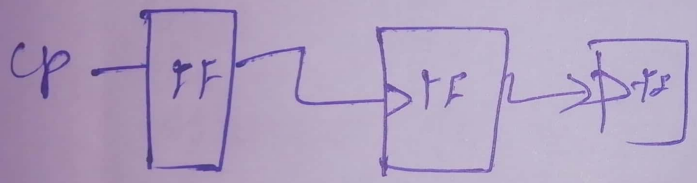
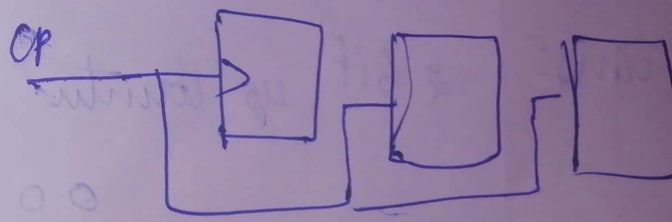
MODULE - V

Counter

Asynchronous counter (Ripple counter)



Synchronous counter



⇒ up counter
⇒ Down counter
⇒ up / down counter

Asynchronous - toggling FF
JK / T flipflop

Counter → -ve edge triggered is commonly used

⇒ Design a two bit asynchronous up counter / two bit ripple up counter / Mod-4 (4 states 2 bit) / asynchronous upcounter / Mod-4 ripple up counter.

Mod 4 - (0-3) 2 flipflop
Mod 16 - (0-15) 4 flip

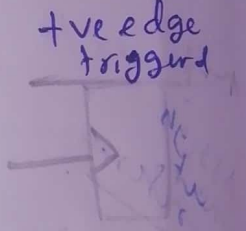
n-flipflop
2^n-states

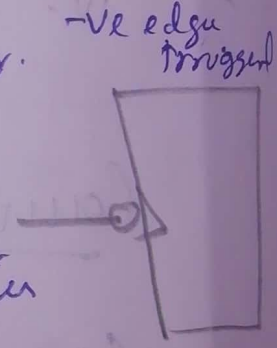
Two bit
00, 01, 10, 11
2 flipflop

-ve edge triggered

Q as next clock, upcounter

$\bar{Q}$ " " " , down counter.





+ve edge triggered

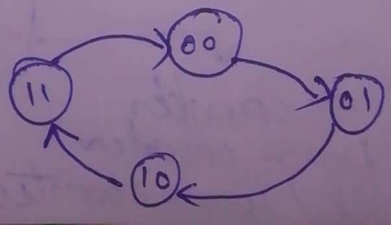
Q as next clock, down counter

$\bar{Q}$ " " " , up counter

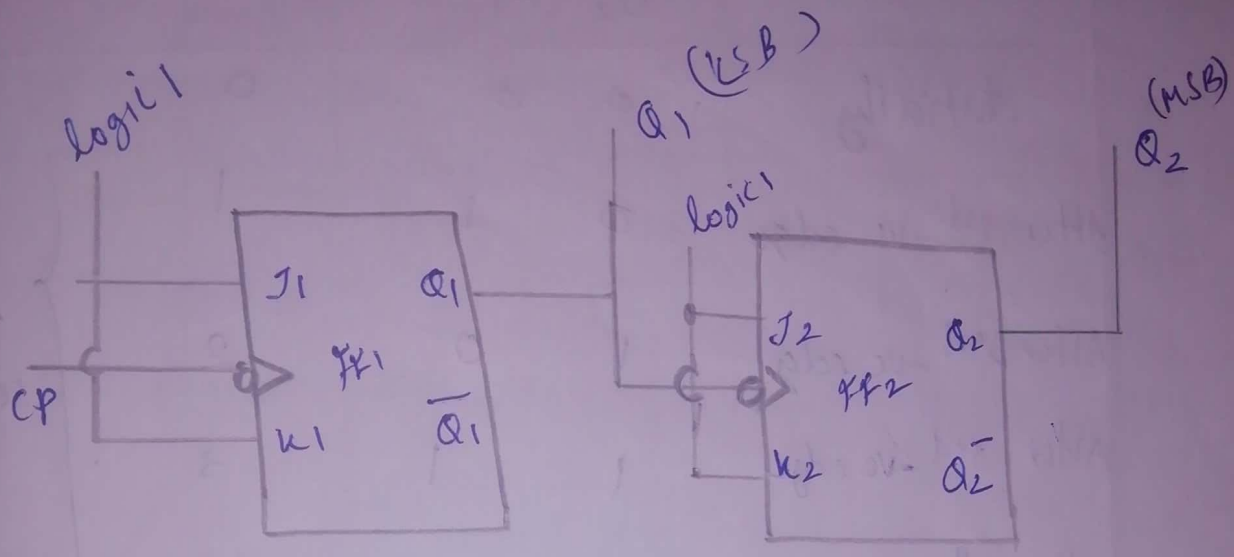
Ans:- 2 bit up counter

00 01 10 11

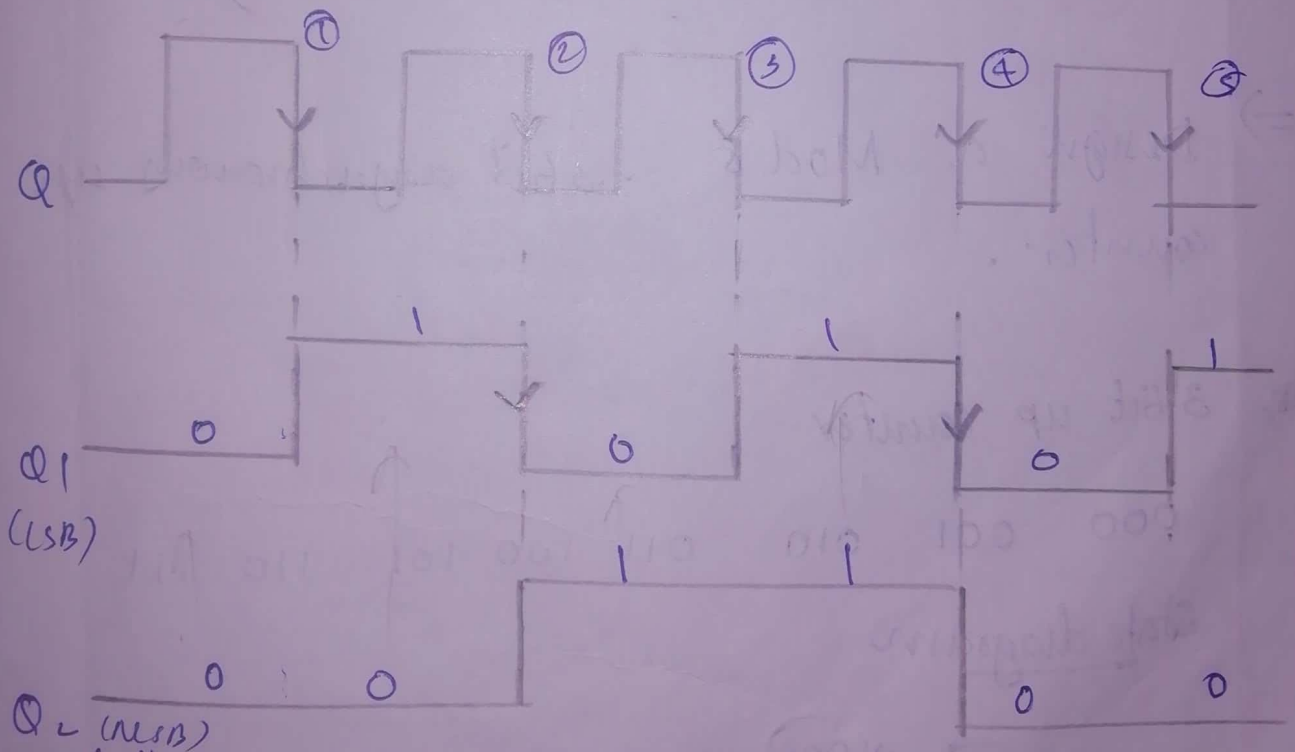
state diagram



Circuit diagram



Timing Diagram



Initially
Assume $Q_1 = 0$, $Q_2 = 0$

clock (CP)	state Q_2 Q_1	Decimal
Initially	0 0	0
After 1st -ve edge	0 1	1
After 2nd -ve edge	1 0	2
After 3rd -ve edge	1 1	3
" 4th "	0 0	0
" 5th "	0 1	1

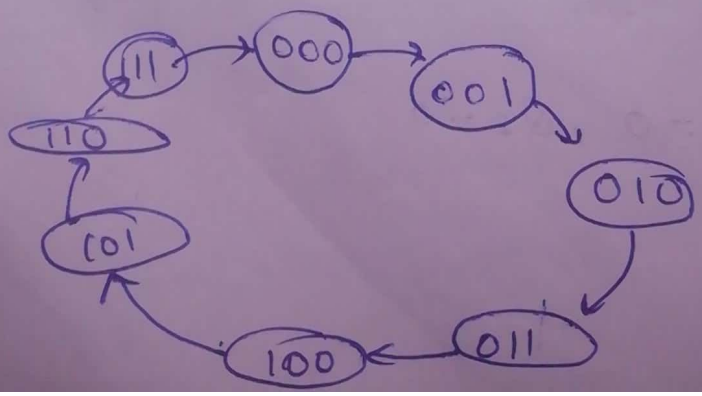
up counting.

⇒ Design a Mod 8 - 3 bit asynchronous up counter.

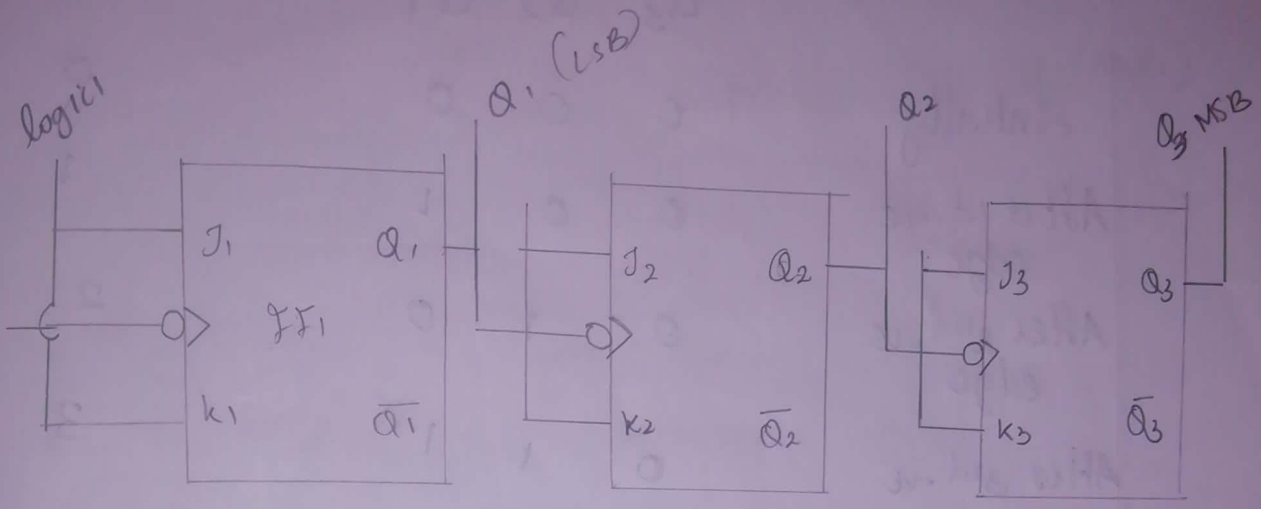
Ans:- 3 bit up counter

000 001 010 011 100 101 110 111

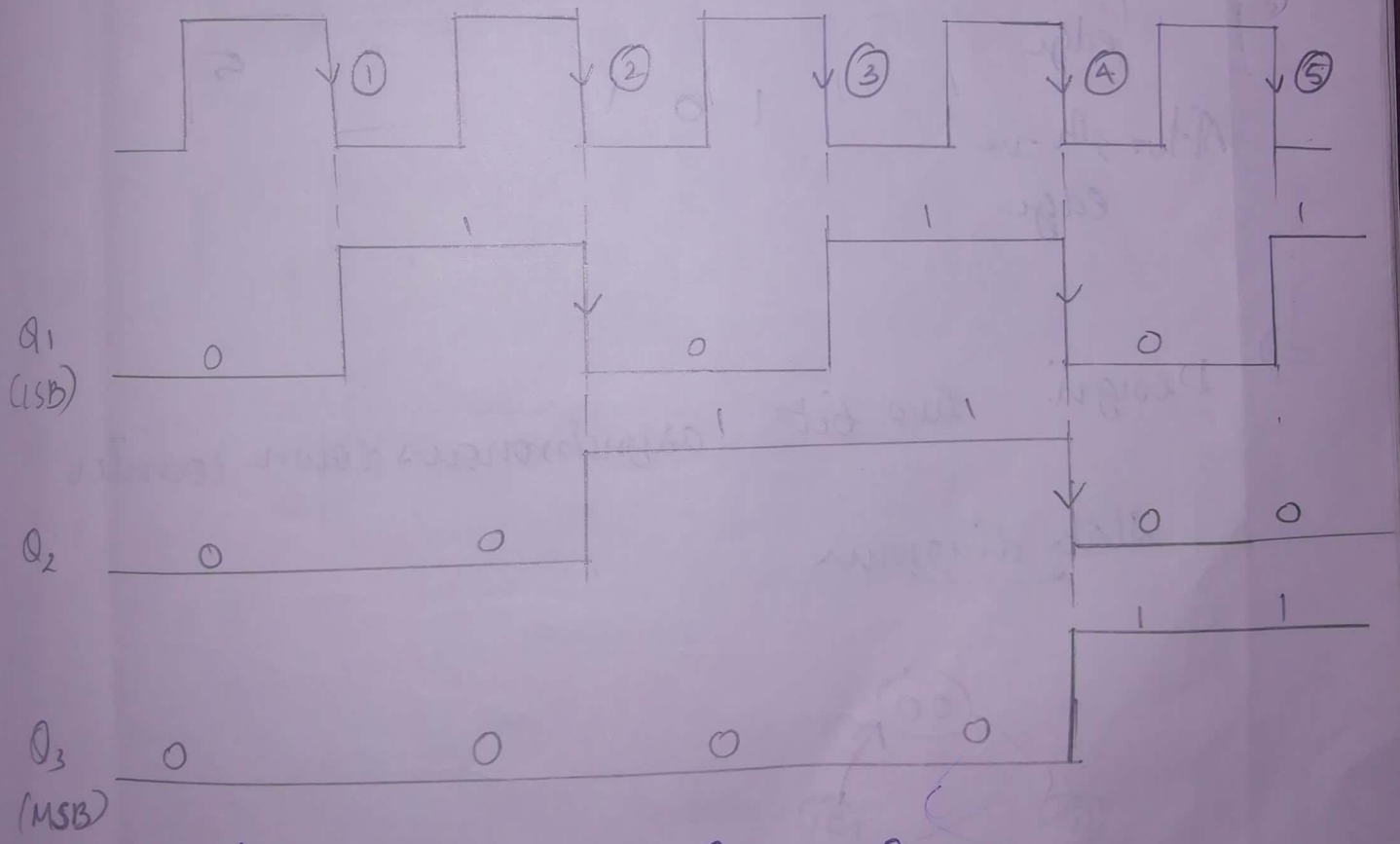
State diagram



Circuit diagram



Timing Diagram

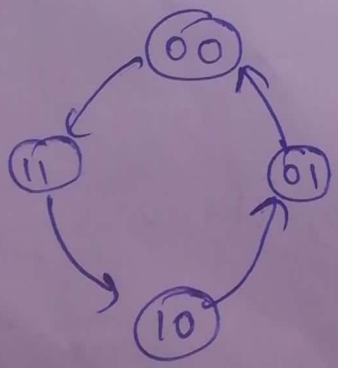


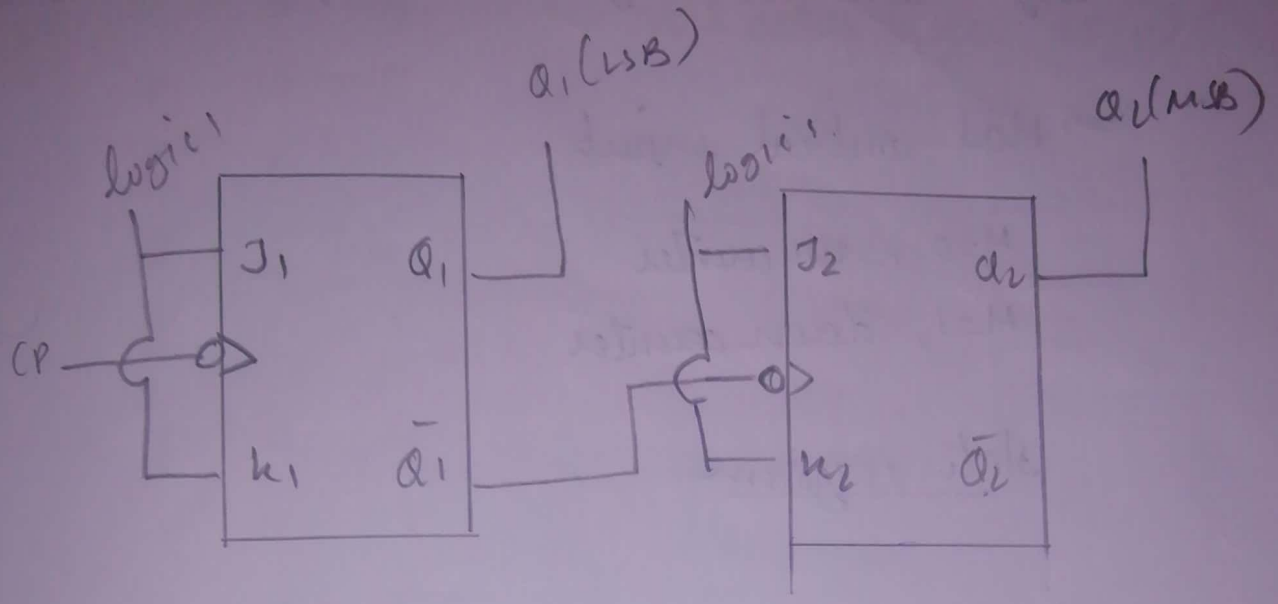
Initially assume $Q_1 = 0$, $Q_2 = 0$, $Q_3 = 0$

Clock(P)	State Q_3	Q_2	Q_1	Decimal
Initially	0	0	0	0
After 1st -ve edge	0	0	1	1
After 2nd -ve edge	0	1	0	2
After 3rd -ve edge	0	1	1	3
After 4th -ve edge	1	0	0	4
After 5th -ve edge	1	0	1	5

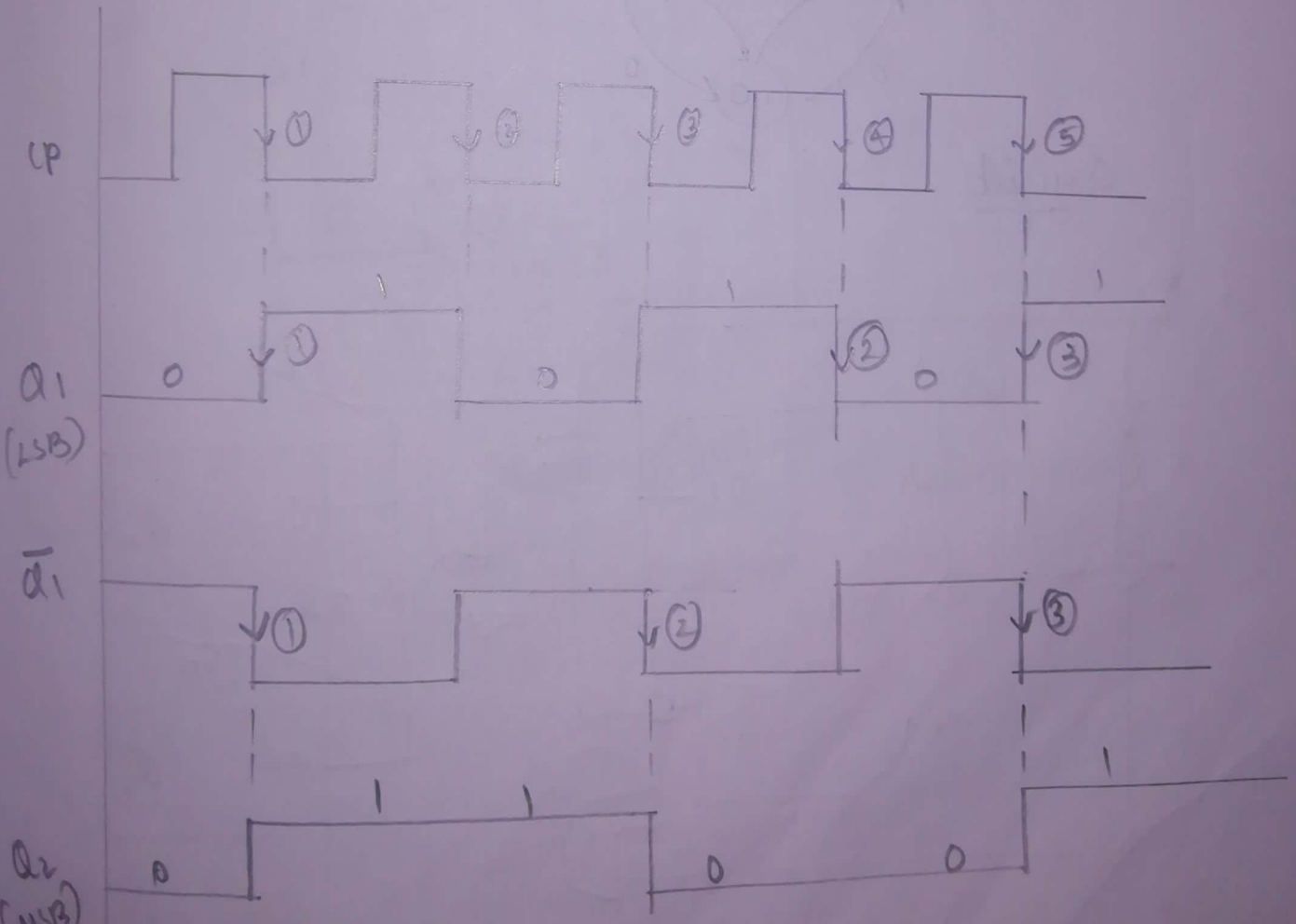
⇒ Design two bit asynchronous down counter

State diagram





Timing Diagram



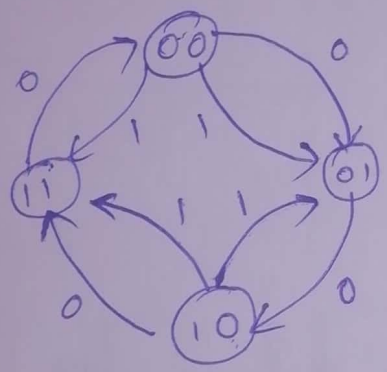
⇒ Design two bit asynchronous up/down counter

⇒ Mod control input

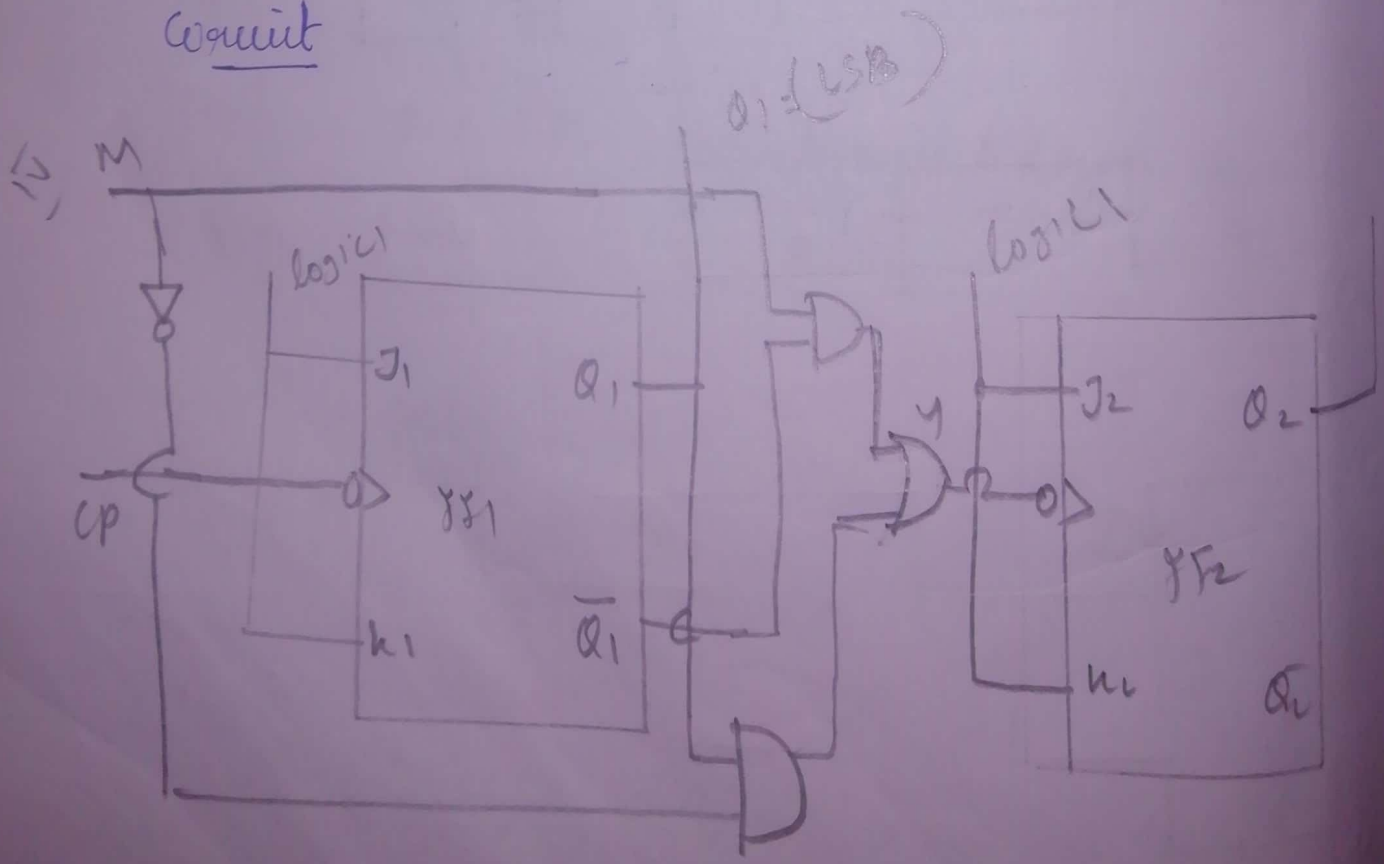
M=0, up counter

M=1, down counter

State diagram



Circuit



II

M	Q_1	$\bar{Q}_1$	clock pulse to next FF (y)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

III

y / M \ $Q_1\bar{Q}_1$	0 0	0 1	1 1	1 0
0	1	0	0	0
1	1	1	0	0

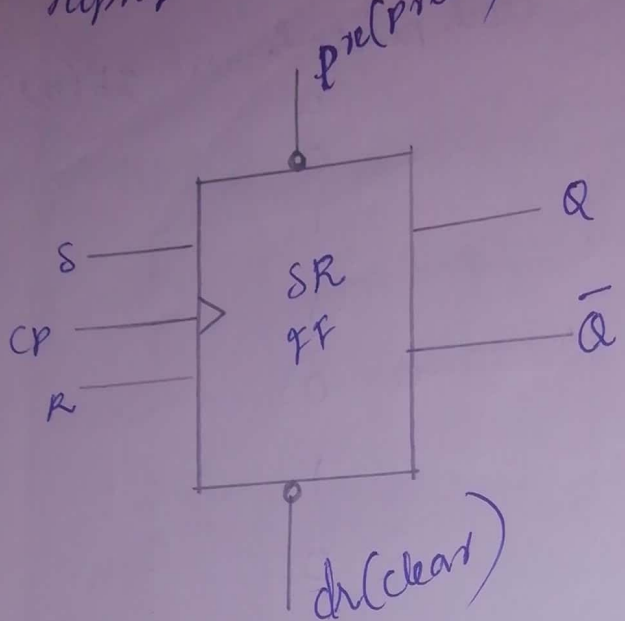
$M\bar{Q}$

$\bar{M}Q$

$y = \bar{M}Q + M\bar{Q}$

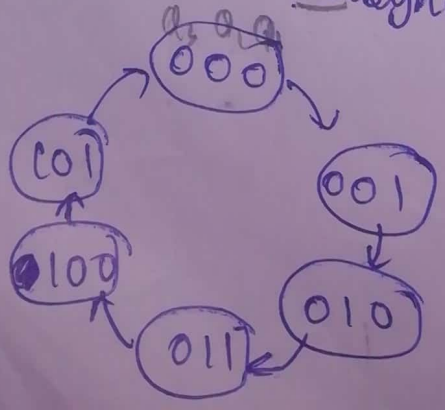
Flip flop with direction of synchronous & Asynchronous

Active low

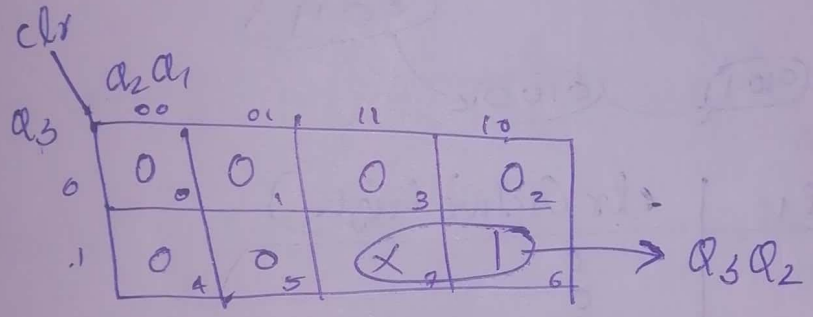


pre	clr	Q
0	0	indeterminate
0	1	1
1	0	0
1	1	Normal operation

⇒ Design a mod 6 asynchronous up counter

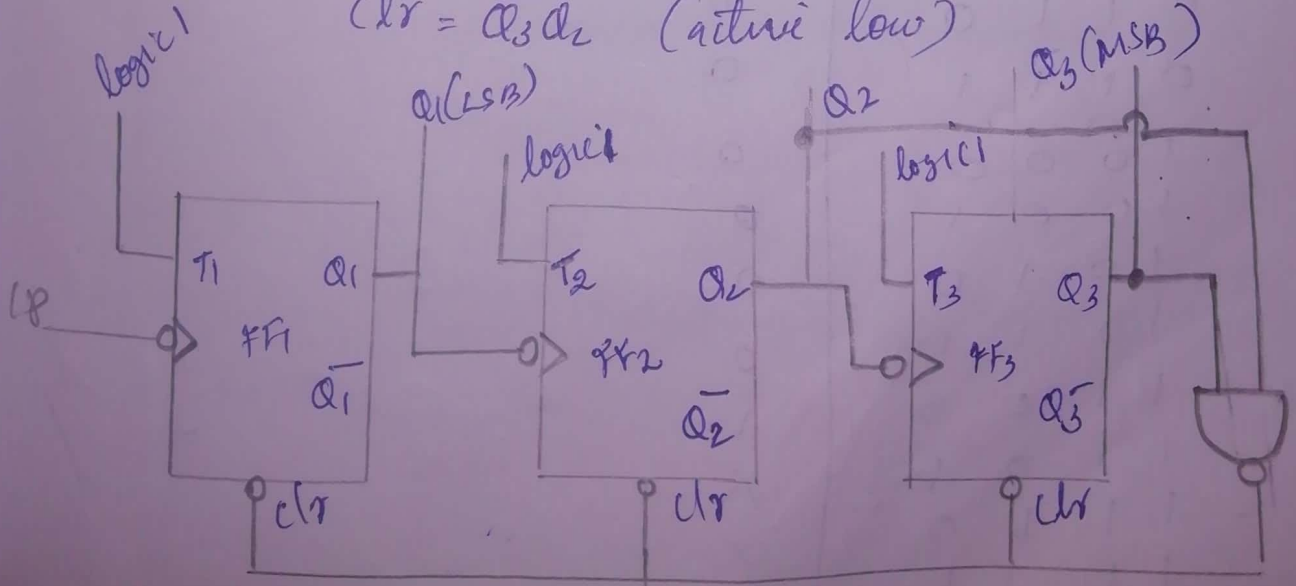


Q_3	Q_2	Q_1	clr (active high)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	X



$clr = Q_3Q_2$ (active high)

$clr = \overline{Q_3Q_2}$ (active low)

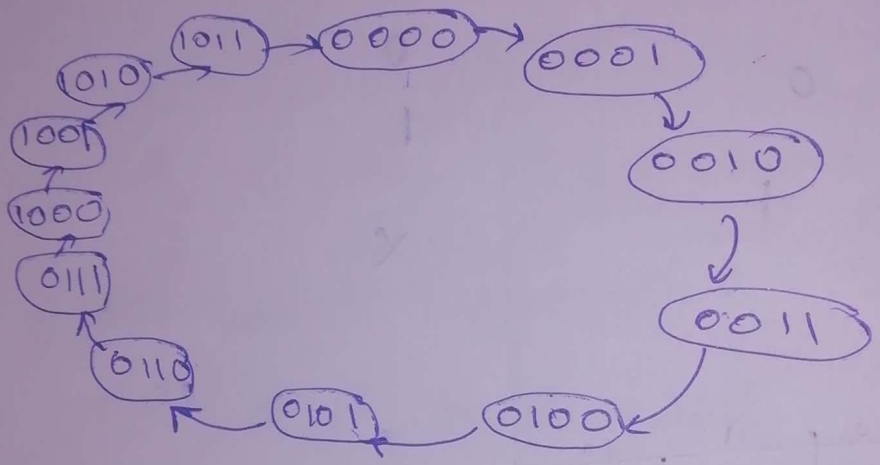


⇒ Design and implement a mod 12 asynchronous up counter using JK flipflop

⇒ Design asynchronous BCD counter (decade counter) → mod 10 asynchronous up or down counter

Answers

⇒



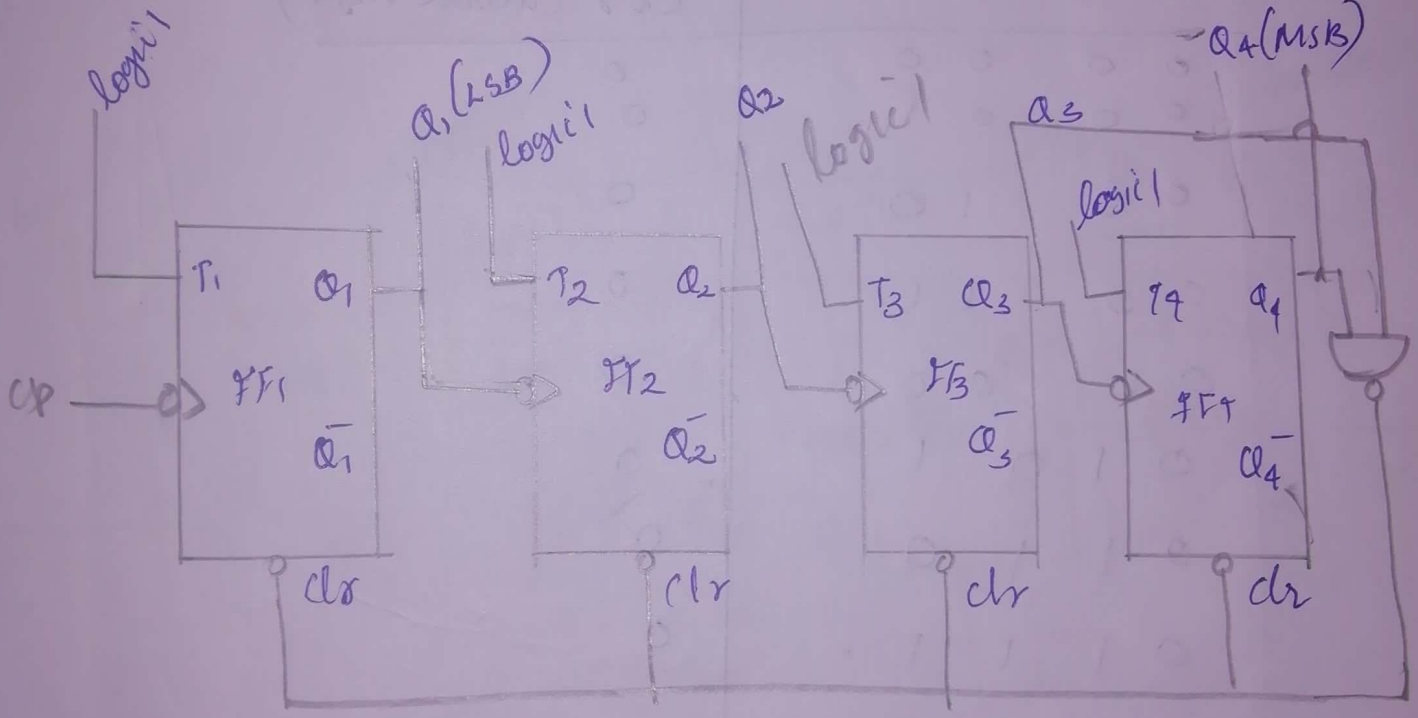
Q_4 Q_3 Q_2 Q_1	clr (Active high)
0 0 0 0	0
0 0 0 1	0
0 0 1 0	0
0 0 1 1	0
0 1 0 0	0
0 1 1 0	0
0 1 1 1	0
1 0 0 0	0
1 0 0 1	0
1 0 1 0	0
1 0 1 1	0
1 1 0 0	1
1 1 0 1	X
1 1 1 0	X
1 1 1 1	X

clr, Q_4Q_3 \ Q_2Q_1	00	01	11	10
00	0 (0)	0 (2)	0 (8)	0 (2)
01	0 (4)	0 (5)	0 (7)	0 (6)
11	1 (12)	X (15)	X (15)	X (14)
10	0 (8)	0 (9)	0 (11)	0 (10)

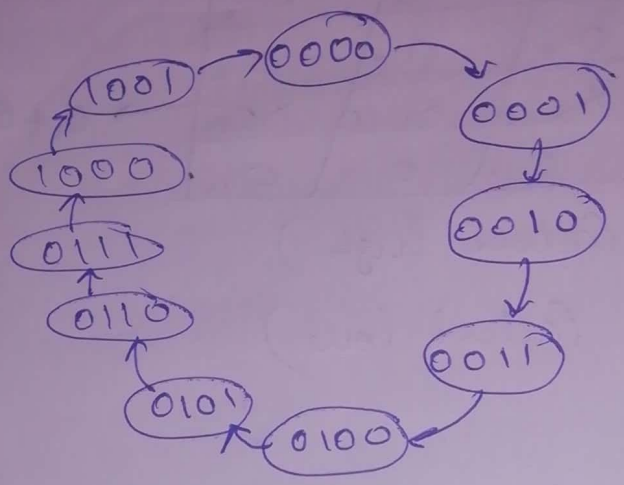
→ Q_4Q_3

clr = $Q_4 Q_3$ (active high)

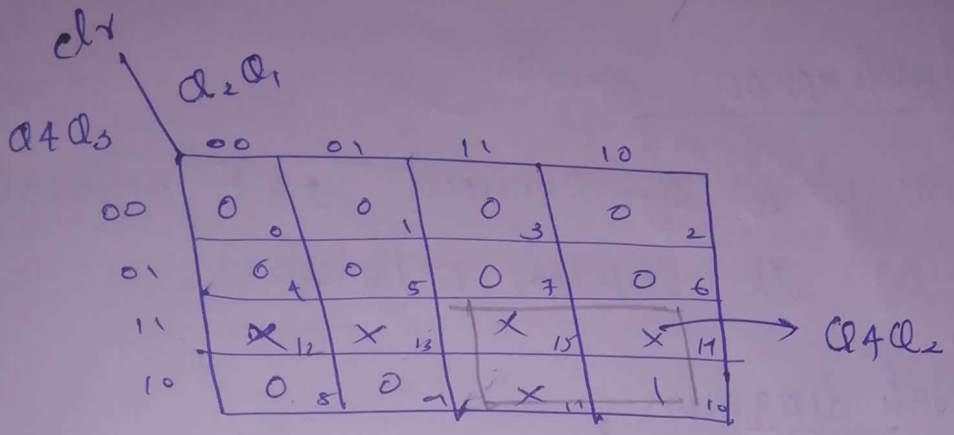
clr = $\overline{Q_4 Q_3}$ (active low)



⇒ Asynchronous BCD counter.



Q_4 Q_3 Q_2 Q_1	clr (active high)
0 0 0 0	0
0 0 0 1	0
0 0 1 0	0
0 0 1 1	0
0 1 0 0	0
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	X
1 1 0 0	X
1 1 0 1	X
1 1 1 0	X
1 1 1 1	X



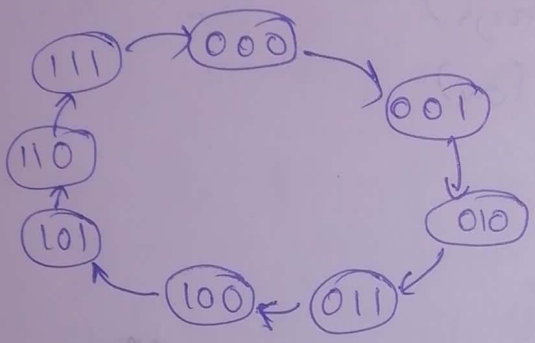
$clr = Q_4Q_2$ (active high)

$clr = \overline{Q_4Q_2}$ (active low)

Synchronous Counter

⇒ Design & implement a Synchronous 3 bit upcounter using JK flipflop, D flipflop.

JK flipflop State diagram.



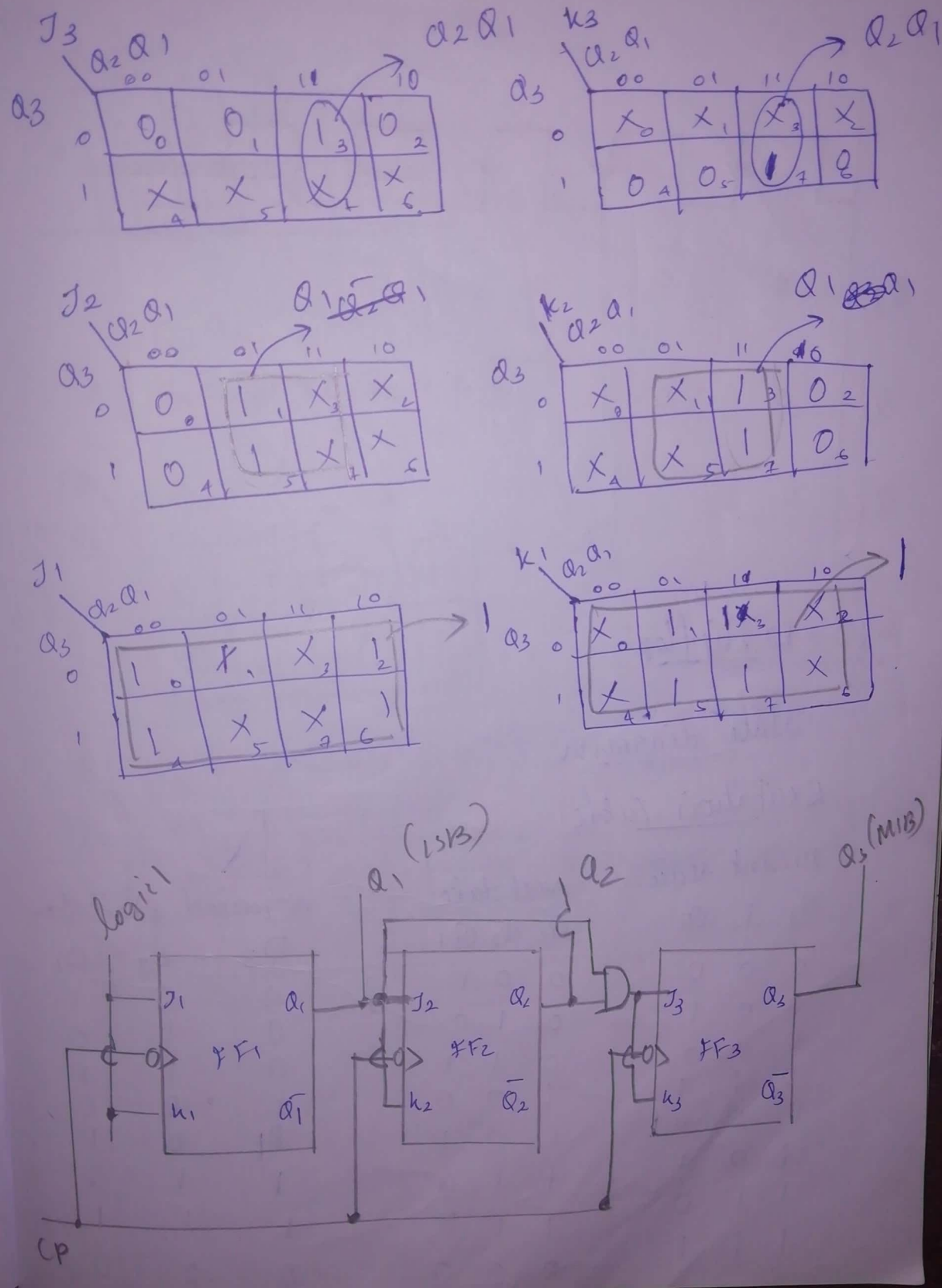
$2^3 = 8$ states

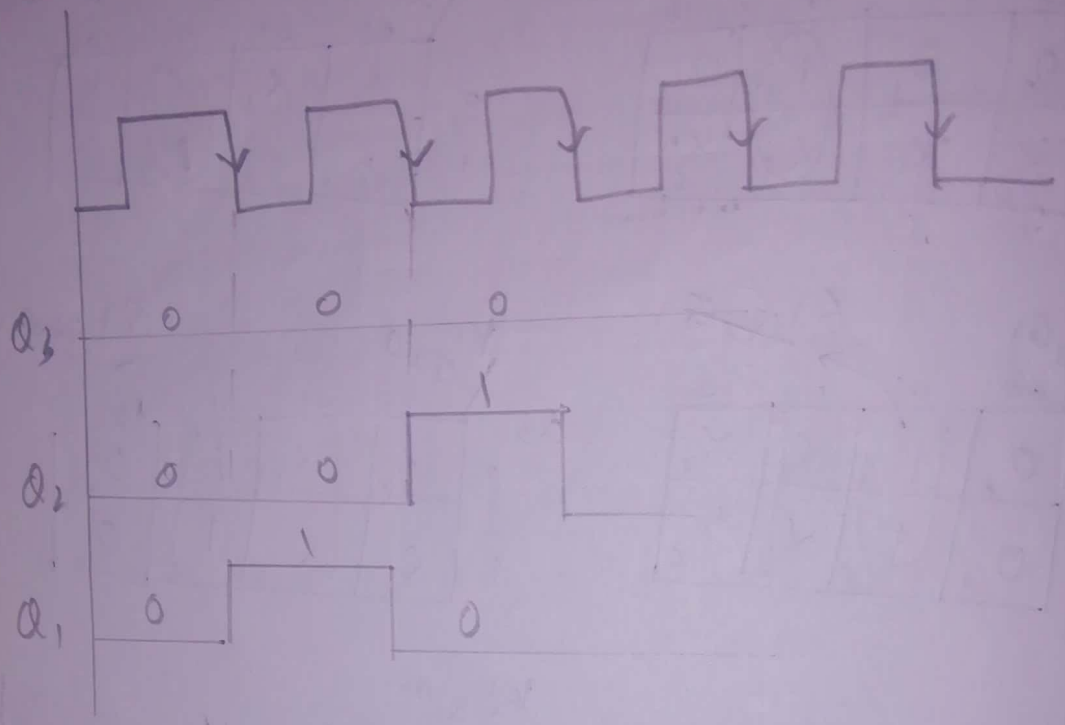
Q(t)	Q(t+1)	J
0	0	0 X
0	1	1 X
1	0	X 1
1	1	X 0

Excitation table for circuit

present state			Next state			Required excitation					
Q_3	Q_2	Q_1	Q_3	Q_2	Q_1	J_3	K_3	J_2	K_2	J_1	K_1
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	X	1
1	1	1	0	0	0	X	1	X	1	X	1

CP





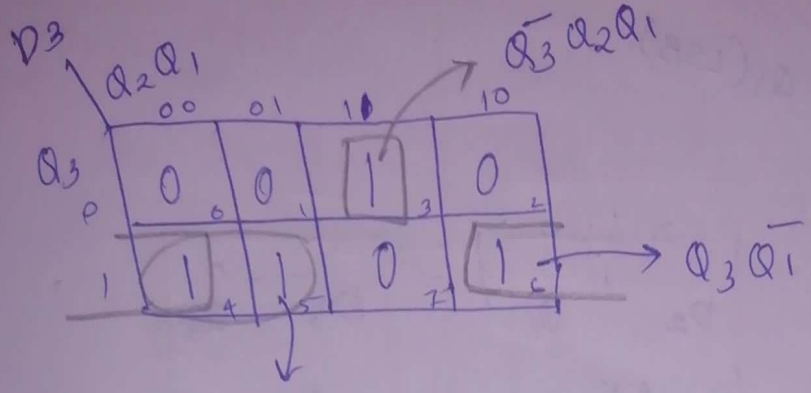
00 | 0
01 | 1
10 | 0
11 | 1

D flipflop

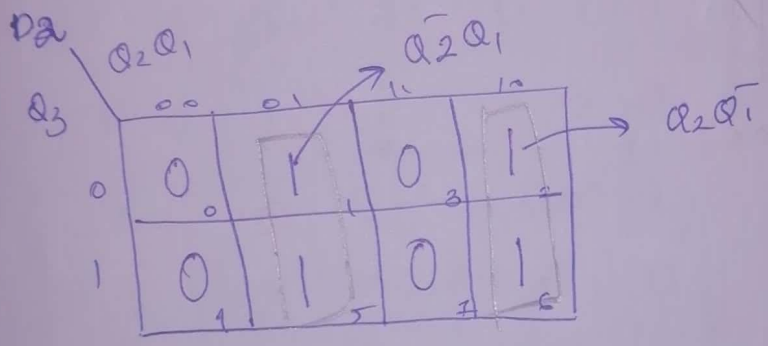
State diagram

Excitation Table

present state			Next state			Required Excitation		
Q_3	Q_2	Q_1	Q_3	Q_2	Q_1	D_3	D_2	D_1
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	0
0	1	0	0	1	1	0	1	1
0	1	1	1	0	0	1	0	0
1	0	0	1	0	1	1	0	1
1	0	1	1	1	0	1	1	0
1	1	0	1	1	1	1	1	1
1	1	1	0	0	0	0	0	0



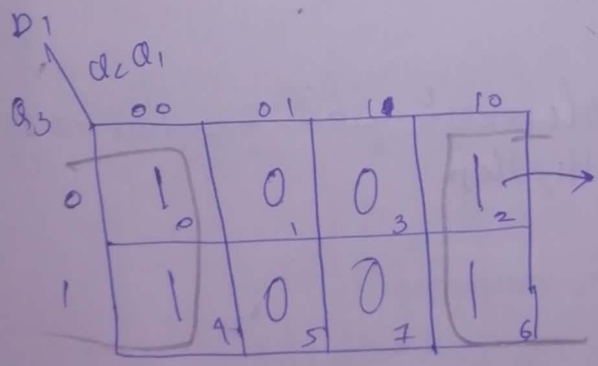
$$D_3 = \bar{Q_3}Q_2Q_1 + Q_3\bar{Q_2} + Q_3\bar{Q_1}$$



$$D_2 = \bar{Q_2}Q_1 + Q_2\bar{Q_1}$$

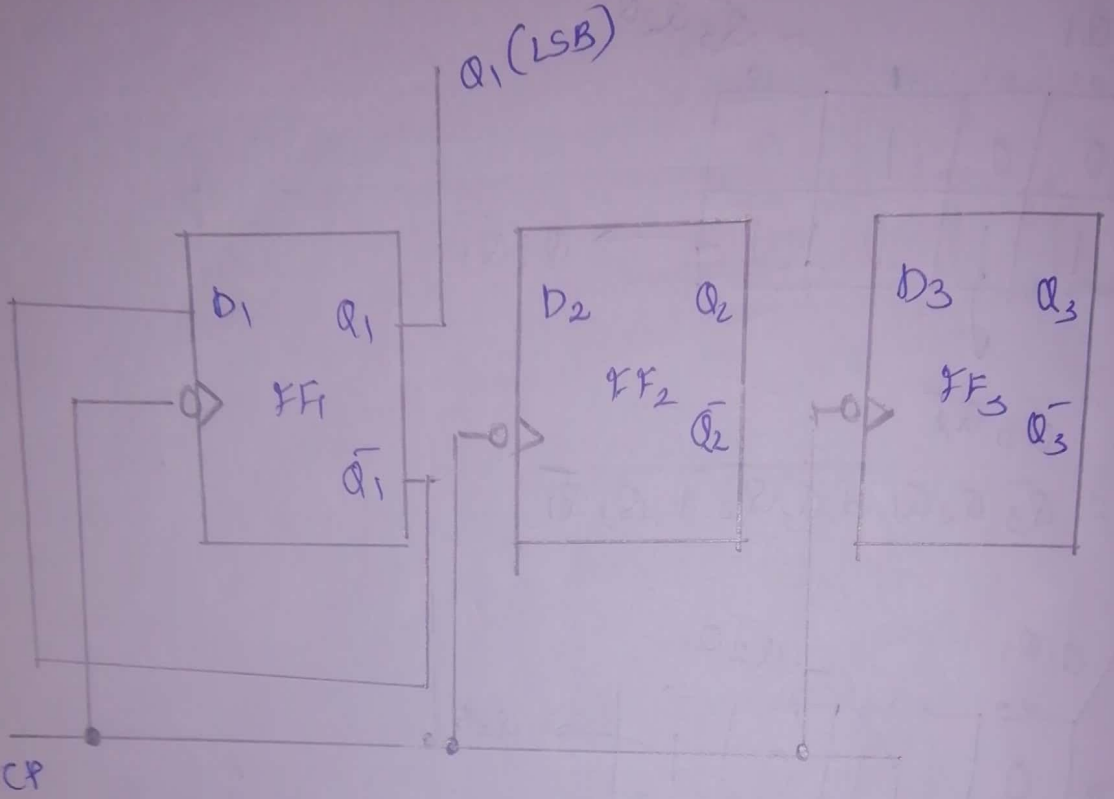
$$= Q_2 \oplus Q_1$$

00	0
01	1
10	0
11	1



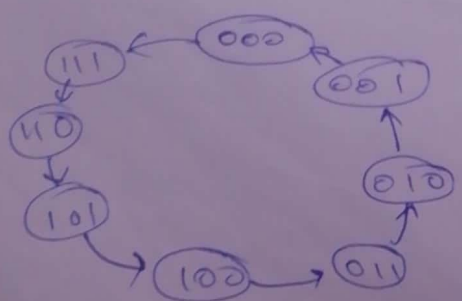
citation

	D_1
	1
	0
	1
	0
	1
	0
	1
	0



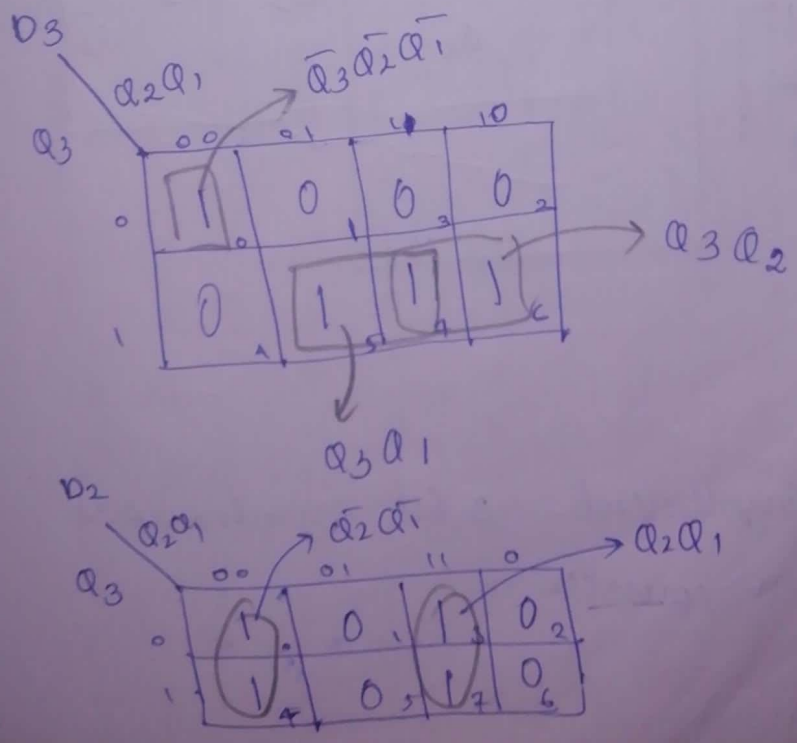
⇒ Design and implement 3bit Synchronous down counter using D-FlipFlop

State diagram



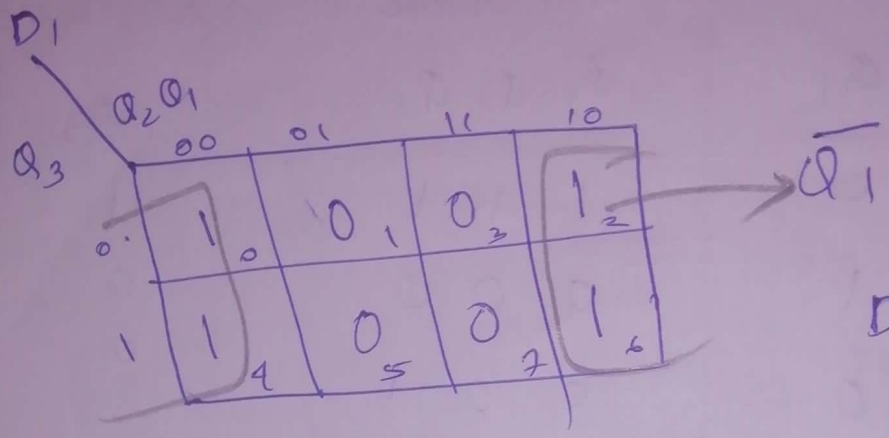
Present state			Next state			Required Excitation		
Q_3	Q_2	Q_1	Q_3	Q_2	Q_1	D_3	D_2	D_1
0	0	0	1	1	1	1	1	1
0	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	1
0	1	1	0	1	0	0	1	0
1	0	0	0	1	1	0	1	1
1	0	1	1	0	0	1	0	0
1	1	0	1	0	1	1	0	1
1	1	1	1	1	0	1	1	0

000
011
100
111

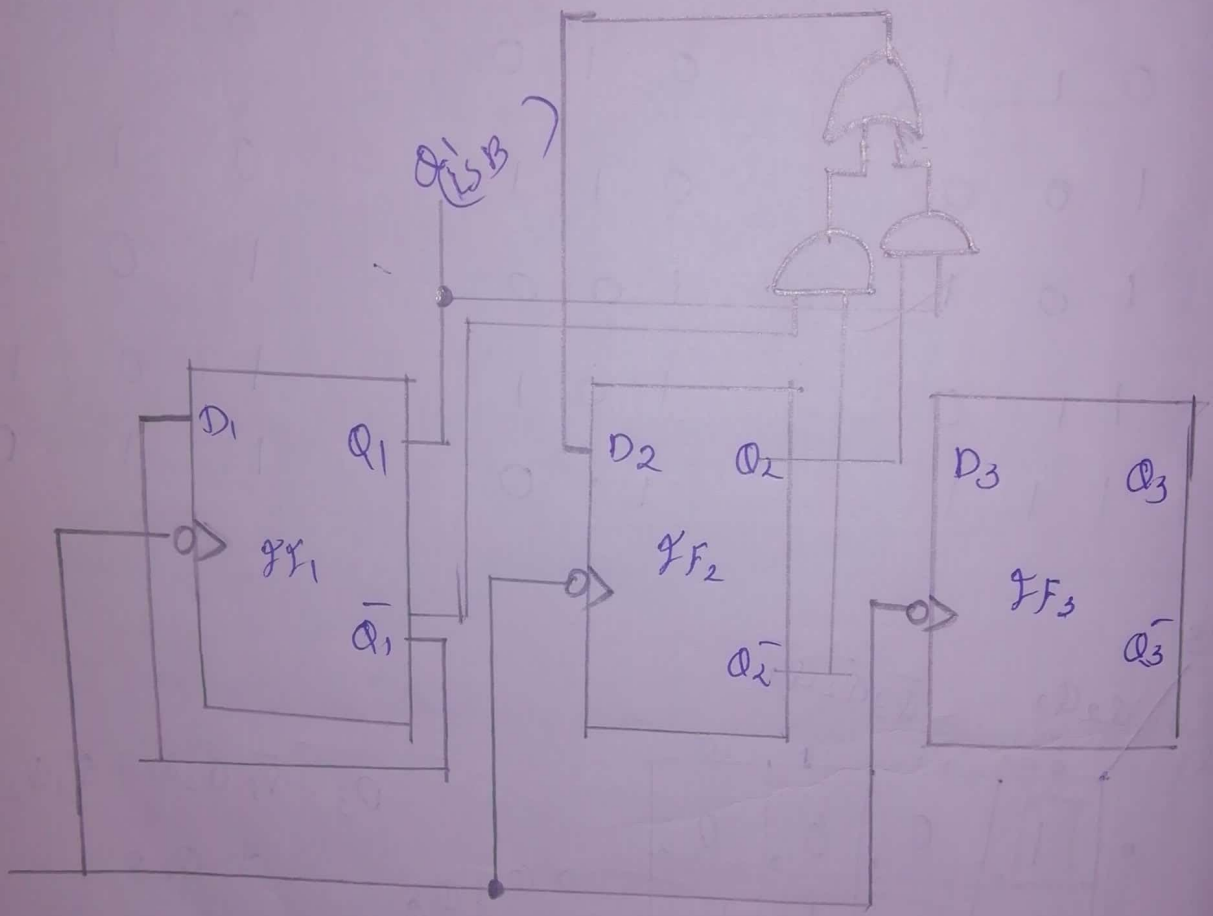


$D_3 = \bar{Q}_3\bar{Q}_2\bar{Q}_1 + Q_3Q_2 + Q_3Q_1$

$D_2 = \bar{Q}_2\bar{Q}_1 + Q_2Q_1$

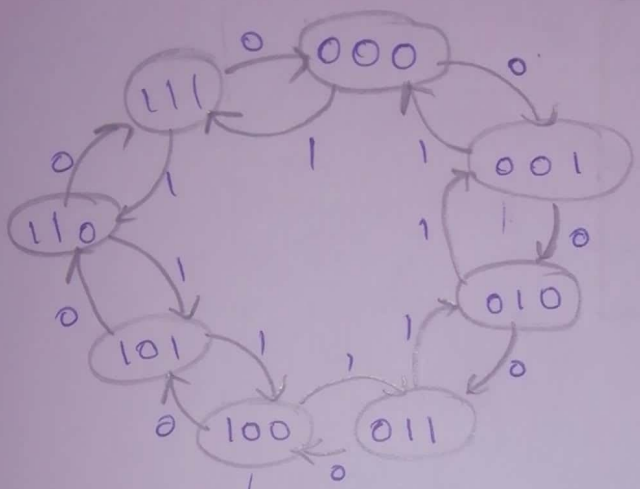


$D_1 = \bar{Q}_1$



⇒ Design and implement 3 bit synchronous up or down counter.

⇒ Suppose Mode control input
M=0, up
M=1, down

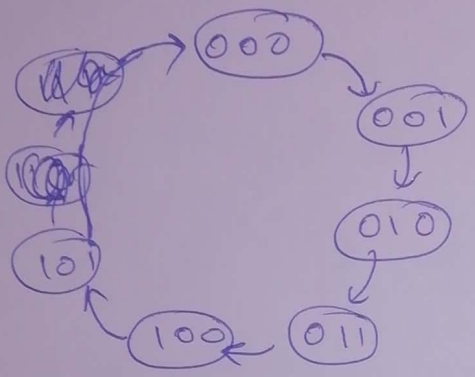


0 0	0
0 1	1
1 0	1
1 1	0

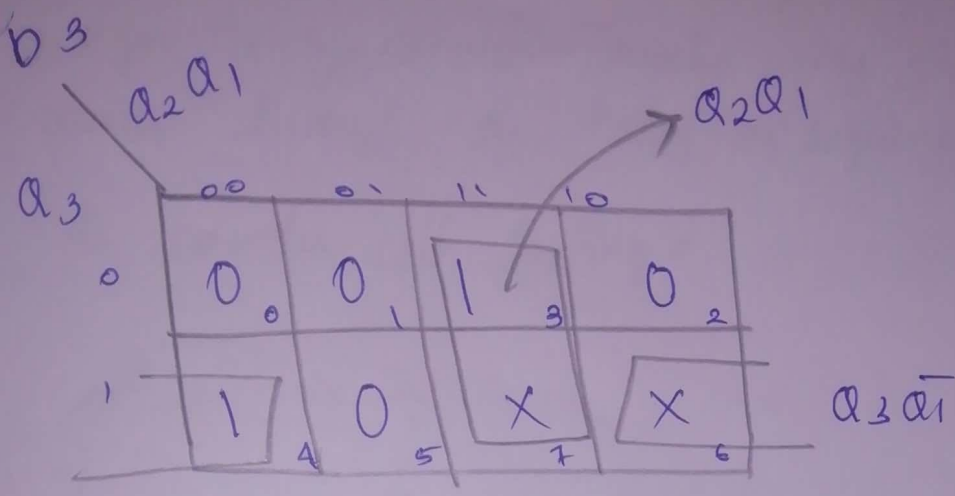
Mode M	present state Q_3 Q_2 Q_1	Next state Q_3 Q_2 Q_1	Required excitations T_3 T_2 T_1
0	0 0 0	0 0 1	0 0 1
0	0 0 1	0 1 0	0 1 1
0	0 1 0	0 1 1	0 0 1
0	0 1 1	1 0 0	1 1 1
0	1 0 0	1 0 1	0 0 1
0	1 0 1	1 1 0	0 1 1
0	1 1 0	1 1 1	0 0 1
0	1 1 1	0 0 0	1 1 1
1	0 0 0	1 1 1	1 1 1
1	0 0 1	0 0 0	0 0 1
1	0 1 0	0 0 1	0 1 1
1	0 1 1	0 1 0	0 0 1
1	1 0 0	0 1 1	1 1 1
1	1 0 1	1 0 0	0 0 1
1	1 1 0	1 0 1	0 1 1
1	1 1 1	1 1 0	0 0 1

⇒ Design and implement mod 6 synchronous up counter.

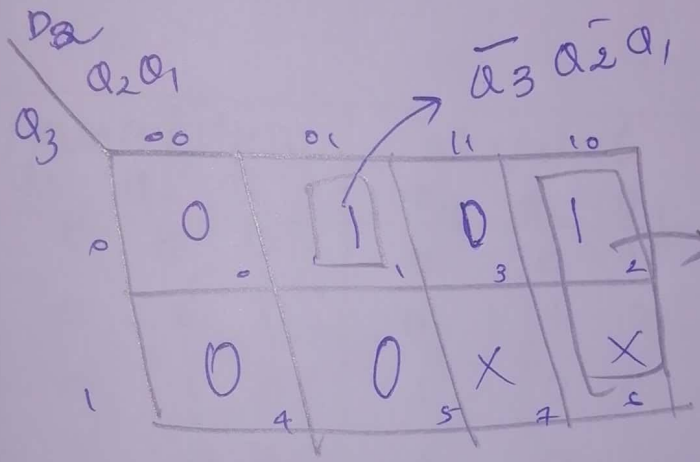
State diagram



Present state			Next state			Required excitation		
Q_3	Q_2	Q_1	Q_3	Q_2	Q_1	D_3	D_2	D_1
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	0
0	1	0	0	1	1	0	1	1
0	1	1	1	0	0	1	0	0
1	0	0	1	0	1	1	0	1
1	0	1	0	0	0	0	0	0
1	1	0				x	x	x
1	1	1				x	x	x



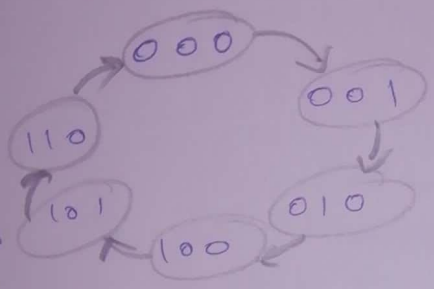
$D_3 = Q_2Q_1 + Q_3\bar{Q}_1$



$D_2 = \bar{Q}_3\bar{Q}_2Q_1 + Q_2\bar{Q}_1$

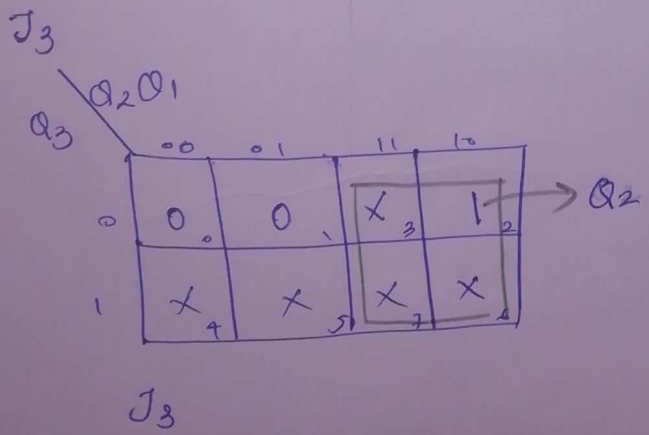
D1
1
0
1
0
1
0
x
x

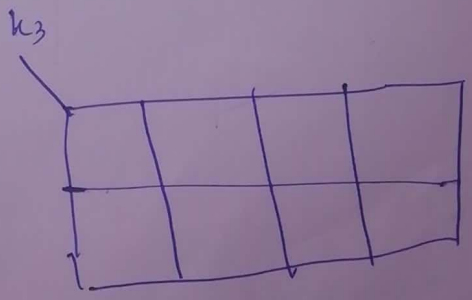
=) Design a synchronous counter using JK flipflop which counts through the states or sequence 0, 1, 2, 4, 5, 6, 0, ... & counter self starting ?



0 0	0 x
0 1	1 x
1 0	x 1
1 1	x 0

Present state			Next state			Required Excitation					
Q_3	Q_2	Q_1	Q_3	Q_2	Q_1	J_3	K_3	J_2	K_2	J_1	K_1
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	1	0	0	1	x	x	1	0	x
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	0	0	0	x	1	x	1	0	x





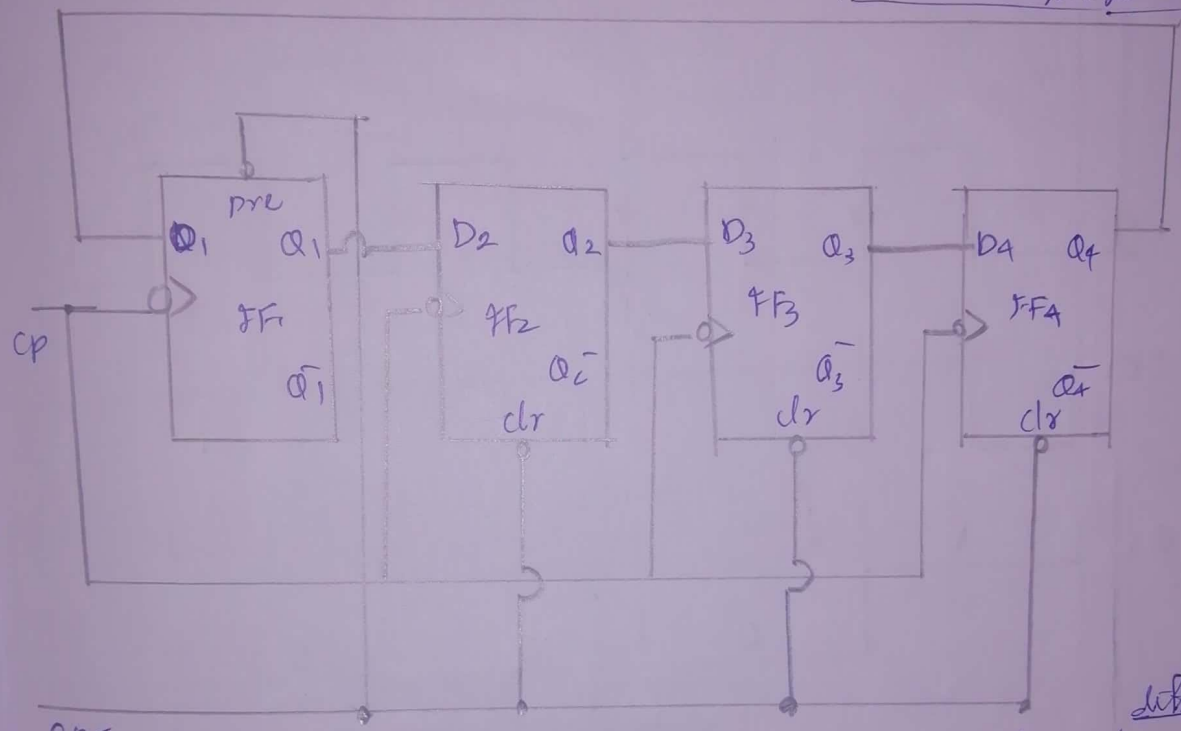
J_3

Scanned by CamScanner

Ring counter ⟹ D flipflop (must) [~~4bit~~]

Q. 4 bit ring counter

No: of states = No: of flipflops used



ORI

Cover ~~Loading~~ ori inputs

ORI	CP	Q_1	Q_2	Q_3	Q_4
0	x	1	0	0	0
1	↓	0	1	0	0
1	↓	0	0	1	0
1	↓	0	0	0	1
1	↓	1	0	0	0

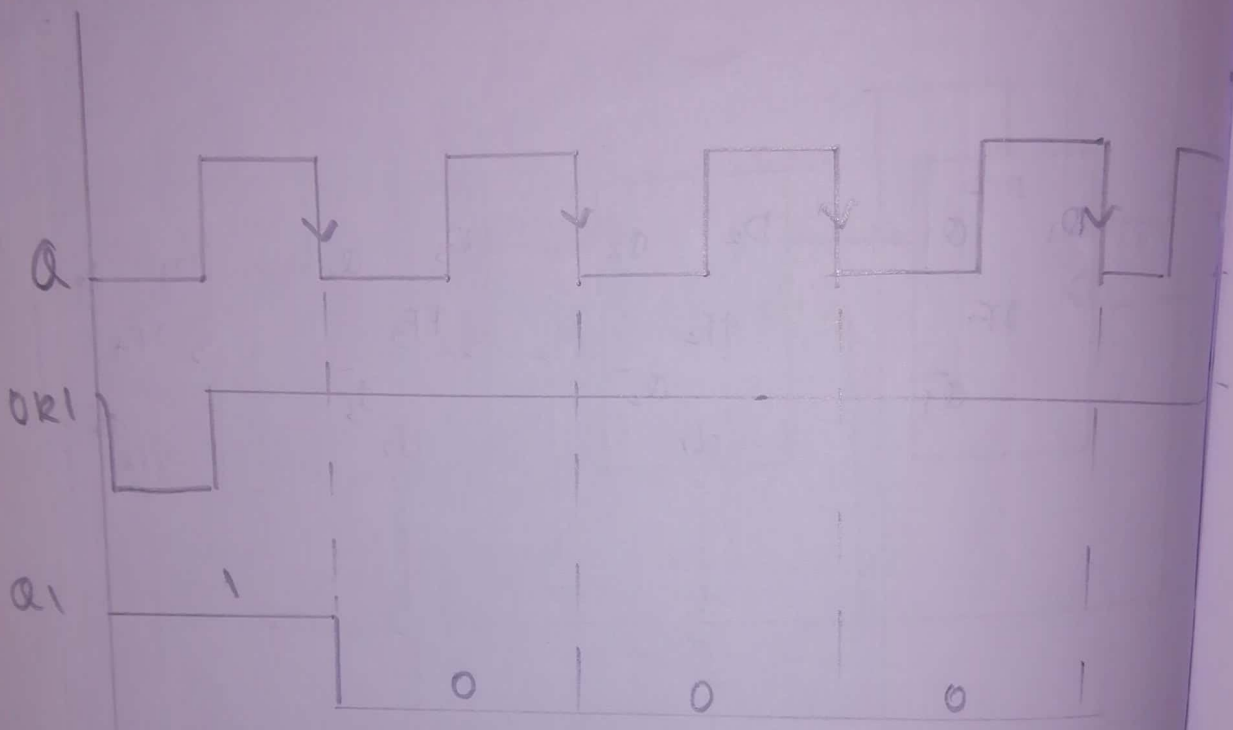
diff

Johnson' counter

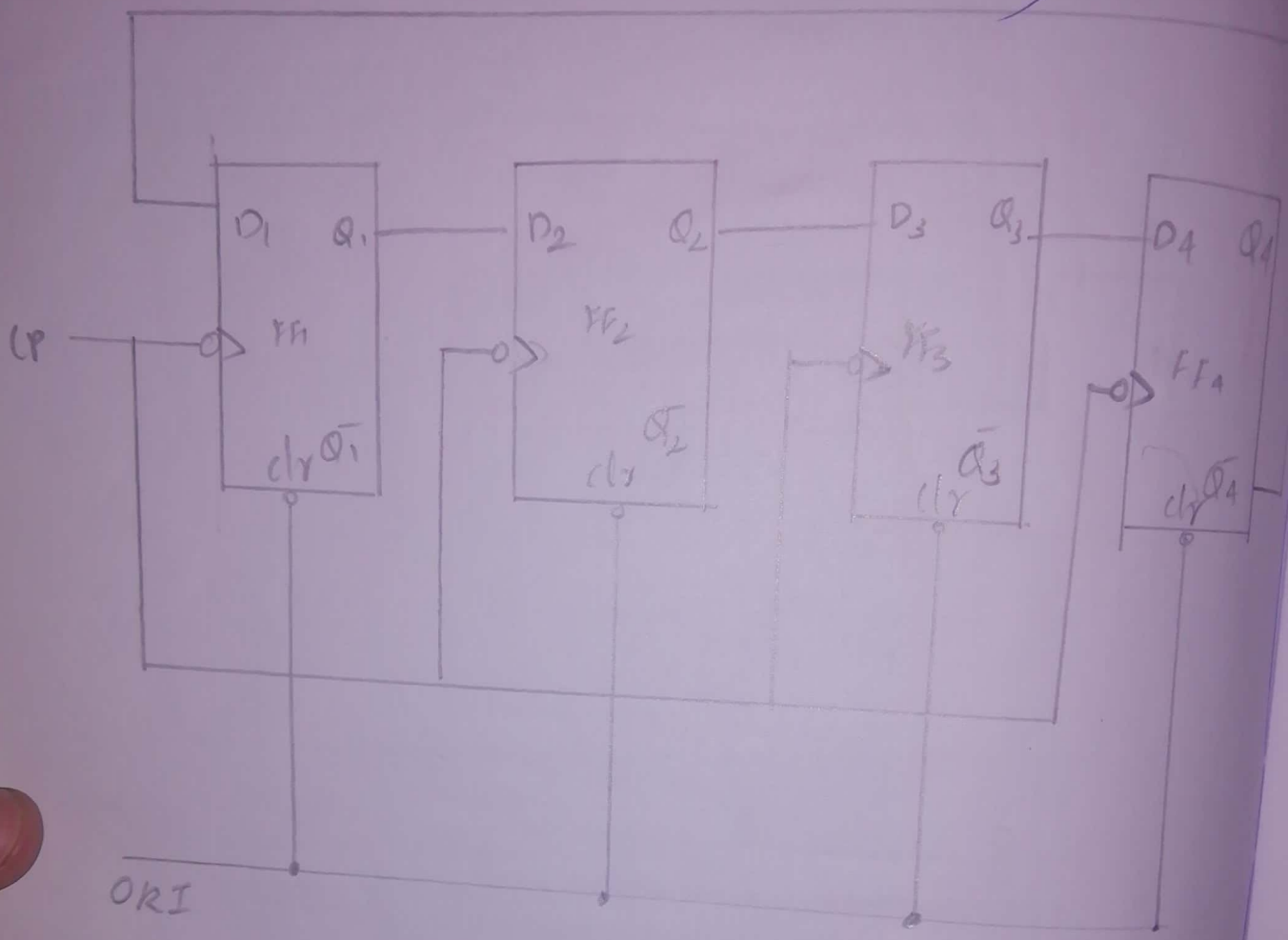
1) $\bar{Q}$ → compd o/p is given as D i/p

2) No: of states = 2 × no: of ~~flipflop~~

3) ORI to all clear.



Johnson's Ring counter (Twisted / switch Tail Ring counter)

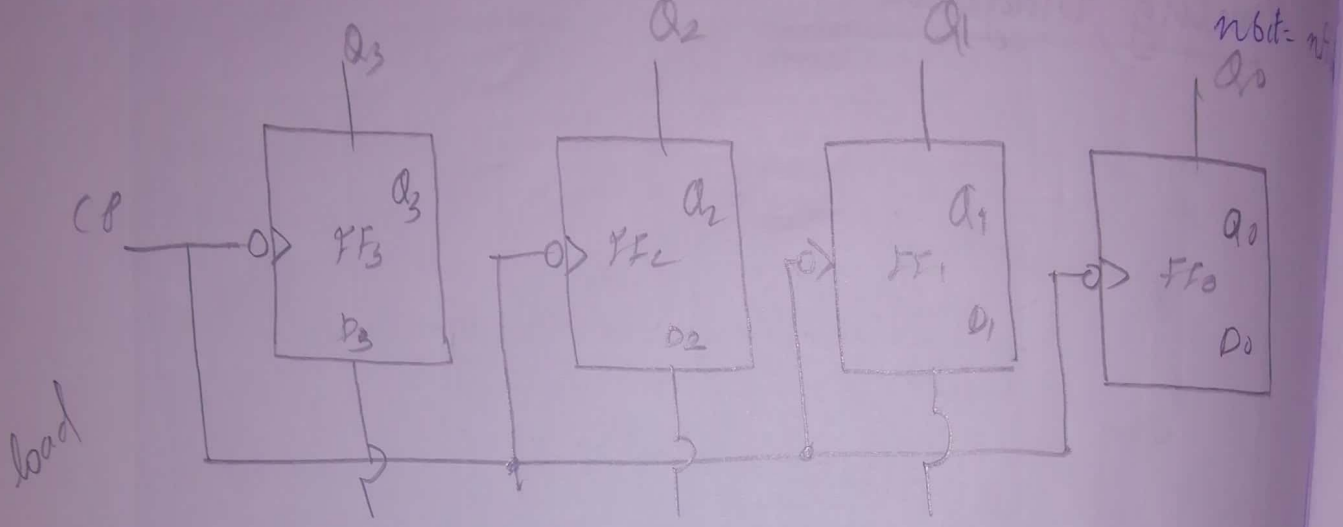


ORI	CP	Q_1	Q_2	Q_3	Q_4
0	x	0	0	0	0
1	↓	1	0	0	0
1	↓	1	1	0	0
1	↓	1	1	1	0
1	↓	1	1	1	1
1	↓	0	1	1	1
1	↓	0	0	1	1
1	↓	0	0	0	1

8 distinct states

2×4 = 8

Registers

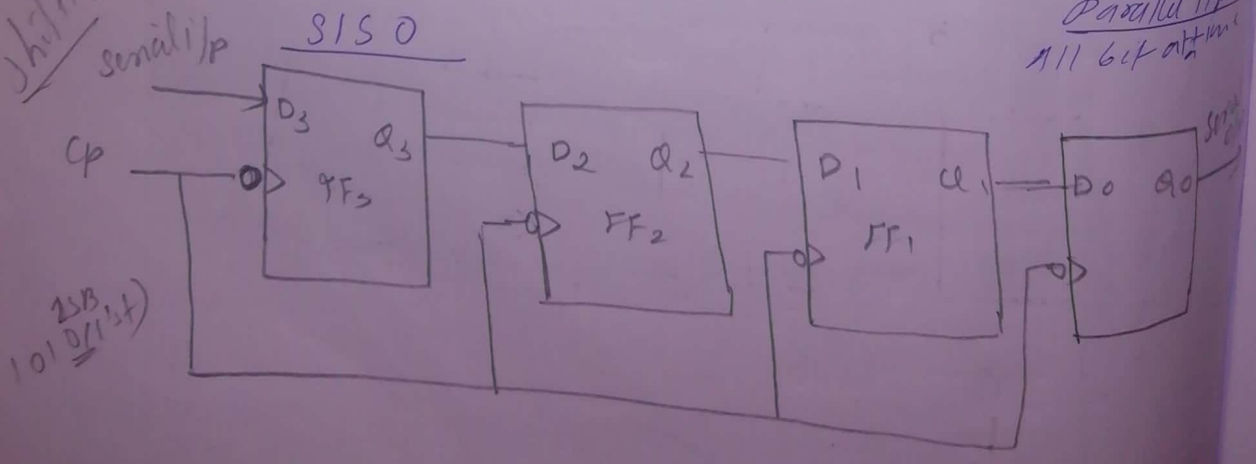


-ve edge **Register** → group of flipflop + gate combination

1) i/p and o/p → SISO — serial i/p serial o/p
→ SIPO — serial i/p parallel o/p
→ PISO
→ PIPO

2) Application → shift register
→ storage register

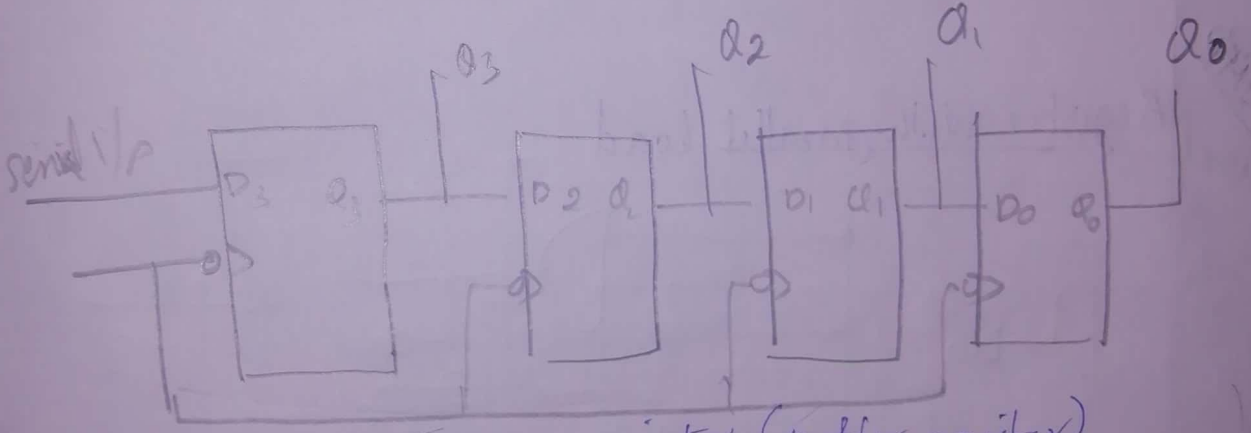
Shifting



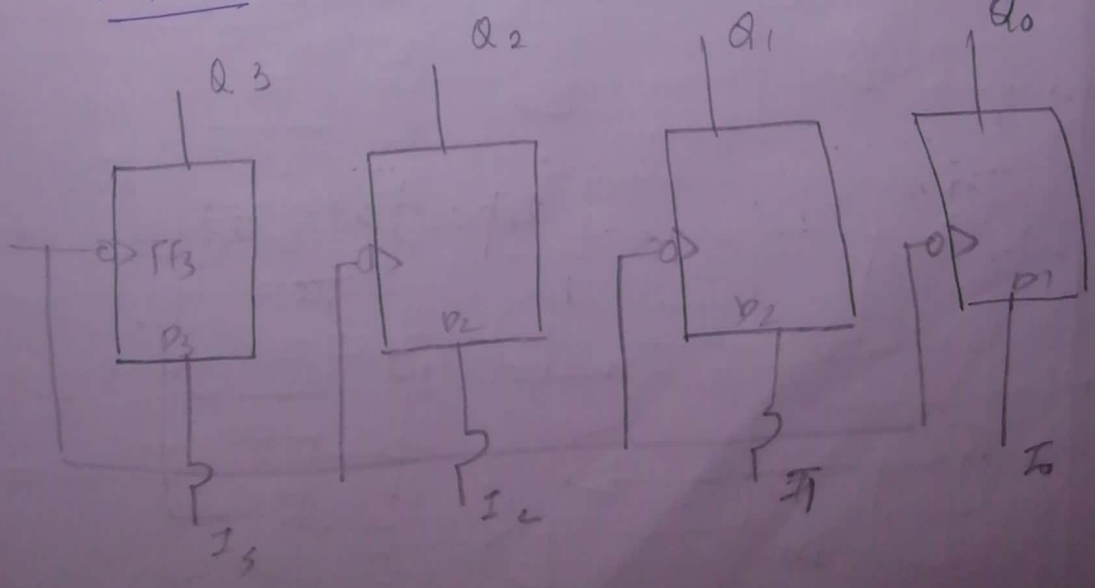
Serial i/p
one bit at a time
Parallel i/p
All bit at a time

CP	Q_3	Q_2	Q_1	Q_0
Initially	0	0	0	0
↓	0	0	0	0
↓	1	0	0	0
↓	0	1	0	0
↓	0	0	1	0

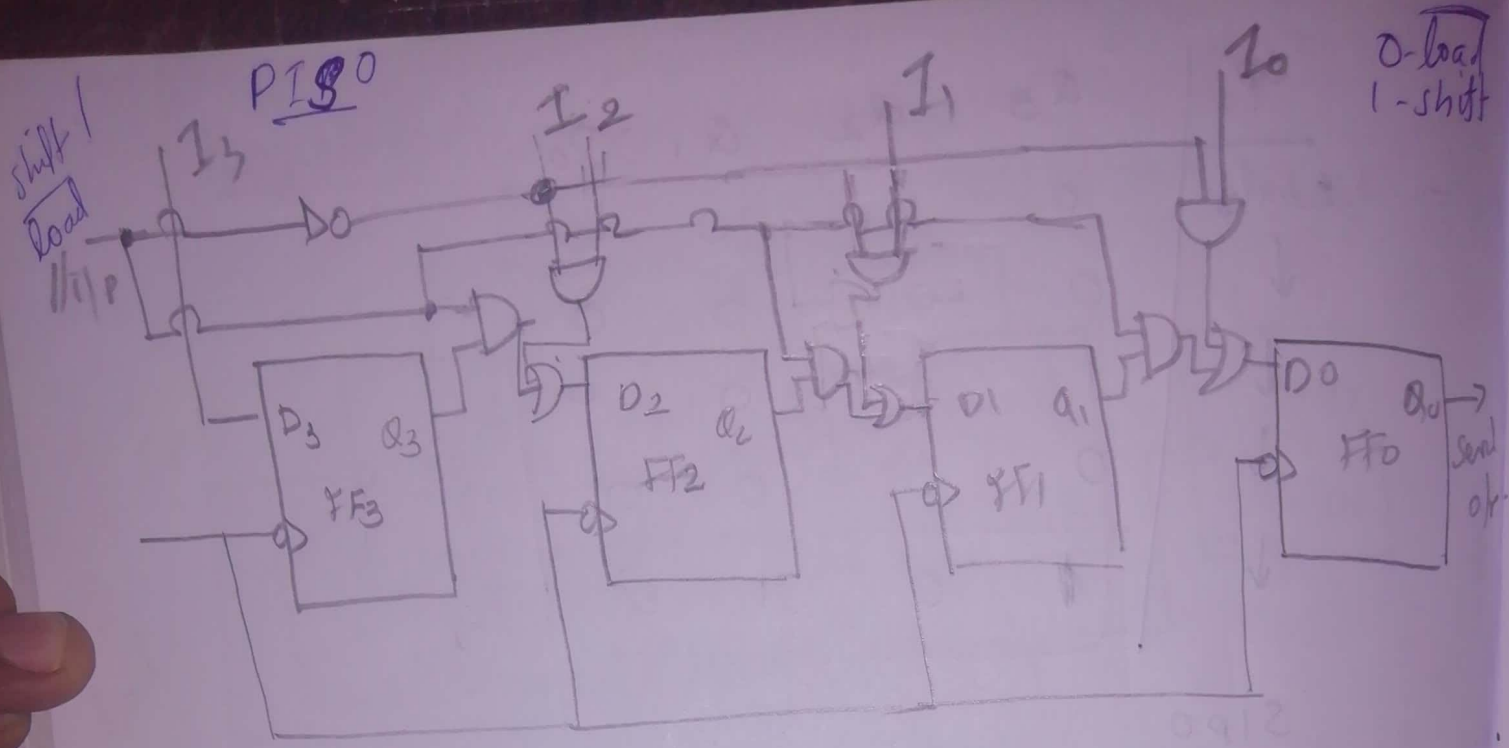
SIPO



PIPO – storage register (Buffer register)

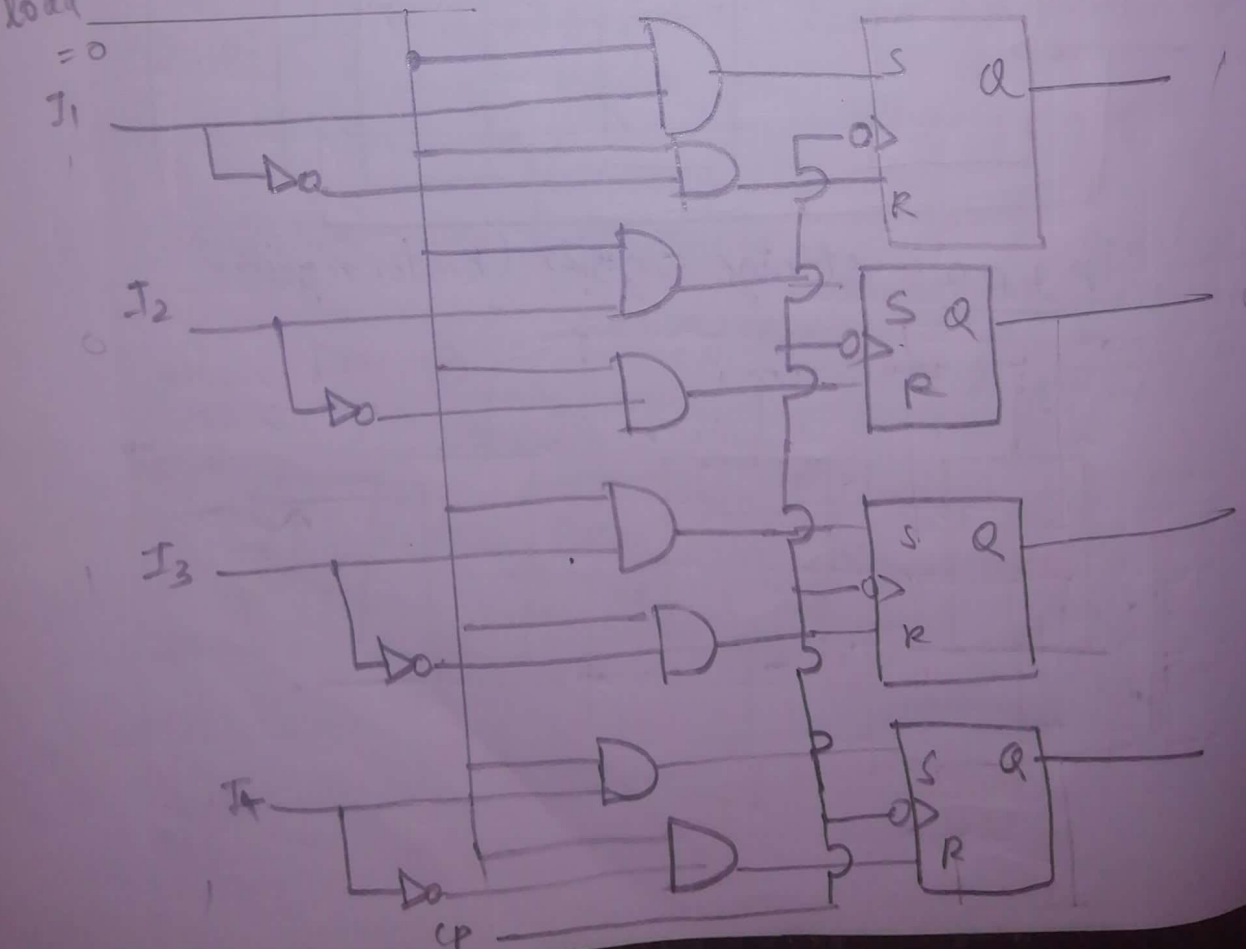


PISO



8MP

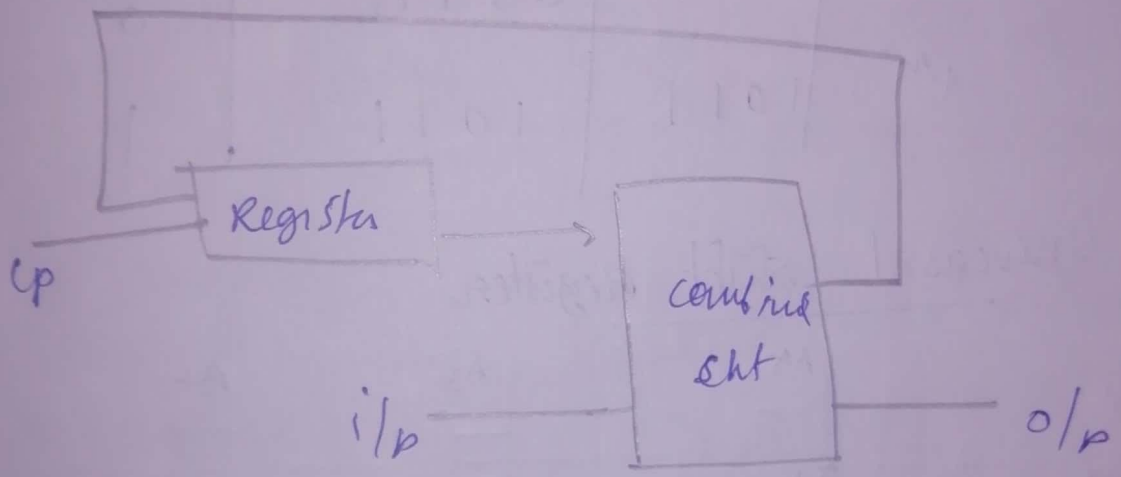
Register with parallel load



if I=1, S=1, R=0

I=0, S=0, R=1

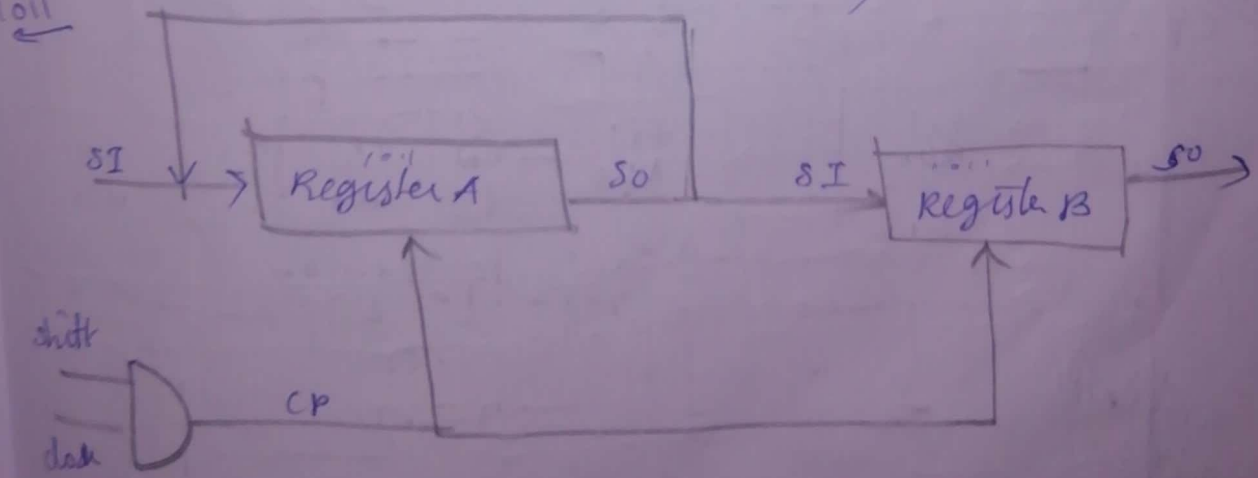
Sequential Logic Implementation



IMP

1) Serial Transfer → Application of shift register

2) Serial Adder

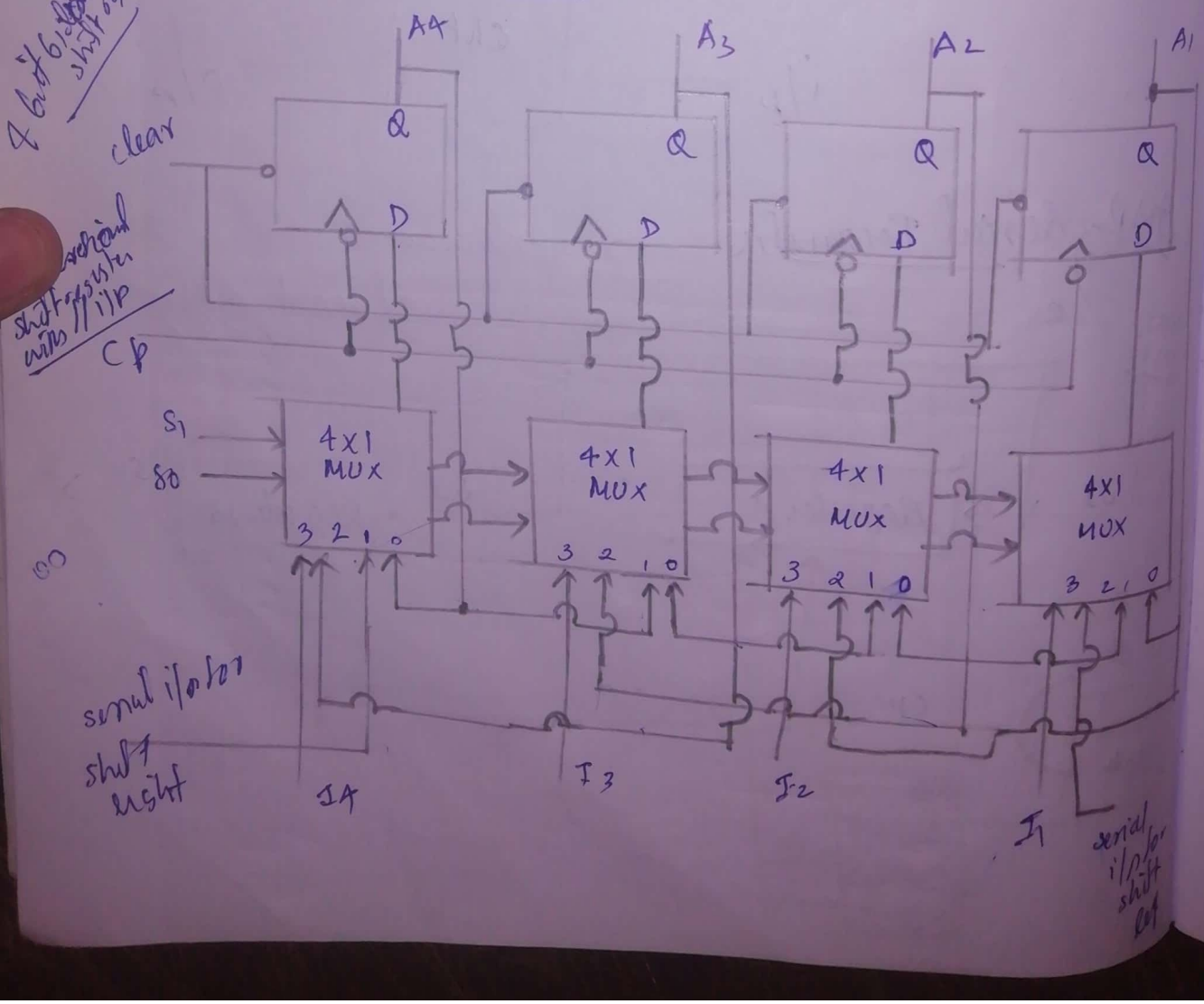


	Register A	Register B	SO Reg B
Initially	1011	0011	1
1	1101	1001	1
2	1110	1100	0
3rd	0111	0110	0
4th	1011	1011	1

V. IMP

4 bit 6 [illegible] shift register

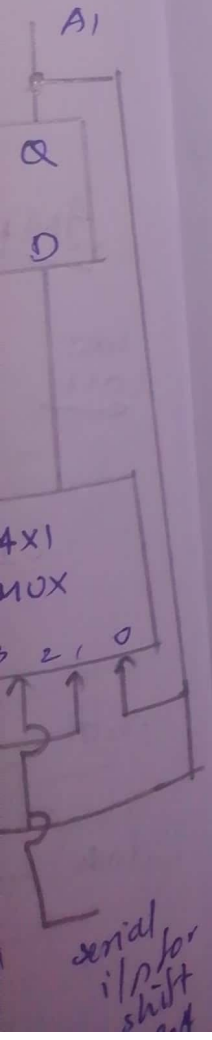
Universal shift Register



⇒ Shift registers can be used for converting serial data to parallel data and vice versa

⇒ The most general shift register has all the capabilities listed below

1. A clear control to clear the register to zero
2. A cp input for clock pulses to synchronise all operations
3. A shift-right control to enable the shift right operation and the serial input and output lines associated with the shift right.
4. A shift left control to enable the shift left operation and the serial i/p and o/p lines associated with the shift ~~ri~~ left.
5. A parallel load control to enable a parallel transfer and the 'n' i/p lines associated with the parallel transfer.
6. 'n' parallel output lines
7. A control state that leads the information in the register, unchanged even though clock pulses are continuously applied.



Function Table

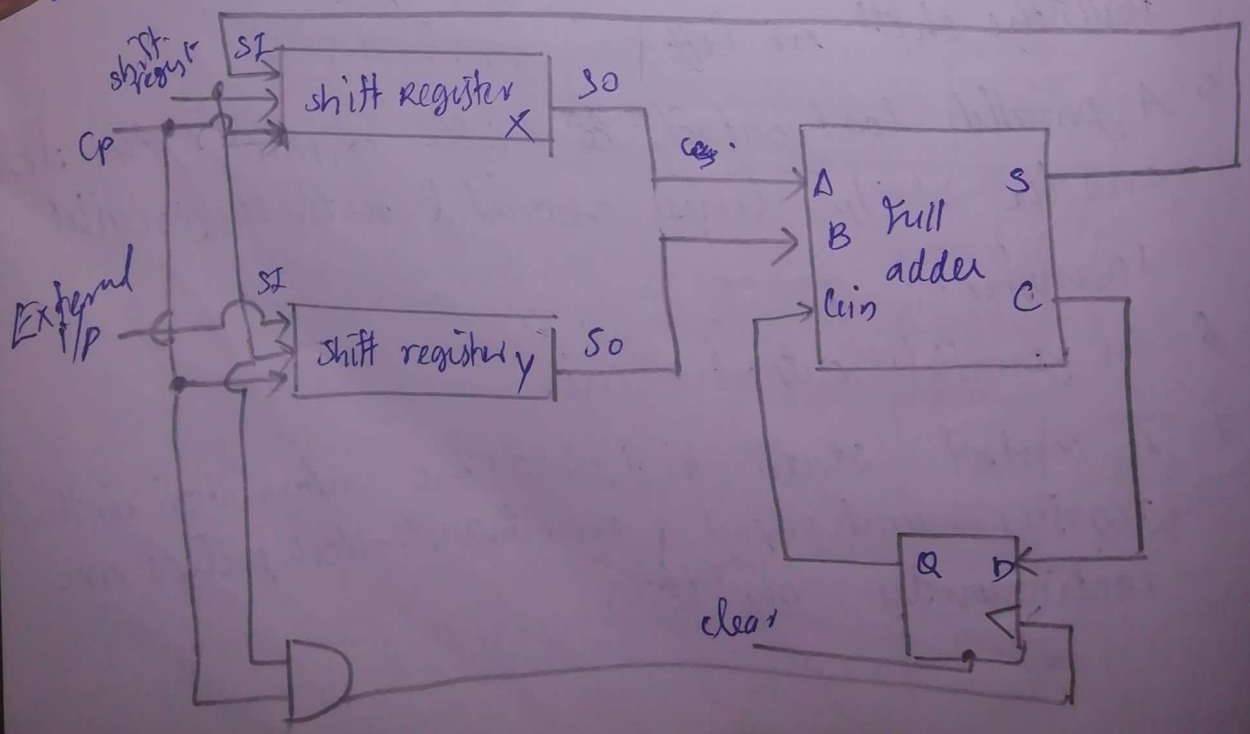
Mode control		Register operation
S_1	S_0	
0	0	Storage
0	1	shift right
1	0	shift left
1	1	parallel load

Application of shift register

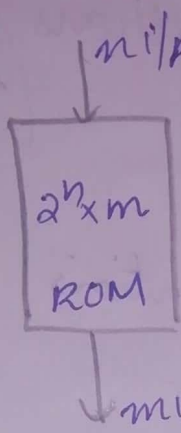
Serial Addition [value of register x & y, 1 bit adding at a time]

IMP

Serial Adder



ROM

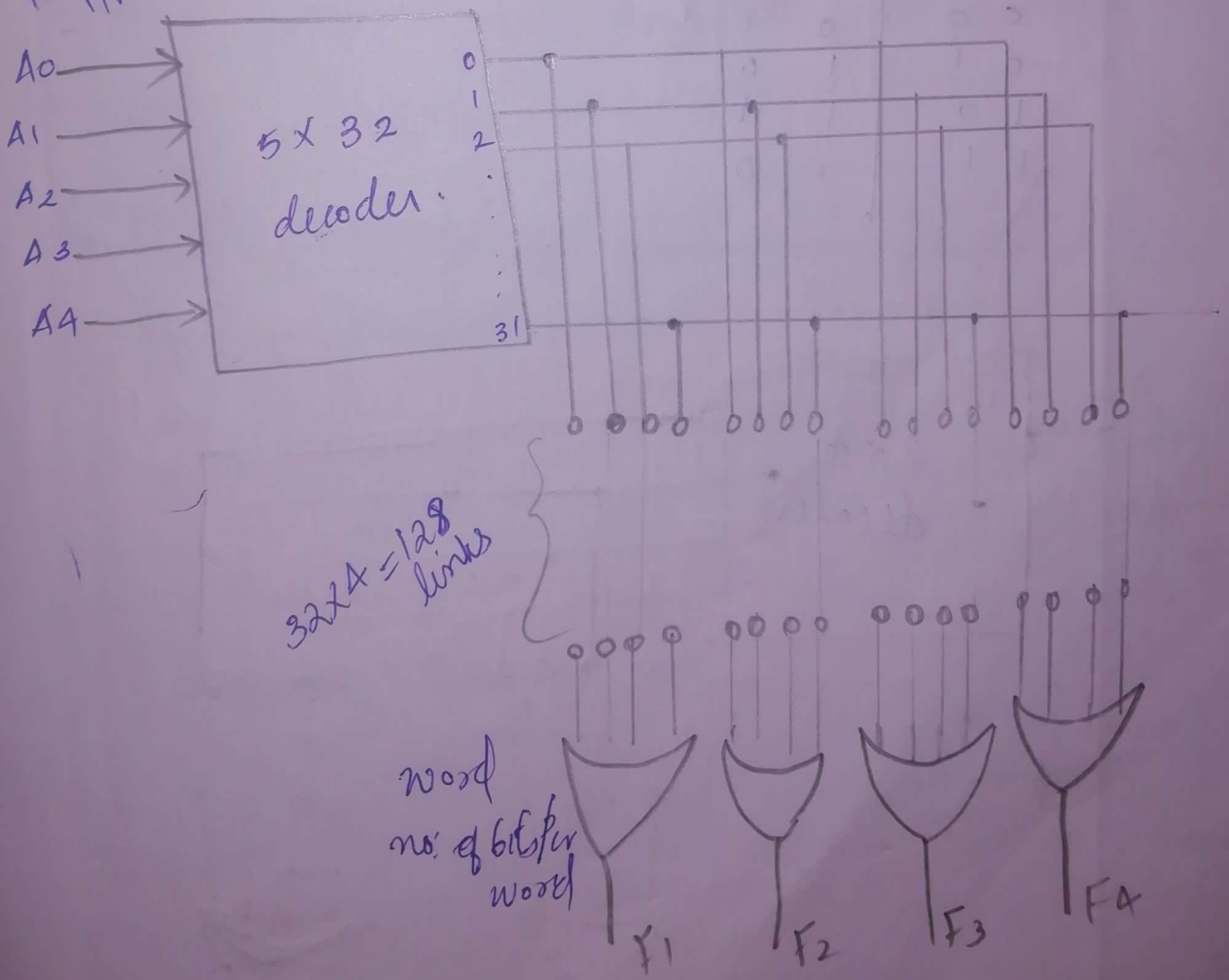


→ memory address (word)
(cross word + bit)

32 × 4 ROM

No: of bit = OR gates

m → how many bit in a word



$n \Rightarrow$ i/p $= 2 \Rightarrow 2^n$ o/p min

⇒ Implement Boolean functions $F_1(A_1A_0) = \Sigma(1,2,3)$

$F_2(A_1A_0) = \Sigma(0,2)$

m

using ROM

i/p min no: of functions

2 × 4 2

decoder 4×2 ROM

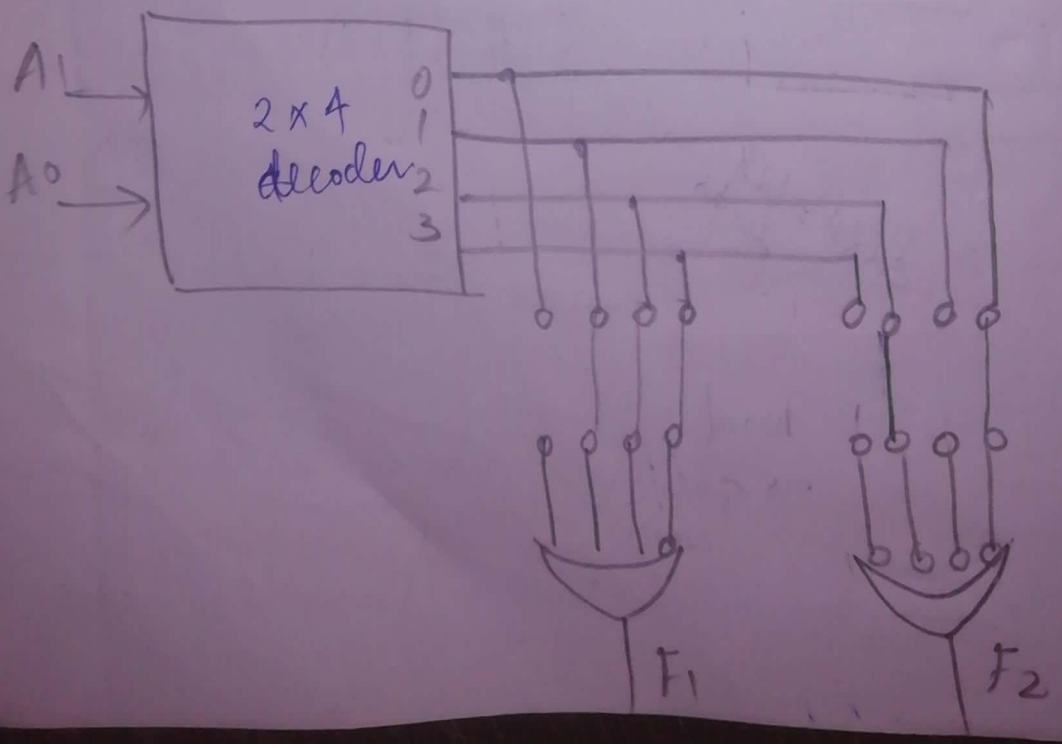
$2^{n} \times m$

32×4

5 i/p

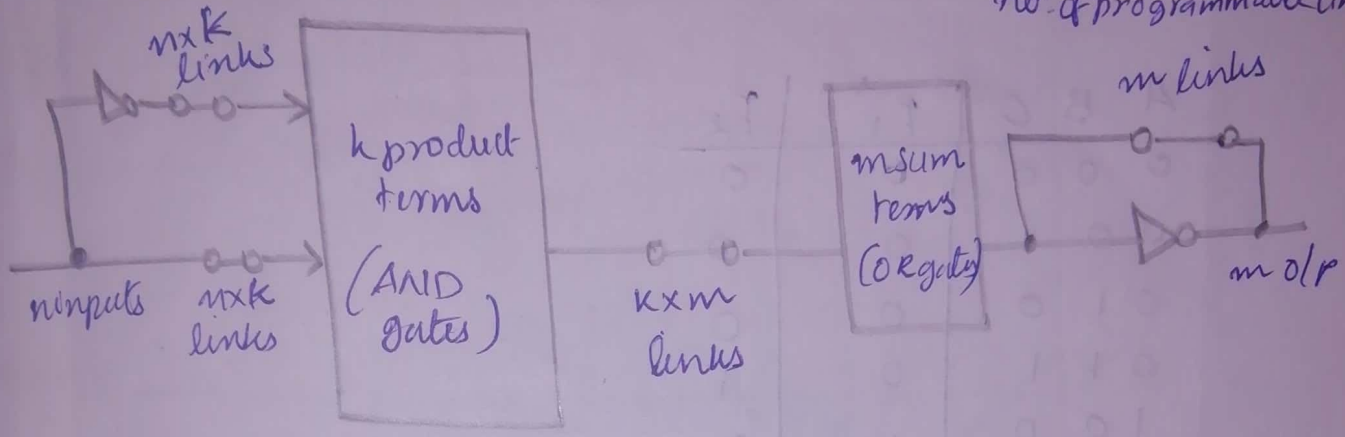
4 o/p

A_1	A_0	F_1	F_0
0	0	0	1
0	1	1	0
1	0	1	1
1	1	1	0



Programmable Logic Array (PLA)

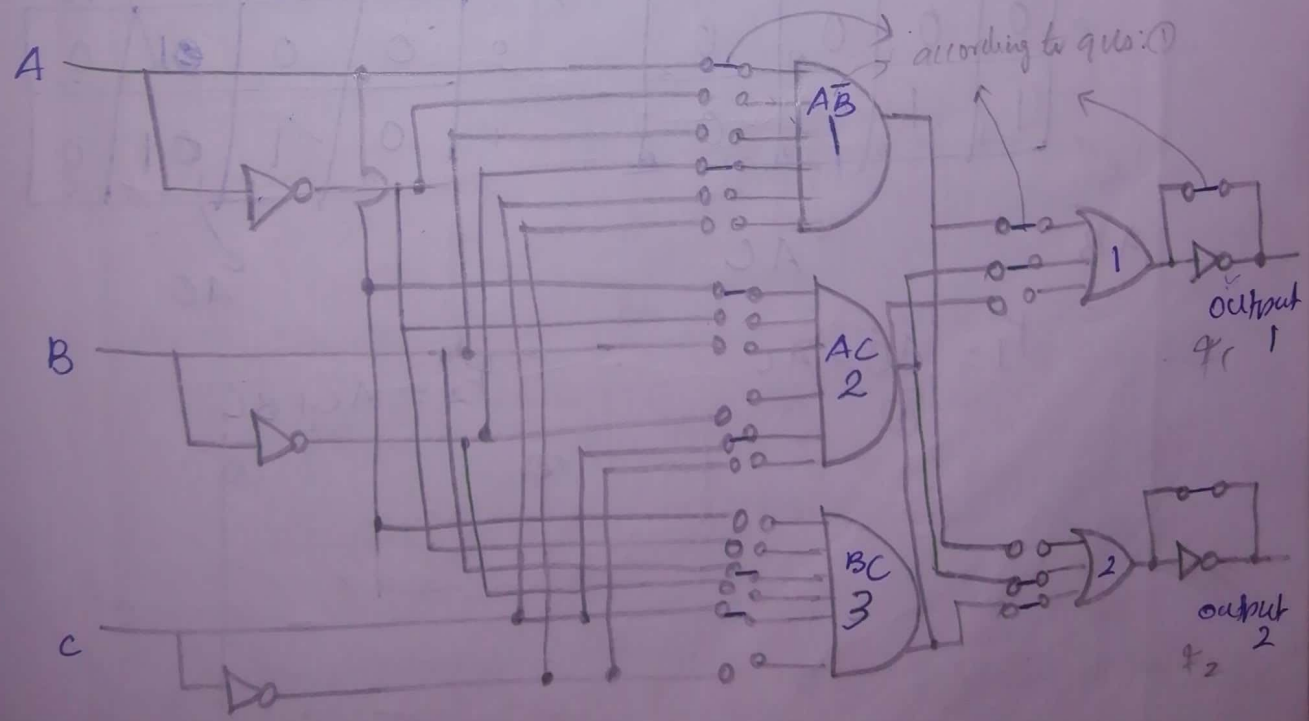
n - i/p
k - product term
m - output
no: of programmable links



PLA with three inputs, 3 product term and two output.

n = 3
k = 3
m = 2

No: of programmable links = 2n x k + k x m + m

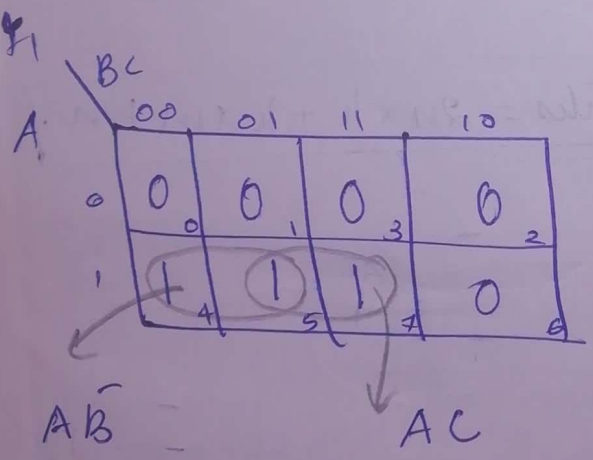


⇒ Implement functions F_1 and F_2, the following truth table using PLA with 3 inputs, 3 product terms and two outputs.

A	B	C	F_1	F_2
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	0
1	0	1	1	1
1	1	0	0	0
1	1	1	1	1

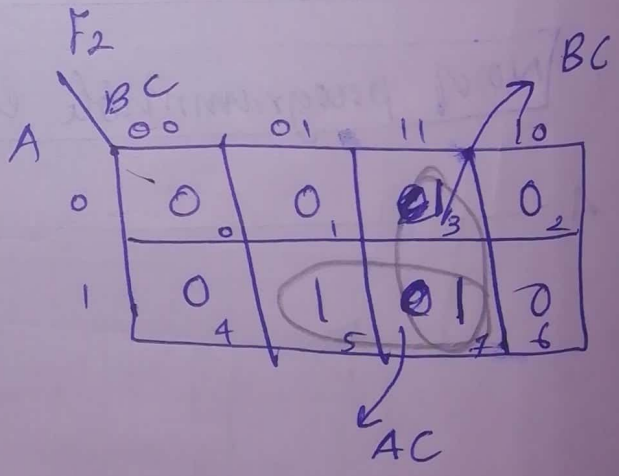
1) Convert to SOP

2)



$F_1 = A\bar{B} + AC$

① ②



$F_2 = AC + BC$

③

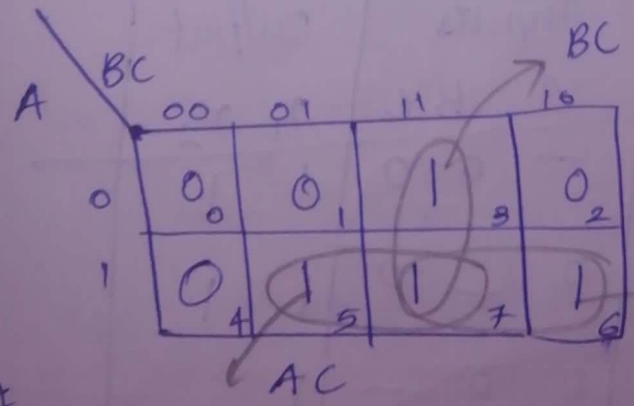
PLA program Table.

Product term		Inputs A B C			Output F_1	Output F_2
AB'	1	1	0	—	1	—
AC	2	1	—	1	1	1
BC	3	—	1	1	—	1
				T/C	T	T

V.IMP

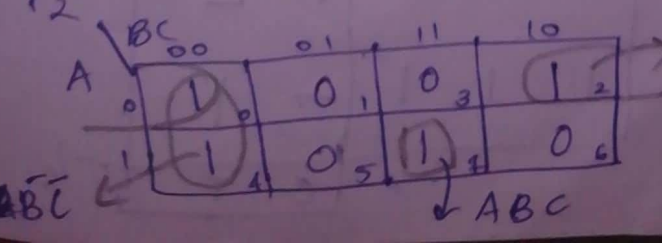
⇒ A combinational circuit is defined by the functi
$F_1(A,B,C) = \Sigma(3,5,6,7)$, $F_2(A,B,C) = \Sigma(0,2,4,$
Implement the circuit with PLA having 3 i/p
4 product term and 2 outputs.

F_1



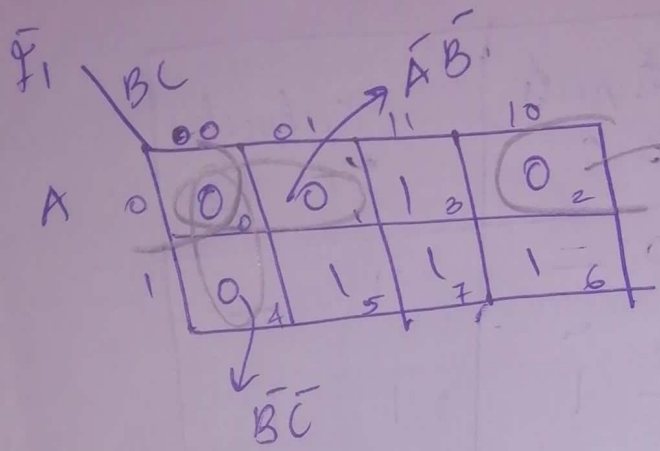
$F_1 = AC + A\bar{B} + BC$

F_2

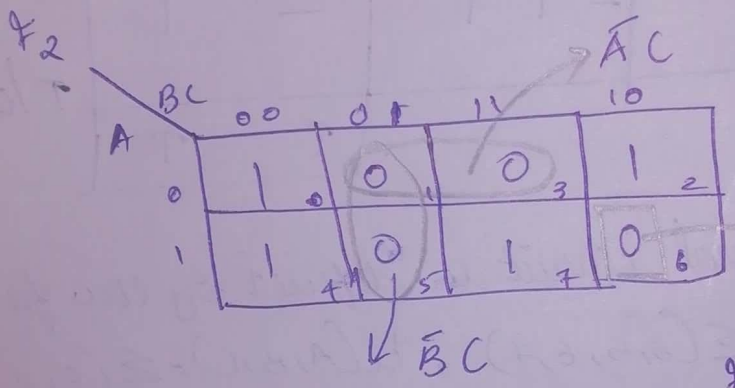


$F_2 = \bar{A}\bar{C} + ABC + B\bar{}$

circuit
Cmpt



$$\bar{F}_1 = \bar{A}\bar{B} + \bar{A}\bar{C} + \bar{B}\bar{C}$$



$$F_2 = AB\bar{C} + \bar{A}C + \bar{B}C$$

consider $\bar{F}_1$ & F_2 $[\bar{A}\bar{B}, \bar{A}\bar{C}, \bar{B}\bar{C}, ABC]$

PLA ~~program table~~

Product terms	Inputs A	Inputs B	Inputs C	output $\bar{F}_1$	output F_2
$B'C'$	—	0	0	1	1
$A'C'$	0	—	0	1	1
$A'B'$	0	0	—	1	—
ABC	1	1	1	—	1
				C	T

T/C