$ input condition.

Case III ($S=0, R=1$)

$\Rightarrow Q_n=1, \bar{Q}_n=0$ (Latch in SET state)

For gate $G_1 \rightarrow R=1, \bar{Q}_n=0$, o/p of $G_1=0$ ie $Q_{n+1}=0$

For Gate $G_2 \rightarrow S=0, Q_n=0$, o/p of $G_2 = 1$ i.e. $\overline{Q_{n+1}} = 1$ 4

$\Rightarrow Q_n = 0, \overline{Q_n} = 1$ (Latch is in RESET state)

For Gate $G_1 \Rightarrow R=1, \overline{Q_n}=1$, o/p of $G_1 = 0$ i.e. $Q_{n+1} = 0$

For Gate $G_2 \Rightarrow S=0, Q_n=0$, o/p of $G_2 = 1$ i.e. $\overline{Q_{n+1}} = 1$

Hence we can conclude that the latch remains in the RESET state when $S=0$ & $R=1$

Case IV ($S=1, R=1$)

$\Rightarrow Q_n = 1, \overline{Q_n} = 0$ (Latch is in SET state)

For Gate $G_1 \Rightarrow R=1, \overline{Q_n}=0$, o/p of $G_1 = 0$ i.e. $Q_{n+1} = 0$

For Gate $G_2 \Rightarrow S=1, Q_n=0$, o/p of $G_2 = 0$ i.e. $Q_{n+1} = 0$

$\Rightarrow Q_n = 0, \overline{Q_n} = 1$ (Latch is in RESET state)

For Gate $G_1 \rightarrow R=1, \overline{Q_n}=1$, o/p of $G_1 = 0$ i.e. $Q_{n+1} = 0$

For Gate $G_2 \rightarrow S=1, Q_n=0$, o/p of $G_2 = 0$ i.e. $\overline{Q_{n+1}} = 0$

Hence we can see that in all conditions;

when $S=R=1$, the output is invalid state

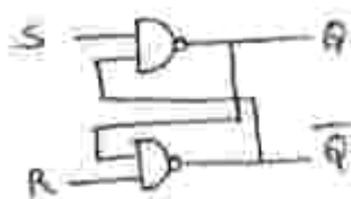
Note

Present state (PS) - state of flipflop before applying input, Q_n

Next state (NS) - state of flipflop after applying input, Q_{n+1}

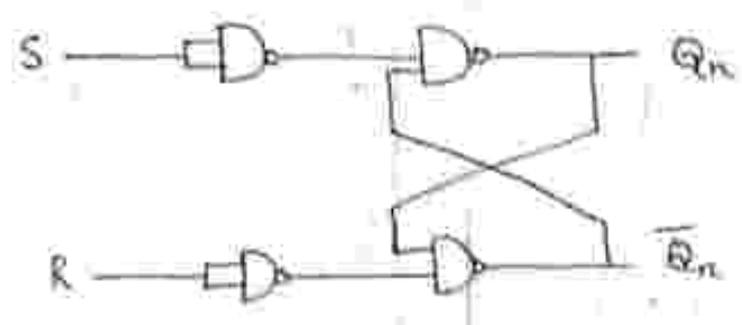
NAND Gate SR Latch (Active Low SR Latch)

An active low SR latch is shown below.



The operation of this latch is the reverse of the operations of NOR gate latch.

An active-low NAND latch can be converted to an active high NAND latch by inserting an inverter at the SR inputs as shown below.



Inputs		Outputs
S	R	
0	0	No change
0	1	0
1	0	1
1	1	Forbidden

Working of NAND Based SR latch

Case I (S=0, R=0)

⇒ $Q_n = 1$ & $\bar{Q}_n = 0$ (Latch is in set state)

The input for G_1 is \bar{S} & \bar{Q}_n

The input for G_2 is \bar{R} & Q_n

For Gate $G_1 \rightarrow \bar{S} = 1$ & $\bar{Q}_n = 0$ o/p for $G_1 \rightarrow Q_{n+1} = 1$

For Gate $G_2 \rightarrow \bar{R} = 1$ & $Q_n = 1$ o/p for $G_2 \rightarrow \bar{Q}_{n+1} = 0$

ie o/p is in SET state

⇒ $Q_n = 0$ & $\bar{Q}_n = 1$ (Latch is in Reset state)

For Gate $G_1 \rightarrow \bar{S} = 1, \bar{Q}_n = 1$, o/p for $G_1 \Rightarrow Q_{n+1} = 0$

For Gate $G_2 \rightarrow \bar{R} = 1, Q_n = 0$, o/p for $G_2 \Rightarrow \bar{Q}_{n+1} = 1$

ie o/p is in RESET state

Hence we conclude that when $S=0, R=0$ the input condition, the o/p remains in no change condition (NC).

Case II (S=1, R=0)

⇒ $Q_n = 1$ & $\bar{Q}_n = 0$ (Latch is in set state)

For Gate $G_1 \rightarrow \bar{S} = 0$ & $\bar{Q}_n = 0$ then o/p for $G_1 \rightarrow Q_{n+1} = 1$

For Gate $G_2 \rightarrow \bar{R}=1, Q_n=1, \text{ o/p for } G_2 \rightarrow \bar{Q}_{n+1}=0$
 $\Rightarrow Q_n=0, \bar{Q}_n=1$ (Latch is in RESET state)

For Gate $G_1 \rightarrow \bar{S}=0, \bar{Q}_n=1, \text{ o/p for } G_1 \rightarrow Q_{n+1}=1$

For Gate $G_2 \rightarrow \bar{R}=1, Q_n=1, \text{ o/p for } G_2 \rightarrow \bar{Q}_{n+1}=0$

Hence we can conclude that with $S=1$ & $R=0$ input condition flip flop is in SET state.

Case III ($S=0, R=1$)

$\Rightarrow Q_n=1, \bar{Q}_n=0$ (Latch is in SET state)

For Gate $G_2 \rightarrow \bar{R}=0, Q_n=1, \text{ o/p for } G_2 \rightarrow \bar{Q}_{n+1}=1$

For Gate $G_1 \rightarrow \bar{S}=1, \bar{Q}_n=1, \text{ o/p for } G_1 \rightarrow Q_{n+1}=0$

$\Rightarrow Q_n=0, \bar{Q}_n=1$ (Latch is in RESET state)

For Gate $G_2 \rightarrow \bar{R}=0, Q_n=0, \text{ o/p for } G_2 \rightarrow \bar{Q}_{n+1}=1$

For Gate $G_1 \rightarrow \bar{S}=1, \bar{Q}_n=1, \text{ o/p for } G_1 \rightarrow Q_{n+1}=0$

Hence we can conclude that with $S=0, R=1$ input condition flip flop is in RESET state.

Case IV ($S=1, R=1$)

$\Rightarrow Q_n=1, \bar{Q}_n=0$ (Latch is in SET state)

For Gate $G_2 \rightarrow \bar{R}=0, Q_n=1, \text{ o/p for } G_2 \rightarrow \bar{Q}_{n+1}=1$

For Gate $G_1 \rightarrow \bar{S}=0, \bar{Q}_n=1, \text{ o/p for } G_1 \rightarrow Q_{n+1}=1$

Unpredictable output state

$\Rightarrow Q_n=0, \bar{Q}_n=1$ (Latch is in RESET state)

For Gate $G_1 \rightarrow \bar{S}=0, \bar{Q}_n=1, \text{ o/p for } G_1 \rightarrow Q_{n+1}=1$

For Gate $G_2 \rightarrow \bar{R}=0, Q_n=1, \text{ o/p for } G_2 \rightarrow \bar{Q}_{n+1}=1$

unpredictable output state.

Hence we can conclude that with $S=1, R=1$ input condition, the output of the latch is indeterminate (forbidden state).

Truth table of SR latch

S	R	PS Q_n	NS Q_{n+1}	State
0	0	0	0	No change Q_n
0	0	1	1	
0	1	0	0	Reset
0	1	1	0	
1	0	0	1	Set
1	0	1	1	
1	1	0	X	Invalid Indeterminate
1	1	1	X	

19/07

Excitation Table

P.S Q_n		N.S Q_{n+1}	
S	R	S	R
0	X	0	0
1	0	1	0
0	1	0	1
1	0	X	0

The excitation table includes the excitations (input signals) required to take flipflop from PS to NS.

Characteristic Equation-

The characteristic Equation of a flipflop is the equation expressing the next state of a flipflop in terms of its present state & present excitations.

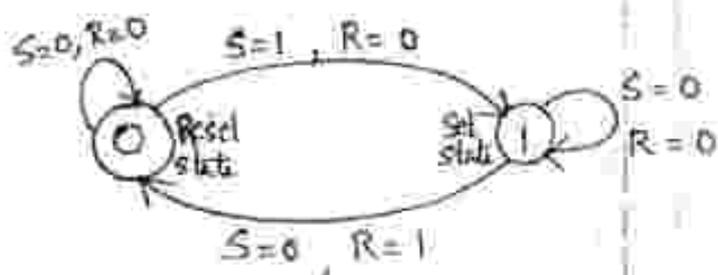
To obtain the characteristic equation

- 1) Write the excitation table of the flipflop.
- 2) Draw K-map for the next state of the flip-flop in terms of its present state & inputs.
- 3) Simplify it.

	SR			
Q_n	00	01	11	10
0				1
1	1			1

$$Q_{n+1} = S\bar{R} + Q_n\bar{R} = \underline{\underline{R(S + Q_n)}}$$

State diagram of SR latch



→ SR latch or flipflop is having 2 states, SET & RESET.

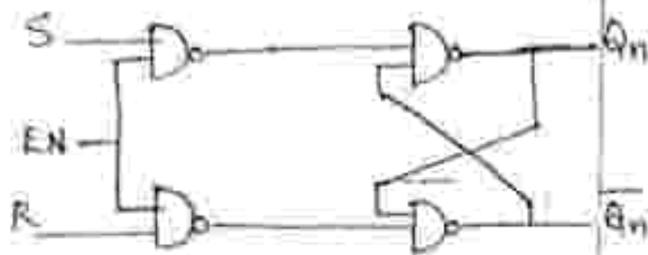
→ On applying SET input FF set to 1.

→ On applying Reset input FF reset to 0.

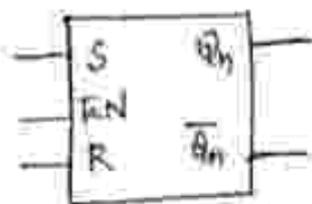
Gated Latches (clocked Flip-Flop)

- A gated SR latch requires an ENABLE (EN) input. Its S & R inputs will control the state of the flip-flop only when the ENABLE is HIGH.
- When the ENABLE is LOW, the inputs become ineffective.

* no change of state takes place.
 → The ENABLE input may be a clock. So a gated SR latch is also called a clocked SR latch or synchronous SR latch.



(a) Logic diagram



(b) Logic symbol

EN	S	R	Q_n	Q_{n+1}	State
1	0	0	0	0	No change
1	0	0	1	1	
1	0	1	0	0	Reset
1	0	1	1	0	
1	1	0	0	1	Set
1	1	0	1	1	
1	1	1	0	X	Forbidden state
1	1	1	1	X	
0	X	X	0	0	No change
0	X	X	1	1	

(c) Truth table

Triggering of Flipflops

2 types of triggering is used in Flipflops.

- 1) Level triggering
- 2) Edge triggering

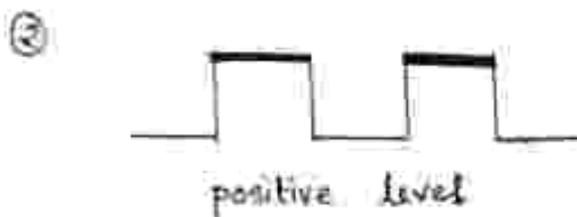
Level triggering

In level triggering the clock input triggers the flipflop when the clock pulse goes HIGH. They are of 2 types

- ① positive level triggered
- ② negative level triggered.

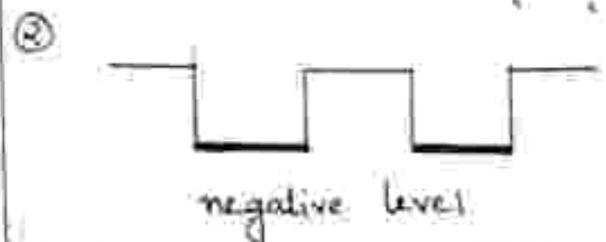
Positive level triggering

① When the flipflop changes the state during the positive level of clock is termed as positive level triggered.



Negative Level triggering

① When the flipflop changes the state during the negative level of clock is termed as negative level triggered

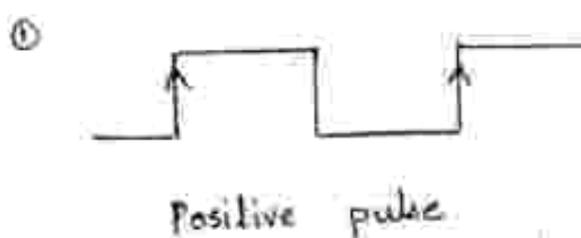


Edge Triggering

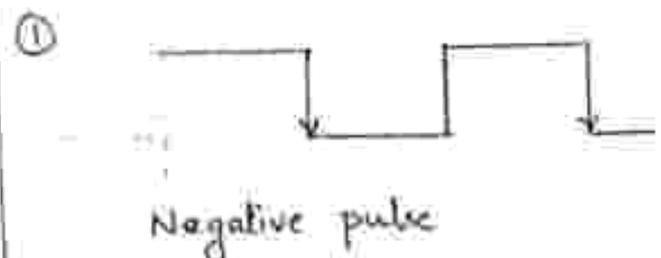
The term edge-triggering means, the flipflop changes state during the 0 to 1 transition or 1 to 0 transition at the edge. They are of 2 types.

- ① positive edge triggered
- ② negative edge triggered.

Positive edge triggered

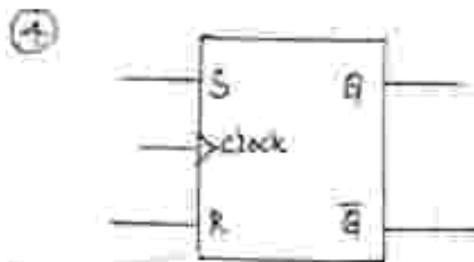


Negative edge triggered



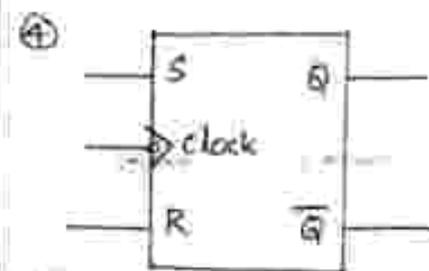
② Positive edge triggered flipflops are those in which state transition take place only at the positive going (0 to 1 or low to HIGH) edge of clock pulse.

③ Positive edge triggering is indicated by a triangle at the clock terminal.



② Negative edge triggered flipflops are those in which the state transitions take place only at the negative going (1 to 0 or HIGH to Low) edge of the clock signal.

③ Negative edge triggering is indicated by a triangle with a bubble at the clock signal.



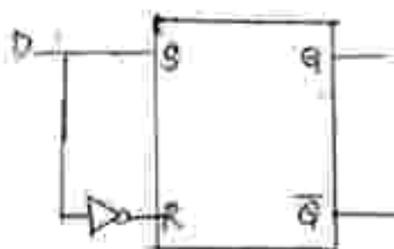
D Flipflop (Delay Flipflop)

The edge-triggered D flipflop has only one input terminal called the Delay (D) input and 2 outputs Q & \bar{Q} .

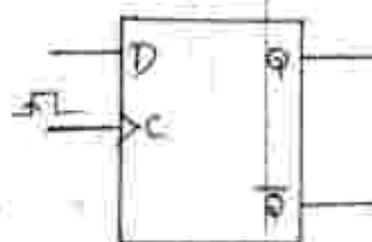
The D flipflop may be obtained from an SR flipflop by just passing an inverter between the S & R terminal.

The logic symbol and truth table of positive edge triggered

D flipflop is shown below.



(a) D flipflop from SR flipflop



(b) Logic symbol of D flipflop

C	D	Q _n	Q _{n+1}	State
↑	0	0	0	Reset
↑	0	1	0	Reset
↑	1	0	1	Set
↑	1	1	1	Set
0	X	0	0	No change
0	X	1	1	No change

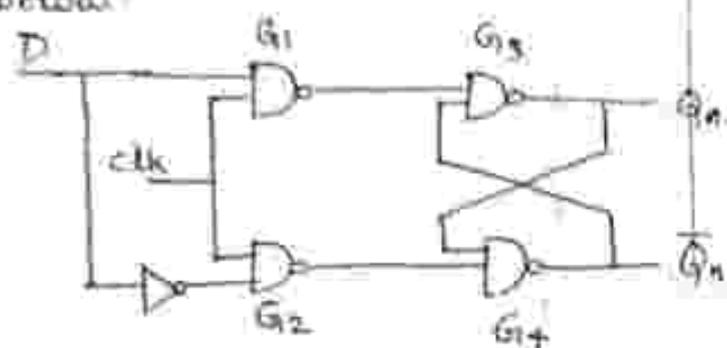
(c) Truth table

The flipflop has only one synchronous control input in addition to the clock input

The output \bar{Q} will go to the same state that is present on the D input at the positive going transition of the clock pulse

Working

The circuit diagram of edge-triggered D flipflop is shown below:



1) When the clock input is low, the D input has no effect and the output remains in no change condition

2) When the clock input goes high the Q output will take on the value of D input

Case I

If $clk = 1$ & $D = 1$

o/p of $G_1 \rightarrow 0$ o/p of $G_2 \rightarrow 1$

\Rightarrow Let flipflop is in SET state ($Q_n = 1, \bar{Q}_n = 0$)

o/p of $G_3 \rightarrow G_1 = 0, \bar{Q}_n = 0$ o/p of $G_3 = 1$ (i.e. $Q_{n+1} = 1$)

o/p of $G_4 \rightarrow G_2 = 1, Q_n = 1$ o/p of $G_4 = 0$ (i.e. $\bar{Q}_{n+1} = 0$)

\Rightarrow Flipflop in RESET state ($Q_n = 0, \bar{Q}_n = 1$)

o/p of $G_3 \rightarrow G_1 = 0, \bar{Q}_n = 1$ o/p of $G_3 = Q_{n+1} = 1$

o/p of $G_4 \rightarrow G_2 = 1, Q_n = 1$ o/p of $G_4 = \bar{Q}_{n+1} = 0$

Hence we can conclude that with $clk = 1, D = 1$ the output $Q_{n+1} = 1$ irrespective of present state

Case II

When $clk=1, D=0$

⇒ Flipflop is in SET state ($Q_{n+1}, \bar{Q}_n=0$)

o/p of $G_1 \rightarrow 1$, o/p of $G_2 \rightarrow 0$

o/p of $G_4 \rightarrow G_2=0, Q_n=1$, o/p of $G_4 = \bar{Q}_{n+1} = 1$

o/p of $G_3 \rightarrow G_1=1, \bar{Q}_n=0$, o/p of $G_3 = Q_{n+1} = 0$

⇒ Flipflop is in RESET state ($Q_n=0, \bar{Q}_n=1$)

o/p of $G_4 \rightarrow G_2=0, Q_n=0$, o/p of $G_4 = \bar{Q}_{n+1} = 1$

o/p of $G_3 \rightarrow G_1=1, \bar{Q}_n=0$, o/p of $G_3 = Q_{n+1} = 0$

ie with $clk=1, D=0$, the o/p of flipflop remains at $Q_{n+1}=0$, irrespective of present state

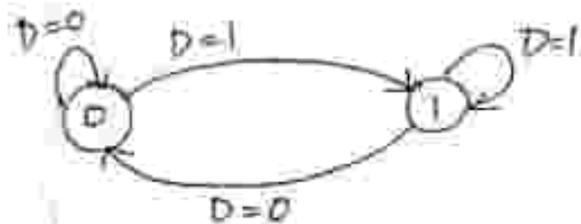
Truth table

clk	D	Q_{n+1}
1	0	0
1	1	1
0	x	No change

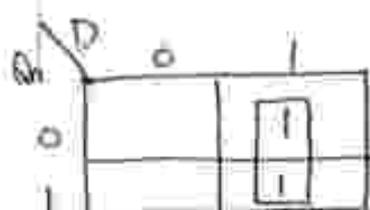
Characteristic Table

PS Q_n	NS Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

State diagrams



Characteristic equation



$$Q_{n+1} = D$$

Note

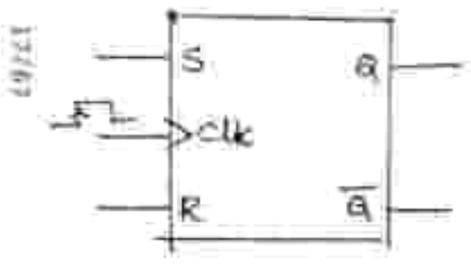
The digital systems can operate either synchronously or asynchronously.

In synchronous systems, the exact time at which any output can change states are determined by clock signals.

The flipflop using clock signals are called clocked flip-flops. Clocked flip-flops may be positive edge triggered or negative edge triggered.

The edge-triggered SR flip-flop

The logic symbol and truth table for a positive edge triggered SR flip-flop is shown below.

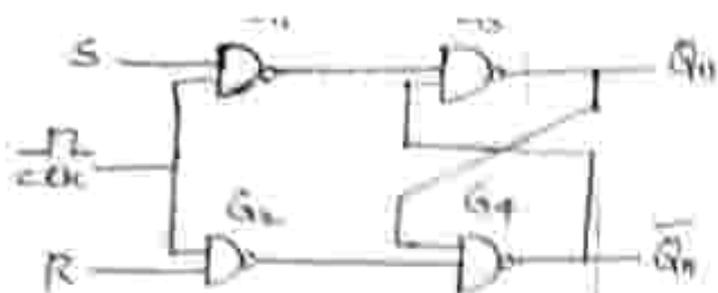


(a) logic symbol

C	S	R	Q _n	Q _{n+1}	State
↑	0	0	0	0	No change
↑	0	0	1	1	
↑	0	1	0	0	Reset
↑	0	1	1	0	
↑	1	0	0	1	set
↑	1	0	1	1	
↑	1	1	0	X	Indeterminate
↑	1	1	1	X	
∞	X	X	0	0	No change (NC)
0	X	X	1	1	

(b) Truth table

The truth table of a negative edge triggered SR flip-flop is same as that of a positive edge triggered SR flip-flop, except that the clock 'C' points downwards.



(c) circuit diagrams of edge triggered SR flipflop

Case I

⇒ Input condition $\text{clock} = 0, \underline{S=0, R=0}$

Flipflop remains in present state itself (PS)

⇒ Input condition $\text{clk} = 1, S=0, R=0$

→ FF in set condition ($Q_n = 1, \bar{Q}_n = 0$)

For $G_1 \Rightarrow$ I/p's are $S=0, \text{clk}=1$

O/p of $G_1 = 1$

For $G_2 \Rightarrow$ I/p's are $R=0, \text{clk}=1$

O/p of $G_2 = 1$

For $G_3 \Rightarrow$ I/p's are $G_1=1, \bar{Q}_n=0, \text{O/p of } G_3 = Q_{n+1} = 1$

For $G_4 \Rightarrow$ I/p's are $G_2=1, Q_n=1, \text{O/p of } G_4 = \bar{Q}_{n+1} = 0$

Flipflop in set condition

For $G_1 \Rightarrow$ I/p's are $S=0, \text{clk}=1, \text{O/p of } G_1 = 1$

For $G_2 \Rightarrow$ I/p's are $R=0, \text{clk}=1, \text{O/p of } G_2 = 1$

For $G_3 \Rightarrow$ I/p's are $G_1=1, \bar{Q}_n=1, \text{O/p of } G_3 = Q_{n+1} = 0$

For $G_4 \Rightarrow$ I/p's are $G_2=1, Q_n=0, \text{O/p of } G_4 = \bar{Q}_{n+1} = 1$

Flipflop in reset condition

Hence we can conclude that with $S=0, R=0, \text{clk}=1$. The flipflop remains in No change (NC) condition

Case II

$$S=0, R=1, \text{ clk}=1$$

⇒ FF in Set state ($Q_n=1, \bar{Q}_n=0$); Initial Condition

$$\text{o/p of } G_1 = 1$$

$$\text{o/p of } G_2 = 0$$

For gate G_4 ⇒ i/p's are $G_2=0, Q_n=1$; o/p of $G_4 = \bar{Q}_{n+1} = 1$

For gate G_3 ⇒ i/p's are $G_1=1, \bar{Q}_n=1$; o/p of $G_3 = Q_{n+1} = 0$

FF is in Reset Condition

⇒ FF in Reset state ($Q_n=0, \bar{Q}_n=1$)

$$\text{o/p of } G_1 = 1$$

$$\text{o/p of } G_2 = 0$$

For G_4 ⇒ i/p's are $G_2=0, Q_n=0$; o/p of $G_4 = \bar{Q}_{n+1} = 1$

For G_3 ⇒ i/p's are $G_1=1, \bar{Q}_n=1$; o/p of $G_3 = Q_{n+1} = 0$

FF is in set condition

⇒ Considering 2 condition we can conclude that with $S=0, R=1, \text{ clk}=1$, the b/f (flipflop) is at Reset condition

Case III

$$S=1, R=0, \text{ clk}=1$$

⇒ FF in set Initial Condition ($Q_n=1, \bar{Q}_n=0$)

$$\text{o/p of } G_1 = 0, \text{ o/p of } G_2 = 1$$

For G_3 ⇒ $G_1=0, \bar{Q}_n=0$; o/p of $G_3 = Q_{n+1} = 1$

For G_4 ⇒ $G_2=1, Q_n=1$; o/p of $G_4 = \bar{Q}_{n+1} = 0$

FF is in set condition

\Rightarrow FF is in Reset initial condition ($Q_n = 0, \bar{Q}_n = 1$)

For $G_3 \Rightarrow G_1 = 0, \bar{Q}_n = 1, \text{ o/p of } G_3 = Q_{n+1} = 1$

For $G_4 \Rightarrow G_2 = 1, Q_n = 1, \text{ o/p of } G_4 = \bar{Q}_{n+1} = 0$

FF is again in set condition

\Rightarrow Hence we can conclude with $S=1, R=0, \text{ clk}=1$ input condition the flipflop o/p sets

Case IV

$S=1, R=1, \text{ clk}=1$

\Rightarrow FF in set initial condition ($Q_n = 1, \bar{Q}_n = 0$)

O/p of $G_1 = 0$

O/p of $G_2 = 0$

For $G_3 \Rightarrow G_1 = 0, \bar{Q}_n = 0, \text{ o/p of } G_3 = Q_{n+1} = 1$

For $G_4 \Rightarrow G_2 = 0, Q_n = 1, \text{ o/p of } G_4 = \bar{Q}_{n+1} = 1$

Invalid state

\Rightarrow FF in Reset initial condition ($Q_n = 0, \bar{Q}_n = 1$)

For $G_3 \Rightarrow G_1 = 0, \bar{Q}_n = 0, \text{ o/p of } G_3 = Q_{n+1} = 1$

For $G_4 \Rightarrow G_2 = 0, Q_n = 1, \text{ o/p of } G_4 = \bar{Q}_{n+1} = 1$

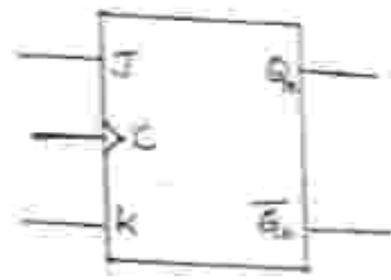
\rightarrow Hence we can conclude that $S=1, R=1, \text{ clk}=1$ input condition the flipflop output is indeterminate.

\Rightarrow The excitation table, state diagram & characteristic equation of SR flipflop is same as that of SR latch (NAND based)

Edge Triggered JK Flipflop

The characteristics of JK flipflop is similar to that of SR flipflop, except that it has no invalid state like that of SR flipflop.

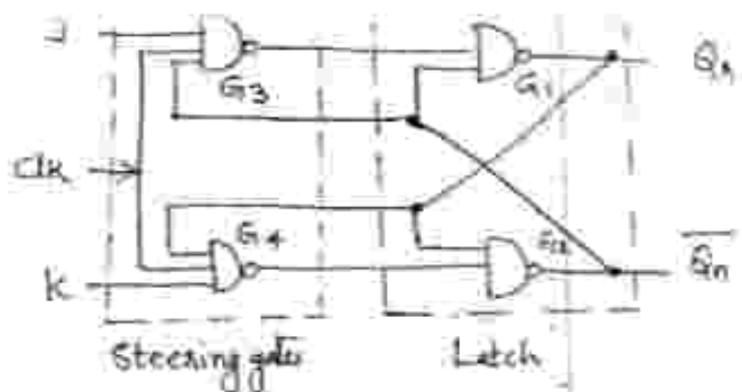
The logic symbol & truth table for positive edge triggered JK flipflop is shown below



(a) logic symbol

C	J	K	Qn	Qn+1	State
↑	0	0	0	0	No change (No)
↑	0	0	1	1	
↑	0	1	0	0	Reset
↑	0	1	1	0	
↑	1	0	0	1	Set
↑	1	0	1	1	
↑	1	1	0	1	Toggle
↑	1	1	1	0	
0	X	X	0	0	No change (No)
0	X	X	1	1	

(b) Truth table



(c) Simplified circuit diagram of the edge triggered JK flipflop

09/12

Working

Case I ($J=0, K=0, CLK=1$)

\Rightarrow FF is in set condition ($Q_n=1, \bar{Q}_n=0$)

For $G_3 \Rightarrow J=0, \bar{Q}_n=0, CLK=1$, o/p of $G_3=1$

For $G_4 \Rightarrow K=0, Q_n=1, CLK=1$, o/p of $G_4=1$

For $G_1 \Rightarrow G_3=1, \bar{Q}_n=0$, o/p of $G_1 = Q_{n+1} = 1$

For $G_2 \Rightarrow G_4=1, Q_n=1$, o/p of $G_2 = \bar{Q}_{n+1} = 0$

FF is in set condition.

\Rightarrow FF is in Reset condition ($Q_n=0, \bar{Q}_n=1$)

For $G_3 \rightarrow J=0, \bar{Q}_n=1, CLK=1$, o/p of $G_3=1$

For $G_4 \rightarrow K=0, Q_n=0, CLK=1$, o/p of $G_4=1$

For $G_1 \rightarrow G_3=1, \bar{Q}_n=1$, o/p of $G_1 = Q_{n+1} = 0$

For $G_2 \rightarrow G_4=1, Q_n=0$, o/p of $G_2 = \bar{Q}_{n+1} = 1$

FF is in Reset condition

\Rightarrow Hence we can conclude the Flipflop remains in present

state when $J=0, K=0$ input condition.

Case II ($J=0, K=1$)

\Rightarrow FF is in set condition ($Q_n=1, \bar{Q}_n=0$)

For Gate $G_3 \Rightarrow CLK=1, J=0, \bar{Q}_n=0$, o/p of $G_3=1$

For $G_4 \Rightarrow CLK=1, K=1, Q_n=1$; o/p of $G_4=0$

For $G_2 \Rightarrow G_4=0, Q_n=1$, o/p of $G_2 = \bar{Q}_{n+1} = 1$

For $G_1 \Rightarrow G_3=1, \bar{Q}_n=1$, o/p of $G_1 = Q_{n+1} = 0$

FF is in Reset condition ($Q_n=0, \bar{Q}_n=1$)

⇒ FF is in reset condition ($Q_n = 0, \bar{Q}_n = 1$)

For Gate $G_3 \Rightarrow \text{clk} = 1, J = 0, \bar{Q}_n = 1, \text{o/p of } G_3 = 1$

$G_4 \Rightarrow \text{clk} = 1, K = 1, Q_n = 0, \text{o/p of } G_4 = 1$

For Gate $G_1 \Rightarrow G_3 = 1, \bar{Q}_n = 1, \text{o/p of } G_1 = \bar{Q}_{n+1} = 0$

$G_2 \Rightarrow G_4 = 1, \bar{Q}_n = 0, \text{o/p of } G_2 = \overline{Q_{n+1}} = 1$

FF in reset condition

Hence the output of JK FF with $J=0, K=1$ input condition is 'RESET'.

Case III ($J=1, K=0$)

⇒ Present state of flipflop is in set condition ($Q_n = 1, \bar{Q}_n = 0$)

For Gate $G_3 \Rightarrow \text{clk} = 1, J = 1, \bar{Q}_n = 0, \text{o/p of } G_3 = 1$

For Gate $G_4 \Rightarrow \text{clk} = 1, K = 0, Q_n = 1, \text{o/p of } G_4 = 1$

For gate $G_1 \Rightarrow G_3 = 1, \bar{Q}_n = 0, \text{o/p of } G_1 = \bar{Q}_{n+1} = 1$

$G_2 \Rightarrow G_4 = 1, Q_n = 1, \text{o/p of } G_2 = \overline{Q_{n+1}} = 0$

FF in set condition

⇒ FF in Reset initial condition ($Q_n = 0, \bar{Q}_n = 1$)

For Gate $G_3 \Rightarrow \text{clk} = 1, J = 1, \bar{Q}_n = 1, \text{o/p of } G_3 = 0$

$G_4 \Rightarrow \text{clk} = 1, K = 0, Q_n = 0, \text{o/p of } G_4 = 1$

For Gate $G_1 \Rightarrow G_3 = 0, \bar{Q}_n = 1, \text{o/p of } G_1 = Q_{n+1} = 1$

Gate $G_2 \Rightarrow G_4 = 1, Q_n = 1, \text{o/p of } G_2 = \overline{Q_{n+1}} = 0$

∴ FF in set condition.

⇒ Hence we can conclude with $J=1, K=0$, input condition, the FF is in set condition.

Case IV ($J=1, K=1$)

⇒ FF is in set condition, ∴ $Q_n = 1, \bar{Q}_n = 0$

For Gate G3 \rightarrow $clk=1, J=1, \bar{Q}_n=0, o/p \text{ of } G3 = 1$
 $G4 \rightarrow clk=1, K=1, Q_n=1, o/p \text{ of } G4 = 0$
 For Gate G1 $\rightarrow G3=1, \bar{Q}_n=1, o/p \text{ of } G1 = Q_{n+1} = 0$
 $G2 \rightarrow G4=0, Q_n=1, o/p \text{ of } G2 = \bar{Q}_{n+1} = 1$

FF in reset condition.

\Rightarrow Reset present state for same input condition ($Q_n=0, \bar{Q}_n=1$)

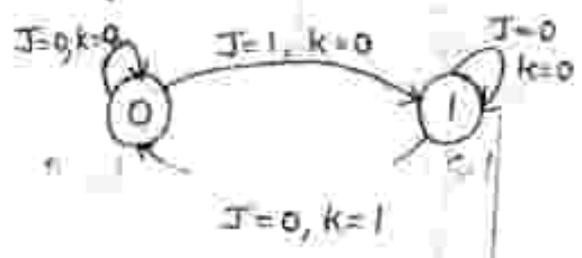
For Gate G3 $\rightarrow clk=1, J=1, \bar{Q}_n=1, o/p \text{ of } G3 = 0$
 $G4 \rightarrow clk=1, K=1, Q_n=0, o/p \text{ of } G4 = 1$

For Gate G2 $\rightarrow G4=1, Q_n=1, o/p \text{ of } G2 = \bar{Q}_{n+1} = 0$
 $G1 \rightarrow G3=0, \bar{Q}_n=1, o/p \text{ of } G1 = Q_{n+1} = 1$

FF in set condition.

\Rightarrow Here we can conclude that with $J=1, K=1$ input condition the output toggles corresponding to present state conditions.
 i.e. present state Q_n is set then Q_{n+1} reset.
 if Q_n is reset then Q_{n+1} set

State diagram



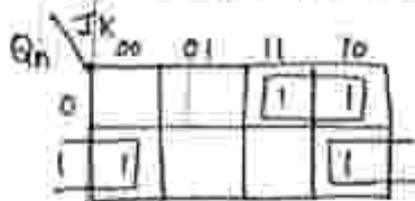
Truth table

J	K	Q_{n+1}	\bar{Q}_{n+1}
0	0	No change	
1	0	1	0
1	1	Toggle	

Excitation Table

Q_n	Q_{n+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Characteristic Equation



$Q_{n+1} = J\bar{Q}_n + \bar{K}Q_n$

Race Around Condition

For the JK FF discussed above, consider the excitations

$$J = K = 1$$

If we are applying a clock pulse of width ' t_p ' then the output will change from 0 to 1 after a time interval Δt .

Where $\Delta t \rightarrow$ propagation delay through 2 NAND gates in series.



Again if we have the condition $J = K = 1$ & $Q = 1$. After another time interval Δt , the output of Q will become 0. Hence the output will oscillate back & forth between 0 & 1 in the duration ' t_p ' of the clock pulse width.

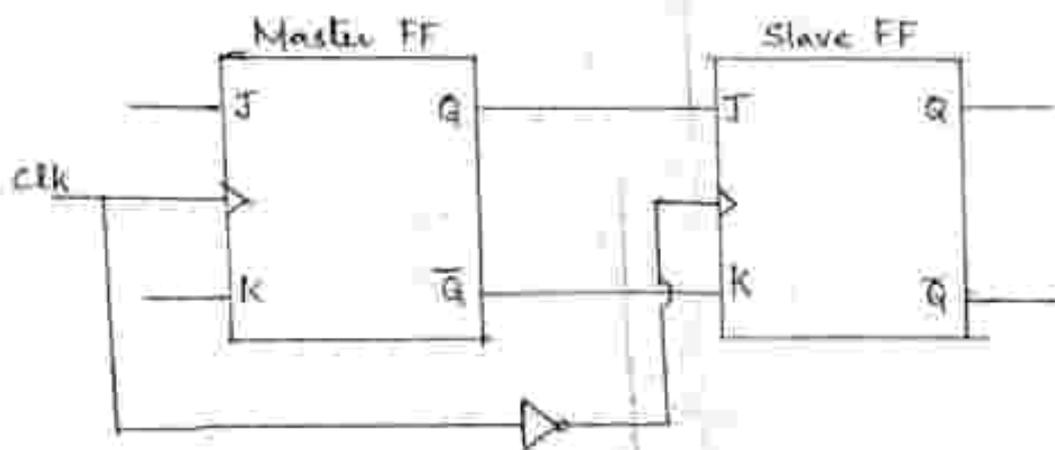
At the end of the clock pulse, the value of Q is ambiguous.

This situation is known as a race-around condition.

There are 2 methods to eliminate race around condition.

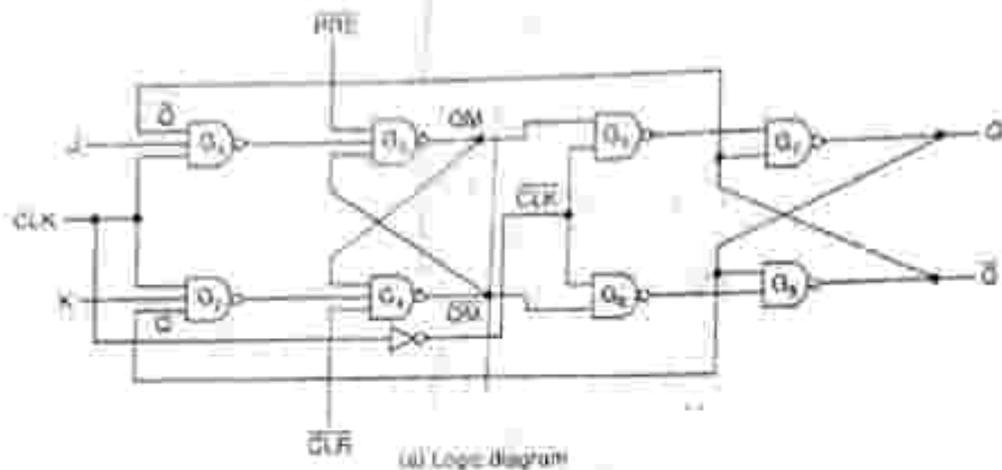
- 1) use of Master-slave FF
- 2) use edge triggering instead of level triggering.

Master Slave JK FF



For the first +ve cycle of clock, the master FF works and

and in -ve cycle, the slave FF works. So after 1 complete cycle we get an output. The logic diagram & truth table of master slave JK FF is shown below.



(a) Logic Diagram

Inputs			Output	Comments
J	K	C	Q	
0	0	1	Q ₀	No change
0	1	1	0	RESET
1	0	1	1	SET
1	1	1	Q ₀	Toggle

(b) Truth table

Figure 10.40 The master-slave J-K flip-flop.

EXAMPLE 10.6 The waveforms shown in Figure 10.41a are applied to the master-slave J-K flip-flop shown in Figure 10.41b. Draw the output waveforms.

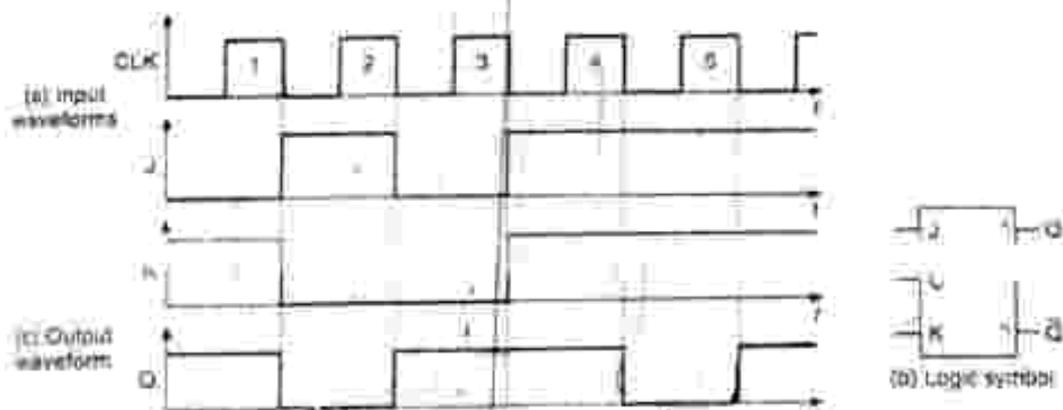


Figure 10.41 Example 10.6. Waveforms—master-slave J-K flip-flop.

Solution

The output waveform shown in Figure 10.41c is drawn as explained below. Initially, $J = 0$ and $K = 1$ and the flip-flop is assumed to be in SET state, i.e. $Q = 1$.

Working

The first flipflop called master is a JK Flipflop and the second flipflop called slave is an SR flip-flop.

The master is driven by the positive edge of the clock pulse and slave by the negative edge of the clock pulse. When the clock input has a positive edge, the master acts according to its JK inputs. But the slave does not respond since it requires a negative clock at the input. When the clock is negative, the slave flipflop copies the master outputs. But the master does not respond to the feedback from \bar{Q} , since it requires a positive edge at its clock input. Thus we get output for $J=1, K=1$ input combination only after a complete clock cycle. The problems of race around condition is thus eliminated.

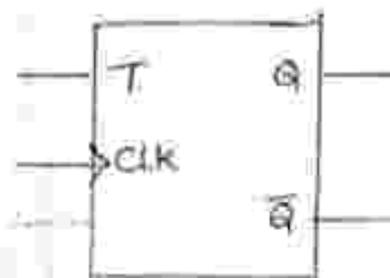
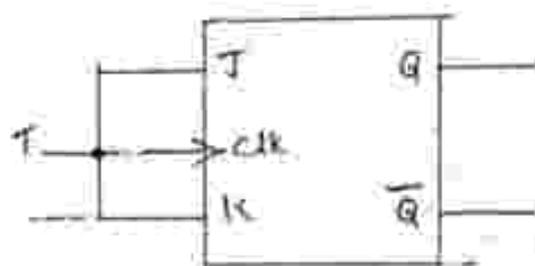
T Flipflop (Toggle Flipflop)

A T flipflop has a single control input labelled T for toggle.

When T is HIGH, the flipflop toggles on every new clock pulse.

When T is LOW, the flipflop remains in whatever state it was before.

A T flipflop is obtained from a JK flipflop by connecting its J & K inputs together and labelling the common connection as T.

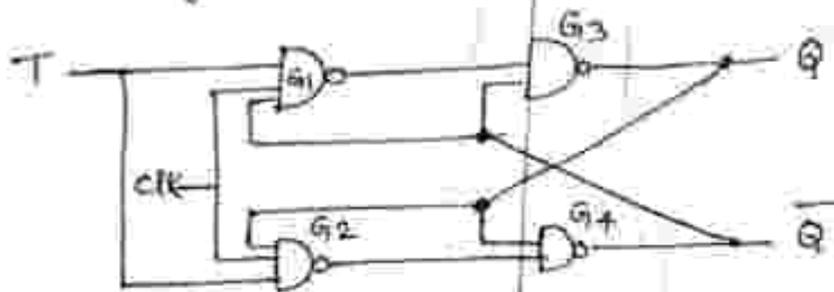


Truth table

T	Q_{n+1}
0	Q_n
1	$\overline{Q_n}$

C	T	Q_n	Q_{n+1}	state
↑	0	0	0	No change
↑	0	1	1	No change
↑	1	0	1	Toggle
↑	1	1	0	Toggle
0	X	0	0	No change
0	X	1	1	No change

Circuit Diagram



Working

Case I

When $T = 0$ ($J = K = 0$)

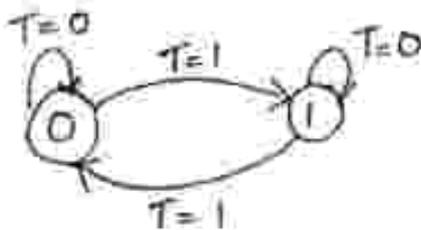
Q output will not change i.e. $Q_{n+1} = Q_n$

Case II

When $T = 1$ ($J = K = 1$) the output will be in toggle state

i.e. $Q_{n+1} = \overline{Q_n}$

state change



Excitation table

Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

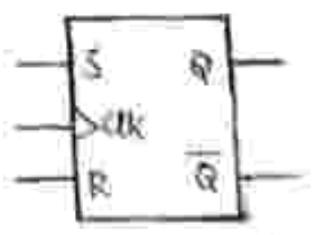
Characteristic Equation

	T=0	T=1
Q_n	0	1
0	0	1
1	1	0

$Q_{n+1} = T\overline{Q_n} + \overline{T}Q_n$

Summary

SR Flipflop



(a) logic symbol

(b) Truthtable

S	R	Q_{n+1}
0	0	Q_n (No)
0	1	0
1	0	1
1	1	? (Invalid)

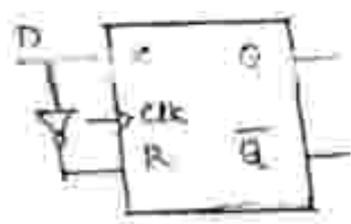
C	S	R	Q_n	Q_{n+1}
↑	0	0	0	0
↑	0	0	1	1
↑	0	1	0	0
↑	0	1	1	0
↑	1	0	0	1
↑	1	0	1	1
↑	1	1	0	X
↑	1	1	1	X

(c) Excitation Table

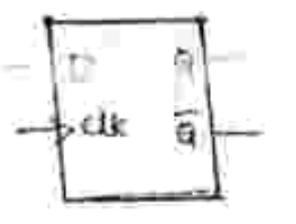
PS		NS		Required Inputs	
Q_n	Q_{n+1}	S	R	S	R
0	0	0	X	0	X
0	1	1	0	1	0
1	0	0	1	0	1
1	1	X	0	X	0

(d) Characteristic Equation of SR Flipflop: $Q_{n+1} = S\bar{R} + Q_n\bar{R}$
 $= \bar{R}(S + Q_n)$

D Flipflop



(a) logic symbol



(b) logic symbol

(c) Truthtable

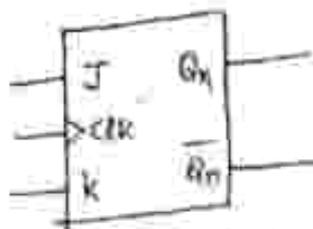
D	Q_{n+1}	C	D	Q_n	Q_{n+1}
0	0	↑	0	0	0
1	1	↑	0	1	0
		↑	1	0	1
		↑	1	1	1

(d) Excitation Table

PS	NS	D
Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

(e) Characteristic Equation of D Flipflop
 $Q_{n+1} = D$

JK Flipflop



(a) logic symbol

(b) Truth table

J	k	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q_n}$

C	J	k	Q_n	Q_{n+1}
↑	0	0	0	0
↑	0	0	1	1
↑	0	1	0	0
↑	0	1	1	0
↑	1	0	0	1
↑	1	0	1	1
↑	1	1	0	1
↑	1	1	1	0

(c) Excitation Table

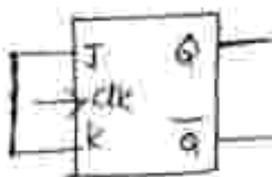
Q_n	Q_{n+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

(d) Characteristic Equation of JK flipflop

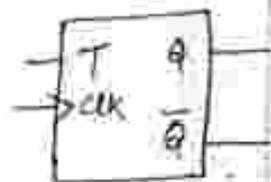
$$Q_{n+1} = J\overline{Q_n} + \overline{K}Q_n$$

30/67

T Flipflop



(a) logic symbol



(b) logic symbol

(c) Truth table

T	Q_{n+1}	C	T	Q_n	Q_{n+1}
0	Q_n	↑	0	0	0
0	Q_n	↑	0	1	1
1	$\overline{Q_n}$	↑	1	0	1
1	$\overline{Q_n}$	↑	1	1	0

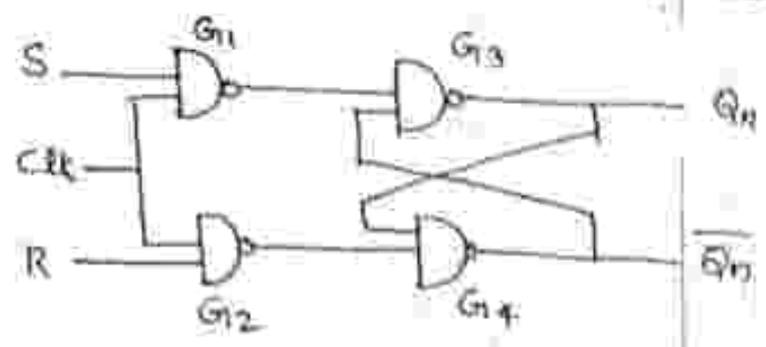
(d) Excitation Table

PS	NS	Required Inputs
Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

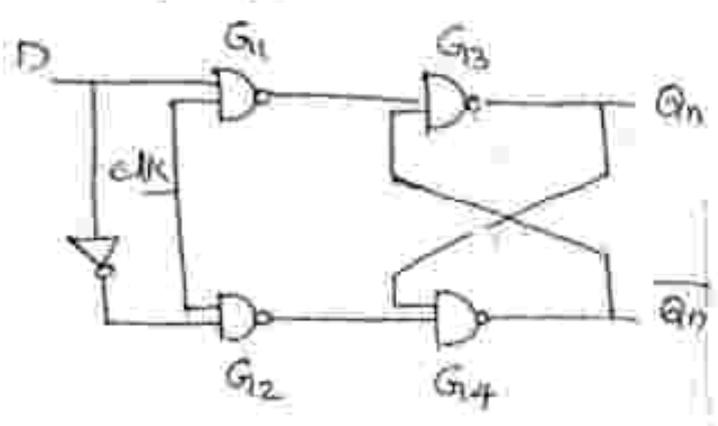
(e) Characteristic Equation of T flipflop is $Q_{n+1} = T\overline{Q_n} + \overline{T}Q_n$

Circuit Diagrams

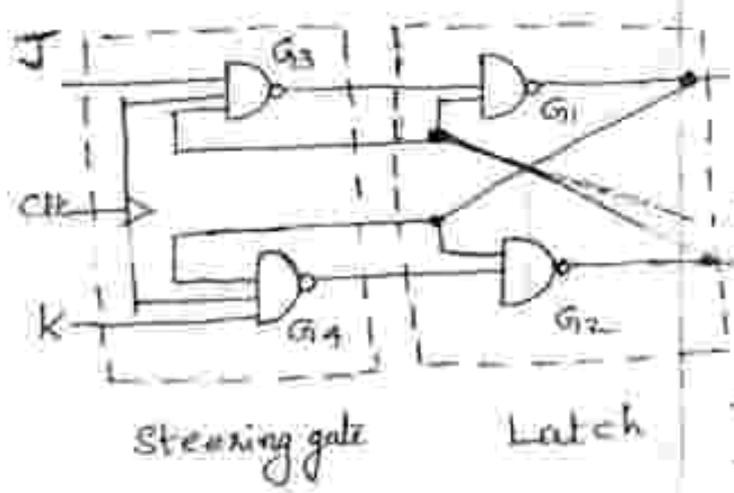
SR Flipflop (clocked SR latch)



D Flipflop

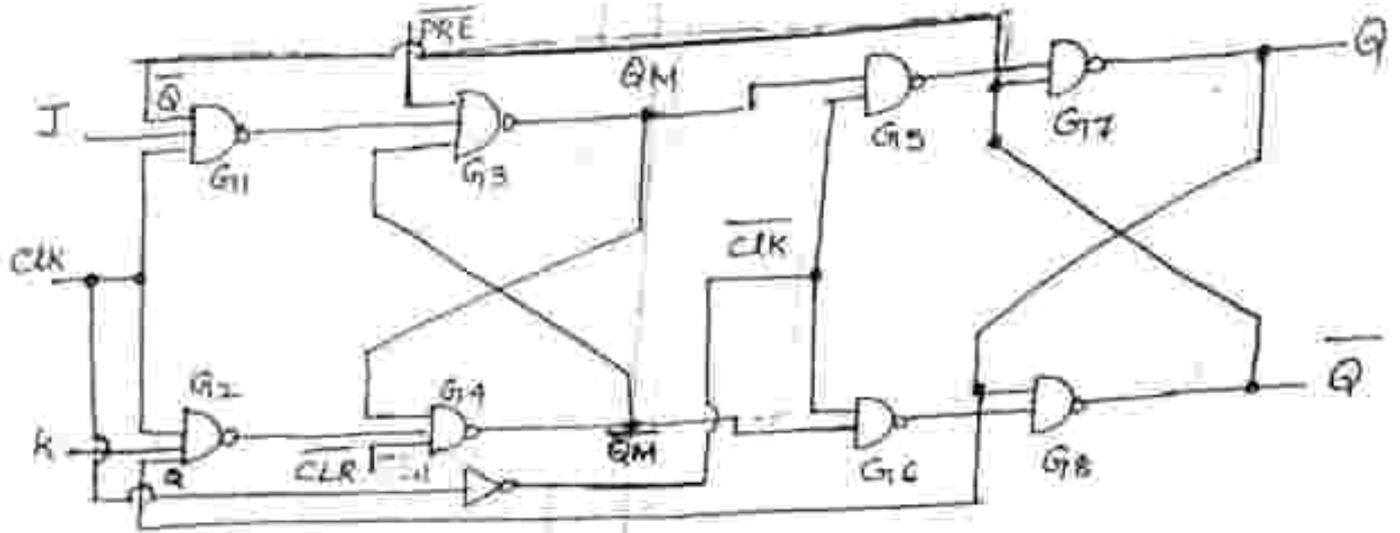


JK Flipflop

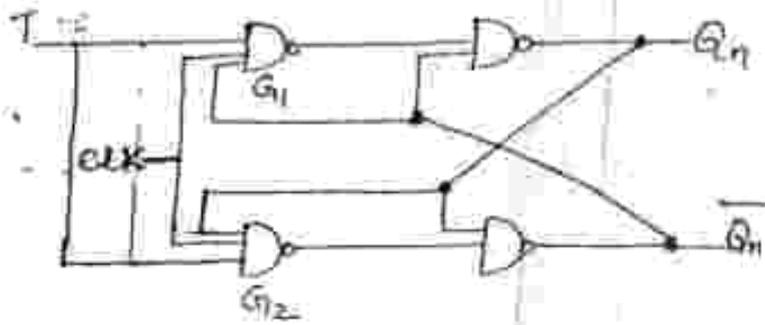


$J=1, K=1, Q_n=0$
 for $G_3, J=1, clk=1, \bar{Q}_n=1$ o/p of $G_3=0$
 for $G_4, J=1, clk=1, Q_n=0$ o/p of $G_4=1$
 for $G_1, G_3=0, \bar{Q}_n=1$ o/p of $G_1=1$ \bar{Q}_{n+1}
 for $G_2, G_4=0, Q_n=1$ o/p of $G_2=0$ Q_{n+1}
 $Q_n, J=1, K=1, Q_n=1$
 for $G_3, J=1, clk=1, \bar{Q}_n=0$ o/p of $G_3=1$
 for $G_4, J=1, clk=1, Q_n=1$ o/p of $G_4=0$
 for $G_1, G_3=1, \bar{Q}_n=0$ o/p of $G_1=0$ S_{n+1}
 for $G_2, G_4=0, Q_n=1$ o/p of $G_2=1$

Master Slave JK Flipflop

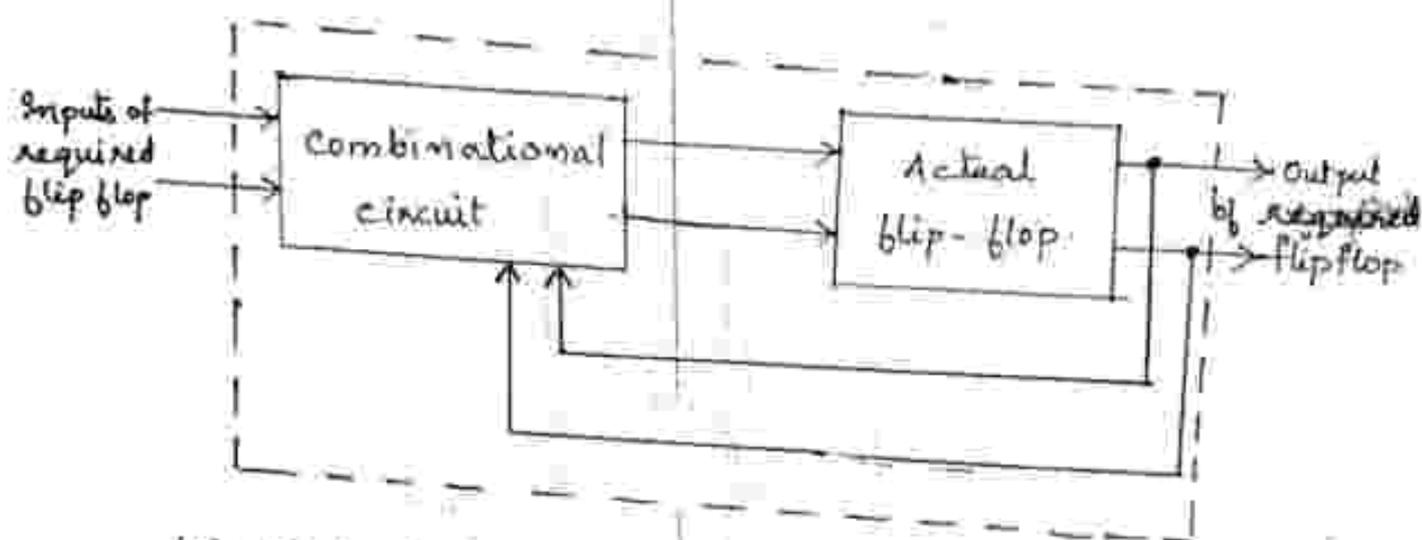


T Flipflop



Conversion Of Flip-Flops.

It means to convert one type of flip-flop to another type. For that we have to obtain the expressions for the inputs of existing flipflops in terms of inputs of required flipflop.



(a) Block diagram of conversion of flipflop.

Flipflop Conversion Includes the following steps

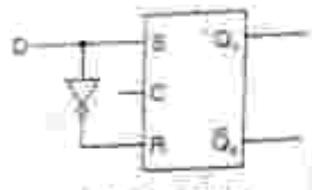
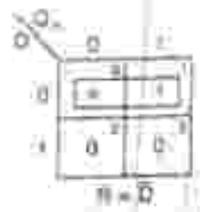
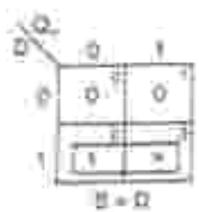
- 1) Write the excitation table of required flipflop, then the existing flipflop
- 2) Simplify the expression for inputs of existing flip-flop in terms of input of required flipflop & present state variables using K-Map.
- 3) Draw the desired logic diagram.

Realisation of D flipflop using SR flipflop

(SR flip-flop to D flipflop)

External inputs	Present State	Next State	Flip-flop inputs	
D	Q_n	Q_{n+1}	S	R
0	0	0	0	x
0	1	0	0	1
1	0	1	1	0
1	1	1	x	0

(a) Conversion table



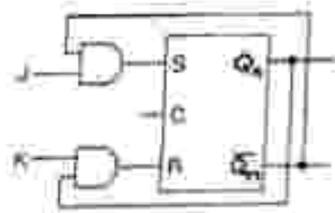
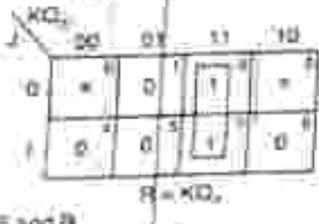
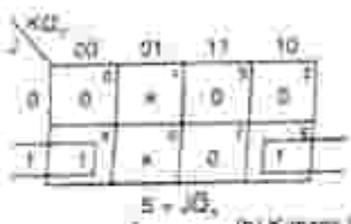
(b) K-maps for S and R
(c) Logic diagram
Figure 10.46 Conversion of S-R flip-flop to D flip-flop.

Realisation of JK flipflop using SR flipflop

(Conversion of SR FF to JK FF)

External inputs		Present State	Next State	Flip-flop inputs	
J	K	Q_n	Q_{n+1}	S	R
0	0	0	0	0	x
0	0	1	1	x	0
0	1	0	0	0	x
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	1	x	0
1	1	0	1	1	0
1	1	1	0	0	1

(a) Conversion table

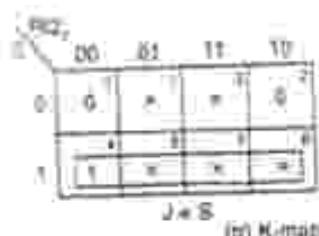


(b) K-maps for S and R
(c) Logic diagram
Figure 10.44 Conversion of S-R flip-flop to J-K flip-flop.

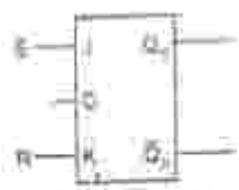
Conversion of JK flipflop to SR flipflop

External Inputs		Present State	Next State	Flip-Flop Inputs	
S	R	Q_n	Q_{n+1}	J	K
0	0	0	0	0	x
0	0	1	1	x	0
0	1	0	0	0	x
0	1	1	0	0	1
1	0	0	1	1	x
1	0	1	1	x	0

(a) Conversion table



(b) K-maps for J and K



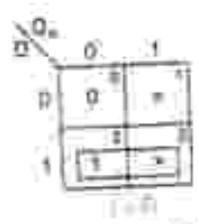
(c) Logic diagram

Figure 10.45 Conversion of JK flip-flop to SR flip-flop

Conversion of JK Flipflop To D Flipflop

External Input	Present State	Next State	Flip-Flop Inputs	
D	Q_n	Q_{n+1}	J	K
0	0	0	0	x
0	0	1	x	1
0	1	0	1	x
0	1	1	x	0

(a) Conversion table



(b) K-maps for J and K

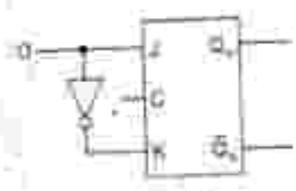


Figure 10.49 Conversion of JK flip-flop to D flip-flop

10/10

Conversion of JK flipflop To T flipflop

External Input	Present State	Next State	Flip-flop inputs	
			J	K
0	0	0	0	0
0	1	1	0	0
1	0	1	1	0
1	1	0	0	1

(a) Conversion table

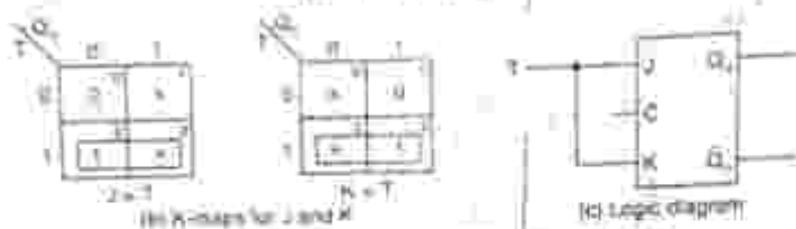


Figure 10.46 Conversion of JK flipflop to T flipflop

Conversion of D flipflop To SR flipflop

External Input		Present State	Next State	Flip-flop input
D	Q			
0	0	0	0	0
0	1	1	0	0
1	0	0	1	0
1	1	1	1	0

(a) Conversion table



Figure 10.47 Conversion of D flipflop to SR flipflop

Handwritten note: $D = S = R$

Conversion of D Flipflop To Jk flipflop

External Inputs		Present State	Next State	Flip-flop Input
J	K	Q_n	Q_{n+1}	D
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0

(A) Conversion table

Figure 10.50

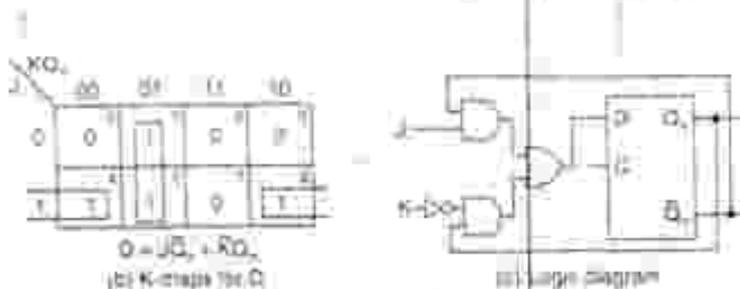


Figure 10.50. Conversion of D flip-flop to J-K flip-flop.

Asynchronous Inputs

The SR, D & JK inputs of the flipflops are called synchronous inputs, because their effect on the flipflop output is synchronized with the clock input.

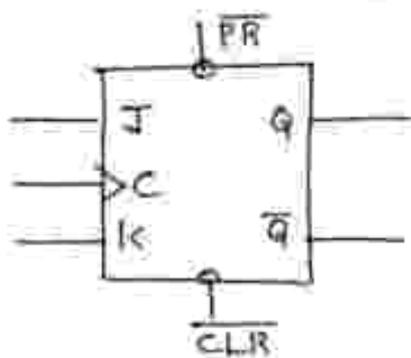
Most Flip-flops IC have one or more asynchronous inputs. These asynchronous inputs affect the flipflop output independently of the synchronous inputs & clock input. These inputs can be used to set the flipflop to 1 state or Reset it to 0 state at any time regardless of conditions of other inputs.

PRESET & CLEAR are the asynchronous inputs.

They are active low inputs.

An active low level at the PRESET input will SET

the FF and an active low level at the CLEAR input will RESET it. So for normal FF operation we set $\text{preset} = \text{clear} = 1$



COUNTERS

The digital counter is a sequential circuit that count clock pulses.

It is also called frequency divider.

It divides the clock frequency by its modulation.

The counter is composed of flipflops.

In counter circuit, the flipflop works as a toggle switch.

Classification

- 1) Asynchronous Counter
- 2) Synchronous Counter

Comparison Between Asynchronous Counter & Synchronous Counter

Table 12.1 Synchronous versus asynchronous counters

Asynchronous counters	Synchronous counters
1. In this type of counter FFs are connected in such a way that the output of first FF drives the clock for the second FF, the output of the second the clock of the third and so on.	1. In this type of counter there is no connection between the output of first FF and clock input of next FF and so on.
2. All the FFs are not clocked simultaneously.	2. All the FFs are clocked simultaneously.
3. Design and implementation is very simple even for more number of states.	3. Design and implementation becomes tedious and complex as the number of states increases.
4. Main drawback of these counters is their low speed as the clock is propagated through a number of FFs before it reaches the last FF.	4. Since clock is applied to all the FFs simultaneously the total propagation delay is equal to the propagation delay of only one FF. Hence they are faster.

Asynchronous Counter

In asynchronous counter, each ff is triggered by the output from previous flipflop.

It is also called ripple counter. Here JK Flipflops are connected in toggle mode (ie $J = K = 1$)

Asynchronous counters are made of T flipflops.

(ie JK flipflop connected in toggle mode ($J = K = 1$))

Mod N counter

Modulus:

The no: of states through which the counter passes before returning to the starting state is called modulus of the counter

Eg Mod N counter means, the counter counts from 0 to N-1. ie N states are there.

Eg Mod 4 counter \rightarrow counts from 0 to 3 (4 states)

Mod 8 counter \rightarrow counts from 0 to 7 (8 states)

Mod 16 counter \rightarrow counts from 0 to 15 (16 states)

Equation representing the no: of states and no: of flipflops used: $2^n = N$

where $n \Rightarrow$ no: of flipflops used

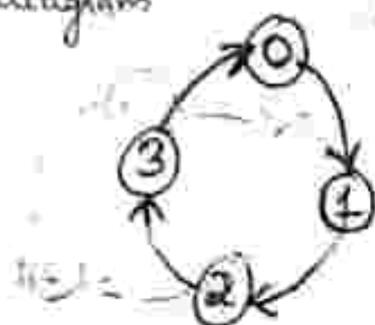
$N \Rightarrow$ modulus of counter

Eg: Mod 8 counter $\rightarrow 2^3 = 8, n = 3 \rightarrow 3$ flipflops used

Mod 16 counter $\rightarrow 2^4 = 16, N = 16, n = 4 \rightarrow 4$ flipflops used

Up Counter (2 bit)

- 2 bit ripple / asynchronous counter
- Mod 4 counter
- Counting sequence is $0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 0 \rightarrow 1 \rightarrow 2 \dots$
- State diagram



- Here $2^n = 2^2 = 4$, $n=2$ ∴ 2 flips can be used.
- It can be implemented using T or JK Flipflop.

2 bit up counter using negative edge triggered flip flop

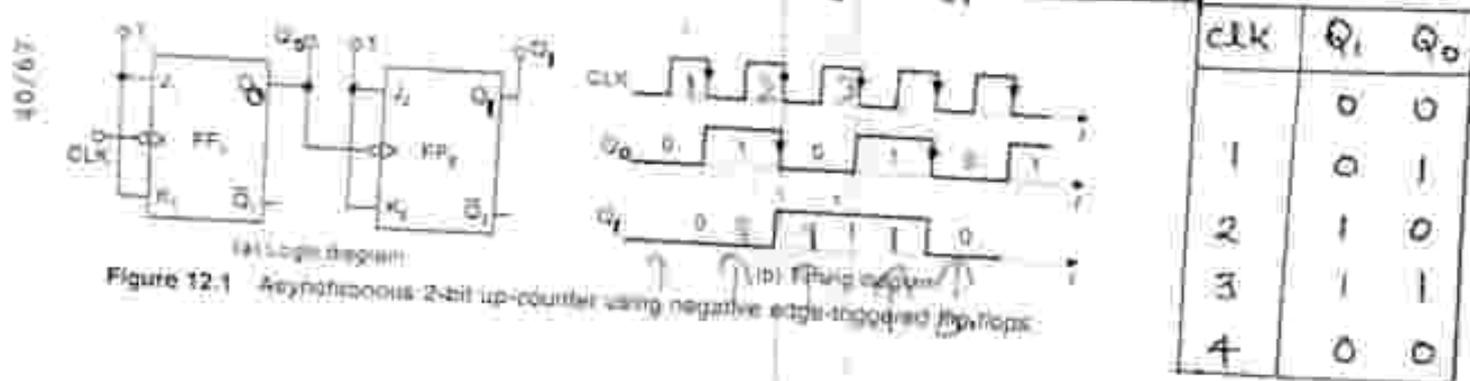


Figure 12.1 Asynchronous 2-bit up-counter using negative edge-triggered flip-flops.

- The counter initially reset to $Q_1, Q_0 = '00'$
- When the 1st clock pulse is applied FF1 toggles at the -ve going edge of the clock pulse. Then Q_0 changes from low (zero) to High (One) state.
- The o/p of FF1 i.e. Q_0 is the clock input to FF2, $Q_0=1$, which is positive (means +ve edge clock).
- Since Q_0 is positive FF2 not respond.

So after one clock pulse $Q_1 Q_0 = 01$

→ During the next -ve going edge of the clock pulse FF1 toggles and Q_0 change from '1' to '0' (High to low)

& this negative going signal activate FF2. So

Q_1 output of FF2 changes from '0' to '1' (Low to High)

i.e. after second clock pulse $Q_1 Q_0 = 10$

→ During the next -ve going edge of the clock pulse Q_0 changes from '0' to '1' & Q_1 is the, FF2 not respond $Q_1 = 1$

i.e. after third clock pulse $Q_1 Q_0 = 11$

→ During the fourth -ve going edge of the clock pulse Q_0 change from '1' to '0' & Q_1 is -ve, FF2 toggles and output of FF1 Q_1 changes from '1' to '0'.

i.e. after fourth clock pulse $Q_1 Q_0 = 00$

Two-bit Ripple Up-Counter Using Positive Edge-triggered Flip-flops.

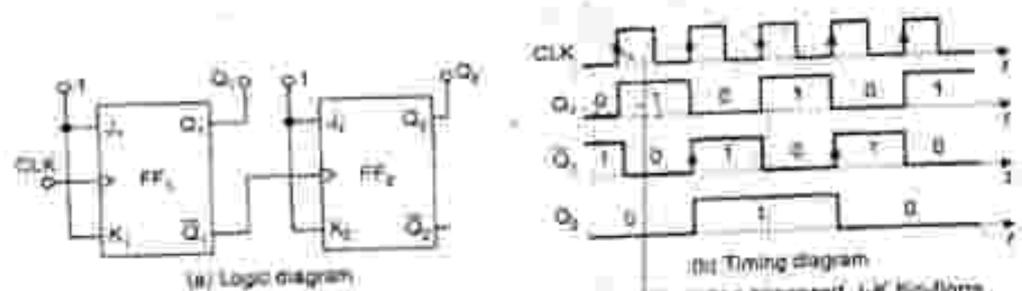


Figure 12.4 Asynchronous 2-bit up-counter using positive-edge triggered J-K flip-flops.

The \bar{Q}_1 output of the first FF is connected to the clock of FF2.

The external clock signal is applied to the first flip-flop FF1.

→ Initially both flip-flops are in reset state $Q_1 Q_2 = 00$

→ The FF1 toggles at the positive-going edge of each clock pulse & output of FF1 that is $Q_1 = 1$. \bar{Q}_1 is connected to FF2. FF2 does not toggle (FF2 toggles at the +ve edge of the clock, here $\bar{Q}_1 = 0$). ∴ Q_2 remains in reset state.

After 1st clock pulse $Q_1 Q_2 = 10$

→ During the 2nd clock pulse FF1 toggles & output $Q_1 = 0$

$\bar{Q}_1 = 1$ ∴ FF2 also toggles and output $Q_2 = 1$.

After 2nd clock pulse $Q_1 Q_2 = 01$

→ During the 3rd clock pulse FF1 toggles & output $Q_1 = 1$

$\bar{Q}_1 = 0$ ∴ FF2 does not toggle & output $Q_2 = 1$ (previous o/p)

After 3rd clock pulse $Q_1 Q_2 = 11$

→ During the 4th clock pulse FF1 toggles & output $Q_1 = 0$

$\bar{Q}_1 = 1$. FF2 also toggles and output $Q_2 = 0$

After 4th clock pulse $Q_1 Q_2 = 00$

Two bit Ripple Down Counter Using Negative Edge-triggered Flip-flops.

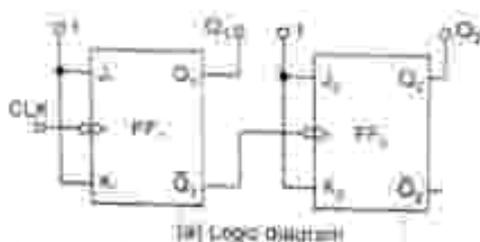


Figure 12.2. Asynchronous 2-bit down-counter using negative edge-triggered flip-flops.

For down counting, \bar{Q}_1 of FF1 is connected to the clock of FF2.

→ Let initially all the FFs be reset i.e. let the count be 00.

→ At the negative going edge of the first clock pulse FF1 toggles so, Q_1 goes from a 0 to a 1 and \bar{Q}_1 goes from a 1 to a 0.

This negative-going signal at \bar{Q}_1 applied to the clock input of FF2 toggles FF2 & therefore Q_2 goes from a 0 to 1.

After one clock pulse $Q_2 = 1$ & $Q_1 = 1$ i.e. $Q_2 Q_1 = 11$

→ During the negative edge of the 2nd clock pulse Q_1 changes from a 1 to 0 and \bar{Q}_1 from a 0 to 1. This positive going signal at \bar{Q}_1 does not affect FF2 and therefore Q_2 remains at a 1.

After 2nd clock pulse $Q_2 Q_1 = 10$

→ At the negative edge of the 3rd clock pulse FF1 toggles, Q_1 goes from 0 to 1 and \bar{Q}_1 from a 1 to 0. This negative going signal at \bar{Q}_1 toggles FF2 and Q_2 changes from a 1 to 0.

After 3rd clock pulse $Q_2 Q_1 = 01$

→ At the negative edge of the 4th clock pulse FF1 toggles, Q_1 goes from 1 to 0 and \bar{Q}_1 from a 0 to 1. This positive going signal at \bar{Q}_1 does not affect FF2 and Q_2 remains at a 0.

43/07

After 4th clock pulse $Q_2 Q_1$ is 00.
 → For subsequent clock pulses the counter goes through the same sequence of states i.e. the counter counts in the order 00, 11, 10, 01, 00 & 11

Two bit Ripple Down-Counter Using Positive Edge-triggered Flip-Flops



Figure 12.5 Asynchronous 2-bit down-counter using positive edge-triggered J-K flip-flops.

In the 2 bit ripple down counter Q_1 output of the first flipflop is connected to the clock of FF2. The external clock signal is applied to FF1. The FF1 toggles at the positive-going edge of each clock pulse. Initially FF1 & FF2 in reset state $Q_2 Q_1 = 00$.

→ At the +ve edge of first clock pulse

FF1 toggles & $Q_1 = 1$, FF2 toggles & $Q_2 = 1$

After 1st clock cycle $Q_2 Q_1 = 11$

→ At the +ve edge of 2nd clock pulse

FF1 toggles & $Q_1 = 0$, FF2 do not toggles & $Q_2 = 1$

After 2nd clock cycle $Q_2 Q_1 = 10$

→ At the +ve edge of 3rd clock pulse

FF1 toggles & $Q_1 = 1$, FF2 toggles & $Q_2 = 0$

After 3rd clock cycle $Q_2 Q_1 = 01$

→ At the +ve edge of 4th clock pulse

FF1 toggles & $Q_1 = 0$, FF2 do not toggles & $Q_2 = 0$

After 4th clock cycle $Q_2 Q_1 = 00$

At the +ve edge of 5th clock cycle

FF1 toggles & $Q_1 = 1$, FF2 toggles & $Q_2 = 1$

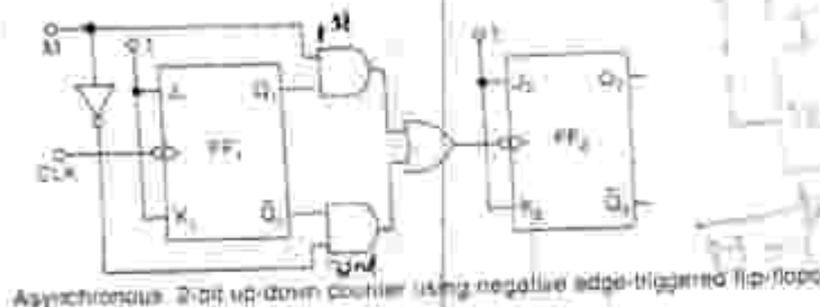
After 5th clock pulse $Q_2 Q_1 = 11$

→ The FF2 toggles whenever Q_1 changes from a 0 to a 1

The counting sequence is 00, 11, 10, 01, 00, 11... etc.

Two bit up-down counter using negative edge triggered flip-flops

(Modulus 4 up-down counter)



→ An up-down counter is also called a forward/backward counter or a bidirectional counter.

→ An up-down counter is a counter which can count both in upward & downward direction.

→ A control signal or a mode signal M is required to choose the direction of count. When $M=1$ for up counting Q_1 is transmitted to clock of FF2 and when $M=0$ for down counting \bar{Q}_1 is transmitted to clock of FF2.

→ Clock signal to FF2 = $(Q_1 \cdot \text{Up}) + (\bar{Q}_1 \cdot \text{Down}) = Q_1 M + \bar{Q}_1 \bar{M}$

$M=1$ Upcounting [o/p of 2nd AND gate remains zero ($\bar{M}=0$ connected to 2nd gate)]

→ FF1 & FF2 initially in reset state $Q_2 Q_1 = 00$

→ 1st clock cycle's -ve edge FF1 toggles $Q_1 = 1$, $M=1$, o/p of OR gate is one that is CLK to FF2 is +ve & FF2 do not toggle, $Q_2 = 0$

After 1st clock cycle $Q_2 Q_1 = 01$

→ 2nd clock cycle (-ve edge)

FF1 toggles, $Q_1 = 0$ & $M = 1$, o/p of OR gate is zero

clk to FF2 is -ve, FF2 toggles & $Q_2 = 1$

After 2nd clock cycle $Q_2 Q_1 = 10$

→ At -ve edge of 3rd clock cycle

FF1 toggles, $Q_1 = 1$ & $M = 1$, o/p of OR gate is one

clk to FF2 is +ve, FF2 do not toggles & $Q_2 = 1$

After 3rd clock cycle $Q_2 Q_1 = 11$

→ During up counting, counting sequence is 00, 01, 10, 11, 00, 01, ...

M=0 Downcounting

→ FF1 & FF2 initially in reset state $Q_2 Q_1 = 00$

→ At the -ve edge of 1st clock cycle

FF1 toggles, $Q_1 = 1$, $M = 0$ & $\bar{Q}_1 = 0$, $\bar{M} = 1$, o/p of 2nd AND gate is one & o/p of OR gate is one
clk to FF2 is -ve, FF2 toggles & $Q_2 = 1$

After 1st clock cycle $Q_2 Q_1 = 11$

→ At the -ve edge of 2nd clock cycle

FF1 toggles, $Q_1 = 0$, $M = 0$, $\bar{Q}_1 = 1$, $\bar{M} = 1$, o/p of OR gate is one

clk to FF2 is +ve, FF2 do not toggles & $Q_2 = 1$

After 2nd clock cycle $Q_2 Q_1 = 10$

→ At the -ve edge of 3rd clock cycle

FF1 toggles, $Q_1 = 1$, $M = 0$ 1st AND gate o/p is zero

$\bar{Q}_1 = 0$, $\bar{M} = 1$ 2nd AND gate o/p is zero

Therefore OR gate o/p is zero

clk to FF2 is -ve, FF2 toggles & $Q_2 = 0$

After the 3rd clock cycle $Q_2 Q_1 = 01$

At the -ve edge of 4th clock cycle

FF1 toggles, $Q_1 = 0, M = 0$ 1st AND gate o/p is zero

$\bar{Q}_1 = 1, M = 1$ 2nd AND gate o/p is one

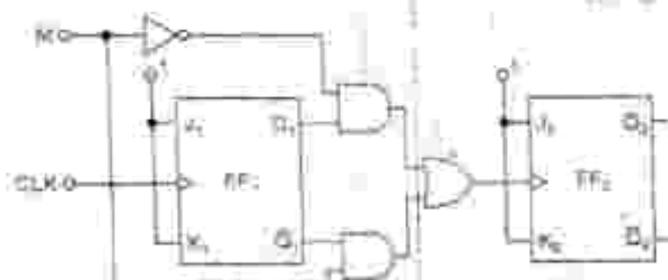
OR gate o/p is one. clk to FF2 is +ve, FF2 do not toggle and Q_2 remains at 0

After 4th cycle cycle $Q_2 Q_1 = 00$

→ During down counting, counting sequence is 00, 01, 10, 01, 00, 11, ...

Two bit Up-down Counter using positive edgedged flip-flops

(Modulus 4 Up-down Counter)

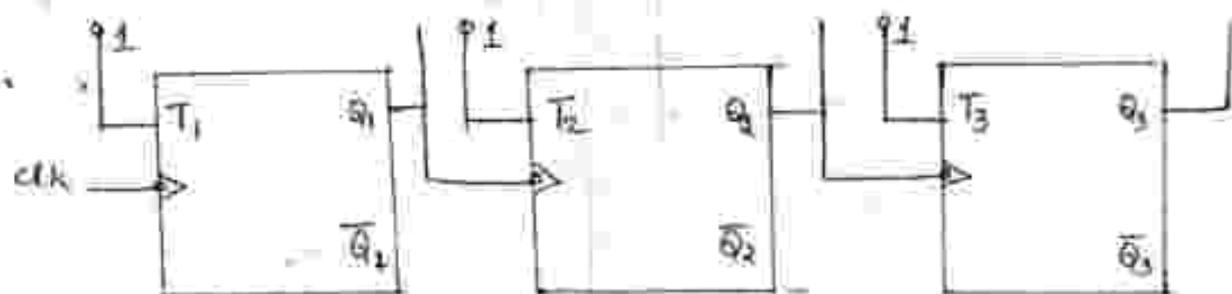


Logic diagram of a two-bit up/down counter using positive edge-triggered flip-flops.

When $M=1$ for up counting, \bar{Q}_1 is transmitted to the clock of FF2. and when $M=0$ for down counting, Q_1 is transmitted to the clock of FF2.

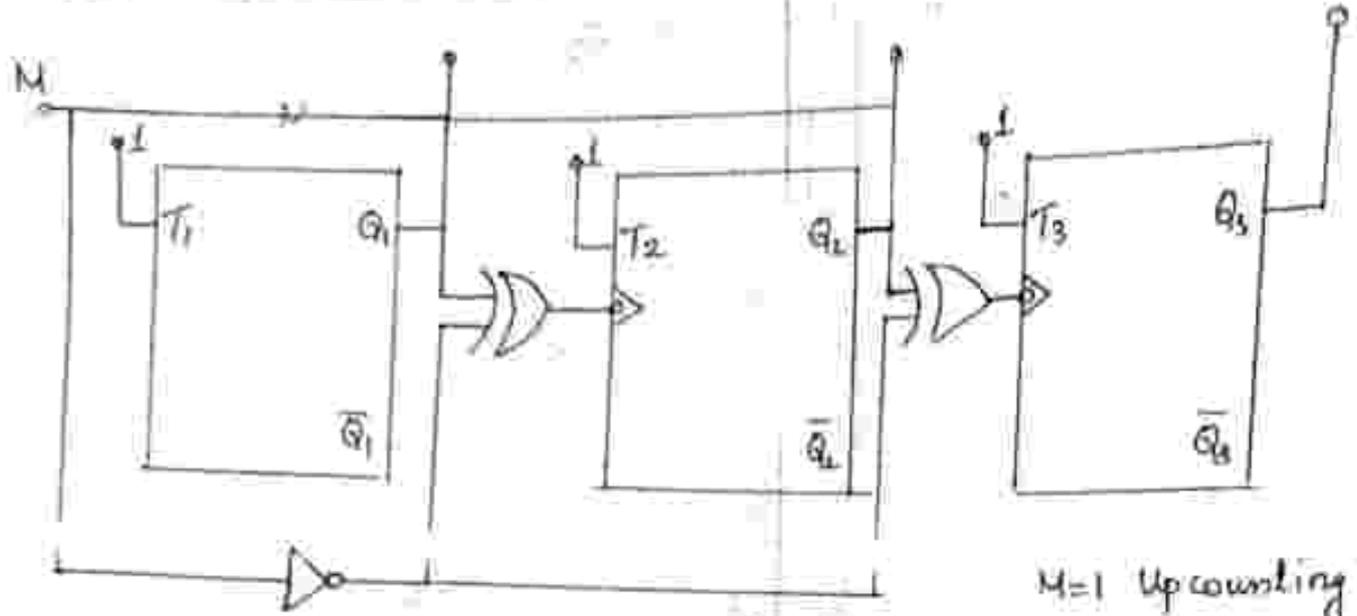
clock signal to FF2 = $(\bar{Q}_1 \cdot \text{Up}) + (Q_1 \cdot \text{Down}) = \bar{Q}_1 M + Q_1 \bar{M}$

3 Bit Up-Counter (Modulus 8 Counter)



Counting sequence is 000, 001, 010, 011, 100, 101, 110, 111, 000, ...

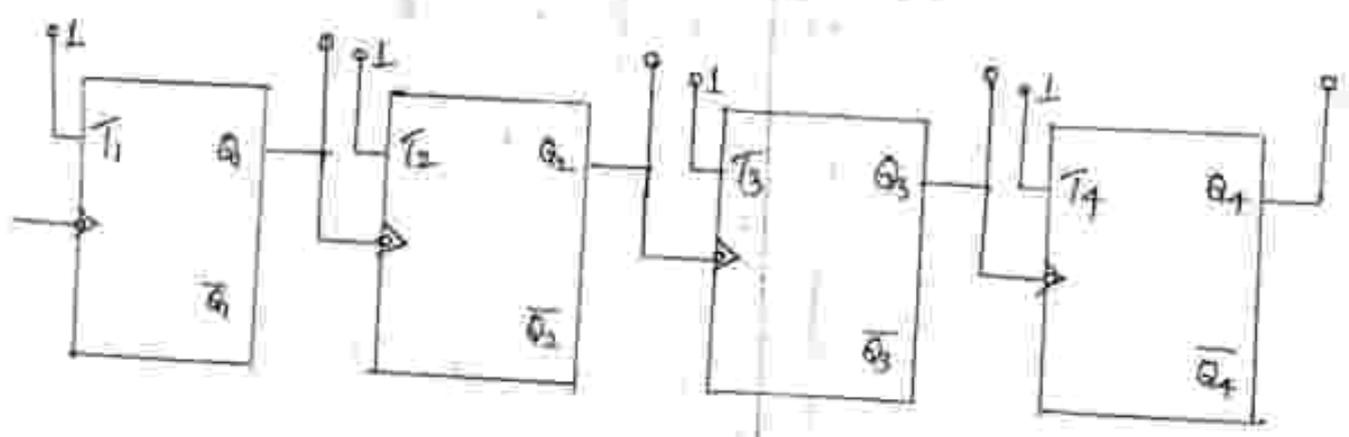
3 bit Up-down Counter (Mod 8 Up/Down Counter)



$M=1$ Upcounting
 000, 001, 010, 011, 100,
 101, 110, 111, 000...
 $M=0$ downcounting
 000, 111, 110, 101, 100,
 011, 010, 001, 000, 111...



4 Bit Up Counter (Mod 16 Counter)



Design Of Asynchronous Counter

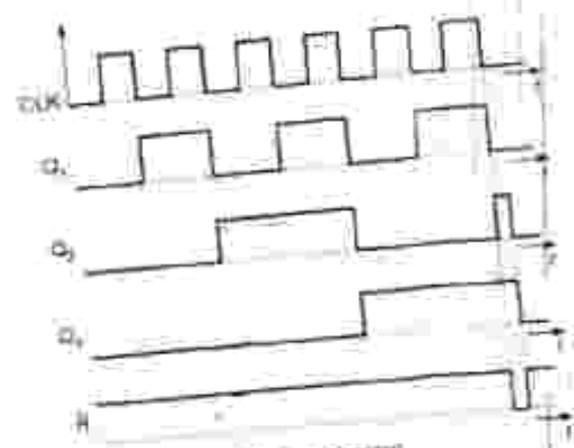
To design an asynchronous counter, first write the counting sequence, then tabulate the values of reset signal R for various states of the counter & obtain the minimal expression for R or \bar{R} . Provide a feedback such that R or \bar{R} resets all the FFs after the desired count.

Design of a Mod-6 Asynchronous Counter Using TFFs.

A mod 6 counter has six stable states 000, 001, 010, 011, 100 & 101. When the 6th clock pulse is applied, the counter temporarily goes to 110 state, but immediately resets to 000 because of the feedback provided. It is a divide by-6 counter, in the sense that it divides the input clock frequency by 6. For the design, write a truth table

With the present state outputs Q_3, Q_2 & Q_1 as the variables and reset R as the output and obtain an expression for R in terms of Q_3, Q_2 & Q_1 . That decides the feed back to be provided. From the truth table $R = Q_3 Q_2$.

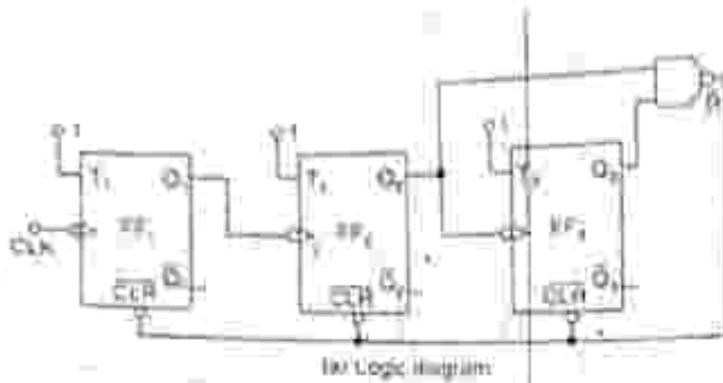
For active low reset \bar{R} is used.



After pulses	State			R
	Q_3	Q_2	Q_1	
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	0
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1

(a) Timing diagram
Figure 12.7. Asynchronous mod-6 counter using TFFs.

43/67



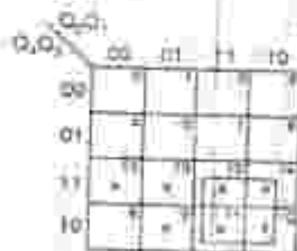
Design of a Mod-10 Asynchronous Counter Using TFF's

A mod-10 counter is a decade counter. It is also called BCD counter or divide-by-10 counter. It requires 4 FF's. So there are 16 possible states, out of which ten are valid & the remaining six are invalid. The counter has ten stable states, 0000 through 1001 i.e. it counts from 0 to 9. The initial state is 0000 and after nine clock pulses it goes to 1001. When the tenth clock pulse is applied, the counter goes to state 1010 temporarily, but because of the feedback provided, it resets to initial state 0000.

- 50/67
- $R=1$ for the state 1010
 - $R=0$ for 0000 to 1001
 - $R=X$ (don't care for 1011 to 1111)

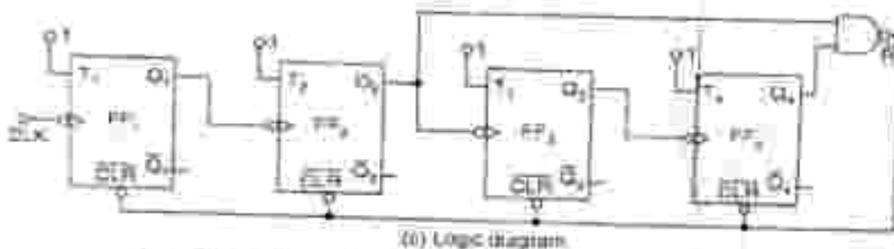
After pulses	Count			
	Q_3	Q_2	Q_1	Q_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	0	0	0	0

(a) Count table



(b) K-Map

From k map $R = Q_3 Q_2$
 feedback provided from
 2nd & 4th FFs
 For active low reset
 $\overline{R}(\text{CLR})$ is used.



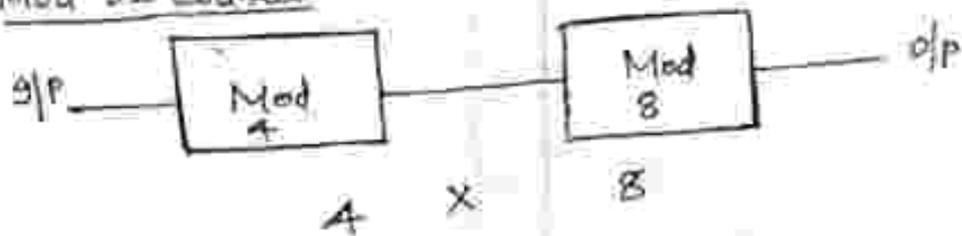
(c) Logic diagram

Figure 12.8. Asynchronous mod-10 counter using T flip-flops

Cascading Of Counters

When we are cascading Mod M & Mod N counters, it will become 'MN' counter.

Eg: Mod 32 Counter



∴ fin: Mod 32

Synchronous Counters

Synchronous counters are counters in which all the flipflops are triggered simultaneously by the clock input pulses. Since all the FFs change simultaneously, in synchronization with the clock pulse, the propagation delay of FF is less than that of asynchronous counters.

Advantages

- 1) High speed
- 2) Less decoding problems

Disadvantage

- 1) Having more circuitry

Design Of Synchronous Counter

For systematic design of synchronous counters, the following procedure is used.

- 1) Find the no: of FF's required
- 2) Draw the state diagrams showing all possible states.
- 3) Select the FF & write the excitation table.
- 4) Using K-map, obtain the minimal expression for the excitations.

5) Draw the logic diagrams based on minimal expressions. Note

Q1) Design a mod-4 synchronous up counter using JK Flipflop

(2 bit Counter)

Step I

No: of state $N = 2^n$

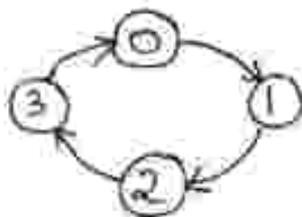
No: of FF's required (n)

$n = 2$ that is 2 flipflops are required

No: of state, $N = 2^2 = 4$

Step II

State diagram



32/67

Step III

Excitation Table of JK flipflop

Q_n	Q_{n+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Step IV

Excitation Table

PS		NS		J_B	K_B	J_A	K_A
Q_B	Q_A	Q_{B+1}	Q_{A+1}				
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

Note

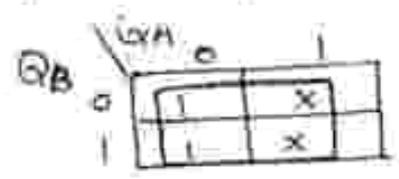
Find Q_B to Q_{B+1} transition & refer excitation table to find $J_B K_B$. Similarly Q_A to Q_{A+1} for $J_A K_A$.

Step V

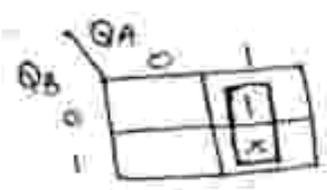
K Map Simplification

Present state variables are entered in k-map. Solve for

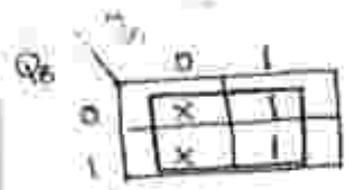
J_A, K_A & J_B, K_B



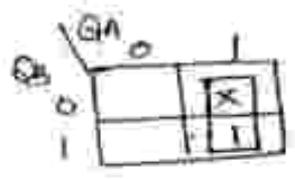
$J_A = 1$



$J_B = Q_A$



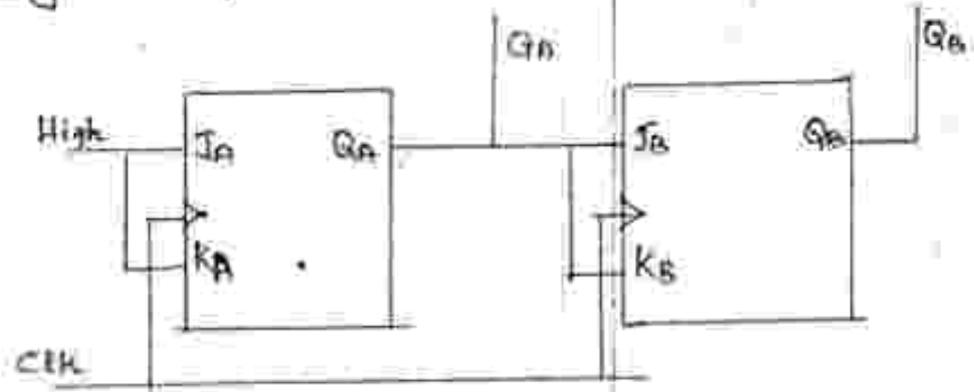
$K_A = 1$



$K_B = Q_A$

Step VI

Logic diagram



Q2) Design a mod 4 down counter using JK flipflop

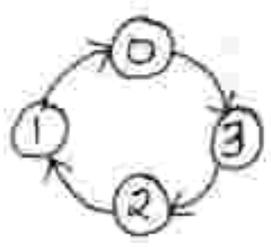
Step I

No. of states = 4

No. of flipflop's required, $N = 2^n$, $4 = 2^2$ i.e. $n = 2$

Step II

State diagram



Step III

Excitation Table of JK Flipflop

Q_n	Q_{n+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Step IV

Excitation Table

PS		NS		J_B K_B		J_A K_A	
Q_B	Q_A	Q_{B+1}	Q_{A+1}				
0	0	1	1	1	X	1	X
0	1	0	0	0	X	X	1
1	0	0	1	X	1	1	X
1	1	1	0	X	0	X	1

Handwritten Karnaugh maps and logic equations for the JK flipflop design.

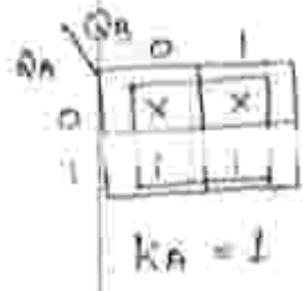
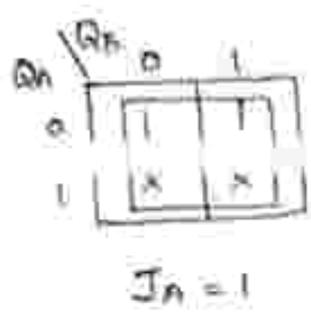
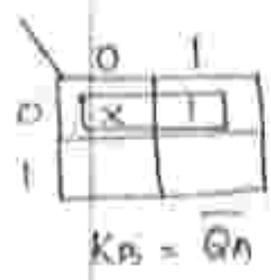
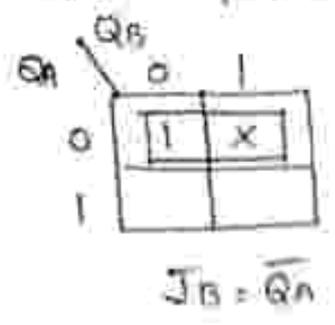
JK Flipflop Equations:

- $J_B = 1$
- $K_B = 1$
- $J_A = 1$
- $K_A = 1$

The Karnaugh maps show the derivation of these equations from the excitation table. The J_B and K_B maps are both 1s, and the J_A and K_A maps are also 1s.

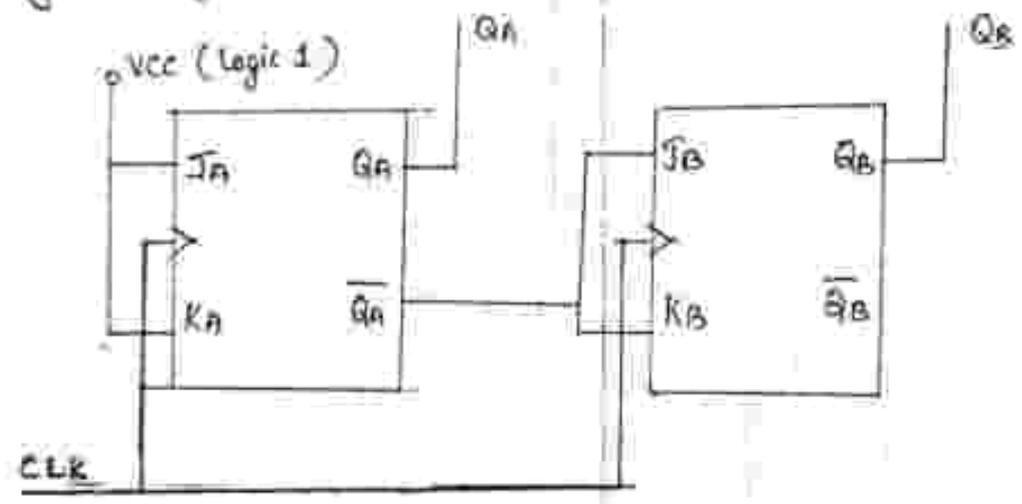
Step V

K Map Simplification



Step VI

Logic Diagram

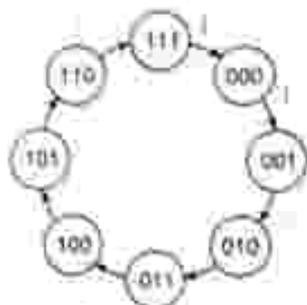


Q3) Design a mod 8 Up Counter using JK FF

Step I

No. of states = $N = 8$

No. of flipflops required $n = 3$ $[N = 2^n \text{ i.e. } 8 = 2^3]$



(a) State diagram

PS			NS			Required excitations					
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	0	0	1	0	x	0	x	1	x
0	0	1	0	1	0	0	x	1	x	x	1
0	1	0	0	1	1	0	x	x	0	1	x
0	1	1	1	0	0	1	x	x	1	x	1
1	0	0	1	0	1	x	0	0	x	1	x
1	0	1	1	1	0	x	0	1	x	x	1
1	1	0	1	1	1	x	0	x	0	1	x
1	1	1	0	0	0	x	1	x	1	x	1

(b) Excitation table

Figure 12.16 A 3-bit up-counter

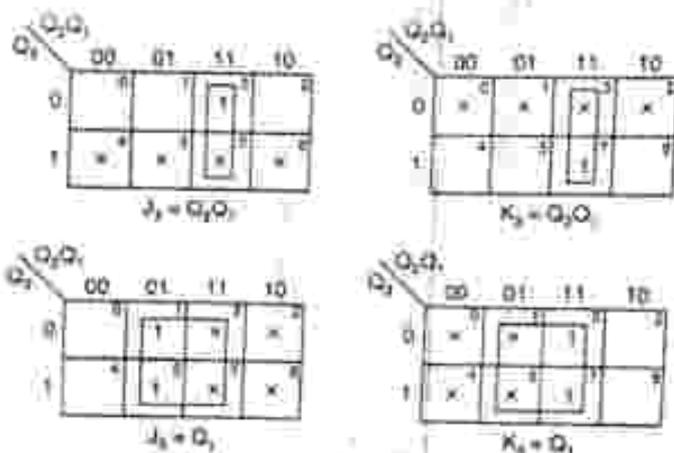
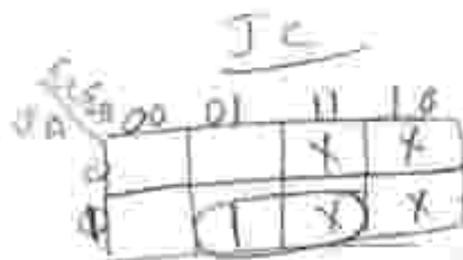


Figure 12.17 Karnaugh maps for a 3-bit up-counter

56/67



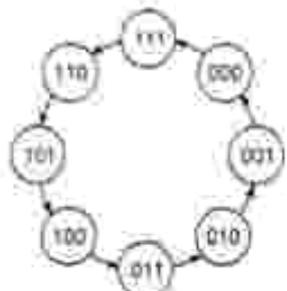
Design a mod 8 Synchronous down Counter Using JK Flip-flop

3 bit Synchronous down Counter

Step I

No: of states, $N = 8$

No: of flipflops required, $n = 3$ $[N = 2^n = 8 = 2^3]$



(a) State diagram

PS			NS			Required excitations					
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	1	1	1	1	x	1	x	1	x
1	1	1	1	1	0	x	0	x	0	x	1
1	1	0	1	0	1	x	0	x	1	1	x
1	0	1	1	0	0	x	0	0	x	x	1
1	0	0	0	1	1	x	1	1	x	1	x
0	1	1	0	1	0	0	x	x	0	x	1
0	1	0	0	0	1	0	x	x	1	1	x
0	0	1	0	0	0	0	x	0	x	x	1

(b) Excitation table

Figure 12.18. A 3-bit down-counter.

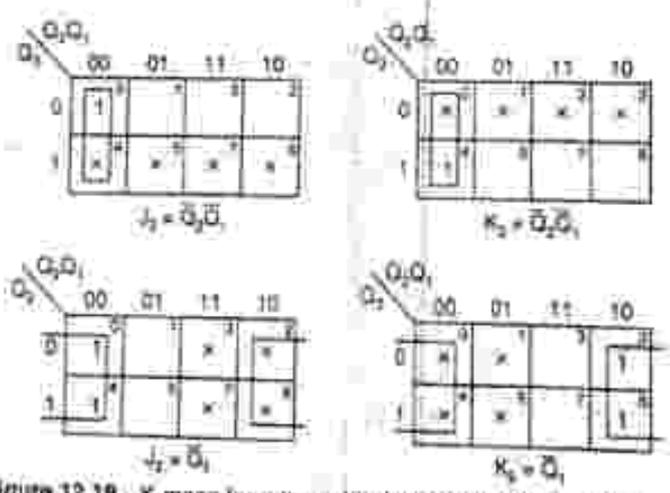


Figure 12.19 K-maps for a three-bit down counter using J-K FFs.

Q5) Design of a Synchronous Mod-6 Counter using JK Flipflop

Step 1

Mod 6 counter, counting sequence is 000, 001, 010, 011, 100, 101, 000. It has six states. So it requires

$$n = 3 \text{ FFs}$$

3 FFs can have eight states. So the remaining two states

110 & 111 are invalid.

Step 2: State diagram

Step 3: The type of FF's & the excitation table

Step 4: Minimal expressions using K Map

Step 5: Logic diagram

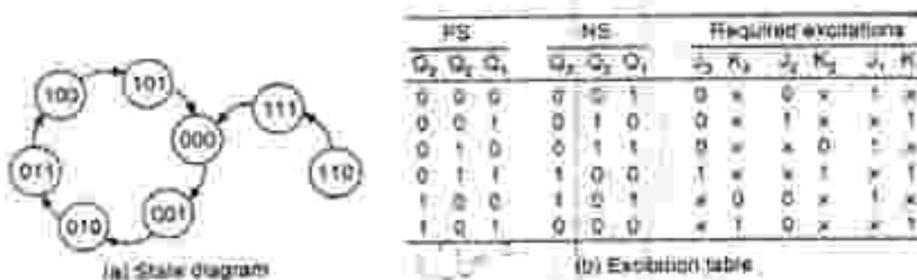


Figure 12.36 Synchronous mod-6 counter using J-K flip-flops.

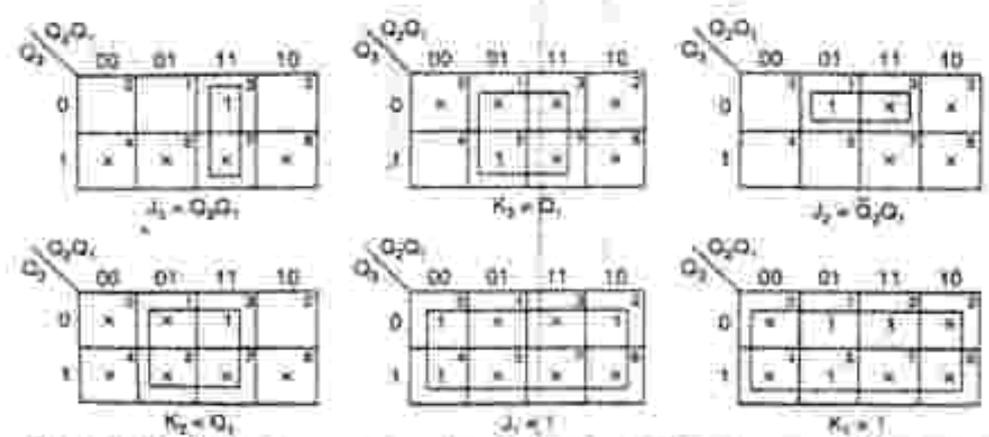


Figure 12.37 K-maps for excitations of synchronous mod-6 counter using J-K flip-flops.

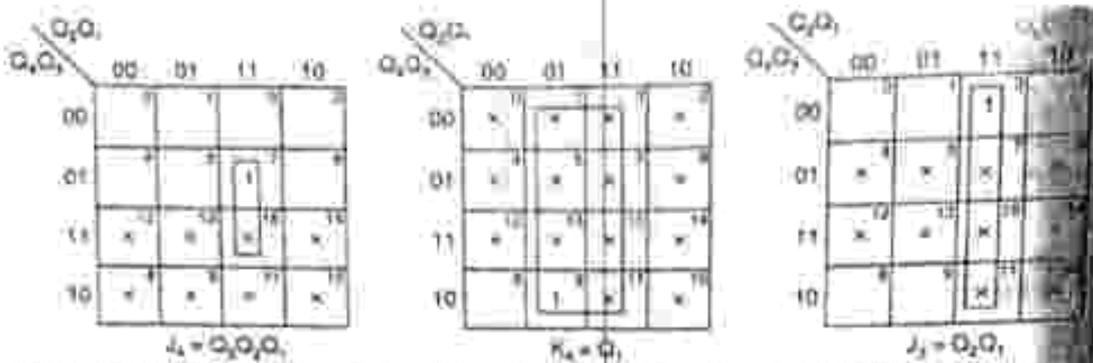


Figure 12.33 K-maps for excitations of synchronous BCD counter using J-K flip-flops (Contd.)

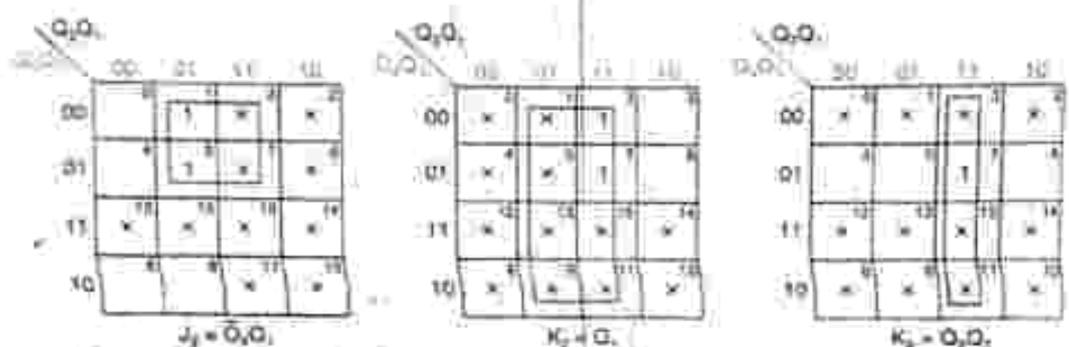


Figure 12.33 K-maps for excitations of synchronous BCD counter using J-K flip-flops.

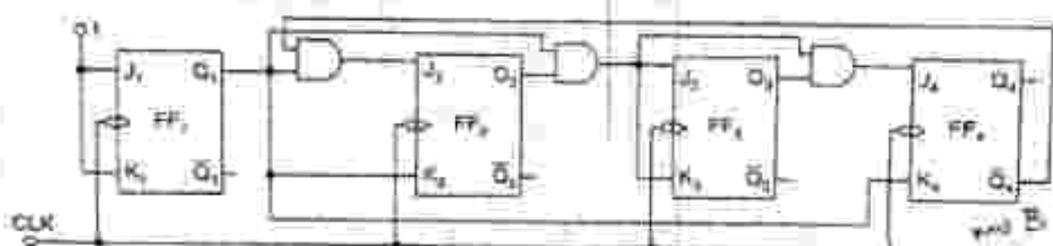


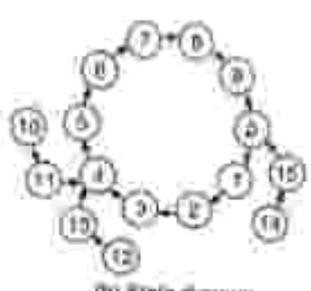
Figure 12.34 Logic diagram of synchronous BCD counter using J-K flip-flops.

Lock Out Condition

In a counter, if the next state of some unused state is again an unused state, then it never arrives at a used state & the circuit is said to be locked.

To avoid lockout condition, take the NS of an unused state as 0000 (initial count) or to another count.

PS				Present state								NS			
Q ₃	Q ₂	Q ₁	Q ₀	J ₁	K ₁	J ₂	K ₂	J ₃	K ₃	J ₄	K ₄	Q ₃	Q ₂	Q ₁	Q ₀
1	0	1	0	0	0	0	0	0	0	1	1	1	0	1	1
1	0	1	1	0	1	1	1	0	1	1	1	0	1	0	0
1	1	0	0	0	0	0	0	0	0	1	1	1	1	0	1
1	1	0	1	0	1	0	0	0	1	1	1	0	1	0	0
1	1	1	0	0	0	0	0	0	0	1	1	1	1	1	1
1	1	1	1	1	1	1	1	0	1	1	1	0	0	0	0



(a) Table to check for lockout
 (b) State diagram
 Figure 12.35 Checking for lock out of Synchronous BCD counter using J-K flip-flops.

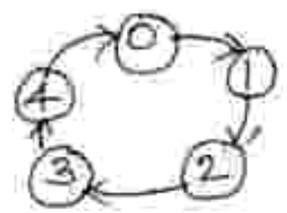
Design a mod 5 synchronous up counter using DFF without lockout condition.

Step 1

No. of state $\rightarrow 5$
 No. of FF's $\rightarrow 2^3, n=3$

Step 2

state diagram



Step 3

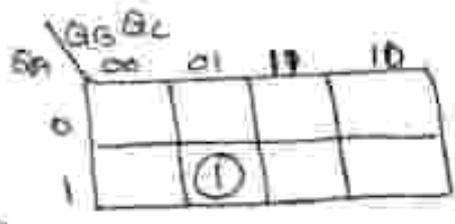
Excitation table of DFF

Q _n	Q _{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

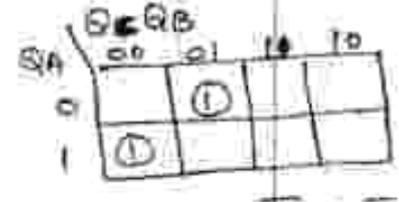
Excitation Table & Type of flipflop

PS			NS			Dc	D _B	D _A
Q _c	Q _B	Q _A	Q _c (t+1)	Q _B (t+1)	Q _A (t+1)			
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	0
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	0	0
1	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0

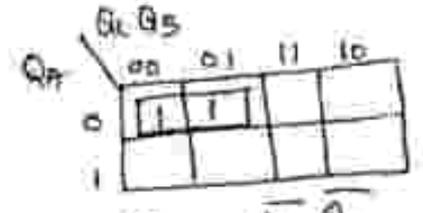
Step 4: K Map simplification



$$D_c = \bar{Q}_c Q_B Q_A$$

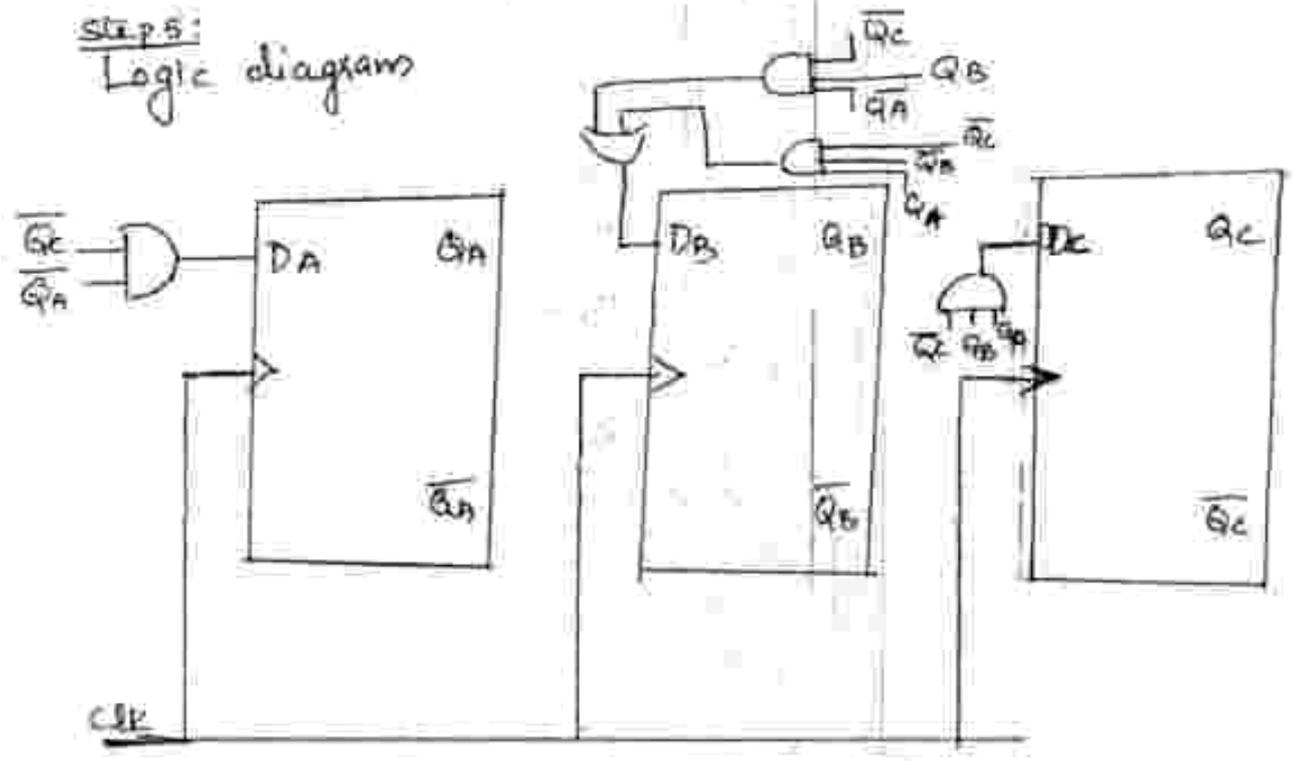


$$D_B = \bar{Q}_c \bar{Q}_B \bar{Q}_A + \bar{Q}_c Q_B \bar{Q}_A + Q_c \bar{Q}_B Q_A$$



$$D_A = \bar{Q}_c \bar{Q}_A$$

Steps: Logic diagrams



Random Sequence Detector

21) Design a type D counter that goes through states 0, 1, 2, 4, 0... The undesired states must always go to zero (000) on the next clock pulse

Step 1: The no. of flipflops

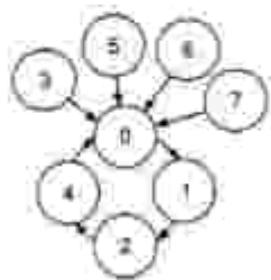
This counter has only 4 stable states 0, 1, 2 & 4. This counter has only 4 stable states 0, 1, 2 & 4 (000, 001, 010, 100), but it requires 3 FF's because it counts 4 (100) as well. Three FF's can have eight states. So the remaining four states (011, 101, 110, 11) are undesired. These undesired states must go to 000 after the next clock pulse. So, no don't cares.

Step 2: state diagrams

Step 3: The type of flip-flops & the excitation table

Step 4: Minimal expressions using K map

Step 5: logic diagrams.



(a) State diagram

NS	NS			Required excitations		
	Q ₂	Q ₁	Q ₀	D ₂	D ₁	D ₀
0	0	0	0	0	0	1
1	0	1	0	0	1	0
2	0	1	1	1	0	0
4	1	0	0	0	0	0
3	0	0	0	0	0	0
5	1	0	1	0	0	0
6	1	1	0	0	0	0
7	1	1	1	0	0	0

(b) Excitation table

Figure 12.45 Example 12.5: 0, 1, 2, 4, 0, ... counter.

K Map Equation

$$D_3 = \overline{Q_3} \cdot Q_2 \cdot \overline{Q_1} ; D_2 = \overline{Q_3} \cdot \overline{Q_2} \cdot Q_1 ; D_1 = \overline{Q_3} \cdot \overline{Q_2} \cdot \overline{Q_1}$$

63/67

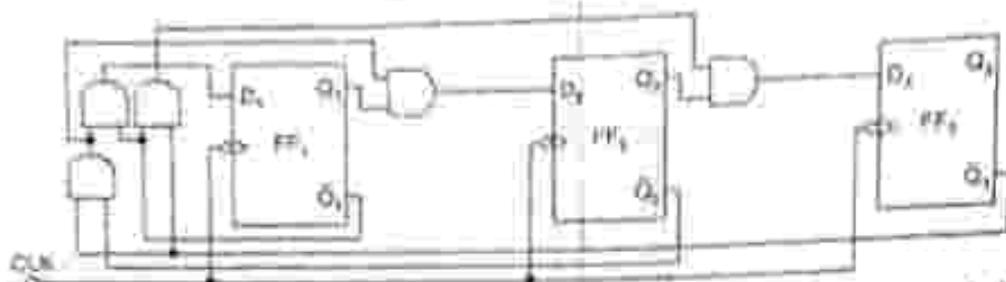


Figure 12.46: Example 12.5: Logic diagram of type D counter that goes through states 0, 1, 2, 4, 0, ...

Q2) Design a JK counter that goes through states 3, 4, 6, 7 &

5, ...

etc. - 1

No. of flip-flops: The counter has only 4 states:

3, 4, 6, 7 (011, 100, 110, 111), but the maximum count is 7. A counter using 3 flip-flops can have eight states.

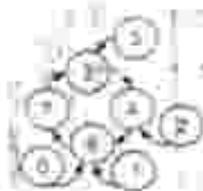
So the remaining four states (000, 001, 010, 101) are invalid.

Step 2: state diagram

Step 3: The type of flip-flops & the excitation table

Step 4: Minimal expression using K Map

Step 5: Logic diagram



(a) State diagram

PS			NS			Required excitation					
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0	J_2	K_2	J_1	K_1	J_0	K_0
0	1	1	1	0	0	1	0	1	0	1	0
1	0	0	1	1	0	0	0	1	0	0	0
1	1	0	1	1	1	0	0	0	0	1	0
1	1	1	0	1	1	0	1	0	0	0	0

(b) Excitation table

Figure 12.47: Example 12.6: Type JK 3, 4, 6, 7, 5 counter

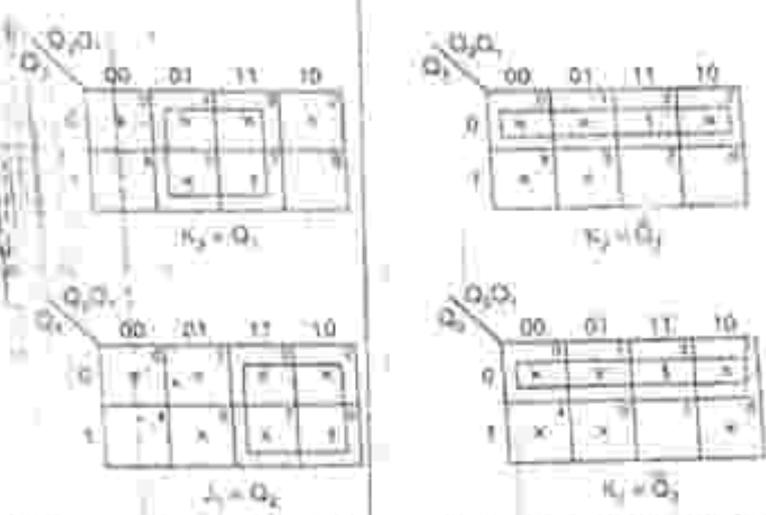


Figure 12.48 Example 12.6: K-maps for excitations of type J-K 3, 4, 5, 7, 3 counter

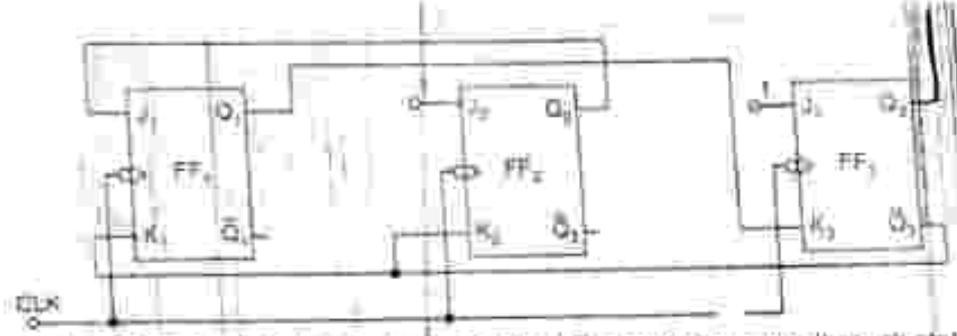


Figure 12.49 Example 12.3: Logic diagram of the J-K counter that goes through states 5, 4, 6, 7, 3.

Q3) Design a type T counter goes through states 0, 3, 5, 6, 0, ...

Step 1

The no: of flip-flops : This counter has only four stable states, but it requires three FF's, because it counts 101 & 110. Three Flipflops can have 8 states, out of which states 000, 011, 101, 110 are valid & states 001, 010, 100, 111 are invalid.

Step 2: State diagram

Step 3: The type of flip-flops & the excitation table

Step 4: Minimal expressions using K Map

Step 5: Logic diagram

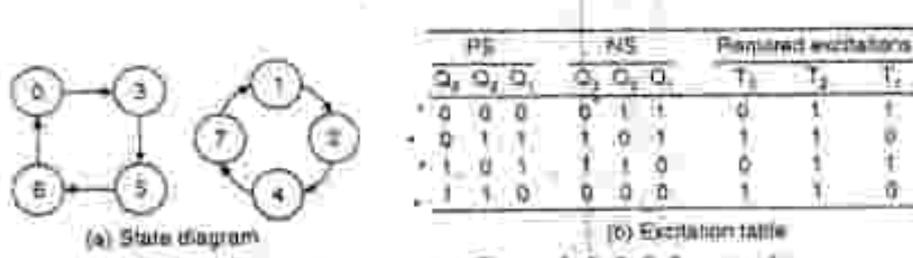


Figure 12.39 Example 12.4: T type, 0, 3, 5, 6, 0 - counter.

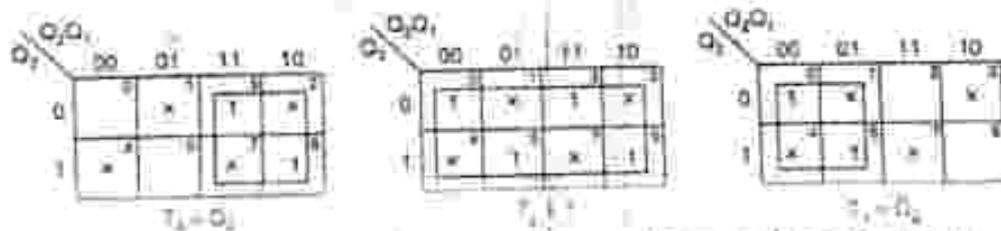
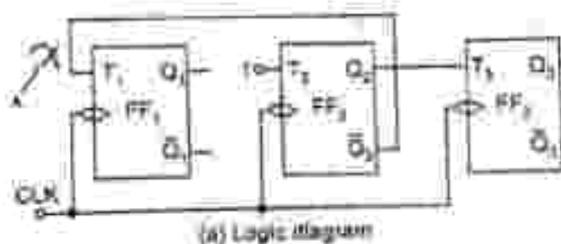


Figure 12.40 Example 12.4: K-maps for excitations of T type, 0, 3, 5, 6, 0 - counter.



PS			Present inputs			NS		
Q_2	Q_1	Q_0	T_2	T_1	T_0	Q_2	Q_1	Q_0
0	0	1	0	1	1	0	1	0
0	1	0	1	1	0	1	0	0
1	0	0	0	1	1	1	1	1
1	1	1	1	1	0	0	0	1

Figure 12.41 Example 12.4: T type counter that goes through states 0, 3, 5, 6, 0, ...

✓ Data may be available in parallel form or in serial form. Multi-bit data is said to be in parallel form when all the bits are available simultaneously.

The data is said to be in serial form when the data bits appear sequentially, at a single terminal.

Data may also be transferred in parallel form or in serial form.

Parallel data transfer is the simultaneous transmission of all bits of data from one device to another.

Serial data transfer is the transmission of one bit of data at a time from one device to another.

As a flip-flop can store only one bit of data, a 0 or a 1, it is referred to as a single-bit register.

When more bits of data are to be stored, a no. of FF's are used.

A register is a set of FF's used to store binary data.

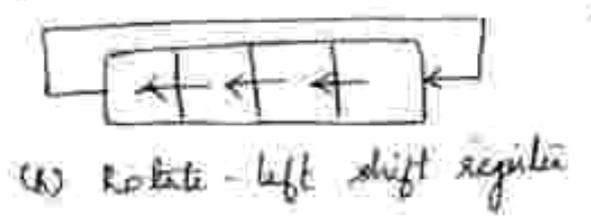
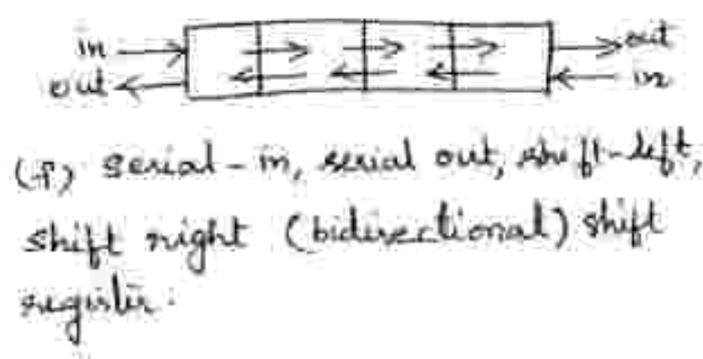
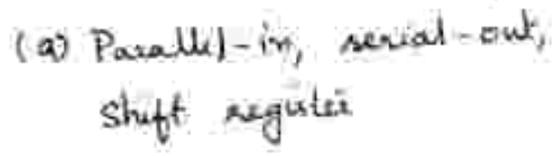
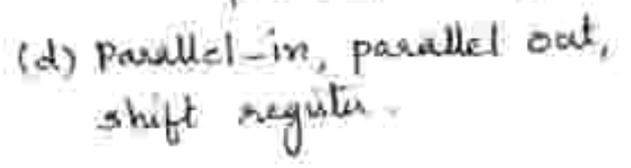
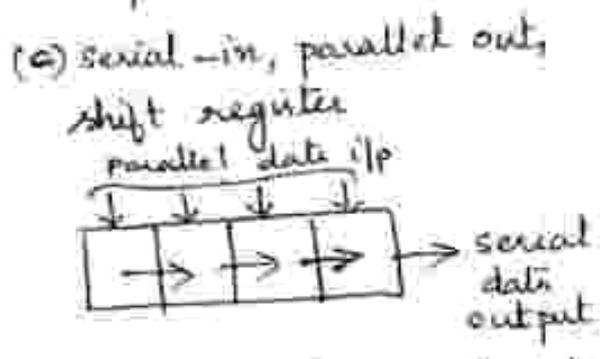
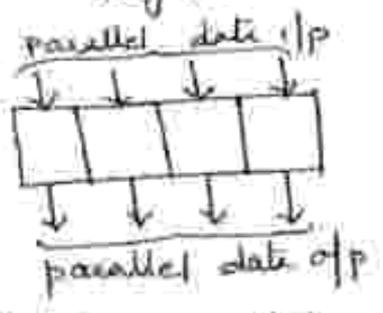
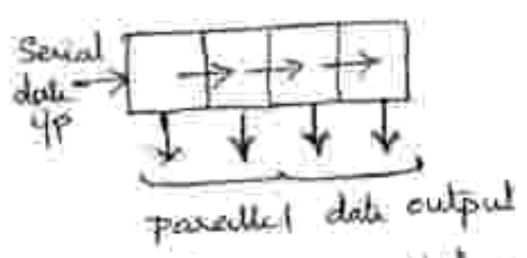
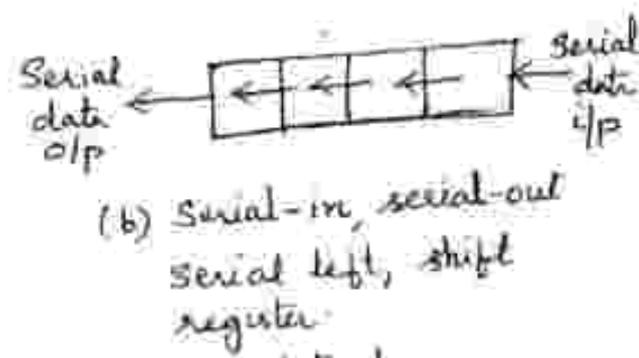
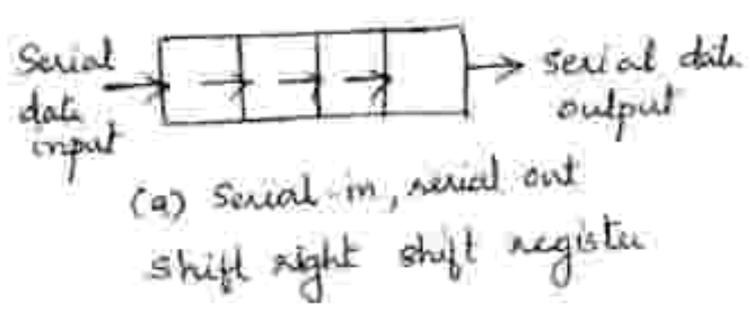
✓ The storage capacity of a register is the no. of bits (1's & 0's) of digital data it can retain.

Shift registers are a type of logic circuits closely related to counters.

They are used basically for the storage & transfer of digital data.

The basic difference between a shift register & a counter is that, a shift register has no specified sequence of states except in certain very specialised applications, whereas a counter has a specified sequence of states.

Data Transfer in Registers

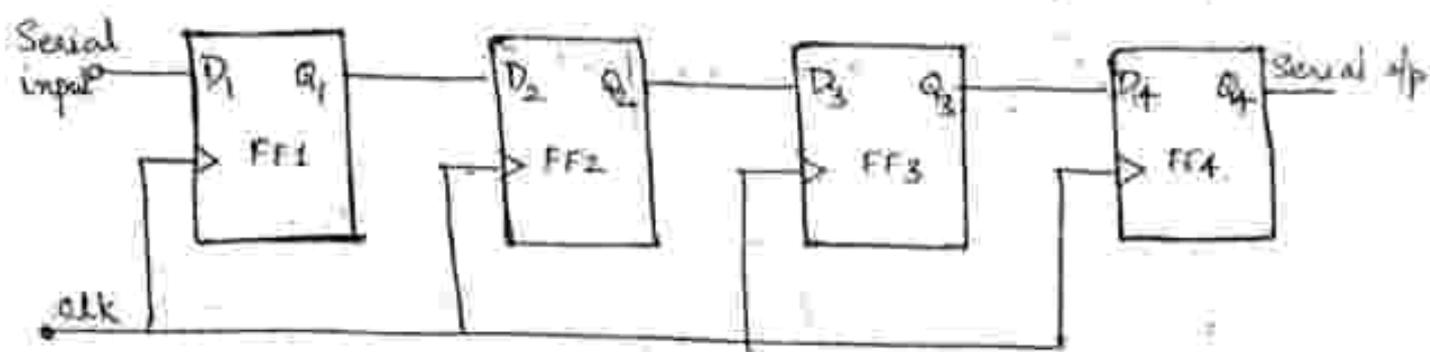


Shift Register

Serial - In Serial - Out, Shift Register

This type of shift register accepts data serially, i.e. one bit at a time and also outputs data serially.

The logic diagram of a 4 bit serial-in, serial-out, shift-right, shift register is shown below



With four stages i.e. four FF's, the register can store upto 4 bits of data.

Serial data is applied at the D input of the first FF.

The Q output of the 1st FF is connected to the D input of the 2nd FF, the Q output of the 2nd FF is connected to the D input of the 3rd FF & the Q output of the 3rd FF is connected to the D input of the 4th FF.

The data is outputted from the Q terminal of the last FF.

When serial data is transferred into a register, each new bit is clocked into the first FF at the positive-

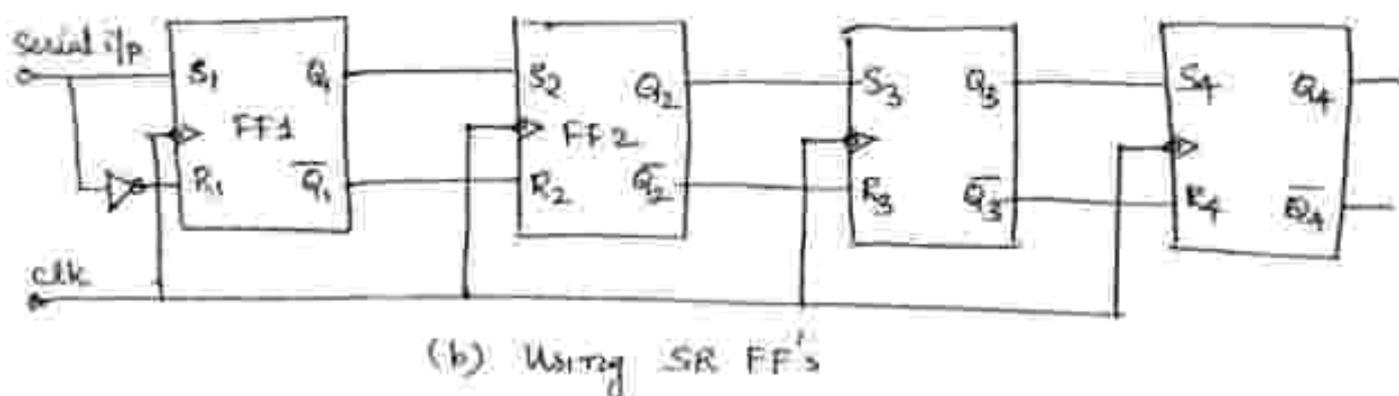
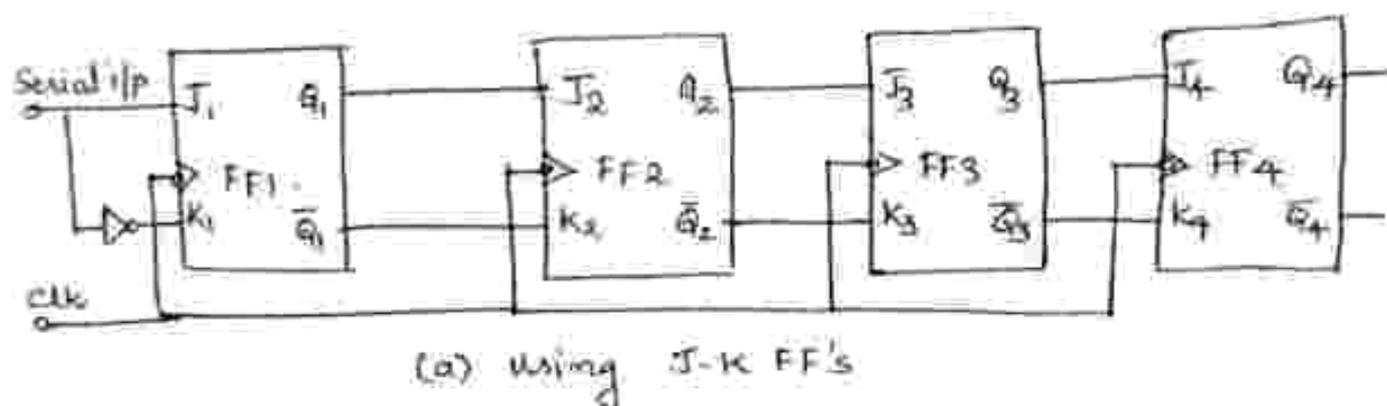
going edge of each clock pulse.

The bit that was previously stored by the first FF is transferred to the second FF.

The bit that was stored by the second FF is transferred to the 3rd FF & so on.

The bit that was stored by the last FF is shifted out.

A shift register can also be constructed using JK FF's or S-R FF's as shown in below:



The data is applied at the J(S) ip of the 1st FF.

The complement of this is fed to the K(R) terminal of the 1st FF.

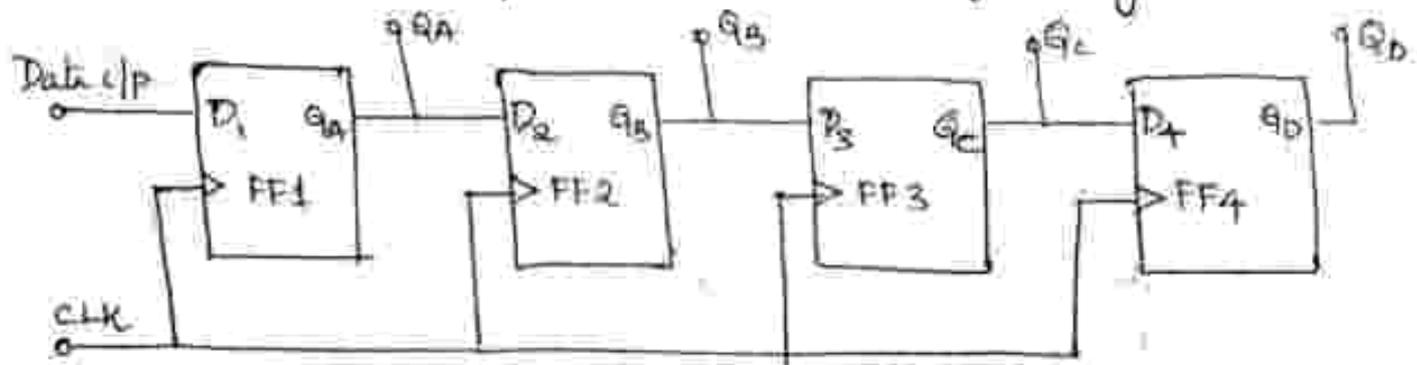
The Q output of the 1st FF is connected to the J(S)

1/2 of the 2nd FF, the Q of the 2nd FF to J(s) of the 3rd FF & so on.

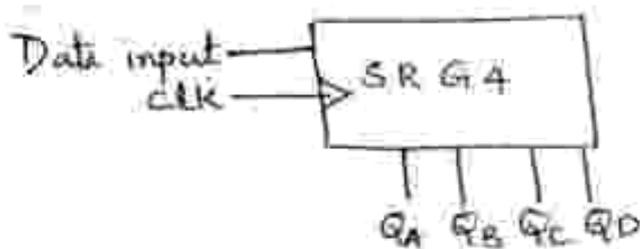
Also \bar{Q}_1 is connected to K_2 (R_2), \bar{Q}_2 is connected to K_3 (R_3) & so on.

Serial - In Parallel - Out Shift Register

Figure shows the logic diagram & logic symbol of a 4 bit serial-in, parallel-out shift register.



(a) Logic diagram



(b) Logic symbol

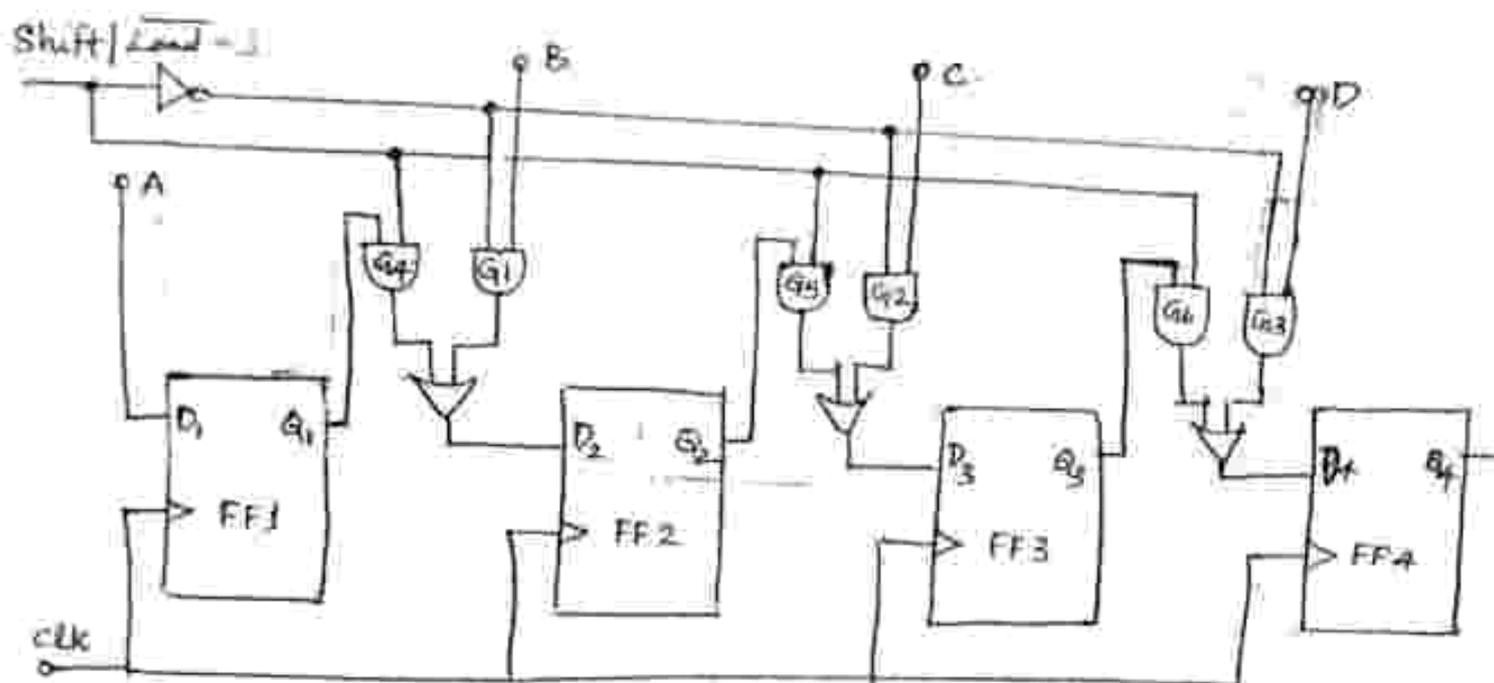
In this type of register, the data bits are entered into the register serially, but the data stored in the register is shifted out in parallel form.

Once the data bits are stored, each bit appears on its respective output line & all bits are available simultaneously.

rather than on a bit-by-bit basis as with the serial output.

The serial-in parallel-out, shift register can be used as a serial in, serial out, shift register if the output is taken from the Q terminal of the last FF.

Parallel - IN, SERIAL - OUT SHIFT REGISTER



(a) Logic diagram



(b) Logic symbol.

Figure illustrates a 4 bit parallel-in, serial out, shift register using DFF's.

There are four data lines A, B, C & D through which the data is entered into the register in parallel form.

The signal $\text{Shift}/\overline{\text{LOAD}}$ allows (a) the data to be entered in parallel form into the register & (b) the data to be shifted out serially from terminal Q_4 .

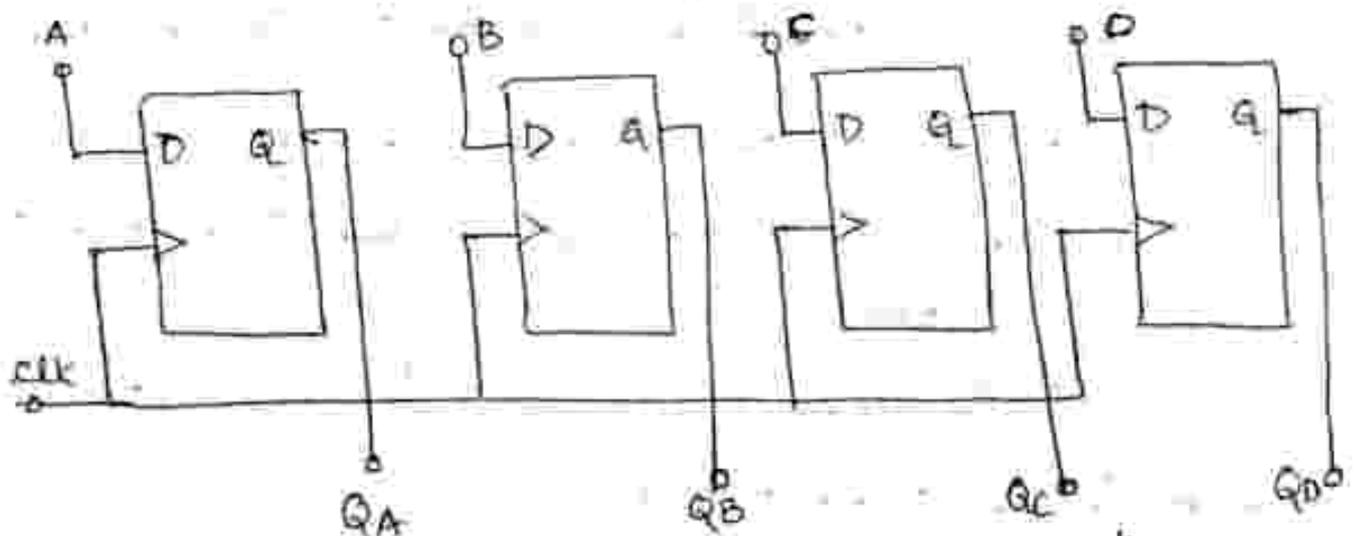
When $\text{Shift}/\overline{\text{LOAD}}$ line is HIGH, gates G_1, G_2 & G_3 are disabled, but gates G_4, G_5 & G_6 are enabled allowing the data bits to shift-right from one stage to the next.

When $\text{Shift}/\overline{\text{LOAD}}$ line is LOW, gates G_4, G_5 & G_6 are disabled, whereas gates G_1, G_2 & G_3 are enabled allowing the data input to appear at the D inputs of the respective FFs.

When a clk pulse is applied, these data bits are shifted to the Q output terminals of the FFs & therefore data is inputted in one step.

Parallel - IN Parallel OUT, Shift Register

In parallel-in, parallel-out, 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) Logic diagram of a 4 bit PISO shift register

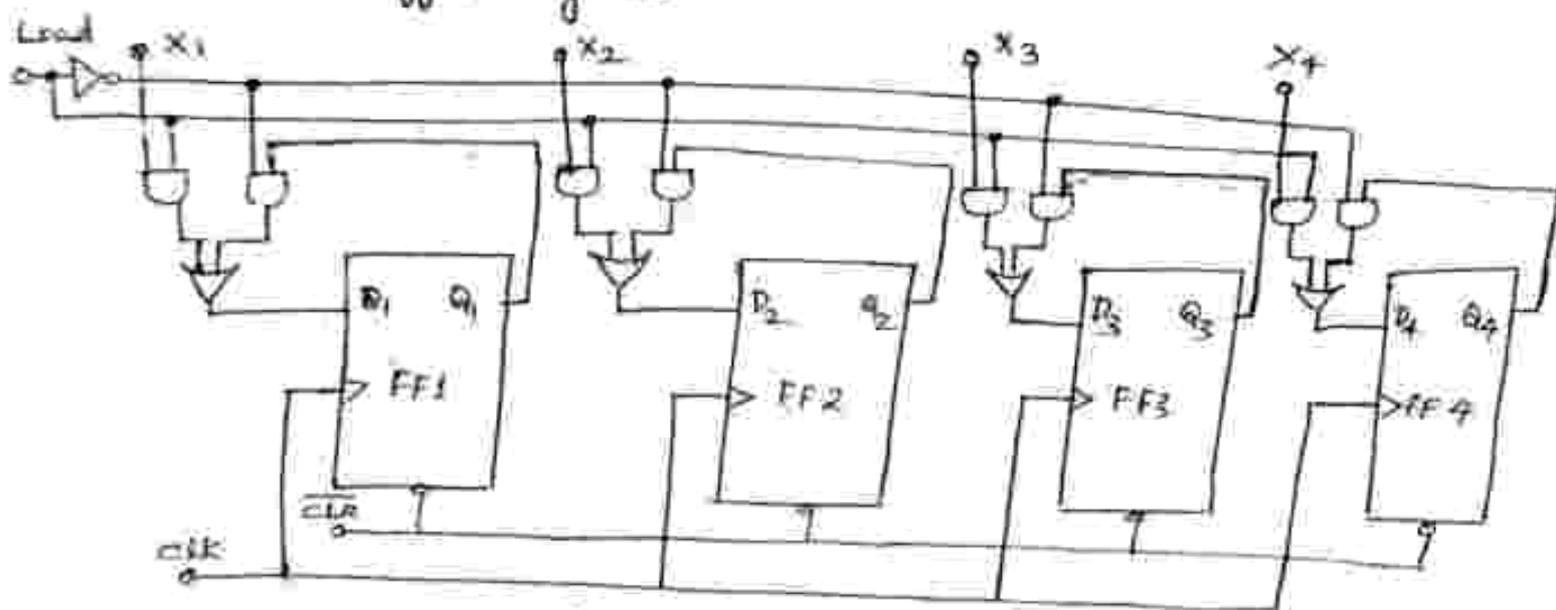
Figure shows a 4 bit PISO shift register using DFF's. Data is applied to the D input terminals of the FF's. When a clock pulse is applied, at the positive-going edge of that pulse, the D inputs are shifted into the Q outputs of the FF's.

The register now stores the data.

The stored data is available instantaneously for shifting out in parallel form.

Shift Register with parallel LOAD/SHIFT

Controlled Buffer Register



(a) Logic diagram of a 4 bit controlled buffer register

Figure shows a controlled buffer register..

If \overline{CLR} goes Low, all the FF's are RESET & the output becomes, $Q = 0000$

When \overline{CLR} is HIGH, the register is ready for action. LOAD is the control input.

When LOAD is HIGH, the data bits x can reach the

D) inputs of FFs.

At the true going edge of the next clock pulse, the register is loaded if

$$Q_4 Q_3 Q_2 Q_1 = X_4 X_3 X_2 X_1$$

$$Q = X$$

When LOAD is Low, the X bits cannot reach the FF's

At the same time, the inverted signal $\overline{\text{LOAD}}$ is HIGH.

This forces each flip-flop output to feed back to its data input.

Therefore, data is circulated or retained as each clock pulse arrives. In other words, the contents of the register remain unchanged in spite of the clock pulses.

Shift Register Counters

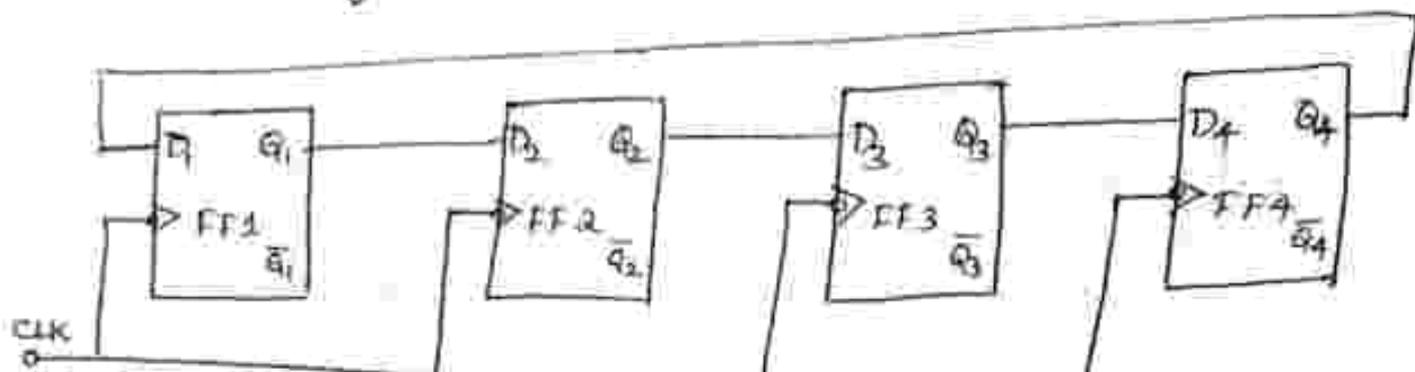
One of the applications of shift registers is that they can be arranged to form several types of counters.

Shift register counters are obtained from serial-in, serial-out shift registers by providing feedback from the output of the last FF to the input of the first FF. These devices are called counters because they exhibit a specified sequence of states.

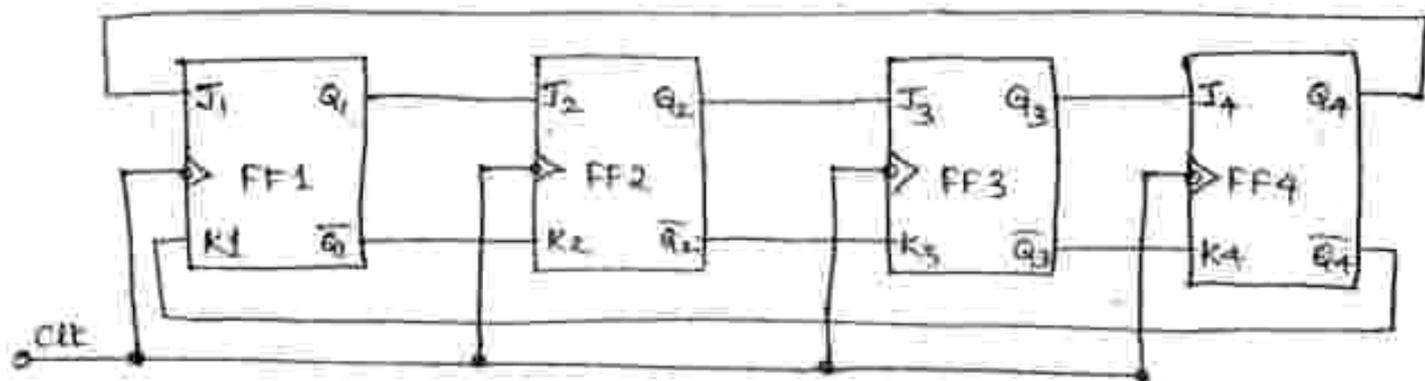
The most widely used shift register counter is the ring counter as well as the twisted ring counter.

Ring Counter

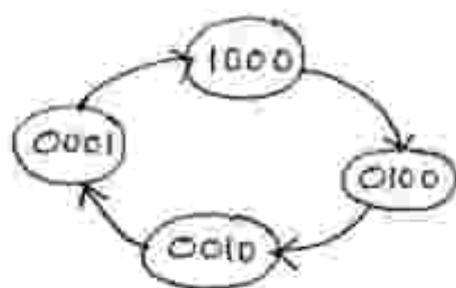
This is the simplest shift register counter.
The basic ring counter using D FF's is shown in figure (a)



(a) Logic diagram of a 4 bit ring counter using D flip-flops



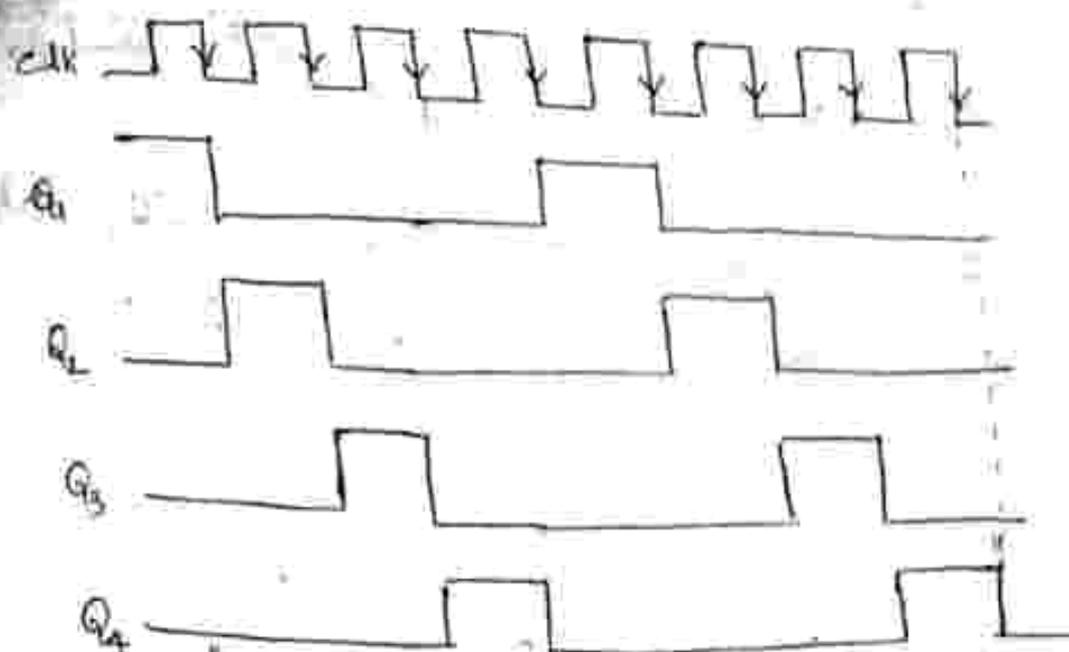
(b) Logic diagram of a 4 bit ring counter using JK flipflops



(c) state diagram

Q_1	Q_2	Q_3	Q_4	After clock pulse
1	0	0	0	0
0	1	0	0	1
0	0	1	0	2
0	0	0	1	3
1	0	0	0	4
0	1	0	0	5
0	0	1	0	6
0	0	0	1	7

(d) Sequence table



(e). Timing diagram of a 4 bit ring counter.

The FF's are arranged in a normal shift register i.e. the Q o/p of each stage is connected to the D i/p of the next stage, but the Q output of the last FF is connected back to the D input of the 1st FF such that the array of FF's is arranged in a ring & \therefore , the name ring counter.

In most instances, only a single 1 is in the register & is made to circulate around the register as long as clock pulses are applied.

Initially the 1st FF is preset to a 1. So, the initial state is 1000, i.e. $Q_1 = 1$, $Q_2 = 0$, $Q_3 = 0$, $Q_4 = 0$.

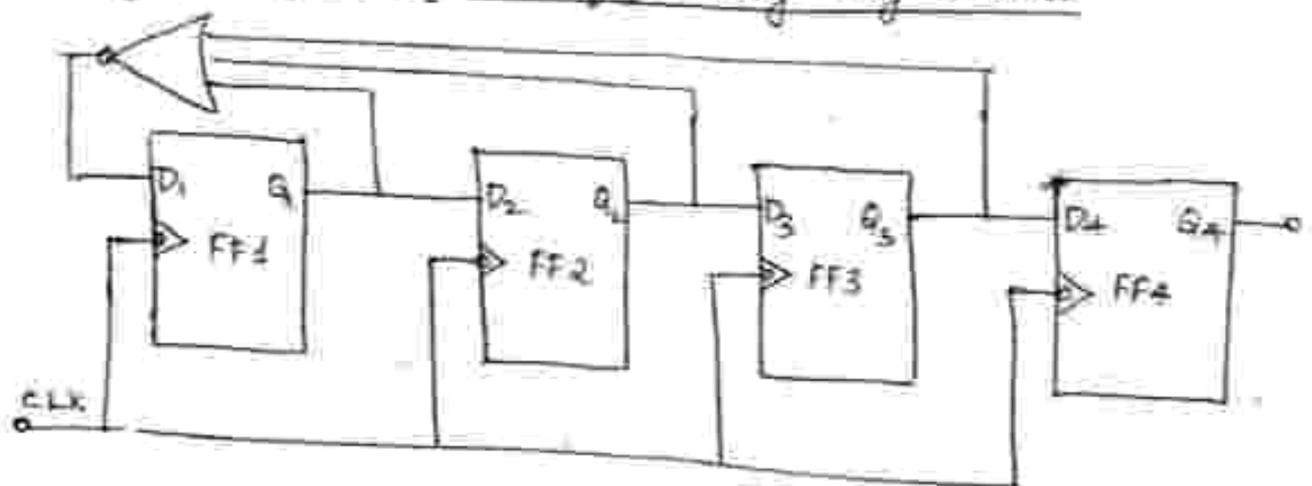
After each clock pulse, the contents of the register are shifted to the right by one bit & Q_4 is shifted back to Q_1 .

The sequence repeats after 4 clock pulses.

The no: of distinct states in the ring counter is the max of the ring counter is equal to no: of FF's used in the counter.

An n -bit ring counter can count only n bits, whereas an n -bit ripple counter can count 2^n bits.

Logic-diagram of a Self Starting Ring Counter



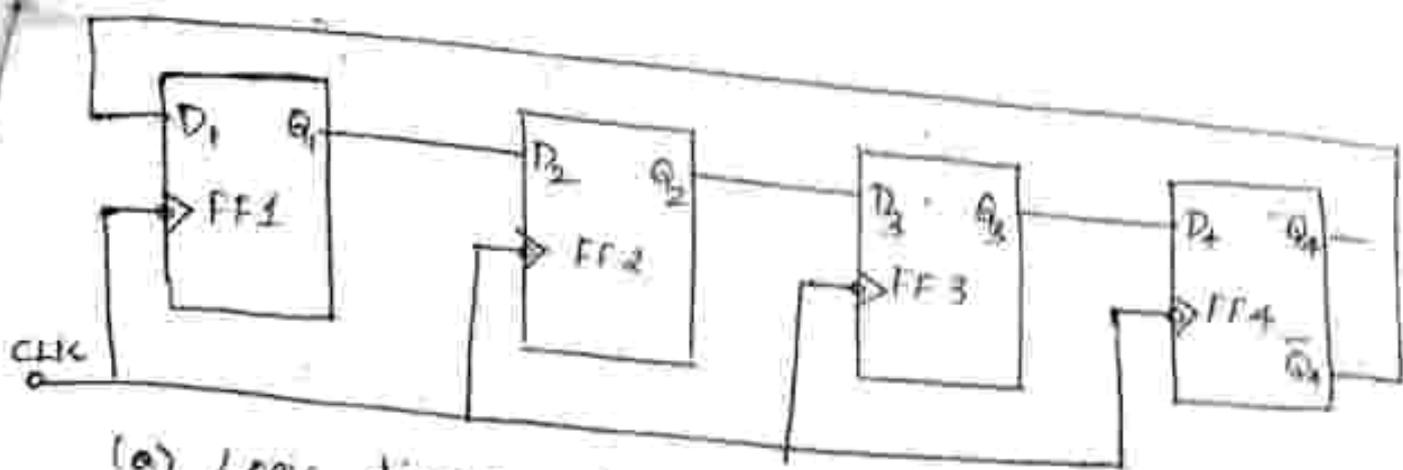
Johnson Counter (Twisted Ring Counter)

This counter is obtained from a serial-in, serial out shift register by providing feedback from the inverted output of the last FF to the D input of the first FF.

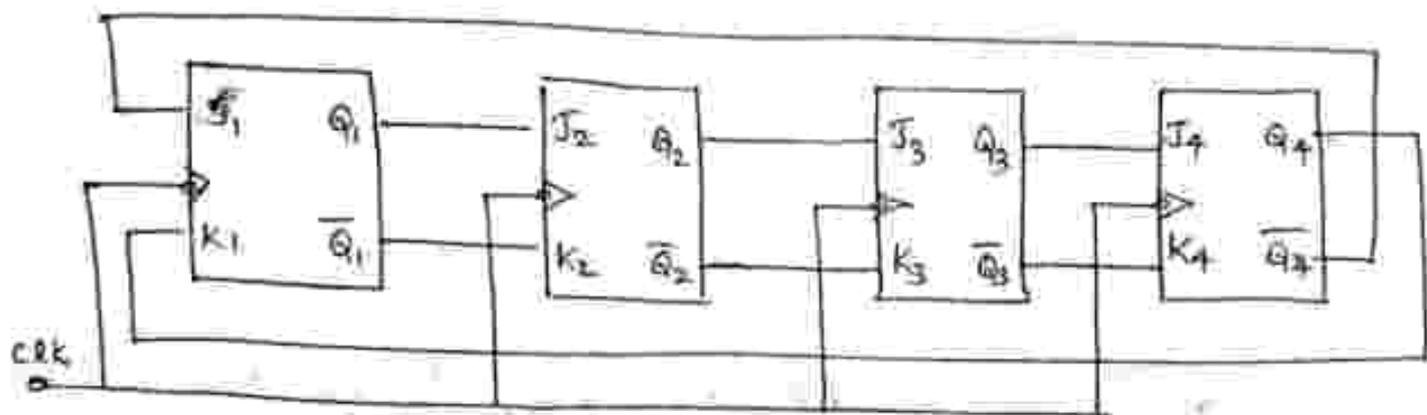
The Q output of each stage is connected to the D input of the next stage, but the \bar{Q} output of the last stage is connected to the D input of the first stage, therefore the name twisted ring counter.

This feedback arrangement produces a unique sequence of states.

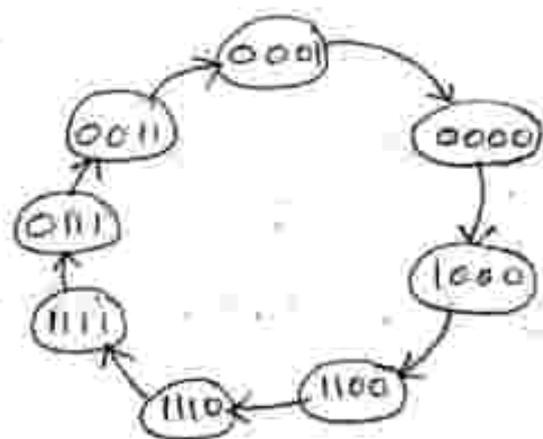
The logic diagram of a 4 bit Johnson counter using D FF is shown in figure (a)



(a) Logic diagram of a 4 bit twisted ring counter using D flip-flops.



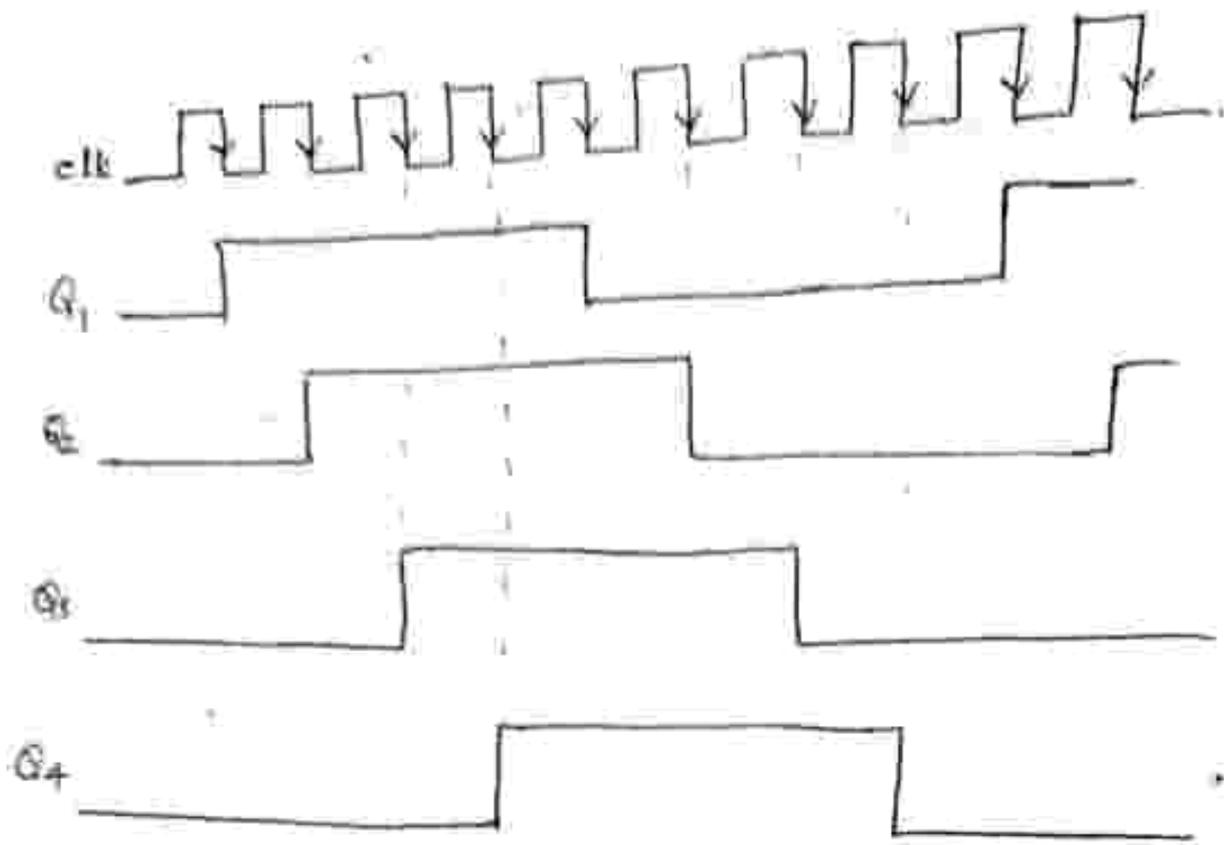
(b) Logic diagram of a 4 bit twisted ring counter using JK flip-flops.



(c) state diagram

Q_1	Q_2	Q_3	Q_4	After clock pulse
0	0	0	0	0
1	0	0	0	1
1	1	0	0	2
1	1	1	0	3
1	1	1	1	4
0	1	1	1	5
0	0	1	1	6
0	0	0	1	7
0	0	0	0	8
1	0	0	0	9

(d) sequence table



(e) Timing diagram of a 4 bit twisted ring counter.

Let initially all the FFs be reset, i.e. the state of the counter be 0000.

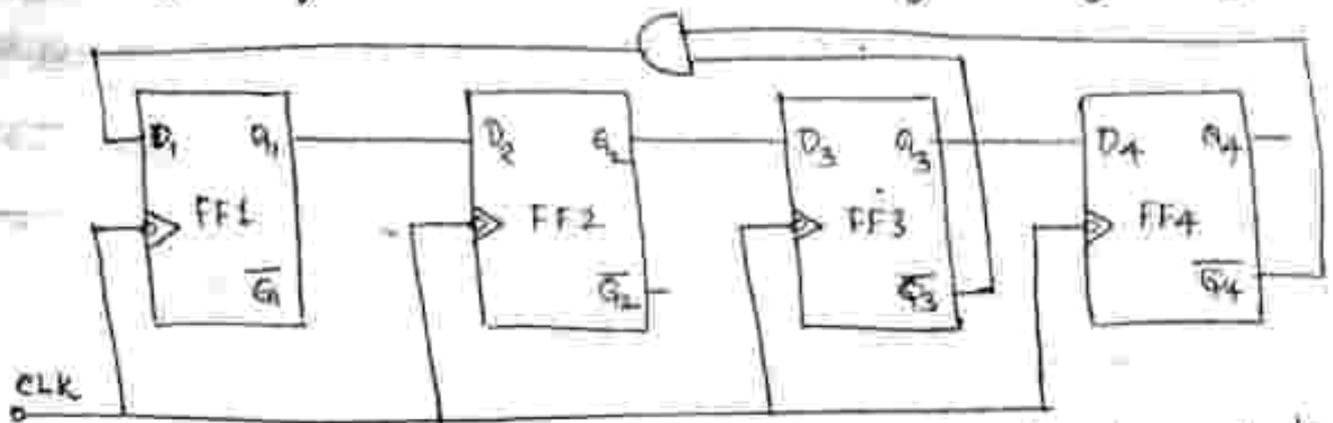
After each clock pulse, the level of Q_1 is shifted to Q_2 , the level of Q_2 to Q_3 , Q_3 to Q_4 & the level of \bar{Q}_4 to Q_1 & the sequence given in sequence table. This sequence is repeated after every eight clock pulses.

An n FF Johnson counter can have $2n$ unique states & can count up to $2n$ pulses. So it is a mod- $2n$ counter.

Lock out

Both types of ring counters suffer from the problem of lock out i.e. if the counter find itself in an unused state, it will persist in moving from one unused state to another & will never find its way to a used state.

This difficulty can be corrected by adding a gate.



Logic diagram of a 4 bit Johnson counter designed to prevent lock out.

Bidirectional Shift Register

A bidirectional shift register is one in which the data bits can be shifted from left to right or from right to left.

Figure shows the block diagram of a 4 bit serial in, serial out bidirectional shift register using D flip flop.

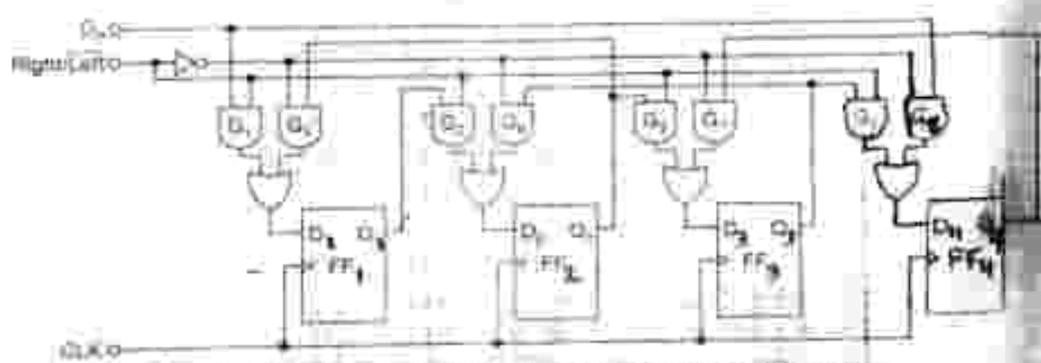


Figure 11.10 Logic diagram of a 4 bit bidirectional shift register.

When $\text{Right}/\overline{\text{Left}} = 1$. Here, Right/Left is the mode signal. When

The logic circuit works as a shift right shift register. i.e. it enables the AND gates (G₁, G₂, G₃, G₄) and disables other AND gates (G₅, G₆, G₇, G₈). Then the Q output of each FF is passed through the gate to the D input of the following FF.

When a clock pulse occurs, the data bits are then shifted one place to the right.

When Right / $\overline{\text{Left}} = 0$,

The logic circuit works as a shift left shift register, i.e. it enables the AND gates C_{01} , C_{12} , C_{23} and C_{34} and disables the other AND gates. (C_{01} , C_{23} , and C_{34}) - Then the Q output of each FF is passed to the D input of the preceding FF.

When a clock pulse occurs, the data bits are then shifted one place to the left. Hence the circuit works as a bidirectional shift register.

Universal Shift Register.

Universal Shift Register is a bidirectional register, whose input can be either in serial form or in parallel form and whose output also can be either in serial form or in parallel form. The most general shift register has the

the following capabilities

1. A clear control to clear the register to 0.
2. A clock input to synchronize the operation.
3. A shift-right control, to enable the shift-right operation.
4. A shift-left control to enable the shift-left operation.
5. A parallel load control to enable a parallel transfer and n input lines associated with the parallel transfer.
6. n parallel output lines.
7. A control state that leaves the information in the register unchanged, in the presence of the clock.

Figure below shows a universal shift register realized using multiplexers.

A bit universal shift register is shown. It consists of ^{four} $n/2$ D flipflops and 4 multiplexers.

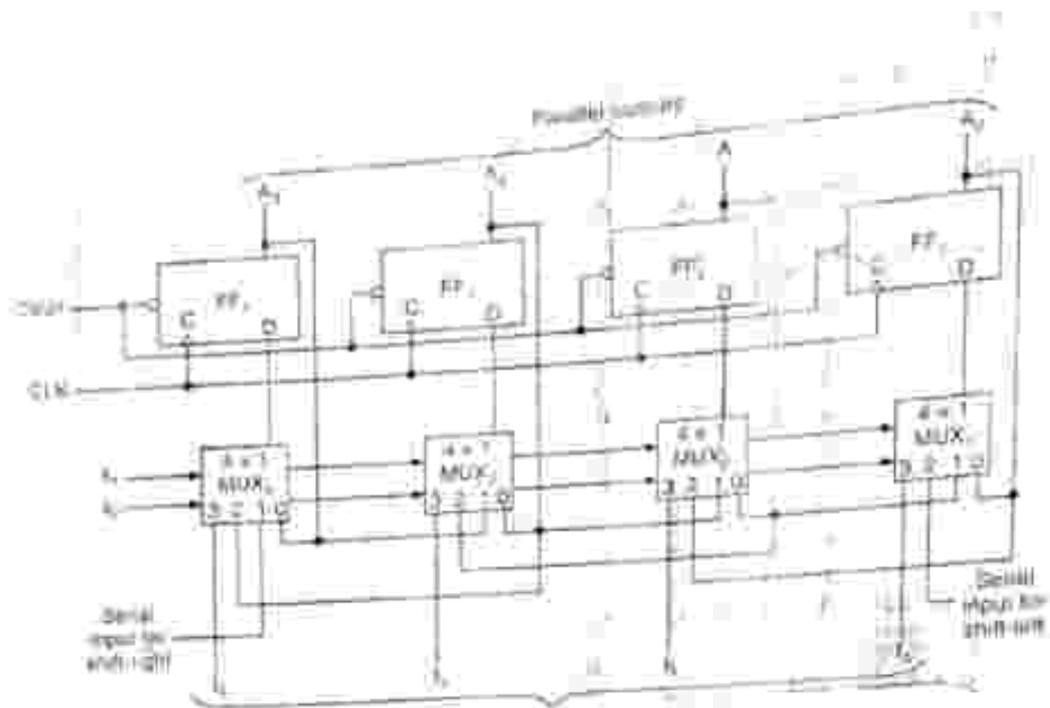


Figure 11.11 4-bit universal shift register

Table 11.1 Function table for the register of Figure 11.11

Mode control		Register operation
S_1	S_0	
0	0	No change
0	1	Shift right
1	0	Shift left
1	1	Parallel load

The four multiplexers have two common selection inputs S_1 and S_0 .

Input 0 in each MUX is selected when

$$S_1, S_0 = 00$$

Input 1, when $S_1, S_0 = 01$

Input 2, when $S_1, S_0 = 10$

Input 3, " $S_1, S_0 = 11$

The selection input control the mode of operation of the register according to the

function entries in the above table

When $S_1 S_0 = 00$, ~~terminal 0~~ terminal 0 of MUX is ^{connected.} ~~change~~ No change of state occurs.

When $S_1 S_0 = 01$, terminal 1 of MUX is ~~input~~ connected to D_{in} of each FF

⊖ This causes a shift-right operation.

with the serial input transferred in to FF_n

When $S_1 S_0 = 10$, ~~terminal 2~~ terminal 2 of MUX ^{The results} connected, shift left operation.

with the other serial input - going

to FF₁

Finally when $S_1 S_0 = 11$, ~~terminal 3~~ terminal 3 of MUX is ^{connected.} ~~Results~~ terminal 3 of MUX is connected. Then the binary information on the parallel input lines is transferred in to the register simultaneously during the next clock edge.