Che-8 Sequential Logic circuits - II

SPP

(8.1) Registers & shift Registers.

Registers

⇒ Flip Flop can store only one bit of data '0' or '1', it is referred as single bit Registers.

⇒ When no. of more bits of data are to be stored, No. of FFs are required.

⇒ Register is a set of FF used to store binary data.

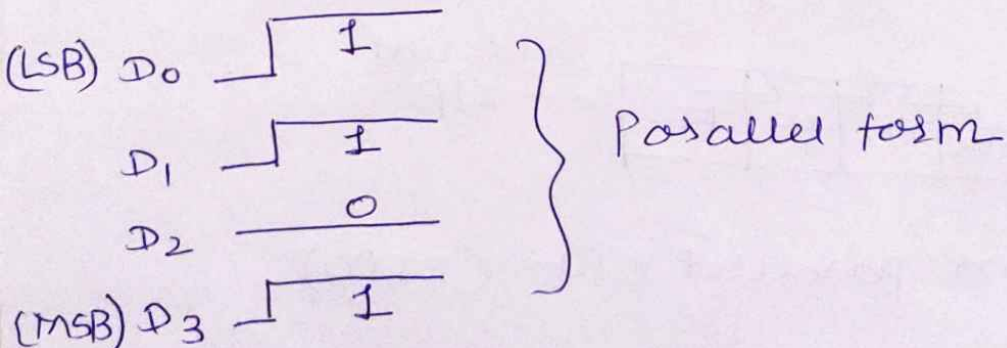
⇒ An n-bit Register has a group of 'n' Flip Flop & n-bit storage capacity.

⇒ Data is in serial form & parallel form.

⇒ Data in serial form entered one bit at a time.

D → 1 0 1 1 } serial form.

⇒ Data in parallel form entered no. of bit at a time. suppose 4-bit data so 4-bit data entered at a time



Shift Register

⇒ Shift Registers are a type of logic circuit used for the storage & transfer of digital data.

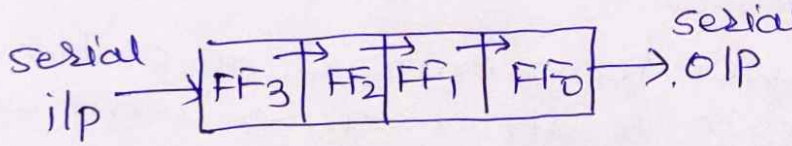
⇒ Register capable of shifting binary data either to the right or to the left is called shift Register.

⇒ Shift Register consists of chain of FF connected in cascade, with output of one FF connected to i/p of next FF.

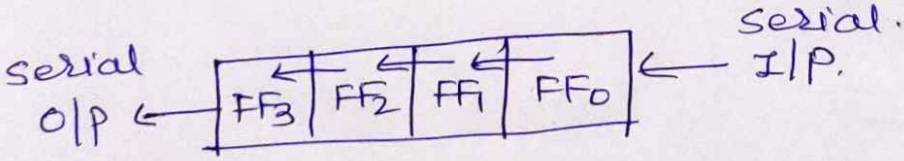
⇒ All FF receives a common clock pulse which causes the shift from one stage to the next.

Types of shift Register:

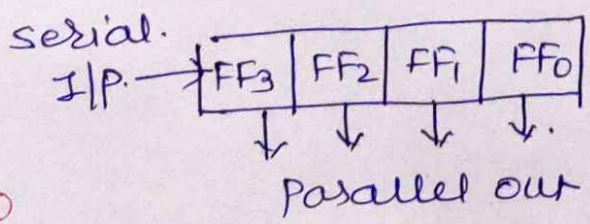
(1) serial-In serial out (SISO) - shift Right



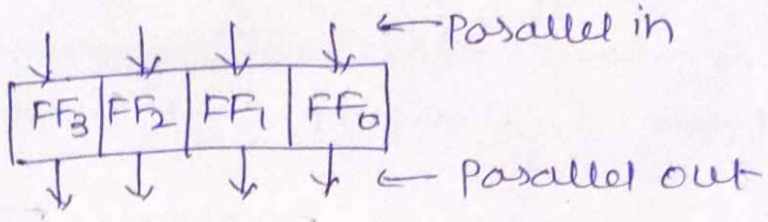
(2) serial-In serial out (SISO) - shift Left



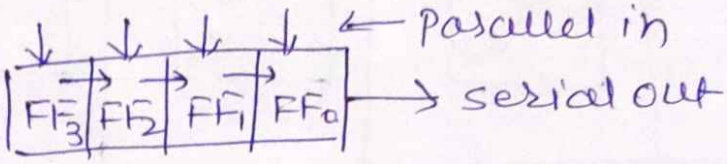
(3) serial in parallel out (SIPO)



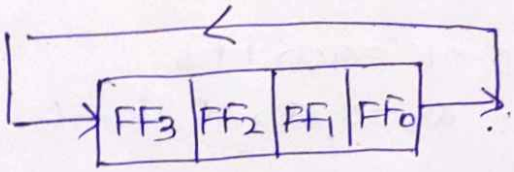
(4) Parallel in parallel out (PIPO)



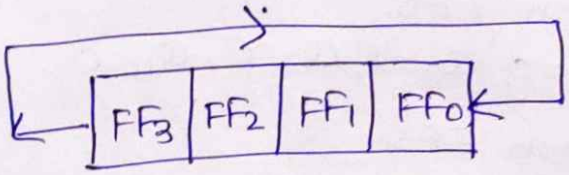
(5) Parallel in serial out (PISO)



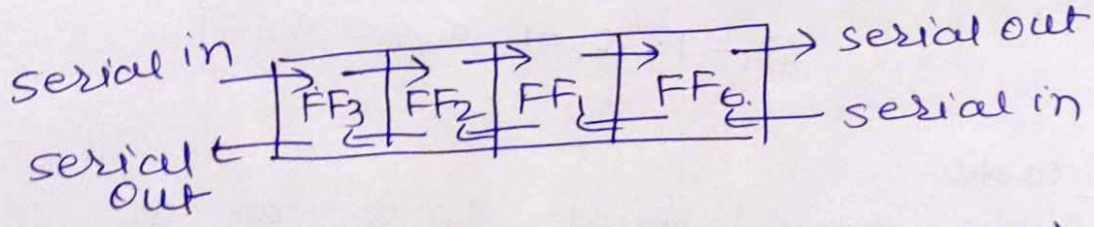
(6) Rotate Right shift Register (RRS)



(7) Rotate Left shift Register (RLS)



(8) Bidirectional shift Register (BDS)



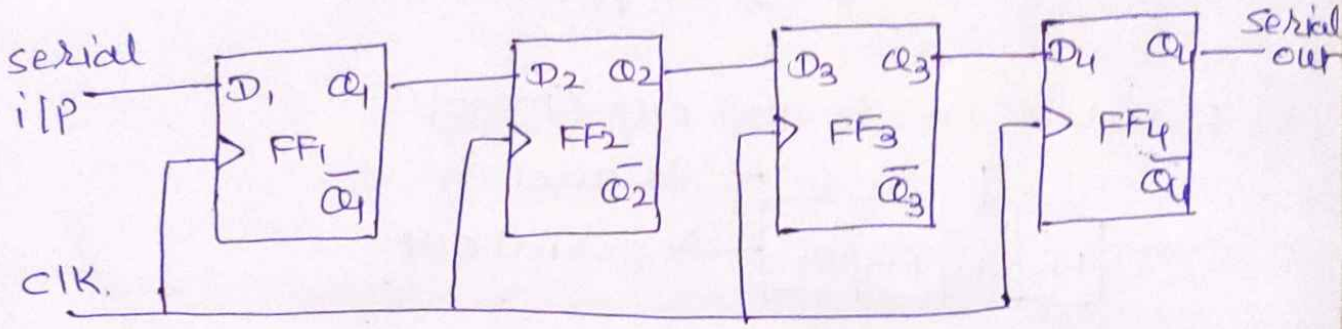
(9) Universal shift Register (US)

perform four operation → No change
→ Parallel load.
→ shift Left
→ shift Right

(8.2) Design of shift Registers (D FFs)

(8.2.1) Serial In serial out (SISO) shift Register

Ex:1 Design 4-bit SISO using D-FF. (shift Right)



Operation: Serial Data i/p → 1011 (4-bit data)

initially No clock, o/p of each FFs

$Q_1=0, Q_2=0, Q_3=0, Q_4=0.$

1st clock, o/p of each FFs

$Q_1=1, Q_2=0, Q_3=0, Q_4=0.$

2nd clock, o/p of each FFs.

$Q_1=0, Q_2=1, Q_3=0, Q_4=0.$

3rd clock, o/p of each FFs.

$Q_1=1, Q_2=0, Q_3=1, Q_4=0.$

4th clock, o/p of each FFs

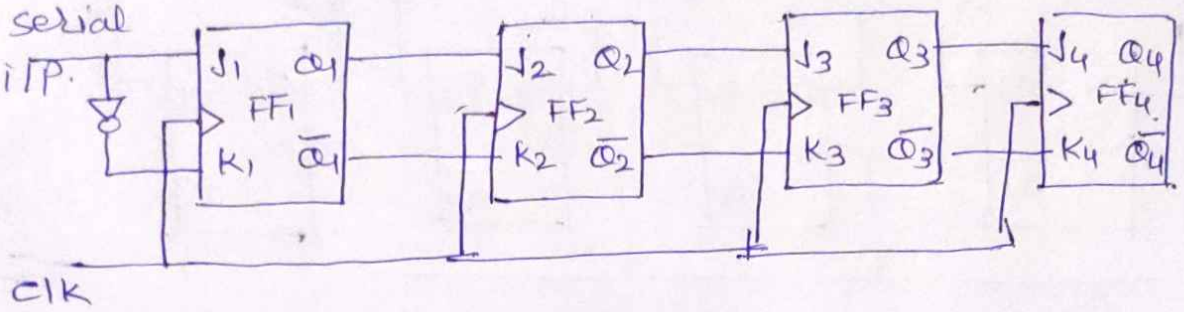
$Q_1=1, Q_2=1, Q_3=0, Q_4=1.$

Operation Table

After clock pulse	serial i/p	Q_1	Q_2	Q_3	Q_4
0	1	0	0	0	0
1	0	1	0	0	0
2	1	0	1	0	0
3	1	1	0	1	0
4	—	1	1	0	1

Ex:2 Design 4-bit SISO using JK FF. (Shift Right) SDP

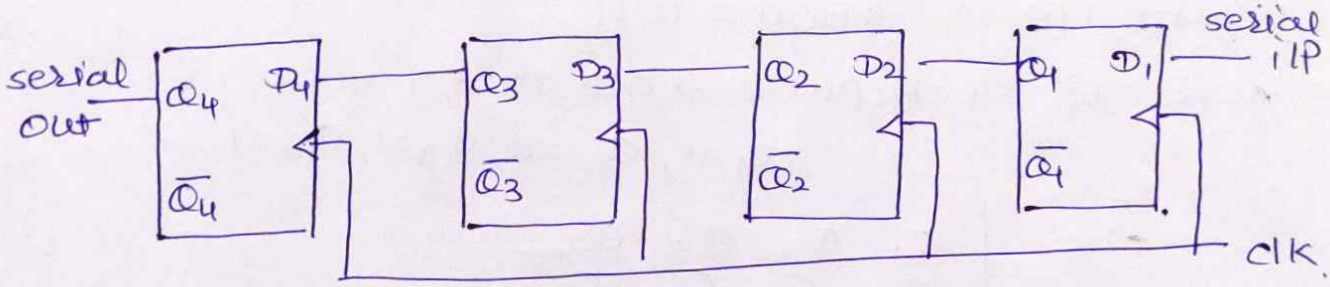
Ans: Required FF is D FF so convert JK to D FF.



Operation & operation Table

⇒ same as Example-1.

Ex:3 Design 4-bit SISO using D FF (Shift Left)

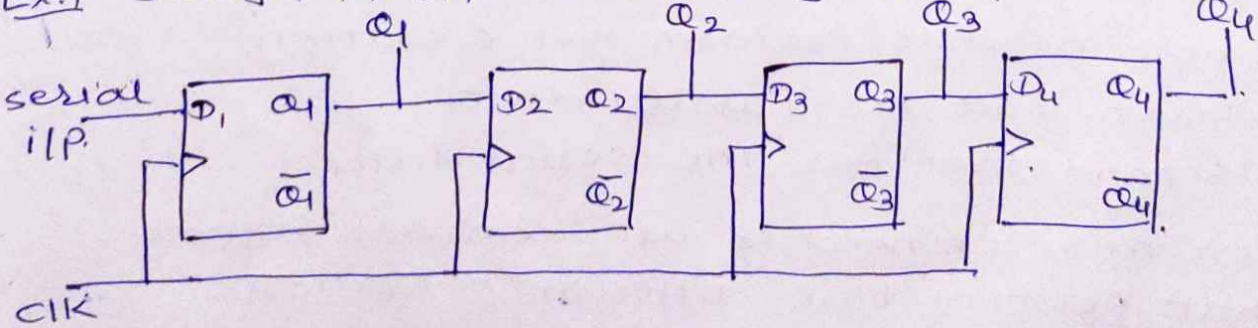


Operation & operation Table.

⇒ Do yourself by the help of Example-1.

(8.2.2) serial in parallel out (SIPO) shift Register

Ex:1 Design 4-bit SIPO using D-FF.

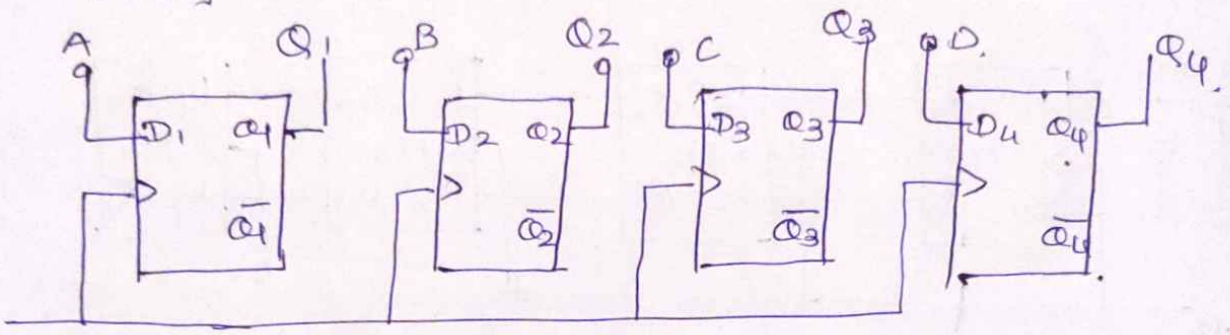


Operation & operation Table.

→ Do your self.

(8.2.3) Pasallel-In Pasallel out (PIPO) shift Reg. SDP

Ex:1 Design pasallel in pasallel out 4-bit shift Reg. using D-FF.



Operation & Operation Table.

initialy No clock. → O/P of each FF is

$Q_1=0, Q_2=0, Q_3=0, Q_4=0$.

Data i/P. ⇒ ABCD = 1011

After 1st clock pulse → O/P of each FF is

$Q_1=1, Q_2=0, Q_3=1, Q_4=1$.

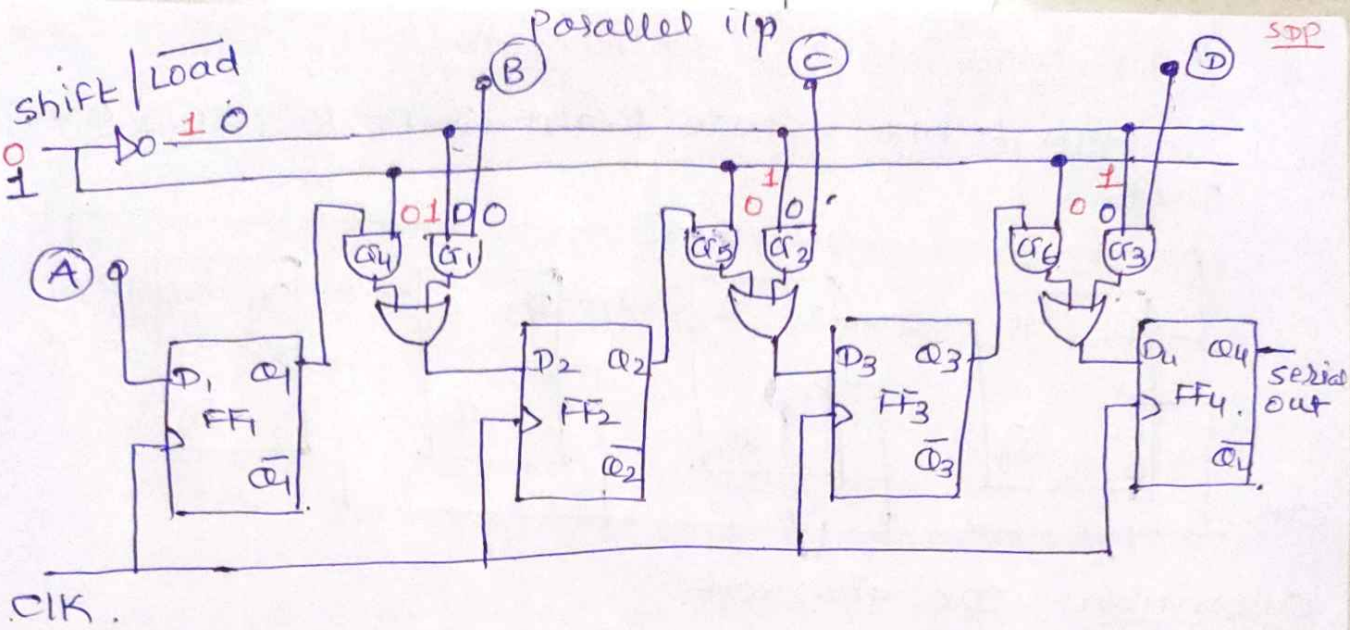
Clk	Q_1	Q_2	Q_3	Q_4
0	0	0	0	0
1	1	0	1	1.

(8.2.4) Pasallel in sesial out (PISO) shift Reg

Ex:1 Design pasallel in sesial out shift Reg. using D-FF.

Ans: here need to pesfosm two opesation.
Fisst load the pasallel data.
second tsansfes the sesial data.

⇒ Fos this we need to use contsol signal that pesfosm two diffesent opesation when i/p is 0 os 1.



Operation

⇒ 4 bit parallel data i/p ⇒ A, B, C, D

⇒ First need to Load the parallel data, SO Make shift / $\overline{Load}$ pin = 0 (Red Mask)

gate G_4, G_5, G_6 are Inactive.

gate G_1, G_2, G_3 are active, so load Parallel data from A, B, C, D.

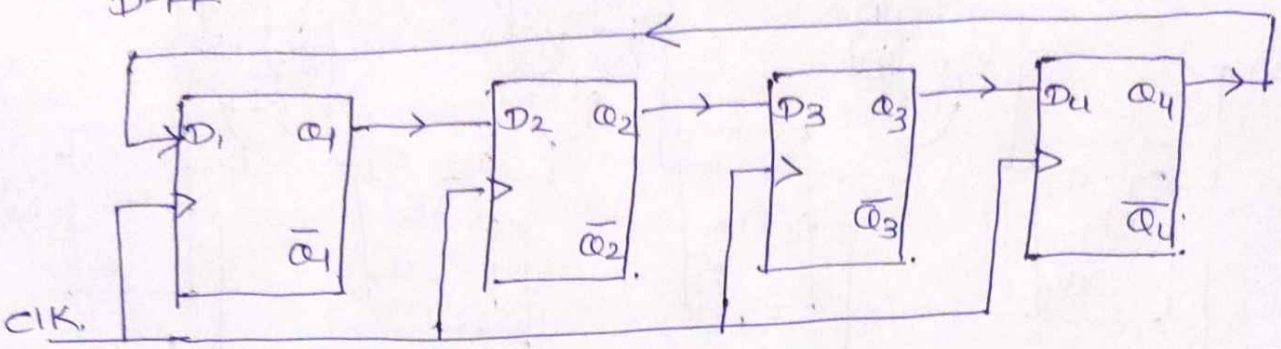
⇒ Make shift / $\overline{Load}$ pin = 1 (Blue mask).

gate G_1, G_2, G_3 are Inactive.

gate G_4, G_5, G_6 are active, so data x'fer from Left to Right & out at Q_4 terminal.

(8.2.5) Rotate Right shift Register.

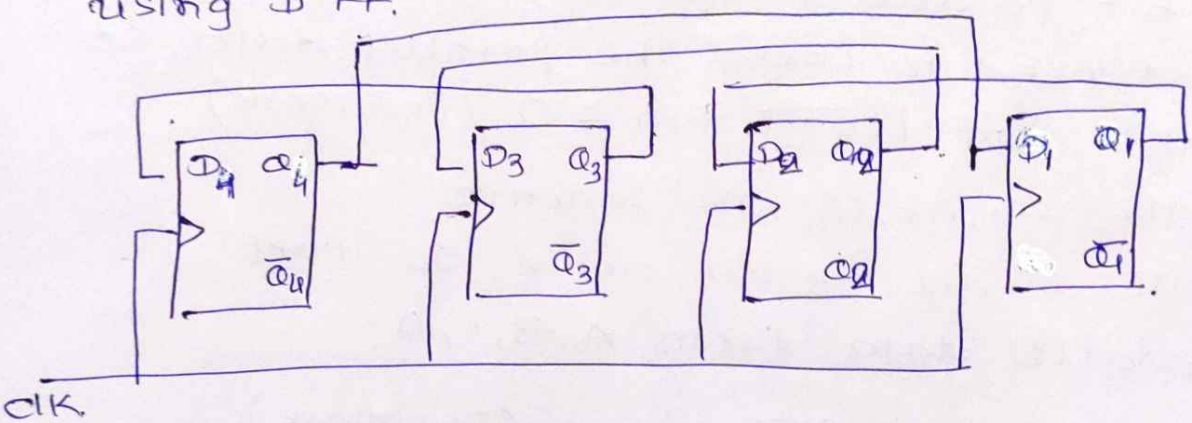
Ex:1 Design 4-bit Rotate Right shift Register using D-FF



Operation: Do yourself.

(8.2.6) Rotate Left shift Register

Ex:2 Design 4-bit Rotate Left shift Register using D-FF.



operation: Do yourself

(8.2.7) Bidirectional shift Register.

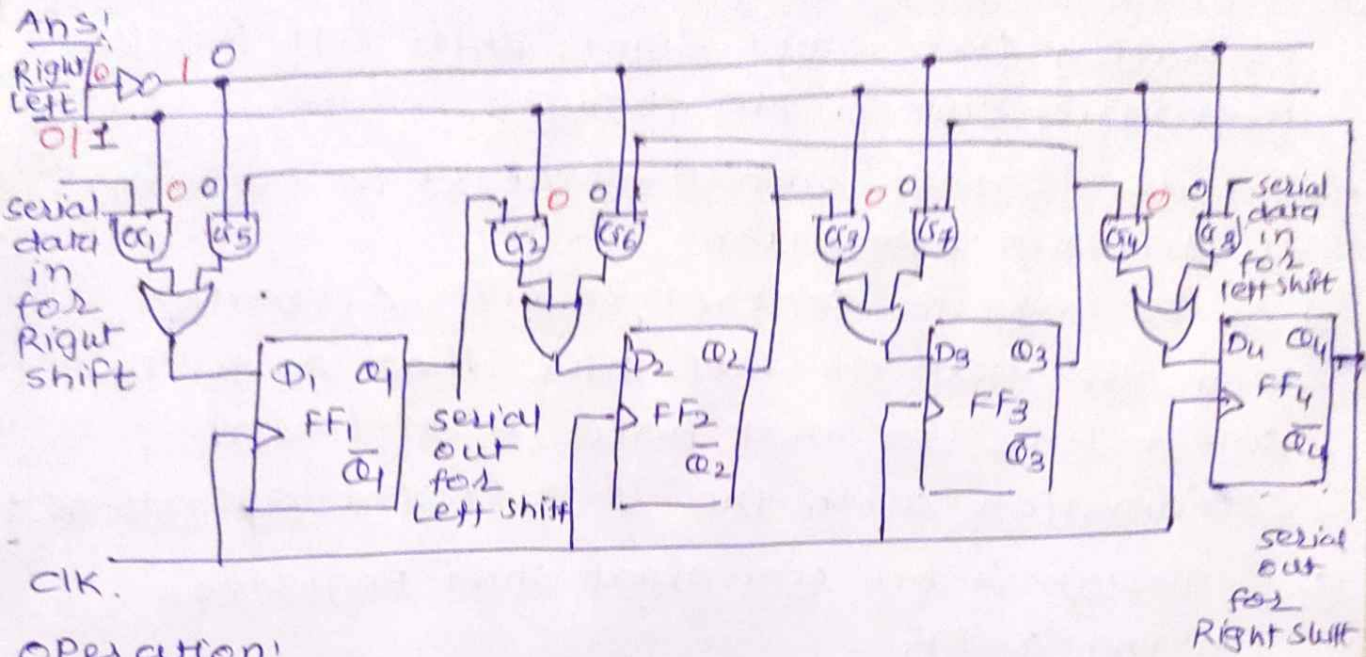
⇒ It perform two operation → shift Right
→ shift Left.

⇒ control signal Right/$\overline{\text{Left}}$ used to control the shift Right & shift Left operation.

⇒ Right/$\overline{\text{Left}}$ = 0 (Low) ⇒ Data shift left
= 1 (High) ⇒ Data shift Right.

Ex:1 Design 4-bit Bidirectional shift Register using D-FF.

Ans:



Operation:

⇒ Right/$\overline{\text{Left}}$ = 0 (Low).

gate G_1, G_2, G_3 & G_4 are Inactive ⇒ No Right shift

gate G_5, G_6, G_7, G_8 are active so data take from G_8 gate & x'fer Left ~~to~~ FF_4 to FF_1 & data out at Q_1.

⇒ Right/$\overline{\text{Left}}$ = 1 (High)

gate G_5, G_6, G_7, G_8 are Inactive ⇒ No Left shift

gate G_1, G_2, G_3 & G_4 are active. So data take from G_1 gate & x'fer Right FF_1 to FF_4 & data out at Q_4.

(8.2.8) Universal shift Register

⇒ universal shift Register perform 4 different operation like shift Right, shift Left, Parallel in parallel out & No change.

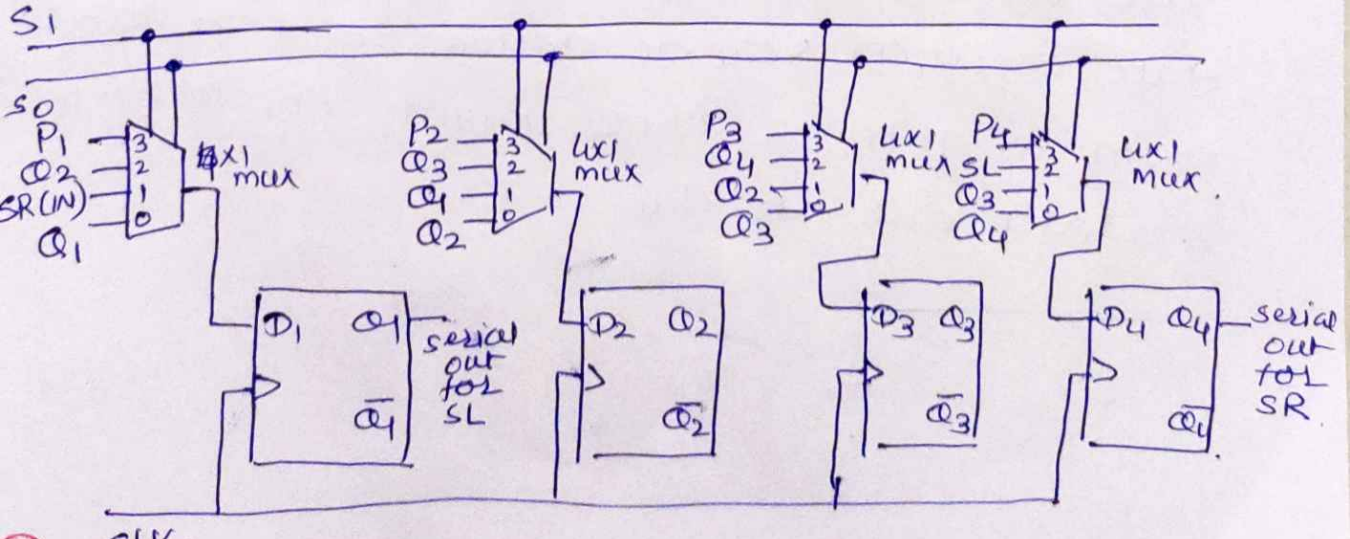
⇒ Mode control signal Required to perform 4 different operation.

⇒ It is easy to control 4 diff. operation with the help of 4x1 mux that have Two select line so that have 4 differen combination with the 4-different operations.

Ex:) Design 4-bit universal shift Register using D-FF.

Ans: here we will use 4x1 mux for the four different operations.

Mode control S_1	S_0	operation
0	0	No change
0	1	shift Right
1	0	shift Left
1	1	Parallel Load.



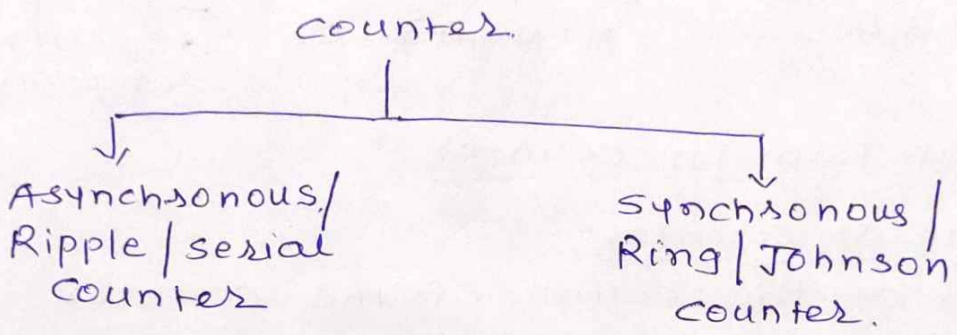
(8.3) Counter

⇒ The digital ckt used for counting pulses is known as counter.

⇒ It is a sequential ckt.

⇒ It is group of FF with a clock signal Applied.

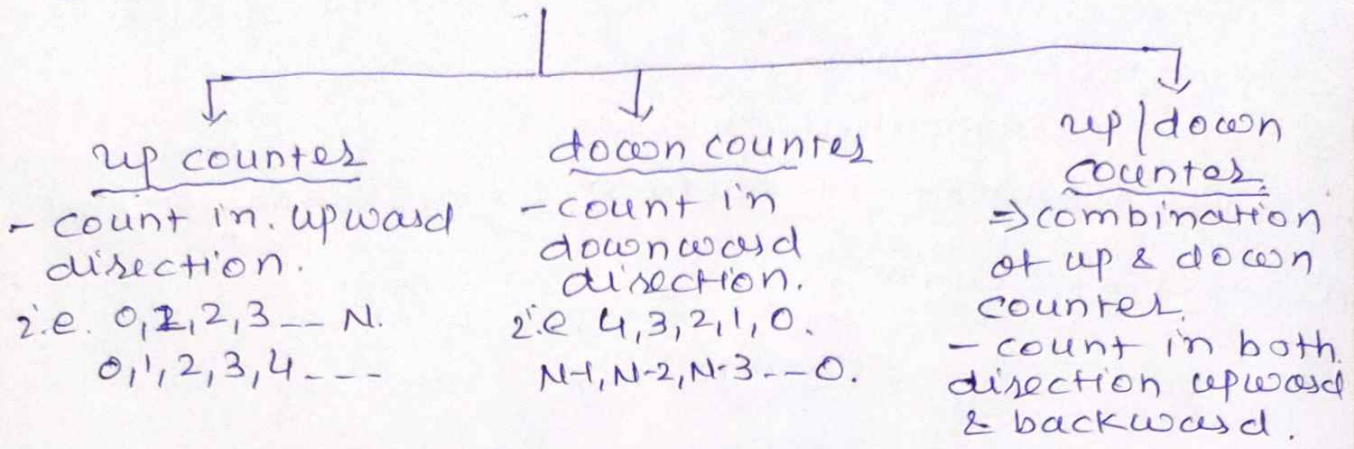
(*) Types of counter



(*) Difference bet^n Asynchsonous & Synchsonous Counter.

Asynchsonous counter	Synchsonous counter
(1) the extesnal clk signal is applied to First FF & then o/p of FF is connected to clock of next Flip Flop.	(1) The extesnal clock is applied to all FF pasallely.
(2) clock is not simultaneously.	(2) clock is simultaneously
(3) ciscuit is simple for the no. of state Increased.	(3) ciscuit become complicated as no. of state is increased.
(4) speed is slow as clock is not simulata-neously	(4) speed is fast as clk signal is applied. Pasallel to all FF.

⇒ Synchsonous & Asynchsonous counter classified as.



⊛ useful Term in counter.

(1) state of counter:

⇒ each of the counts of the counter is called as state of counter.

⇒ N-bit counter ⇒ 2^N states.

i.e 2 bit counter ⇒ $2^2 = 4$ states.

for up counter ⇒ 0,1,2,3

for down counter ⇒ 3,2,1,0.

(2) Modulus | Mod | divide by counter.

⇒ The No. of states through which the counter passes before returning to starting state is called 'Modulus' of counter.

⇒ Mod of counter = No. of states

⇒ divide by counter = No. of states.

In general:

⇒ N-bit counter ⇒ 2^N states

⇒ Mod-2^N counter

= divide by 2^N counter.

Ex. 3-bit counter = $2^3 = 8$ states (0 to 7)

= Mod-8 counter

= divide by 8 counter.

SPP

Maximum count $= 2^N - 1$

for 3 bit $= 2^3 - 1 = 8 - 1 = 7$ (0 to 7)

(8.3.1) Design of Asynchsonous counter (T & JK FF only)

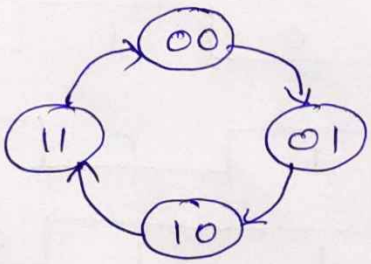
(*) steps to design Asynchsonous counter

(1) state diagsam
(2) state table (PS & NS)
(3) clock conditions.
(4) Logic diagsam / ckt diagsam
(5) wavetosm. (output)

Ex:1 Design 2-bit up Asynchsonous counter using +ve edge trigger clk & T FF.

Ans:

step:1 state diagsam.



step:2 State Table.

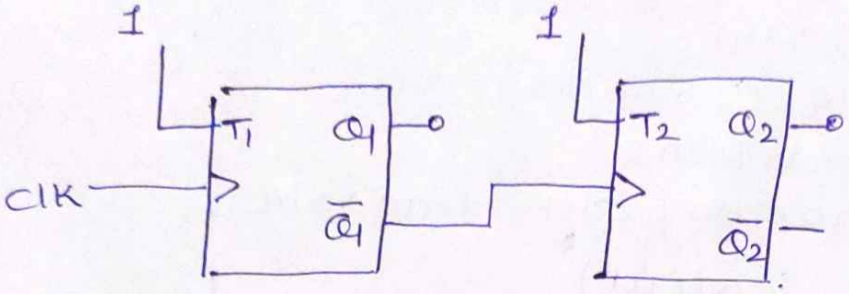
clk.	PS Q_1	PS Q_0	NS $Q_1^{\uparrow}$	NS $Q_0^{\uparrow}$
↑	0	0	0	1
↑	0	1	1	0
↑	1	0	1	1
↑	1	1	0	0

step:3 clock condition

Triggering	up	down.
+ve	$\overline{Q}$	Q
-ve	Q	$\overline{Q}$

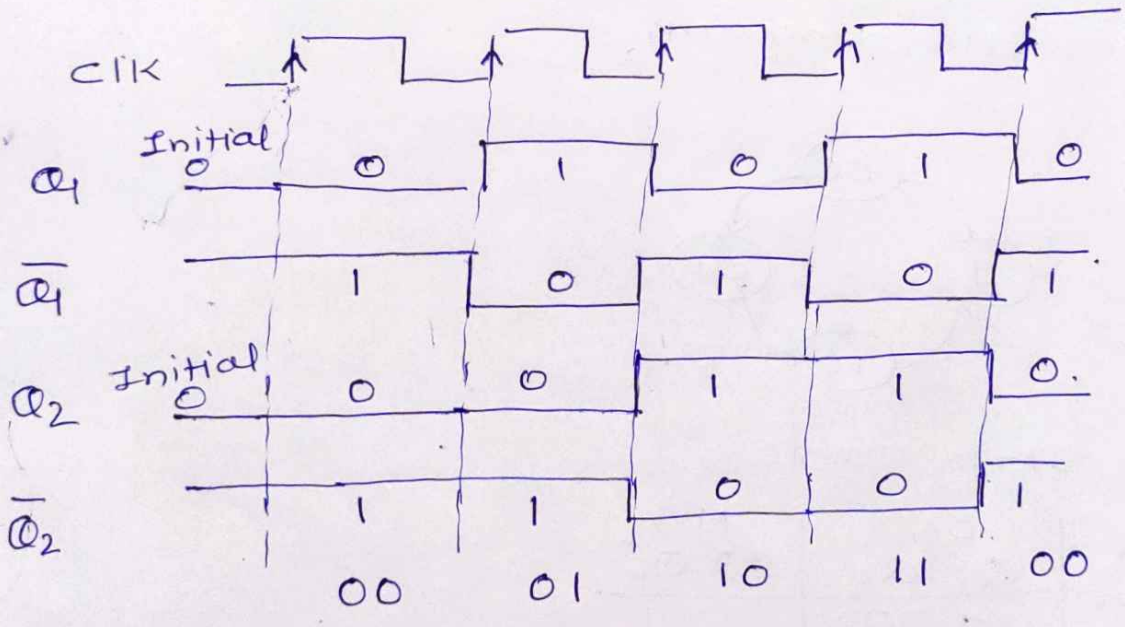
⇒ here In this example, +ve edge trigger clk & up counting sequence so clk of the Next FF is taken from $\bar{Q}$.

Step:4 logic diagram.



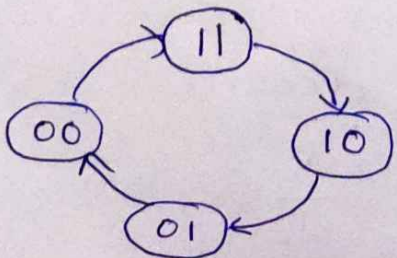
Step:5 output waveform. (from PS table)

⇒ Here clk is taken from $\bar{Q}$ so need to draw waveform of Q & $\bar{Q}$.



Ex:2 Design 2-bit down Asynchronous counter using -ve edge trigger clk & JK FF.

Step:1 state diagram.



Step-2 State table.

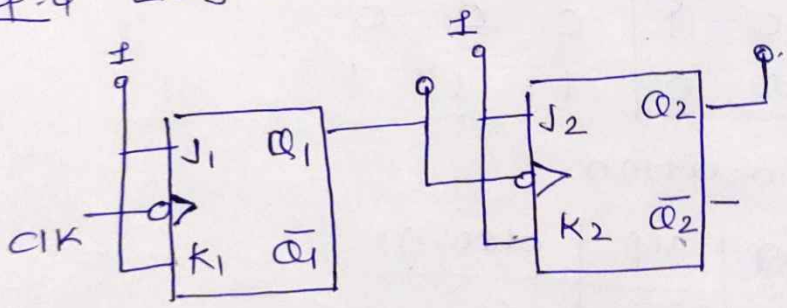
CLK.	PS Q_2	PS Q_1	NS Q_2^*	NS Q_1^*
↓	1	1	1	0
↓	1	0	0	1
↓	0	1	0	0
↓	0	0	1	1

Step:3 Clock conditions.

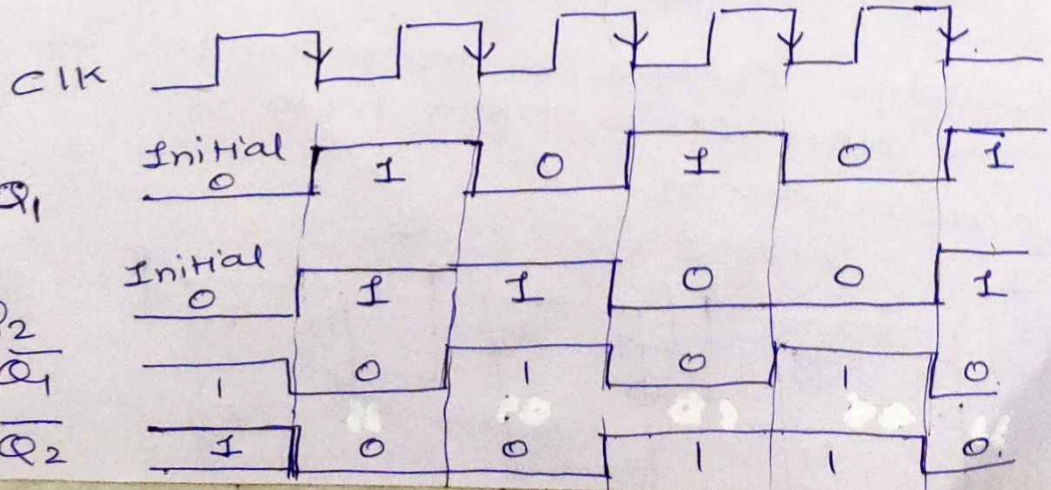
Triggering	up	down
+ve	$\bar{Q}$	Q
-ve	Q	$\bar{Q}$

here in this example, -ve edge triggers clk & down counter so clk of the next FF is taken from Q.

Step:4 Logic diagram.

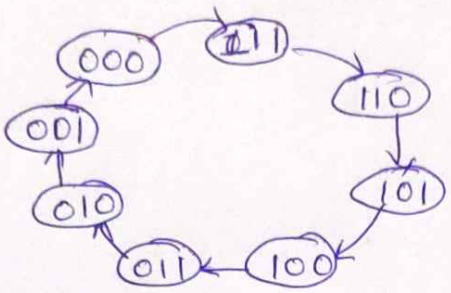


Step:5 Wavefosm.



Ex:3 Design 3-bit Asynchsonous down counter using +ve edge Trigger clk & T FF.

Step:1 state diagram.



Step:2 State table

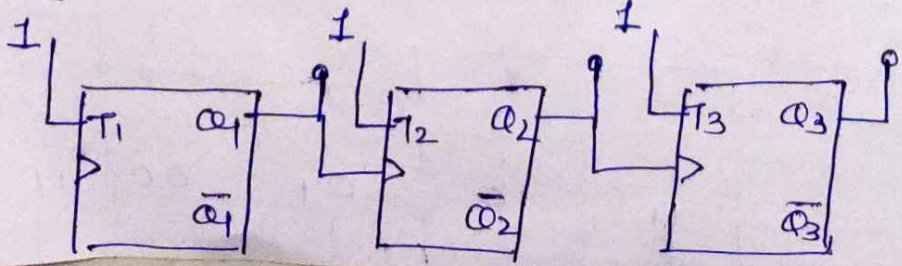
CLK	PS Q_3	Q_2	Q_1	NS Q_3^+	Q_2^+	Q_1^+
↑	1	1	1	1	1	0
↑	1	1	0	1	0	1
↑	1	0	1	1	0	0
↑	1	0	0	0	1	1
↑	0	1	1	0	1	0
↑	0	1	0	0	0	1
↑	0	0	1	0	0	0
↑	0	0	0	1	1	1

Step:3 clock condition

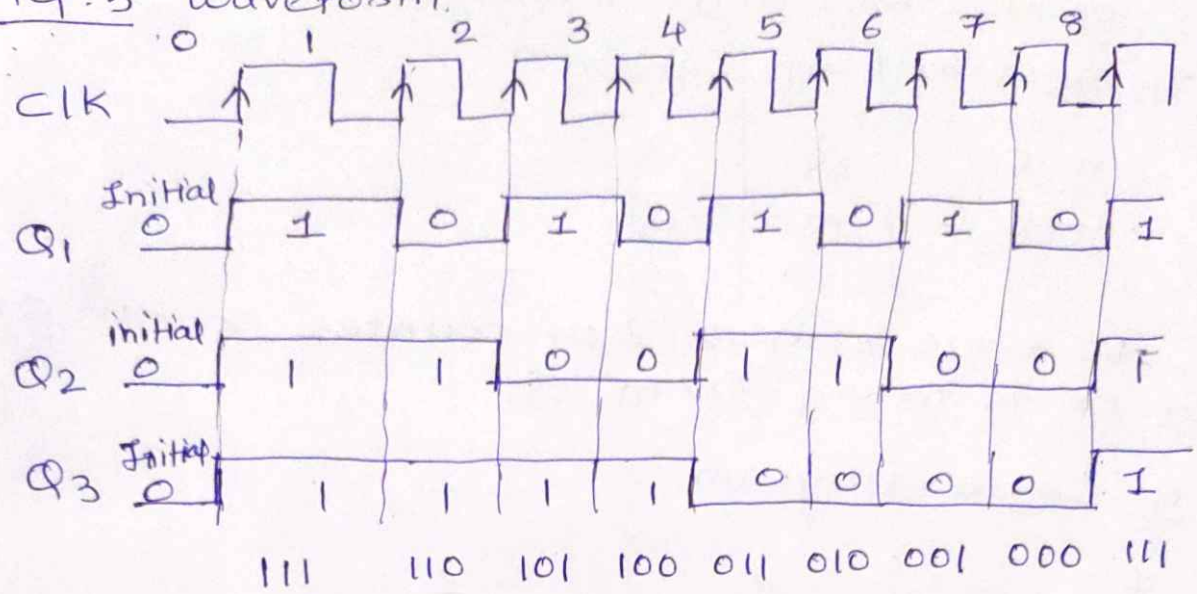
Triggering	up	down
+ve	$\bar{Q}$	Q
-ve	Q	$\bar{Q}$

here, +ve edge Trigger & down counter so clk of the Next FF is taken from Q.

Step:4 Logic diagram

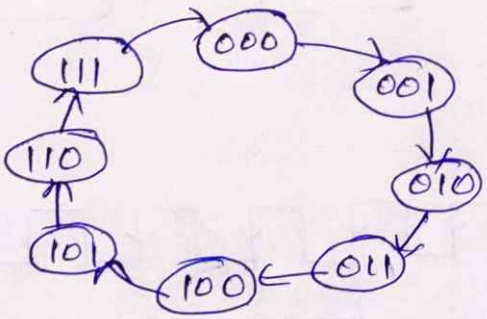


Step:5 Waveform.



Ex:4 Design a 3-bit up Asynchronous counter using -ve edge trigger & JK FF.

Step:1 State diagram.



Step:2 state table.

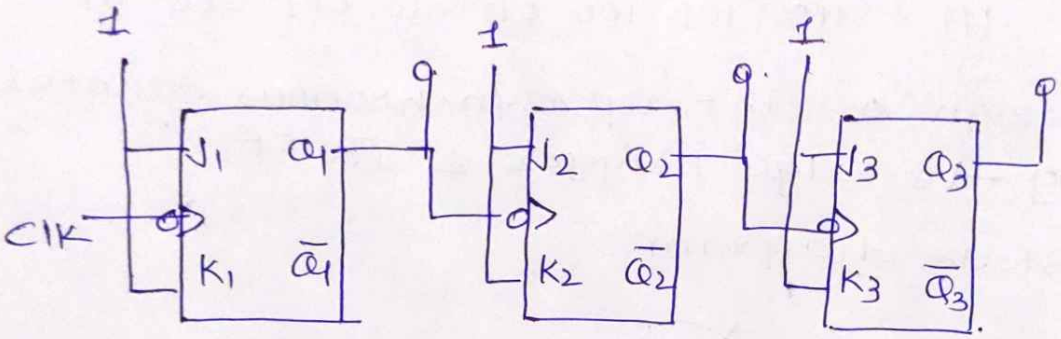
CLK	PS Q_3 Q_2 Q_1	NS Q_3^+ Q_2^+ Q_1^+
↓	0 0 0	0 0 1
↓	0 0 1	0 1 0
↓	0 1 0	0 1 1
↓	0 1 1	1 0 0
↓	1 0 0	1 0 1
↓	1 0 1	1 1 0
↓	1 1 0	1 1 1
↓	1 1 1	0 0 0

Step:3 clock condition

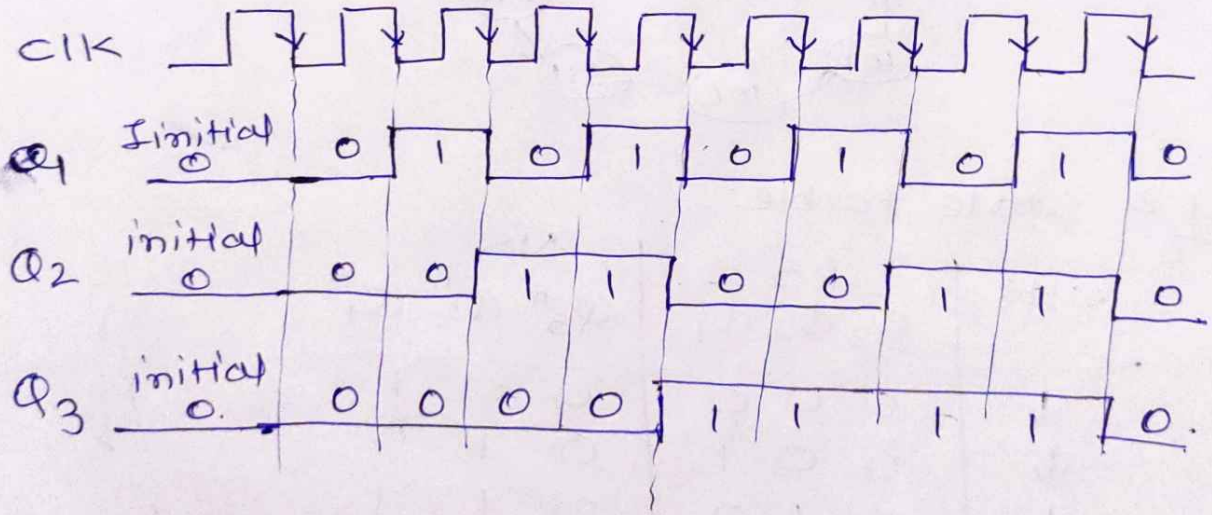
Triggering	up	down
+ve	$\bar{Q}$	Q
-ve	Q	$\bar{Q}$

here, -ve edge trigger & up counter so clk of next FF is taken from Q.

Step:4 Logic diagram.

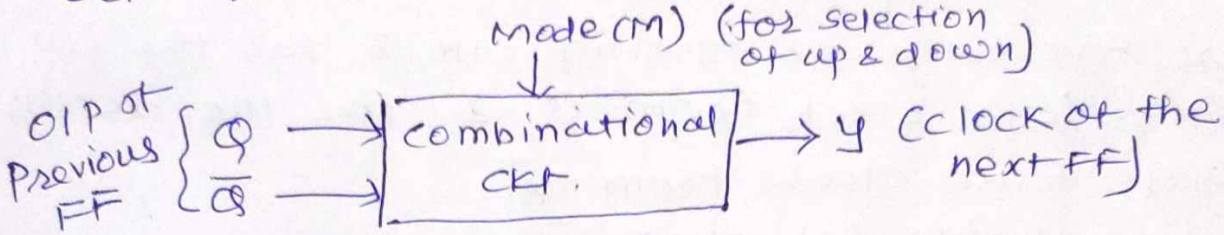


Step:5 Waveform.

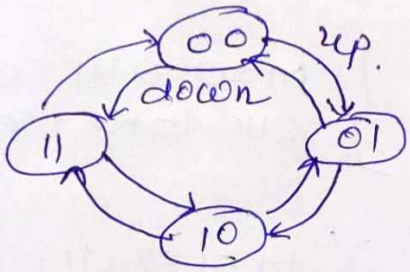


Ex:5 Design 2-bit up/down Asynchsonous counter using -ve edge triggering clk & T FF.

⇒ Need to Find the combinational ckt used betn o/p of FF & clock of the next FF.



step:1 state diagram.



step:2 state table.

clk	PS Q_2	PS Q_1	NS $Q_2^{\uparrow}$	NS Q_1	
↓	0	0	0	1	up
↓	0	1	1	0	
↓	1	0	1	1	
↓	1	1	0	0	
↓	1	1	1	0	down
↓	1	0	0	1	
↓	0	1	0	0	
↓	0	0	1	1	

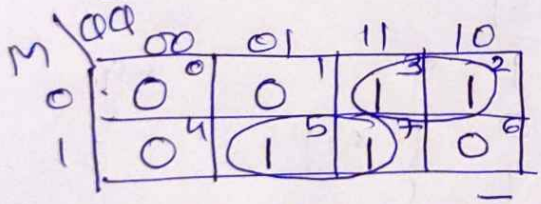
Step:3 clock condition.

Triggering	up	down
+ve	$\bar{Q}$	Q
-ve	Q	$\bar{Q}$

here -ve edge triggering clk & for the up seq. take clock from Q & for the down seq. take clock from $\bar{Q}$.

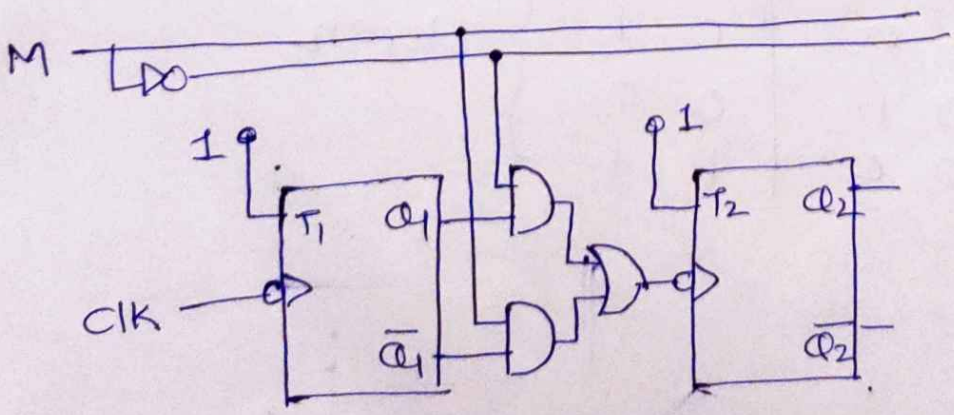
Design table for clock.

M	Q	$\bar{Q}$	Y (clk)	
0	0	0	0	M=0 ⇒ up counting clk take from Q.
0	0	1	0	
0	1	0	1	
0	1	1	1	
1	0	0	0	M=1 ⇒ down counting clk take from $\bar{Q}$
1	0	1	1	
1	1	0	0	
1	1	1	1	

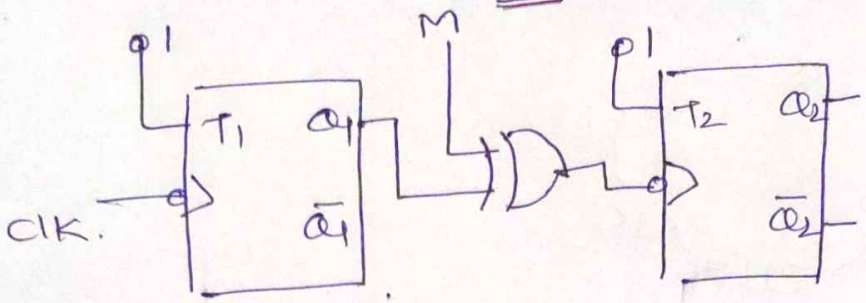


$Y(clk) = \bar{M}Q + M\bar{Q} = M \oplus Q$.

Step:4 Logic diagram.



OR (using Ex-OR gate)



⊛ MOD-N Asynchsonous / Ripple counter

⇒ MOD-N counter
where N = No. of states

i.e if MOD-5 counter.
then No of state = 5
counting seq ⇒ 0 to 4 (0,1,2,3,4,0,1,2 -)

Ex:6 Design Mod-6 Ripple counter using T FF.

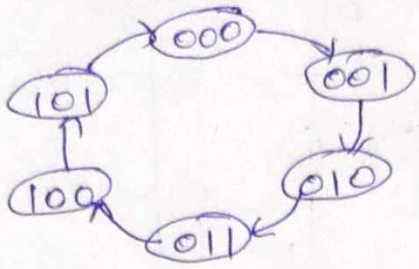
Ans: Mod-6 ⇒ No. of states = 6
⇒ counting seq ⇒ 0,1,2,3,4,5,0,1,2 ---
⇒ state = 000,001,010,011,100,101
It is also called divide by 6 counter.

⇒ In general count seq ⇒ 0 to 5 (000 to 101)
Reset seq ⇒ 110
uncount seq ⇒ 111 (consider as don't care).

⇒ Total FF Required ⇒ here max^m count is 5 so 3-bit operation
↓↓
3 FFs. Req.

⇒ total 8 states in 3-bit counter.
0 to 5 (000 to 101) ⇒ Valid state.
6 & 7 (110 & 111) ⇒ Invalid state.

Step:1 state diagram.



Step:2 state Table

clk.	PS Q_3 Q_2 Q_1	NS Q_3^+ Q_2^+ Q_1^+
↓	0 0 0	0 0 1
↓	0 0 1	0 1 0
↓	0 1 0	0 1 1
↓	0 1 1	1 0 0
↓	1 0 0	1 0 1
↓	1 0 1	0 0 0

Note: take always -ve edge trigger clock while designing MOD-N counter.

Step:3 clock condition

⇒ here -ve edge trigger clock & up seq. so take clock from Q_3.

Step:4 Reset condition.

clk seq.	PS Q_3 Q_2 Q_1	R (Reset)
0	0 0 0	0
1	0 0 1	0
2	0 1 0	0
3	0 1 1	0
4	1 0 0	0
5	1 0 1	0
6	1 1 0	1
7	1 1 1	X

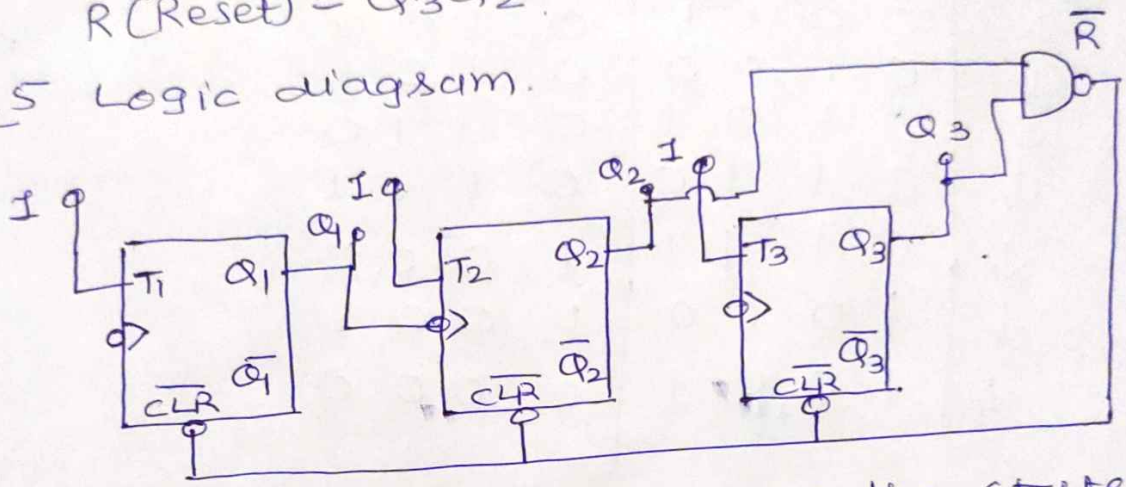
(Remaining state are consider as don't case)

K map for R(Reset)

Q_3 \ Q_2Q_1	00	01	11	10
0	0 (0)	0 (1)	0 (3)	0 (2)
1	0 (4)	0 (5)	X (7)	1 (6)

$R(Reset) = Q_3Q_2$

Step:5 Logic diagram.



⇒ Here R = I but for the CLEAR the state of FF we have to Required LOW i/p so use one ~~not~~ NOT gate after AND gate.

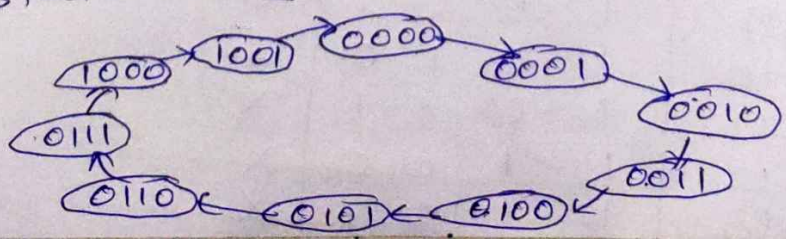
⇒ NAND gate.

Ex:2 Design MOD-10 / BCD / Decade Ripple counter using D FF.

ANS: MOD-10 ⇒ No. of states ⇒ 10.
Counting seq ⇒ 0000 to 1001
Reset cond ⇒ 1010
Invalid / Don't care seq = 1011 to 1111.

FF Req ⇒ 0 to 9 seq so 4-bit counter = 4 FFs Req.

Step:1 state diagram.



Step:2 state table

clk	PS $Q_3Q_2Q_1Q_0$	NS $Q_3^{\uparrow}Q_2^{\uparrow}Q_1^{\uparrow}Q_0^{\uparrow}$
↓	0 0 0 0	0 0 0 1
↓	0 0 0 1	0 0 1 0
↓	0 0 1 0	0 0 1 1
↓	0 0 1 1	0 1 0 0
↓	0 1 0 0	0 1 0 1
↓	0 1 0 1	0 1 1 0
↓	0 1 1 0	0 1 1 1
↓	0 1 1 1	1 0 0 0
↓	1 0 0 0	1 0 0 1
↓	1 0 0 1	0 0 0 0

Step:3 clock condition.

⇒ MOD-N counter so take -ve edge triggered clock pulse.

Step:4 Reset condition.

clk seq	PS $Q_3Q_2Q_1Q_0$	R (Reset)
0	0 0 0 0	0
1	0 0 0 1	0
2	0 0 1 0	0
3	0 0 1 1	0
4	0 1 0 0	0
5	0 1 0 1	0
6	0 1 1 0	0
7	0 1 1 1	0
8	1 0 0 0	0
9	1 0 0 1	0
10	1 0 1 0	1
11	1 0 1 1	X
12	1 1 0 0	X
13	1 1 0 1	X

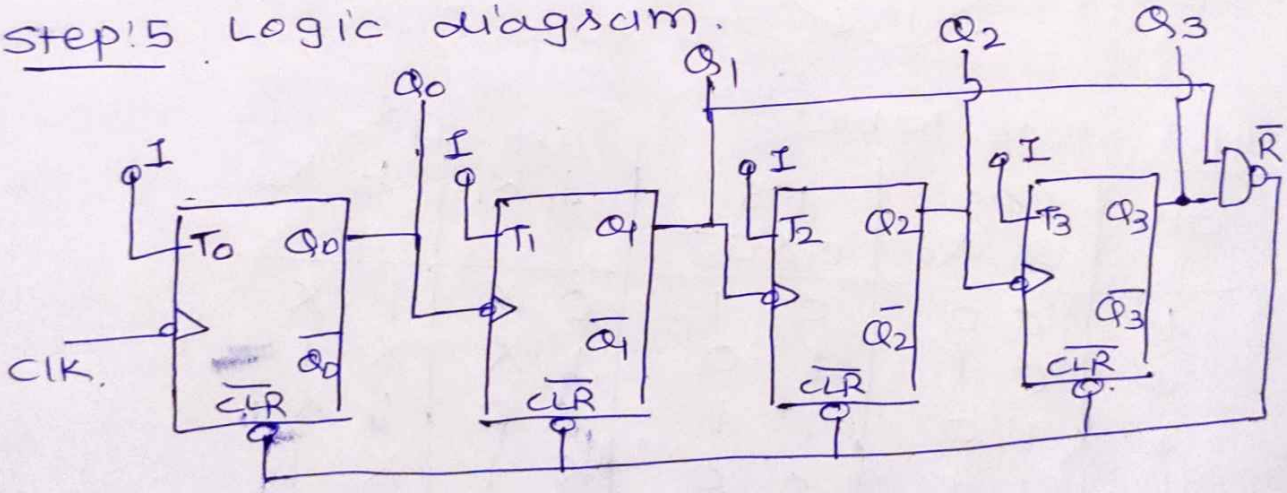
Clk seq	PS $Q_3Q_2Q_1Q_0$	R (Reset)
14	1 1 1 0	X
15	1 1 1 1	X

K-map for R (Reset).

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	0 (0)	0 (1)	0 (3)	0 (2)
01	0 (4)	0 (5)	0 (7)	0 (6)
11	X (12)	X (13)	X (15)	X (14)
10	0 (8)	0 (9)	X (11)	1 (10)

$$R = Q_3Q_1$$

Step 5 Logic diagram.



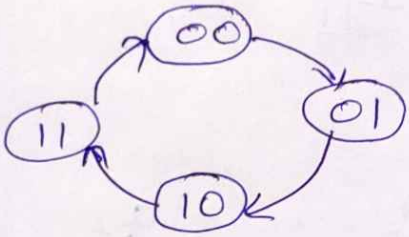
Design of synchsonous counter

⇒ Steps to design synchsonous counter

(1) State diagsam.
(2) State table (PS, NS, Ext. I/Ps)
(3) Boolean eqn
(4) Logic diagsam.

Ex:1 Design 2-bit synchsonous up counter using -ve edge trigger clk & JK FF.

Step:1 state diagsam.



Step:2 state table

clk	PS Q_1 Q_0	NS Q_1^* Q_0^*	Ext. I/Ps J_1 K_1	J_2 K_2
↓	0 0	0 1	0 X	1 X
↓	0 1	1 0	1 X	X 1
↓	1 0	1 1	X 0	1 X
↓	1 1	0 0	X 1	X 1

Step:3 boolean eqn

$J_1 \Rightarrow$

Q_1 \ Q_0	0	1
0	0	1
1	X	X

$J_1 = Q_0$

$K_1 \Rightarrow$

Q_1 \ Q_0	0	1
0	X	X
1	0	1

$K_1 = Q_0$

$J_2 \Rightarrow$

Q_1 \ Q_0	0	1
0	1	X
1	1	X

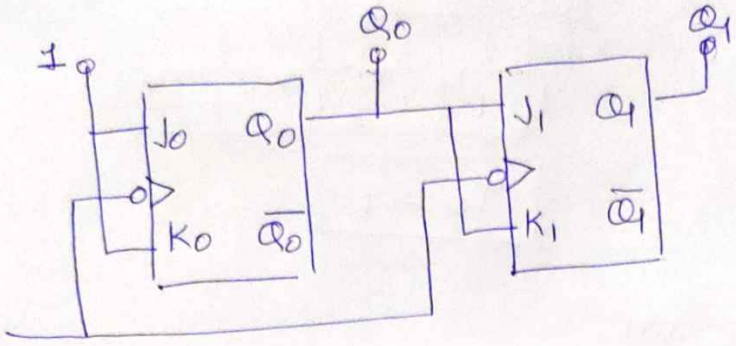
$J_0 = 1$

$K_2 \Rightarrow$

Q_1 \ Q_0	0	1
0	X	1
1	X	1

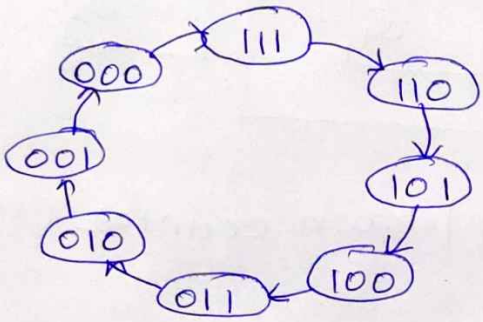
$K_0 = 1$

Step:4 logic diagram.



Ex:2 Design 3-bit down synchronous counter using +ve edge trigger clk & TFF.

Step:1 State diagram.



Step:2 State Table

	clk	PS Q_2	Q_1	Q_0	NS Q_2^+	Q_1^+	Q_0^+	Ext. I/Ps T_2	T_1	T_0
7	↑	1	1	1	1	1	0	0	0	1
6	↑	1	1	0	1	0	1	0	1	1
5	↑	1	0	1	1	0	0	0	0	1
4	↑	1	0	0	0	1	1	1	1	1
3	↑	0	1	1	0	1	0	0	0	1
2	↑	0	1	0	0	0	1	0	1	1
1	↑	0	0	1	0	0	0	0	0	1
0	↑	0	0	0	1	1	1	1	1	1

Step:3 Boolean eqn

$T_2 \Rightarrow$

Q_2 \ Q_1Q_0	00	01	11	10
0	1 (0)	0 (1)	0 (3)	0 (2)
1	1 (4)	0 (5)	0 (7)	0 (6)

$T_2 = \bar{Q}_1\bar{Q}_0$

For T_1 ⇒

Q_2 \ Q_1Q_0	00	01	11	10
0	1 (0)	0 (1)	0 (3)	1 (2)
1	1 (4)	0 (5)	0 (7)	1 (6)

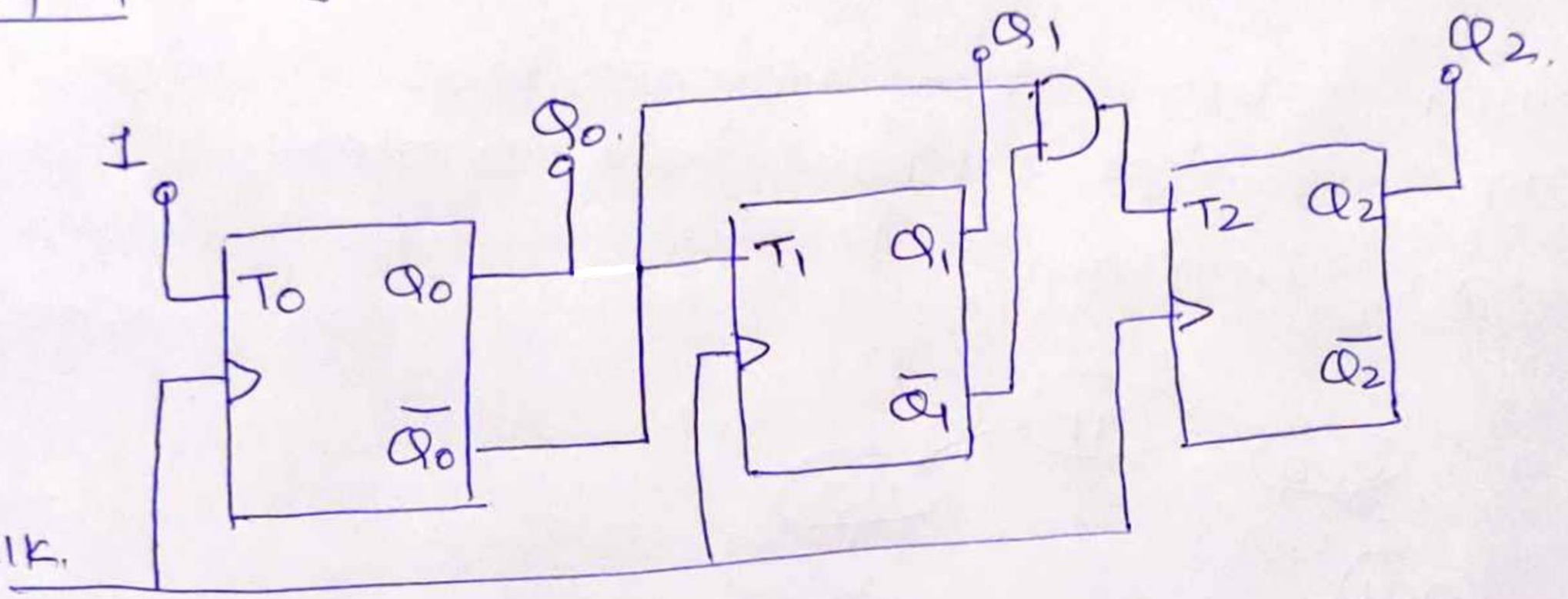
$$T_1 = \overline{Q_0}$$

for T_0 ⇒

Q_2 \ Q_1Q_0	00	01	11	10
0	1 (0)	1 (1)	1 (3)	1 (2)
1	1 (4)	1 (5)	1 (7)	1 (6)

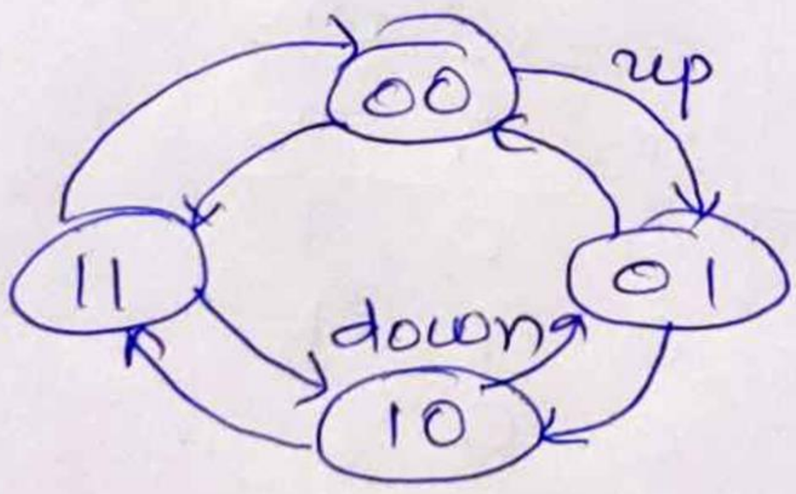
$$T_0 = 1$$

Step:4 Logic diagsam.



Ex:3 Design 2-bit up/down counter (synchsonous) using T-FF.

Step:1 state diagsam.



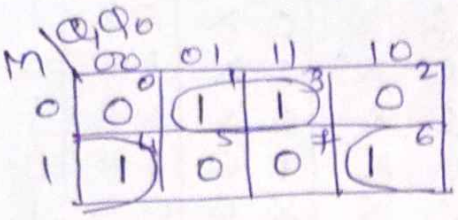
M=0 ⇒ up
M=1 ⇒ down

Step:2 state table.

	M	PS Q_1	PS Q_0	NS $Q_1^{\uparrow}$	NS $Q_0^{\uparrow}$	Ext-I/Ps T_1	Ext-I/Ps T_0
up.	0	0	0	0	1	0	1
	0	0	1	1	0	1	1
	0	1	0	1	1	0	1
	0	1	1	0	0	1	1
down	1	0	0	1	1	1	1
	1	0	1	0	0	0	1
	1	1	0	0	1	1	1
	1	1	1	1	0	0	1

Step:3 Boolean eqn.

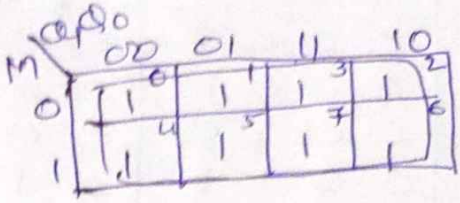
for $T_1 \Rightarrow$



$T_1 = \bar{M}Q_0 + M\bar{Q_0}$

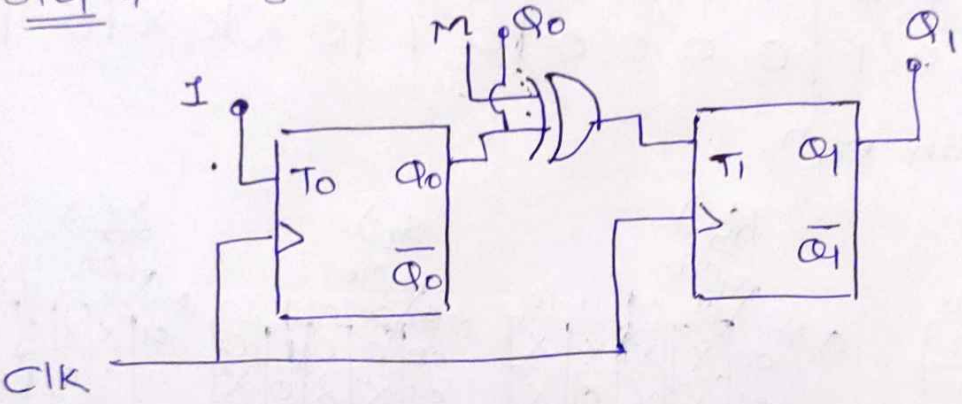
$= M \oplus Q_0$.

for $T_0 \Rightarrow$.



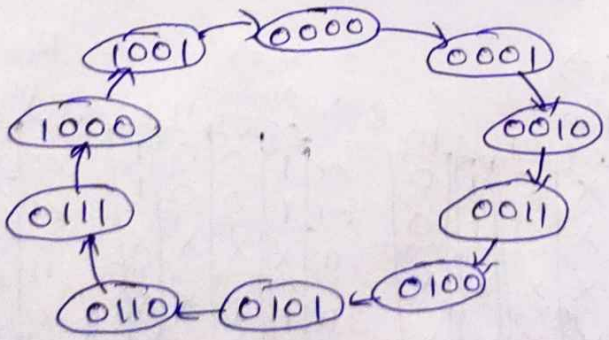
$T_0 = 1$

Step:4 Logic diagram.



Ex:4 Design BCD synchronous counter using SR FF.

Step:1 State diagram.

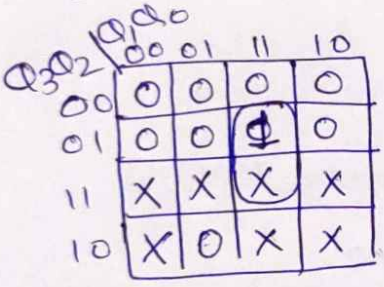


Step:2 State Table.

Clk	PS $Q_3Q_2Q_1Q_0$	NS $Q_3^+Q_2^+Q_1^+Q_0^+$	Ext. IIPs. S_3R_3	S_2R_2	S_1R_1	S_0R_0
↑	0 0 0 0	0 0 0 1	0 X	0 X	0 X	1 0
↑	0 0 0 1	0 0 1 0	0 X	0 X	1 0	0 1
↑	0 0 1 0	0 0 1 1	0 X	0 X	X 0	1 0
↑	0 0 1 1	0 1 0 0	0 X	1 0	0 1	0 1
↑	0 1 0 0	0 1 0 1	0 X	X 0	0 X	1 0
↑	0 1 0 1	0 1 1 0	0 X	X 0	1 0	0 1
↑	0 1 1 0	0 1 1 1	0 X	X 0	X 0	1 0
↑	0 1 1 1	1 0 0 0	1 0	0 1	0 1	0 1
↑	1 0 0 0	1 0 0 1	X 0	0 X	0 X	1 0
↑	1 0 0 1	0 0 0 0	0 1	0 X	0 X	0 1

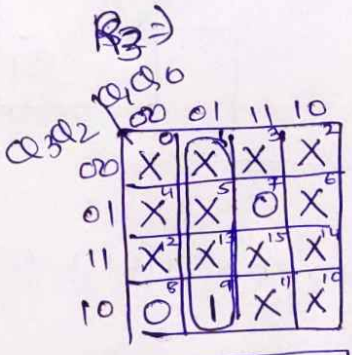
Step:3 Boolean eqn.

$S_3 \Rightarrow$



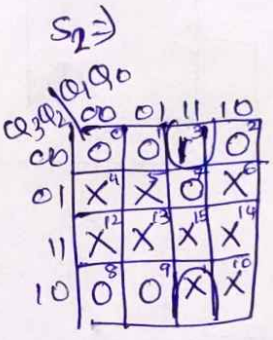
$S_3 = Q_2Q_1Q_0$

$R_3 \Rightarrow$



$R_3 = \bar{Q}_1Q_0$

$S_2 \Rightarrow$

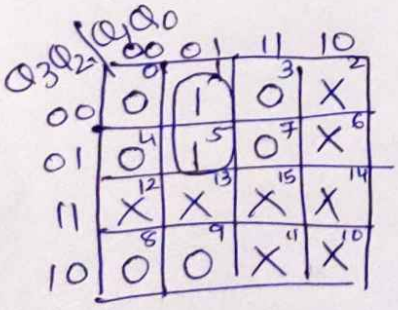


$S_2 = \bar{Q}_2Q_1Q_0$

$R_2 \Rightarrow$

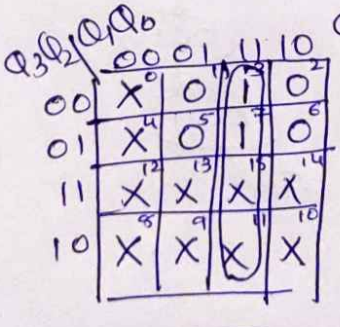
$R_2 = Q_2Q_1Q_0$

$S_1 \Rightarrow$



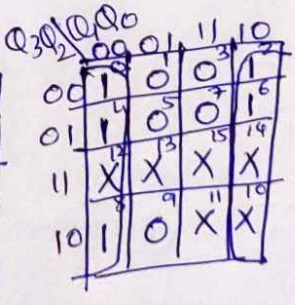
$S_1 = Q_3\bar{Q}_1Q_0$

$R_1 \Rightarrow$



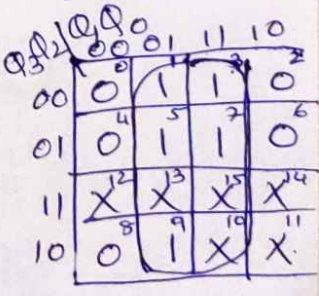
$R_1 = Q_1Q_0$

$S_0 \Rightarrow$



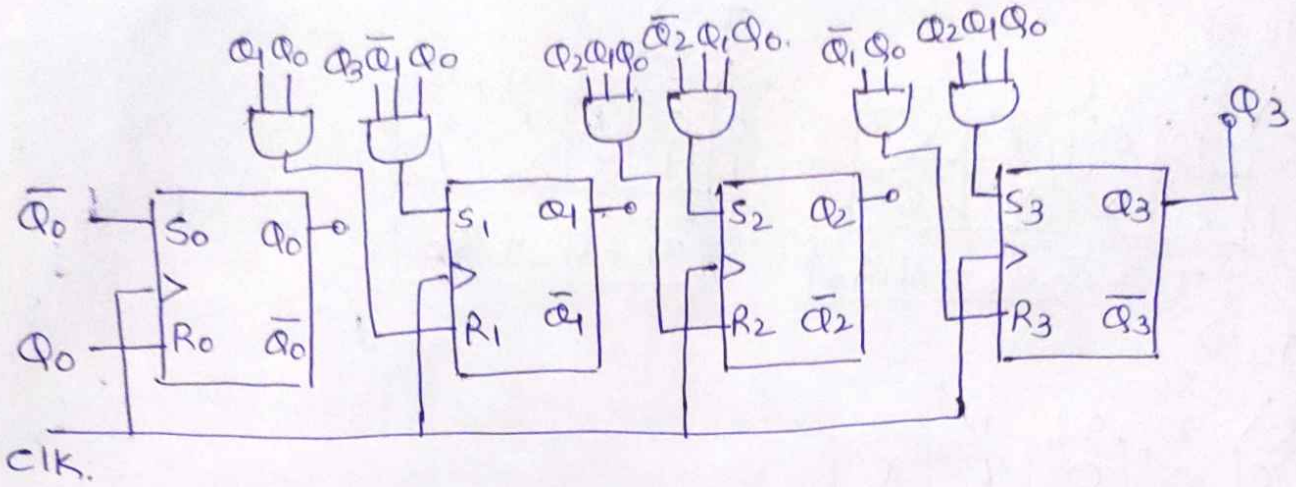
$S_0 = \bar{Q}_0$

$R_0 \Rightarrow$



$R_0 = Q_0$

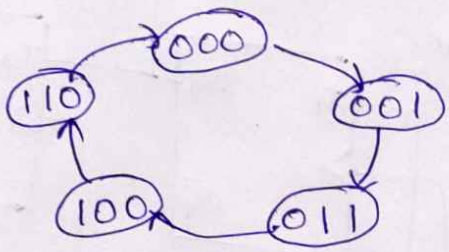
Step:4 Logic diagsam.



Ex:5 Design synchsonous counter that follows the sequence: 0,1,3,4,6,0,1,3---. using T FF.

Step:1 State diagsam.

here max^m count = 6. so 3-bit = 3FF Req.



Step:2 State table

	clk	PS $Q_2Q_1Q_0$	NS $Q_2^+Q_1^+Q_0^+$	Ext I/Ps. T_2 T_1 T_0
0	↑	0 0 0	0 0 1	0 0 1
1	↑	0 0 1	0 1 1	0 1 0
3	↑	0 1 1	1 0 0	1 1 1
4	↑	1 0 0	1 1 0	0 1 0
6	↑	1 1 0	0 0 0	1 1 0

here count seq ⇒ 0,1,3,4,6.

uncount seq ⇒ 2,5,7 ⇒ consider as don't care

Step:3 Boolean eqn.

$T_0 \Rightarrow$

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

$T_0 = \bar{Q}_2\bar{Q}_0 + Q_1Q_0$

$T_1 \Rightarrow$

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

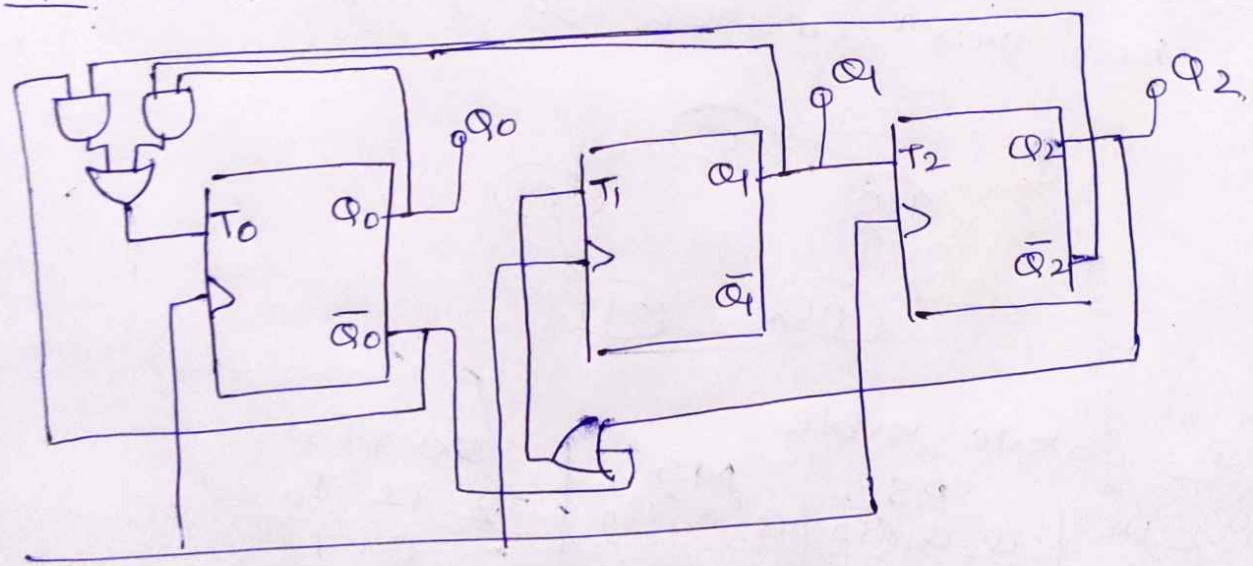
$T_1 = Q_2 + Q_0$

$T_2 \Rightarrow$

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

$T_2 = Q_1$

Step:4 Logic diagram

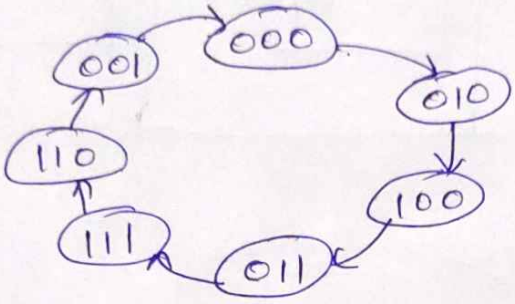


Ex:6 Design counter (synchronous) that perform following seq. using T FF.

0, 2, 4, 3, 7, 6, 1, 0, 2, 4 ---

Step:1 state diagram

here max^m count = 7 so 3-bit = 3 FF Req.

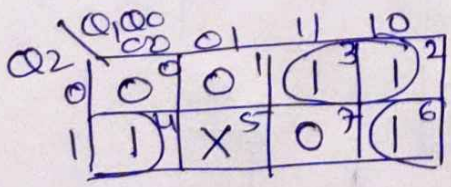


Step:2 state table.

CLK	PS Q_2 Q_1 Q_0	NS Q_2^+ Q_1^+ Q_0^+	Ext. IIPs T_2 T_1 T_0
↑	0 0 0	0 1 0	0 1 0
↑	0 0 1	0 0 0	0 0 1
↑	0 1 0	1 0 0	1 1 0
↑	0 1 1	1 1 1	1 0 0
↑	1 0 0	0 1 1	1 1 1
↑	1 0 1	X X X	X X X
↑	1 1 0	0 0 1	1 1 1
↑	1 1 1	1 1 0	0 0 1
↑			

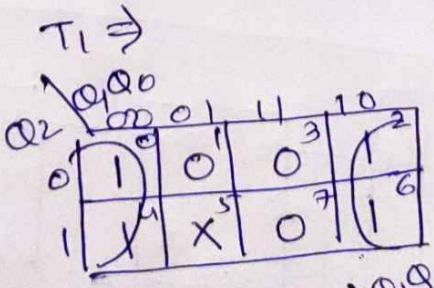
Step:3 Boolean eqn.

$T_2 \Rightarrow$



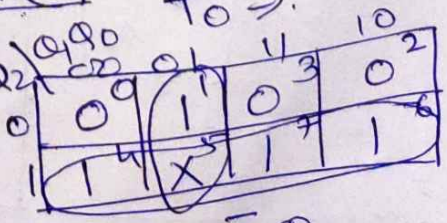
$T_2 = Q_2\bar{Q}_0 + \bar{Q}_2 Q_1$

$T_1 \Rightarrow$



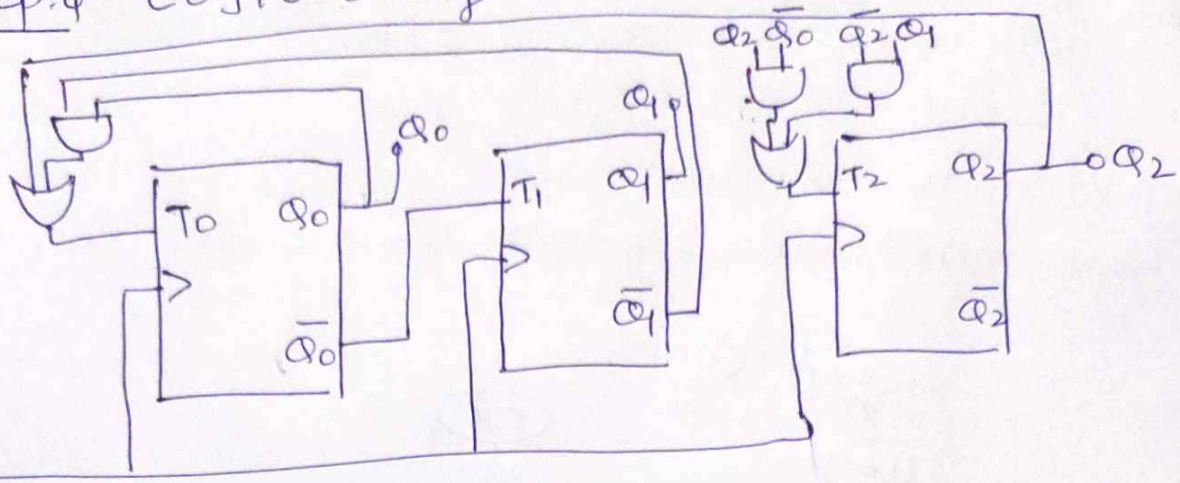
$T_1 = \bar{Q}_0$

$T_0 \Rightarrow$



$T_0 = Q_2 + \bar{Q}_1 Q_0$

Step:4 Logic diagsam.



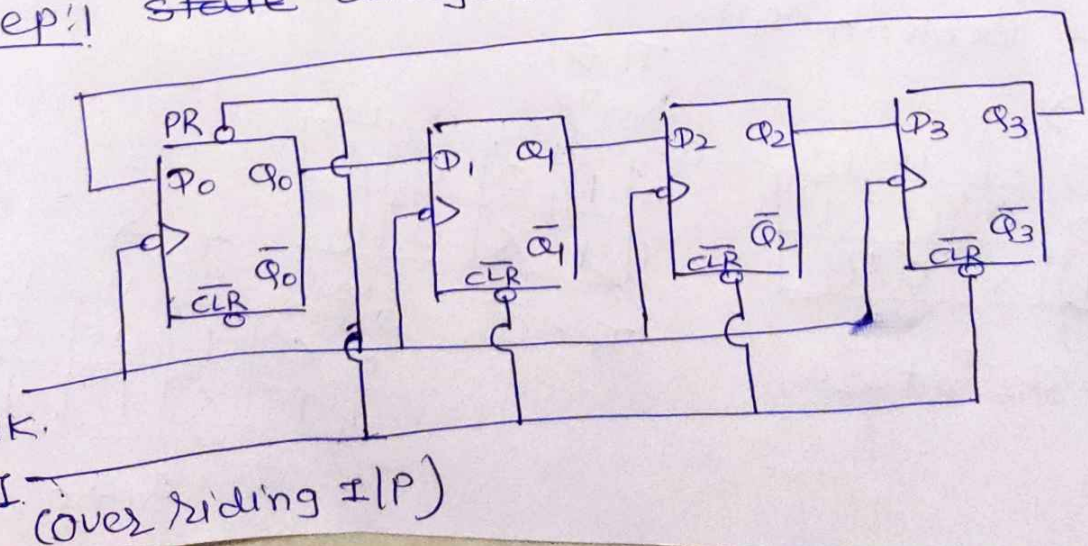
(8.3.3) Shift Register counter.

(1) Ring counter.

⇒ It is Application of shift Register.
⇒ OIP of Last FF is connected to iIP of First FF.
⇒ It is synchronous counter.
⇒ It is Rotate Right shift Register.
⇒ No. of states = No. of FFs.

Ex:1 Design 4-bit Ring counter D-FF & take -ve edge trigger clock.

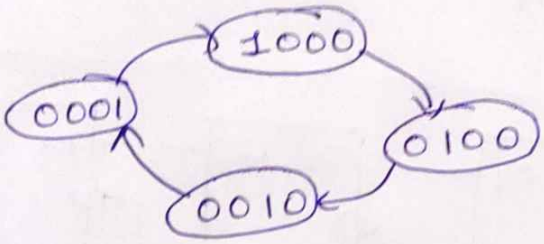
Ans:
Step:1 Logic ~~State~~ diagsam.



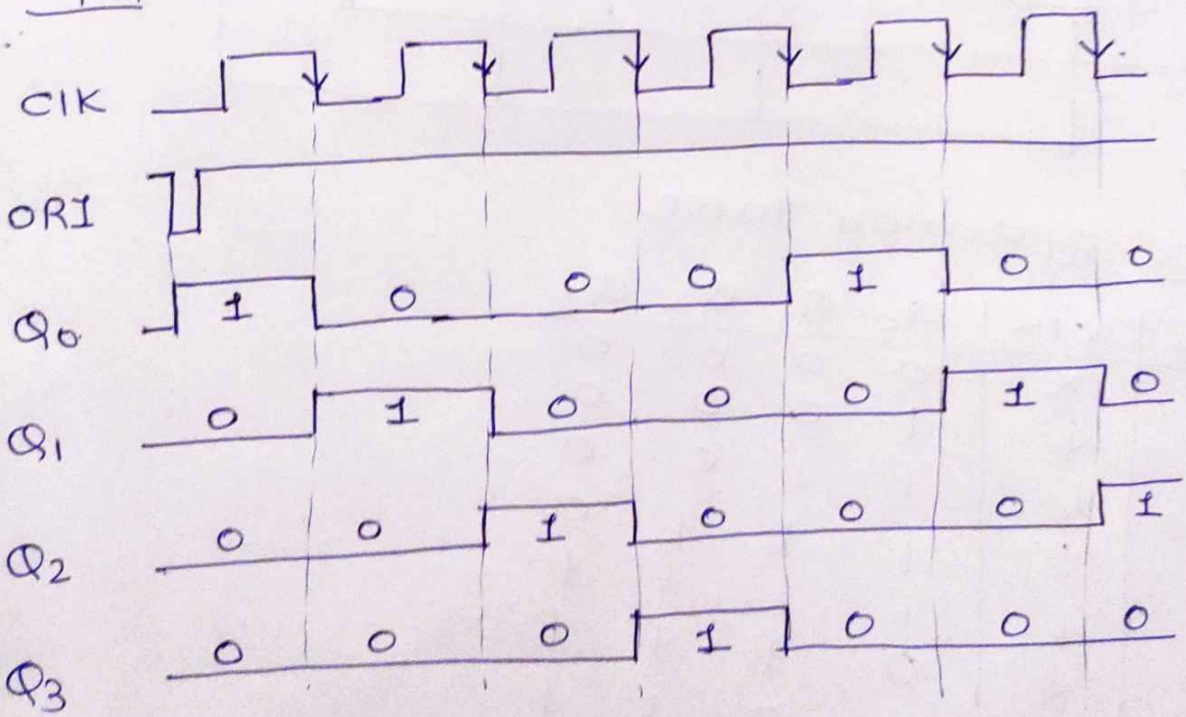
Step:2 operation Table.

ORI	CLK	Q_0	Q_1	Q_2	Q_3
↓	X	1	0	0	0
↑	↓	0	1	0	0
↑	↓	0	0	1	0
↑	↓	0	0	0	1
↑	↓	1	0	0	0

Step:3 state diagsam.



Step:4 waveform.

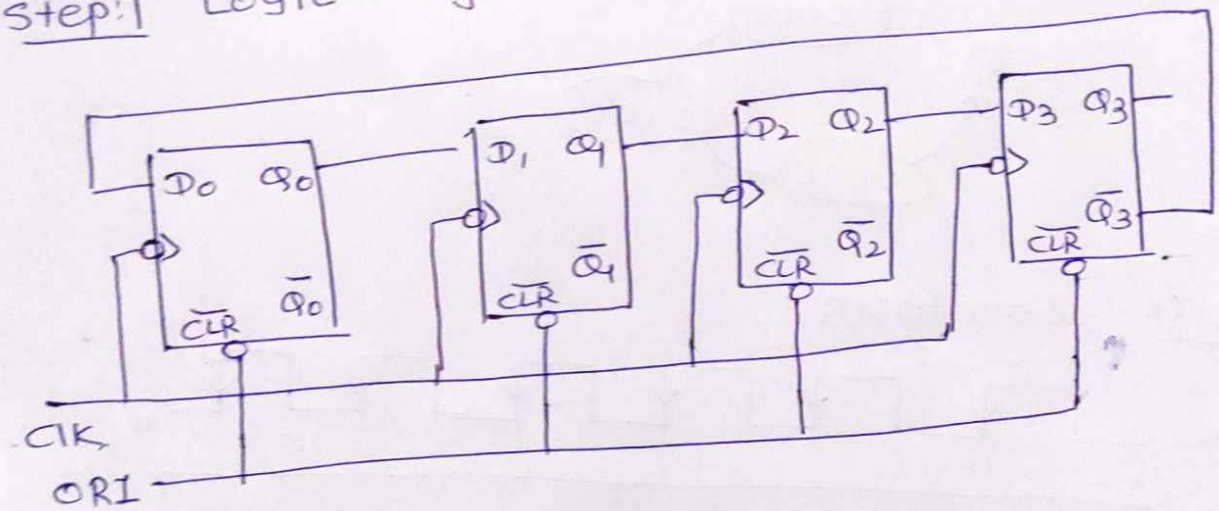


(2) Johnson counter / Twisted Ring / Tail Ring counter

⇒ It is synchsonous counter
⇒ It is Applicaution of shift Register.
⇒ DisAdvantage of Ring counter is that it is working for limited state which is Resolve in Johnson counter
⇒ No of states = 2 × No. of FFs.

Ex:1 Design 4-bit Johnson counter using D FF.

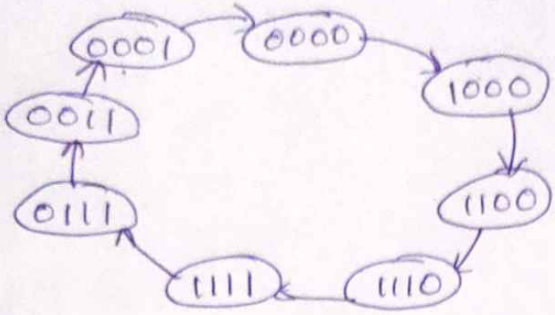
Step:1 Logic diagsam.



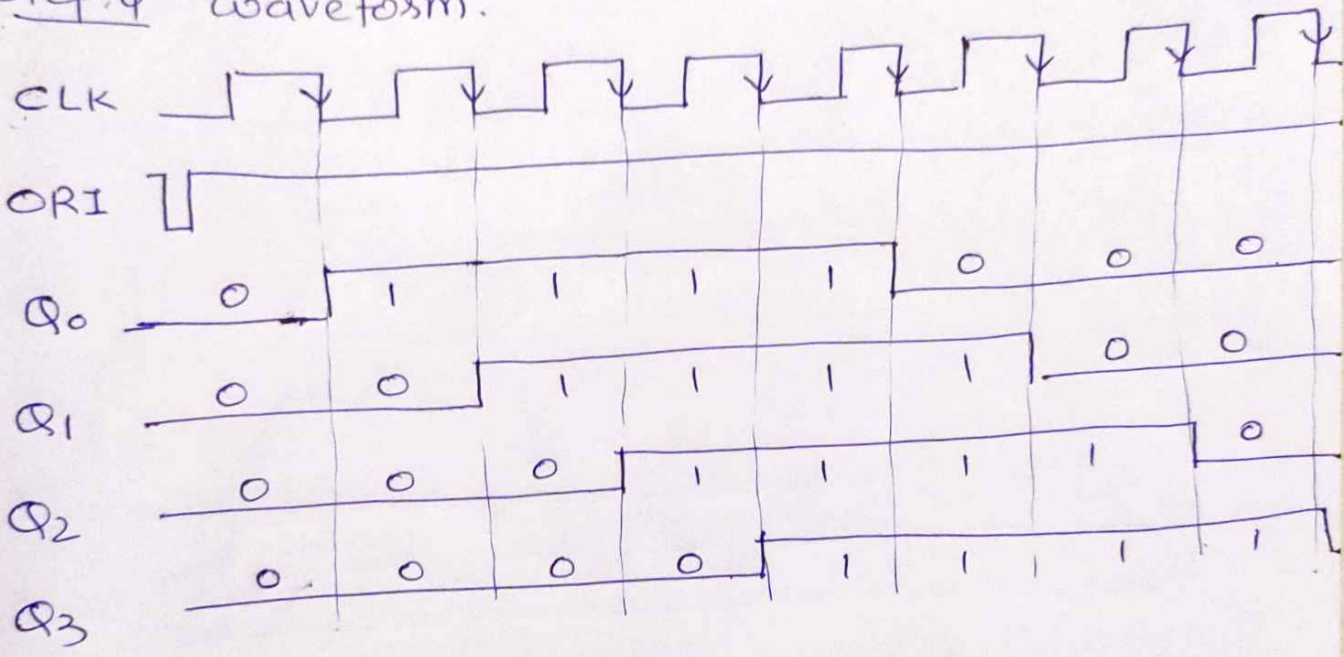
Step:2 operation Table.

ORI	elk	Q_0	Q_1	Q_2	Q_3
↓	X	0	0	0	0
↑	↓	1	0	0	0
↑	↓	1	1	0	0
↑	↓	1	1	1	0
↑	↓	1	1	1	1
↑	↓	0	1	1	1
↑	↓	0	0	1	1
↑	↓	0	0	0	1
↑	↓	0	0	0	0

Step:3 State diagram



Step:4 waveform.



* Johnson counter using JK FF.