22/09/2014

UNIT-IV

REGISTERS AND COUNTERS

Applications of Flip-flops :-

(i) It can be used as a memory element.

(ii) It can be used to eliminate key debounce.

(iii) It is used as a basic building block in sequential circuits such as counters and Registers.

(iv) It can be used as a delay element.

(v)

REGISTERS :-

(i) Register is a group of flip-flops.

(ii) n-bit Register consists of 'n' number of flip-flops and it stores 'n' bit binary information.

Types of Shift Registers :-

1. Serial In serial out shift registor (SISO) :-

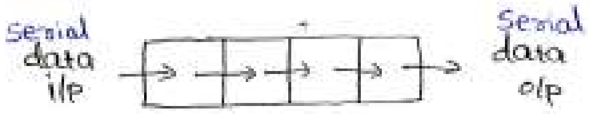


Fig: Data flow in SISO shift right register

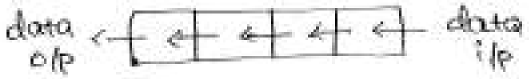


Fig: Data flow in SISO shift left register.

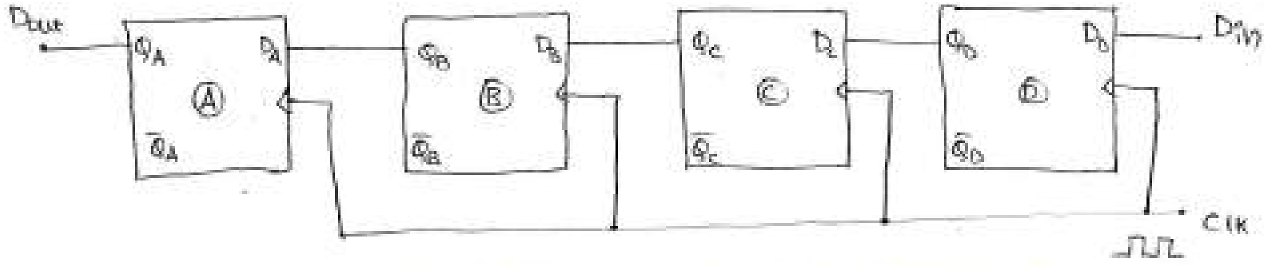


Fig: 4-bit SISO shift left register.

i/p to reg 1011.

Clk	reg. o/p (Dout) Q_A	Q_B	Q_C	Q_D	
0	0	0	0	0	
_⌐ (or) ↑	0	0	0	1	
↑	0	0	1	0	
↑	0	1	0	1	
↑	1	0	1	1	→ i/p data stored in register.
↑	0	1	1	0	
↑	1	1	0	0	
↑	1	0	0	0	→ all 4 data i/p bits are transfered to o/p side
↑	0	0	0	0	→ reg. cleared.

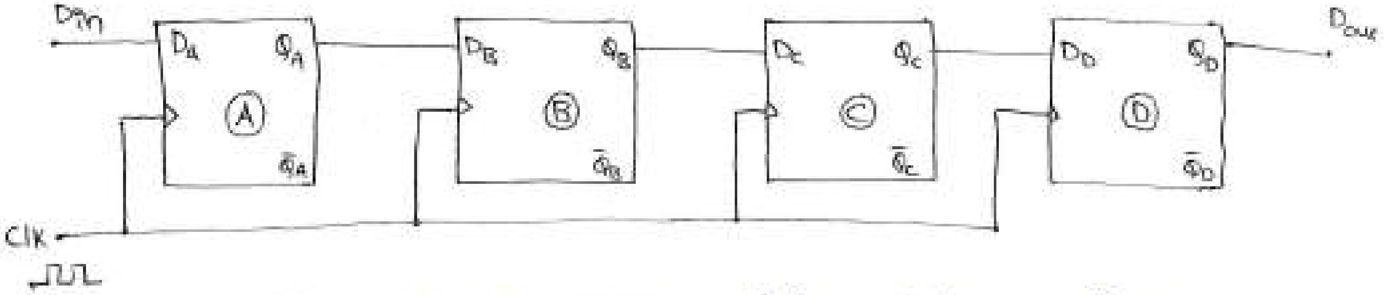


Fig:- 4-bit SISO shift right register.

2. Serial IN Parallel OUT shift Register (SIPO):-

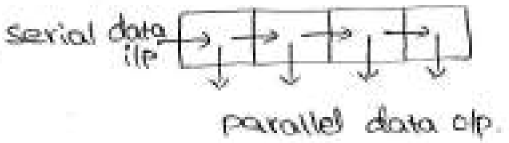


Fig:- data flow in SIPO shift right register.

In this case, data bits enters into register in serial, but o/p is taken in parallel.

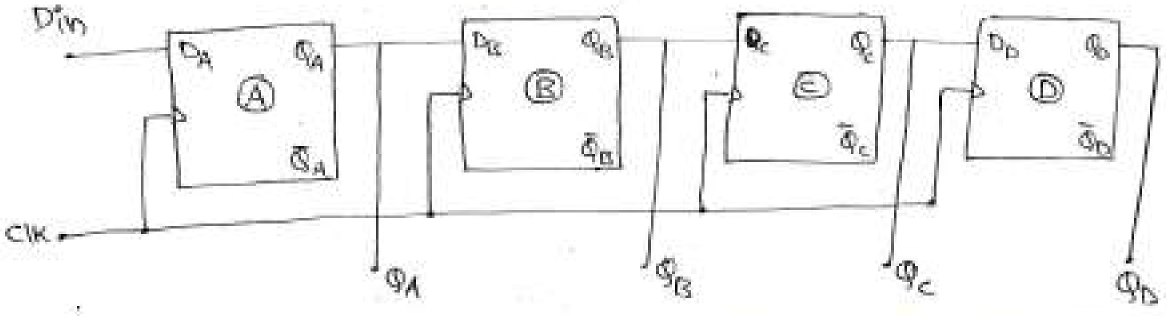


Fig:- 4-bit SIPO shift right register.

D_{in} → serial data i/p

$Q_A\, Q_B\, Q_C\, Q_D$ → 4-bit parallel data o/p.

Register with parallel load :-

* The transfer of new information into a register is refered to as loading Register.

* If all the bits of Register are loaded simultane-ously with a common clock pulse, we say that the loading is done in parallel.

3. Parallel IN parallel OUT shift register (PIPO):-

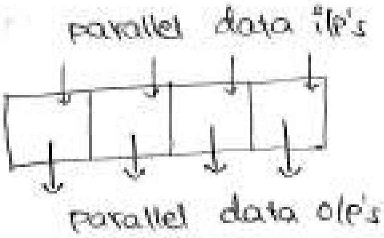


Fig:- data flow in PIPO shift register.

Data bits are entered into reg in parallel and o/p is taken in parallel.

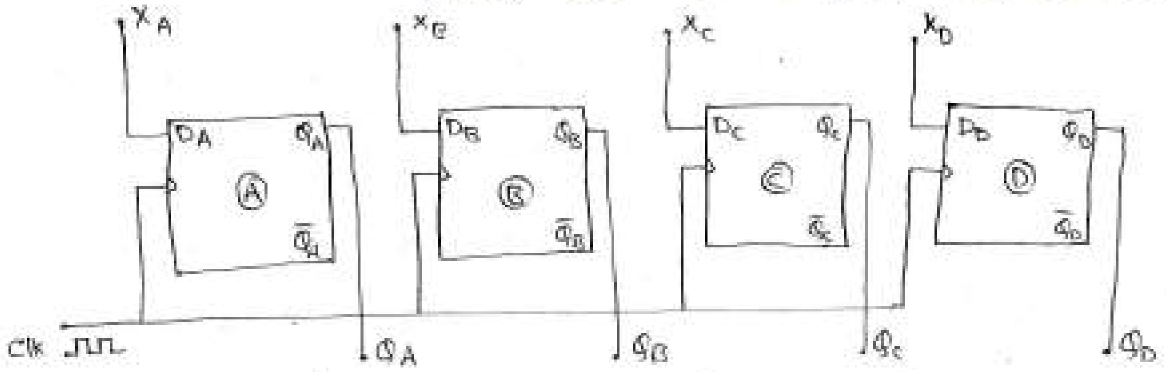


Fig: 4-bit PIPO shift register.

$X_A X_B X_C X_D$ → parallel data i/p's

$Q_A\, Q_B\, Q_C\, Q_D$ → 4-bit parallel data o/p.

07/10/2014

4. Parallel In serial out shift Register (PISO) :-

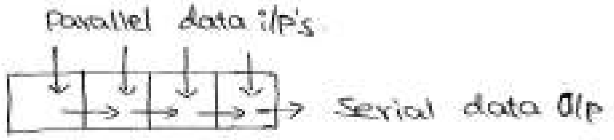


Fig: Dataflow in PISO shift right register.

I/p data entered into register parallely but o/p is taken serially.

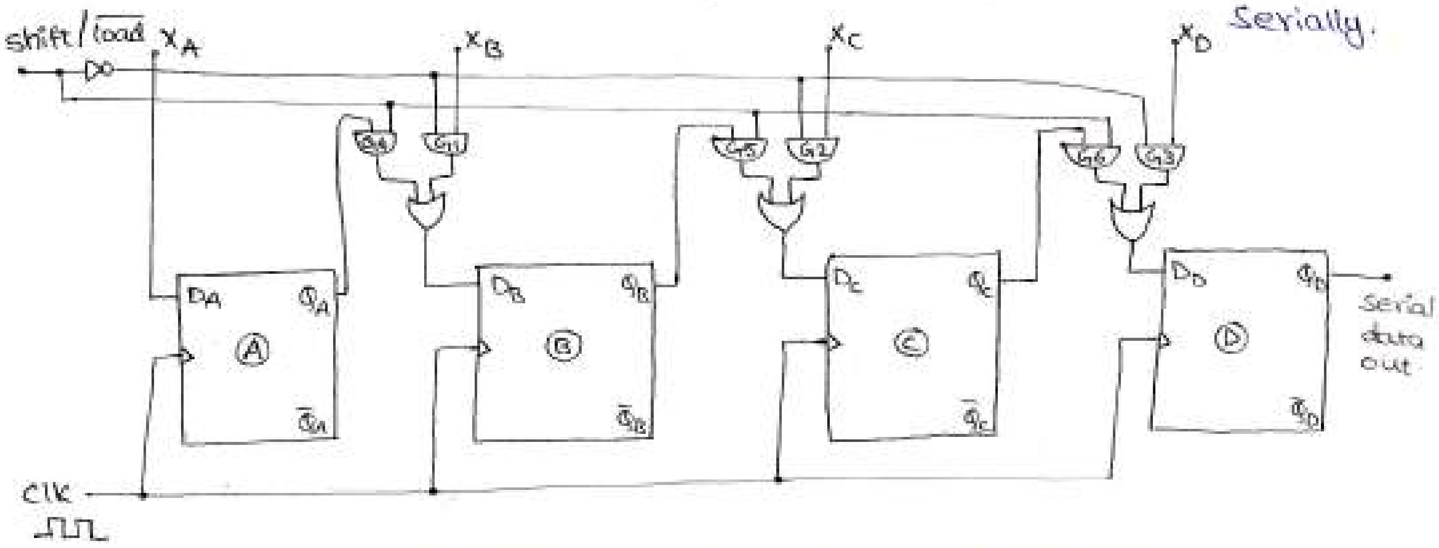


Fig: 4-bit PISO shift right register.

$X_A X_B X_C X_D$ → parallel data i/p's

If shift/$\overline{load}$ = 0 then parallel data loads into reg.

when shift/$\overline{Load}$ = 1, reg shifts the stored data to right

* Asynchronous (or) direct i/p's of flip-flops :-

(i) Set (or) pre-set:-

If set i/p is high then flip-flop o/p is '1' (for any data i/p) irrespective of data i/p's & clk i/p

If set is high i.e 1 $\overset{then}{\Rightarrow}$ flip-flop will set

If $\overline{Set}$ is low i.e 0 $\overset{then}{\Rightarrow}$ flip-flop will set.

(ii) clear (or) reset :-

If reset i/p is high then flip-flop o/p is '0' (for any data i/p) irrespective of clk & data i/p's

* Universal shift register :-

If the register has both shift (left & Right) and Parallel load capabilities, it is refered to as universal shift register.

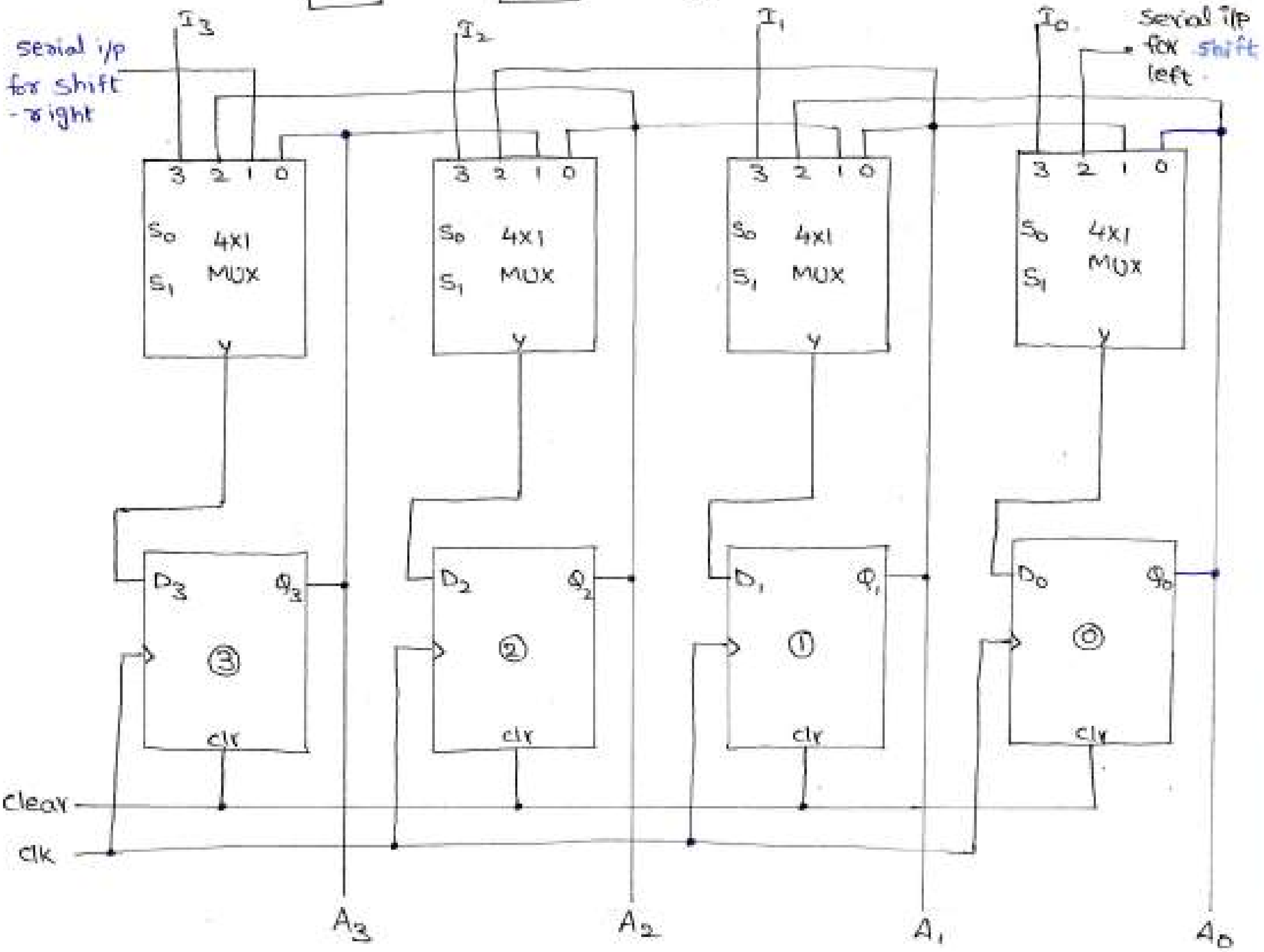
→ acts like PISO, PIPO, SISO, SIPO

→ acts like shift (right & left)

→ (i.e) performs all operations.

Note:- } both are same.

clr
$\overline{clr}$



S_0, S_1 → are common selection i/p's, but not o/p's of previous mux. For simplicity we have drawn in the above way.

Fig:- 4-bit universal shift register.

Function table :-

Mode control S_1	S_0	Reg. operation
0	0	NO change
0	1	shift right
1	0	shift left
1	1	parallel load

$I_3 I_2 I_1 I_0$ → parallel i/p's
$A_3 A_2 A_1 A_0$ → parallel o/p's.

~~for '00' we observe all '0' connections in MUX, gives o/p to D_1, D_2, D_3, D_0.~~

~~Similarly. for '11' we observe all '3' connections in MUX, gives o/p to D_0, D_1, D_2, D_3.~~

* Serial Adder :-

It performs serial addition.

Block diagram :-

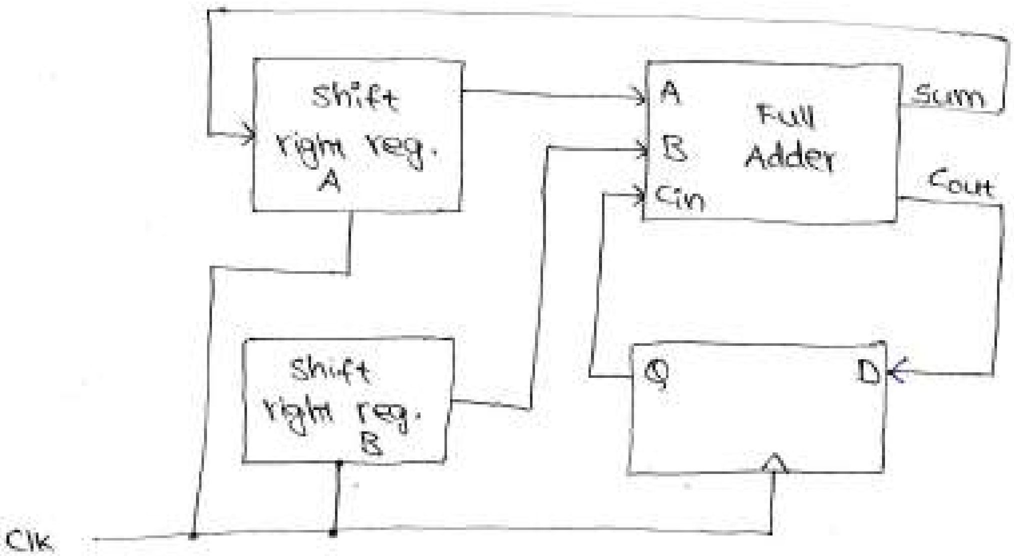


Fig: Serial Adder

let us consider two, 4-bit numbers →

$$\begin{array}{r} \overset{0}{1}\overset{1}{0}\overset{1}{1}1 \\ +1001 \\ \hline \underbrace{1}_{\text{end carry}}\underbrace{0100}_{\text{Sum}} \end{array}$$ → Intermediate carry

considering A & B are 4 bit registers (SISO shift right reg.)

A reg. content is 1011.
B reg. content is 1001.

serial adder performs operation.

$\boxed{A \leftarrow A+B}$ Arrow indicates dataflow direction.

when clk is applied Reg.'s performs shift right operation.

During shift right, LSB comes out & goes to full Adder. and FF adds A, B, C_{in} bits and transfers sum to A reg. & C_{out} to D-FF.

After performing serial addition contents of

reg A → 0100 (sum)
B → 0000
FF D → 1 (end carry)

Serial Transfer :-

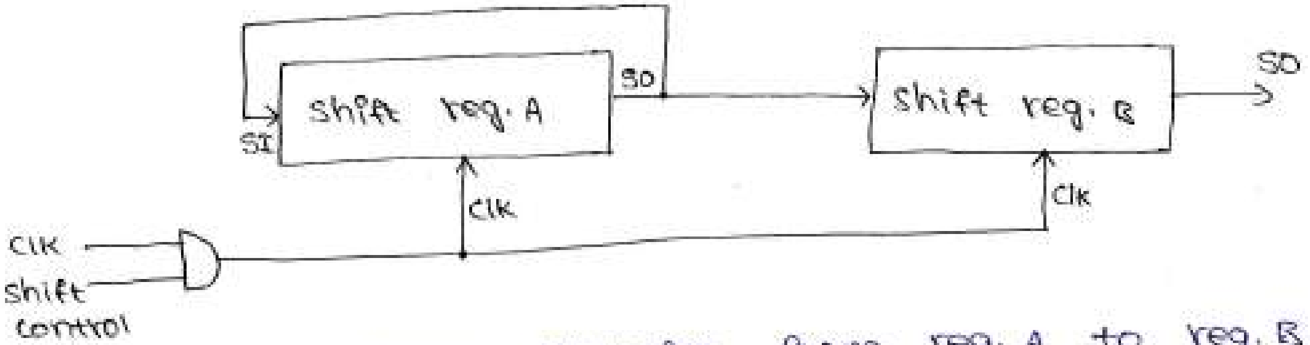


Fig:-(a) serial transfer from reg. A to reg. B.

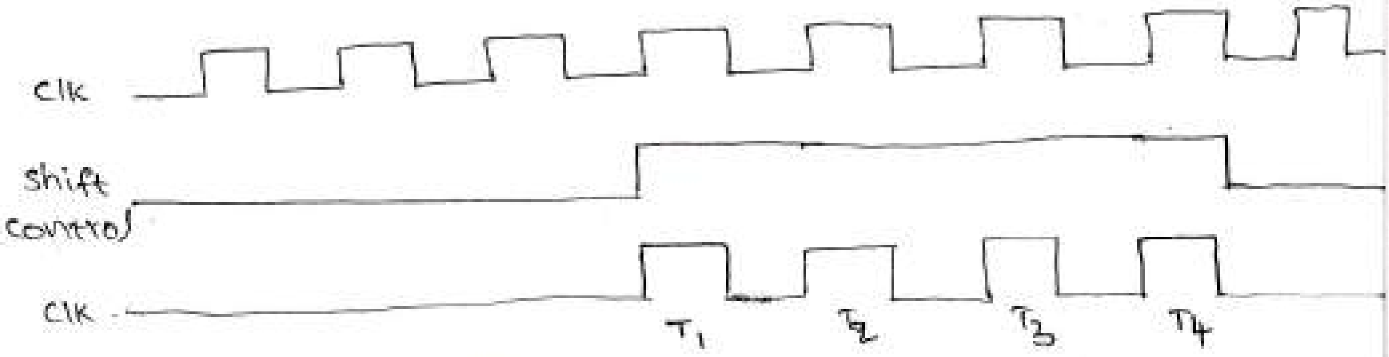


Fig:-(b) Timing diagram

considering A & B are 4-bit SISO shift right reg.'s.

SO → serial out ; SI → serial In.

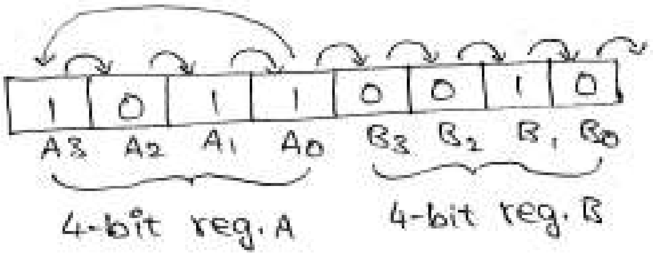
The above ckt performs operation

B ← A (or) A → B Arrow indicates direction of dataflow.

After performing register transfer operation A=B

Fig (a) performs shift operation only when shift control is high because reg.'s receives clk i/p only when shift control =1.

Initial values of A,B ⇒



serial transfer

Timing pulse	shift reg. A.	shift reg. B
Initial value of reg.	1011	0010
After T₁	1101	1001
T₂	1110	1100
T3	0111	0110
T₄	1011	1011

After T₄, both A & B reg's content is same.

(i.e) A=B.

05/10/2

COUNTERS :-

A counter is a register capable of counting the number of clock pulses arriving at it clock input. Count represents the number of clock pulses arrived.

(or)

A counter is an essentially a register that goes through a predetermined sequence of binary states.

* counters are available in two categories.

1. Asynchronous counters.
2. synchronous counters

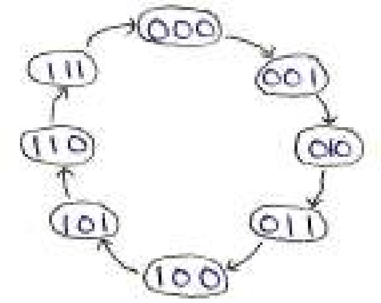
Asynchronous counters	synchronous counters
1. In this type of counter flip-flops are connected in such a way that o/p of 1st FF drives the clk for the next FF.	1. In this type, there is no connection b/w o/p of 1st FF and clk i/p of the next FF.
2. All the FF are not clocked simultaneously.	2. All the FF are clocked simultaneously.
3. logic circuit is very simple even for more number of states	3. Design involves complex logic circuits as no. of states increases.
4. Main drawback of these counters is their low speed as the clk is propagated through no. of FF before it reaches last FF.	4. As the clk is simultaneously given to all FFs there is no problem of propagation delay. Hence they are high speed counters

Asynchronous (or) Ripple (or) serial counters (or) Non-synchronous counter:-

1. 3-bit Asynchronous up counter :-

An upcounter is a counter which counts in the upward direction. (i.e) 0, 1, 2, 3, ... n

state diagram:-



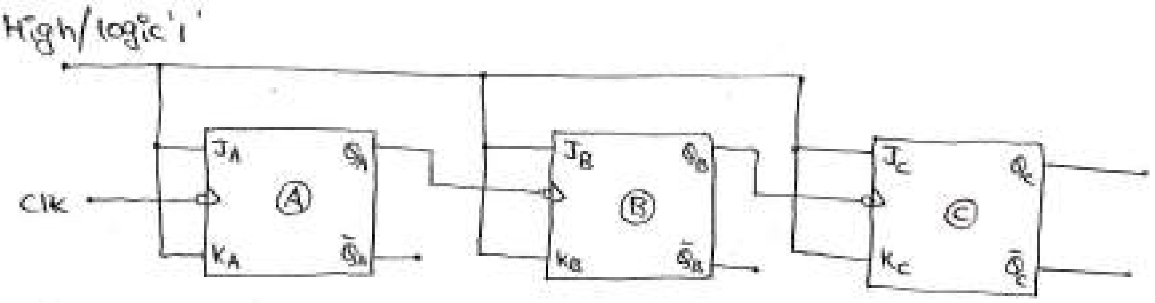
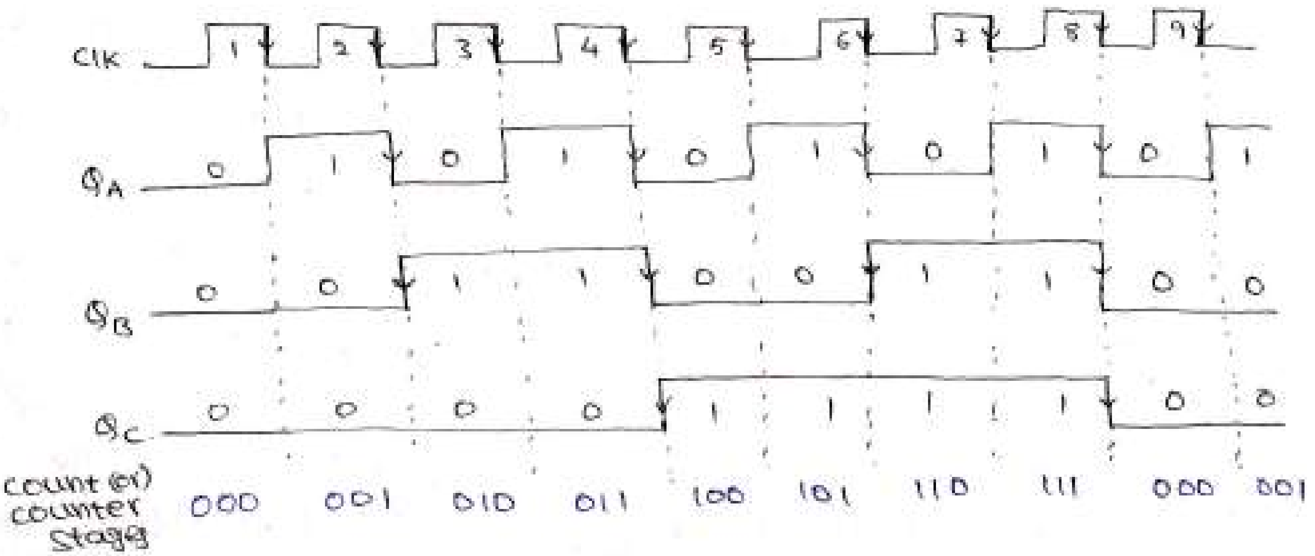


Fig:- 3-bit ripple up counter using '-ve' edge triggered JK FF's.

counter o/p → Q_C Q_B Q_A
MSB LSB

o/p waveforms



* The number of states through which the counter passes before returning to the starting state is called the Modulus of the counter.

(or)

The Mod number of a counter is the total no. of states it sequences through in each complete cycle

* Since a 3-bit counter has 8 states it is called a Mod-8 (or) Modulus-8 counter (or) Modulo-8 counter.

3-bit ripple up counter using T-FF's.

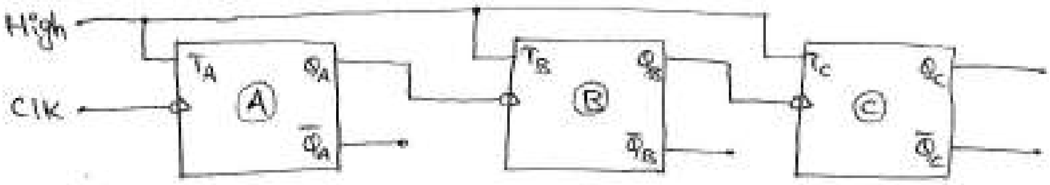


Fig: 3-bit Asynchronous up counter using '-ve' edge triggered T-FF's.

counter o/p → Q_C Q_B Q_A
MSB LSB.

Note:- For designing ripple up counter using '+ve' edge triggered FF's connect $\bar{Q}$ o/p's to clk i/p's of next flip-flops.

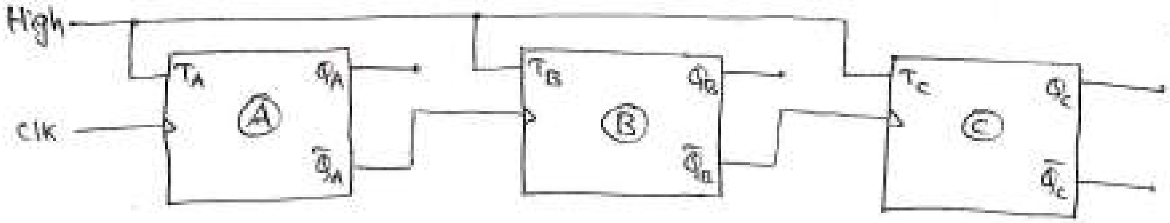


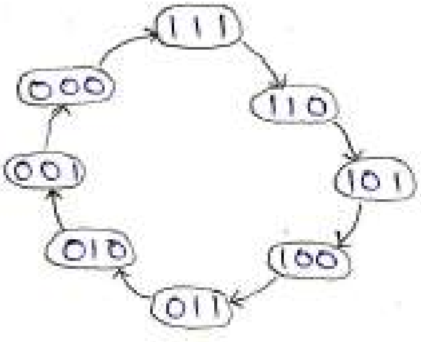
Fig:- 3-bit ripple up counter using '+ve' edge triggered T-FF's.

Counter o/p → Q_C Q_B Q_A
MSB LSB.

2. 3-bit Asynchronous down counter :-

Down counter counts in downward direction. (i.e) n, n-1, n-2, ..., 2, 1, 0.

State diagram:-



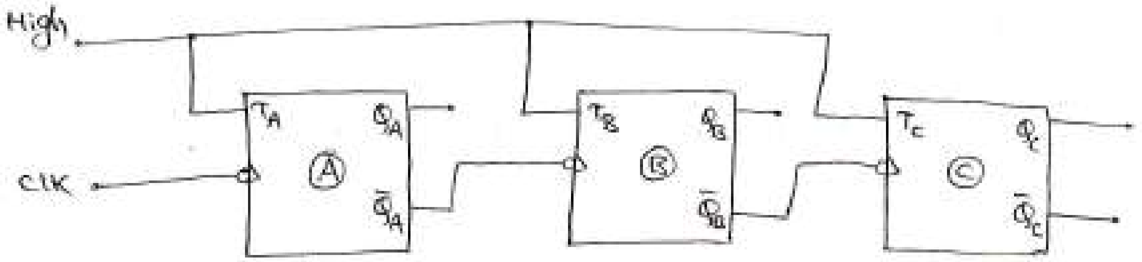
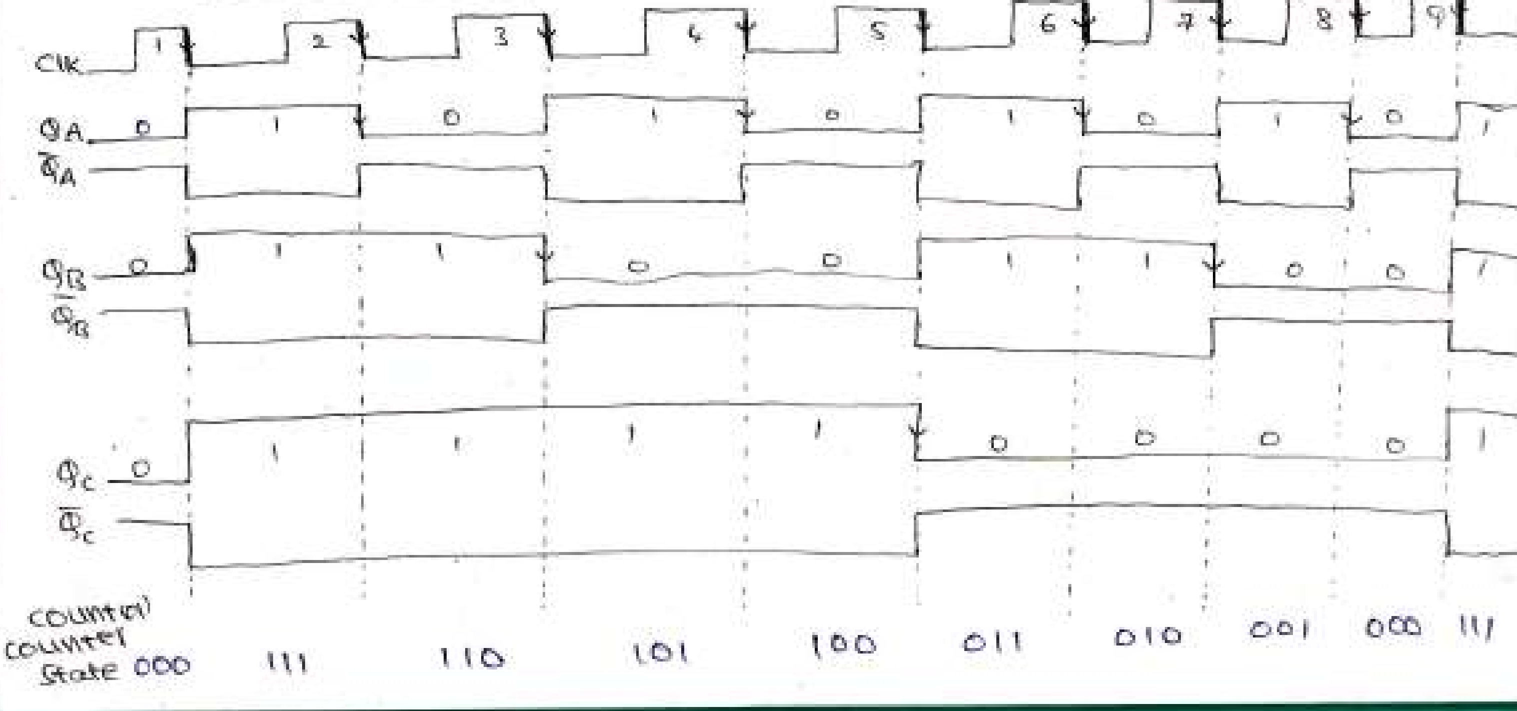


Fig:- 3-bit ripple down counter using '-ve' edge triggered T-FF's.

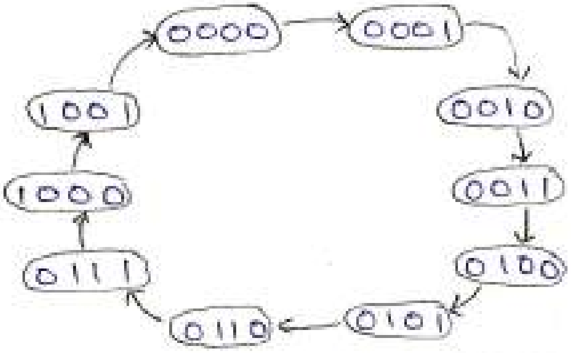
counter o/p → Q_C Q_B Q_A
MSB LSB

o/p waveforms



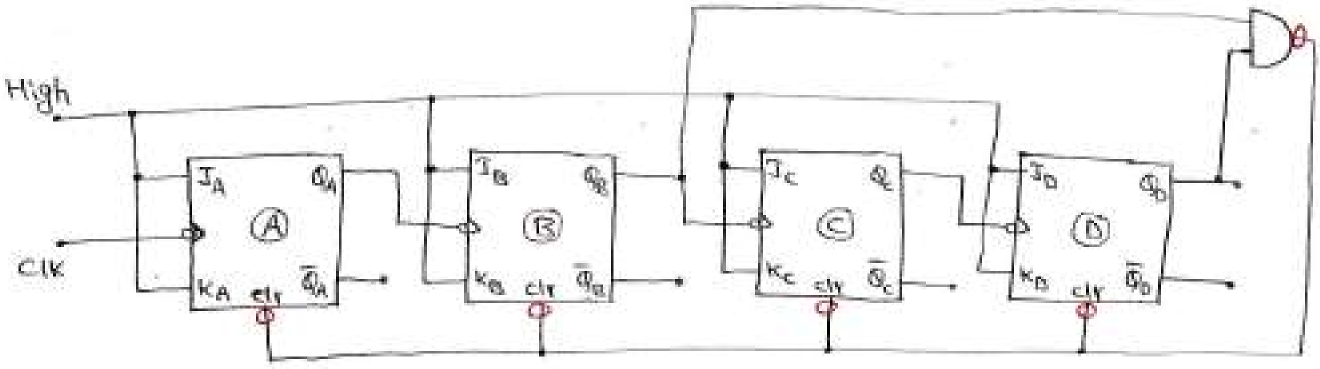
3. Asynchronous Mod-10 counter (or) Decade counter (or) BCD counter :-

state diagram:-



for NAND gate draw bubbles for clr i/p

for AND gate Dont draw bubbles



Fg:- MOD-10 ripple counter.

counter o/p → Q_D Q_C Q_B Q_A
MSB LSB

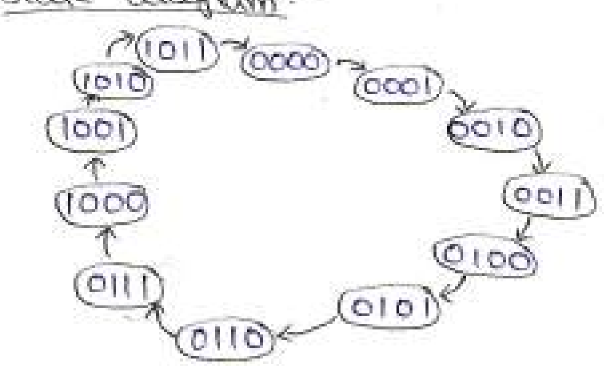
* When counter o/p is $\overset{Q_DQ_CQ_BQ_A}{1010}$, o/p of And gate becomes 1. Since o/p of And gate is connected to reset (or) clear i/p's of FF's, counter enters into 0000 state after 1010 state.

Truth table:-

clk	Q_D	Q_C	Q_B	Q_A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	0	0	0	0
11	0	0	0	1
12	0	0	1	0
13	0	0	1	1
14	0	1	0	0

4. Asynchronous Mod-12 counter :-

State diagram:-



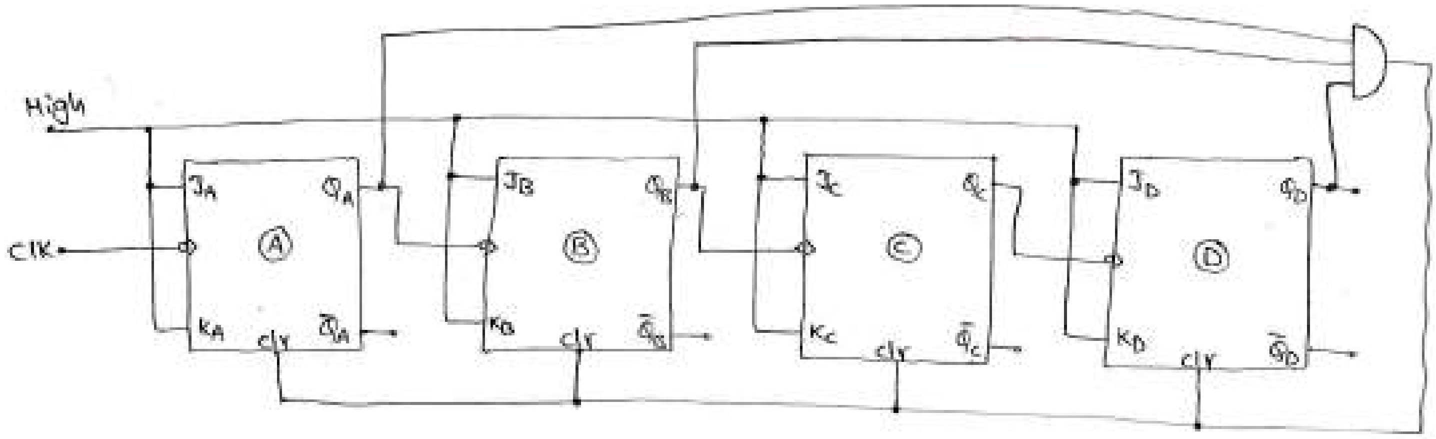


Fig: MOD-12 ripple counter

counter o/p → $Q_D\ Q_C\ Q_B\ Q_A$
MSB LSB

* When counter o/p is $\overset{Q_DQ_CQ_BQ_A}{1011}$, o/p of And gate becomes 1. Since o/p of And gate is connected to reset (or) clear i/p's of FF's, counter enters into 0000 state after 1011 state.

Truth table:-

CLK	Q_D	Q_C	Q_B	Q_A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0

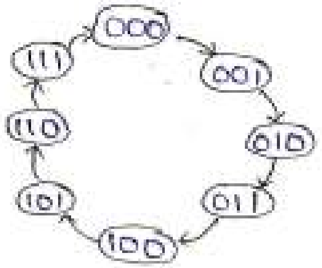
clk	Q_D	Q_C	Q_B	Q_A
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	0	0	0	0
13	0	0	0	1
14	0	0	1	0
15	0	0	1	1
16	0	1	0	0
17	0	1	0	1

14/10/2014

Synchronous (or) Parallel counters :-

1. 3-bit synchronous Binary up counter (or) Mod-8 Sync. counter:-

state diagram:-



Logic diagram:-

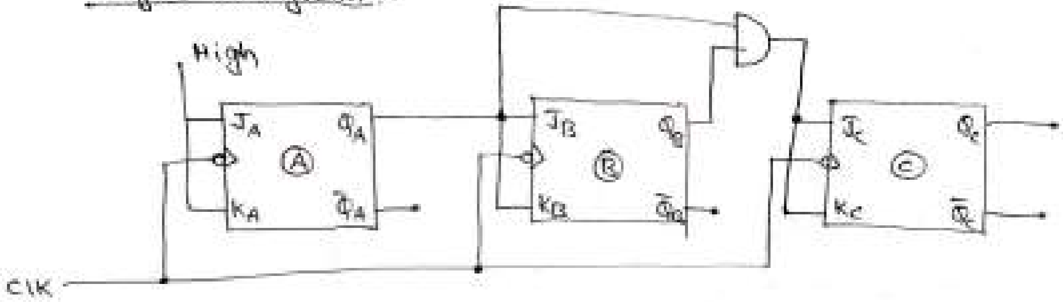


Fig:- MOD-8 Sync. counter

FF i/p's

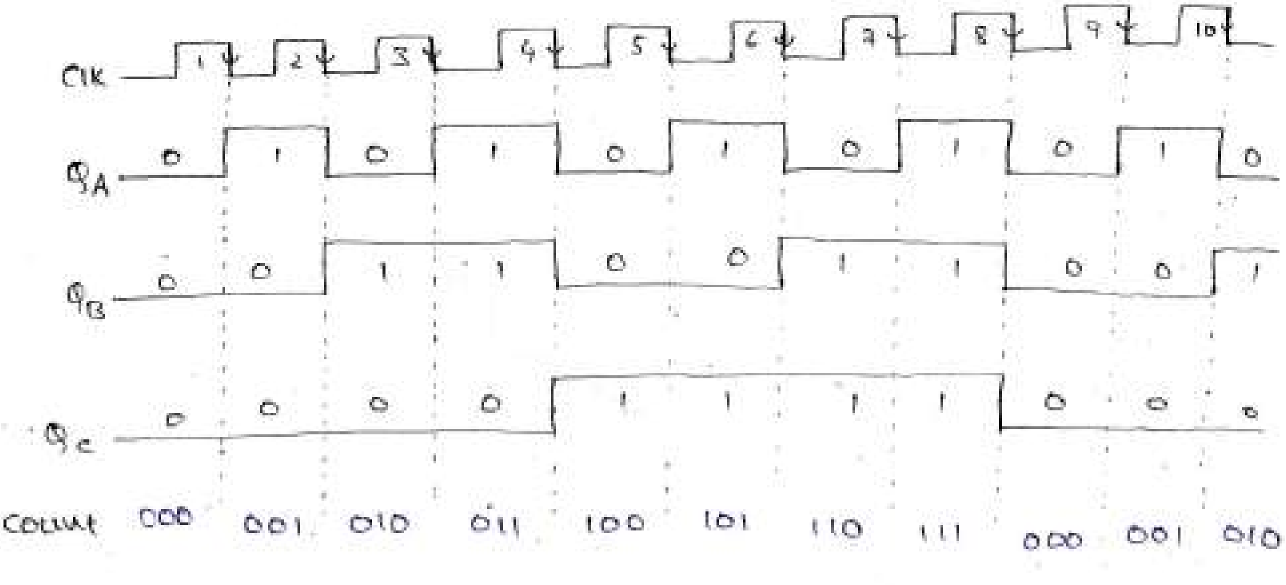
$J_A = K_A = 1$

$J_B = K_B = Q_A$

$J_C = K_C = Q_A \cdot Q_B$

counter o/p ⟶ $Q_C \; Q_B \; Q_A$
MSB LSB

output waveforms:-



2. 4-bit Synchronous Binary up counter (or) MOD-16 counter:-

Block diagram:-

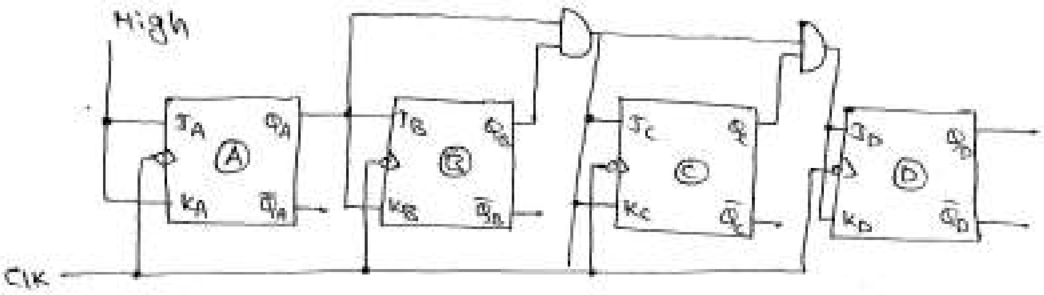


Fig:- MOD-16 Binary up counter

FF i/p's

$J_A = K_A = 1$

$J_B = K_B = Q_A$

$J_C = K_C = Q_A \cdot Q_B$

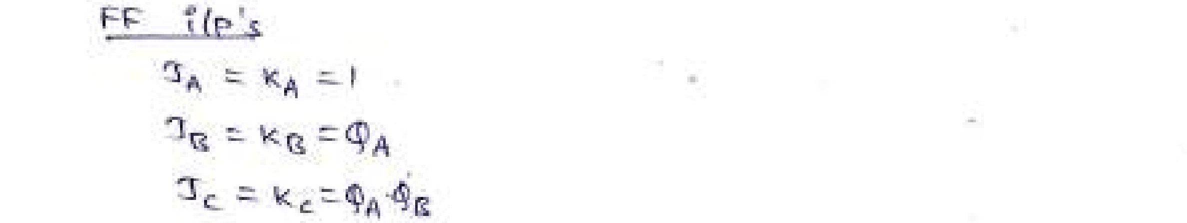
$J_D = K_D = Q_A \cdot Q_B \cdot Q_C$

counter o/p → $Q_D \; Q_C \; Q_B \; Q_A$

MSB LSB

3. 4-bit Synchronous Down counter (or) MOD-16 Down counter:-

State diagram:-



Logic diagram:-

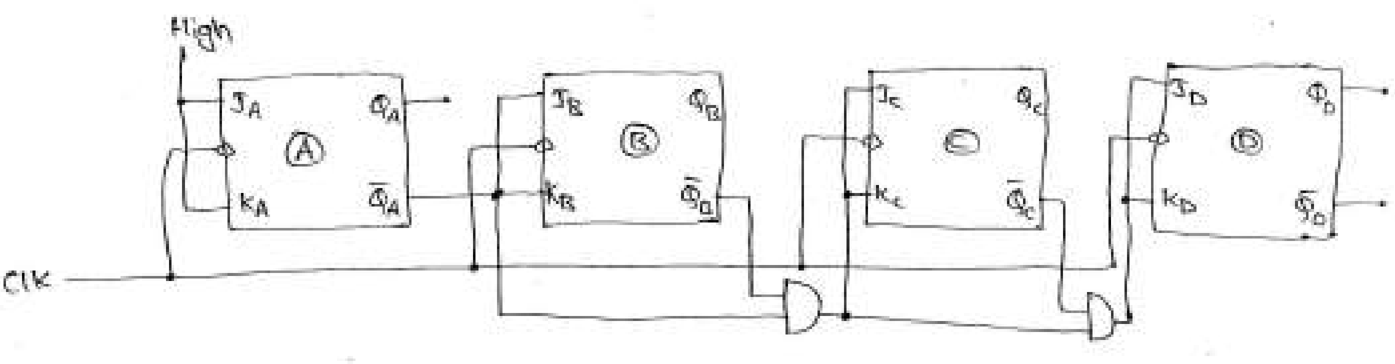


Fig: MOD-16 Down counter

FF i/p's

$J_A = K_A = 1$

$J_B = K_B = \bar{Q}_A$

$J_C = K_C = \bar{Q}_A \cdot \bar{Q}_B$

$J_D = K_D = \bar{Q}_A \cdot \bar{Q}_B \cdot \bar{Q}_C$

counter o/p → $\underset{MSB}{Q_D}\ Q_C\ Q_B\ \underset{LSB}{Q_A}$

Truth table (or) Function table :-

clk	counter o/p			
	Q_D	Q_C	Q_B	Q_A
0	0	0	0	0
1	1	1	1	1
2	1	1	1	0
3	1	1	0	1
4	1	1	0	0
5	1	0	1	1
6	1	0	1	0
7	1	0	0	1
8	1	0	0	0
9	0	1	1	1

clk	counter o/p			
	Q_D	Q_C	Q_B	Q_A
10	0	1	1	0
11	0	1	0	1
12	0	ϕ	0	0
13	0	0	1	1
14	0	0	1	0
15	0	0	0	1
16	0	0	0	0
17	1	1	1	1
18	1	1	1	0
19	1	1	0	1

* Design of Synchrous counters :-

Q. Design synchrous MOD-5 counter using JK flip-flops.

Sol:- Note:- If counter type is not mentioned then design up counter.

Step 1:- Determine number of flip-flops.

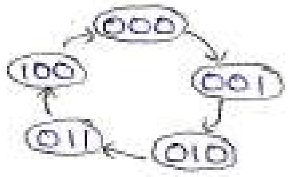
The no. of FF's required to design MOD-5 synchronous up counter can be determine by the equation, $2^n \geq N$ where $n \rightarrow$ no. of FF's
$N \rightarrow$ MOD no.

the possible value of 'n' which satisifies the above equation is 3.

thus MOD-5 counter uses 3 FF's.

Step 2:- Flip-flop type - JK FF.

Step 3 :- State diagram



Step 4 :- Excitation table

Present stage			Next stage			FF i/p's					
Q_C	Q_B	Q_A	Q_{C+1}	Q_{B+1}	Q_{A+1}	J_C	K_C	J_B	K_B	J_A	K_A
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	0	0	0	X	1	0	X	0	X
1	0	1	X	X	X	X	X	X	X	X	X
1	1	0	X	X	X	X	X	X	X	X	X
1	1	1	X	X	X	X	X	X	X	X	X

∵ JK FF Excitation table

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

Step 5 :- K-map simplification

for J_C

$Q_C \backslash Q_BQ_A$	00	01	11	10
0			1	
1	X	X	X	X

$\therefore J_C = Q_BQ_A$

for J_B

$Q_C \backslash Q_BQ_A$	00	01	11	10
0		1	X	X
1		X	X	X

$\therefore J_B = Q_A$

for J_A

$Q_C \backslash Q_BQ_A$	00	01	11	10
0	1	X	X	1
1		X	X	X

$\therefore J_A = \bar{Q}_C$

for K_C

$Q_C \backslash Q_BQ_A$	00	01	11	10
0	X	X	X	X
1	1	X	X	X

$\therefore K_C = 1$

for K_B

$Q_C \backslash Q_BQ_A$	00	01	11	10
0	X	X	1	
1	X	X	X	X

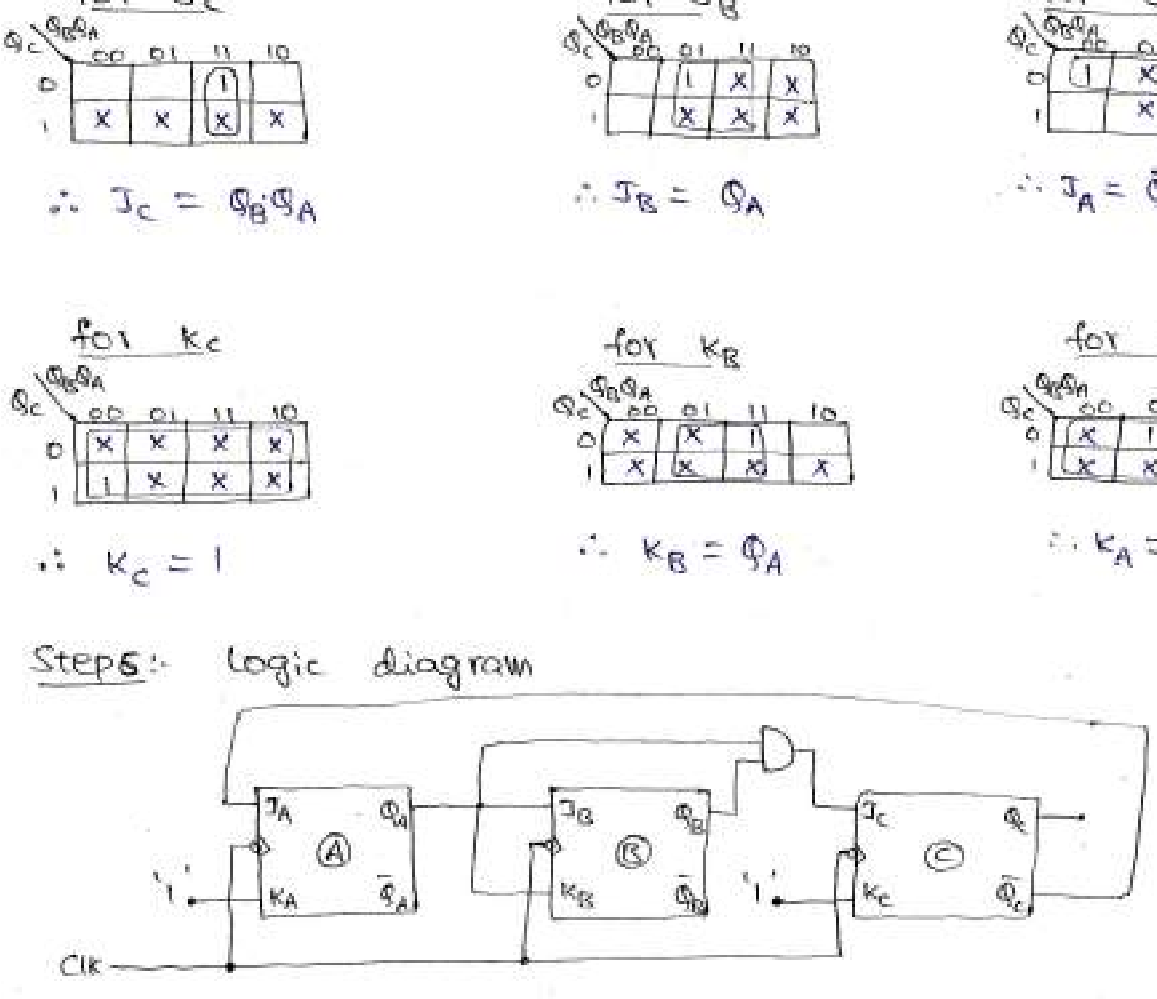
$\therefore K_B = Q_A$

for K_A

$Q_C \backslash Q_BQ_A$	00	01	11	10
0	X	1	1	X
1	X	X	X	X

$\therefore K_A = 1$

Step 6 :- logic diagram



15/10/2024

Q. Design MOD-10 synchronous up counter using T-flip-flops (Decade counter / BCD counter / MOD-10 counter).

Sol: Step 1: Determine no. of FF's required.

$2^n \geq N$ where N → Mod No.
n → no. of FF's.

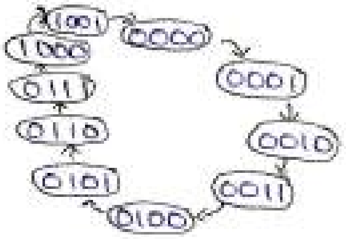
$2^n \geq 10$

$n = 4$

∴ No. of FF's required = 4

FF Type ⟶ T-FF

Step 2: State diagram



Step 3: Excitation table

Present stage				Next stage				FF i/p's			
Q_D	Q_C	Q_B	Q_A	Q_{D+1}	Q_{C+1}	Q_{B+1}	Q_{A+1}	T_D	T_C	T_B	T_A
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	1
0	0	1	1	0	1	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	1
0	1	0	1	0	1	1	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	1
0	1	1	1	1	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	1
1	0	0	1	0	0	0	0	1	0	0	1
1	0	1	0	X	X	X	X	X	X	X	X
1	0	1	1	X	X	X	X	X	X	X	X
1	1	0	0	X	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X	X

∵ T-FF excitation table

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

Step 4 : K-map simplification

for T_D

$Q_DQ_C \backslash Q_BQ_A$	00	01	11	10
00				
01			1	
11	X	X	X	X
10		1	X	X

$\therefore T_D = Q_D \cdot Q_A + Q_C Q_B Q_A$

for T_C

$Q_DQ_C \backslash Q_BQ_A$	00	01	11	10
00			1	
01			1	
11	X	X	X	X
10			X	X

$\therefore T_C = Q_B \cdot Q_A$

for T_B

$Q_DQ_C \backslash Q_BQ_A$	00	01	11	10
00		1	1	
01		1	1	
11	X	X	X	X
10			X	X

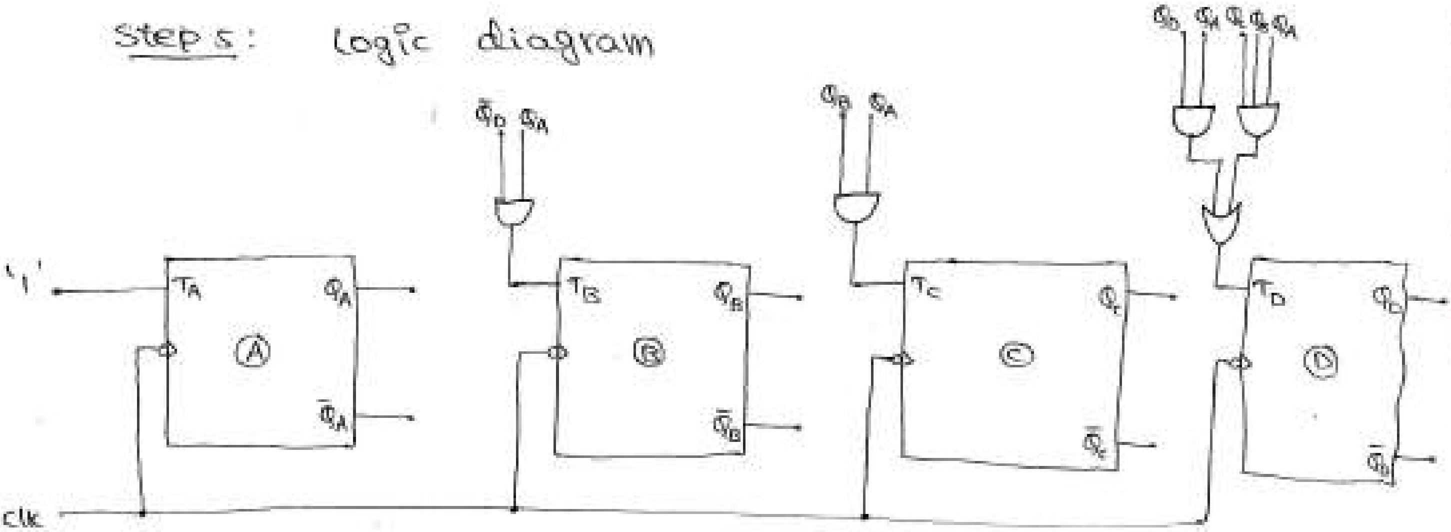
$\therefore T_B = \bar{Q}_D \cdot Q_A$

for T_A

$Q_DQ_C \backslash Q_BQ_A$	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	X	X	X	X
10	1	1	X	X

$\therefore T_A = 1$

Step 5 : logic diagram



counter o/p → Q_D Q_C Q_B Q_A

MSB LSB

Counter with unused States :-

Q. Design a modulo-5 counter to count the sequence 0, 1, 3, 7, 6. Design should include circutary to ensure that if we end up in an unwanted state, the next clock pulse will reset the counter to $Q_2 \cdot Q_1 \cdot Q_0 = 000$ use T-flipflops.

Sol:- Step 1: no. of FF's required to design MOD-5 counter can be determined by equation,

$2^n \geq N$ where $n \rightarrow$ no. of FF's
$N \rightarrow$ MOD no.

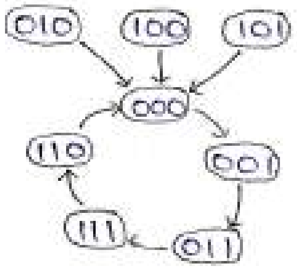
$\therefore 2^n \geq 5$

$\therefore n = 3$

$\therefore$ $\boxed{\begin{array}{l}\text{no. of FF's required} \longrightarrow 3 \\ \text{FF's using} \longrightarrow \text{T FF's}\end{array}}$

Step 2: State diagram

used states ⟶ 0, 1, 3, 7, 6
unused states ⟶ 2, 4, 5



($\because$ for unused states next stage is 000)

Step 3: Excitation table

Present State			Next state			FF i/p's		
Q_2	Q_1	Q_0	Q_{2+1} (or) Q_{2+}	Q_{1+1} Q_{1+}	Q_{0+1} Q_{0+}	T_2	T_1	T_0
0	0	0	0	0	1	0	0	1
0	0	1	0	1	1	0	1	0
0	1	0	0	0	0	0	1	0
0	1	1	1	1	1	1	0	0
1	0	0	0	0	0	1	0	0
1	0	1	0	0	0	1	0	1
1	1	0	0	0	0	1	1	0
1	1	1	1	1	0	0	0	1

$\because$ T-FF excitation table

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

Step 4 : K-Map simplification

for T_2

$Q_2 \backslash Q_1Q_0$	00	01	11	10
0			1	
1	1	1		1

$\therefore T_2 = Q_2 Q_1 + Q_2 \bar{Q}_0 + \bar{Q}_2 Q_1 Q_0$

for T_1

$Q_2 \backslash Q_1Q_0$	00	01	11	10
0		1		1
1				1

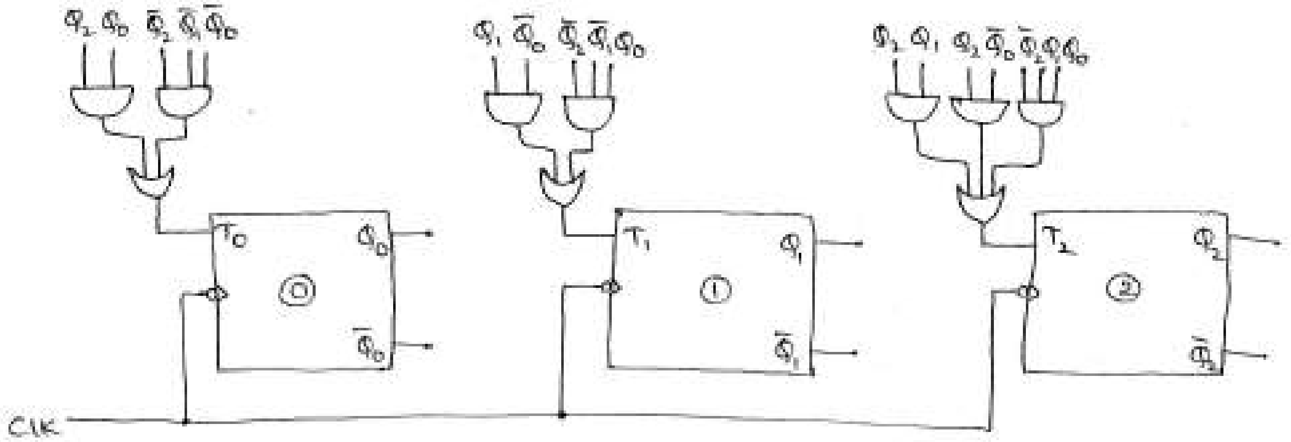
$\therefore T_1 = Q_1 \bar{Q}_0 + \bar{Q}_2 \bar{Q}_1 Q_0$

for T_0

$Q_2 \backslash Q_1Q_0$	00	01	11	10
0	1			
1		1	1	

$\therefore T_0 = \bar{Q}_2 \bar{Q}_1 \bar{Q}_0 + Q_2 Q_0$

Step 5 : Logic diagram



$Q_2 Q_0$ $\bar{Q}_2 \bar{Q}_1 \bar{Q}_0$
$Q_1 \bar{Q}_0$ $\bar{Q}_2 \bar{Q}_1 Q_0$
$Q_2 Q_1$ $Q_2 \bar{Q}_0$ $\bar{Q}_2 Q_1 Q_0$
T_0
Q_0
0
$\bar{Q}_0$
T_1
Q_1
1
$\bar{Q}_1$
T_2
Q_2
2
$\bar{Q}_2$
CLK

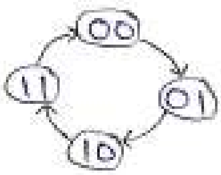
counter o/p → Q_2 Q_1 Q_0
MSB LSB

Q. Design a Synchronous counter with States 0,1,2,3, 0,1, ... using JK FF's.

Sol: counter states are 0,1,2,3,0,1, ...

∴ It is MOD-4 counter.

State diagram:-



Q. Design synchronous counter using JK FF. to count the following sequence

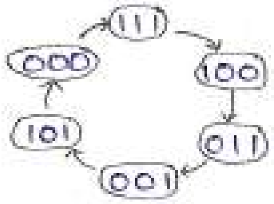
7, 4, 3, 1, 5, 0, 7, ...

Sol:. Total no. of states → 6

∴ It is MOD-6 counter.

Used states → 7,4,3,1,5,0.

State diagram:-



Unused states → 2,6 .

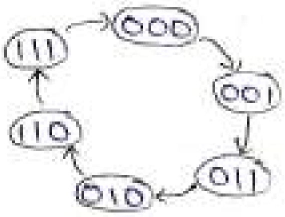
Treat unused states as don't cares.

Q. Design a synchronous gray code MOD-6 up counter.

Sol:.

	Binary	Gray
0	0 0 0	0 0 0
1	0 0 1	0 0 1
2	0 1 0	0 1 1
3	0 1 1	0 1 0
4	1 0 0	1 1 0
5	1 0 1	1 1 1

State diagram:-



Unused states are 4,5.

∴ treat them as don't cares.

* Synchronous UP/Down counter (or) Multimode counter (or) Bidirectional counter :-

Q. Design 3-bit synchronous up/down counter.

Sol: → The counter which is capable of progressing in either direction (i.e) in ascending order (incrementing order) (or) descending order (decreasing order) through a certain counting sequence is known as up/down counter.

→ Usually, up/down operation of the counter is controlled by $UP/\overline{down}$ (M) signal.

→ When M=1, the counter goes through up sequence (0, 1, 2, ----, n).

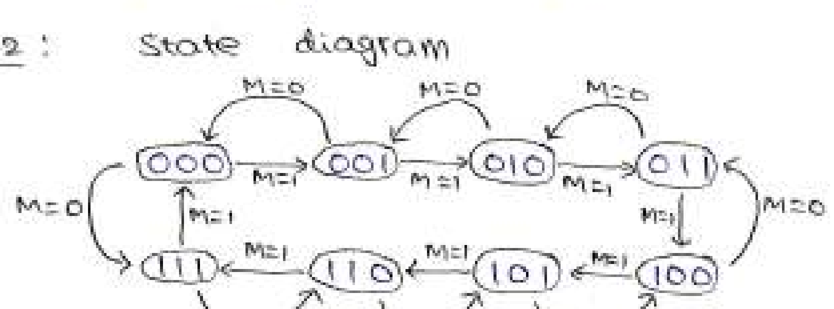
→ When M=0, the counter goes through Down sequence (n, n-1, - - - -, 2, 1, 0).

Step 1 :

no. of FF's required → 3 (∵ it is 3-bit counter)

FF's using → T FF's.

Step 2 : State diagram



Step 3: Excitation table

	control i/p	present State			Next Stage			FF i/p's		
	UP/down (M)	Q_c	Q_B	Q_A	Q_{c+1}	Q_{B+1}	Q_{A+1}	T_c	T_B	T_A
Down	0	0	0	0	1	1	1	1	1	1
	0	0	0	1	0	0	0	0	0	1
	0	0	1	0	0	0	1	0	1	1
	0	0	1	1	0	1	0	0	0	1
	0	1	0	0	0	1	1	1	1	1
	0	1	0	1	1	0	0	0	0	1
	0	1	1	0	1	0	1	0	1	1
	0	1	1	1	1	1	0	0	0	1
UP	1	0	0	0	0	0	1	0	0	1
	1	0	0	1	0	1	0	0	1	1
	1	0	1	0	0	1	1	0	0	1
	1	0	1	1	1	0	0	1	1	1
	1	1	0	0	1	0	1	0	0	1
	1	1	0	1	1	1	0	0	1	1
	1	1	1	0	1	1	1	0	0	1
	1	1	1	1	0	0	0	1	1	1

∵ T-FF excitable table

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

Step 4: K-Map simplification

for T_c

$MQ_c \backslash Q_BQ_A$	00	01	11	10
00	1			
01	1			
11			1	
10			1	

$$\therefore T_c = \bar{M}\bar{Q}_B\bar{Q}_A + MQ_BQ_A$$

$$= M \odot Q_B \odot Q_A$$

for T_B

$MQ_c \backslash Q_BQ_A$	00	01	11	10
00	1			1
01	1			1
11		1	1	
10		1	1	

$$\therefore T_B = \bar{M}\bar{Q}_A + MQ_A$$

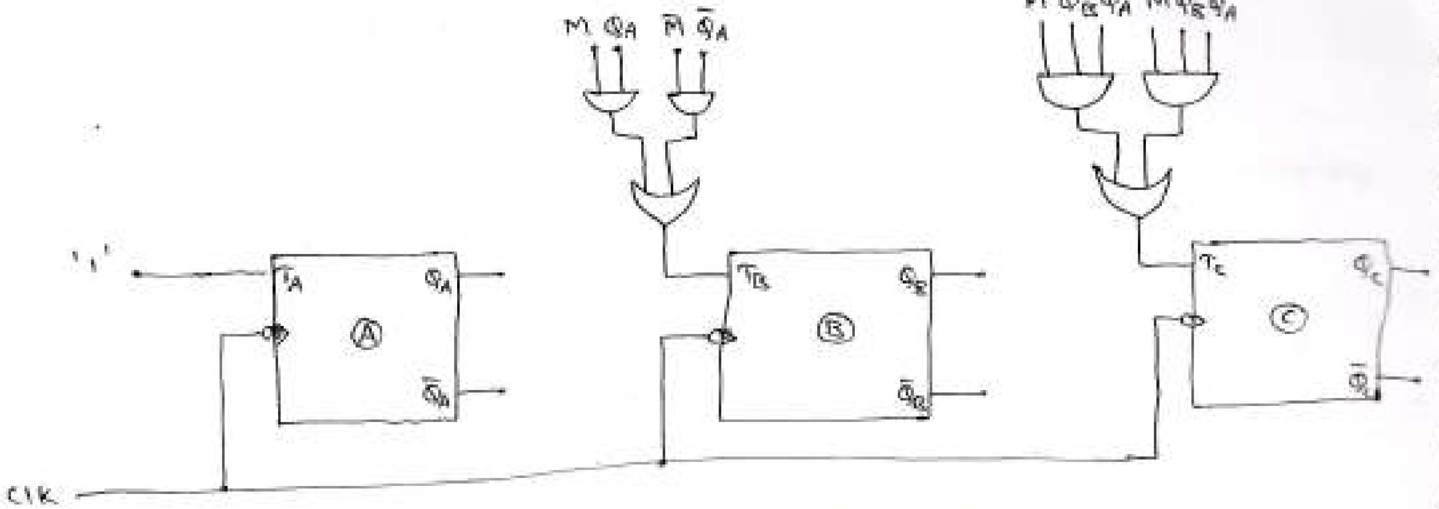
$$= M \odot Q_A$$

for T_A $MQ_c \backslash Q_BQ_A$

1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1

$$\therefore T_A = 1$$

logic diagram :-



counter o/p → Q_C Q_B Q_A

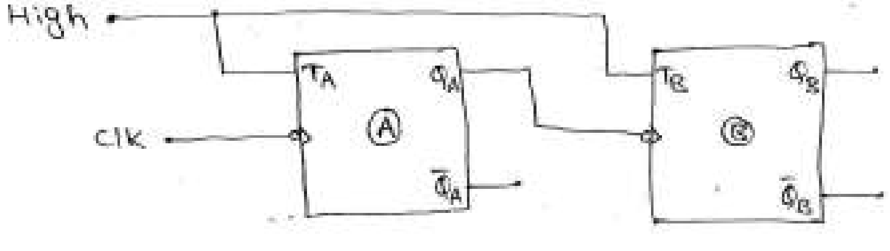
MSB LSB

16/10/2014

* Frequency dividers :-

2-bit Asynchronous counter (or) MOD-4 counter (or) Divide-by-4 counter :-

logic diagram :-

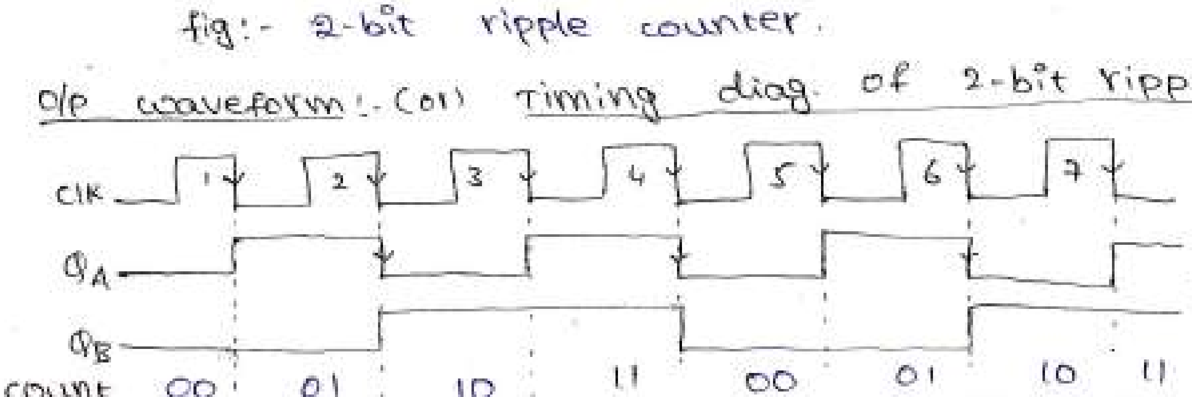


counter o/p → Q_B Q_A

MSB LSB

fig:- 2-bit ripple counter.

o/p waveform :- (or) Timing diag. of 2-bit ripple counter :-



* From the timing diagram it is clear that the frequency of Q_A is one half of the frequency of clk signal, Q_B is one half of Q_A and ~~th~~ is one fourth of the clk frequency.

* If the clk frequency is 1000 Hz, then $Q_A = 500$ Hz & $Q_B = 250$ Hz. Hence the 2-bit ripple counter is also called as divide-by-4 counter.

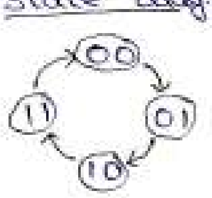
Note:- Each FF acts as a divide by 2 (÷ 2) frequency divider. (i.e) Each FF divides the incoming clk signal frequency by 2.

3-bit counter → MOD-8 counter → Divide-by-8 counter
4-bit counter → MOD-16 counter → Divide-by-16 counter
MOD-5 counter → Divide-by-5 counter
MOD-6 counter → Divide-by-6 counter

State table :-

The State table represents the state diagram in tabular form.

State diag:-



State table :-

Present State	Next State
0 0	0 1
0 1	1 0
1 0	1 1
1 1	0 0

* Lock out condition :-

In a counter if the next state of some unused state is again an unused state and if by chance the counter happen to find itself in the unused states and never arrived at a used state then the counter is said to be in the "lockout condition".

The circuit that goes in lockout condition is called "bushless circuit"

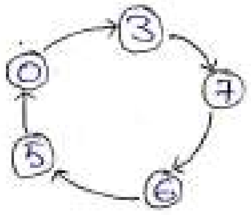


Fig:- a) Desired sequence

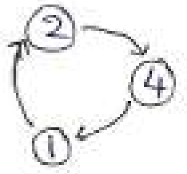


Fig:- b) Unused states forming lockout.

To avoid lockout condition, the unused states are introduced infront of the used states

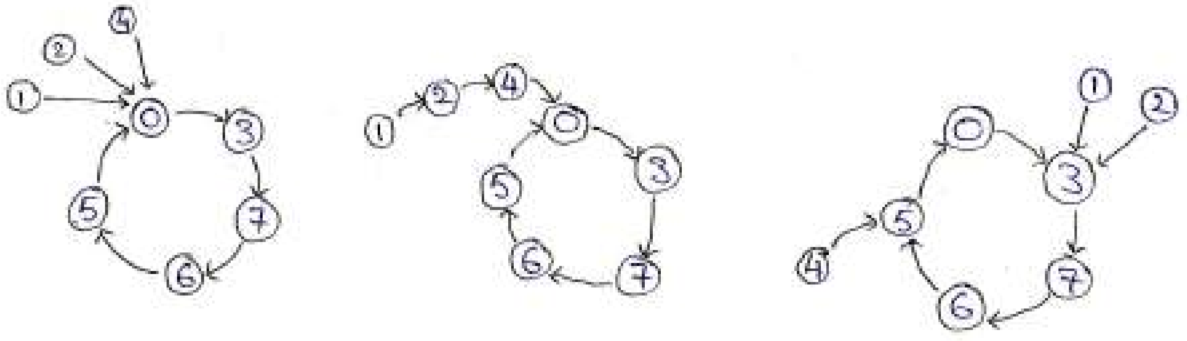
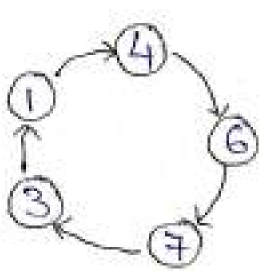


Fig:- state diagrams for removing lockout.

Q:- Design a synchronous counter for the following sequence. 4 → 6 → 7 → 3 → 1 → 4 .....
Avoid lockout condition use JK type of design.

Sol:- State diagram:-



Used states → 1, 3, 4, 6, 7

Unused states → 0, 2, 5

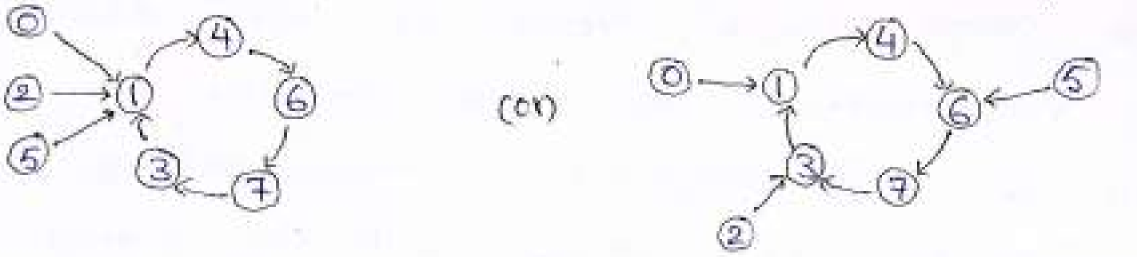


Fig:- state diagrams for Avoiding lockout condition.

* Shift Register counters :-

A shift Register counter is basically a shift register wch with the serial o/p connected back to the serial i/p to produce special sequences. these devices are classified as counters because they exhibit a special sequence of states.

Two of the Most common types of shift Register counters are

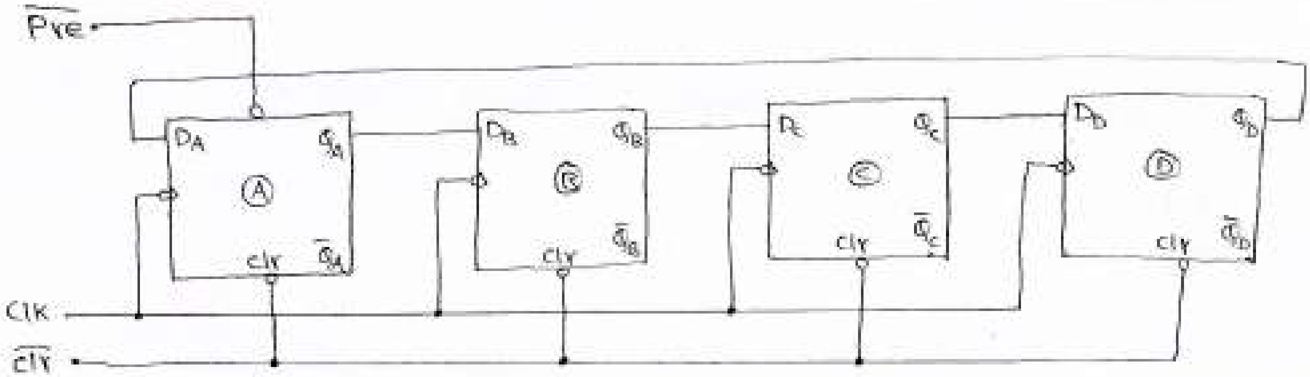
(i) Ring counter

(ii) Johnson counter

1. Ring counter :-

A Ring counter is a circular shift Register with only 1 FF being set at any particular time, all others are cleared.

The single bit is shifted from one FF to the next to produce the sequence of timing signals.



counter o/p → Q_D Q_C Q_B Q_A
MSB LSB

clr→ clear (or)
Reset i/p
Pre→ set (or)
Preset i/p

Fig:- 4-bit ring counter.

The above figure shows the logic diagram of 4-bit ring counter. As shown in the fig., the Q output of each stage is connected through the D input of the next stage and the output of the last stage is fedback to the i/p of first stage. The $\overline{clr}$ & $\overline{Pre}$ i/p's are used to make the o/p of first stage to 1 and remaining o/p's to zeros (i.e) $Q_A Q_B Q_C Q_D = 1000$.

Truth table:-

Clk	Q_A	Q_B	Q_C	Q_D
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

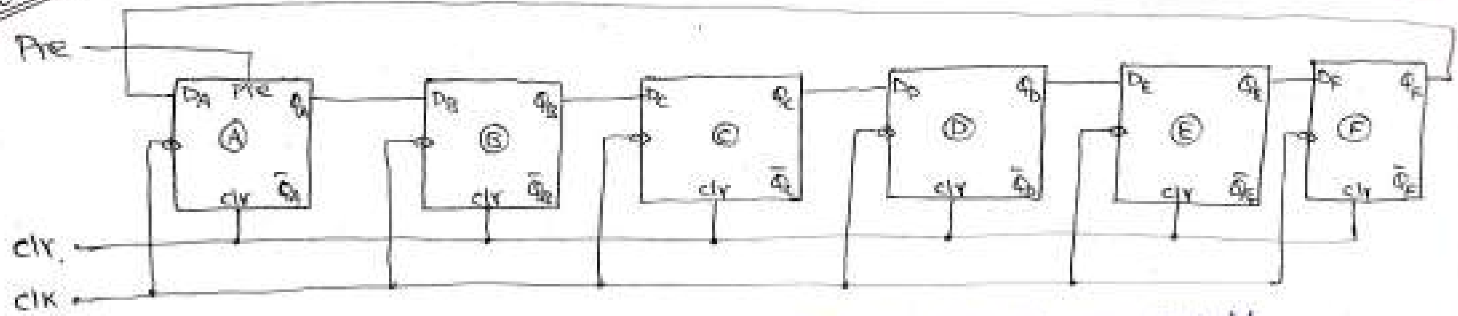
Since the 4-bit ring count has 4 distinct states, it is also known as a MOD-4 counter.

Note:- A MOD-N ring counter will require 'N' no. of flip-flops connected together to circulate a single data bit providing 'N' different o/p states.

Q:- Implement a MOD-6 Ring counter using suitable FF's.

Sol:-

21/10/2014



Pre → preset (or) set i/p

clr → clear (or) Reset i/p

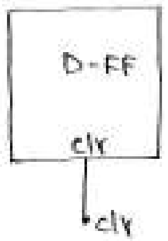
counter o/p → $Q_F\ Q_E\ Q_D\ Q_C\ Q_B\ Q_A$
MSB LSB

Eg: MOD-6 Ring counter

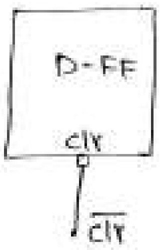
Truth table :-

clk	Q_A	Q_B	Q_C	Q_D	Q_E	Q_F
0	1	0	0	0	0	0
1	0	1	0	0	0	0
2	0	0	1	0	0	0
3	0	0	0	1	0	0
4	0	0	0	0	1	0
5	0	0	0	0	0	1
6	1	0	0	0	0	0

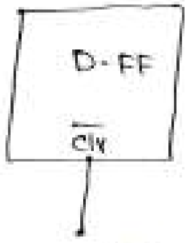
*



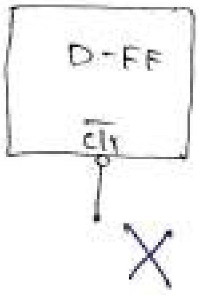
If clr = 1, then FF clears.



If $\overline{clr}$=0 then FF clears.



If $\overline{clr}$=0 then FF clears.



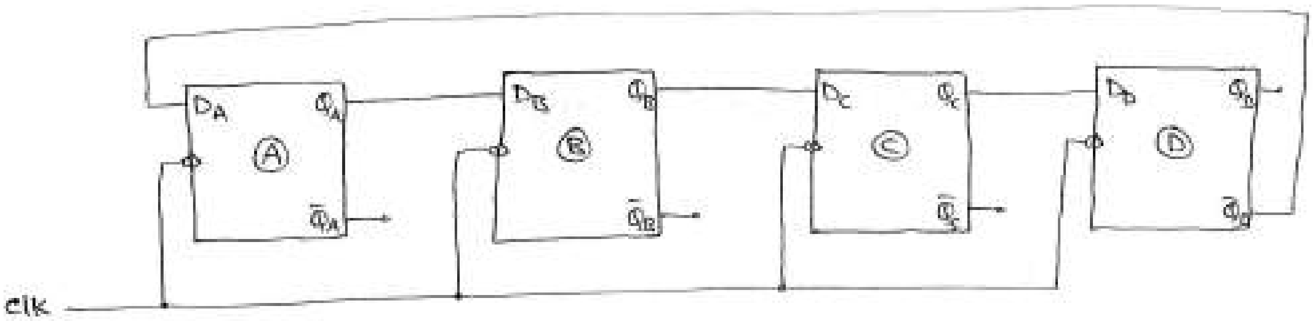
* Johnson counter (or) Twisted ring counter (or) switch tile ring counter :-

(i) The 'n' bit ring counter circulates a single bit among the FF's to provide 'n' different states.

(ii) The no. of states can be doubled if the shift register is connected as a switch tile counter.

(iii) Johnson counters have basic counting cycles of length 2n. where 'n' is the no. of FF's.

(iv). In Johnson counter, the compliment of the o/p of the last FF is connected back to the i/p of the first FF.



counter o/p → Q_D Q_C Q_B Q_A
MSB LSB

Fig:- 4-bit Johnson ring counter.

Truth table (or) State sequence :-

clk	Q_A	Q_B	Q_C	Q_D
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1
8	0	0	0	0

UNIT – 5

Memory and Programmable Logic

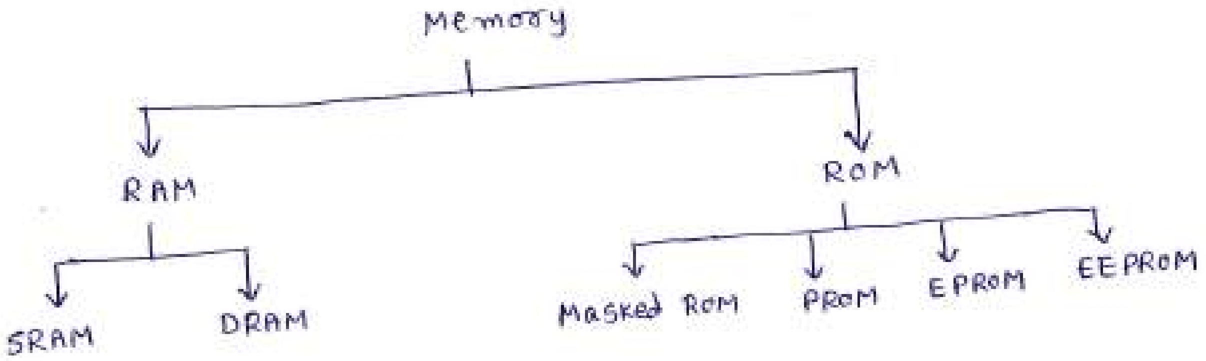
Memory

* Memory is a set of registers which holds instructions and data for processing.

* Memory unit stores instructions and data in Binary form

* There are 2 types of Memories that are used in digital systems. They are RAM, ROM

Classification of Memories



Random Access Memory (RAM)

* RAM is called "Random Access Memory" because any storage location can be accessed directly

* The process of storing new information into Memory is referred to as a Memory write operation

* the process of transferring the stored information out of Memory is referred to as a Memory Read operation

* RAM Performs both Read & write operations. that's why it is called as Read/write Memory

* RAM is volatile Memory i.e if Power is OFF, the stored information will be lost. that's why we store only temporary data in RAM. So, RAM is called as data Memory