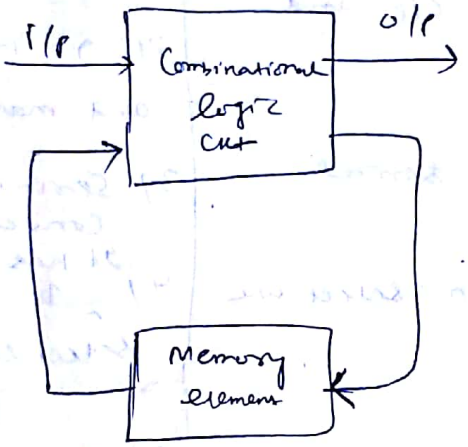
Sequential Logic ckt :-

→ In sequential logic ckt the o/p not only depends upon the present i/p but also depends on the past i/p.

→ Although every digital system is likely to have combinational ckts, most systems encountered in practice also include storage elements.



→ A block diagram of sequential ckt is shown in fig. It is a combination of a combinational logic ckt and memory element.

→ Therefore the o/p of the sequential logic ckt depends on the present state of i/p and state of the memory element.

→ The state of memory element in term depends on the previous state o/p, which in term depends on the previous state i/p.

→ The Basic sequential logic ckts are Latch and flip-flops.

Latches :-

→ A latch is a type of temporary storage device that has 2 stable states (bistable)

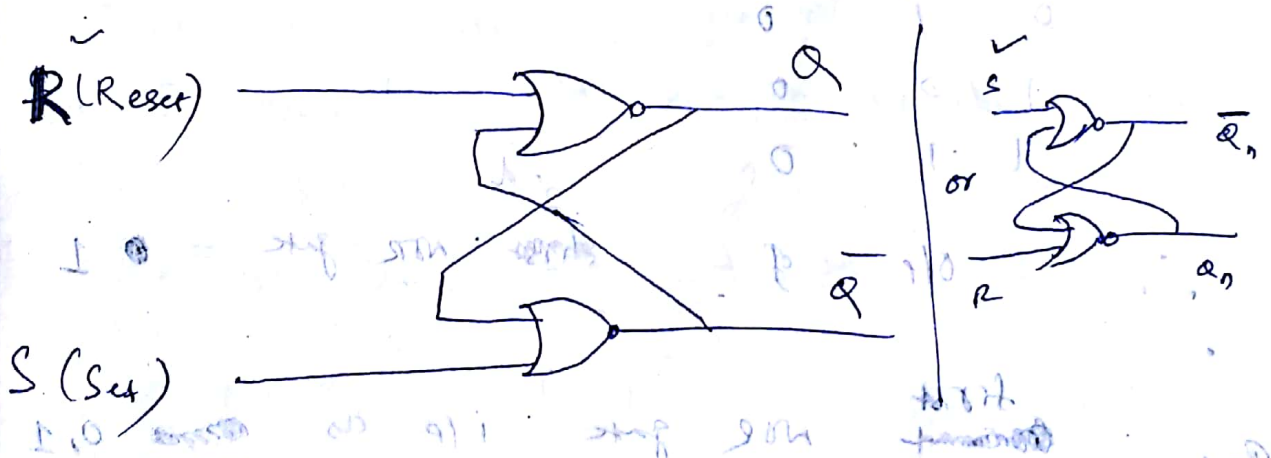
→ Two stable states are SET (1) and RESET (0). they (Latches) can retain either of these states indefinitely, making them useful as storage devices.

→ For storing 1 bit data, latch is used.

SR Latch :-

→ The SR Latch is a ckt with 2 cross-coupled NOR gates or two cross coupled NAND gates.

→ It has 2 I/Ps labelled S for Set and R for Reset



Jump

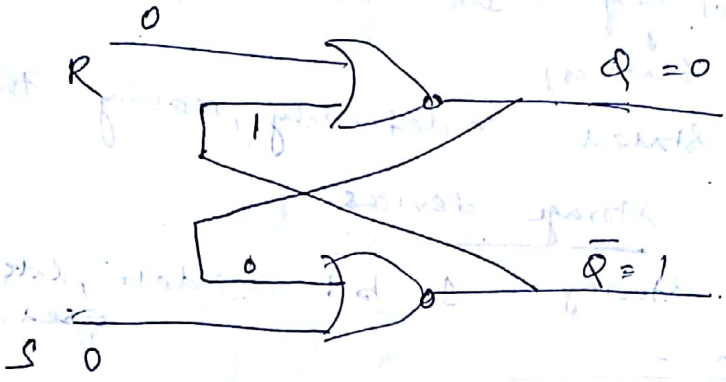
S	R	Q_{n+1}	$\bar{Q}_{n+1}$	Remarks
0	0	Q_n	$\bar{Q}_n$	No change → In this case, it acts as storage element
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Invalid NOT used

Explanation :-

Let $S=0, R=0, Q=0$

~~Since~~ $Q=0, \bar{Q}=1$,

Now i/p for 2nd NOR gate is 0, 0



~~$S=1, Q=0$~~

NOR gate

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

$\therefore$ o/p of 2nd NOR gate = 1

For first NOR gate i/p is 0, 1

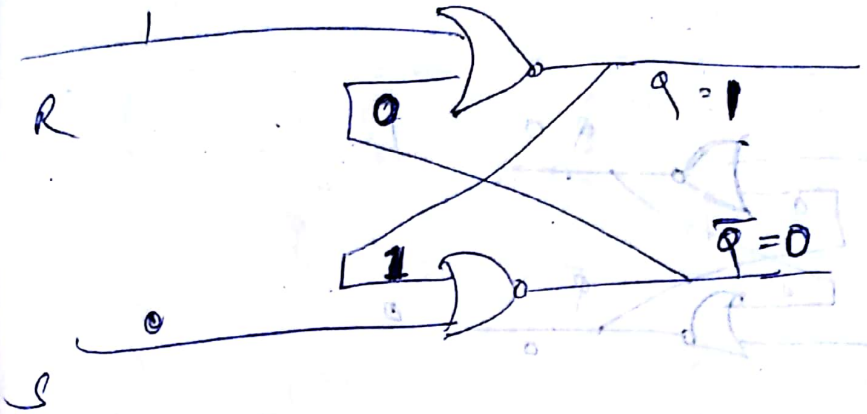
o/p will be 0

S	R	Q_{n+1}	$\bar{Q}_{n+1}$	Remarks
0	0	Q_n	$\bar{Q}_n$	No change

$\therefore$ The o/p remains unchanged

Case – II

$S = 0$, $R = 1$, $Q = 1$



Let $Q = 1$,

i/p for 2nd NOR gate is 1, 0

o/p is 0.

Now i/p for first NOR gate is 1, 0

(because o/p of 2nd NOR gate connected to first NOR gate)

o/p of first NOR gate is 0.

Now the 0 will be connected to 2nd NOR gate i.e 0, 0 i/p.

o/p of 2nd NOR will be 1.

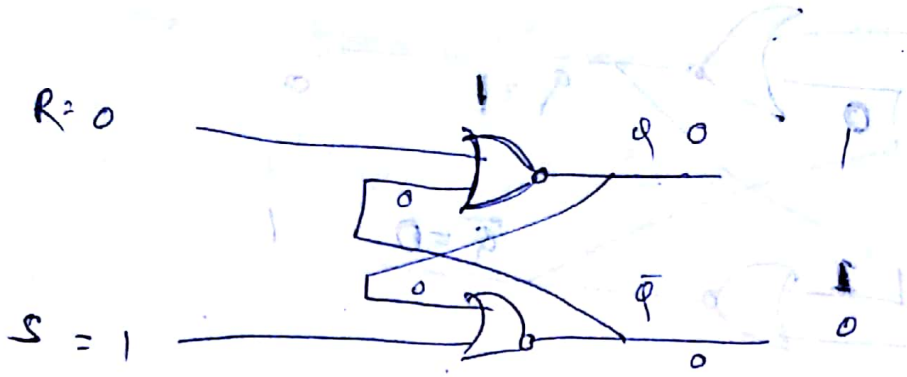
$Q = 0$, $\overline{Q} = 1$

Q is reset to 0.

S	R	Q_{n+1}	$\overline{Q}_{n+1}$	Remark
0	1	0	1	Reset.

Case - III

$S = 1, R = 0, Q = 0$



$Q = 0$, i/p for 2nd gate is 0, 1

o/p of 2nd NOR gate is 0.

Now, i/p for first gate is 0, 0

o/p is 1. Q

Now i/p for 2nd gate is 1, 1

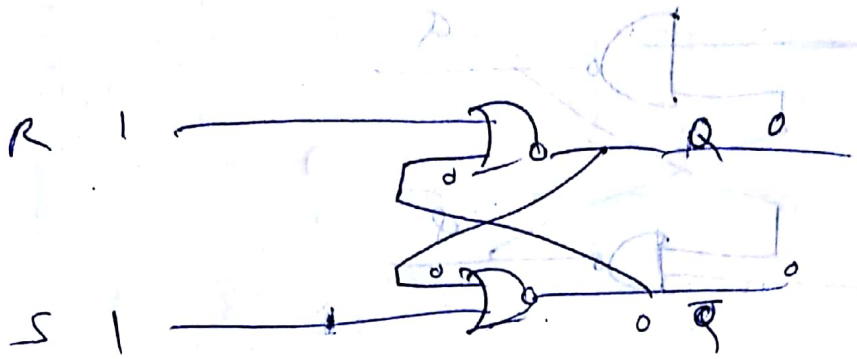
o/p of 2nd gate 0 $\bar{Q}$

∴ Q is set to 1.

S	R	Q_{n+1}	$\bar{Q}_{n+1}$	Remark
1	0	1	0	Set.

Case IV

$S = 1$, $R = 1$, $Q = 0$



$Q = 0$	i/p for 2nd	NOR gate is	0,1
	o/p of 1 "	" " is	0.
Now	i/p for first	NOR gate is	1,0
	o/p of first	NOR gate is	0

~~After~~ from	now onwards	the o/p	
& $\bar{o/p}$	both will remain	at	0,0
When S	Invalid.		

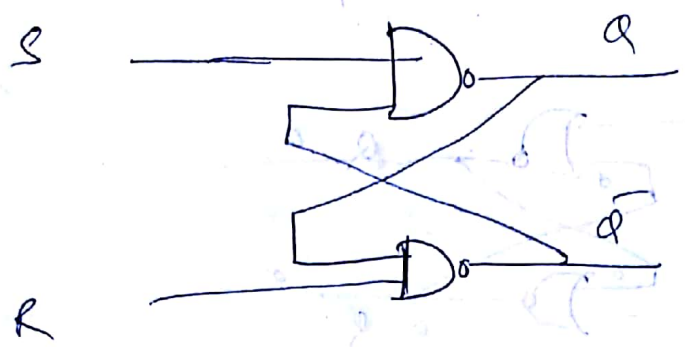
Because $\bar{Q}$ should be complemented of Q, but both remain at 0, 0.

S	R	Q_{n+1}	$\bar{Q}_{n+1}$	Remark
1	1	0	0	Invalid.

Another explanation

When both o/ps are 0,0. and we give R=0, & S=0 then it should retain the value 0,0, as R=0, S=0 is the requirement for a latch behaving as memory device. But the o/p is 0, 1. which is not expected. So S = 1, R=1 is never used. It is a unused or invalid i/p.

SR Latch with NAND gates:-

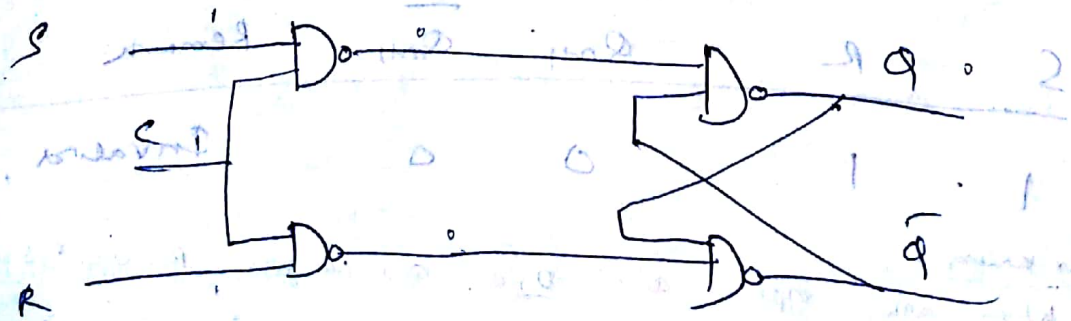


S	R	Q_{n+1}	$\bar{Q}_{n+1}$	Remark
0	0	1	1	Invalid
0	1	1	0	~~Reset~~
1	0	0	1	Set
1	1	Q_n	$\bar{Q}_n$	No change

Imp

S	R	Q_{n+1}	$\bar{Q}_{n+1}$	Remark
0	0	1	1	Invalid / Unused
0	1	1	0	Set
1	0	0	1	Reset
1	1	Q_n	$\bar{Q}_n$	No change → Used as memory element.

SR Latch with Control i/p

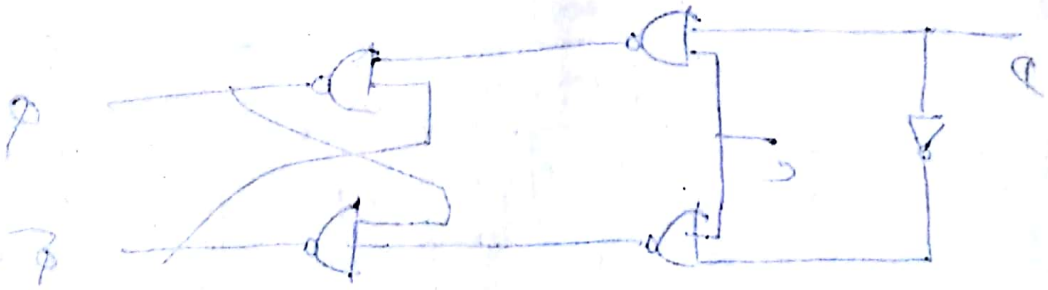


C	S	R	Q_{n+1}	$\bar{Q}_{n+1}$	Remark
0	X	X	Q_n	$\bar{Q}_n$	No change
1	0	0	Q_n	$\bar{Q}_n$	No change
1	0	1	0	1	Reset
1	1	0	1	0	Set
1	1	1	1	1	Invalid

← Control Signal

If C = 0, what ever be the value of S & R (X, X), X → Don't care, the O/P remains unchanged.

(C = 0, O/P of first 2, Nand gates is 1, 1) ← No change, this is the I/Ps for the second 2 Nand gates)



S	R	Q	$\bar{Q}$
		X	0
		0	1
		1	1

D : Latch :-

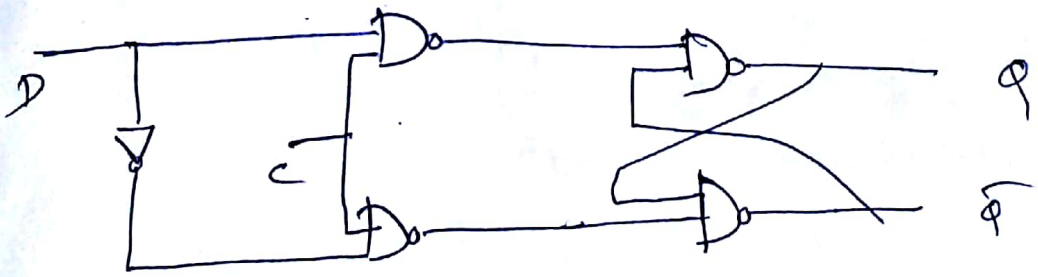
→ One way to eliminate the undesirable condition of indeterminate / invalid condn in SR Latch is to ensure that S & R should never be equal.

→ This is done by D latch.

→ It has 2 I/Ps : D (Data) & C (Control).

2 O/P → Q & $\bar{Q}$

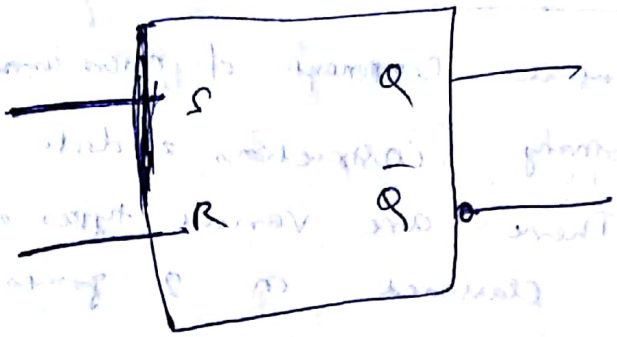
→ We have connected a NOT gate in between SR Latch to convert into D latch.



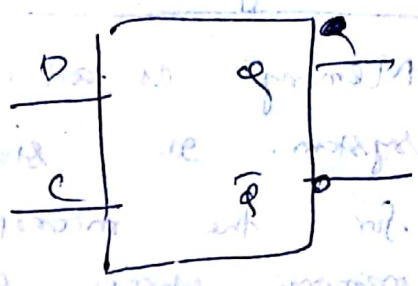
C	D	Q_{n+1}	$\bar{Q}_{n+1}$	Remarks
0	X	Q_n	$\bar{Q}_n$	No change
1	0	0	1	'0' is Latched
1	1	1	0	'1' is Latched

If Control is not enabled, it will remain as it is.

If Control is 1, $\begin{cases} D = 0, & Q = 0 \\ D = 1, & Q = 1 \end{cases}$



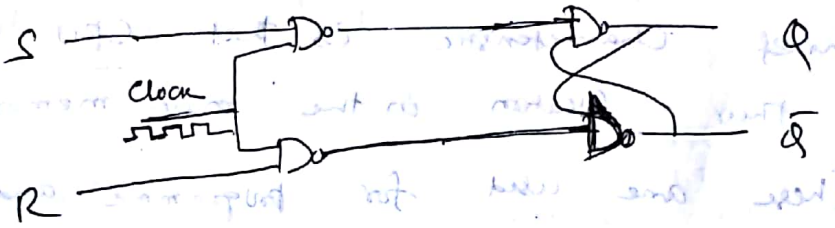
(SR Latch)



(D Latch)

Topic:- SR FIR

→



Clock	S	R	Q_{n+1} $\bar{Q}_{n+1}$	Remark
0	X	X	Q_n $\bar{Q}_n$	No change
↑	0	0	Q_n $\bar{Q}_n$	No change
↑	0	1	0 1	Reset
↑	1	0	1 0	Set
↑	1	1	1 1	Invalid

2-mark - 2013 BPUT

Difference between Latch & FIR

1) The difference between latch and FIF is that latch does not have a clock signal, where as FIF always have a clock signal.

2) Latch is level triggered where as FIR is edge triggered.

Flip-Flops (F/F)

→ A sequential ckt is specified by time sequence of I/Ps, O/Ps & internal states.

→ In synchronous sequential circuits, the change of internal state occures in response to the synchronized clock pulses.

→ Asynchronous sequential ckt don't use clock pulses. The change of internal state occures when there is a change in the I/p variables.

→ The memory elements in synchronous sequential circuits are clocked flip flops.

→ The memory elements in asynchronous sequential circuits are either unclocked flip flops or time-delay elements.

→ The memory capability of a time-delay device is due to the finite time it takes for the signal to propagate through digital gates.

→ An asynchronous sequential ckt quite often resembles a combinational ckt with feedback.

→ The design of asynchronous sequential circuits is more difficult than that of synchronous circuits because of the timing problems involved in the feedback path.

→ In a properly designed synchronous system, timing problems are eliminated by triggering all F/Fs with the pulse edge.

→ Asynchronous circuits are used when speed of operation is important, especially in those cases where the digital system must respond quickly without having to wait for clock pulse.

→ They are more economical to use in small independent systems that require only a few components, as it may not be practical to go the expense of providing a circuit for generating clock pulses.

→ Digital designers often produce a mixed system where some part of the synchronous system has the characteristics of an asynchronous circuit.

In Summary

Synchronous sequential ckts

1) The change of internal state occures in response to the synchronized clock pulses.

2) The memory elements in synchronous ckts are clocked F/Fs.

3) ~~The design~~ The timing problems are eliminated by triggering all F/Fs with pulse edge. this type of ckt is so easier to design.

4) Less speed, because they depend on clock pulse.

5) Costly, because they need additional ckt for generating clock pulse.

6) Ex:- SR F/F, D F/F, T F/F, JK F/F, Synchronous Counters etc.

Asynchronous sequential ckts

1) The change of internal state occures when there is a change in the i/p variables.

2) The memory elements in asynchronous ckts are either unclocked F/Fs or time-delay elements.

3) The design of this type ckt is more difficult because of timing problems involved in the feedback path.

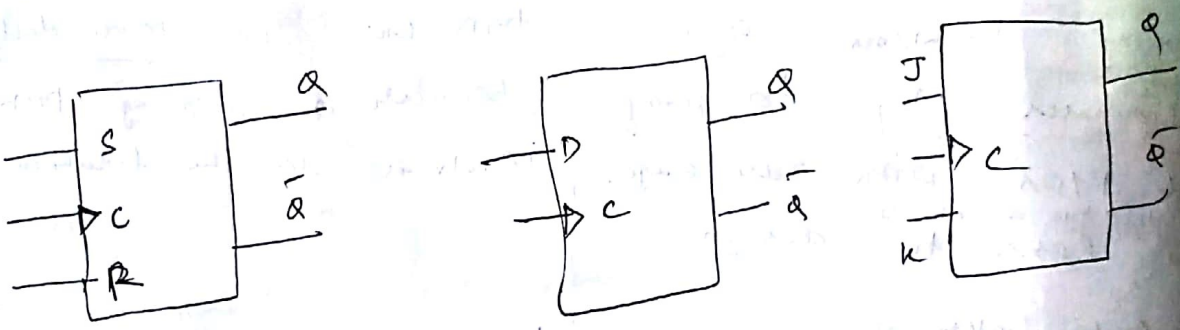
4) These ckts are used when speed of operation is required. Because it does not have to wait for clock pulse.

5) More economical, because they don't require ckt for generating clock.

6) Ex:- SR latch, D-latch, Asynchronous Counter, etc.

Edge-Triggered F/F's

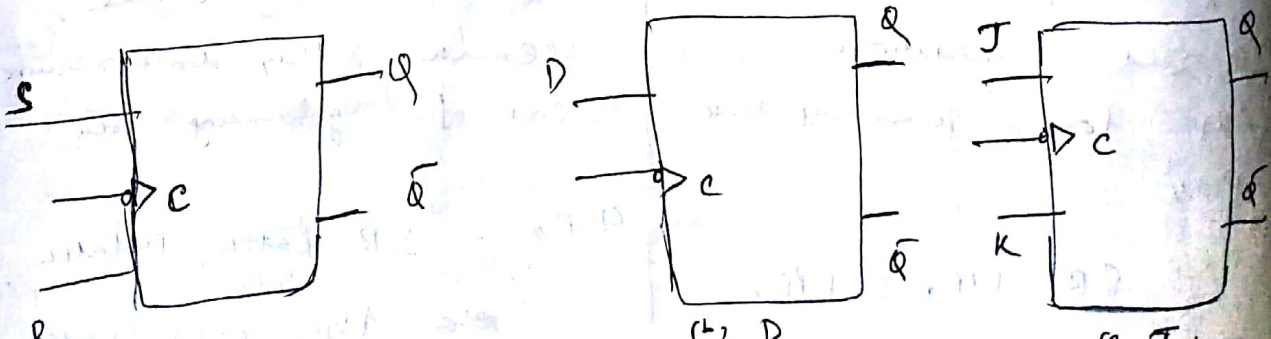
Flip flops are Synchronous bistable devices, also known as bistable multivibrators. In this case, the term Synchronous means that the O/P changes state only at a specified point on a triggering I/P called the Clock, which is designated as a Control I/P, C; that is, Changes in the O/P occurs in Synchronization with the clock.



(a) S-R (b) D (c) J-K

Fig. 1

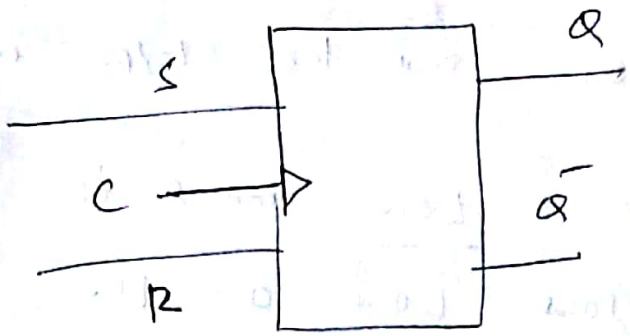
(+ve edge triggered F/F)



(a) S-R (b) D (c) J-K

(Fig 2: -ve edge triggered F/F)

Edge Triggered S-R F/F



→ The S and R I/Ps of the SR F/F are called Synchronous I/Ps because the data on these I/Ps are transferred to the F/F's O/P only on the triggering edge of the clock pulse.

Clock	S	R	Q_{n+1}	$\bar{Q}_{n+1}$	Remark
0	X	X	Q_n	$\bar{Q}_n$	No change
↑	0	0	Q_n	$\bar{Q}_n$	No change
↑	0	1	0	1	RESET
↑	1	0	1	0	SET
↑	1	1	? (1)	? (1)	Invalid

→ When no clock pulse is given, whatever may be the value of S & R the O/P is unchanged or remains same.

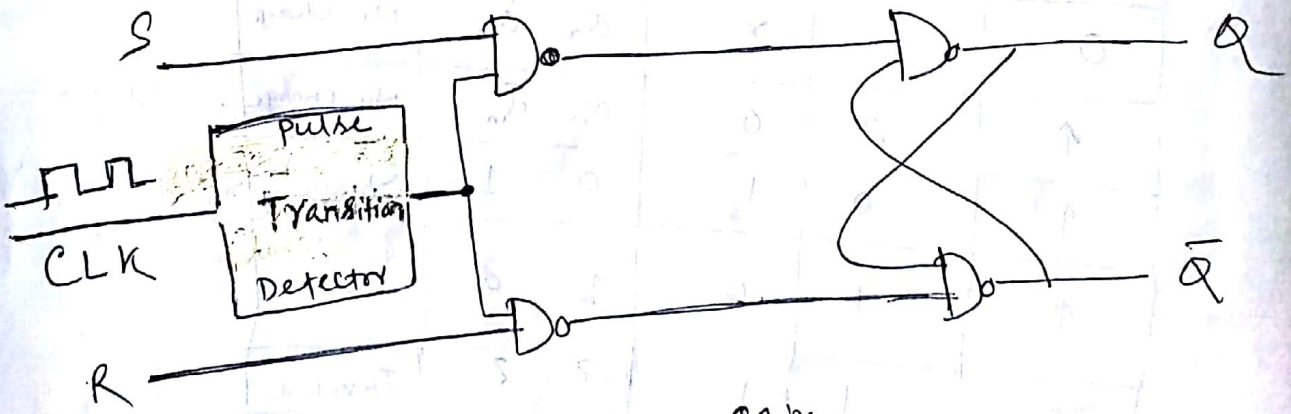
→ When Clock signal is given and

S is HIGH and R is LOW, the Q o/p goes HIGH on the triggering edge of the Clock pulse and the F/F is SET.

When S is LOW and R is HIGH the Q o/p goes LOW on the triggering edge of the clock pulse, and the F/F is RESET.

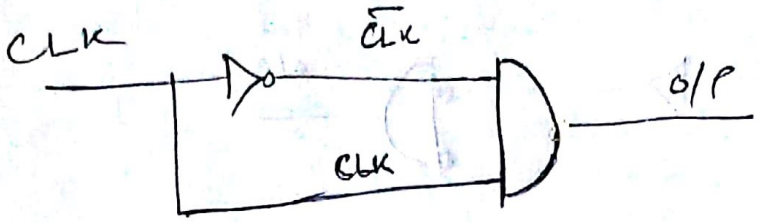
When both S and R are LOW, the o/p does not change from its prior state.

An invalid condition exists when both S and R are HIGH.



Generation of narrow spike

A narrow +ve spike is generated at the rising edge of the Clock using a pulse transition detector. It Consist of an inverter and a AND gate.



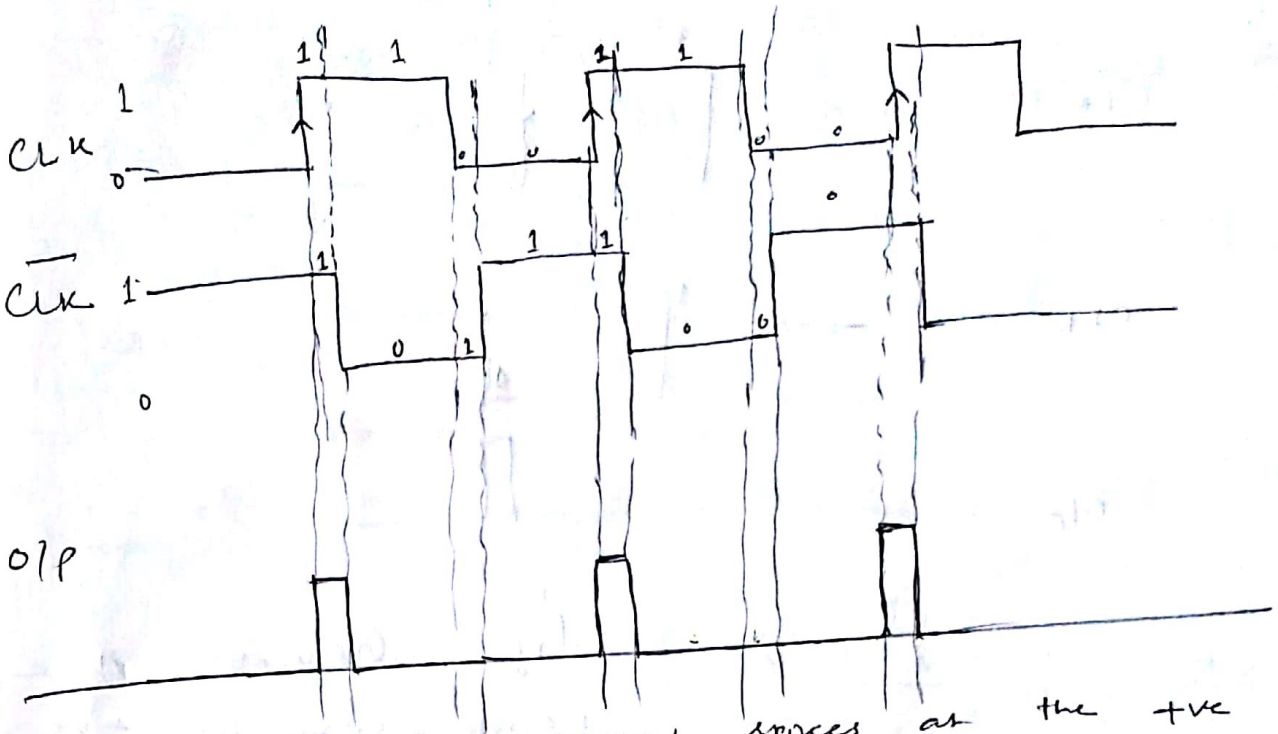
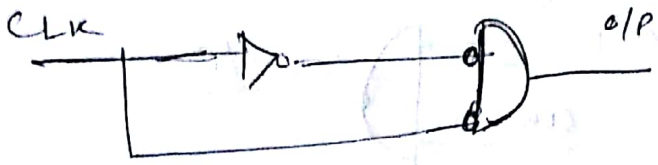
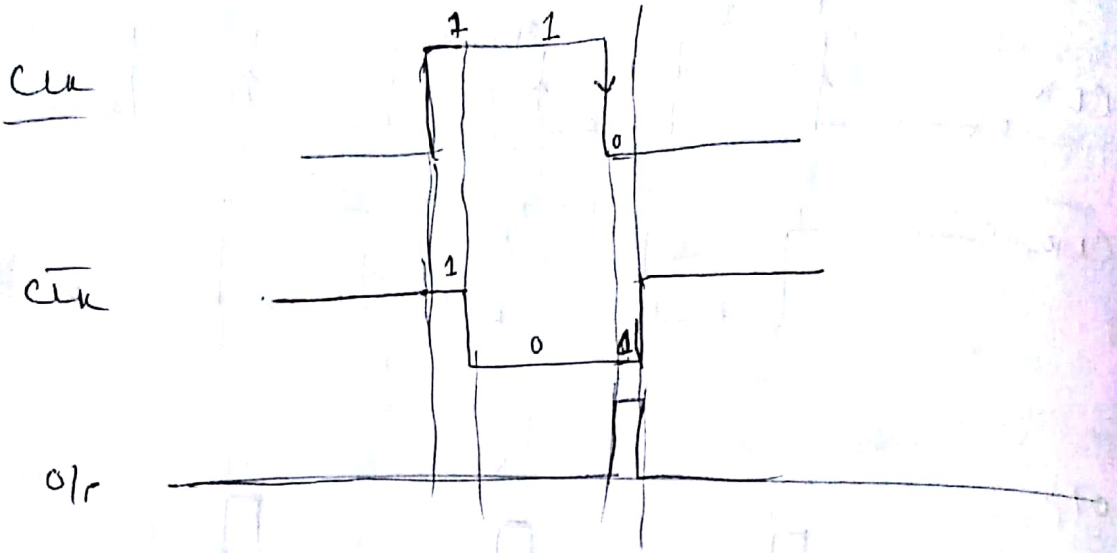


Fig:- Generation of narrow spikes at the +ve going transition of the clock pulse.

→ The inverter produces a delay of few nano seconds. The AND gate produces an O/P spike that is HIGH only for a few nano seconds, when both CLK and $\overline{CLK}$ are HIGH. This results in narrow spike at the O/P of AND gate which occures at the +ve-going transition of the clock signal.

→ Similarly, a narrow +ve spike is generated at the falling edge of the clock by using an inverter and an active-LOW AND gate.

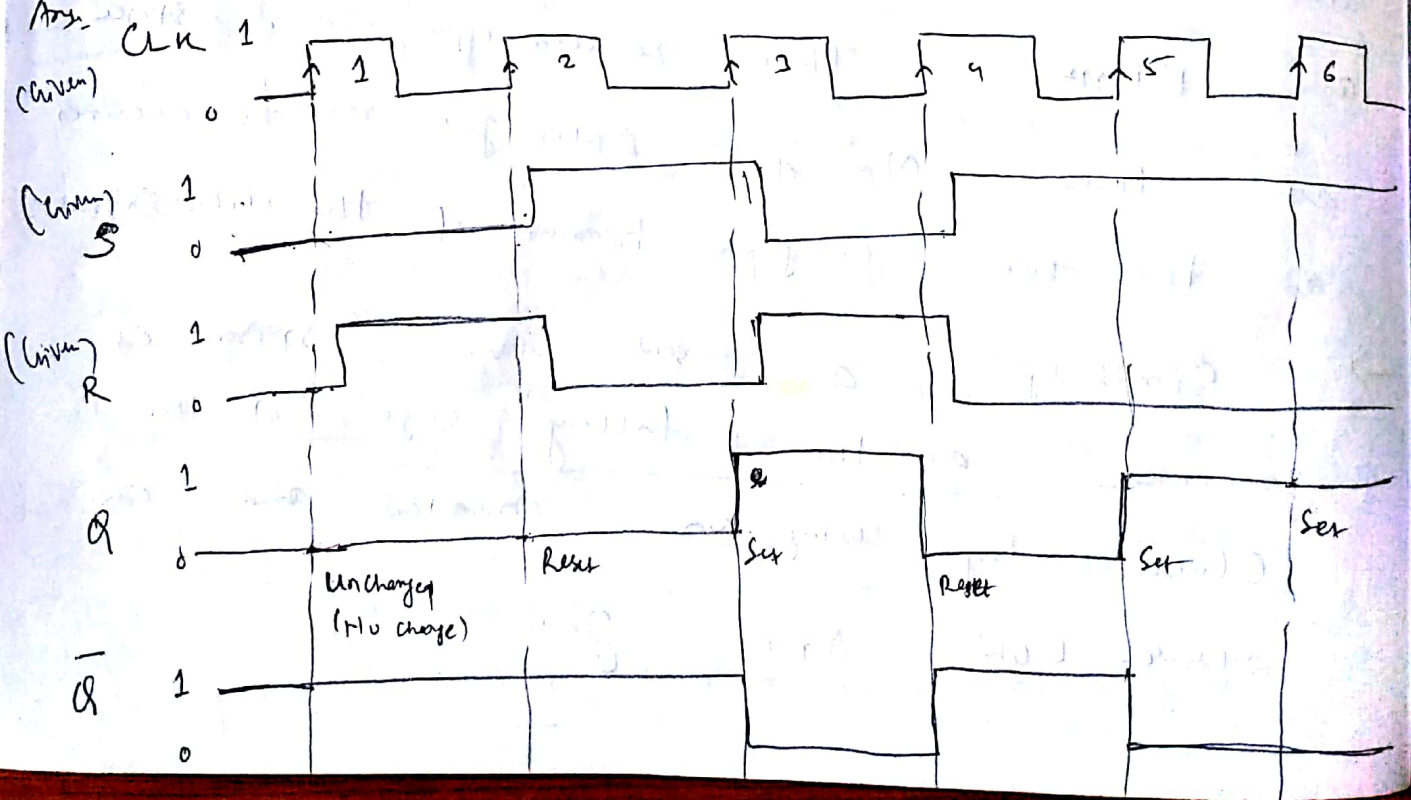




A	B	$\overline{A}$	$\overline{B}$	$\overline{A}\cdot\overline{B}$	o/p
1	1	0	0	0.0	0
1	0	0	1	0.1	0
0	0	1	1	1.1	1

fig:- Generation of a narrow spike at +ve going transition of the Clock pulse.

Q) Determine the Q & $\overline{Q}$ waveforms of the S-R F/F triggered F/F. Assume that the +ve edge initially Reset.



1. At Clock Pulse 1, S = LOW, R = LOW
So Q remains LOW. (No change case)

2. At Clock Pulse 2, (Edge),

2. This will continue till another edge of Clock Pulse is arrived because transition takes place at the +ve edge

3. So at +ve edge of Clock pulse 2, S = LOW, R = HIGH, So Q is LOW.

4. At Clock pulse 3, S = 1, R = 0, O/P is HIGH or SET.

5. At Clock pulse 4, S = 0, R = 1, Q = RESET

5. " " 5, S = 1, R = 0, Q = SET

6, " " 6, S = 1, R = 0, Q = SET

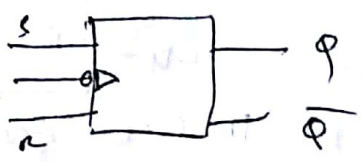
∴ Q stays HIGH.

Once Q is determined, $\bar{Q}$ is simply the Complement of Q.

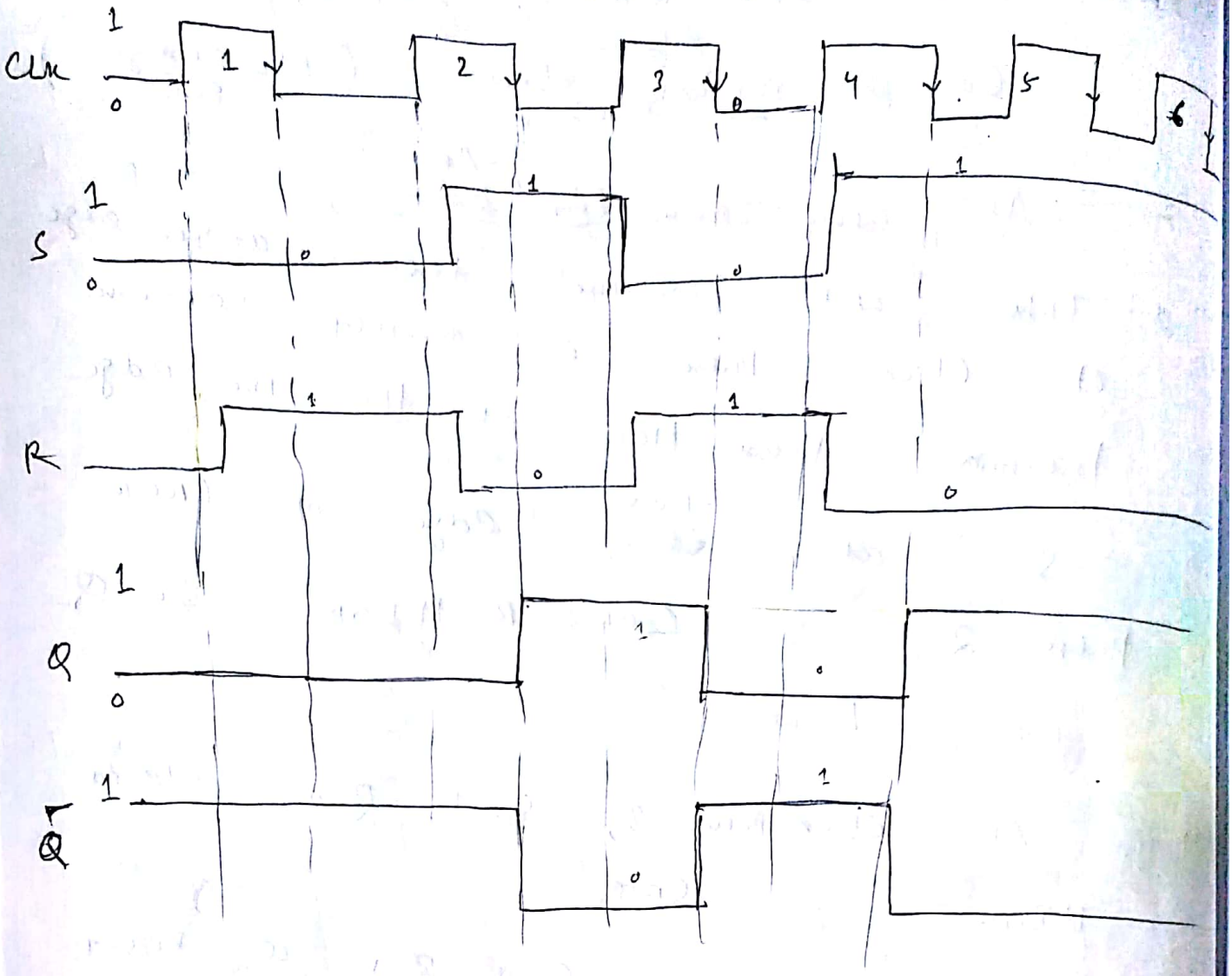
H.W 1) Determine Q & $\bar{Q}$ for the S and R I/Ps of the previous question if the R I/P is

-Ve Edge Triggered

device.



Ans 2.



The Edge Triggered D F/F

The D flip flop is useful when a single data bit (1 or 0) is to be stored. The edge triggered D F/F has one i/p terminal. It may be obtained from an S-R F/F by just putting one inverter between S and R terminal.

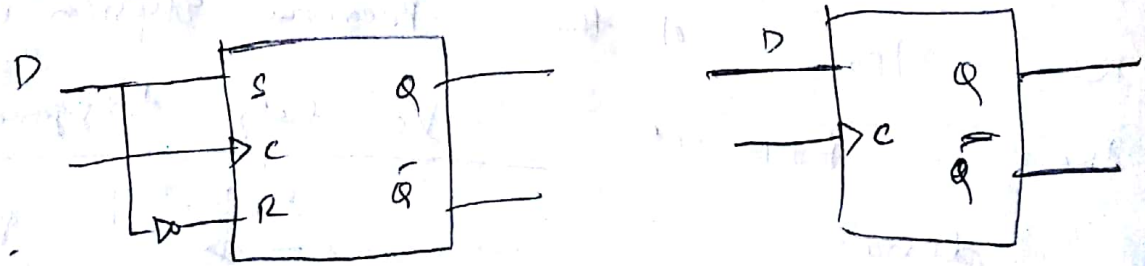
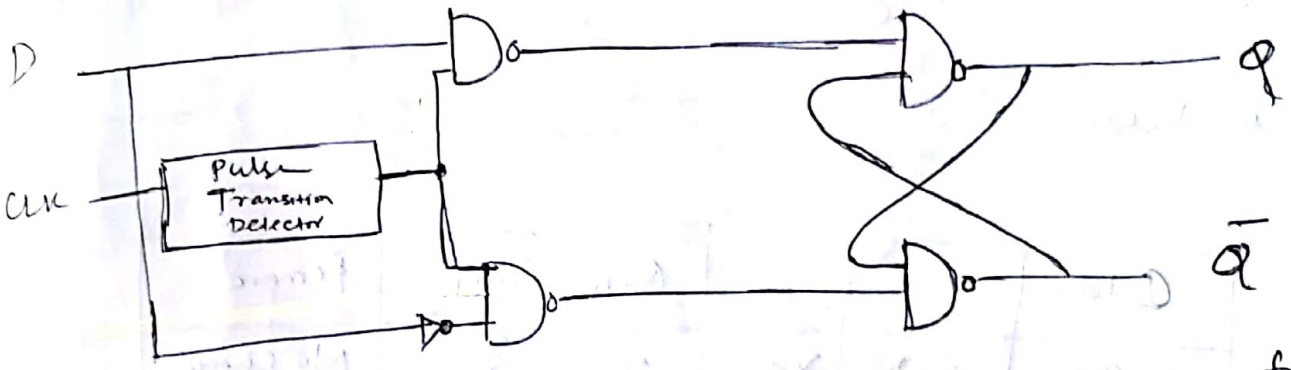


Fig 1:-
(a) D F/F from S-R F/F (b) Logic Symbol.

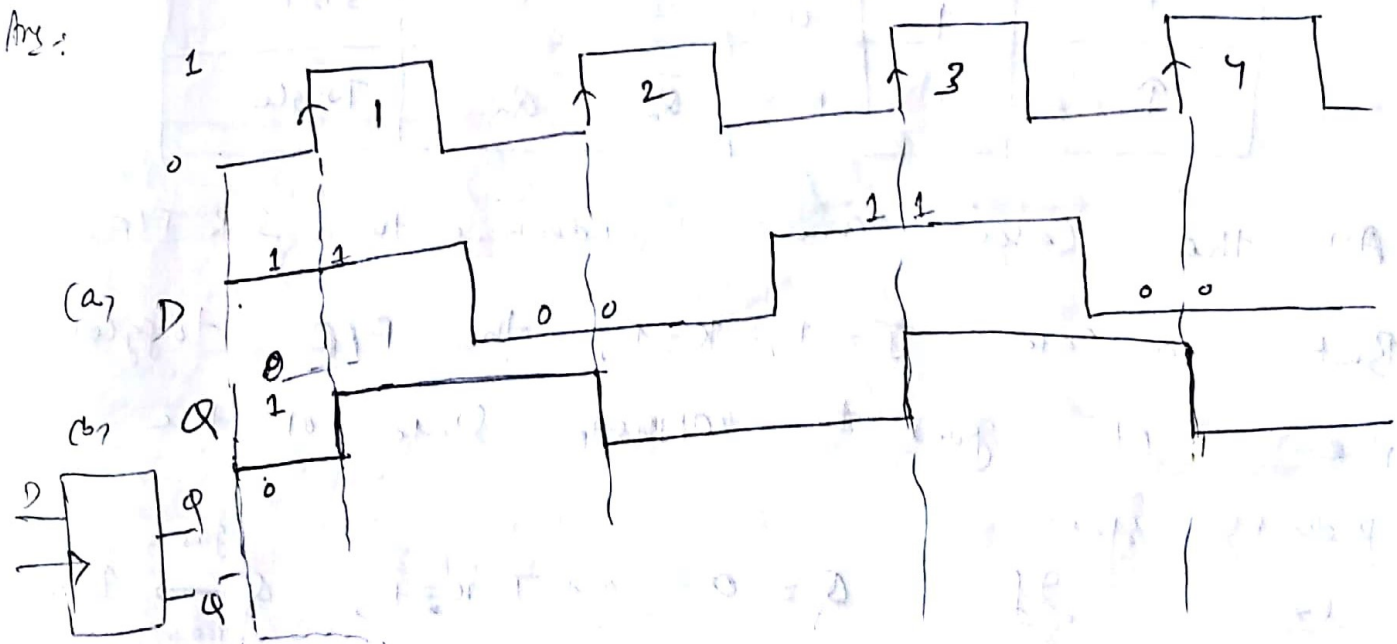
This F/F has only one synchronous control I/P called 'D' (Data), in addition to the Clock.

Clock	D	Q_{n+1}	$\bar{Q}_{n+1}$	Remarks
0	X	Q_n	$\bar{Q}_n$	No change
↑	0	0	1	RESET → '0' is stored.
↑	1	1	0	SET → '1' is stored



Q/2) Given the waveform in figure below, for the D I/P and the clock, Determine Q O/P waveform if the F/F starts out RESET.

Ans:



(+ve edge triggered F/F)

The Edge Triggered J-K F/F

The functioning of the J-K F/F is identical to that of the S-R F/F, except that it has no invalid state like S-R F/F.

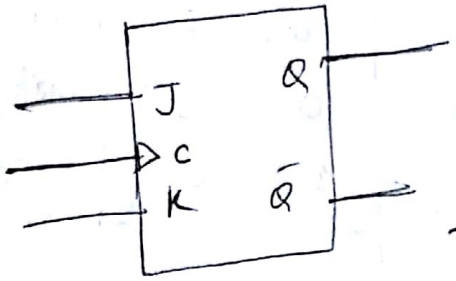


Fig 1:- Logic symbol of +ve edge triggered J-K F/F.

Truth table

Clock	J	K	Q_{n+1}	$\bar{Q}_{n+1}$	Remark
0	X	X	Q_n	$\bar{Q}_n$	No Change
↑	0	0	Q_n	$\bar{Q}_n$	No Change
↑	0	1	0	1	RESET
↑	1	0	1	0	SET
↑	1	1	$\bar{Q}_n$	Q_n	Toggle

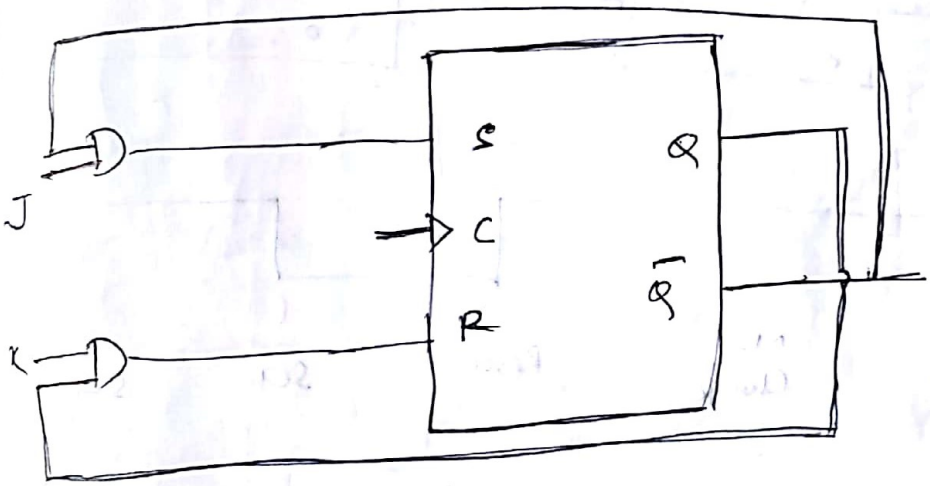
All the cases are identical to S-R F/F. But in case J = 1, K = 1, the F/F toggles i.e it goes to opposite state of the present state.

Ex:- If $Q = 0$ and $J = K = 1$, $Q \xrightarrow{\text{goes to}} 1$.

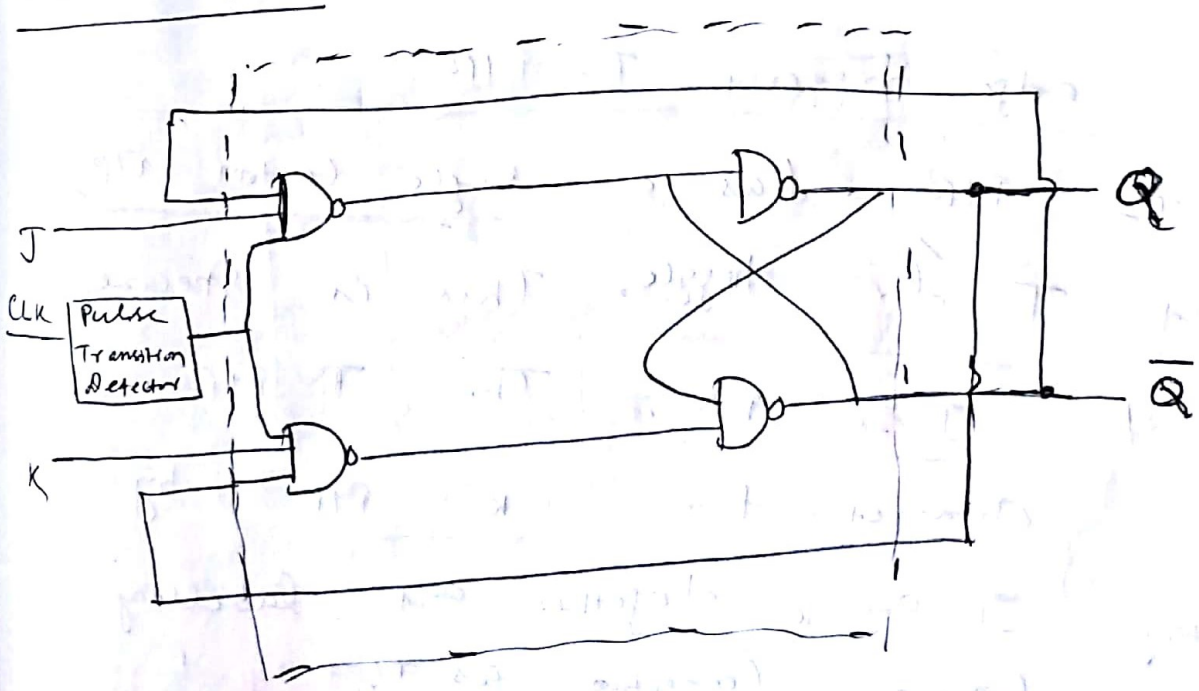
$Q = 1$, and $J = K = 1$, $Q \xrightarrow{\text{goes to}} 0$.

A -ve edge triggered J-K FF operates in the same way as a +ve edge triggered J-K FF except that the changes of the state takes place at the -ve going edge of the clock pulse. In the truth table of -ve edge triggered J-K FF the arrows point downwards.

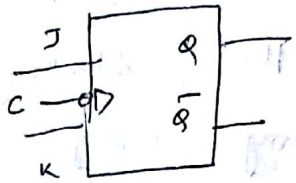
J-K FF Using SR FF



Ckt diagram SR FF



Ex:- 1) The waveforms in fig 2 are applied to J, K and clock i/ps as indicated. Determine the Q o/p, assuming that the F/F is initially RESET.



Ans:-

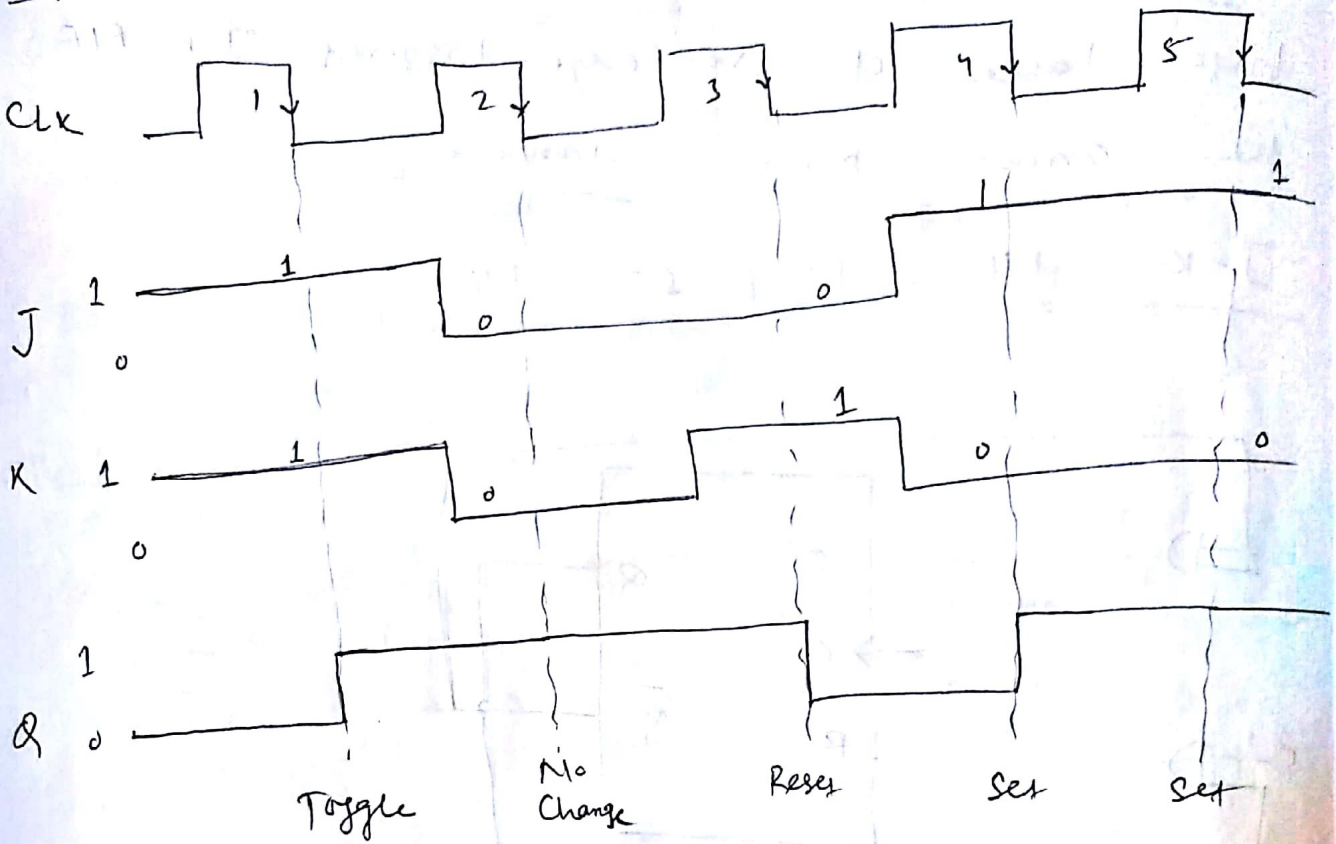
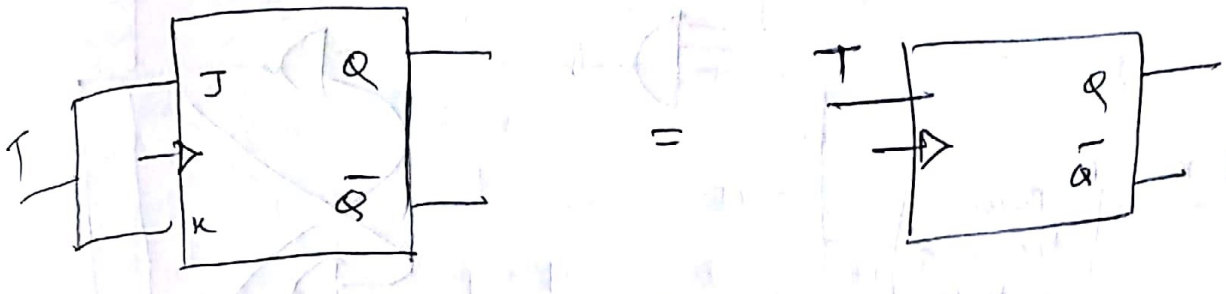


Fig 2:- o/p of -Ve edge triggered J-K F/F

The edge-triggered T-F/F

A T-F/F has a single control i/p, labelled T for toggle. This is special case of J-K F/F. The T F/F can be obtained from J-K F/F by connecting the J and K together and labelling common connection as T.



(a) T- FIR from J-K FIR (b) Logic Symbol of T FIF

→ When $T=1$, i.e $J=K=1$, This is the toggle Case of J-K F/F.

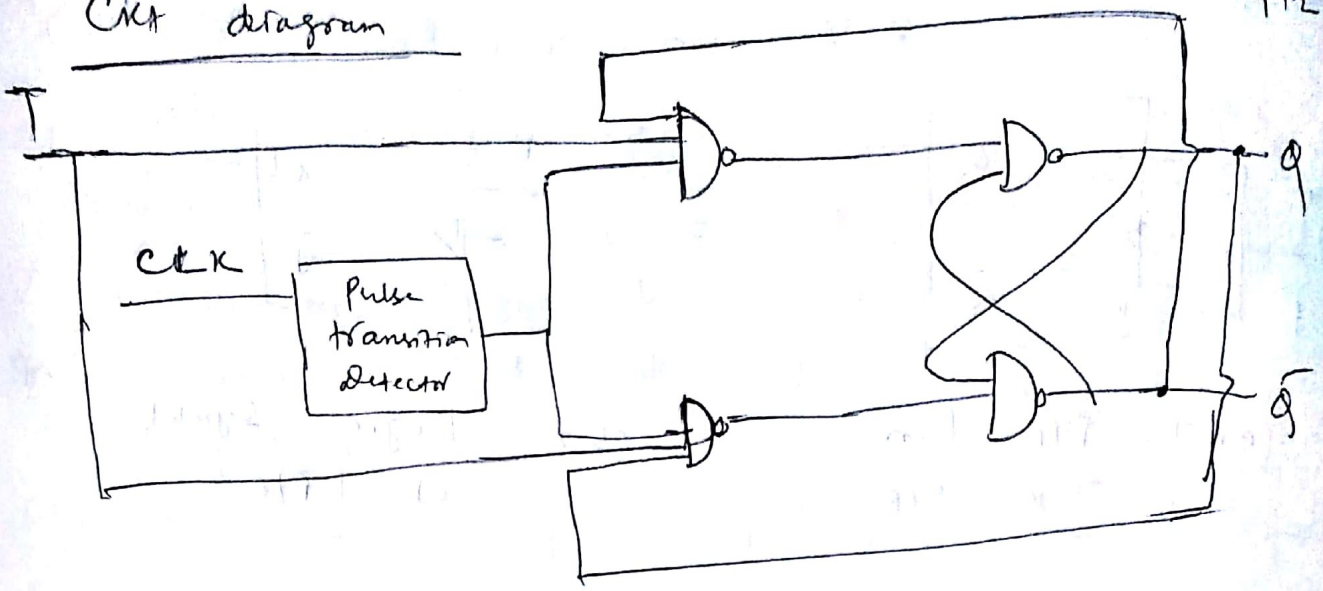
So when $T=1$, the FIR toggles on every new cycle.

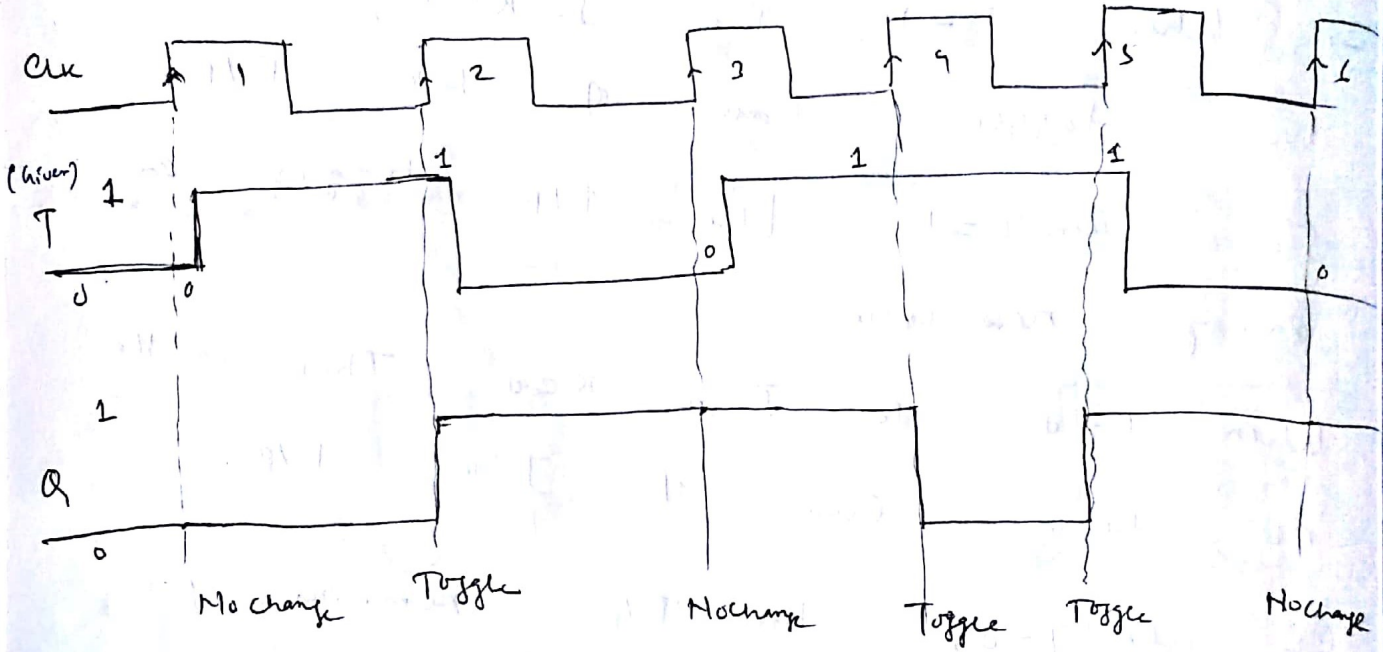
When $T=0$, i.e $J=0$, $K=0$, This is the no Change Case of J-K F/F.

So when $T=0$, the F/R remains in whatever state it was before.

Clock	T	Q_{n+1}	$\bar{Q}_{n+1}$	Remark
0	X	Q_n	$\bar{Q}_n$	No Change
↑	1	$\bar{Q}_n$	Q_n	Toggle
↑	0	Q_n	$\bar{Q}_n$	No Change

Ckt diagram





Triggering

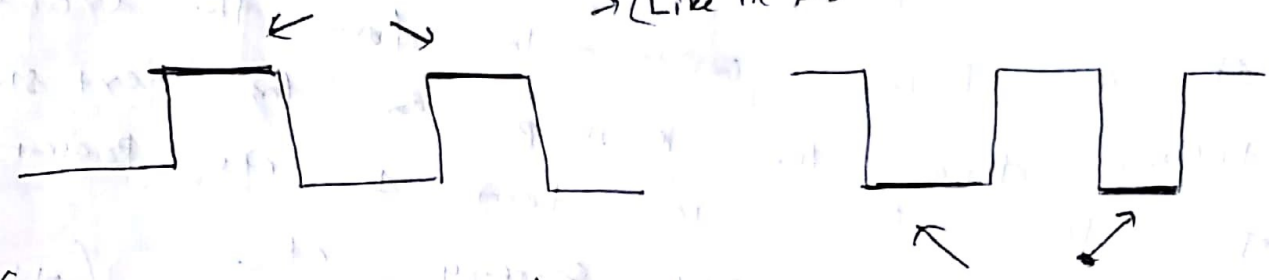
The momentary change in control i/p of a latch or F/F to switch o/p from one state to other is called trigger and the transition it causes is said to trigger the F/F

The process of applying the control signal to change the state of a F/F is called triggering. There are two types of triggering: 1) Level triggering 2) Edge triggering

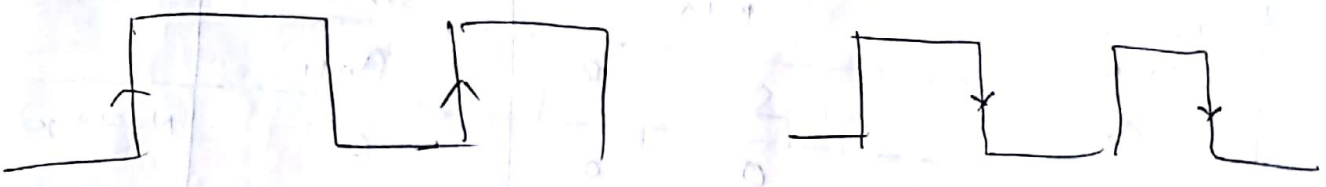
In level triggering, the I/P signal has effect the FF only when the clock is at logic 1 level.

In edge triggering, the I/P signals effect the FF only if they are present at the +ve going & -ve going edge of the clock pulse.

→ (Like in lab breadboard giving 1)



(+ve level triggering) (-ve level triggering)



(+ve edge triggering) (-ve edge triggering)

2-mark-INPUT

Latch	FF
1) Latch does not have a clock signal.	1) FF always have a clock signal
2) Latch is always level triggered	2) FF is always edge triggered.

Characteristic Equations of F/F's.

1) S-R F/F

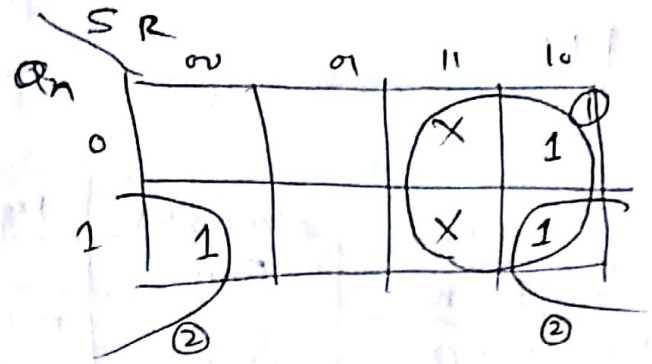
A table which lists the present state, the next state & the excitations of a F/F is called the excitation table.

To obtain the Characteristic eqn of a F/F ~~state the~~ form the excitation table, draw the K-map for next state of the f/f in terms of its present state & I/Ps and simplify it.

Present State	Inputs		Next State	Remark
Q_n	S	R	Q_{n+1}	
0	0	0	0	No change
0	0	1	0	Reset
0	1	0	1	~~Reset~~ Set
0	1	1	✱ X ✱	Invalid
1	0	0	1	No change
1	0	1	0	Reset
1	1	0	1	Set
1	1	1	✱ X ✱	Invalid

(Not required for our reference)

X → Don't Care.



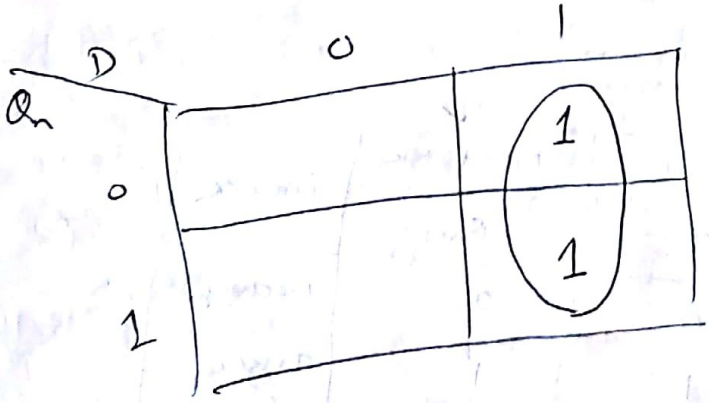
$$Q_{n+1} = S + Q_n \bar{R}$$

$$\therefore \boxed{Q_{n+1} = S + \bar{R} Q_n}$$

This is the Characteristic eqn of S-R F/F.

2) **D F/F**

Present State Q_n	inputs D	Next State Q_{n+1}	Remark
0	0	0	'0' is latched
0	1	1	'1' is latched
1	0	0	'0' is latched
1	1	1	'1' is latched

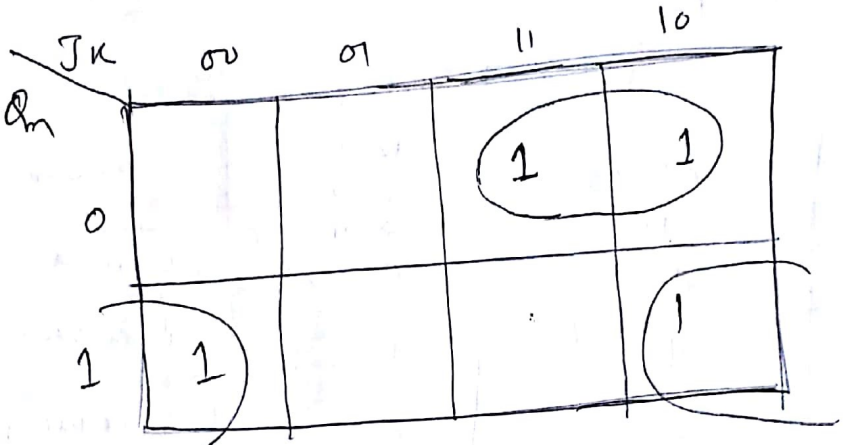


$$\boxed{Q_{n+1} = D}$$

→ Characteristic eqn of D F/F.

3) J-K F/F

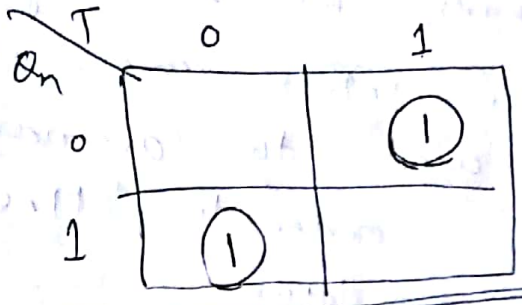
Present State	Inputs	Next State	Remark
Q_n	J K	Q_{n+1}	
0	0 0	0	No change
0	0 1	0	Reset
0	1 0	1	Set
0	1 1	1	Toggle
1	0 0	1	No change
1	0 1	0	Reset
1	1 0	1	Set
1	1 1	0	Toggle



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

4) T - F/F

Present State	Inputs	Next State	Remark
Q_n	T	Q_{n+1}	
0	0	0	No change
0	1	1	Toggle
1	0	1	No change
1	1	0	Toggle



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

Summary

	F/F		Characteristic eqn
1)	SR	$\longrightarrow$	$Q_{n+1} = S + \bar{R}Q_n$
2)	D	$\longrightarrow$	$Q_{n+1} = D$
3)	JK	$\longrightarrow$	$Q_{n+1} = J\bar{Q}_n + \bar{K}Q_n$
4)	T	$\longrightarrow$	$Q_{n+1} = T\bar{Q}_n + \bar{T}Q_n$

Asynchronous inputs

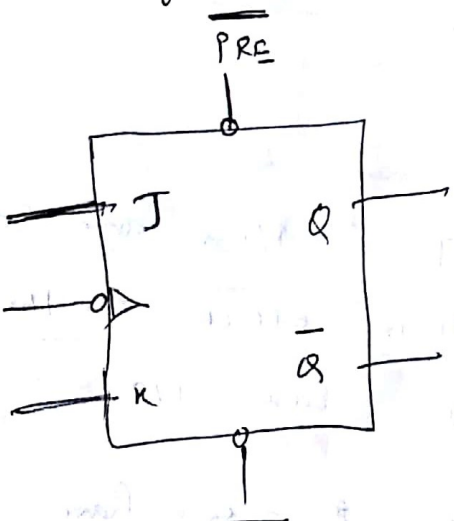
The S-R, D, J-K & T i/ps are called Synchronous i/ps, because their effect on the F/F o/p is synchronized with clock i/p.

(Most IC F/Fs also have one or more Asynchronous i/ps. These Asynchronous i/ps affect the F/F o/p independently of the Synchronous i/ps and Clock i/p. These Asynchronous i/ps can be used to SET the F/F to 1 state or RESET the F/F to the '0' state at any time regardless of the condition at the other i/ps.)

→ In other words, we can say that the asynchronous inputs are override i/ps, which can be used to override all the other i/ps in order to place the F/F in one state or other.

→ They are normally labelled as PRESET (set to 1) and CLEAR (set to 0).

→ The logic symbol and truth table of a J-K F/F with Active LOW PRESET and CLR i/p is shown in figure.



Since Active Low preset and clear i/p, when ~~when~~ $\overline{PRE} = 0$

when at $\overline{PRE}$ terminal 0 is given, it set the F/F to 1.

Similarly, when $\overline{CLR} = 0$, it set the F/F to 0.

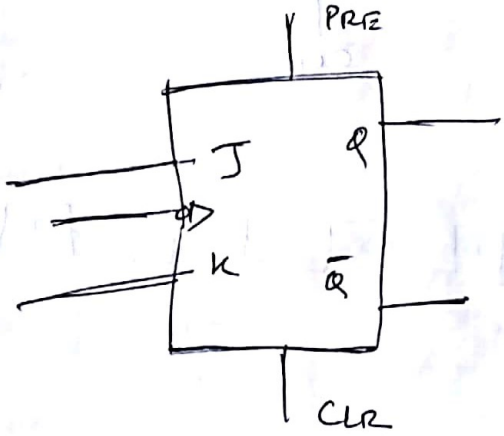
→ ~~when~~ $\overline{PRE} = 0$, $\overline{CLR} = 0$, This condition is never used, since it can result in an invalid state. [Both set and reset at a time cannot be possible]

→ When $\overline{PRE} = 1$, $\overline{CLR} = 1$, that means both asynchronous i/ps are inactive

Now the JK F/F behaves as normal F/F. In otherwords, the clocked operation can take place.

$\overline{PRE}$	$\overline{CLR}$	F/F Response
0	0	Not used.
0	1	Q = 1 (i.e F/F is set)
1	0	Q = 0 (i.e F/F is reset)
1	1	Clocked operation (i.e Normal F/F)

Active HIGH PRESET & CLEAR



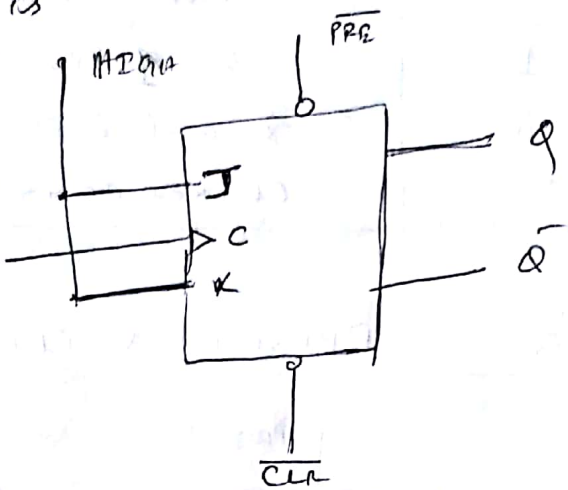
[-Ve edge triggered J-K F/F with active high preset and clear i/ps]

PRE	CLR	F/F Response
0	0	Clocked operation
0	1	Q = 0, RESET
1	0	Q = 1, SET
1	1	Not used.

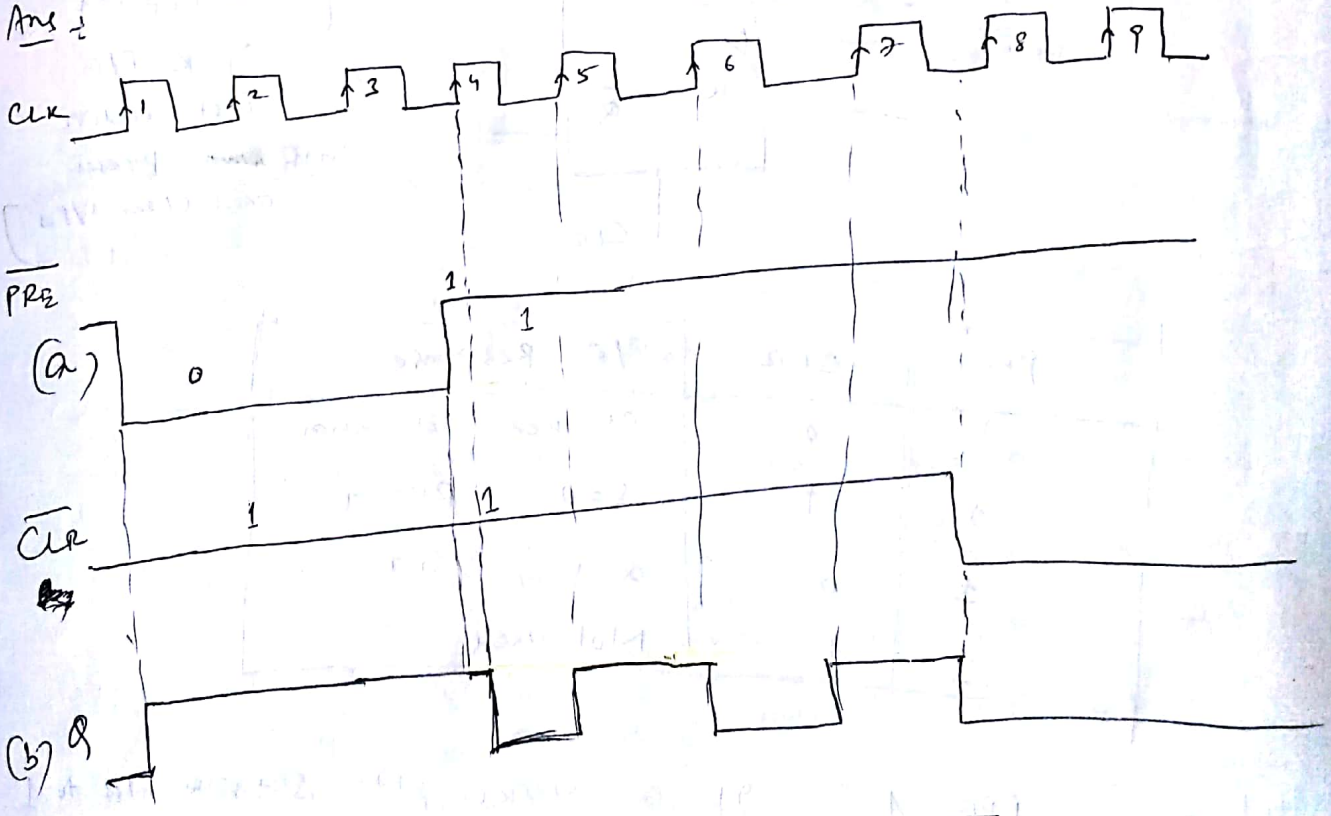
When PPE = 1, it works, it sets F/F to 1

CLR = 1, " works and sets F/F to 0.

Ex :- For the +ve edge triggered JK FF with preset and clear I/Ps in figure below, determine the Q o/p for the I/Ps shown in timing diagram in part (a) if Q is initially low.



Ans :-



1) During Clear pulse 1, 2, 3, $\overline{PRE}$ is Low, so FF is set regardless of synchronous J & K I/Ps

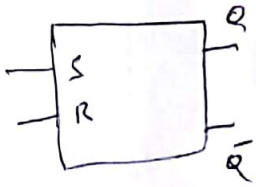
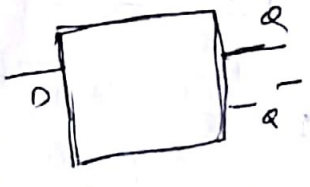
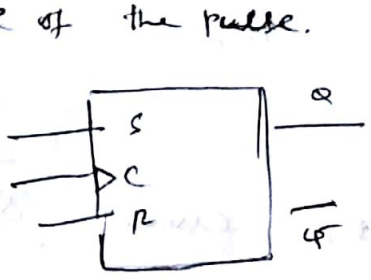
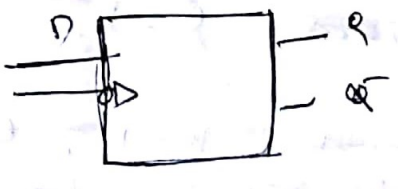
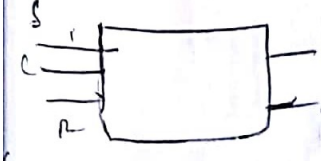
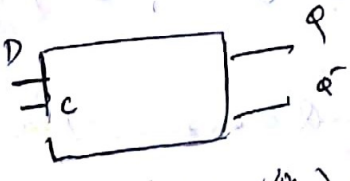
1) For Clock Pulse, 4, 5, 6, 7, since 1st J=1, K=1, T toggle operation takes place at the -ve edges.

3) Between 7 & 8 Pulse, $\overline{CLR}=0$

So the F/F is reset regardless of the Synchronous I/Ps.

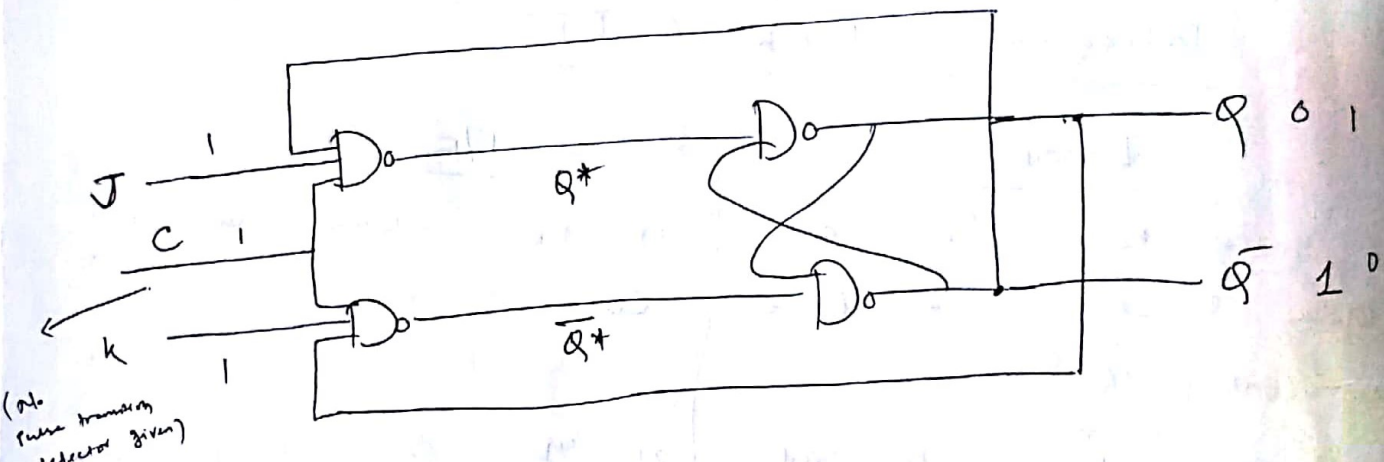
{Triggering. START}

8) Difference Latch / FF

Latch	F/F
1) Latch doesn't have a clock signal. It may have a Control I/P.	1) F/F always have a Clock signal
2) It is always level triggered.	2) It is always edge triggered.
3) The O/P Changes till the level of +ve pulse or -ve pulse.	3) The O/P Changes at the +ve edge or -ve edge of the pulse.
4)  (SR latch)  (D Latch)	4)  (+ve edge triggered 'SR' F/F)  (-ve edge triggered 'D' F/F)
 (SR latch with Control I/P)  (D latch with Control I/P)	
5) Race around problem occures in level triggered latch.	5) No race around problem occures in F/F.

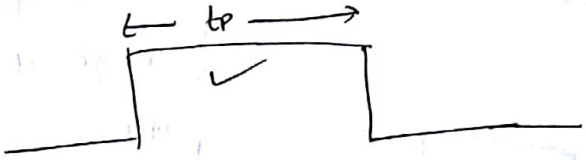
Race around Condition

For a J-K latch/FF Consider the assignment J=K=1. If the width of the clock pulse t_p is too long, the state of the FF will keep on changing from 0 to 1, 1 to 0 and so on. and at the end of the clock Pulse, its state will be uncertain. This Phenomenon is called race around condition.



When J=1, K=1, ~~Clock is~~

In the +ve half cycle of clock or control i/p.



→ Let's assume Q=0, $\bar{Q}$=1

~~J=1, K=1, C=1, $\bar{Q}$=0 $\bar{Q}$=1, ⇒ $\bar{Q}$*=0~~

~~J=1, K=1, K=1, C=1, Q=0, ⇒~~

Refering to above figure

J=1, C=1, $\bar{Q}$ = 1 ⇒ Q* = 0

K=1, C=1, Q = 0 ⇒ $\bar{Q}$* = 1

Then for a SR latch using NAND gate if i/ps are 0 & 1 o/p are 1 & 0. So $Q = 1$, $\bar{Q} = 0$

→ Now $Q \rightarrow$, $\bar{Q} \rightarrow$ } toggled from $\begin{bmatrix} 0 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 \\ 0 \end{bmatrix}$.

→ Since the Clock width t_p is small, i.e t_p is too long compared to the propagation delay (τ) of each NAND gate again this 1 & 0 i/ps are fed to the i/p.

→ $J=1$, $C=1$, $\bar{Q}=0$, $Q^* = 1$

→ $K=1$, $C=1$, $Q=1$, $\bar{Q}^* = 0$.

Now If 1 & 0 is given to SR latch, o/p is 0 & 1.

→ In this way, till the pulse is high o/p toggles from 01 → 10 → 01 → 10 - - - - .

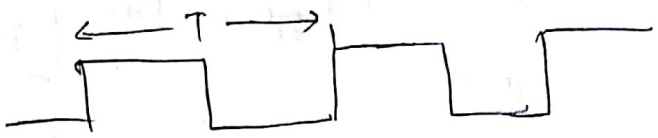
If initially $Q=1$, $\bar{Q}=0$, it toggles from 10 → 01 → 10 → 01 - - - - - .

→ So the o/p is unpredictable when $J=1$, $K=1$ & pulse width > propagation delay.

To avoid race around condition

(i) $\boxed{t_p < \tau}$ (t_p = Pulse width, τ = Propagation delay)

In other way if the clock has a period T.



$\boxed{\frac{T}{2} < \tau}$

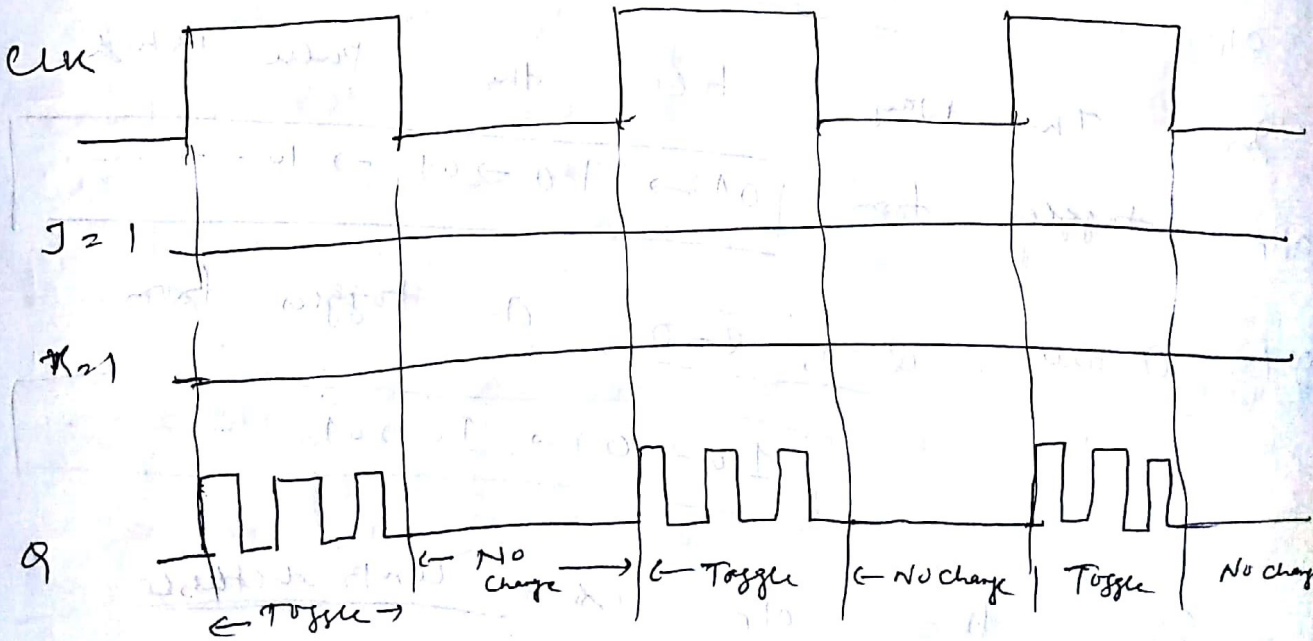
(ii) Using Edge triggering

(iii) Using Master-Slave ~~&~~ latch / FF

(Increase the delay by 2 set of FF-s)

i.e why we are using ~~to~~ edge triggered FFRs rather than latches.

Waveform of Race around Condition



Master Slave latch / (F/R)

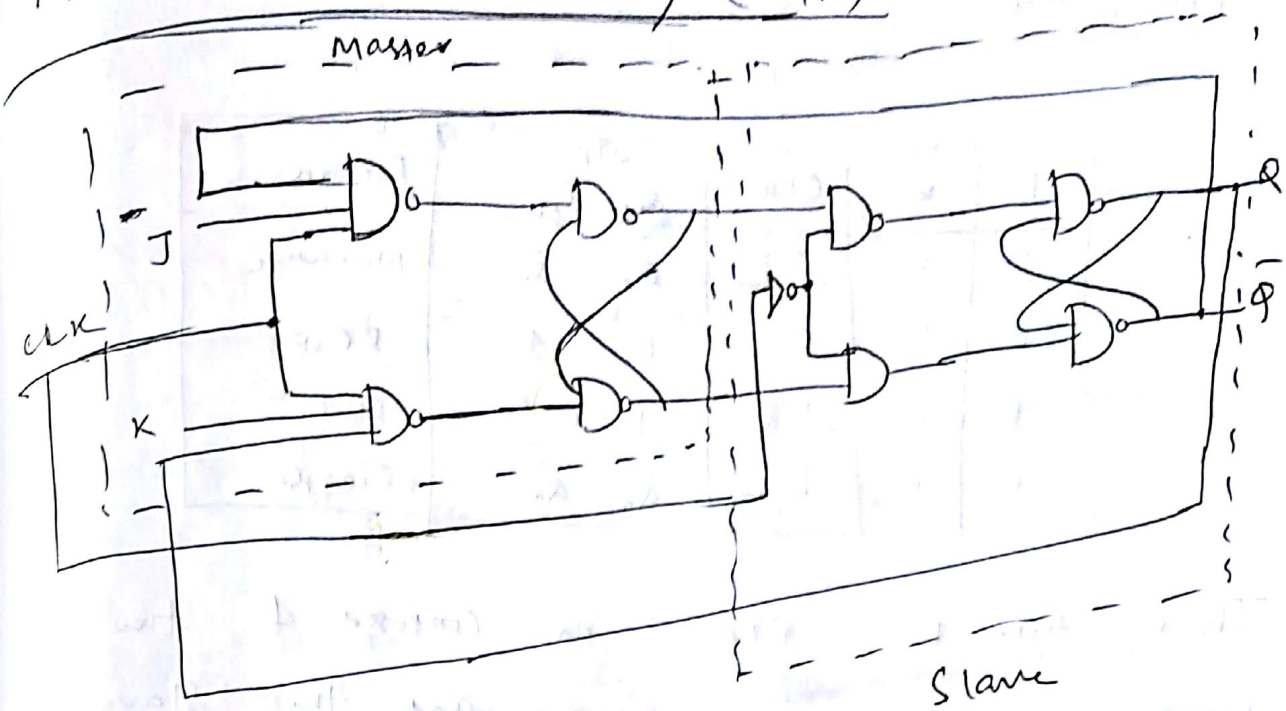


Fig 1: Master Slave J-K latch/RR

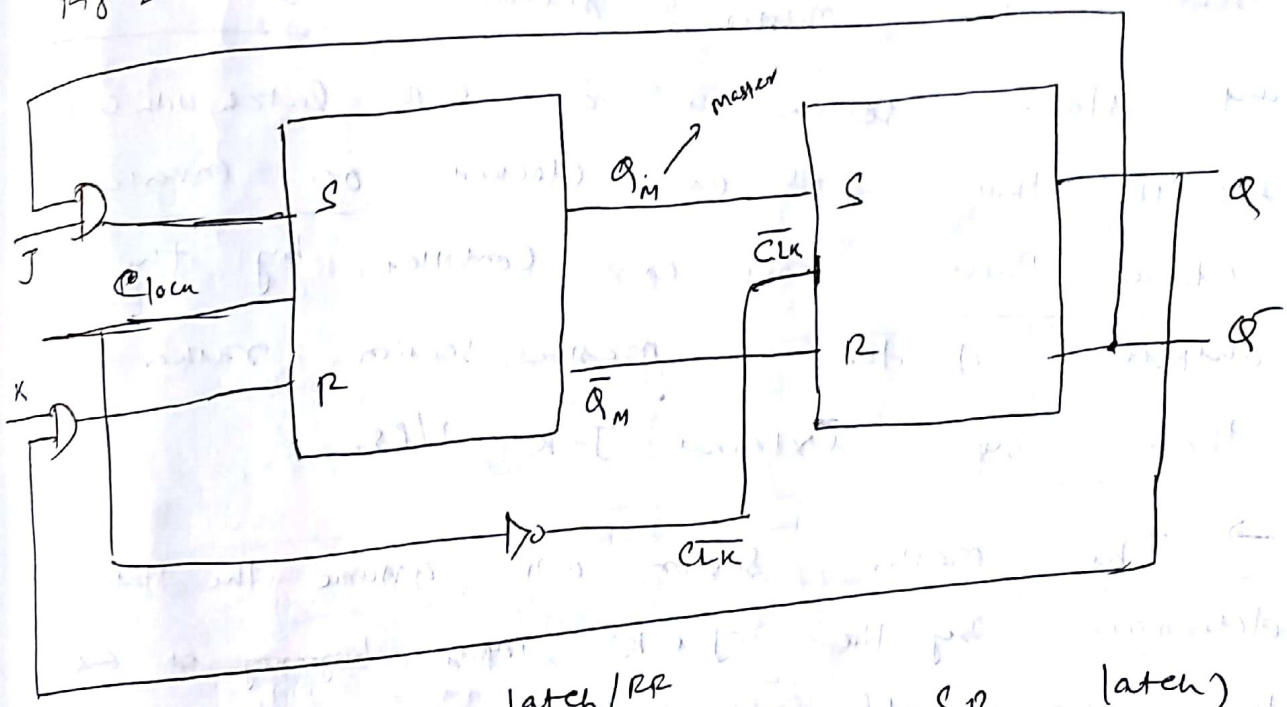


Fig 2: Master Slave latch/RR Using SR latch)

A Master Slave J-K latch / a Pulse triggered master-slave J-K F/F is shown in fig 1. The truth table operation is the same that for the edge triggered J-K F/F except for the way of its clocked

Truth table

J	K	CLK	O/P Q_{n+1}	O/P $\bar{Q}_{n+1}$	Remarks
0	0	⊓	Q_n	$\bar{Q}_n$	No change
0	1	⊓	0	1	RESET
1	0	⊓	1	0	SET
1	1	⊓	$\bar{Q}_n$	Q_n	Toggle

This type of FF is composed of two sections, the master section and the slave section. The master section is a J-K latch/FF and slave section is a S-R latch with control i/p except that it is clocked on inverted Clock Pulse and is controlled by the outputs of the master section rather than by external J-K i/ps.

→ The master section will assume the state determined by the J & K inputs, ~~beginning of~~ during the +ve half cycle of the clock pulse, and the previous i/ps Q & $\bar{Q}$.

→ The state of the master section is then transferred to the slave section.

→ At the +ve half cycle of the clock, Master is ON but the o/p Q_M & $\bar{Q}_M$

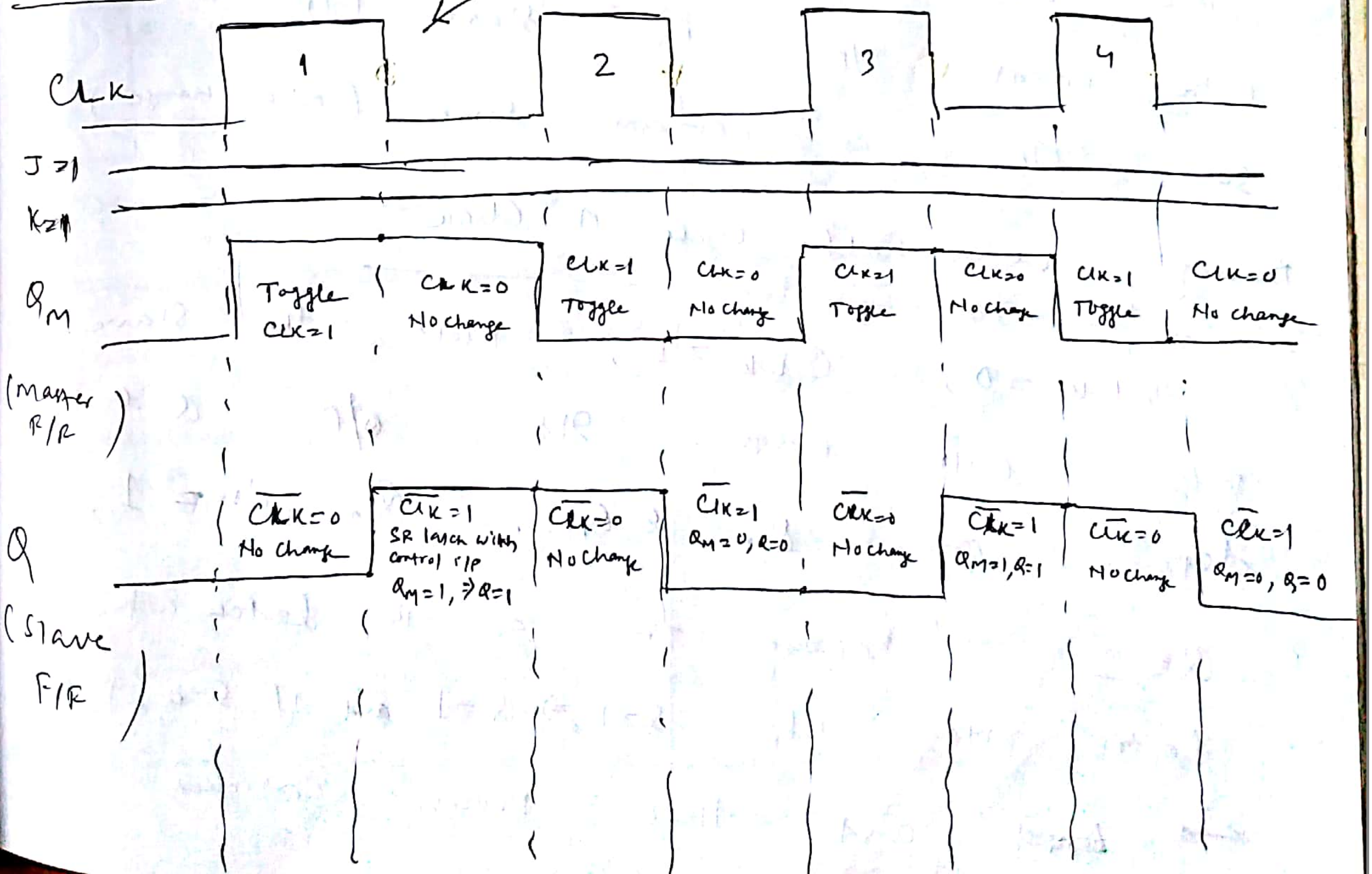
will not to transferring to Q & $\bar{Q}$ because for Slave we have no change i/p. [clock to slave is is −ve pulse as it is inverted o/p of the +ve half cycle of the clock & if clock is zero for gated SR latch we have no change)

→ So during +ve half cycle of the clock o/p is unchanged.

→ During −ve half cycle of clock, clock = 0 has no change state. but master, the slave is on. Whatever the data at Q_M & $\bar{Q}_M$ that is transferred to Q & $\bar{Q}$.

The o/p Q, is similar to −ve edge triggered F/F, although no edge triggering is given.

Waveform :-



Explanation

During CLK = 1, J = 1, K = 1

Master F/F

Assume $Q_M = 0$ Initially.

Since J=1, K=1, it will toggle, O/P is remains set till the CLK=1.

During -ve half cycle, CLK=0. So Q_M remains as it is.

And this process continues for the subsequent cycles.

Slave F/F

Assume Q=0, Initially

during +ve half cycle of the clock,

Since CLK = 1, $\overline{CLK} = 0$. So SR latch with control I/P i.e Slave F/F is in no change mode.

So O/P is remains same (No change)

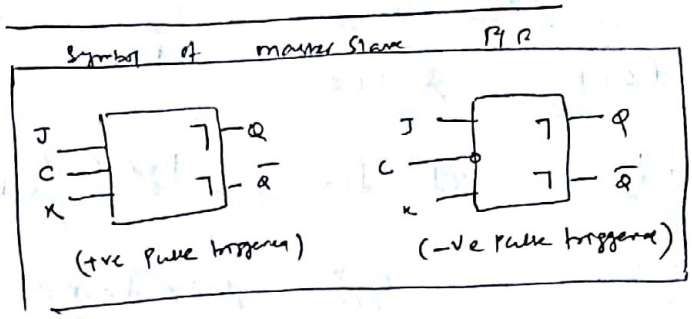
During -ve half cycle of Clock

CLK = 0, $\overline{CLK} = 1$, Now the Slave F/F will work, its O/P Q & $\bar{Q}$ depends on Q_M & $\overline{Q_M}$. Q_M is = 1,

∴ Q = 1 (because for a SR latch with control I/P, if S=1 ⇒ Q=1 and if S=0 ⇒ Q=0)

And this process continues.

The O/P Q, is observe, it is similar to -ve edge triggered J K F/F.



Counters :-

→ A digital Counter is a set of F/Fs whose states changes in response to pulses applied at the I/P to the Counter.

→ The F/Fs are interconnected such that their Combined State at any time is the binary equivalent of the total number of pulses that have occured up to that time.

→ Thus, as its name implies, a Counter is used to Count pulses.

→ Application :-

Digital watches, to create time delays, to produce non-sequential binary Counts, to generate pulse trains and to act as freq Counters etc.

→ Counters may be Asynchronous Counters or Synchronous Counters.

→ In asynchronous counters, Commonly Called

'Ripple Counters', the first F/F i/p is clocked by the external clock pulse and then each successive F/F i/p is clocked by the o/p of the preceding F/F.

In Synchronous Counters, the clock i/p is connected to all of the F/Fs so that they are clocked simultaneously.

Within each of these two categories, Counters are classified primarily by the type of sequence, the number of states or the number of F/Fs on the Counter.

Asynchronous counter	Synchronous Counters
1) In this type of Counter, F/Fs are connected in such a way that the o/p of the first F/F drives the clock for the second F/F, & o/p of 2nd drives the clock of 3rd & so on.	1) In this type of Counter there is no connection between the o/p of the first F/F and clock i/p of the next F/F and so on.
2) All the F/Fs are not clocked simultaneously	2) All the F/Fs are clocked simultaneously.
3) Design & implementation is very simple even	3) Design & implementation becomes tedious and complex as the number

for more number of states.

4) Main drawback of these counters is their low speed as the clock is propagated through a number of F/Fs before it reaches the last F/F.

of state increases.

4) Since the clock is applied to all the F/Fs simultaneously the total propagation delay is equal to the propagation delay of only one F/F. Hence they are faster.

Asynchronous Counter

'n' F/F have 2^n states.

1) 2 bit (Mod-4) Binary Counter

Fig 1 shows a 2-bit counter connected for asynchronous operation. Notice that the clock (CLK) is applied to the clock I/P (C) of only the first F/F, FF0, which

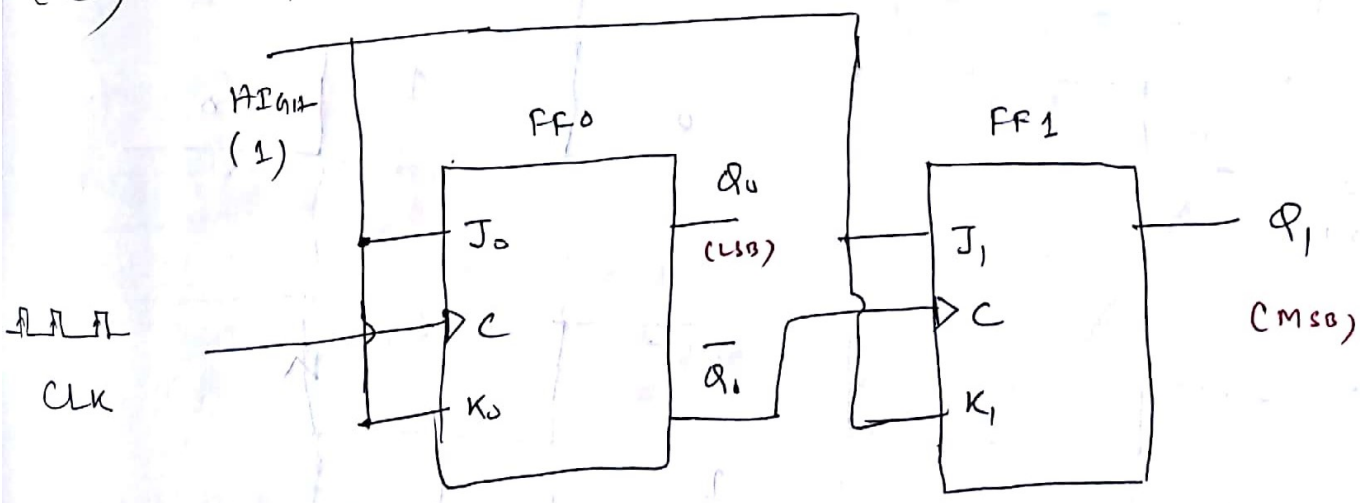


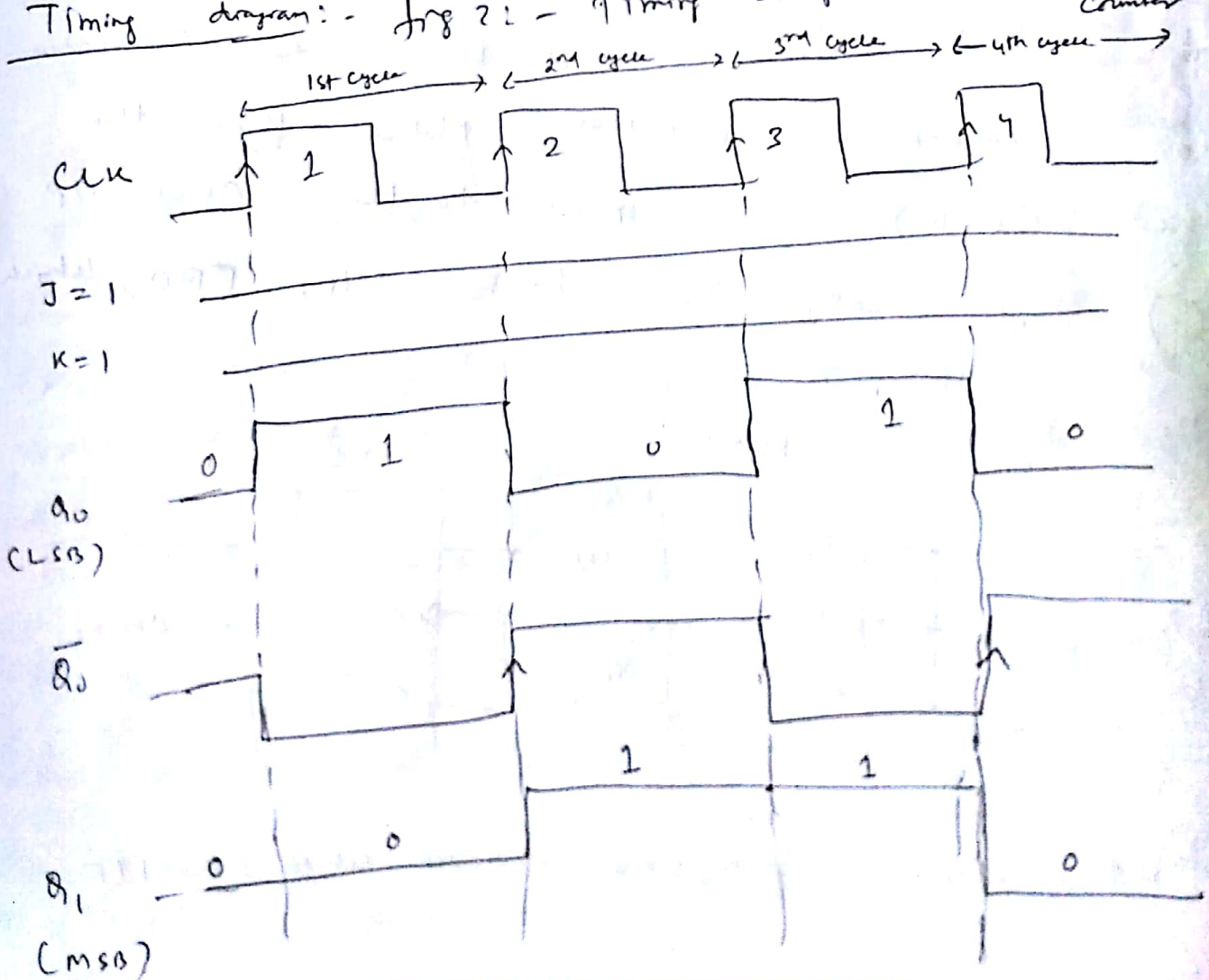
Fig: 1:- 2 bit asynchronous counter using J-K F/F.

is always the Least Significant bit (LSB).

The second F/F, FF1, is triggered by the $\bar{Q}_0$ output of FF0. FF0 changes state at the +ve going edge of each clock pulse, but FF1 changes only when triggered by a +ve going transition of the $\bar{Q}_0$ output of FF0.

Since the two FFs are not triggered simultaneously, so the Counter operation is Asynchronous.

Timing diagram:- fig 2:- Timing diagram for 2 bit binary counter



Explanation :-

J = 1, K = 1.

At the +ve edge of the clock, Q_0 will toggle. Assuming $Q_0 = 0$ initially it will be set to 1.

Since $Q_0 = 1$, $\bar{Q}_0 = 0$.

Since $\bar{Q} = 0$, Q_0 is fed as a clock to 2nd FF, 2nd FF will not change its o/p so $Q_1 = 0$.

∴ for the 1st cycle

$Q_0 = 1, \quad Q_1 = 0$

During 2nd cycle.

At the +ve edge of 2nd cycle of the clock, Q_0 will toggle & becomes 0.

So $\bar{Q}_0 = 1$, for that cycle.

Now $\bar{Q}_0$ act as clock for 2nd FF. it is a +ve edge & toggle from '0' to 1.

So $Q_0 = 0, \quad Q_1 = 1$

During 3rd Cycle

$Q_0 = 1$ (Toggled from 0 to 1) $\Rightarrow \bar{Q}_0 = 0$.

$Q_1 =$ will remain unchange. (As $\bar{Q}_0 = 0$ ~ clock = 0)

$Q_0 = 1, Q_1 = 1$

During 4th Cycle

Q_0 will toggle to = 0, $Q_0 = 0$, $\bar{Q}_0 = 1$, Q_1 will toggle.

form 1 to 0. ∴ $Q_0 = 0, Q_1 = 0$

-5

Clock Pulse	MSB Q_1	LSB Q_0
Initially	0	0
1	0	1
2	1	0
3	1	1
4 (recycle)	0	0

↓ up Counter.

Imp

Since it has 4 states 00, 01, 10, 11 it is mod-4 Counter

→ Since it goes through a binary sequence, the Counter in fig 1 is a binary Counter. It actualy Counts the number of clock pulses up to 3 and on the 4th pulse it recycles to its original state ($Q_1Q_0 = 00$). The term recycle refers to the transition of the Counter from its final state back to its original state.

Note:- i) Q_0 represents LSB ✓
Q_1 " MSB. ✓

2) (i) +ve edge triggered, o/p from $\bar{Q}_0$ → up Counter. ✓
to clock

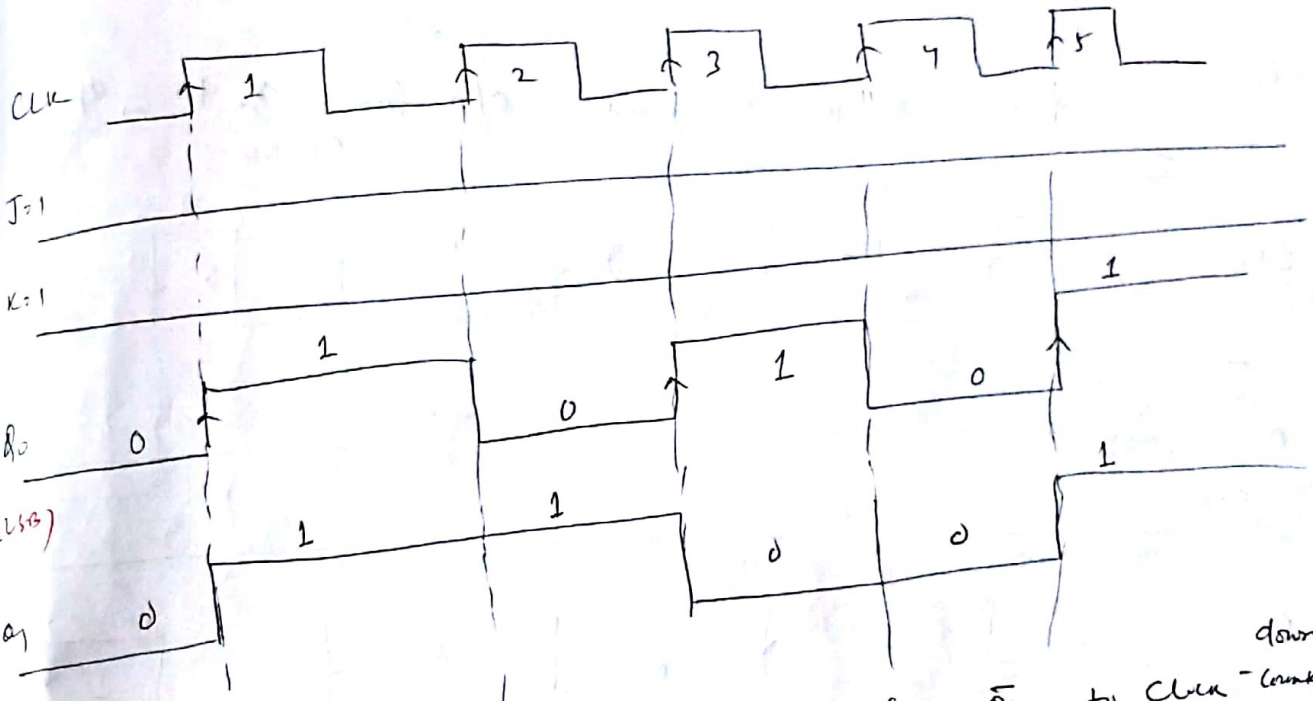
(ii) +ve edge " , o/p from Q_0 → down Counter ✓
to clock

(iii) -ve edge triggered, o/p from $\bar{Q}_0$ → down Counter ✓
to clock

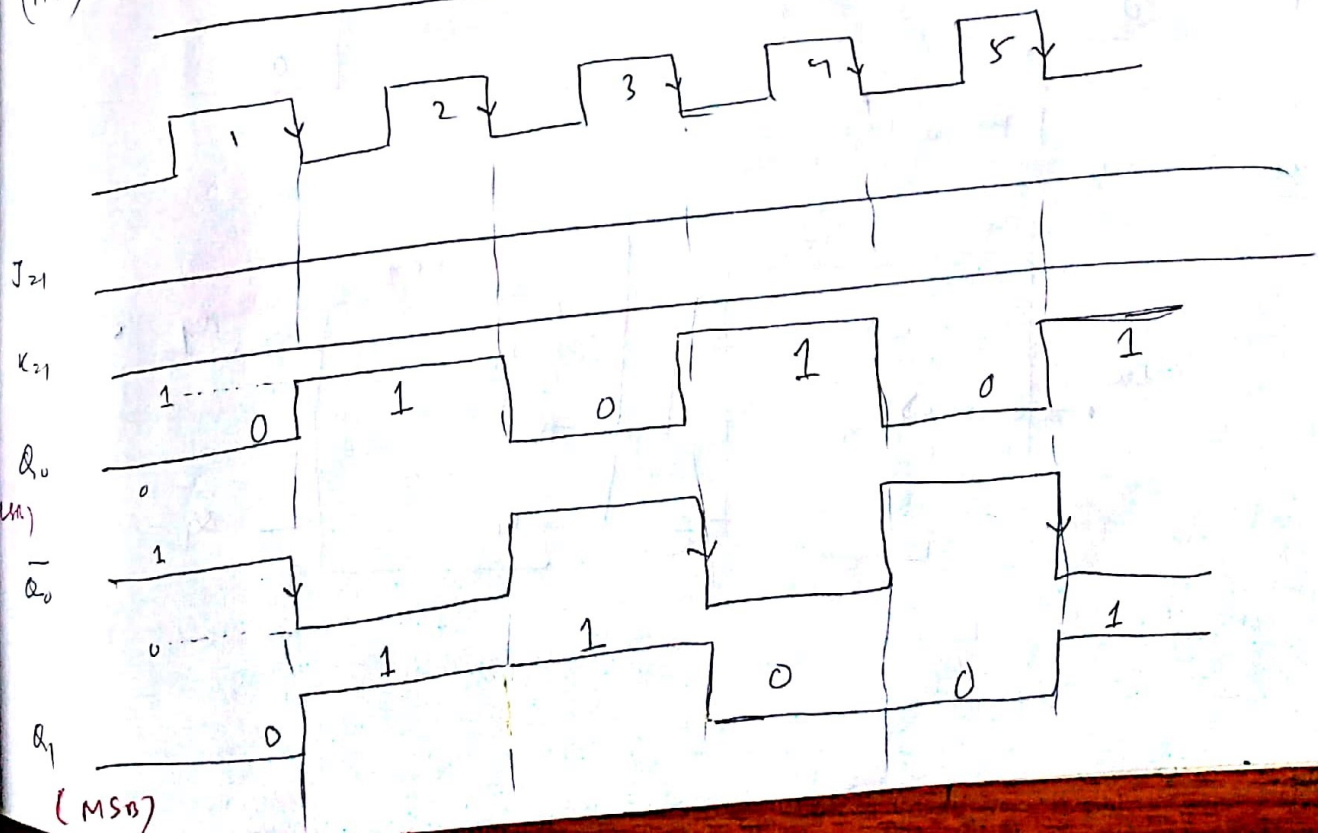
(iv) -ve edge triggered, o/p from Q_0 → up Counter. ✓
to clock

3/ → [Remember the first one, rest can be found out by making a single change " up counter → down counter or vice versa.]

→ Ex. (i) +ve edge triggered, o/p from Q_0 → down counter to the clock.



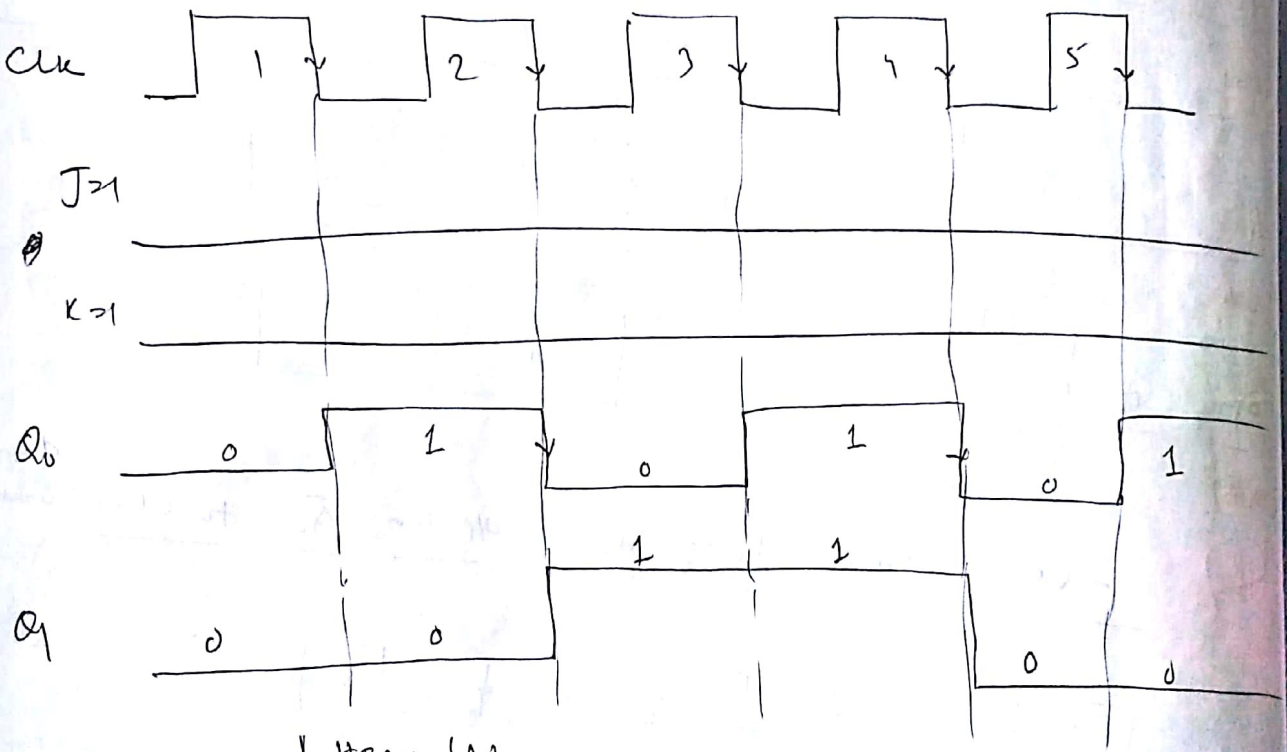
(ii) -ve edge triggered o/p from $\bar{Q}_0$ to clock - down counter.



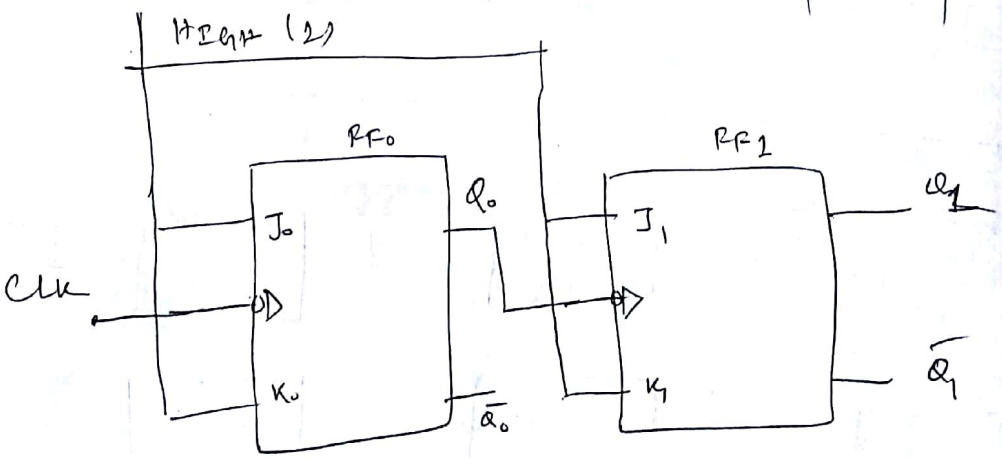
Clock Pulse	MSB Q_1	LSB Q_0
Initially	0	0
1	1	1
2	1	0
3	0	1
4	0	0

down Counter.

(IV) -ve edge triggered o/p from Q_0 to clock → up counter

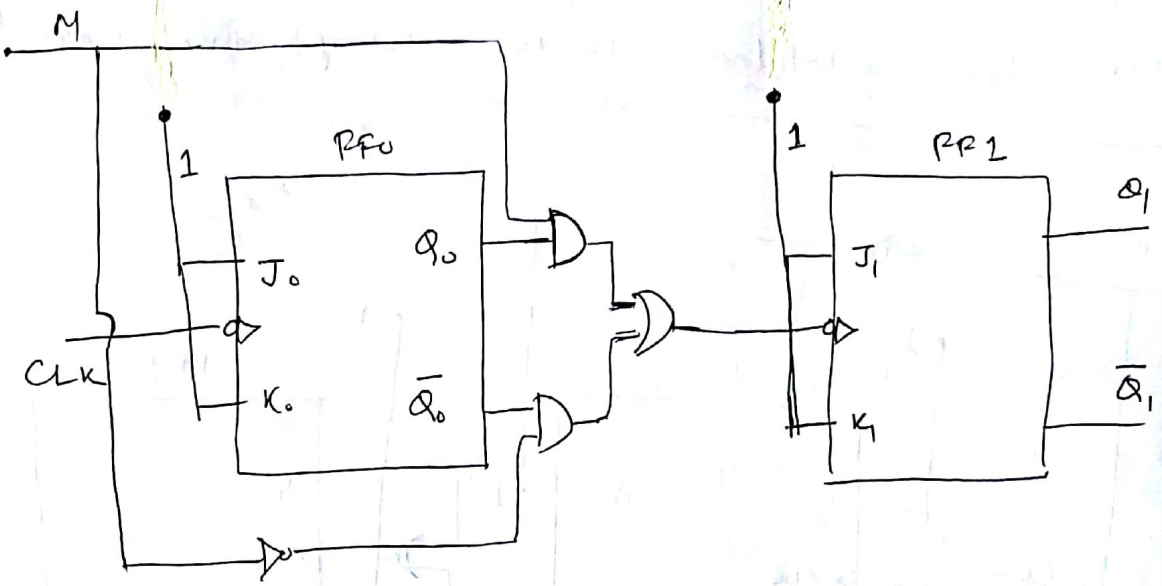


CKT :-



Two bit updown Counter using -ve Edge triggered F/Fs

As the name indicates a up-down Counter is a Counter which can Count both in upward & downward directions. An up-down Counter is also called a forward / backward Counter or a bidirectional Counter. So, a Control Signal or a mode signal M is required to choose the direction of Count.



→ When M=1, for upcounting, Q_0 is transmitted to Clock of FF1 and when M=0 for down Counting, $\bar{Q}_0$ is transmitted to Clock of FF1.

→ This is achieved by using two AND gates & one OR gate as shown in figure.

→ The external clock is applied to FF-0.

Clock signal to FF1, $= Q_0 \cdot M + \bar{Q}_0 \bar{M}$ —— (1)

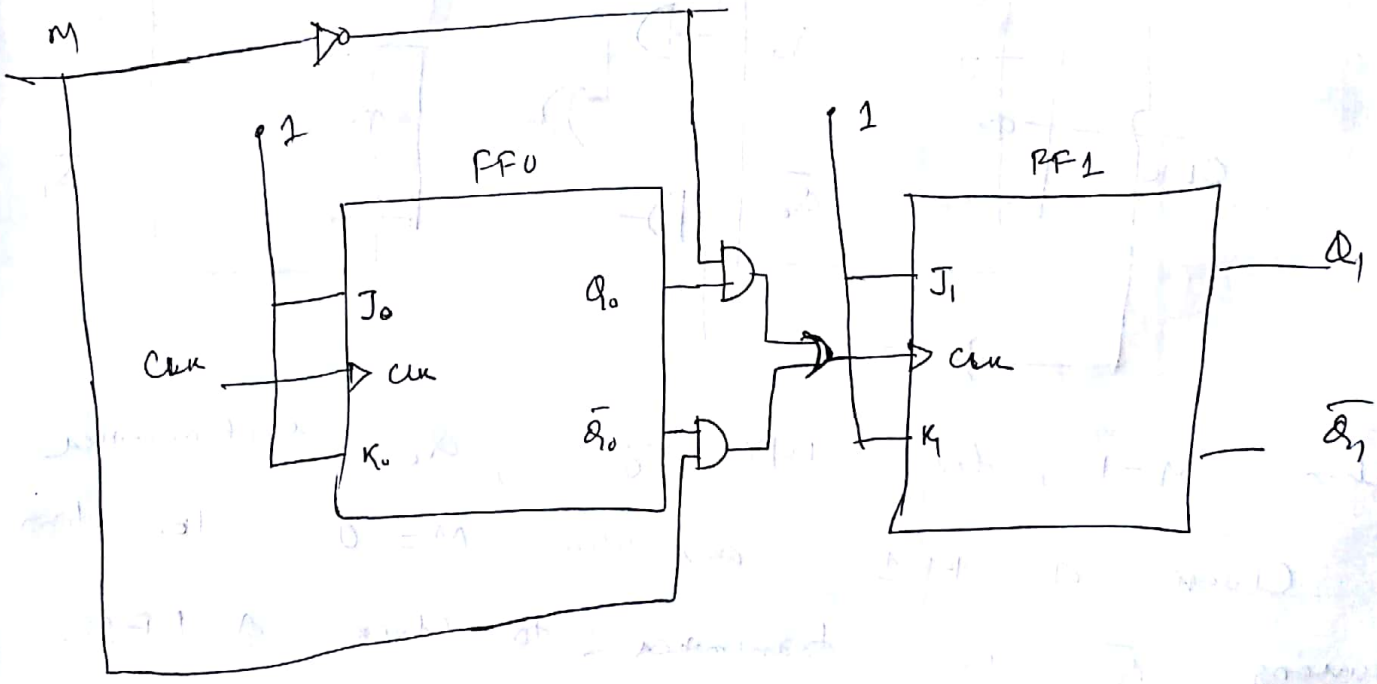
When M = 1, For FF1, Clock signal is from (using eqn (1)), $Q_0 + 0 = Q_0$

i.e -ve edge triggered, Clock from Q_0 = upcounter.

When, M = 0, For FF1, Clock signal is from (using eqn (1)), $0 + \bar{Q}_0 \cdot 1 = \bar{Q}_0$

i.e -ve edge triggered & Clock from $\bar{Q}_0$ = down counter.

Two bit updown counter using +ve edge triggered F/F



Clock to FF1, $Q_0 \bar{M} + \bar{Q}_0 M$

When M = 0, Clock to FF1 is $Q_0 \cdot 1 + 0 = Q_0$

+Ve edge triggered Clock from Q_0 = down counter.

When M = 1, Clock to FF1 is $Q_0 \cdot 0 + \bar{Q}_0 \cdot 1 = \bar{Q}_0$

+ve edge triggered, Clock from $\bar{Q}_0$ → up counter.

Design of (Mod – 6) asynchronous Counter

A Mod-6 Counter has 6 stable states 000, 001, 010, 011, 100 & 101. When the sixth Clock pulse is applied, it should immediately reset to 000 because of the feedback provided.

[Note:- The modulus of a Counter is the number of unique states that the Counter will sequence through. The Maxm possible number of states (Maxm modulus) of a Counter is 2^n, where n is the number of FFs in the Counter.]

→ It is a 'divided-by-6 Counter', in the sense that it divides the I/P clock frequency by 6.

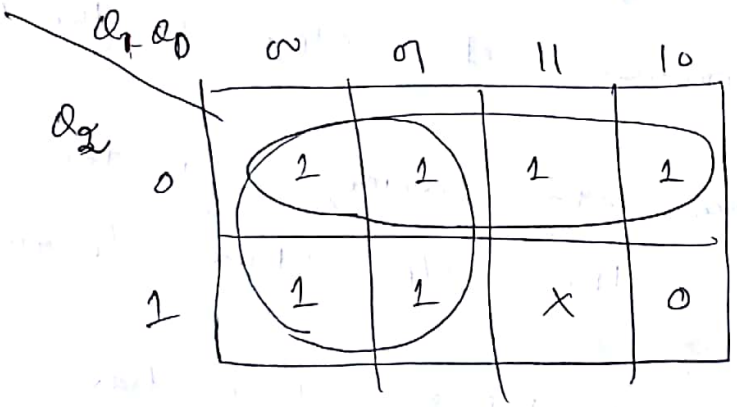
→ It requires 3 FFs, which has 8 possible states, out of which only six are utilized and the remaining two states 110 & 111, are invalid.

→ If initially the Counter is in 000 state, then after the first clock pulse it goes to 001, after the second clock pulse, it goes to 010 & so on. After the sixth

Clock pulse, it goes to 000.

Truth table

Q_2	Q_1	Q_0	R
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	X

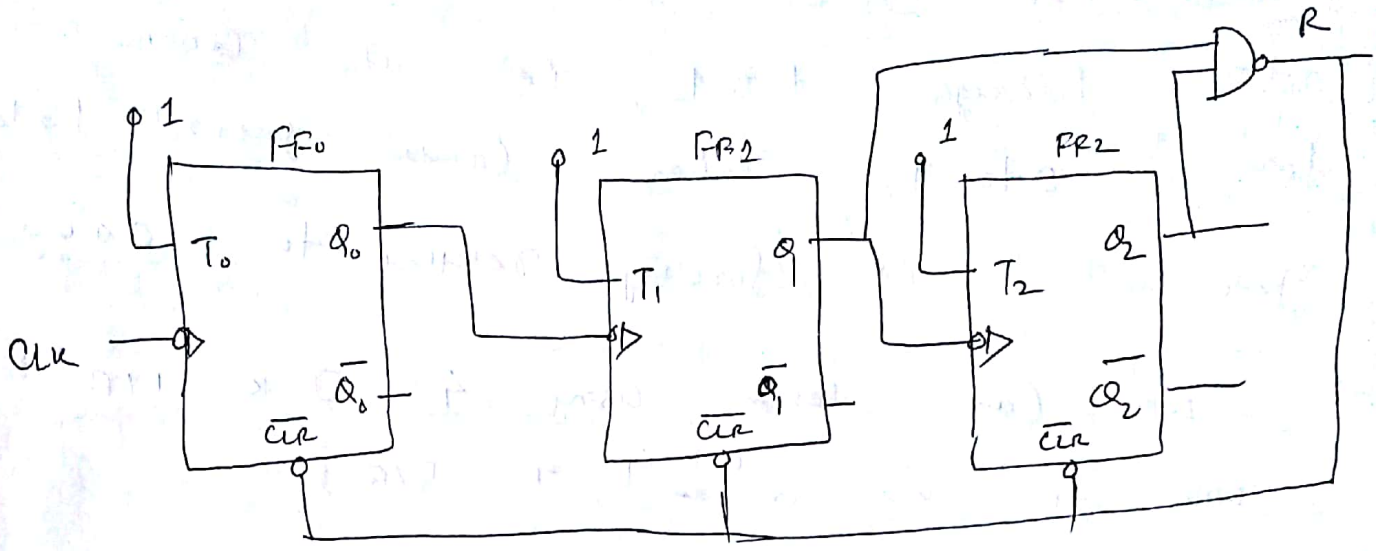


$$R = \bar{Q}_1 + \bar{Q}_2$$

$$= \overline{Q_1 Q_2}$$

→ A J-K FF with J = 1 = K is same as a T FF. So a mod-6 counter can be designed using 3 T FFs, & a up counter can be designed with -ve edge triggered

& Q_0 is used as clock for FF 1.



∴ from 000 to 101, $R = 1$, $\overline{CLR} = 1$

So it behaves a upcounter.

When 110 case arises, $\overline{Q_1 Q_2}$

$(Q_2 Q_1 Q_0)$ $= \overline{1 \cdot 1}$

$= 0$,

∴ $\overline{CLR} = 0$, All the FFs are clear or RESET.

O/Ps

So again the cycle is repeated i.e 000, 001, 010, 011, 100, 101 & so on.

(Mod-10) / (Decade) Asynchronous Counter

→ It is also called BCD Counter or a divided-by-10 Counter.

→ It requires 4 FFs. So we have 16 possible states, out of which

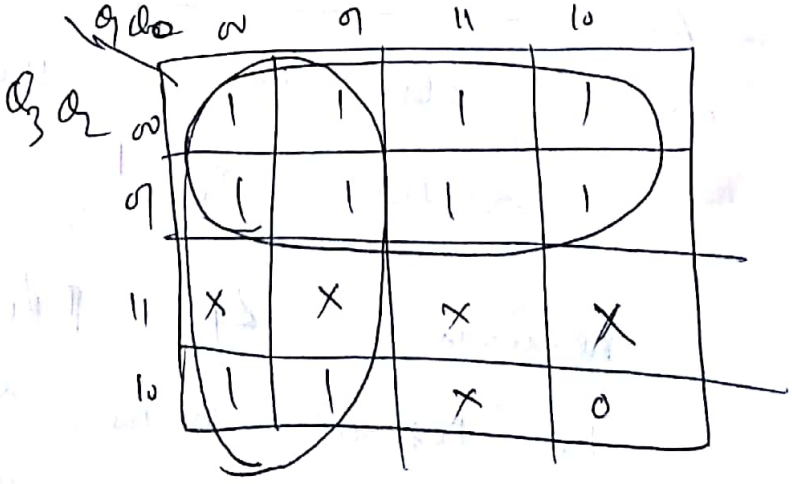
ten are valid & remaining six are invalid. The counter has ten stable states, 0000 through 1001, i.e (7 counts from 0 to 9. When counter goes to 1010 state it is forcibly resetted to 0000.

→ We can design using 4 J-K FIR with J=1, K=1 or with 4, T FIRs.

Q_3	Q_2	Q_1	Q_0	R
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

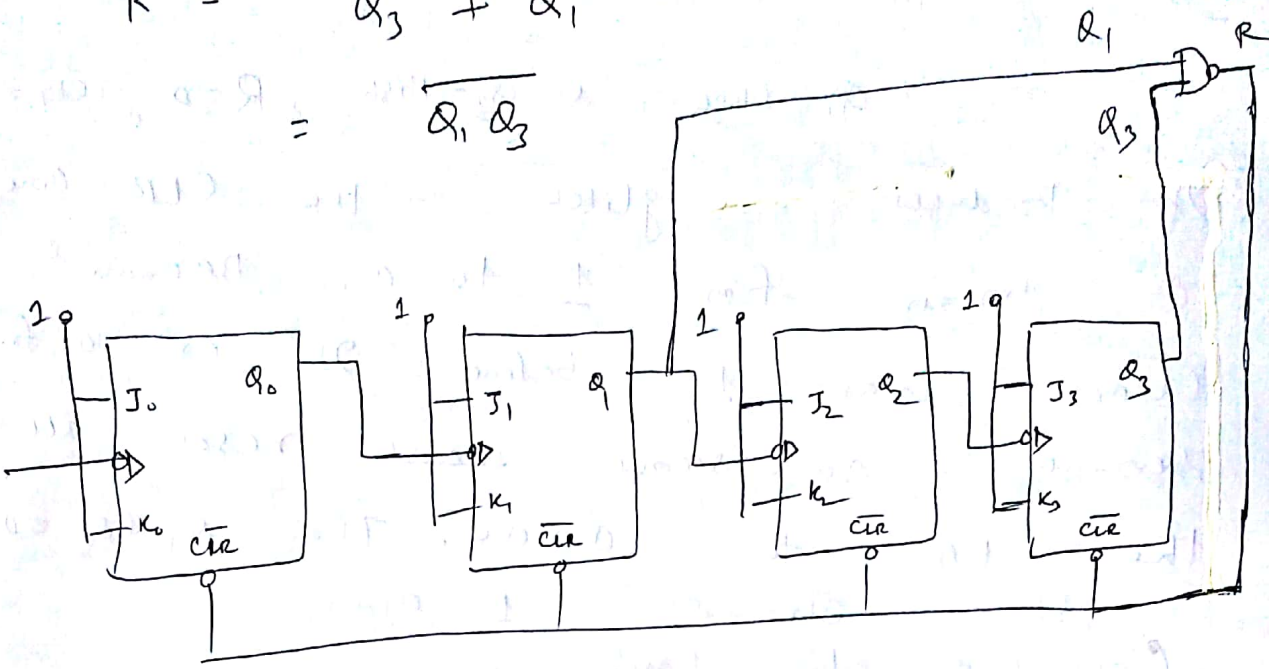
Note:-

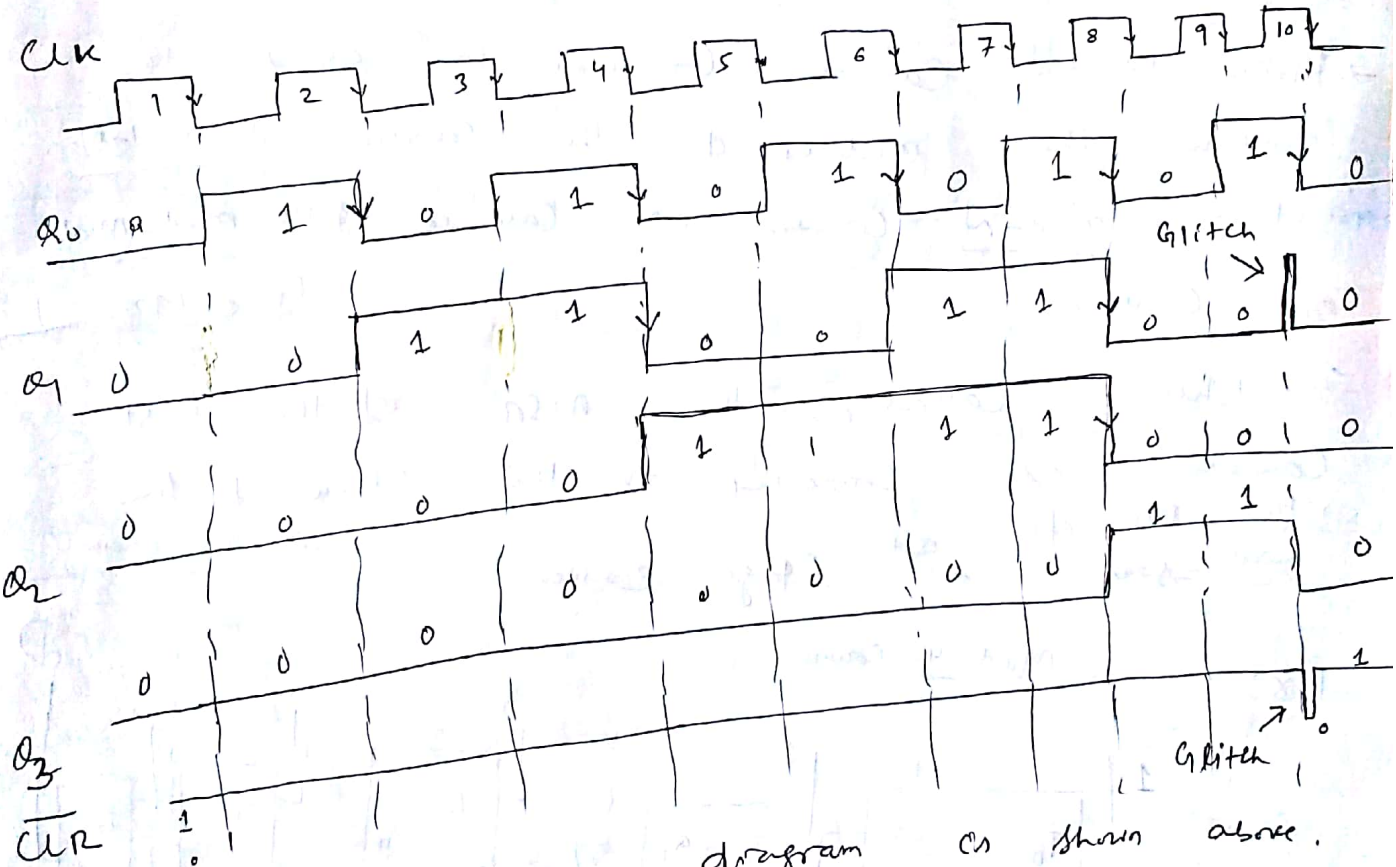
1) Any arbitrary start and arbitrary end counter e.g start at 2 and end at 12 Design the truth table accordingly. What ever the eqn comes connect to $\overline{CLR}$, accordingly.



$$R = \overline{Q_3} + \overline{Q_1}$$

$$= \overline{Q_1 Q_3}$$





The resulting timing diagram is shown above. Notice that there is a glitch on the Q_1 — waveform. The reason for this glitch is that Q_1 must first go to HIGH (1) (Because at 1010, Q_1 = High) before the reset is made

Similarly, for a seventh nano second,

Q_1 = High & Q_3 = High, $R = 0$ $\Rightarrow \overline{CLR} = 0$

It produces a glitch on the CLR line of transition from 1 to 0. Because $\overline{Clear}$ was 1 before. It is 0 for seventh nano second reset all the FF to 0 0 0 0. Then Q_1 & $Q_3 = 0$ so $\overline{CLR}$ become 1 again.

Cascading of Ripple Counters

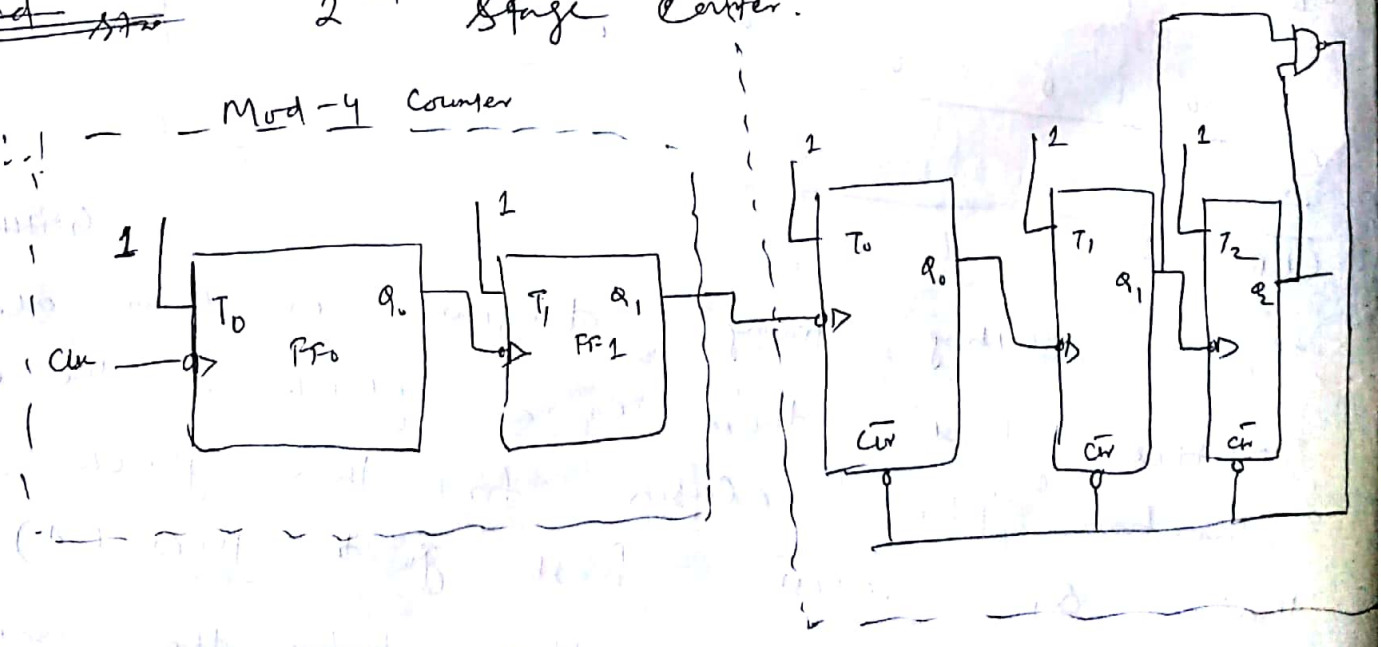
→ Ripple Counters can be connected in cascade to increase the modulus of the counter. A mod-M and a mod-N counter on cascade give mod-MN (i.e M×N) Counter.

→ While Cascading, the MSB of the first Counter is connected to the clock of the 1st FF of the 2nd Stage counter.

Mod 6 Counter

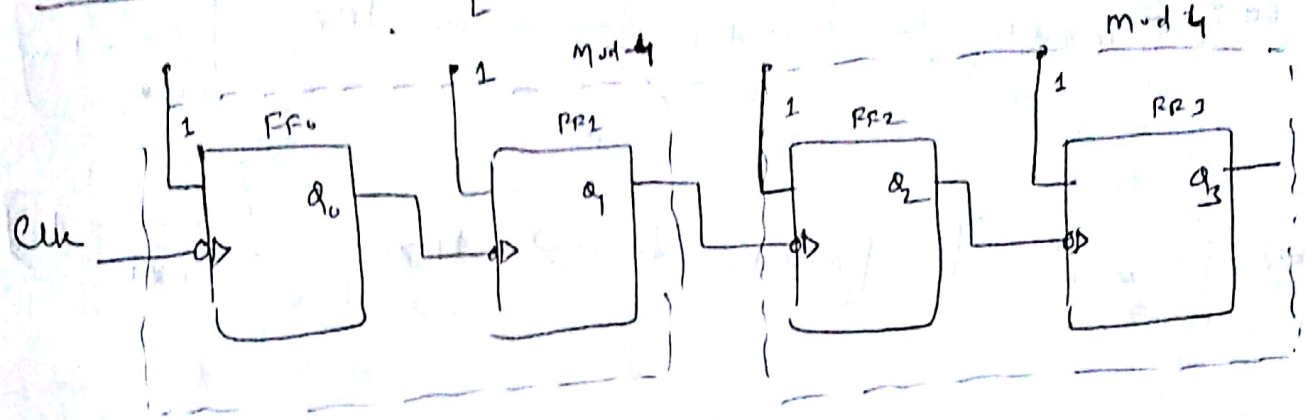
Mod-4 Counter

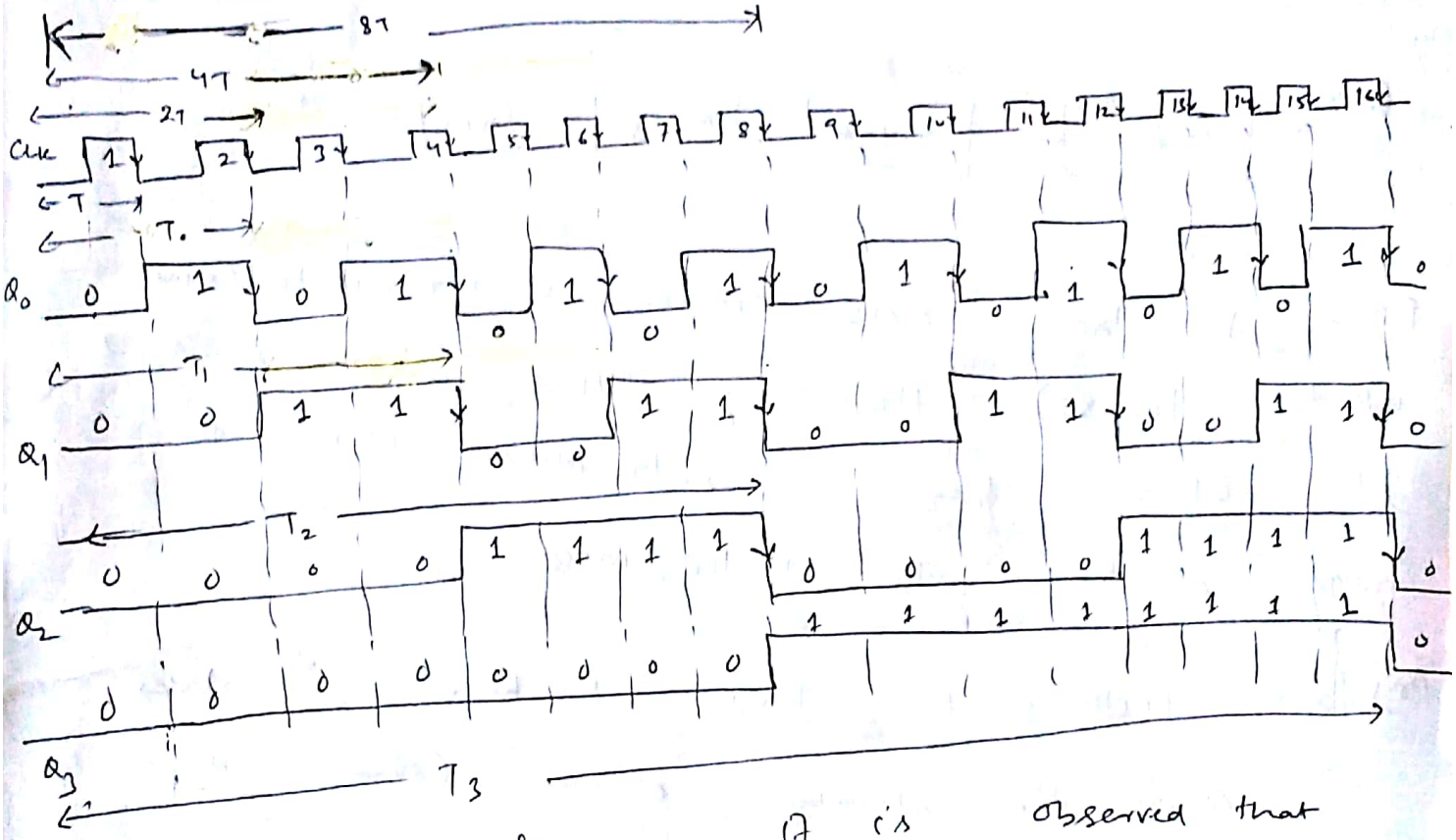
Ex:-



Mod 4 × Mod 6 = Mod-24 Counter.

Let's verify. [Mod-4 × Mod-4 = Mod-16]





From the waveform, it is observed that

frequency of Q_0 waveform = $\frac{1}{2}$ of that clock freq (f_0)

Because at 2 cycle of clock = 1 cycle of Q_0.

freq of $Q_0 \rightarrow \frac{f_0}{2}$, $Q_2 \rightarrow \frac{f_0}{8}$

freq of $Q_1 \rightarrow \frac{f_0}{4}$, $Q_3 \rightarrow \frac{f_0}{16}$

$T_0 = 2T$, $T_1 = 4T$.

$T_2 = 8T$, $T_3 = 16T$

$T \rightarrow$ Time period of clock.

Ans:

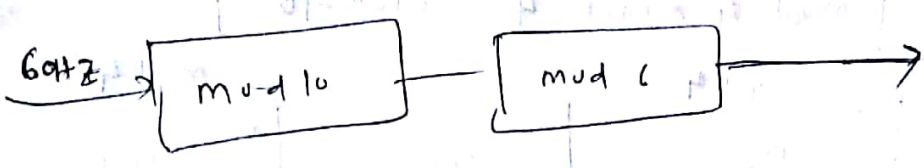
$T_3 = 16T$

$\Rightarrow \frac{1}{T_3} = \frac{1}{16T}$

$\Rightarrow f_3 = \frac{1}{16} \cdot f_0$

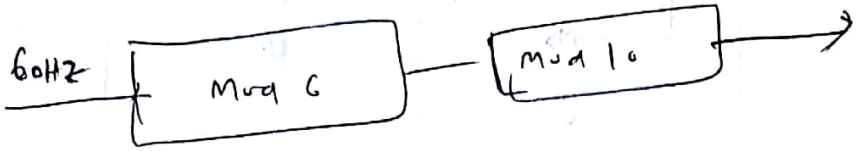
$\Rightarrow \boxed{f_3 = \frac{f_0}{16}}$

1)



$\left[\frac{60}{10\times 6} = 1\right]$

2)



$\left[\frac{60}{6\times 10} = 1\right]$

3)

Ex:- A binary ripple counter is required to count up to $(16,383)_{10}$. How many FFs are required? If Clock freq is 8.192 MHz, what is the freq at the o/p of the MSB

Ans: With N FFs, we have 2^N states And the number it can count $\underline{2^N - 1}$.

2 FF, 4 states, number it can count $2^2 - 1 = 3$

i.e 0 to 3

Similarly here, $2^N - 1 = 16,383$

$\Rightarrow 2^N = 16,384$

$\Rightarrow N \log 2 = \log(16,384)$

$\Rightarrow N = \frac{\log(16,384)}{\log 2} = 14$

No. of FFs required = 14

Freq at o/p of last stage

$$\frac{f_0}{2^{14}} = \frac{8.192 \times 10^6}{16,384} = 500\,Hz$$

(ex:- In Mod '4' Counter At MSB freq (f_4) $= \frac{f_0}{16} = \frac{f}{24} = \frac{f_0}{\text{No. of F/Fs}}$ i.e $\frac{f_0}{\text{No. of States}}$)

Synchronous Counters:-

→ Synchronous Counters are Counters in which all F/Fs are triggered simultaneously by the Clock-input pulses.

→ Whether a FF [toggles or not] depends on FF's input (J,K or D or T, or S,R).

→ Since all the F/Fs change state simultaneously in synchronization with the Clock pulse, the propagation delay of F/Fs don't add together (as in Case of ripple Counters) to produce overall delay.

→ In fact, the propagation delay of a synchronous Counter is equal to propagation delay of just one FF plus the propagation delay of any other gates involved.

Design of Synchronous Counters

Steps

1. Number of F/Fs

Based on the description of the problem, determine the required number of F/Fs (n) i.e such that

$$N \leq 2^n$$

$N \rightarrow$ number of states
$n \rightarrow$ no. of F/Fs.

ex: Mod 10 counter, $N = 10$,

$$10 < 2^3 \times, \quad 10 \leq 2^4 \quad \underline{10 \leq 16}, \quad \boxed{n = 4}$$

2. State diagram

Draw the state diagram showing all possible states.

3. Choice of F/F & excitation table

Select the type of F/F to be used and write the excitation table

4. Minimal expressions for excitations

Obtain the minimal expressions for the excitations of the F/Fs using the K-maps.

5. Logic diagram :-

Draw the logic diagram based on the minimal expressions.

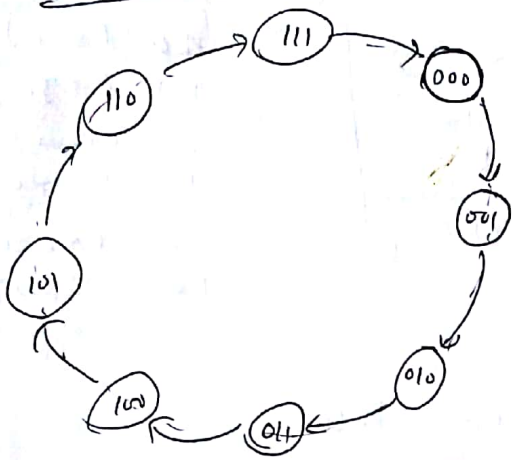
1) Design of Synchronous 3-bit UP-Counter

Steps

1. Number of FFs required = 3

(∵ $N \leq 2^n$, $8 \leq 2^3$)

2. State diagram



3. JK FFs are selected & excitation table is drawn. (Next page)

Note:- Excitation table of various FFs used in Counters

(i) S-R F/F excitation Table (For understanding)

Present State	Next State	Required I/Ps		Remark	
Q_n	Q_{n+1}	S	R	S	R
0	0	0	X	0 0	(No change)
				0 1	(Reset)
0	1	1	0	1 0	(Set)
1	0	0	1	0 1	(Reset)
1	1	X	0	0 0	(No change)
				1 0	(Set)

X → Don't Care

(ii) JK F/F excitation table

Present State Q_n	Next State Q_{n+1}	Requires I/Ps J	K	Remark J	K
0	0	0	X	0	0 (No change)
				0	1 (Reset)
0	1	1	X	1	0 (Set)
				1	1 (Toggle)
1	0	X	1	0	1 (Reset)
				1	1 (Toggle)
1	1	X	0	0	0 (No change)
				1	0 (Set)

(iii) D F/F excitation table

Present State (Q_n)	Next State (Q_{n+1})	Requires I/Ps (D)	Remark (D)
0	0	0	D = 0 (Reset)
0	1	1	D = 1 (Set)
1	0	0	D = 0 (Reset)
1	1	1	D = 1 (Set)

(iv) T F/F excitation table

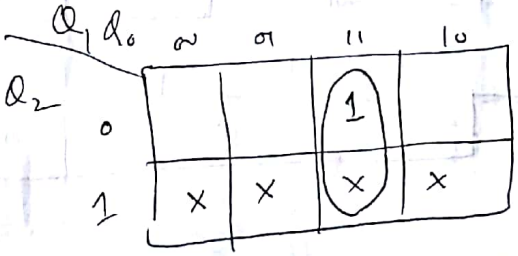
(Q_n) Present State	(Q_{n+1}) Next State	Requires I/P (T)	Remark (T)
0	0	0	0 (No change)
0	1	1	1 (Toggle)
1	0	1	1 (Toggle)
1	1	0	0 (No change)

Excitation table for the J-K FF for 3 bit Synchronous Counter

Present State $Q_2\ Q_1\ Q_0$	Next State $Q_2\ Q_1\ Q_0$	Required excitations $J_2\ K_2$	$J_1\ K_1$	$J_0\ K_0$
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

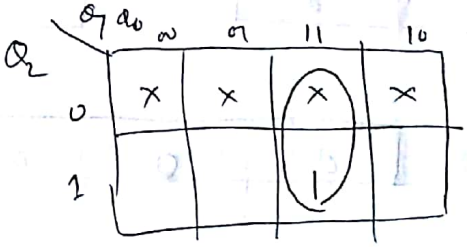
Step 4. Obtain minimal expression [on K-map taking Present State as I/Ps]

For J_2



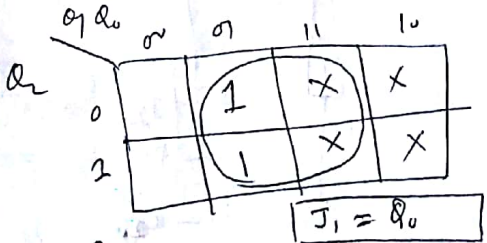
$J_2 = Q_0 Q_1$

For K_2



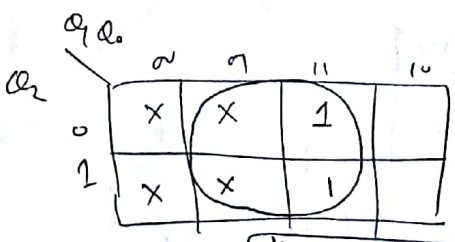
$K_2 = Q_0 Q_1$

J_1



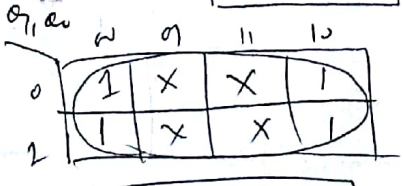
$J_1 = Q_0$

K_1



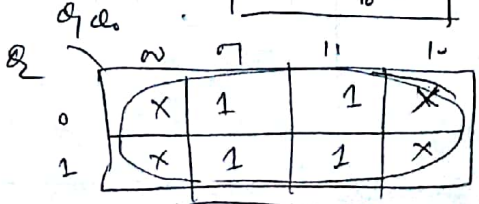
$K_1 = Q_0$

J_0



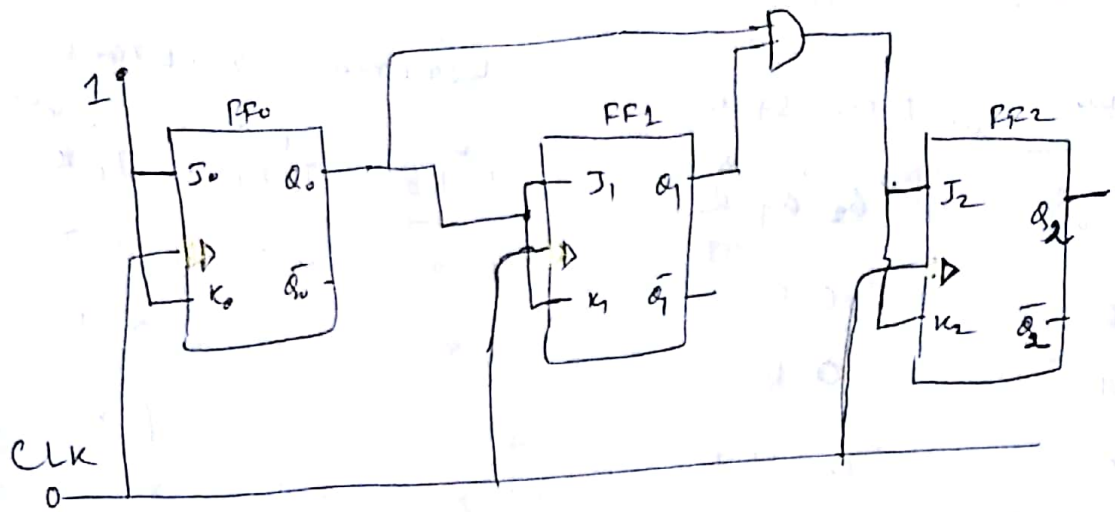
$J_0 = 1$

K_0



$K_0 = 1$

Step 5: Design using +ve edge torggered JK F/F IR



$J_0 = K_0 = 1$, $J_1 = K_1 = Q_0$, $J_2 = K_2 = Q_0 Q_1$

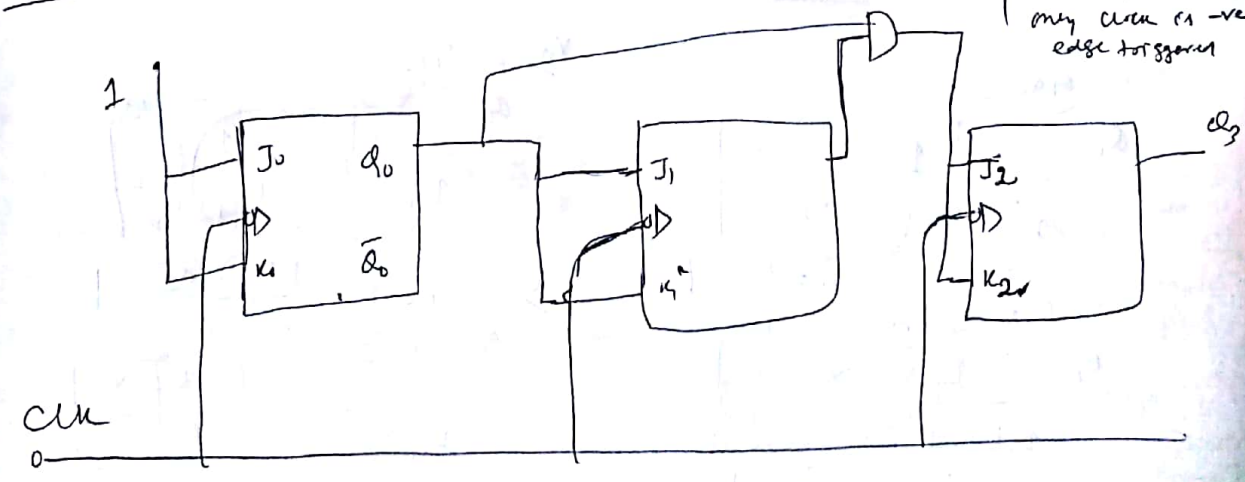
Note :- The design of Synchronous Counter using +ve edge torggered or -ve edge torggered F/F is same.

only the ~~torggering~~ Changes in the o/p takes place at +ve edges or -ve edges.

In circuit, the change is in the Clock symbol only (or).

Design using -ve edge edge torggered JK F/F

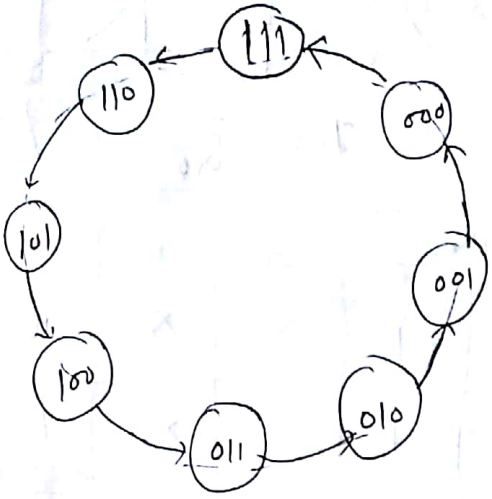
(exactly similar only clock is -ve edge torggered)



Q/ 2) Design of Synchronous 3 bit down Counter.

Steps 1. No of FF/Rs required = 3

2. State diagram



3. Excitation Table

Present State $Q_2\ Q_1\ Q_0$	Next State $Q_2\ Q_1\ Q_0$	Required Excitations $J_2\ K_2$	$J_1\ K_1$	$J_0\ K_0$
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

4. Obtain minimal expression

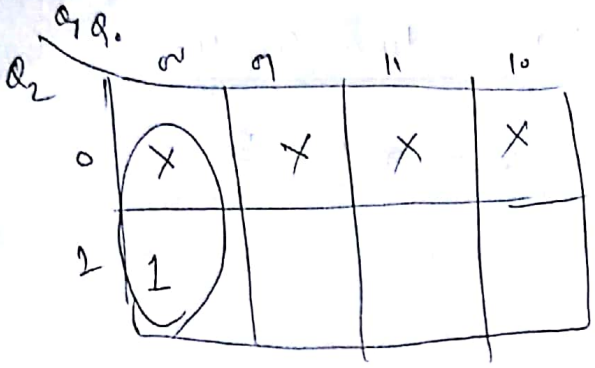
For J_2

Q_2 \ Q_1Q_0	00	01	11	10
0	1			
1	X	X	X	X

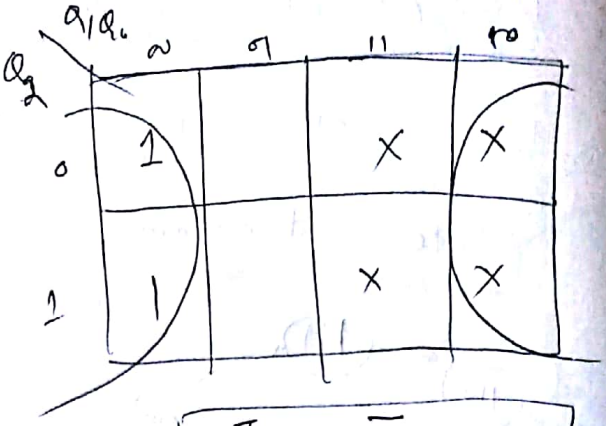
$J_2 = \bar{Q}_1 \bar{Q}_0$

$J_2 = \bar{Q}_0 \bar{Q}_1$

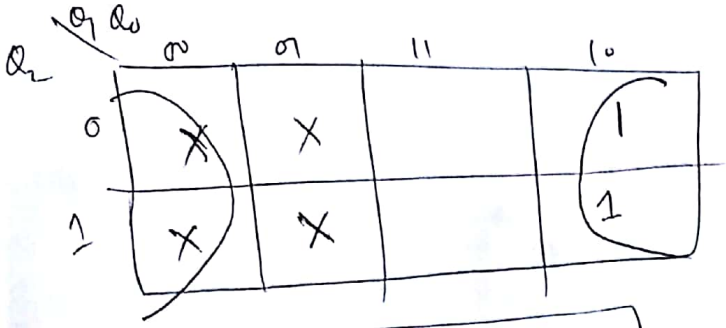
or $\overline{J_2} = \overline{Q_0 + Q_1}$



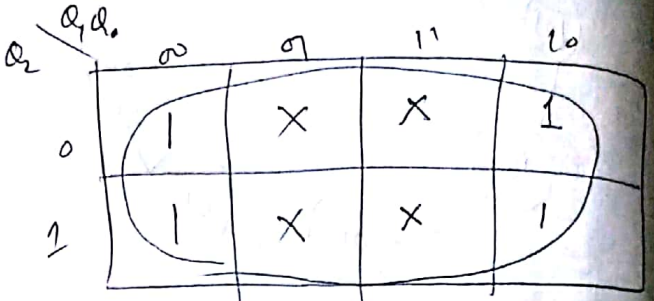
$$K_2 = \bar{Q}_0\bar{Q}_1 = \overline{Q_0 + Q_1}$$



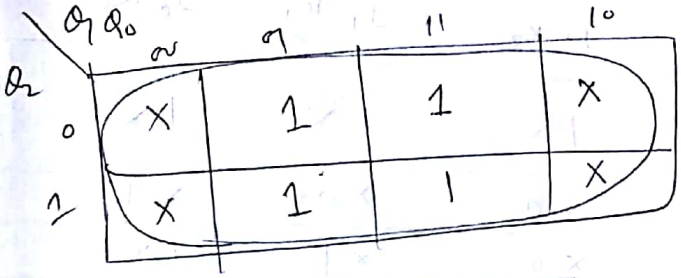
$$J_1 = \bar{Q}_0$$



$$K_1 = \bar{Q}_0$$

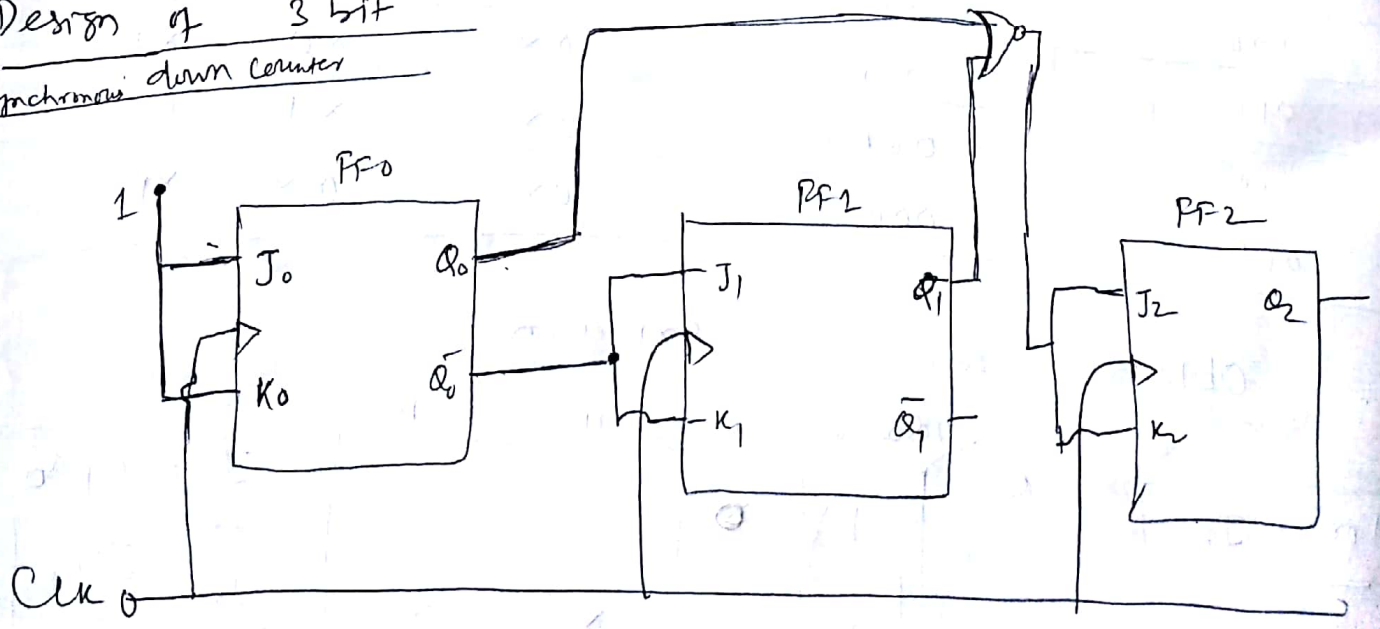


$$J_0 = 1$$



$$K_0 = 1$$

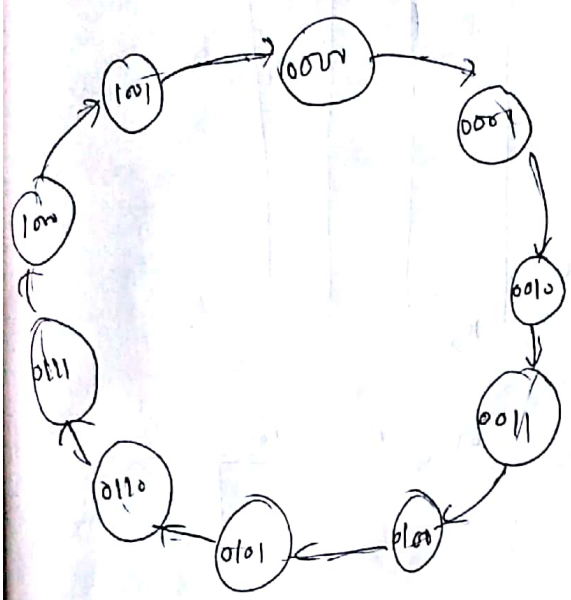
5) Design of 3 bit Synchronous down counter



3) Design of Synchronous Mod-10 Counter

1. No of FF's required = 4.

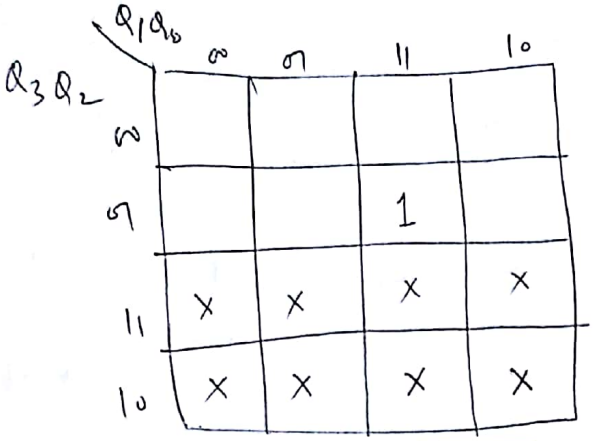
2. State diagram



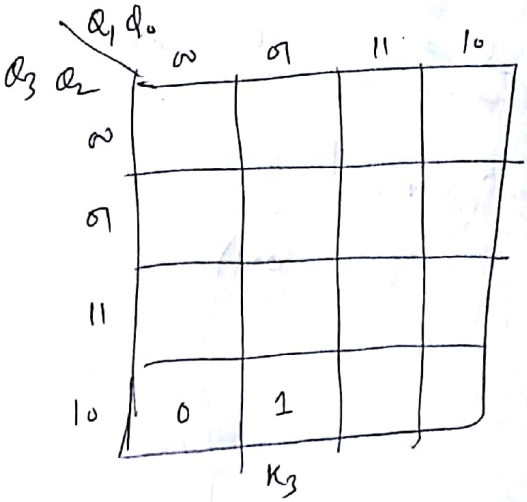
3) Excitation Table

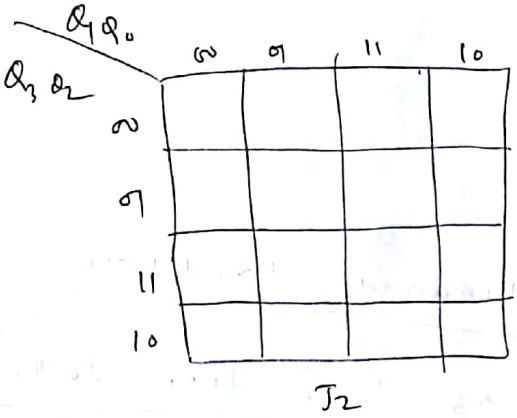
Present State				Next State				Required Excitations							
Q_3	Q_2	Q_1	Q_0	Q_3	Q_2	Q_1	Q_0	J_3	K_3	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	0	0	0	0	1	0	X	0	X	0	X	1	X
0	0	0	1	0	0	1	0	0	X	0	X	1	X	X	1
0	0	1	0	0	0	1	1	0	X	0	X	X	0	1	X
0	0	1	1	0	1	0	0	0	X	1	X	X	1	X	1
0	1	0	0	0	1	0	1	0	X	X	0	0	X	1	X
0	1	0	1	0	1	1	0	0	X	X	0	1	X	X	1
0	1	1	0	0	1	1	1	0	X	X	0	X	0	1	X
0	1	1	1	1	0	0	0	1	X	X	1	X	1	X	1
1	0	0	0	1	0	0	1	X	0	0	X	0	X	1	X
1	0	0	1	0	0	0	0	X	1	0	X	0	X	X	1
1	0	1	0												

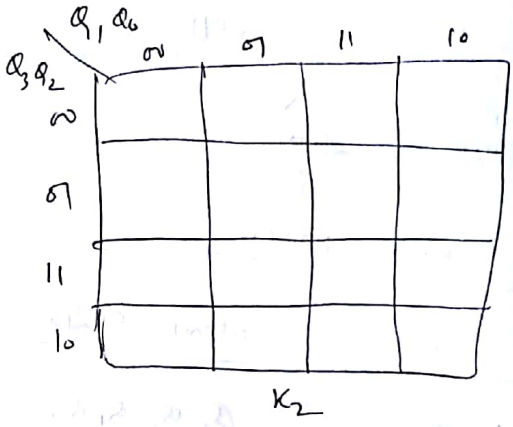
4) Obtain the minimal expression

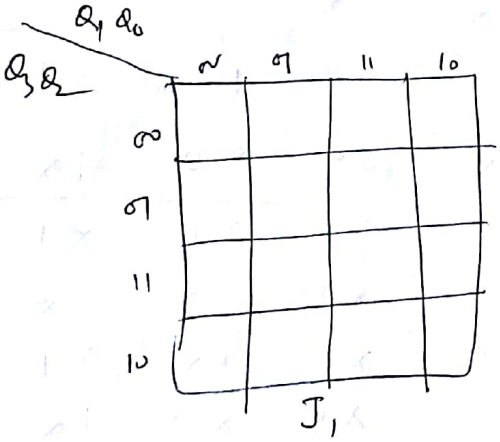


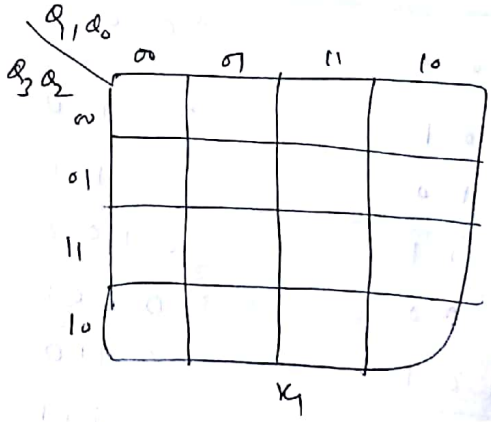
$J_3 =$

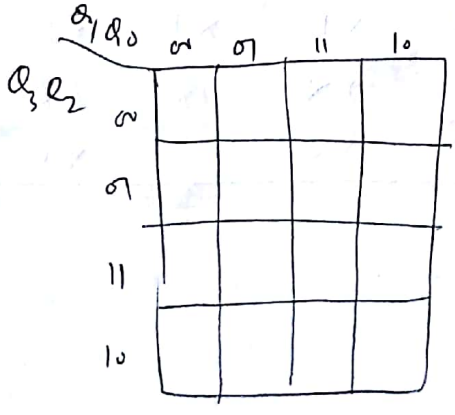


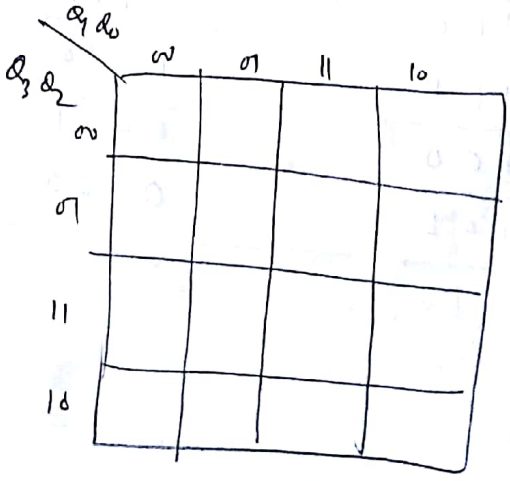












Shift Registers

→ Shift registers are a type of sequential logic circuit closely related to digital Counters. Registers are used primarily for the storage of digital data and typically don't possess a characteristic internal sequence of states as do Counters. (like in upcounter 000 → 001 → 010 ---- the states are)

→ Shift registers consists of an arrangement of F/Fs and are important in applications involving the storage and transfer of data in a digital system.

→ A register is a digital ckt with two basic functions: data storage and data movement.

→ The storage capacity of a register is the total number of bits (1s or 0s) of digital data it can retain. Each stage (F/F) in a shift register represents one bit of storage capacity; therefore the number of stages in a register determine its storage capacity.

→ Ex:- 4 F/Fs, 4 bit can be stored.

→ The shifting capability of a register permits the movement of data from stage to stage within the register or into or out of

The register upon the application of Clock pulses.

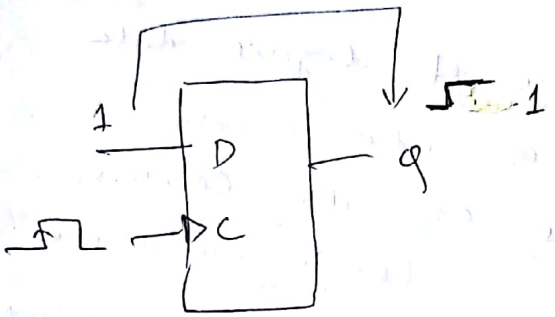


fig 1:-

When a 1 is on D, Q becomes a 1 at the triggering edge of Clk or remains a 1 if already in the Set

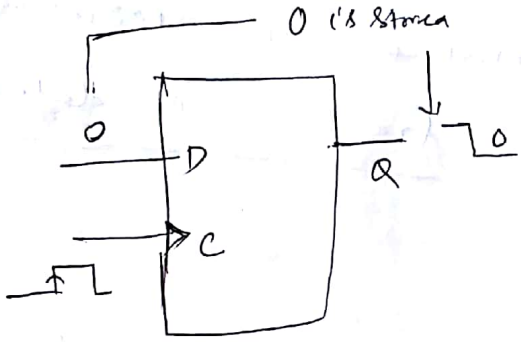
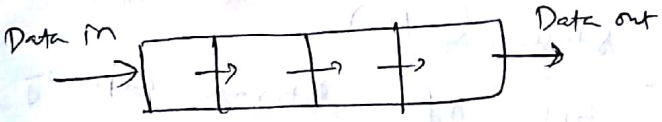


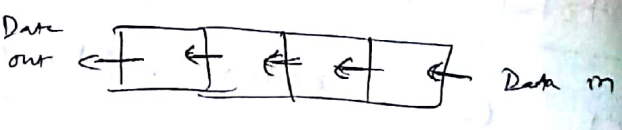
fig 2:-

when a 0 is on D, Q becomes a 0 at the triggering edge of the Clk or remains a '0' if already in the RESET value.

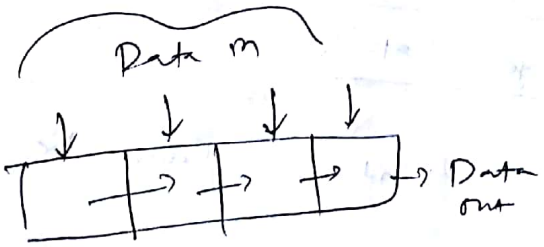
Fig 1 & 2 :- The F/F as a storage element



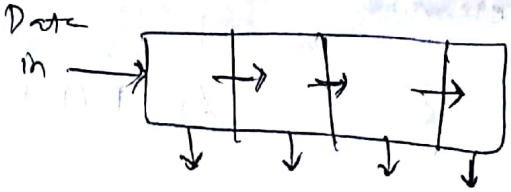
(a) Serial in / Shift right / Serial out



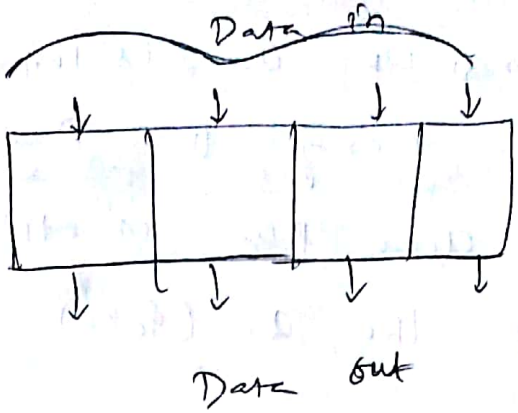
(b) Serial in / Shift left / Serial out



(c) Parallel in / Serial out



(d) Serial in / Parallel out



(e) Parallel in / Parallel out

1) Serial in / Serial out Shift register :-

The Serial in / Serial out shift register accept data serially - i.e one bit at a time on a single line. It produces the stored information on its o/p also in serial form.

Let's first look at the serial entry of data into typical shift register. Consider a 4 bit shift register implemented using D-F/F's.

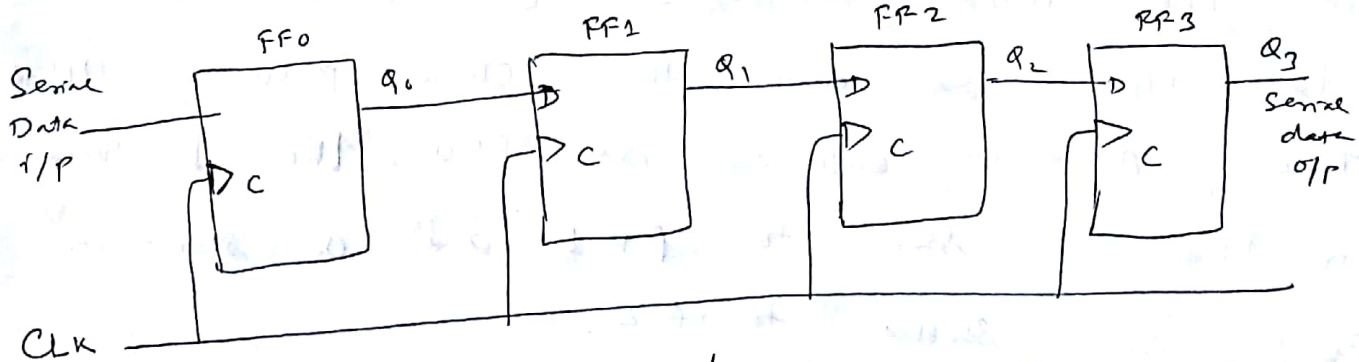


Fig:- Serial in / serial out shift register

→ Consider the entry of 4 bits 1010 into the register, beginning with the rightmost bit (LSB side).

The register is initially clear i.e

$Q_0 = Q_1 = Q_2 = Q_3 = 0$.

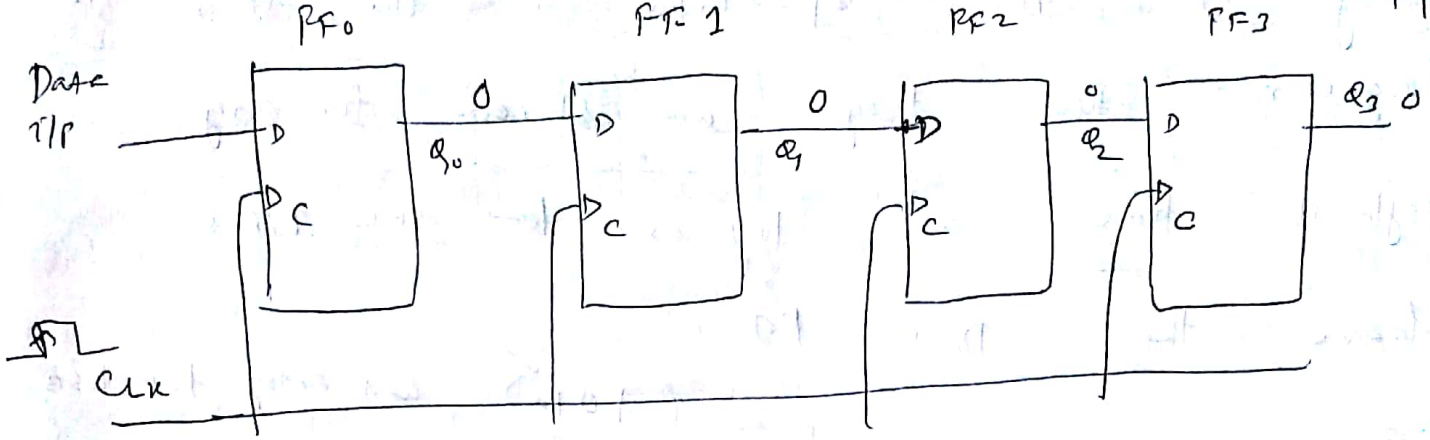
The right most bit '0' is put onto the data i/p line, making the D = 0 for FF0. When the first clock pulse is applied, FF 0 is reset, thus storing the 0. ($Q_0 = 0$)

Next the second bit, which is 1, is applied to the data i/p, making D = 1 for FF 0 and D = 0 for FF 1 because the D i/p of FF 1 is connected to Q_0 o/p. (Since Q_0 was i.e D of FF 1 = 0, $Q_0 = 0$, when Clock pulse applied Q_1 will be $= 0$)

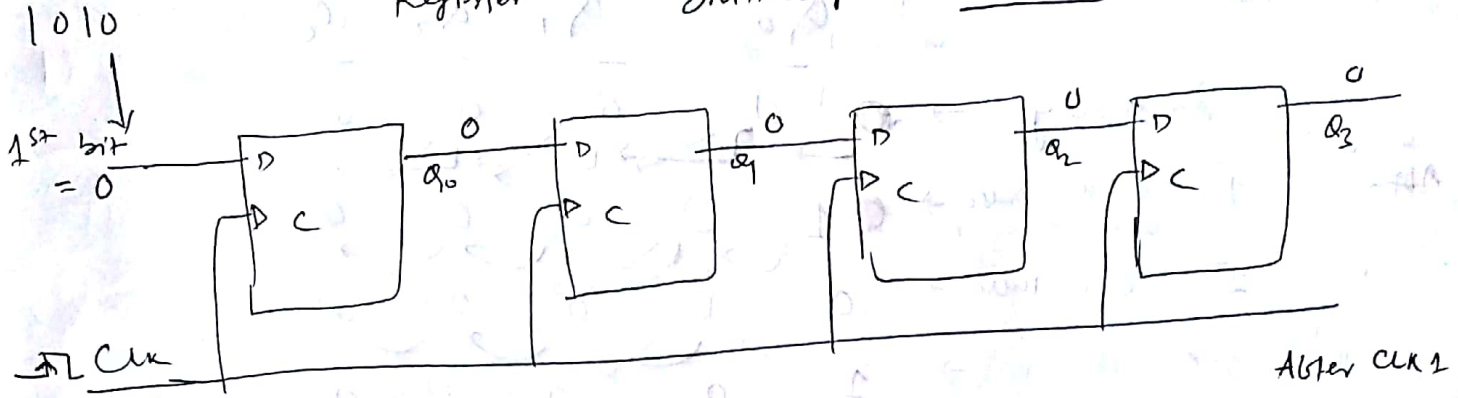
→ So when second Clock Pulse occurs, the 1 on the data i/p is shifted into FF0, causing FF0 to set; and 0 that was in FF0 is shifted to FF 1.

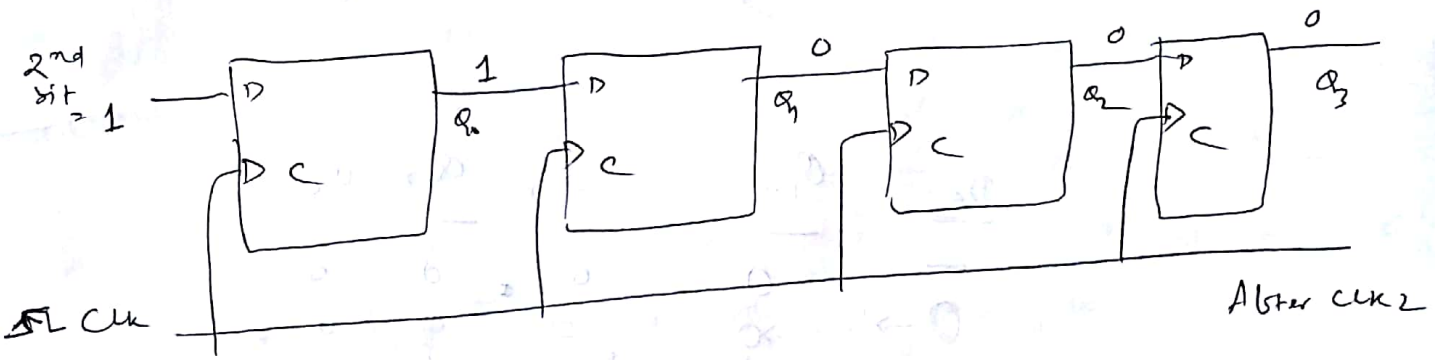
→ The third bit, a '0' is now put onto the data i/p line and the clock pulse is applied. The 0 is entered into FF0, the 1 stored in FF0 is shifted to ff 1 and 0 stored in FF 1 is shifted to FF 2.

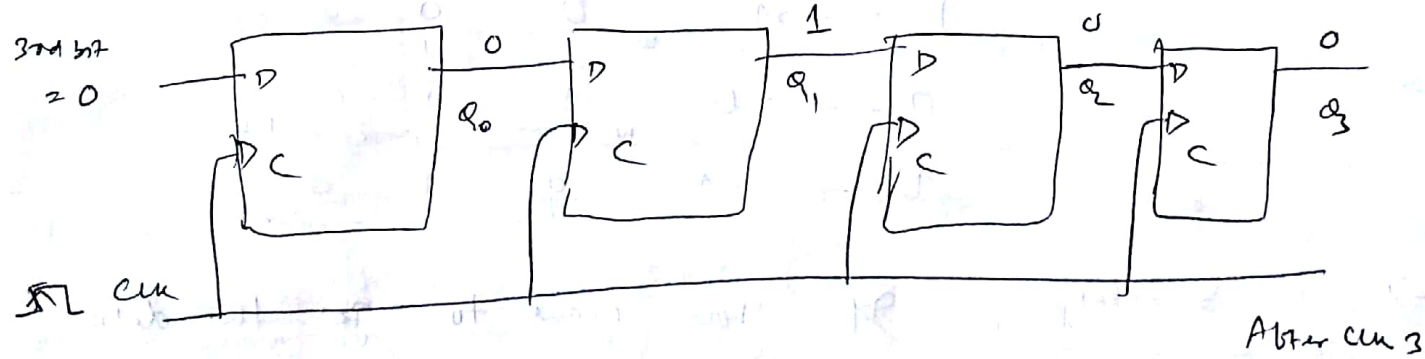
→ The last bit, a 1, is now applied to the data i/p and Clock pulse is applied. This time 1 is entered to FF0, the 0 stored in FF0 is shifted to FF 1, the 1 stored in FF 1 is shifted to FF 2 & 0 stored in FF 2 is shifted to FF 3. This completes the serial

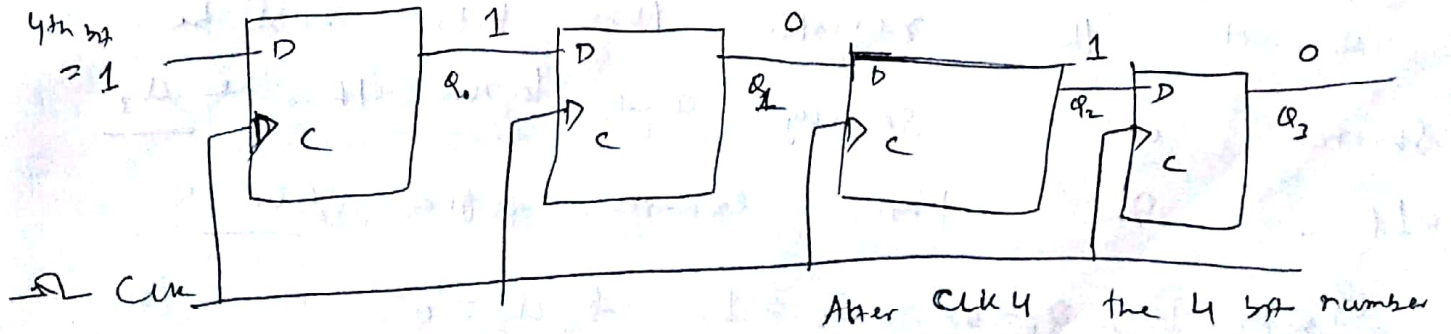


Register Initially CLEAR.





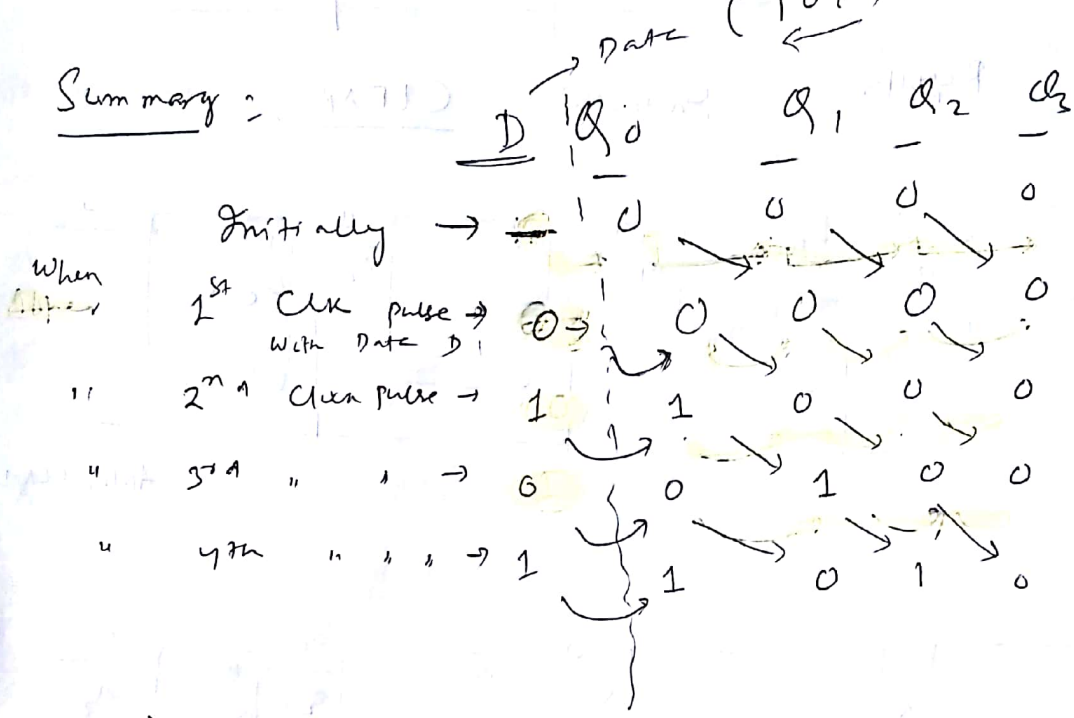




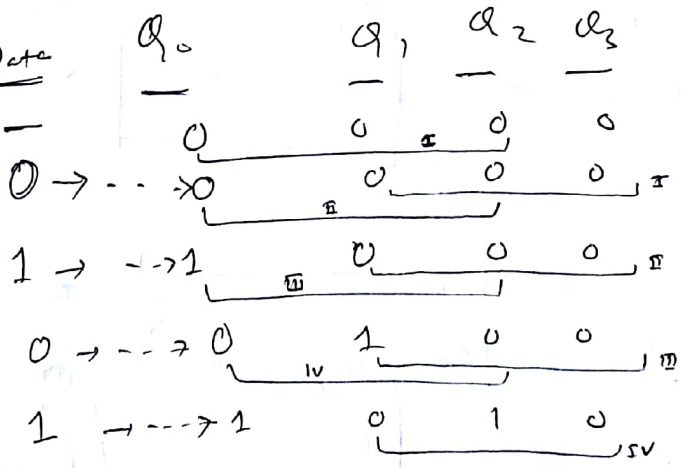
After CLK 4 the 4 bit number is completely stored in register.

entry of the 4 bits into the shift register, where they can be stored for any length of time as long as the F/F-s have the D.C power.

Summary :



i.e



→ Similarly, If you want to get the data out of the register, the bits must be shifted out serially and taken off the Q_3 o/p. In I ass example, after Clk 4, $Q_0 = 1$, $Q_1 = 0$, $Q_2 = 1$ & $Q_3 = 0$.

To get back the data entered / stored, we have to apply 4 more clock pulses with ~~data~~ i/p at D 0000. So we get the data at Q_3.

i/p (at) Data (D)

After CLK 4 →

	Q_0	Q_1	Q_2	Q_3
	1	0	1	0
When CLK 5 ↓ 0 →	0	1	0	1
CLK 6 ↓ 0 →	0	0	1	0
CLK 7 ↓ 0 →	0	0	0	1
CLK 8 ↓ 0 →	0	0	0	0

∴ Four bits (1010) being serially shifted out at Q_3 & register again cleared to (0000)

2) Serial in / Parallel out shift register

Data bits are entered serially (right-most bit first) into this type of register in the same manner as discussed previously.

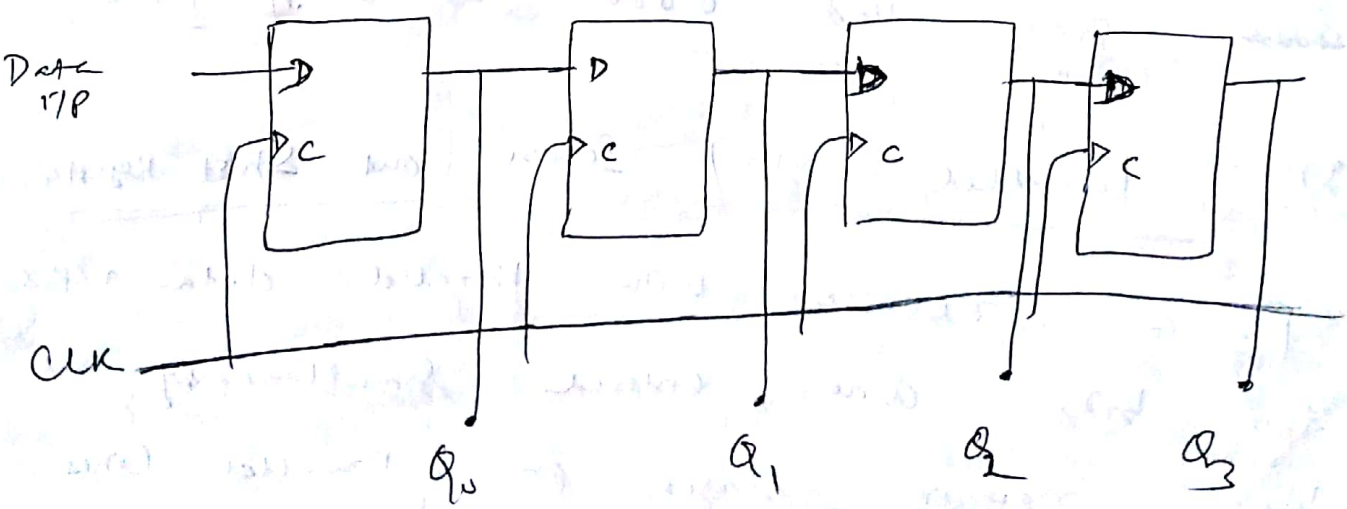
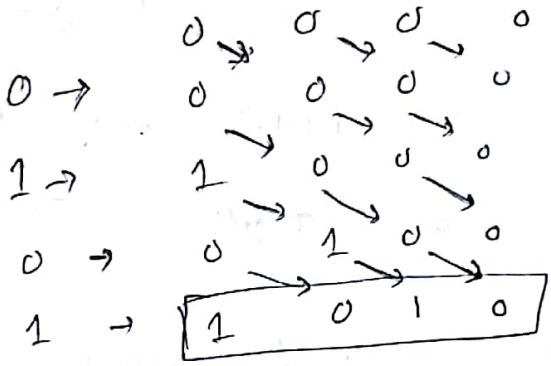


fig:- Serial in / Parallel out shift register

The difference is the way in which the data bits are taken out of the register; in parallel o/p register, the o/p of each stage is available. Once the data are stored, each bit appears on its respective output line, and all bits are available simultaneously, rather than on a bit-by-bit basis as with serial output.

Ex:- 1010 →



So o/p is available at 4 terminals simultaneously.

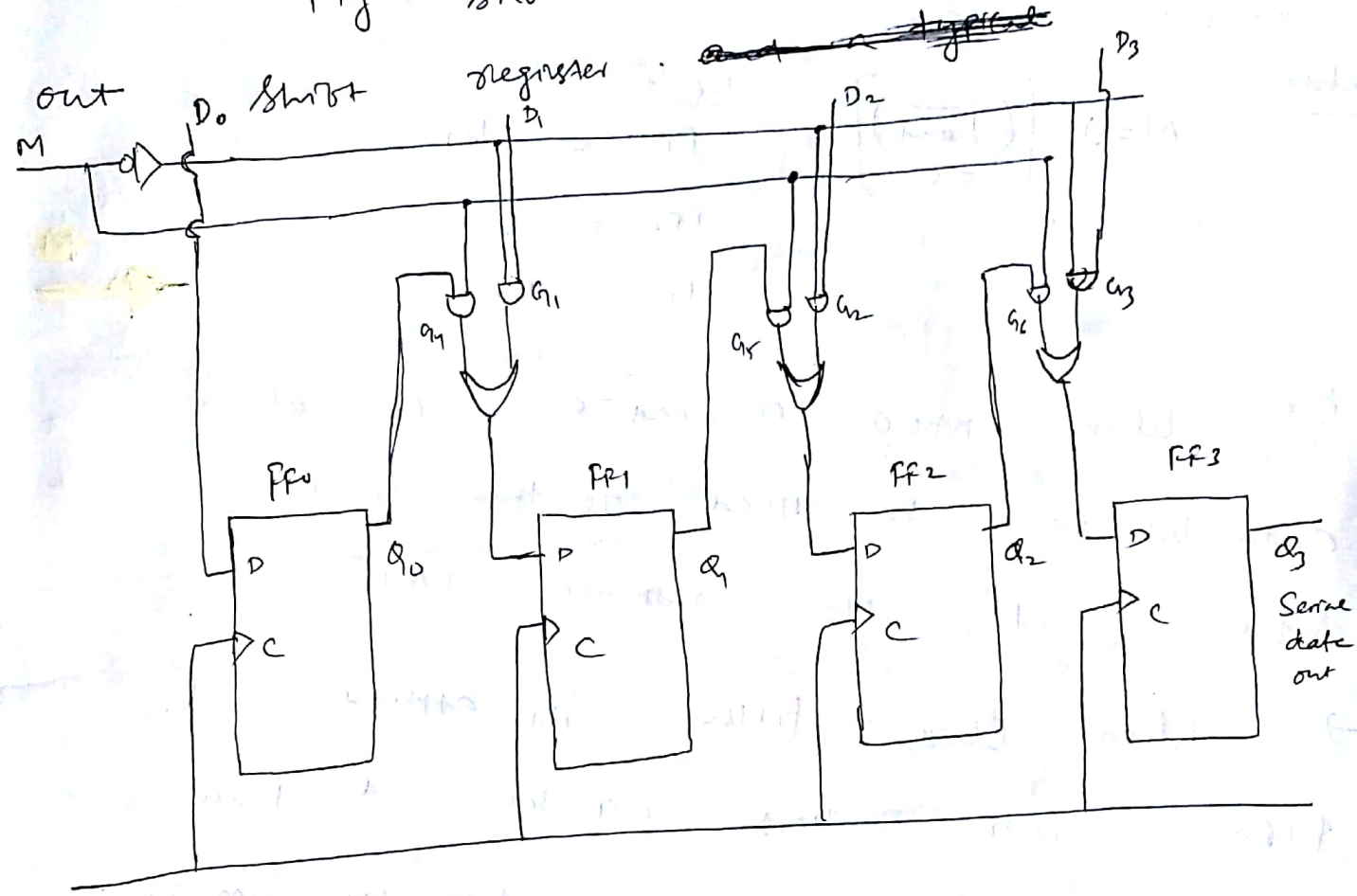
But in serial out, we try to retrieve the data bit by bit at Q_3 terminal by giving n/s 0 0 0 at D of FF0.

3) Parallel in / Serial out Shift Register

For a register with parallel data i/p s the bits are entered simultaneously into their respective stages on parallel lines rather than on a bit-by-bit basis on the one

line as with data i/ps. The Serial out is same as described in Serial in Serial out F/F, once the data is Completely stored in the register.

Fig shows a 4 bit Parallel in/Serial out shift register.



→ Notice that there are 4 data i/p lines D_0, D_1, D_2, D_3 and a M i/p (Some books write Shift/$\overline{Load}$ i/p), which allows four bits of data to load in parallel into register. When M = 0 (for $\overline{Load}$ = 0), gates G_1 & G_2 & G_3 are enabled. (Because when M = 0 it passes through a NOT gate, o/p is 1, they are connected to G_1, G_2 & G_3), G_4, G_5, G_6 's o/p = 0 (As M = 0)

Mathematically 'D' i/p to FF0 = D_0 —— (1)

D i/p to RF1 = $Q_0 \cdot M + D_1 \cdot \bar{M}$ —— (2)

'D' i/p to RF2 = $Q_1 \cdot M + D_2 \cdot \bar{M}$ —— (3)

'D' i/p to RF3 = $Q_2 \cdot M + D_3 \cdot \bar{M}$ —— (4)

When $M = 0$ $\left[\left(\overline{Load}\right) = 0\right]$ i/p to

RF1 = D_1

RF2 = D_2

RF3 = D_3

i.e when $M = 0$ or $\overline{Load} = 0$, it allows data bit to be applied to the D i/p sequence of its respective FFs.

→ When clock pulse is applied, the FFs with D=1 will set & those with D=0 will reset, there by storing all 4 data bits simultaneously.

(i.e Parallel i/n)

→ Now when M = 1, or (Shift = 1)

Using eqn (1), (2), (3), (4)

'D' i/p to FF0 = D_0

'D' " " RF2 = Q_0

'D' " " RF2 = Q_1

'D' " " FF3 = Q_2

→ After passing through NOT gate, G_1, G_2, G_3 are disabled, G_4, G_5, G_6 are enabled.

So data bits are allowed to shift from one stage to next, when a clock pulse is applied.

Thus the OR gate allows either the normal shifting operation (when M = 1 or Shift = 1) or parallel data entry operation (when M = 0 or $\overline{Load} = 0$) depending on which AND gates are enabled by the level on the SHIFT / $\overline{LOAD}$ i/p.

4/ Parallel in/ Parallel out Shift Registers

In a parallel-in, parallel out, shift register, the data bits entered into the register in parallel form and also the data bits taken out of the register in parallel form.

Immediately following the simultaneous entry of all data bits, the bits appears on the parallel outputs.

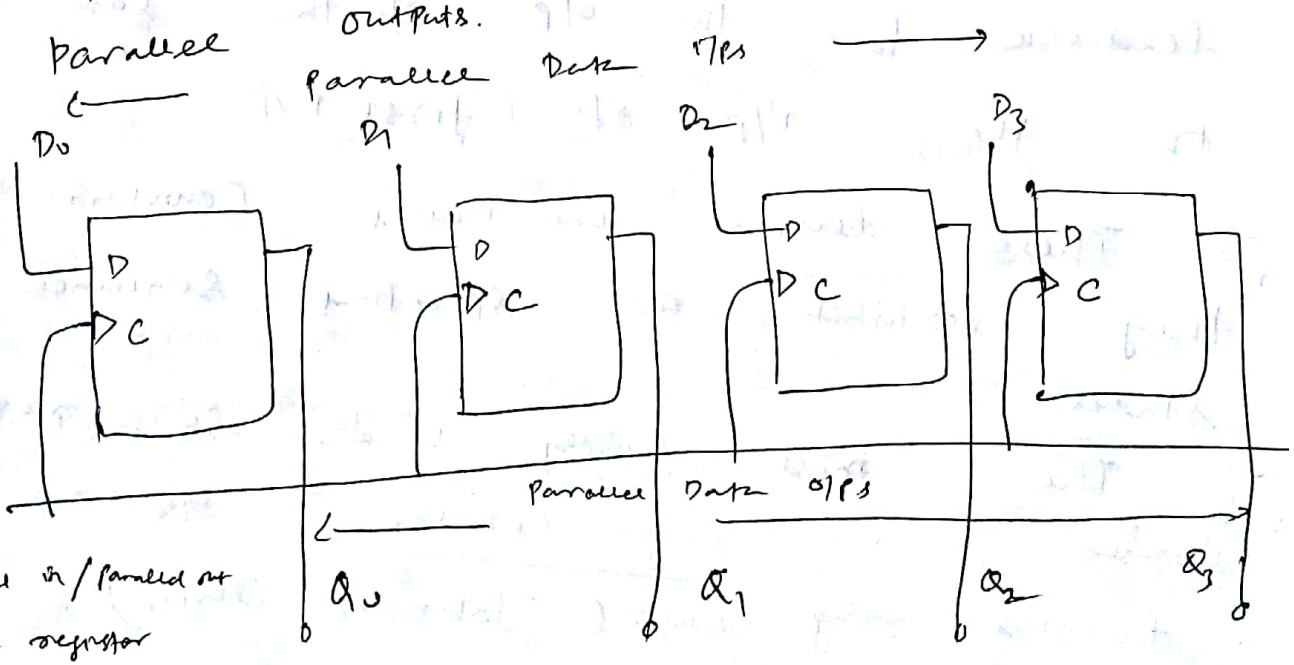


fig. - Parallel in/Parallel out Shift register

Figure shows a 4 bit parallel-in Parallel-out, Shift register using D FFs. Data i/ps is applied to D i/ps terminals of the FFs. When clock pulse is applied, as the +ve-going edge of that pulse, the Data i/ps at D terminal appears at the 'Q' o/p of the FFs. The register now stores the data. The stored data is available instantaneously in parallel form.

Shift Register Counters

→ One of the applications of Shift registers is that they can be arranged to form several types of Counters.

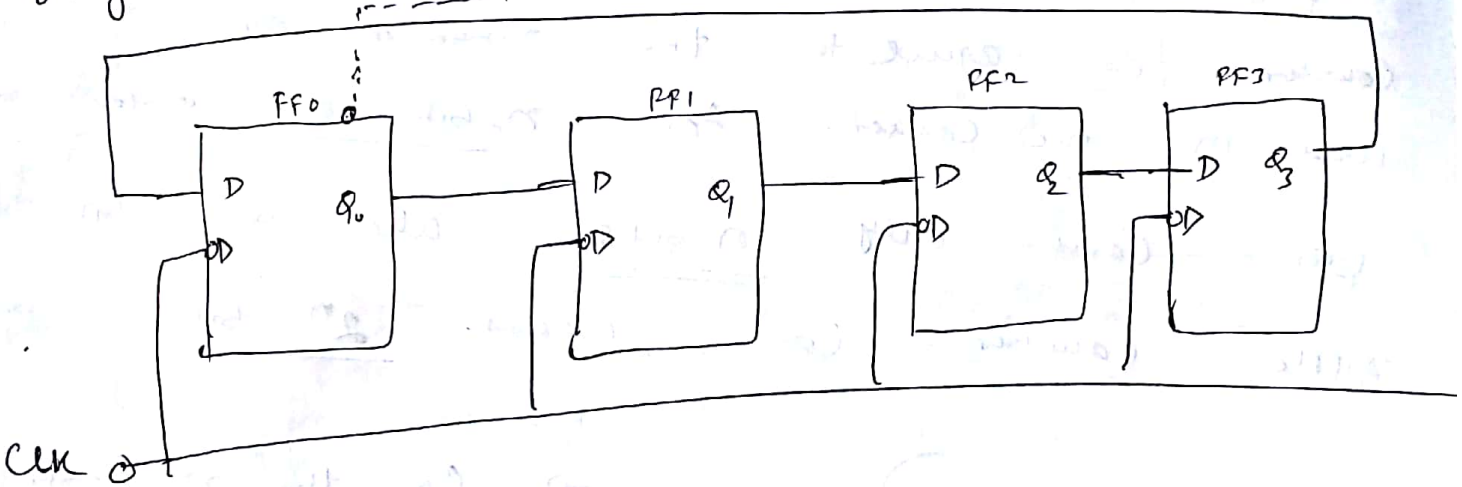
→ Shift register Counters are obtained from Serial-in / Serial out Shift registers by providing feedback from the o/p of the last F/F to the i/p of first F/F.

→ These devices are called Counters because they exhibit a specific sequence of states.

→ The most widely used Shift registers counter is ring Counter as well as twisted ring Counter (Johnson Counter).

Ring Counter:-

This is simplest shift register Counter. The basic ring Counter using D F/Fs is shown in figure. The F/Fs are arranged in as in normal shift register i.e. Q o/p of each stage is connected to the D i/p of the next stage, but the Q o/p of the last F/F is connected back to the D i/p of the first F/F such that the array of F/Fs is arranged in a ring and therefore, the name ring Counter.



→ Initially, the first F/F is preset to 1.

So the initial state is 1 0 0 0.

i.e $Q_0 = 1$, $Q_1 = 0$, $Q_2 = 0$, $Q_3 = 0$.

After each clock pulse, the contents of the registers are shifted to the right by one bit and Q_3 is shifted back to Q_0.

Sequence table

	Q_0	Q_1	Q_2	Q_3
After clock pulse				
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1
4	1	0	0	0
5	0	1	0	0
6	0	0	1	0
7	0	0	0	1

4 States

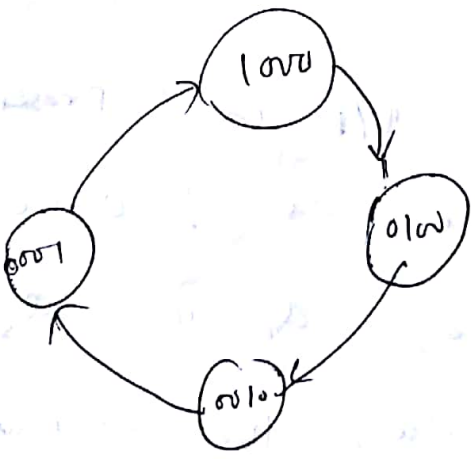
Repeated

(Initial State) (8)

Clock: 9, 10, 11, 12, 13, 14

→ The sequence repeats after 4 clock pulses. The number of distinct states in ring counters is the mod of the ring Counter is equal to the number of F/F's used in the counter. An n-bit ring counter can count only n bits, whereas n-bit ripple counters can count 2^n bits.

→

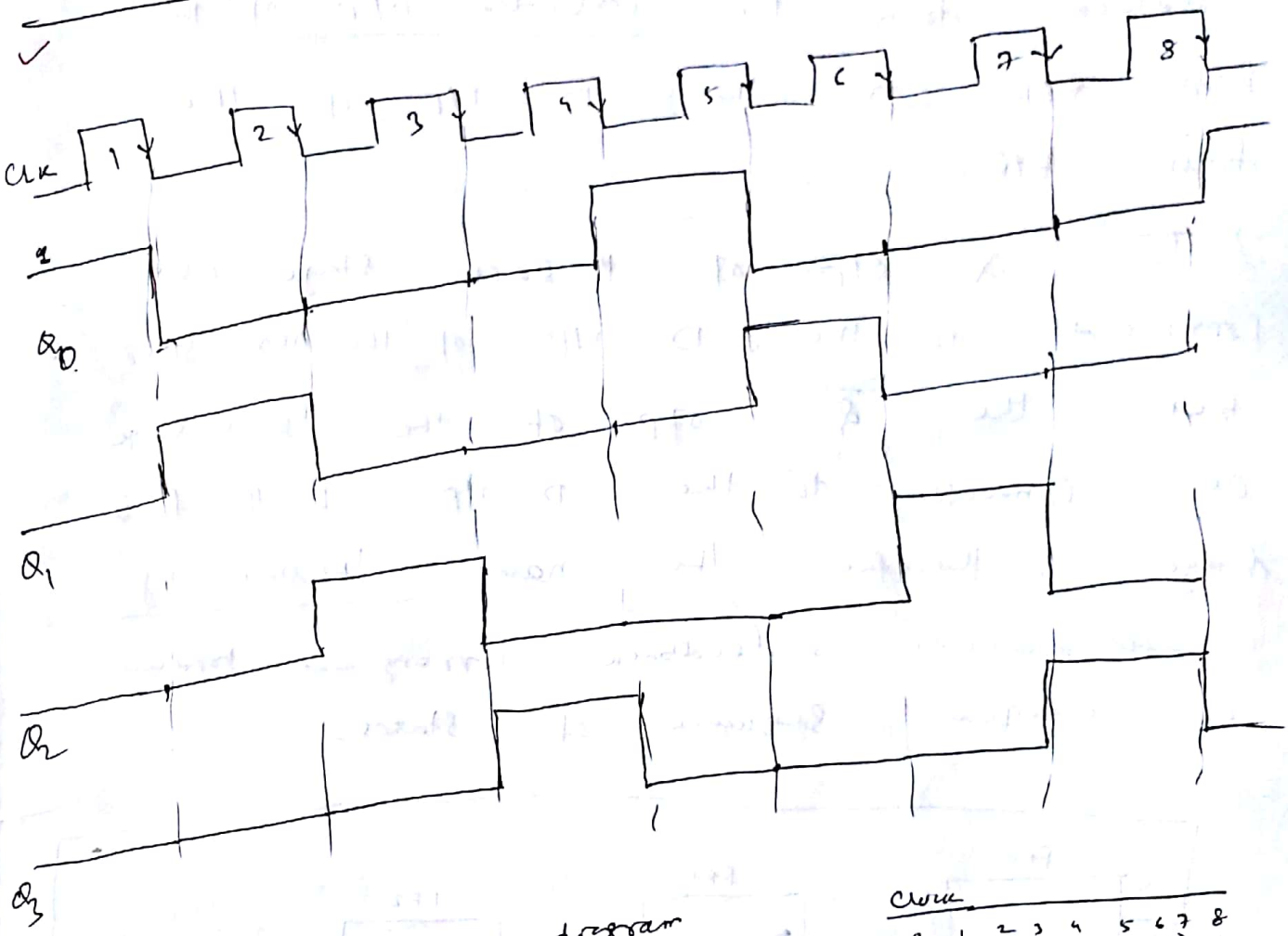


(State diagram)

→ So, the ring counter is uneconomical to ripple counter, but has the advantage of requiring no decoder, since we can read the count by simply noticing which F/F

is set. It is a Synchronous Counter and very fast.

Timing diagram of 4 bit ring Counter



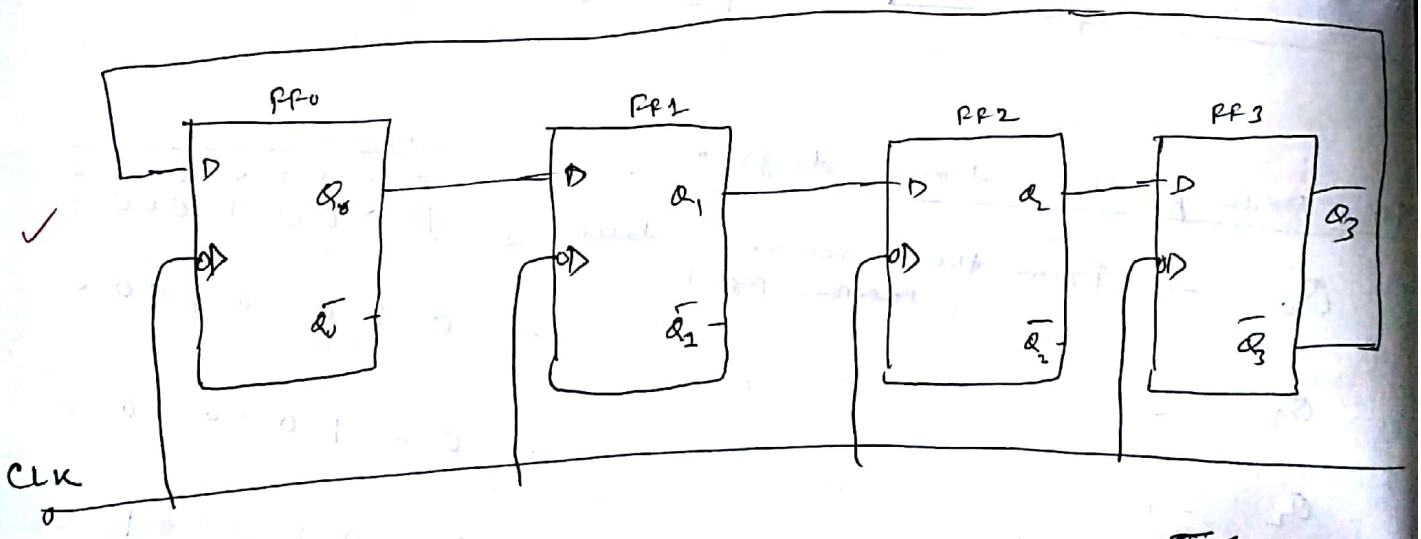
For drawing the timing diagram

		Clock								
		0	1	2	3	4	5	6	7	8
Q_0 → from the sequence table →		1	0	0	0	1	0	0	0	1
(previous one)	→	0	1	0	0	0	1	0	0	0
Q_1 → "	→	0	0	1	0	0	0	1	0	0
Q_2 → "	→	0	0	0	1	0	0	0	1	0
Q_3 → "										

Twisted Ring Counter (Johnson Counter)

→ This Counter is obtained from a Serial in, Serial-out Shift register by providing feedback from the inverted o/p of the last F/F to the D i/p of the first F/F.

→ The Q o/p of each stage is connected to the D i/p of the next stage, but the $\bar{Q}$ o/p of the last stage is connected to the D i/p of the first stage; therefore, the name twisted ring Counter. This feedback arrangement produces a unique sequence of states.

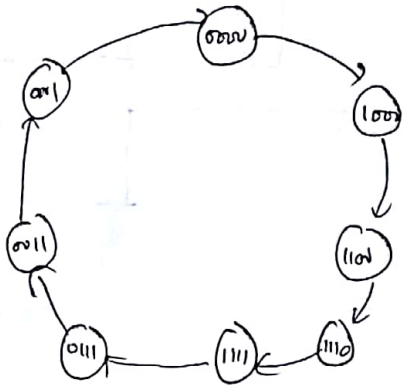


→ The Logic diagram of a 4 bit Johnson Counter using D-F/F is shown in figure.

→ Let initially all the F/F-s are reset i.e the state of the Counter be 0 0 0 0. After each clock pulse, the

level of Q_0 is shifted to Q_1, $Q_1 \to Q_2$, $Q_2 \to Q_3$ and the level of $\overline{Q_3}$ to Q_0. The sequence is repeated after 8 clock pulses.

State diagram



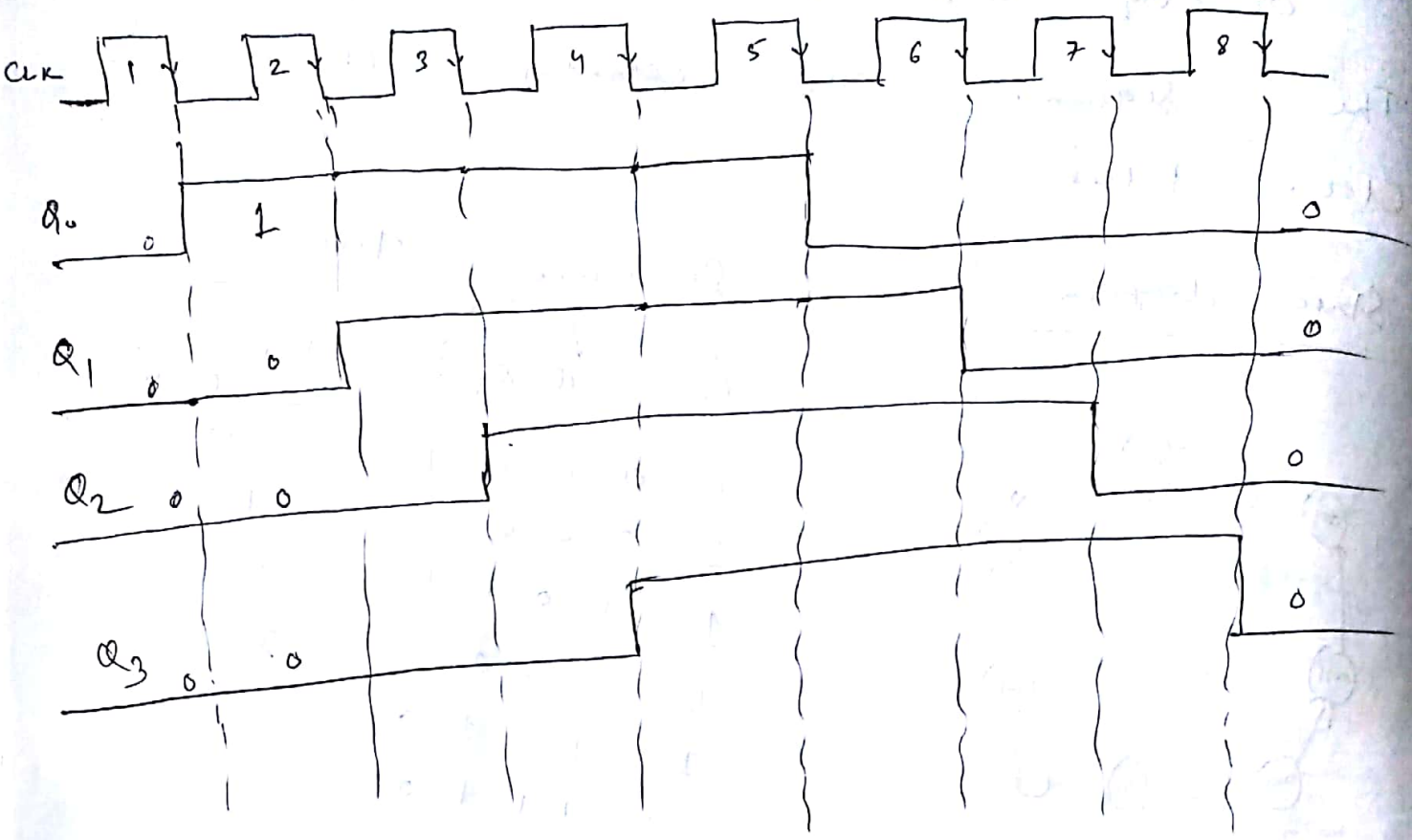
Sequence Table

$Q_0 \to Q_1 \to Q_2 \to Q_3$	$\overline{Q_3}$	After clock pulse
0 0 0 0	1	0
1 0 0 0	1	1
1 1 0 0	1	2
1 1 1 0	1	3
1 1 1 1	0	4
0 1 1 1	0	5
0 0 1 1	0	6
0 0 0 1	0	7
0 0 0 0	1	8

→ So in general,

A Johnson Counter having n F/F-s, has '2n' unique states and can count up to 2n pulses. So it is a mod-2n Counter. It is more economical than ring Counter, but less economical than ripple Counters (2^n).

Timing diagram of 4 bit Johnson Counter



Conversion of FFs

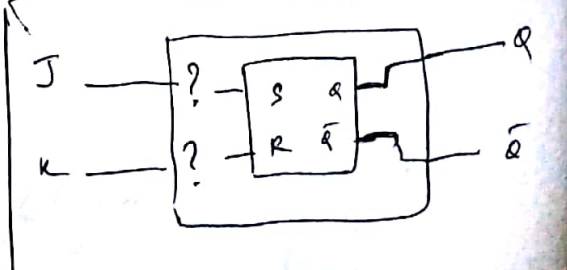
1) S-R FF to J-K F/F

That means we have SR F/F. But we want the functionality of J-K F/F.

So we have to modify SR FF so that it will behave as J-K F/F. So what should be the value of S & R I/Ps?

We have J & K I/Ps. To find what should be the value of S & R interms of J & K & Q_n, $\bar{Q}_n$.

Situation



	External i/ps J	K	Present State Q_n	Next State Q_{n+1}	FF i/p S	R	(From excitation table)
No change	0	0	0	0	0	X	
			1	1	X	0	
Reset	0	1	0	0	0	X	
			1	0	0	1	
Set	1	0	0	1	1	0	
			1	1	X	0	
Toggle	1	1	0	1	1	0	
			1	0	0	1	

Table 2 - Conversion table for SR FF to JK FF.

ex:- How the Next State Column of the conversion table found?

ex: $J=0$, $K=0$, $Q_n=0$, So Q_{n+1} = No Change = 0.

$J=0$, $K=0$, $Q_n=1$, So Q_{n+1} = " = 1.

How the the F/F i/p Column is found? i'n the above table?

From excitation table of SR FF. (As discussed earlier)

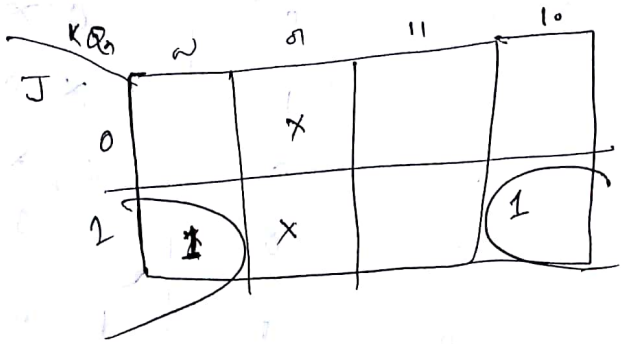
When $Q_n=0$, $Q_{n+1}=0$, S R should be 0 X.

$Q_n=1$, $Q_{n+1}=1$, " " " " X 0.

→ In this way the the Conversion table is made.

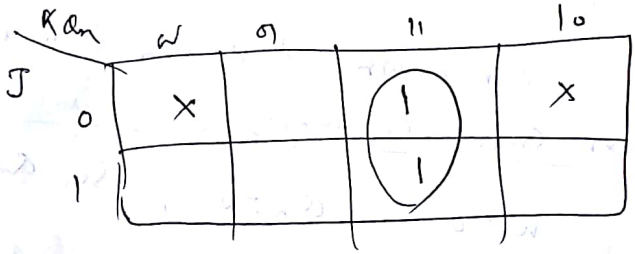
→ Now, we have J & K i/ps, o/p is S & R. To find S ?

Minimize the K-map for 'S'

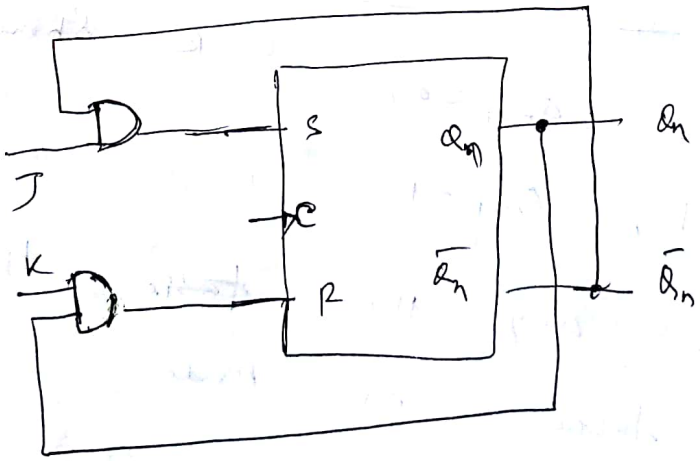


$S = J\bar{Q}_n$

K-Map for 'R'



$R = KQ_n$.



Note:- Total possible permutation.

$$^4P_2 = \frac{4!}{2!} = \frac{24}{2} = 12.$$

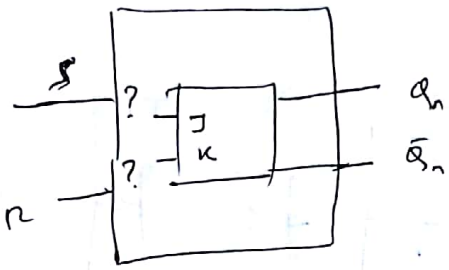
i.e. SR → JK, D, T

JK → SR, D, T

D → SR, T, JK

T → SR, JK, D

2) JK to SR



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	X	1
1	0	0	1	1	X
1	0	1	1	X	0
1	1	0	X	X	X
1	1	1	X	X	X

→ From Excitation table of J-K F/F

Because when S=1, R=1.

It produces invalid o/p.

J

Q_n \ SR	00	01	11	10
0		X	X	
1	1	X	X	X

$J = S$

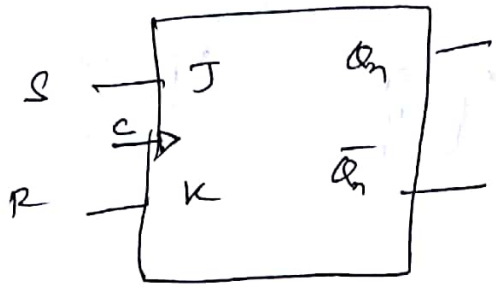
K

Q_n \ SR	00	01	11	10
0	X		1	X
1	X		X	X

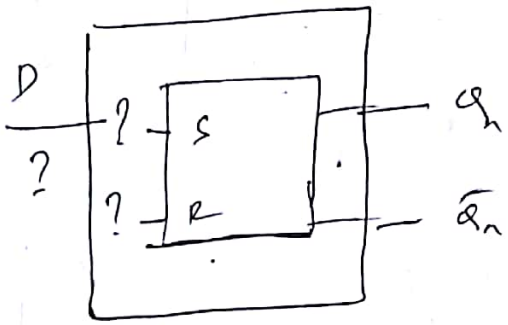
$K = R$

Since

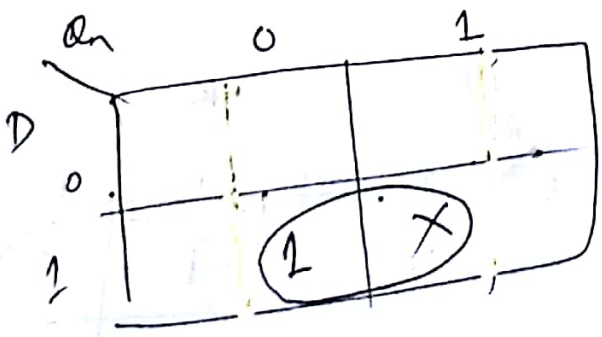
$J = S$ & $K = R$



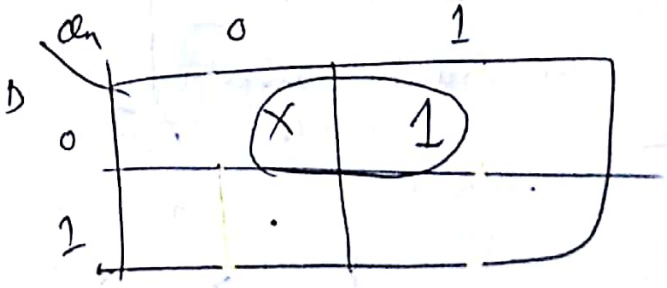
3) SR to D



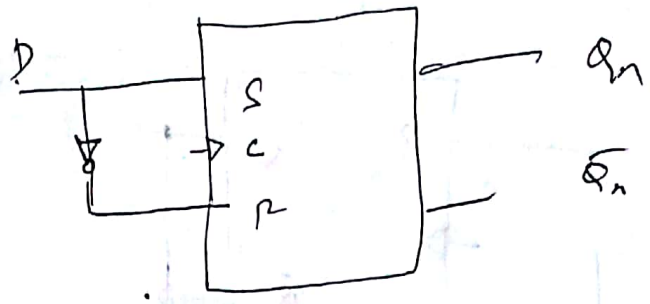
External I/P	Present State	next State	F/F i/p	
D	Q_n	Q_{n+1}	S	R
0	0	0	0	X
0	1	0	0	1
1	0	1	1	0
1	1	1	X	0



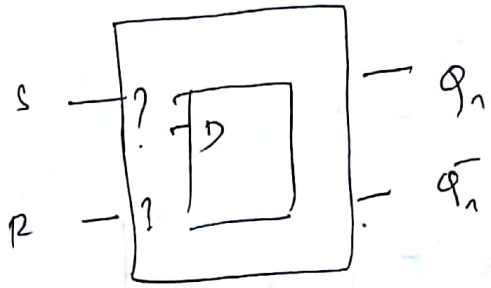
$S = D$



$R = \bar{D}$



4) **D → SR**



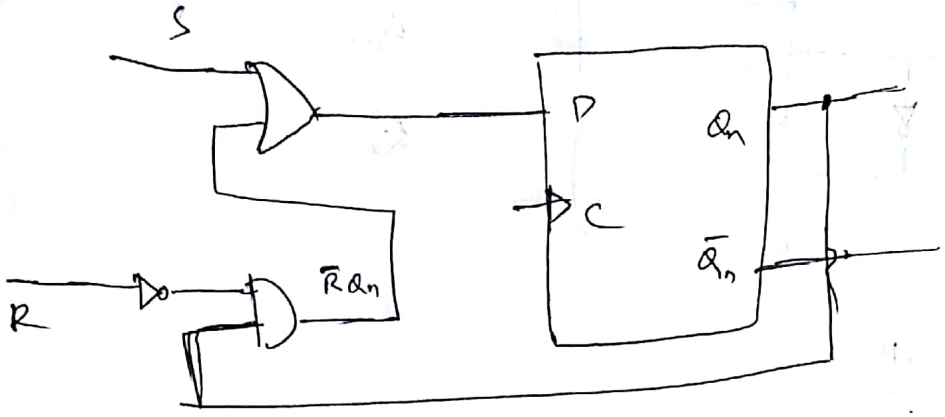
S	R	Q_n	Q_{n+1}	D
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	1	1
1	0	1	1	1
1	1	0	X	X
1	1	1	X	X

Invalid

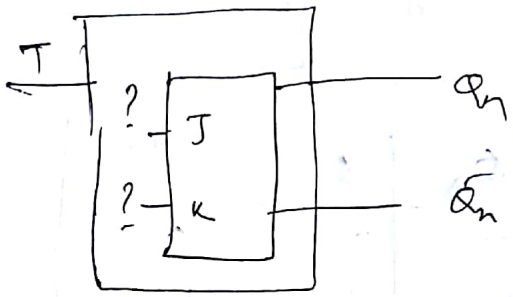
RQn	00	01	11	10
S				
0		1		
1	1	1	X	X

$D = S + \bar{R} Q_n$

c)



5) JK to T



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

N.C no change

Toggle

8

J

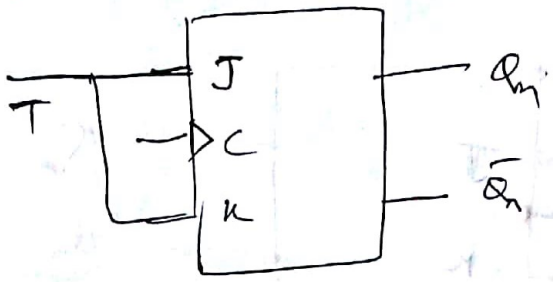
T \ Q_n	0	1
0		X
1	1	X

$J = T$

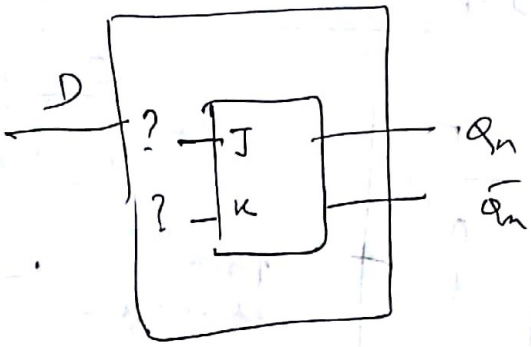
K

T \ Q_n	0	1
0	X	
1	X	1

$K = T$

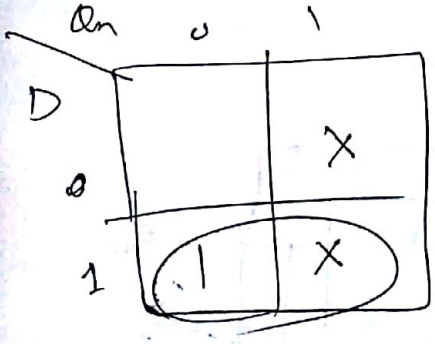


6) J-K to D



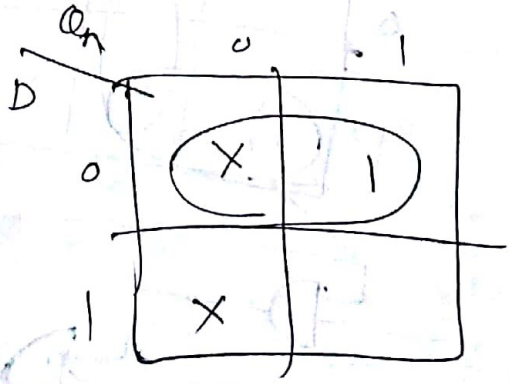
D	Q_n	Q_{n+1}	J	K
0	0	0	0	X
0	1	0	X	1
1	0	1	1	X
1	1	1	X	0

J

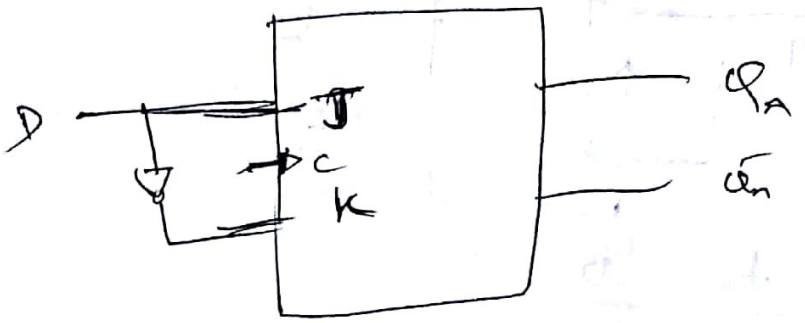


$J = D$

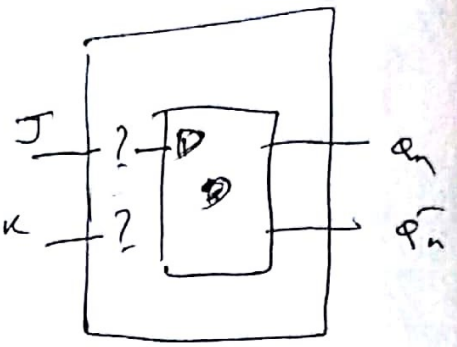
K



$K = \bar{D}$

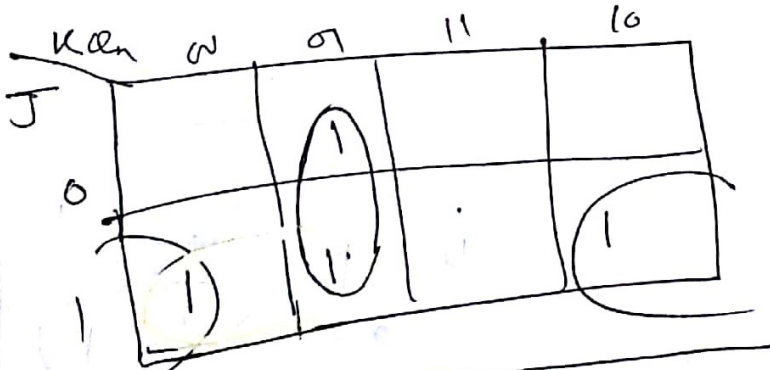


7) D to J-K

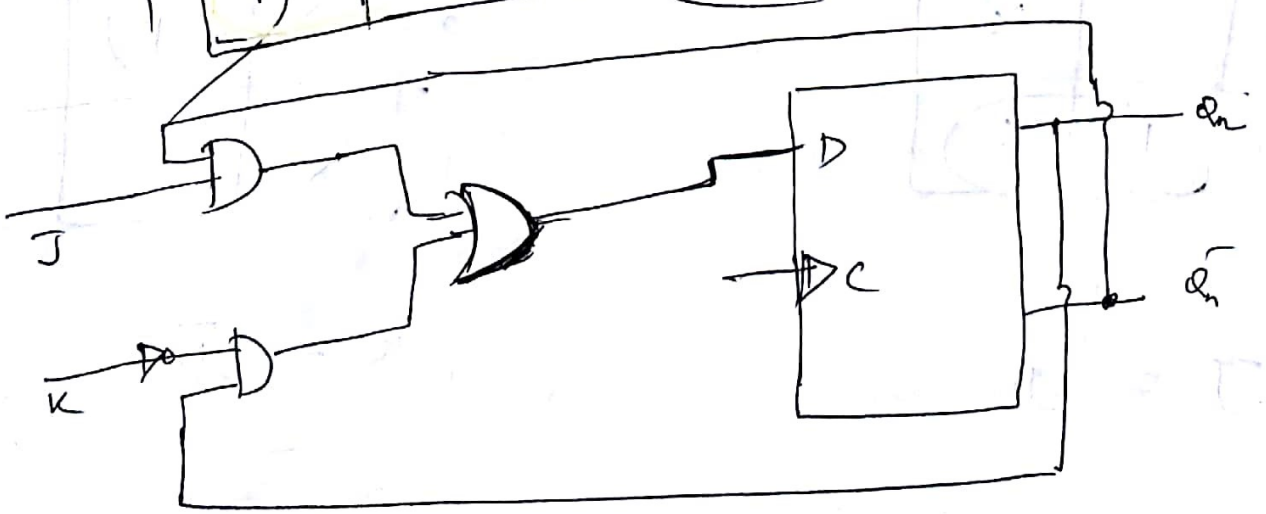


J	K	Q_n	Q_{n+1}	D
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0

→ from excitation table of D F/F



$D = J\bar{Q}_n + \bar{K}Q_n$



Similarly Other F/F Conversions Can be done.

Finite State Machine: (FSM)

→ A finite state machine is an abstract model describing the synchronous sequential machine. The behaviour of a finite state machine is described as a sequence of events that occurs at discrete instant designated as $t = 1, 2, 3$ --- etc.

→ Suppose that the machine has been receiving I/P signals and also responding by producing O/P signals.

→ If, now, at time t, we were apply an I/P signal $x(t)$ to the machine, the response $Z(t)$ would depend on $x(t)$ as well as on the past I/Ps to the machine and since a given machine might have an infinite Varieties of possible histories, it would need infinite capacity for storing them.

→ Since in practice, it is impossible to implement machines which have infinite storage capabilities, we will Concentrate on those machines whose past histories can affect their future behaviour only in finite number of ways. These are called the finite state machines, i.e., machine with fixed number of states.

→ These machine can distinguish among a finite number of classes of I/P histories. These classes of I/P histories are referred to as the internal states of machine. Every FSM, therefore, contains a finite number of memory devices.

State Diagram

Note: Sequential CKts are also called ✓ FSMs. This name is due to the fact that the functional behaviour of these circuits can be represented using finite number of states.

✓ Ex:- Sequence detector

State Diagram:-

The state diagram or state graph is a pictorial representation of the relationship among the present state, the I/P, the next state and O/P of sequential CKt, i.e the state diagram is a pictorial representation of the behaviour of a sequential circuit.

Figure 1 (a) shows the state diagram of a Mealy circuit. The state is represented by circle also called the node or vertex and the transition between states

is indicated by direct lines connecting the circles.

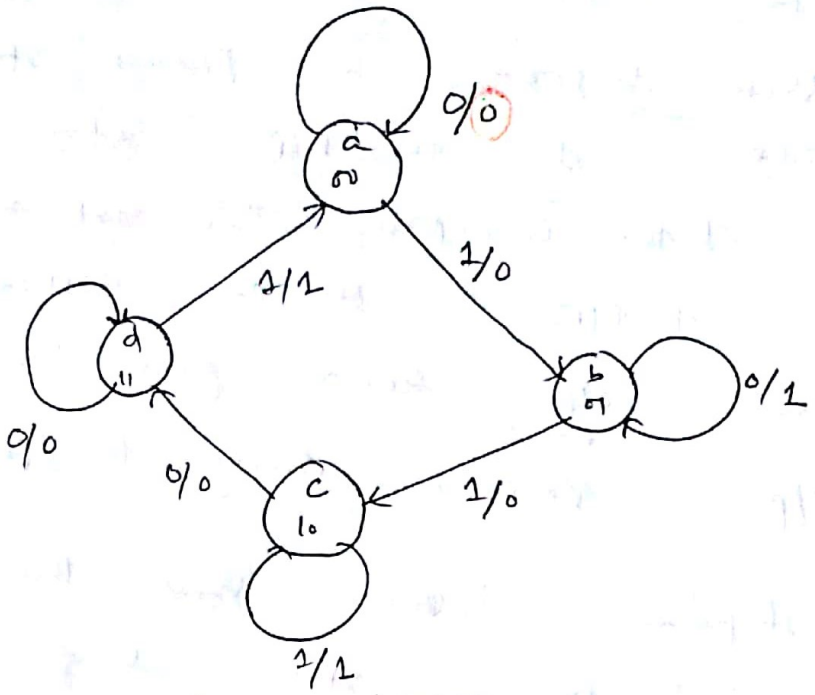


Fig 1: (a) State diagram

→ A directed line connecting a circle with itself indicates that the next state is same as the present state.

→ The binary number inside each circle identifies the 'state' represented by the circle.

→ The directed line are labelled with two binary numbers separated by a '/' (Slash). The I/P value that causes state transition is labelled first and the O/P value that occurs when this I/P is applied during present state is labelled after the symbol (Slash) '/'.

State Table :-

The State table is a tabular representation of the state diagram. The present state designates the state of the F/F before the application of the clock pulse. The next state is the state of F/F after the application of clock pulse & o/p section gives the values of the o/p variable during the present state.

The figure 1 (b) shows the state table for the state diagram shown in figure 1 (a).

Present State	Next State, x=0	Next State, x=1	o/p, x=0	o/p, x=1
a	a	b	0	0
b	b	c	1	0
c	d	c	0	1
d	d	a	0	1

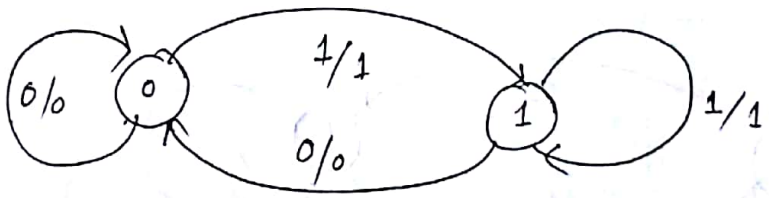
Memory elements -

1) D- F/F

The excitation table, state diagram & state table of a D F/F are shown in figure 2 (a), 2 (b) & 2 (c) respectively.

present State	I/p to FF	Next State
Q(t)	D(t)	Q(t+1)
0	0	0
0	1	1
1	0	0
1	1	1

Fig 2(a): Excitation of D FIF.



(b) State diagram

State table

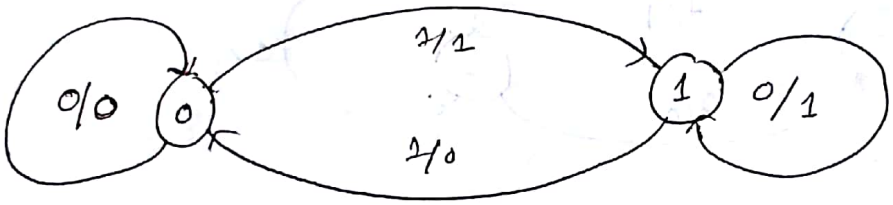
P. State	Nx. State		O/P	
	D=0	D=1	D=0	D=1
0	0	1	0	1
1	0	1	0	1

(c) State table.

2) T F/F

Present State	I/p to F/F	Next State
0	0	0
0	1	1
1	0	1
1	1	0

(a) Excitation table.



(b) State diagram

P. state	Next state		O/P	
	T = 0	T = 1	T = 0	T = 1
0	0	1	0	1
1	1	0	1	0

(c) State Table

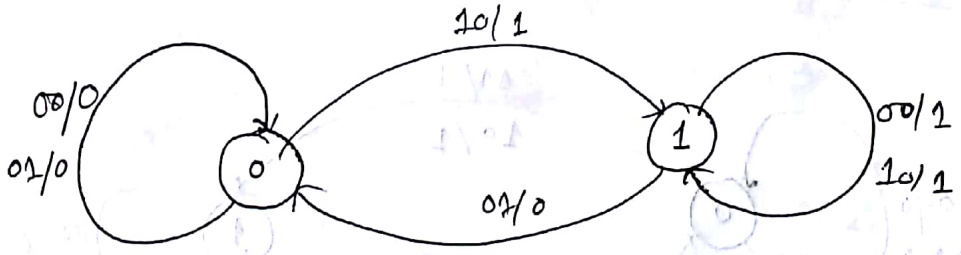
3) SR FIR

Present State	I/P to FF		Next State
Q_t	S	R	Q_{t+1}
0	0	0	0
0	0	1	0
1	1	0	1
1	0	0	1
1	0	1	0
1	1	0	1
0	1	1	×
1	1	1	×

S=1, R=1 Invalid

S=1, R=1 Invalid

(a) Excitation Table



(b) State diagram.

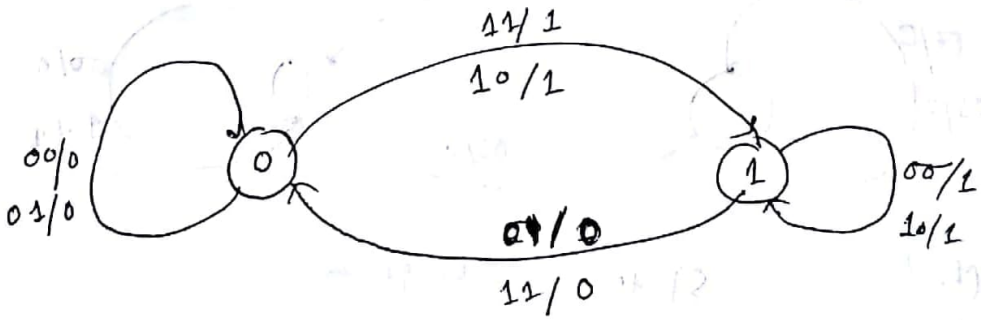
P. S	N.S			O/P		
	SR 00	SR 01	SR 10	SR 00	SR 01	SR 10
0	0	0	1	0	0	1
1	1	0	1	1	0	1

4/ J-K FF

Excitation Table

Present State $Q(t)$	I/P to FF J	K	Next State $Q(t+1)$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

State Diagram



State Table

Present State	Next State JK 00	JK 01	JK 10	JK 11	O/P JK 00	JK 01	JK 10	JK 11
0	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0

The Sequence Detector

Step 1 :- Word Statement of the problem:

A sequence detector is a sequential machine which produces an o/p '1' everytime the desired sequence is detected and o/p '0' at all other time.

Suppose we want to design a sequence detector to detect a sequence 1010 and say that overlapping is permitted i.e. for example, if the I/P sequence is 01101010 the corresponding output sequence is 00000101. We can start the synthesis procedure by constructing the state diagram of the machine.

Step 2 :- State diagram & State table.

The state diagram of sequence detector is shown in fig 1. At time t_1, the machine is assumed to be in the initial state designated arbitrarily as 'A'.

While in this state, the machine can receive first I/P either a '0' or a '1'.

If the i/p bit is a 0, the machine does n't start the detection processes because the first bit in the desired sequence is a 1.

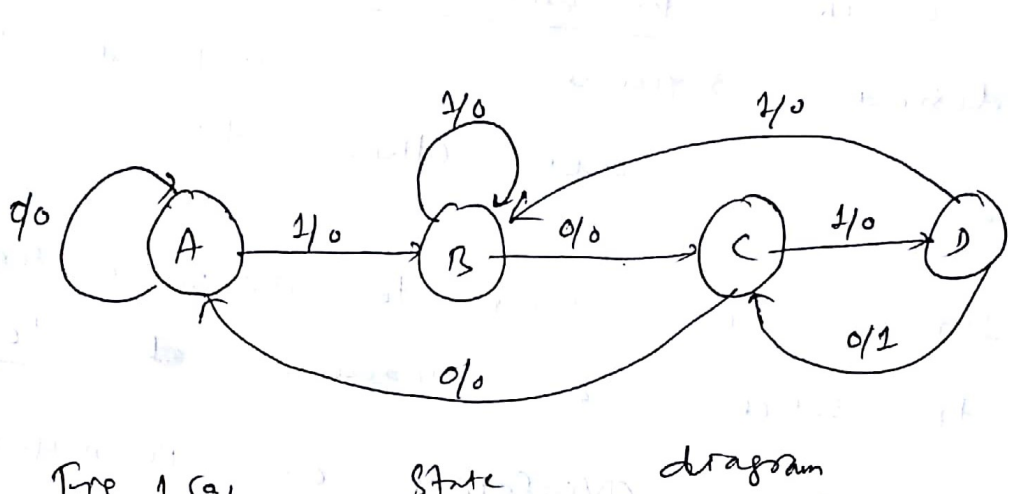


Fig 1(a) State diagram

→ So, it remains in the same state and o/ps a '0'. Hence an arc labelled 0/0 starting and terminating at A is drawn.

If the i/p bit is 1, the detection process starts. So the machine goes to state B and o/ps a '0'. Hence an arrow labelled 1/0 starting at A & terminating at B is drawn.

While in state B, the machine may receive a '0' or a '1' bit. If the bit is a '0', the machine goes to the next state, say state 'C', because the

previous two bits are '10' which are ~~this~~ a part of valid sequence, and o/p is a '0'. So a fine d/o from B to C.

If the bit is a 1, the two bits becomes 11 and this is not a part of the valid sequence.

Since overlapping is permitted the second 1 may be used to start the sequence, so the machine remains in state B only and outputs a '0'. If is shown in by arc 1/0 starting and terminating at B.

While in state C, the machine may receive a 0 or a 1 bit. If it receives a 0, the last three bits received will be 100 and this is not a part of valid sequence and also none of the last two bits can be used to start the new sequence.

So the machine goes to state 'A' to restart the detection process (flashing detector) and outputs a 0.

Hence, an arc labeled 0/0 starting at C and terminating at A is drawn.

If it receives a '1' bit, the last three bit will be 101 which are a part of the valid sequence. So, the machine goes to the next state, say state D, and o/ps a '0'. So an arc labelled 1/0 is drawn from C to D.

While in state D, machine may receive a 0 or a 1. If it receives a '0', the last 4 bits becomes 1010 which is a valid sequence and the machine o/ps a 1. Since overlapping is permitted, the machine can utilize the last two bits 10 to get another 1010 sequence. So, the machine goes to state C. [Note:- If overlapping is not permitted or the machine has to restart after outputting a 1, the machine goes to state A].

→ So an arc labelled 0/1 is drawn from D to C. If it receives a 1 bit, the last 4 bits received will be 1011 which is not a valid sequence. So, the machine o/ps a '0'. Since the 4th bit 1 can become the starting bit

for next sequence, the machine goes to state 'B'. So, an arc labelled 1/0 is drawn from D to B.

Step 3:- State table from state diagram

Present State	Next State		O/P	
	X=0	X=1	X=0	X=1
A	A	B	0	0
B	C	B	0	0
C	A	D	0	0
D	C	B	1	0

Step 4:- State Assignment

There are 4 states, therefore, two state variables are required. Two state variable can have maximum 4 states. Assign the states arbitrarily. Let A → 00, B → 01, C → 10, D → 11. (means A is 0th state, B is 1st state, C = 2nd state, D = 3rd state) (it is nothing to do with sequence detector) be the state assignment. With this the state table becomes. (Transition & output table) i.e called

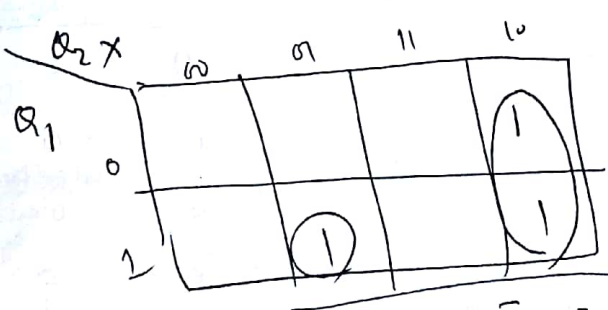
Present State	Next State		O/P	
	X=0	X=1	X=0	X=1
00	00	01	0	0
01	10	01	0	0
10	00	11	0	0
11	10	01	1	0

Step-5 From the previous table write down P.S, N.S & I/P. and Choose type of F/F & form the excitation table.

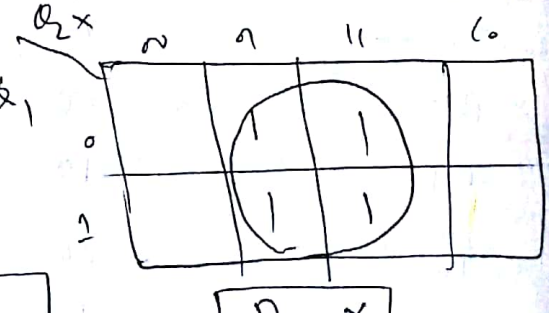
Present State		I/P	Next State		I/P to FFs		O/P
					$Q_1 \to D_1$	$Q_2 \to D_2$	
Q_1	Q_2	X	q_1	q_2	D_1	D_2	Z
0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	0
0	1	0	1	0	1	0	0
0	1	1	0	1	0	1	0
1	0	0	0	0	0	0	0
1	0	1	1	1	1	1	0
1	1	0	1	0	1	0	1
1	1	1	0	1	0	1	0

We have Chosen D F/F & excitation table of D F/F, the D_1 & D_2 are made. [Comparing ~~Q_1~~ present & next state of Q_1 & " " of Q_2]

Step-6 :- Use K Map & minimise expression

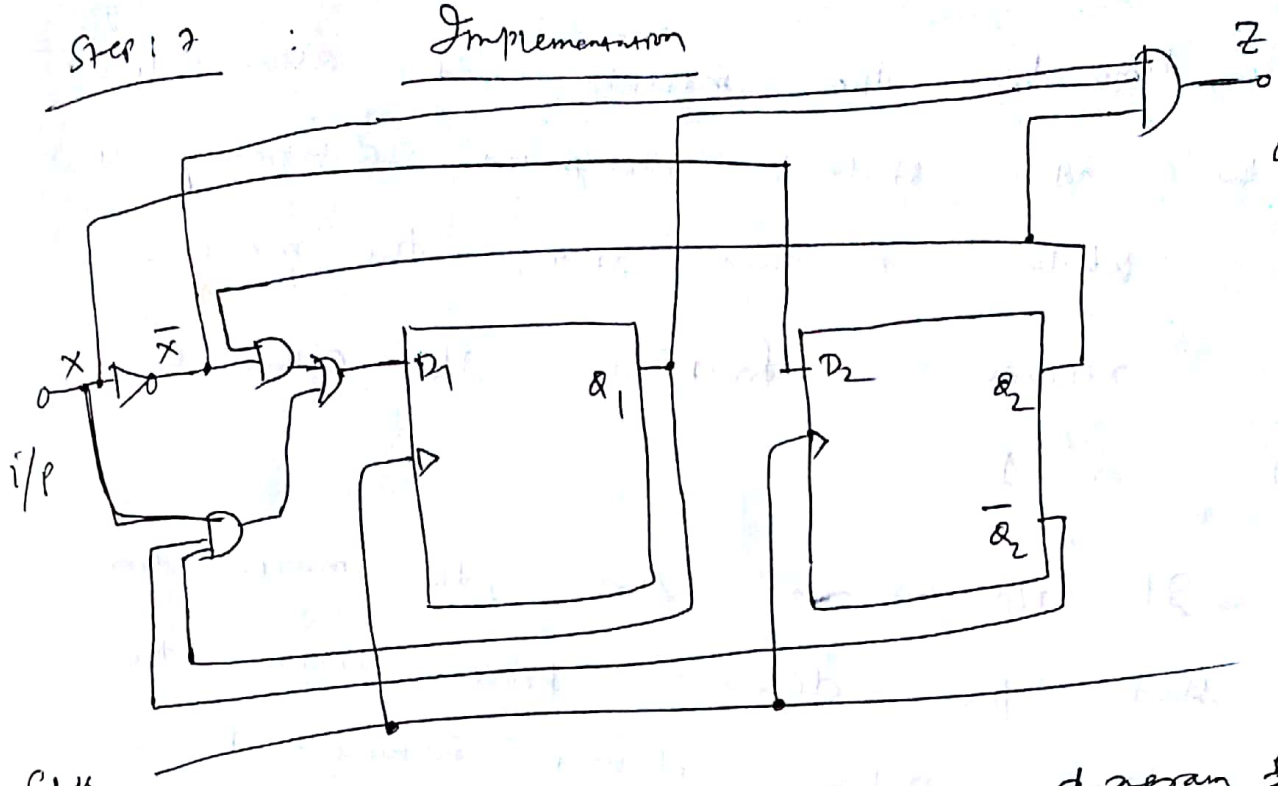


$D_1 = Q_2\bar{X} + Q_1\bar{Q}_2X$



$D_2 = X$

Step : 2 : Implementation



$Z = Q_1 Q_2 \bar{X}$

fig:- Logic diagram of the sequence (1010) detector using 'D' Flipflop.

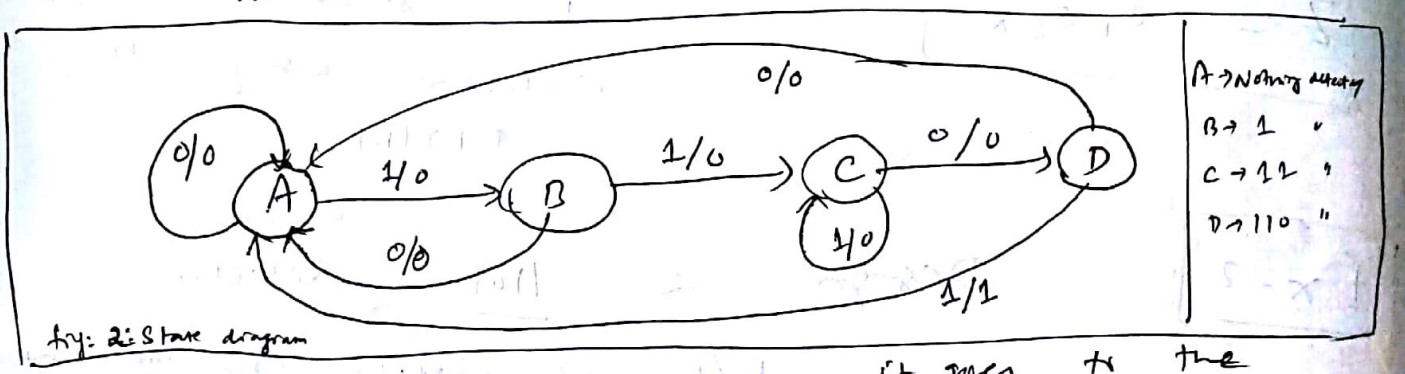
Ex-2 :- Design a 1101 sequence detector using JK F/F & overlapping is not permitted.

Ans : Step 1 :- We want to design a 1101 sequence detector which detects 1101 and produces o/p '1' when 1101 is detected. Here overlapping is not allowed. So once a detection is done it wont retain the bits of 1101 to detect another 1101. It will start from scratch to detect another 1101 sequence.

Step 2 :- Design a state diagram

At time t_1, the machine is assumed to be in the initial state designated arbitrarily as 'A'. While in this state, the machine can receive first bit of ip either a '0' or 1.

If ip bit is zero, the machine does not start the detection process because the first bit in the desired sequence is 1. So it remains in the same state 'A'. & o/p is '0'.



If ip bit 1 it goes to the next state B. & results the o/p Zero. (Since 1101 has not detected yet).

At State B, if ip is Zero the sequence is 10 (Invalid). Neither 0 or 10 can be utilized for further detection of 1101. (Since till now 1101 has not detected, 1st time, after initialization can be made) So the machine goes to State 'A' [Nothing detected] & o/p is '0'.

At State B, if i/p is 1 the sequence is 11 [valid]. So it

goes to the next state 'C'. & o/p is '0'. 229

While at State 'C', if i/p is '0' the sequence become 110. [valid] So it goes to next state 'D'.

But if the i/p is '1', the sequence becomes 111 [Invalid]. Since 1101 has not detected yet, ~~I~~ I can take overlapping or utilized the bits to detect 1101.

From 111, 111 x not valid
|| (valid)

So I can utilized 11 & expect 0 or 1 in the subsequent sequence.

So ~~from State C~~ the machine remains at state 'C' ['c' means '11' detected] & o/p is '0'

At State 'D' if i/p is '0' the sequence becomes 1100 [Invalid].

Since valid sequence is not detected yet lets check whether bits can be utilized.

11 or x	Because our desired sequence 1101
1 or x	
or x	
0 x	

So the machine goes to State 'A' i.e nothing detected, & o/p is '0'.

If i/p is '1', then the

sequence becomes 1101. If the sequence is detected o/p is 1.

Since overlapping is not allowed & the machine goes to initial state A & restart the process.

Fig 2:- Show the state diagram of the entire process discussed. From the state diagram state table can be formed.

Step 3:- State Table

Present State	Next State x=0	Next State x=1	o/p x=0	o/p x=1
A	A	B	0	0
B	A	C	0	0
C	D	C	0	0
D	A	A	0	1

Step 4 State Assignment [4 states : Less bits
A → 00, B → 01, C → 10, D → 11]

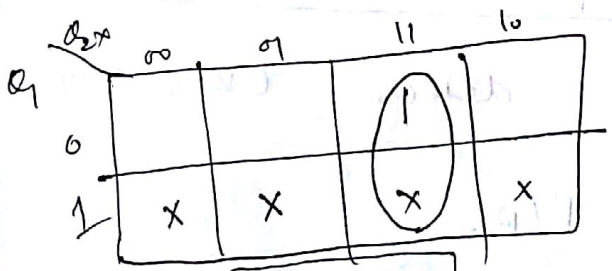
Present State	Next State x=0	Next State x=1	o/p x=0	o/p x=1
00	00	01	0	0
01	00	10	0	0
10	11	10	0	0
11	00	00	0	1

Step-5 : From previous table write P.S, I/P & Next State & choose type of F/F & form excitation table.

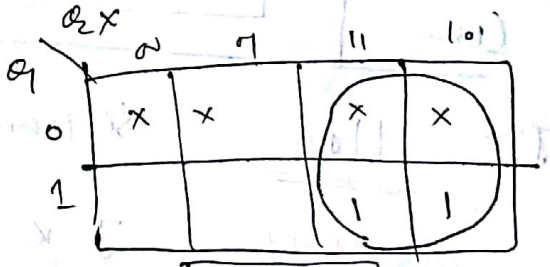
Given : JK R/F.

Present State		I/P	Next State		I/P to FF				O/P (Z)
		X							
Q_1	Q_2		Q_1	Q_2	J_1	K_1	J_2	K_2	
0	0	0	0	0	0	x	0	x	0
0	0	1	0	1	0	x	1	x	0
0	1	0	0	0	0	x	x	1	0
0	1	1	1	0	1	x	x	1	0
1	0	0	1	1	x	0	1	x	0
1	0	1	1	0	x	0	0	x	0
1	1	0	0	0	x	1	x	1	0
1	1	1	0	0	x	1	x	1	1

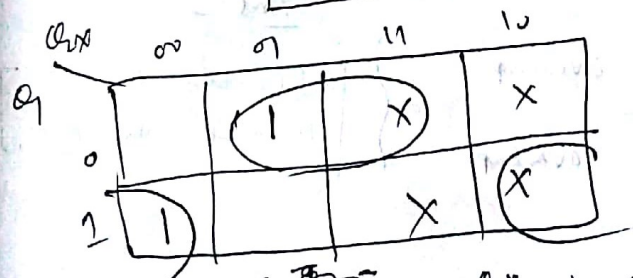
Step 6 :- Use K map & find the minimal expression.

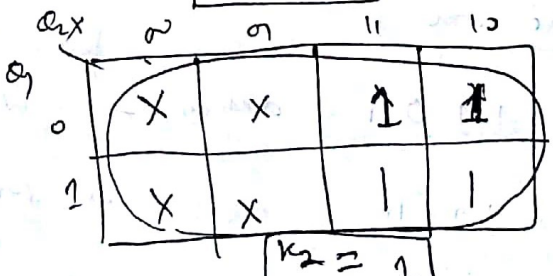


$J_1 = Q_2X$



$K_1 = Q_2$

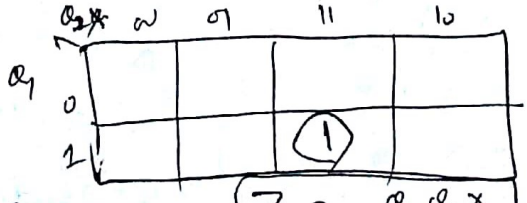




$K_2 = 1$

$J_2 =$

$J_2 = \bar{Q}_1 X + Q_1 \bar{X}$
$= Q_1 \oplus X$



$Z = Q_1 Q_2 X$

Step : 7 Implementation

$J_1 = Q_2 x$, $K_1 = Q_2$

$J_2 = \bar{Q}_1 x + Q_1 \bar{x}$, $K_2 = 1$

$Z = Q_1 Q_2 x$

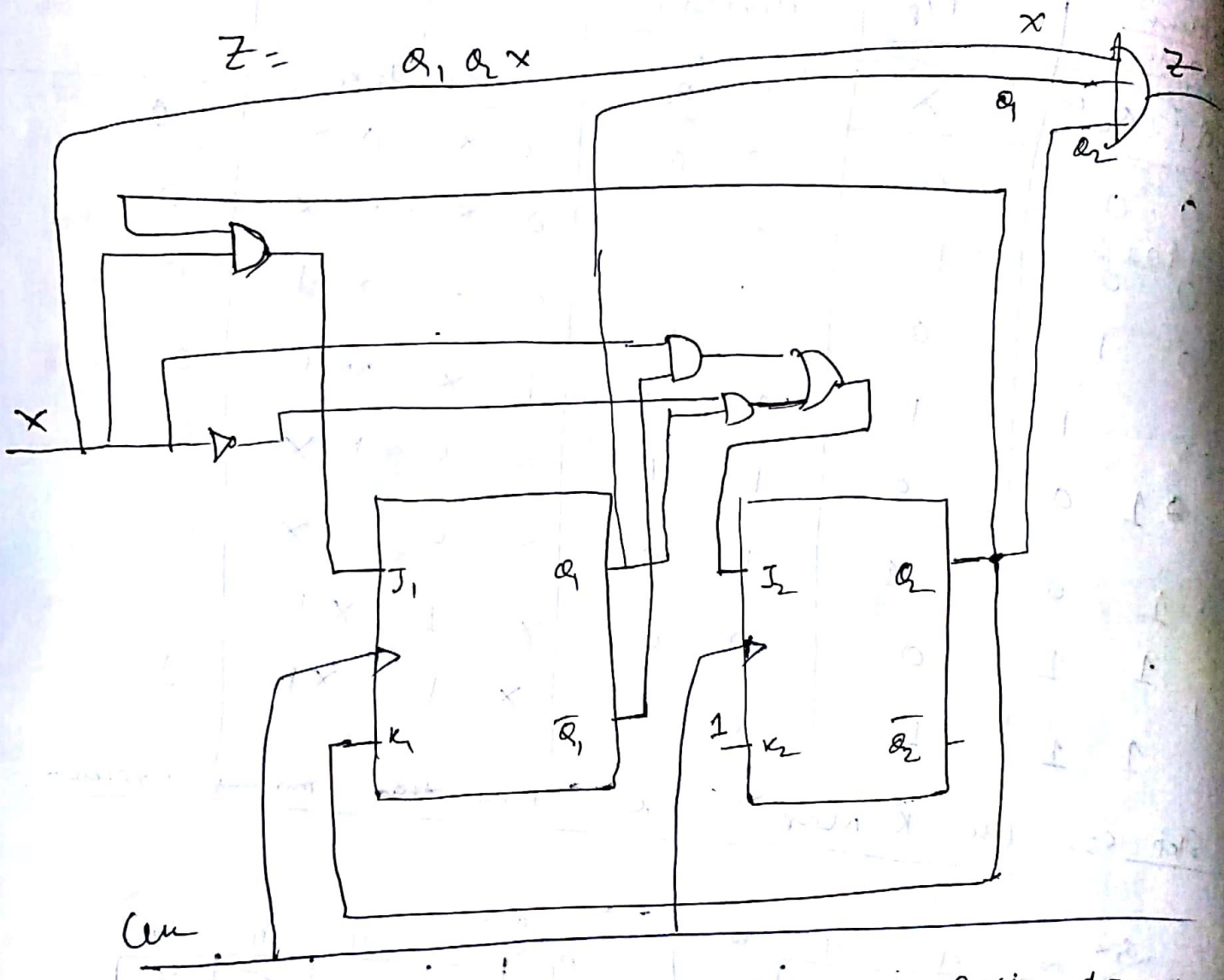


fig:- 1101 sequence detector ckt diagram using J K F/F -

Home work

1) 1101 detector — with overlap
2) 1101 " → without overlap

Ans :- [Search in internet. Ans given.]

5) Design of an SOP circuit to detect the Decimal numbers 5 through 12 in a 4-bit Gray Code I/P (Page 312) Anand Kumar

Ans:- The I/P to the SOP circuit is a 4 bit Gray code. Let the I/P Gray Code be ABCD. There are 16 Possible Combinations of 4-bit Gray Code. All of them are valid & hence there are no don't Cares.

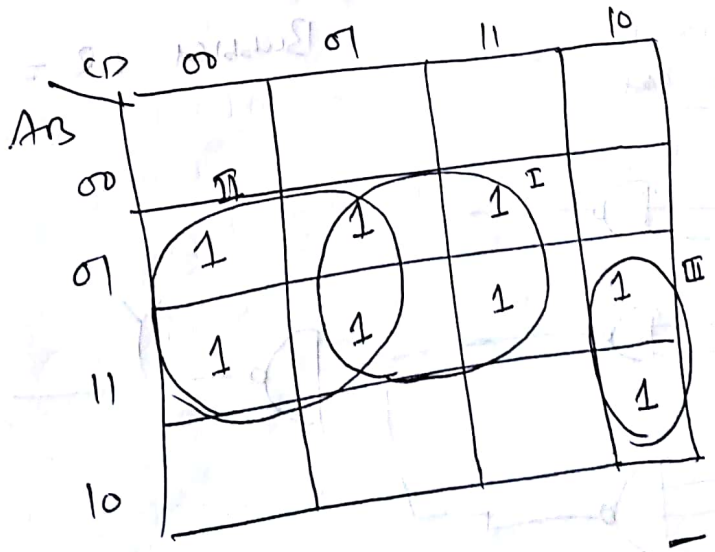
We have to find the Gray code of 0 to 15 decimal number.

for example	Binary Code	Gray code
2 →	0010	0011
3 →	0011	0010

The truth table of SOP circuit is shown.

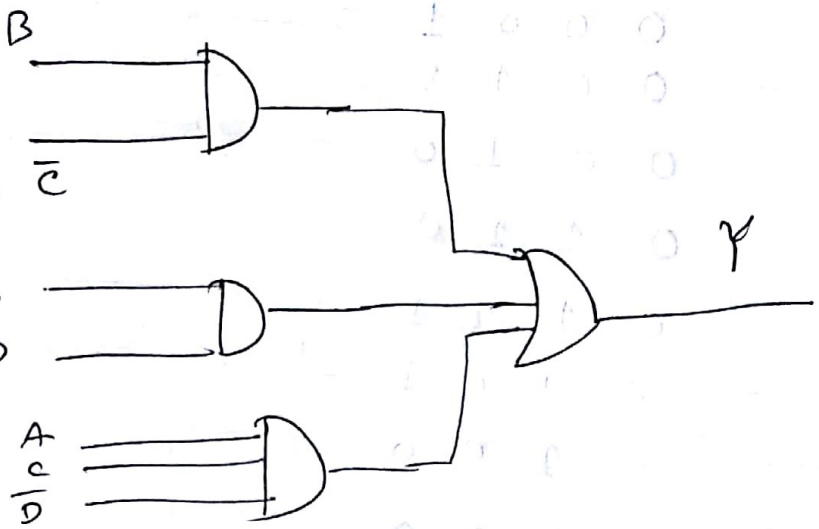
For Decimal number 5 to 12 O/P is 1 and rest of the Cases O/P is Zero.

Decimal Number	4 bit Gray Code A	B	C	D	O/P (Y)
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	1	0
3	0	0	1	0	0
4	0	1	1	0	0
5	0	1	1	1	1
6	0	1	0	1	1
7	0	1	0	0	1
8	1	1	0	0	1
9	1	1	0	1	1
10	1	1	1	1	1
11	1	1	1	0	1
12	1	0	1	0	1
13	1	0	1	1	0
14	1	0	0	1	0
15	1	0	0	0	0

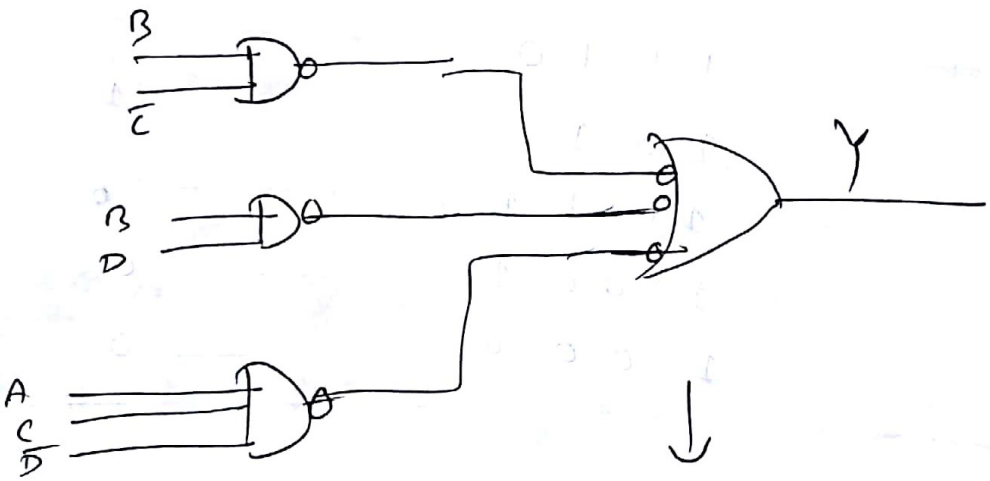


$$Y = I + II + III = BD + B\bar{C} + AC\bar{D}$$

$\therefore Y = B\bar{C} + BD + AC\bar{D}$



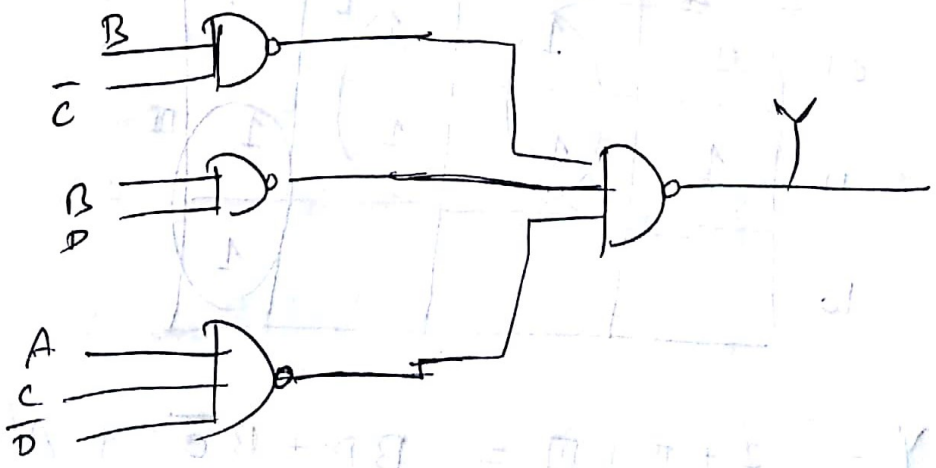
OR Using NAND gate only



Bubbled OR = NAND

Final Ckt

$\therefore$



Priority Encoder

A priority encoder is an encoder circuit that includes the priority function. The operation of priority encoder is such that if two or more i/ps are equal to 1 at the same time, the i/p having the highest priority will take precedence.

The truth table of a 4-i/p priority encoder is given in Table 1.

Table 1:- [Higher the subscript no, higher is the priority i.e $D_3 > D_2 > D_1 > D_0$]

Truth table of a Priority Encoder

i/ps				o/ps		
D_0	D_1	D_2	D_3	A	B	V
0	0	0	0	x	x	0
1	0	0	0	0	0	1
x	1	0	0	0	1	1
x	x	1	0	1	0	1
x	x	x	1	1	1	1

Here, in addition to the o/ps A & B, the circuit has a third o/p designated by 'V'. This is a Valid bit indicator that is

Set to '1' when one or more i/ps are equal to 1.

If all i/ps are '0', there is no valid i/ps and 'V' equals to Zero. Therefore the other two o/ps are not inspected when (V=0) and are specified as don't care conditions. [A & B are X X in first case.]

'X' → Don't care.

According to the truth table, the ~~highest~~ higher the subscript number, the higher the priority of the i/p.

Input D_3 has the highest priority, so regardless of the values of other inputs, when this i/p is 1, the o/p for A B is 11 (binary 3). D_2 has the next priority level. The o/p is 10 if $D_2 = 1$, provided $D_3 = 0$, regardless of values of other two lower priority i/ps, [i.e D_0 & D_1 are don't cares].

The o/p for D_1 is generated only if higher priority i/ps are zero & so on down the priority levels.

The K-Map for simplifying o/ps A & B are shown in fig 1.

For 'A'

(i) A = 'x', when

D_0	D_1	D_2	D_3
0	0	0	0

D_0	D_1	D_2	D_3
x	x	1	0

[Refer Table 1]

(ii) A = 1, when

i.e

0 0	1 0
0 1	1 0
1 1	1 0
1 0	1 0

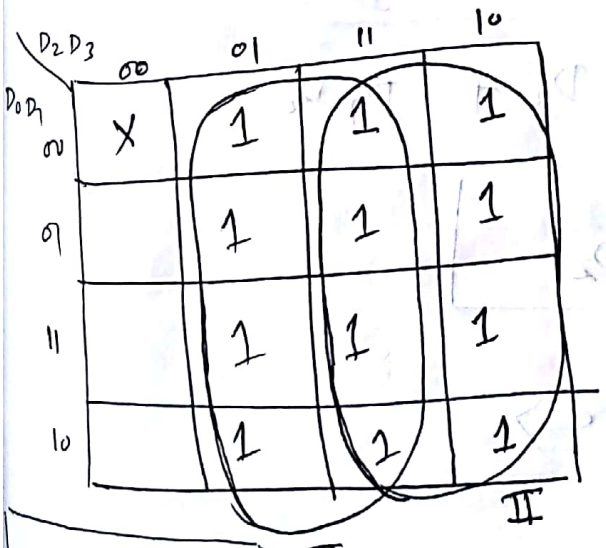
(iii) A = 1, when

D_0	D_1	D_2	D_3
x	x	x	1

i.e

0 0 0	1
0 0 1	1
0 1 0	1
0 1 1	1
1 0 0	1
1 0 1	1
1 1 0	1
1 1 1	1

A



take 'x' I

No need to take 'x' II because it is unnecessary.

Mapping all the 3 cases into the map.

Fig 1:- Maps for priority encoder

$A = D_3 + D_2$

or $\boxed{A = D_2 + D_3}$ —— (1)

B

Similarly for B the Cases are

(i) 0000 → 'x'

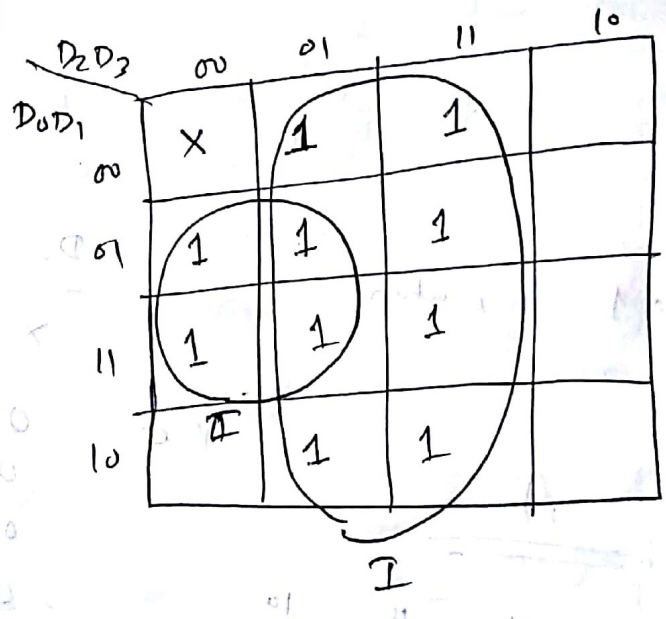
(ii) x100 → 1

i.e 0 100 → 1
1 100

(iii) xxx1 → 1

i.e
0001 1
0011 1
0101 1
0111 1
1001 1
1011 1
1101 1
1111 1

Mapping All the 3 Cases



$$B = I + II = D_3 + D_1\overline{D_2}$$

$$\boxed{B = D_3 + D_1\overline{D_2}}$$

For o/p V <The Cases are>

(i) 1000 → 1

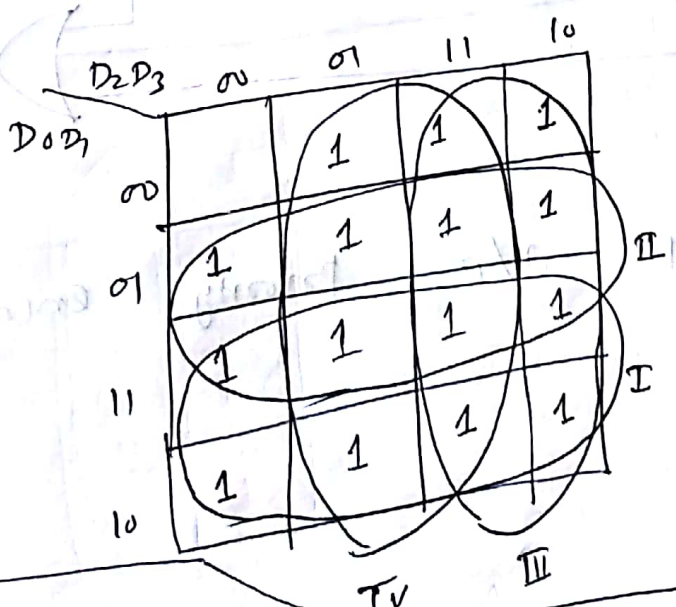
(ii) x100 → 1

i.e $\begin{cases} 0\ 1\ 00 \\ 1\ 1\ 00 \end{cases} \rightarrow 1$

(iii) $\begin{cases} X\ X\ 1\ 0 \\ 00\ \ 1\ 0 \\ 01\ \ 1\ 0 \\ 10\ \ 1\ 0 \\ 11\ \ 1\ 0 \end{cases} \rightarrow 1$

(iv) $\begin{cases} X\ X\ X\ 1 \\ 0\ 0\ 0\ 1 \\ 0\ 0\ 1\ 1 \\ 0\ 1\ 0\ 1 \\ 0\ 1\ 1\ 1 \\ 1\ 0\ 0\ 1 \\ 1\ 0\ 1\ 1 \\ 1\ 1\ 0\ 1 \\ 1\ 1\ 1\ 1 \end{cases} \rightarrow 1$

Matring all the 4 Cases.



$V = I + II + III + IV$

$V = D_0 + D_1 + D_2 + D_3$

∴ The condition for o/p 'V' is an 'OR' function of all the i/p variables.

The Priority encoder is implemented in fig 2. according to the Boolean functions

$$A = D_2 + D_3$$

$$B = D_3 + D_1 \overline{D_2}$$

$$V = D_0 + D_1 + D_2 + D_3$$

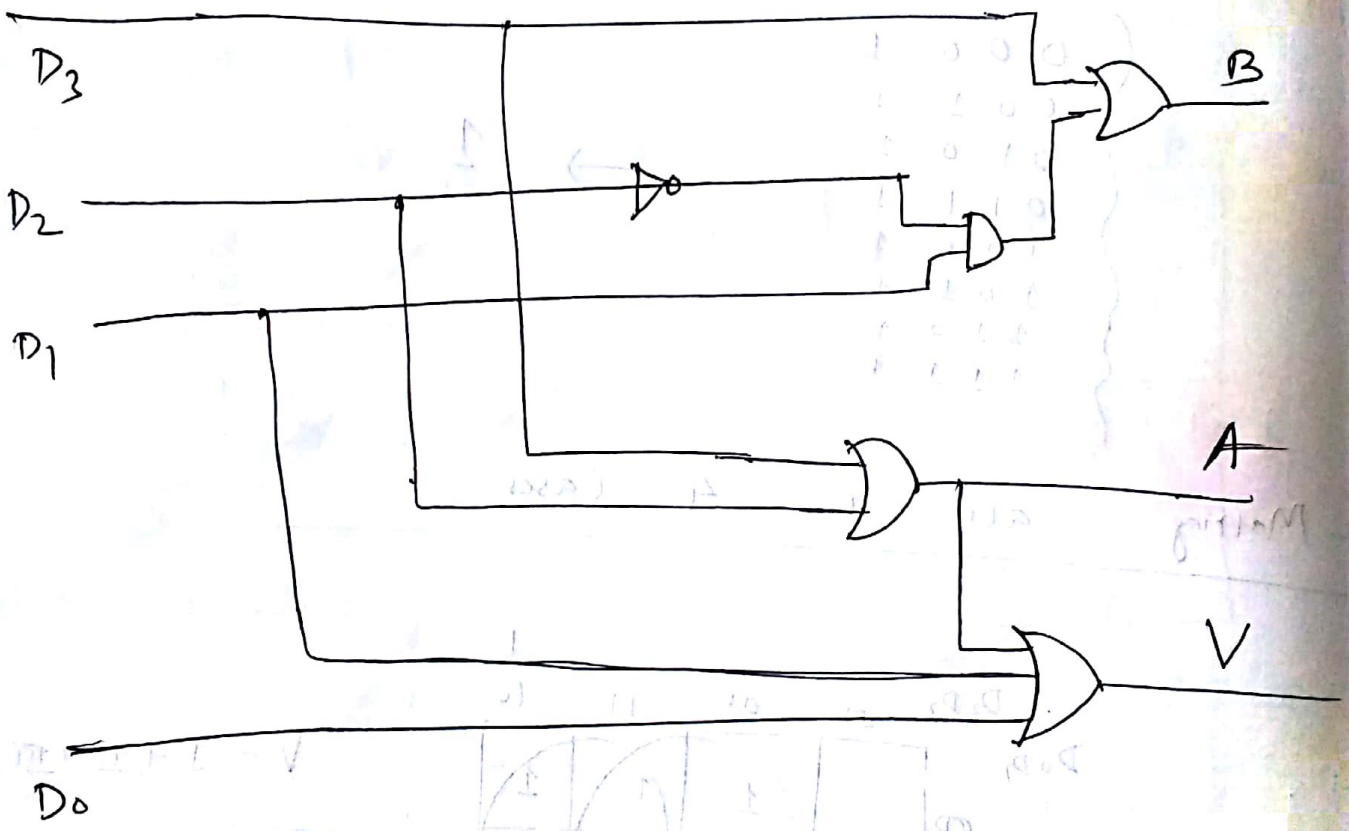


Fig 2:- 4 i/p priority encoder.

Q) Design an asynchronous counter which counts the sequence 1, 2, 3, 4, 1, 2, 3, 4, ——— like that.

that means it start with 1, count up to 4, when it comes 5 it returns again to 1.

Ans:- We require 3 T-FFs which can count 000 to 111.

Q_2	Q_1	Q_0		$\overline{PRE_2}$	$\overline{CLR_2}$	$\overline{PRE_1}$	$\overline{CLR_1}$	$\overline{PRE_0}$	$\overline{CLR_0}$
				x	x	x	x	x	x
0	0	0	→	1	1	1	1	1	1
0	0	1	→	1	1	1	1	1	1
0	1	0	→	1	1	1	1	1	1
0	1	1	→	1	1	1	1	1	1
1	0	0	→	1	1	1	0	0	1
1	0	1	→	1	0	x	x	x	x
1	1	0	→	x	x	x	x	x	x
1	1	1	→	x	x	x	x		

Note:- We will use asynchronous T/Fs $\overline{PRE}$ & $\overline{CLR}$ which are active low type.

$\overline{PRE}$	$\overline{CLR}$	Remark	Q
0	0	Not used	x
0	1	Set	1
1	0	Reset	0
1	1	Clocked operation	Normal T/F.

Explanation next page

When 5 Comes it should go to 1

i.e. 1 0 1
↓ ↓ ↓
0 0 1

i.e. $Q_2 \to 0$ (Reset)
$Q_1 \to 0$ (Reset)
$Q_0 \to 1$ (Set)

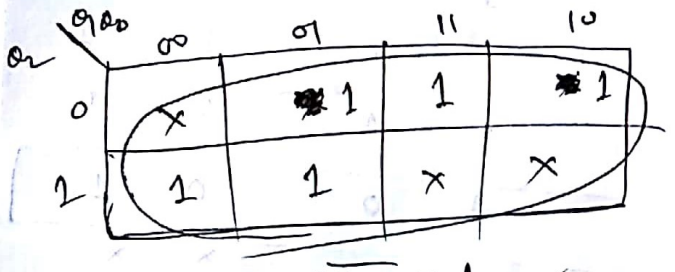
i.e. $\overline{PRE_2} = 1, \overline{CLR_2} = 0$

$\overline{PRE_1} = 1, \overline{CLR_1} = 0$

$\overline{PRE_0} = 0 \quad \overline{CLR_0} = 1$

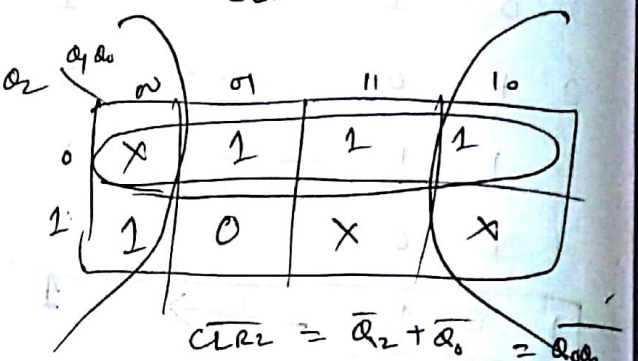
Solving K-Map for

$\overline{PRE_2}$



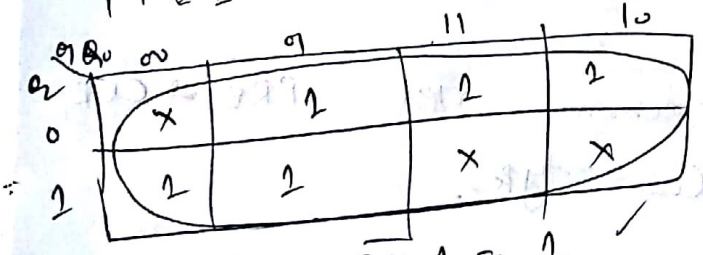
$\overline{PRE_2} = 1$

$\overline{CLR_2}$



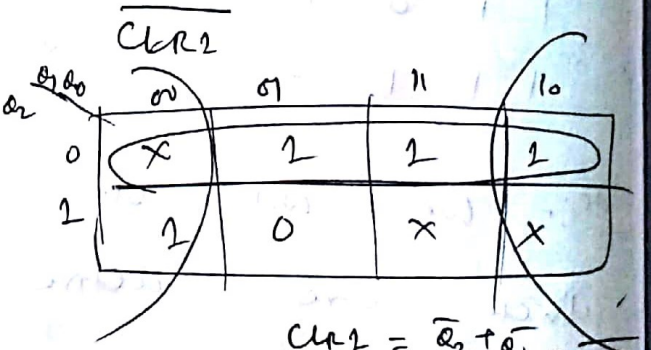
$\overline{CLR_2} = \overline{Q_2} + \overline{Q_0} = \overline{Q_0 Q_2}$

$\overline{PRE_1}$



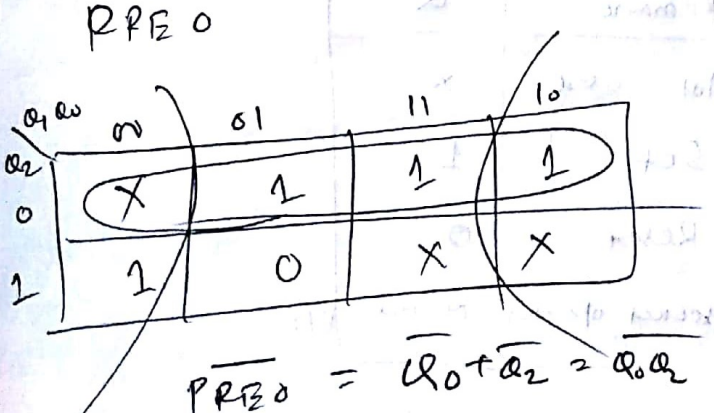
$\overline{PRE_1} = 1$

$\overline{CLR_2}$



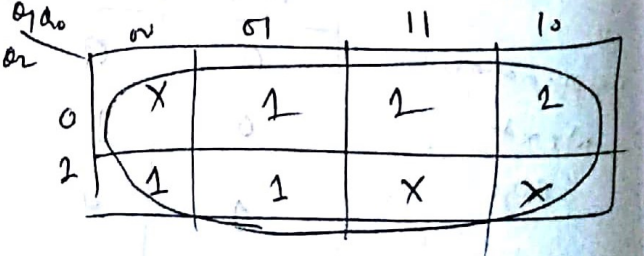
$\overline{CLR_2} = \overline{Q_2} + \overline{Q_0} = \overline{Q_0 Q_2}$

$\overline{PRE_0}$



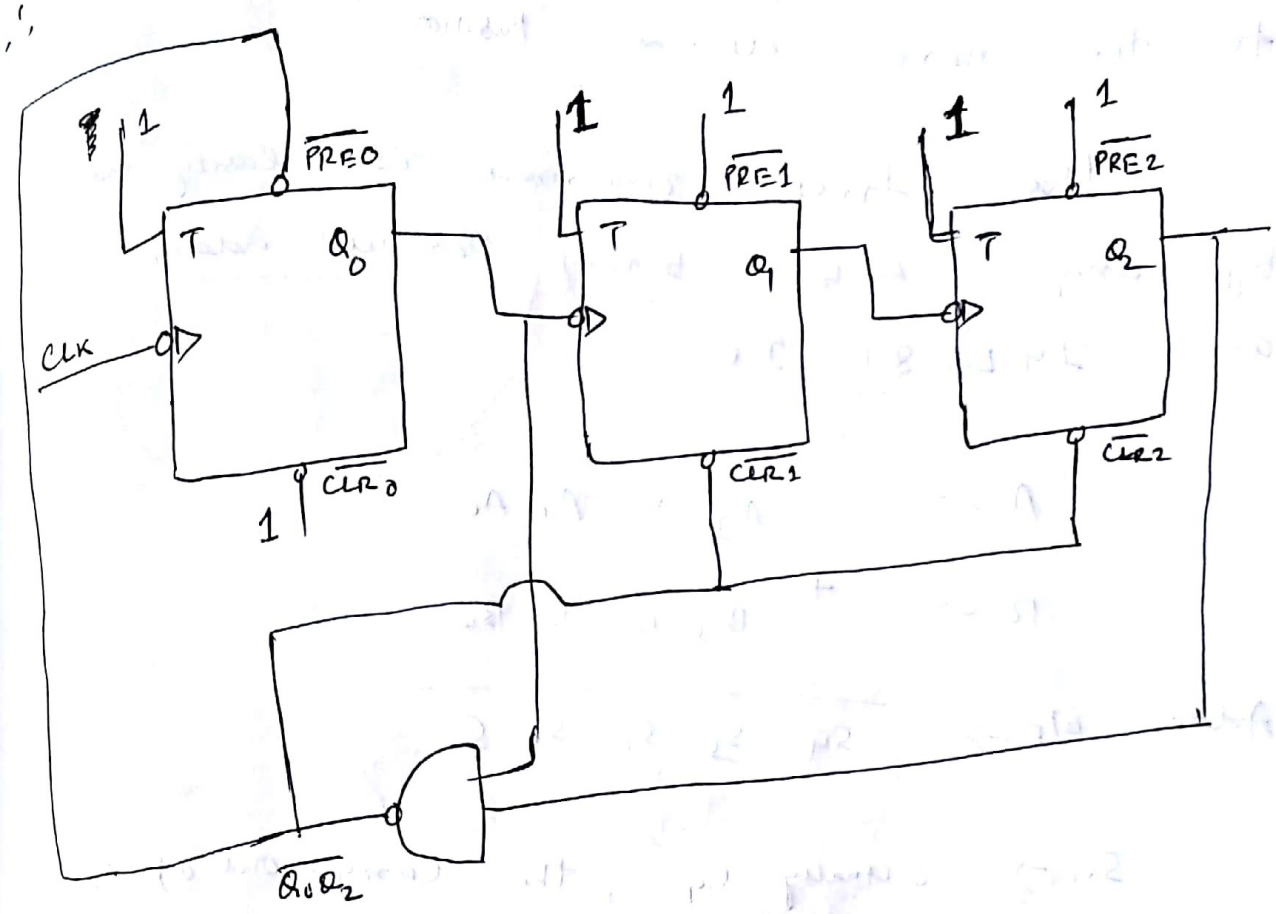
$\overline{PRE_0} = \overline{Q_0} + \overline{Q_2} = \overline{Q_0 Q_2}$

$\overline{CLR_0}$



$\overline{CLR_0} = 1$

The CLK for this Counter.



2) BCD Adder :—

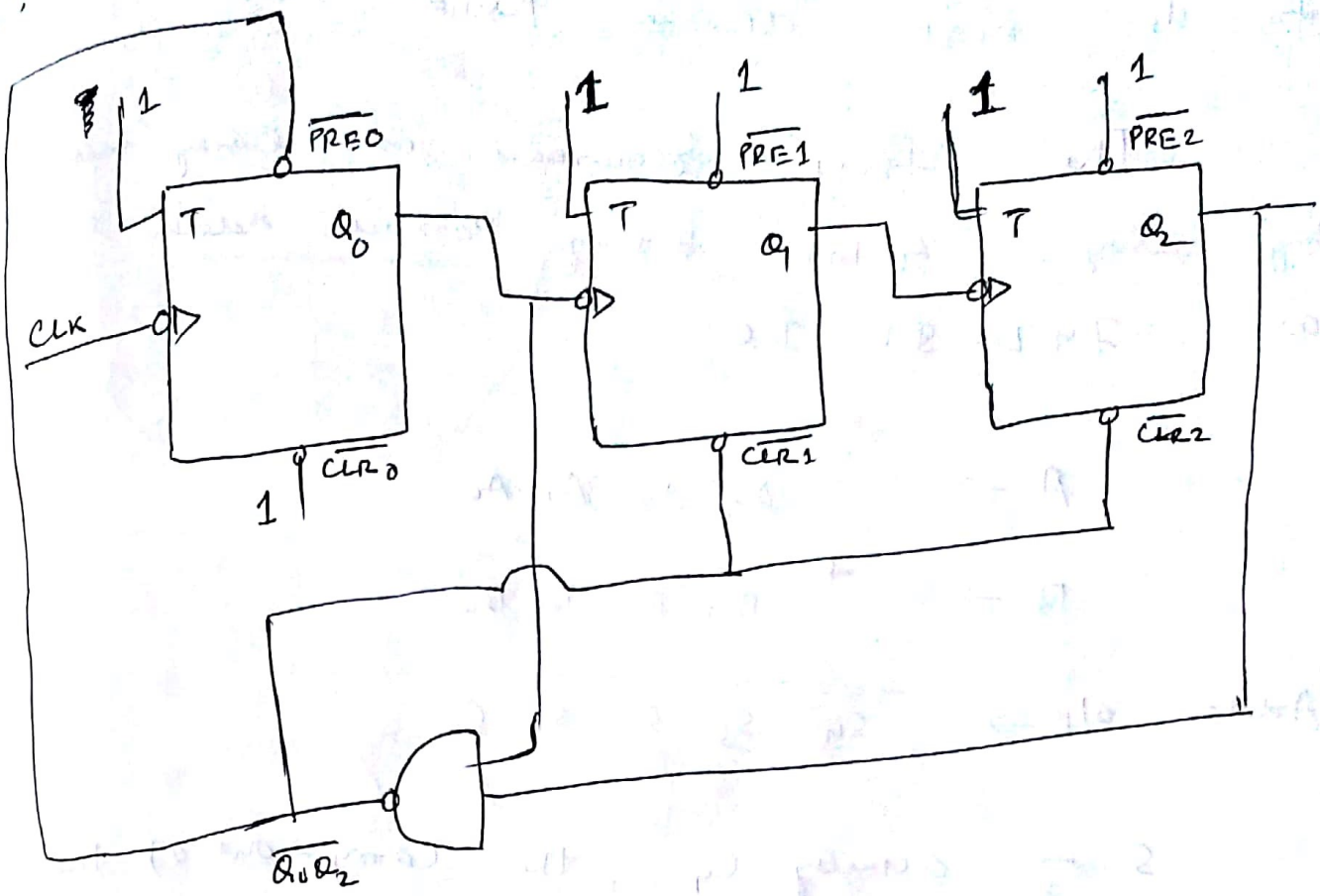
Steps for BCD addition

1) Add the 4-bit BCD code groups for each decimal digit position using ordinary binary addition.

2) For those positions where the sum is 9 or less, the sum is proper BCD form and no correction is needed.

3) When the sum of two digits is greater than 9, a correction of 0110 should be added to that sum, to produce proper BCD result

The CLK for this Counter.



2) BCD Adder :—

Steps for BCD addition

1) Add the 4-bit BCD code groups for each decimal digit position using ordinary binary addition.

2) For those positions where the sum is 9 or less, the sum is proper BCD form and no correction is needed.

3) When the sum of two digits is greater than 9, a correction of 0110 should be added to that sum, to produce proper BCD result.

This will produce a carry to be added to the next decimal position.

The first requirement is easily met by using 4-bit binary parallel adder such as 74LS83 IC.

$A \rightarrow \quad A_3\ A_2\ A_1\ A_0$

$B \rightarrow \quad +\ B_3\ B_2\ B_1\ B_0$

Adder o/p $\rightarrow \quad S_4\ S_3\ S_2\ S_1\ S_0$

$S_4 \rightarrow$ actually C_4, the Carry-out of the MSB bit

$\rightarrow \quad S_4\ S_3\ S_2\ S_1\ S_0$ range is 00000 to 10010

i.e 0 to 18. because maximum 2 numbers can be $\begin{array}{r} 9 \\ +\ 9 \\ \hline 18 \end{array}$.

$\rightarrow$ So BCD adder must include the logic needed to detect sum is greater than 01001, so that Correction can be added in.

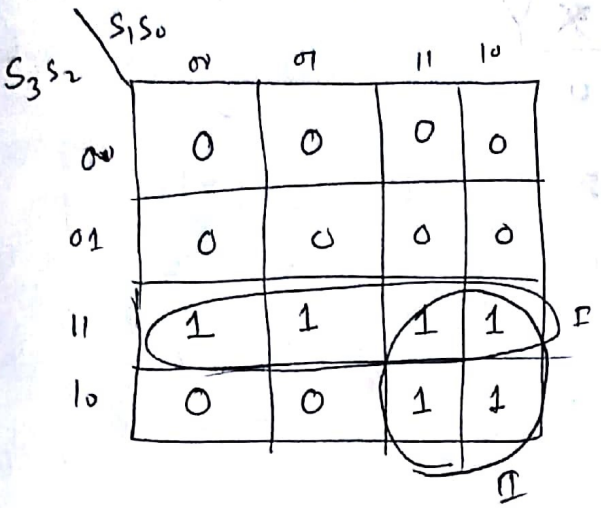
$\rightarrow$ Define a indicator (Y) which is 1 for Sum = 1010 to 10010. Sum $0 \rightarrow 9 \rightarrow 0$ ✗

$10 - 31 \rightarrow x$ Don't Care

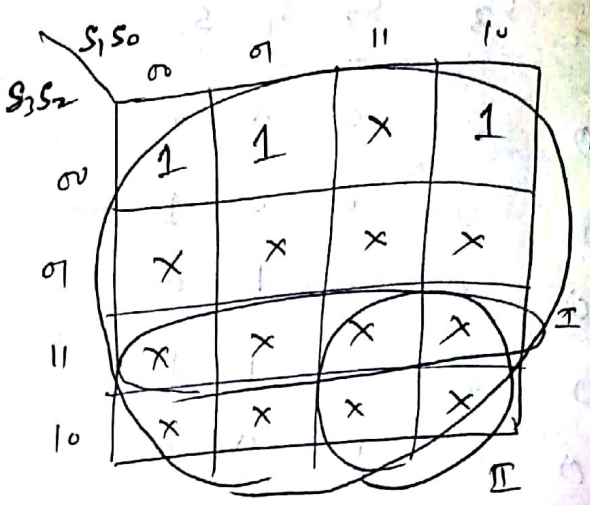
S_4	S_3	S_2	S_1	S_0	Y
0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	0
0	0	1	0	0	0
0	0	1	0	1	0
0	0	1	1	0	0
0	0	1	1	1	0
0	1	0	0	0	0
0	1	0	0	1	0
0	1	0	1	0	1
0	1	0	1	1	1
0	1	1	0	0	1
0	1	1	0	1	1
0	1	1	1	0	1
0	1	1	1	1	1
1	0	0	0	0	1
1	0	0	0	1	1
1	0	0	1	0	1
1	0	0	1	1	x
					x
					x
					x
					x
1	1	1	1	1	x

Don't Care
Max^m = 18
Sum =

Rest



$S_4 = 0$



$S_4 = 1$

$$I = S_3 S_2 \qquad II \rightarrow S_1 S_3$$

$$III \rightarrow S_4 . 1 = S_4$$

$$Y = S_4 + S_3 S_2 + S_1 S_3$$

$$\boxed{Y = S_4 + S_3 (S_1 + S_2)}$$

The ckt consists of 3 basic parts, The two BCD code group $A_3 A_2 A_1 A_0$ & $B_3 B_2 B_1 B_0$ are added together in the upper 4-bit adder, to produce the sum $S_4 S_3 S_2 S_1 S_0$.

The logic gates shown implement the expression for Y. The lower 4bit adder will add the correction 0110 to the sum bits, only when $Y = 1$, producing the final

BCD Sum O/P represented by $\Sigma_3\ \Sigma_2\ \Sigma_1\ \Sigma_0$. This Y is also the Carry out that is produced when Sum is greater than 01001.

Of Course when Y = 0, there is no Carry and no addition of 0110. In such Cases $\Sigma_3\ \Sigma_2\ \Sigma_1\ \Sigma_0 = S_3\ S_2\ S_1\ S_0$.

Ckt

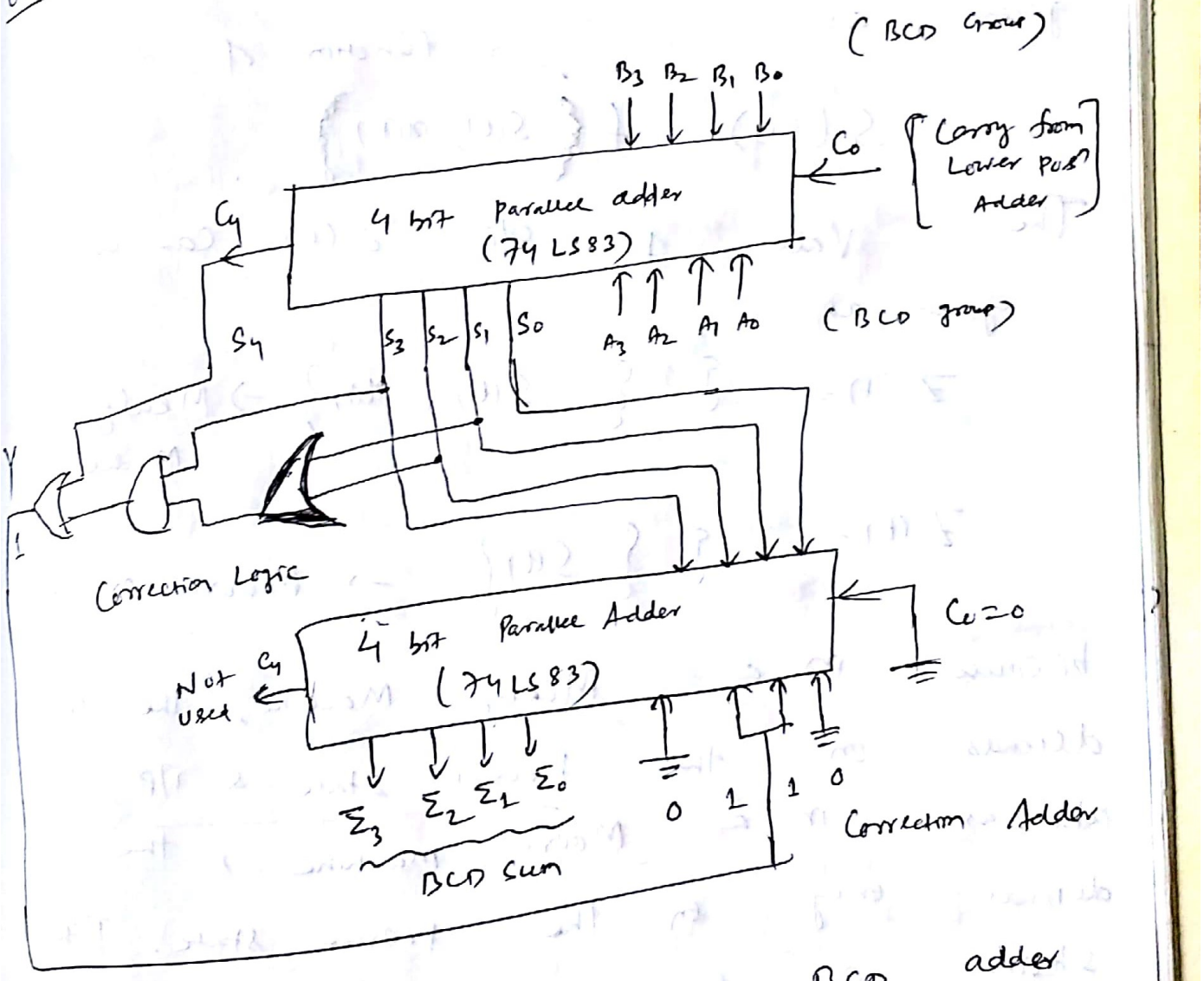


Fig:- Logic diagram of a BCD adder using two 4-bit adder and a Correction-detector circuit.

Mathematical Representation of Synchronous Sequential Machine

→ We know that the next state of a sequential machine depends upon the present state & the present i/p. The relation between the present state $S(t)$, present i/p $x(t)$, and next state $S(t+1)$ can be given as

$$S(t+1) = f\{S(t), x(t)\}$$

→ function of.

The value of o/p $Z(t)$ can be given as

$$Z(t) = g\{S(t), x(t)\} \rightarrow \text{Mealy Model}$$

$$Z(t) = g\{S(t)\} \rightarrow \text{Moore model.}$$

because in a Mealy Machine, the o/p depends on the present state & i/p, whereas in a Moore machine, the o/p depends only on the present state. Table shows a comparison between the Moore Machine & ~~Mealy~~ Mealy machine.

Table 1: Comparison bet Moore & Mealy Machine

Moore Machine	Mealy Machine
1. The o/p is a f^n of present state only. $Z(t) = g\{S(t)\}$	1. The o/p is f^n of present state as well as present i/p. $Z(t) = g\{S(t), x(t)\}$
2. I/p change don't affect the o/p.	2. I/p change may affect the o/p of the ckt.
3. It requires more number of states for implementing same function	3. It requires less number of states for implementing same function.

Mealy Model:-

When the o/p of the sequential ckt depends on both the present state of the F/F and on the i/ps, the sequential circuit is referred to as Mealy circuit or Mealy Machine

Fig 1. shows the logic diagram of a Mealy model. Notice that the o/p depends on the present state as well as i/ps.

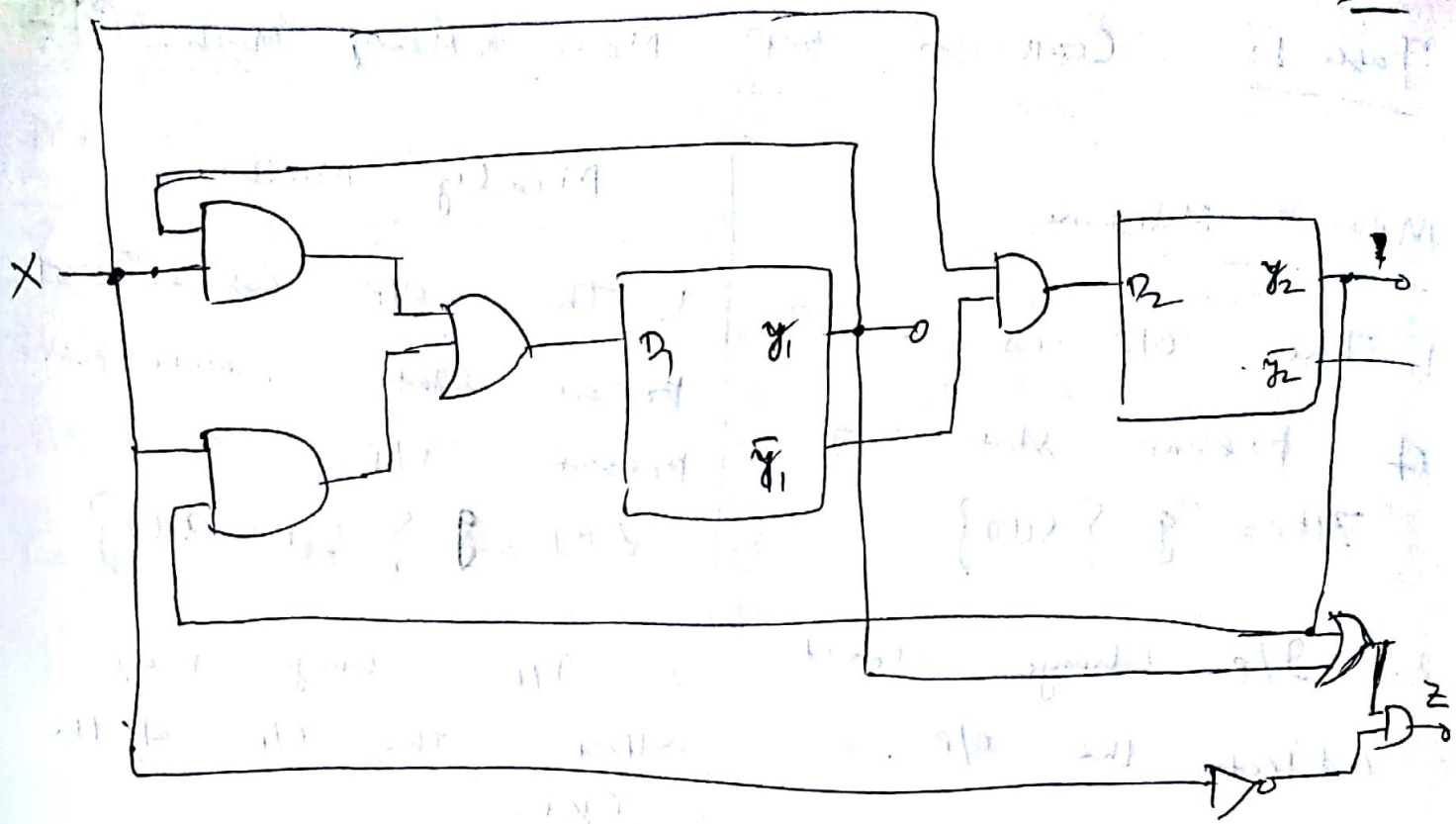


Fig 12 - Logic diagram of Mealy model.

The Mealy model shown in figure consists of two D-FFs, an input (X) and o/p (Z).

Since the D i/p of a FF determine the value of next state, the state equations for the model can be written as

$$y_1(t+1) = y_1(t) \cdot x(t) + y_2(t) \cdot x(t) \quad \text{---(1)}$$

$$y_2(t+1) = \bar{y}_1(t) \cdot x(t) \quad \text{---(2)}$$

$$Z(t) = \{ y_1(t) + y_2(t) \} \cdot \bar{x}(t) \quad \text{---(3)}$$

where $y(t+1)$ is the next state of FF one clock edge later, $x(t)$ is the present

I/P & Z(t) is the present O/P.

O/P $y_1(t+1)$ & $y_2(t+1)$ are represented by $Y_1(t)$ & $Y_2(t)$, in more compact form, the eqns are

$$Y_1 = y_1 x + y_2 x \quad \text{——(4)}$$

$$Y_2 = \bar{y}_1 x \quad \text{——(5)}$$

$$\boxed{Z = (y_1 + y_2)\bar{x}} \quad \text{——(6)}$$

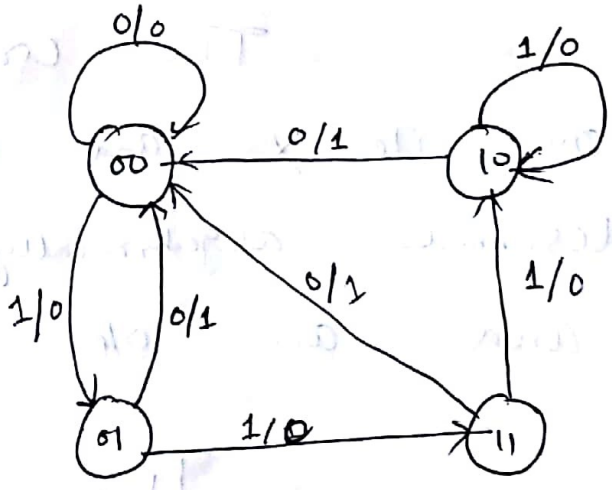
Note:- O/P depends on present state and i/p.

The State table of the Melay model based on the above state equations & O/P eqn is shown in fig 2(a) & State diagram is shown in fig 2(b)

P.S	N.S				O/P	
	x=0		x=1		x=0	x=1
y_1 y_2	Y_1	Y_2	Y_1	Y_2	Z	Z
0 0	0	0	0	1	0	0
0 1	0	0	1	1	1	0
1 0	0	0	1	0	1	0
1 1	0	0	1	0	1	0

(using eqn 4, 5, 6)

(a) State Table



(b) State diagram

Fig 2:- Melley Model.

Note:- Till now all the designs of sequence detector we have done, using Mealy Model.

I/P & Z(t) is the present O/P.

If $y_1(t+1)$ & $y_2(t+1)$ are represented by $Y_1(t)$ & $Y_2(t)$, in more compact form, the eqns are

$$Y_1 = y_1 x + y_2 x \quad \text{——(4)}$$

$$Y_2 = \bar{y}_1 x \quad \text{——(5)}$$

$$Z = (y_1 + y_2)\bar{x} \quad \text{——(6)}$$

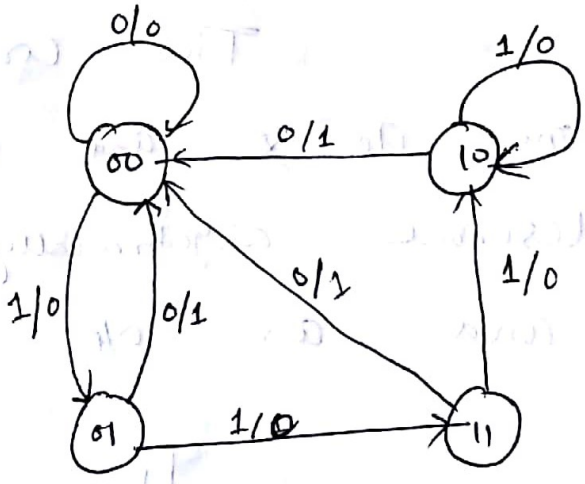
Note:- O/P depends on present state and I/P.

The state table of the Melay model based on the above state equations & O/P eqn is shown in fig 2(a) & state diagram is shown in fig 2(b)

P.S	N.S				O/P	
	x=0		x=1		x=0	x=1
y_1 y_2	Y_1	Y_2	Y_1	Y_2	Z	Z
0 0	0	0	0	1	0	0
0 1	0	0	1	1	1	0
1 0	0	0	1	0	1	0
1 1	0	0	1	0	1	0

(using eqn 4, 5, 6)

(a) State Table



(b) State diagram

Prg 2:- Melay Model.

Note:- Till now all the designs of sequence detector we have done, using Mealy Model.

Moore Model :-

When the o/p the sequential circuit depends only on the present state of the F/F, the sequential circuit is referred to as Moore circuit or Moore Machine.

Fig 3. shows the logic diagram of a Moore ckt.

Notice that the o/p depends only on present state. It does not depend on the i/p at all. The i/p is used only to determine the i/p's of the F/Fs. It is not used to determine the o/p.

The circuit shown has 2 T F/Fs, one i/p x and one o/p z. It can be described algebraically by 2 i/p eqns and an o/p eqn.

$$T_1 = y_2 x$$

$$T_2 = x$$

$$z = y_1 y_2$$

Note:- o/p depends only on present states.

The characteristic eqn of a T F/F is

$$Q(t+1) = T\bar{Q} + \bar{T}Q \quad \text{—(7)}$$

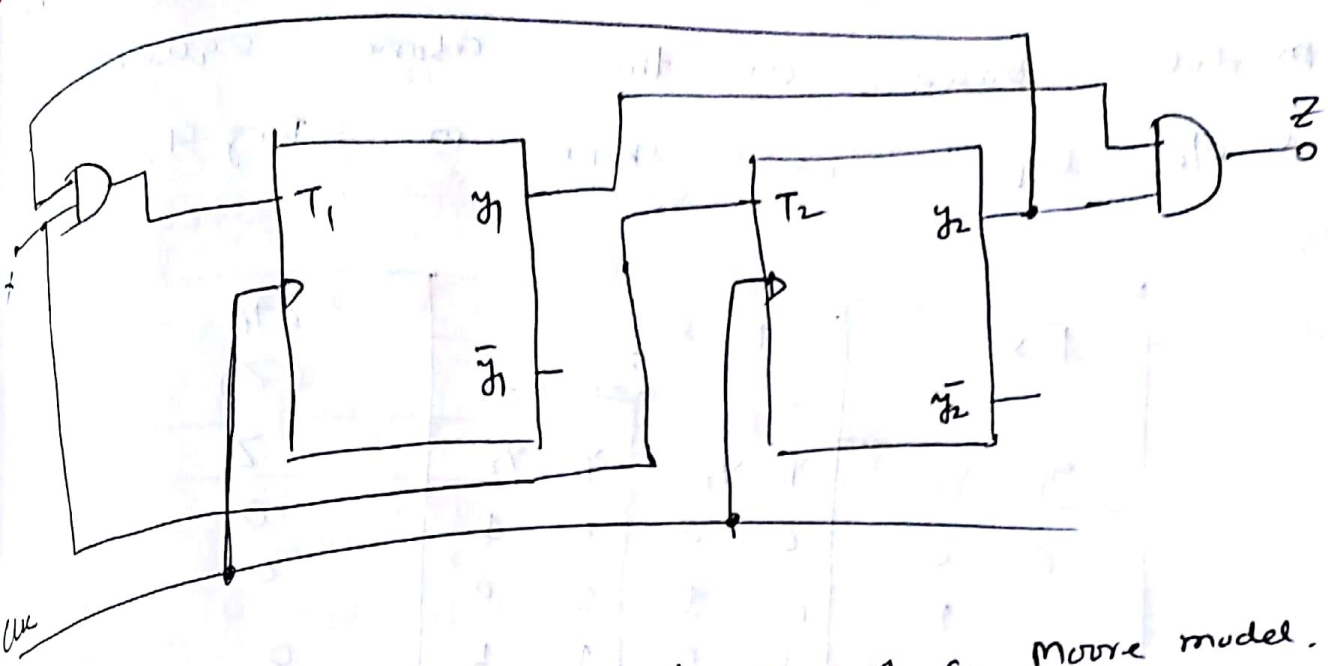


Fig 3:- Logic diagram of a Moore model.

The value for the next state can be derived from the state equations by substituting T_1 and T_2 in the characteristic equation (Eq^n 7)

$$y_1(t+1) = T_1 \bar{Q}_1 + \bar{T}_1 Q_1$$

$$= y_2 x \cdot \bar{y}_1 + \overline{y_2 x}\, y_1$$

$$= y_2 x \bar{y}_1 + (\bar{y}_2 + \bar{x}) y_1$$

$$y_1(t+1) = y_2 x \bar{y}_1 + \bar{y}_2 \cdot y_1 + \bar{x} y_1$$

Rearranging ↓

$$\boxed{Y_1 = y_1 \bar{y}_2 + y_1 \bar{x} + \bar{y}_1 y_2 x} \quad \text{——— (8)}$$

$\therefore T_1 = y_2 x$, $Q_1 = y_1$

$$y_2(t+1) = T_2 \bar{Q}_2 + \bar{T}_2 Q_2$$

$$= x \bar{y}_2 + \bar{x} y_2$$

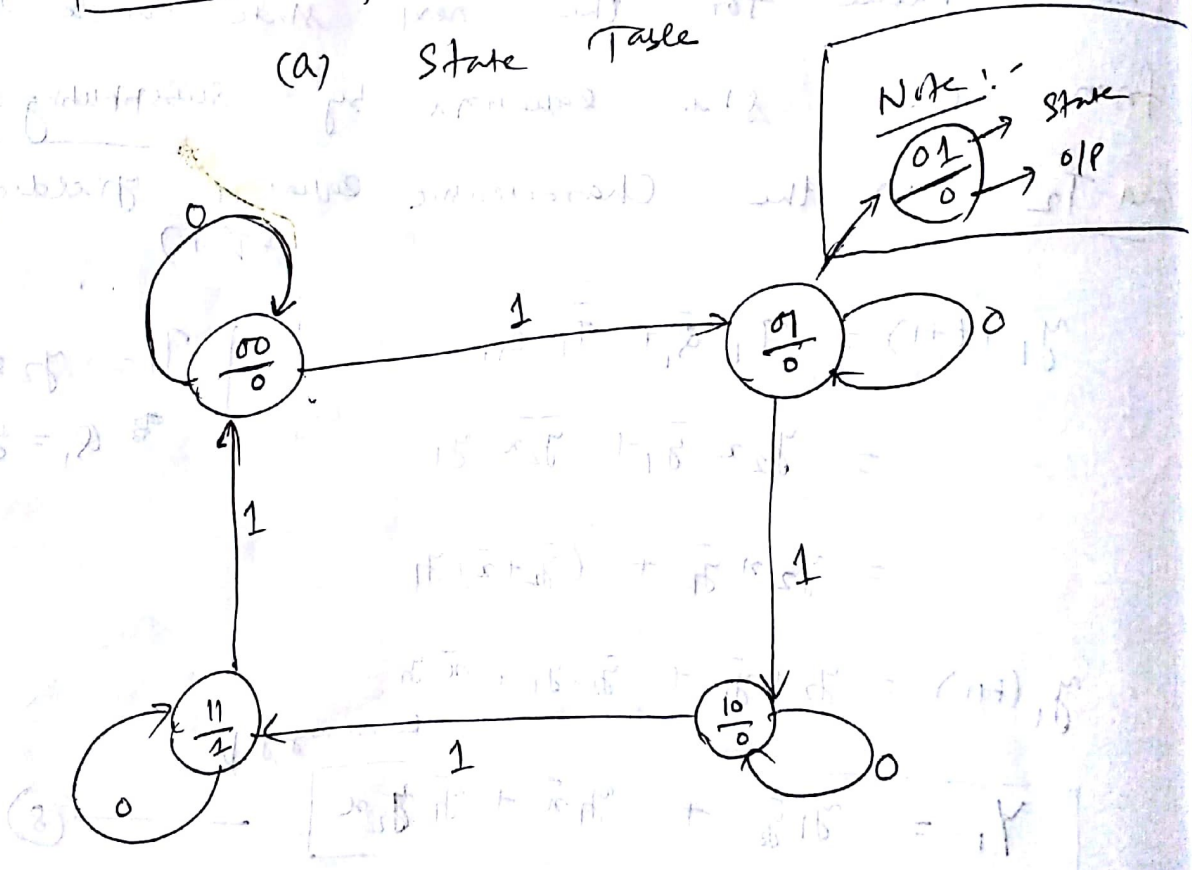
$$\therefore \boxed{Y_2 = x \bar{y}_2 + \bar{x} y_2 = x \oplus y_2} \quad \text{——— (9)}$$

$T_2 = x$, $Q_2 = y_2$

The State table of the Moore model based on the above equations & o/p eqn is shown in Fig 4.

P.S		N.S				o/p (Z)
		X=0		X=1		
y_1	y_2	Y_1	Y_2	Y_1	Y_2	Z
0	0	0	0	0	1	0
0	1	0	1	1	0	0
1	0	1	0	1	1	0
1	1	1	1	0	0	1

(a) State Table



(b) State Diagram

Fig 4:- Moore Model

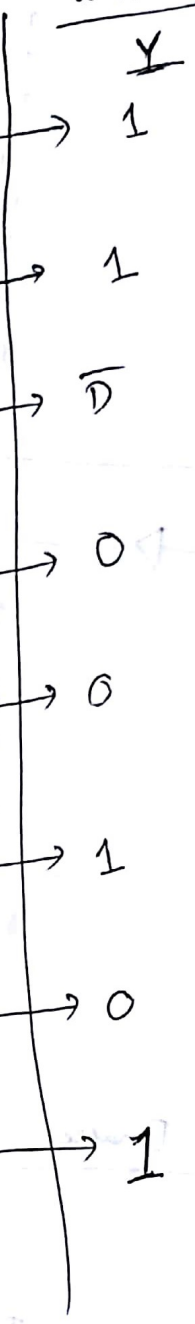
Note:- In Moore model, o/p (Z) has 2 columns for X=0 & X=1, In Moore Z has one column. Since Z is independent of i/p. (X).

Q) Implement SOP (0, 1, 2, 3, 4, 10, 11, 14, 15) Using 3 select line MUX. With Enable I/P.

Ans:

A	B	C	D	Y
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

Without Enable Pin



Using Enabler

Use A as Enabler.

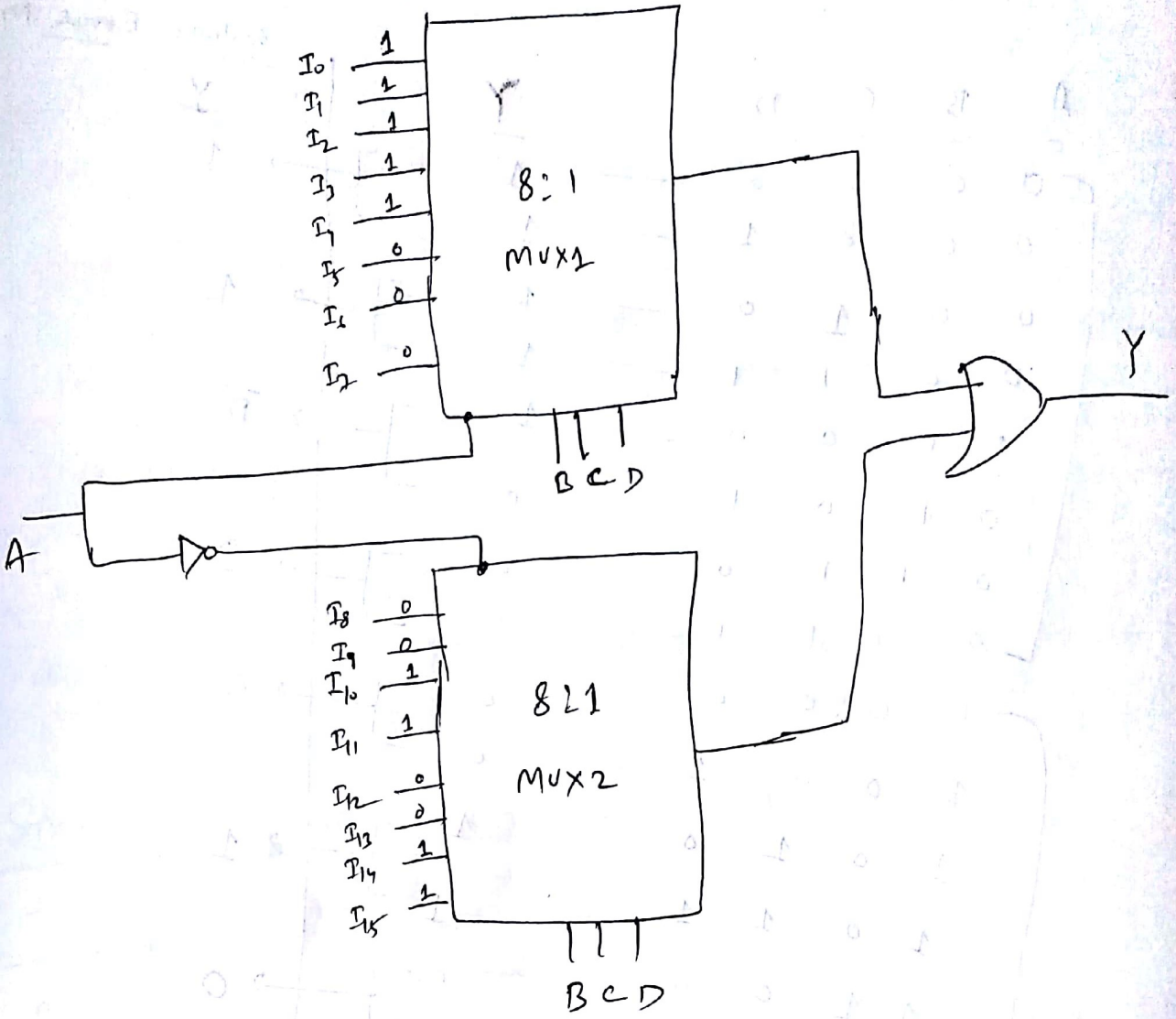
0 MUX 1 activated. when A is

1 MUX 2 is activated. when A is

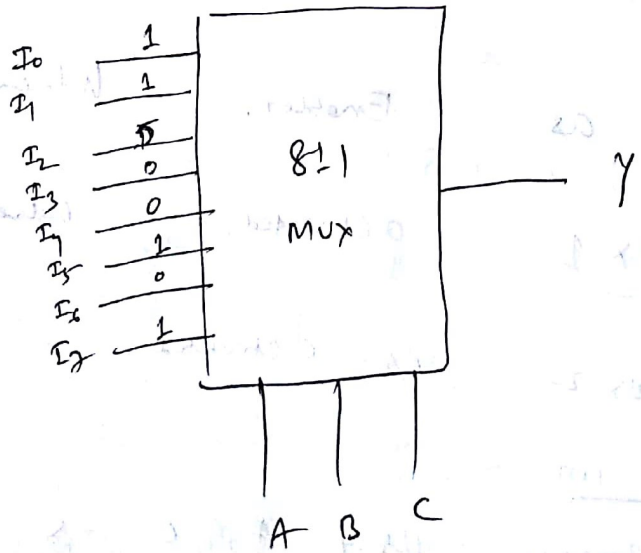
Without Enable Pin

Make the grouping using A, B, C & find the relation bet D & Y

With Enable Pin



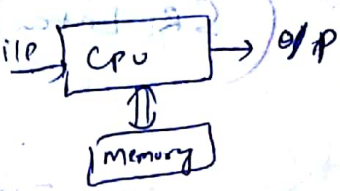
Without Enable



Memory Classification:-

Memory is an essential component of micro computer system. It stores binary instructions & data for the microprocessor. There are various types of memory which can be classified in 2 groups such as

1. Primary Memory.
2. Secondary Memory.



(writing a C program)

Primary Memory:-

The chief characteristic is that CPU directly accesses their location in the main memory.

These are used for programme and data storage, during computer operations. Total capacity of a primary memory varies from a few KB for small computing system to several MB/GB for large systems.

→ If a memory has n address lines then the memory capacity is 2^n bytes.

→ Basically the primary memory are divided into 2 groups.

1. Read/write memory (RAM)
2. Read only memory (ROM)

Read/Write Memory (RAM)

The microprocessor can write into or read from this memory. It is popularly known as RAM (Random Access Memory).

It is used primarily for information that is likely to be altered, such as writing programs or receiving data.

→ This memory is volatile, meaning that when power is turned off, all the contents are destroyed.

→ Two types of R/W memories – Static and dynamic (SRAM / DRAM)

SRAM (Static RAM)

→ This memory is made up of flip-flop and it stores the bit as a voltage.

→ Each memory cell requires six transistors.

→ Therefore the memory chip has low density but high speed.

→ This memory is more expensive & consumes more power than dynamic memory.

→ In high speed processor, SRAM known (as Cache memory), included on the processor chip.

→ In addition, high speed cache memory is also included external to the processor to improve the performance of a system.

DRAM (Dynamic RAM)

→ This memory is made up of MOS transistor and it stores the bit as a charge.

→ The advantage of dynamic memory are that it has high density and low power consumption

and is cheaper than static memory.

→ The disadvantage is that the charge (bit information) leaks, therefore stored information needs to be read and written again every few milliseconds.

→ This is called refreshing the memory, and it requires extra circuitry, adding to the cost of the system.

V. Imp

Static RAM	Dynamic RAM
1) It uses registers & latches / Flip flops for data storage	1) It uses capacitor for data storage.
2) The stored information are retained in it as long as power supply is ON.	2) It losses its content in a very short time even though the power supply is ON.
3) It doesn't require refreshing ckt.	3) It requires refreshing ckt.
4) It is used for small amount of data storage	4) It is used for large amount of data storage
5) It consumes more power and are more expensive	5) It consumes less power & less expensive,
6) Faster	6) Slower
7) It stores the bit as a voltage.	7) It stores the bit as a charge

Both SRAM & DRAM are volatile.

ROM (Read-only-Memory)

→ The ROM is a nonvolatile memory.

→ It retains information even if the power is turned off.

→ The ROM, like the RAM is random-access memory, but the term RAM traditionally means a random-access read/write memory.

→ This memory is used for programs and data that needs not to be altered.

→ As the name, suggests, the information can be read only, which means once a bit pattern is stored, it is permanent.

ROM are classified into 5 types, such as

(a) Masked ROM
(b) PROM
(c) EPROM
(d) EEPROM
(e) Flash Memory

a) Masked ROM :-

In this ROM, a bit pattern is permanently recorded by the masking and metallization process. Memory manufacturers are generally required to do this process. It is expensive and specialized process. But economical for large production quantities.

PROM (Programmable Read only Memory) :-

→ This memory has nichrome and polysilicon wires arranged in a matrix

→ These wires can be functionally viewed as diodes or fuses. This memory can be programmed by the user with a special PROM programmer that selectively burns the fuses according to the bit patterns to be stored. This process is known as burning the PROM and the information is stored permanently.

(like Burning a Read only CD)

EPROM (Erasable Programmable Read only memory)

The information can be erased by exposing the chip to ultra violet light and the chip can be reprogrammed. Because the chip can be used many times, this memory is ideally suited for product development, experimental projects and college laboratories.

EE - PROM (Electrically Erasable PROM)

This memory is functionally similar to EPROM, except that information can be altered by using electrical signals at the register level rather than erasing all the information. This has an advantage in field and remote control applications.

Flash Memory :-

This is a variation of EE-PROM that is becoming popular. The major difference between the flash memory and EE PROM is the erase procedure. The EE-PROM can be erased at a register level, but the flash memory must be erased either in its entirely or at the sector (block) level. Faster than EEPROM.

Secondary Memory :-

If it is necessary to store more data than the maximum capacity of primary memory then secondary memory is employed.

→ The CPU can't directly access a secondary memory device. The secondary memories can be accessed only through I/O i.e input/output ports or in a serial format using proper software and hardware.

→ Magnetic tapes, floppy disks, CD-ROM, Hard-drive, Pen drive etc are common examples of secondary memory.

— u —