

# DIGITAL ELECTRONICS

II B.Tech  
I SEM

## UNIT I :- Number System and Boolean Algebra

Digital systems, Binary numbers, Number Base conversions, octal and Hexadecimal numbers, complements, signed Binary Numbers, Binary codes, Binary storage and Registers, Binary logic, Axiomatic definition of Boolean Algebra, Basic theorems and properties of Boolean Algebra, Boolean Functions, canonical and standard forms, other logic operations, Digital logic Gates, Integrated circuits, Gate-level minimization; The Map method, Four-variable K-Map, Five-variable K-Map, Product of sums simplification, Don't-care conditions, NAND & NOR Implementation, Exclusive-OR functions.

## UNIT-II :- Combinational logic

combinational circuits, Analysis procedure, Design procedure, code-conversion, Binary Adder-Subtractor, carry propagation, Half subtractor, Full subtractor, Binary subtractor, Decimal Adder, BCD adder, Binary Multiplier, Magnitude comparator, Decoders, Encoders and multiplexers.

## UNIT-III :- Sequential logic

Synchronous: sequential circuits, storage elements: latches, introduction to multi-vibrators, flip-flops, Analysis of clocked sequential circuits, state reduction and Assignment, Asynchronous: Analysis procedure, circuits with latches, Design procedure, Reduction of state and flow tables,

## Race-Free State Assignment, Hazards.

### UNIT-IV : Registers and counters

Registers with parallel load, shift registers; serial transfer, Serial Addition, Universal shift Register, Ripple counters; Binary Ripple counter, BCD Ripple counter, Synchronous counters; Binary counter, UP-DOWN counter, BCD counter, Binary counter with parallel load, counter with unused states, Ring counter, Johnson counter.

### UNIT-V : Memory and programmable logic

Random - Access Memory, write and Read operations Timing waveform, Types of Memories, Memory Decoding; Internal construction, coincident Decoding, Address Multiplexing, Error Detection & correction, Hamming code, Single-Error correction, Double-Error Detection, Read only memory; combinational circuit Implementation, Types of ROM's, combinational PLD's, Programmable logic Array, Programmable Array logic, Sequential, Programmable Devices.

UNIT-1

09/07/2014

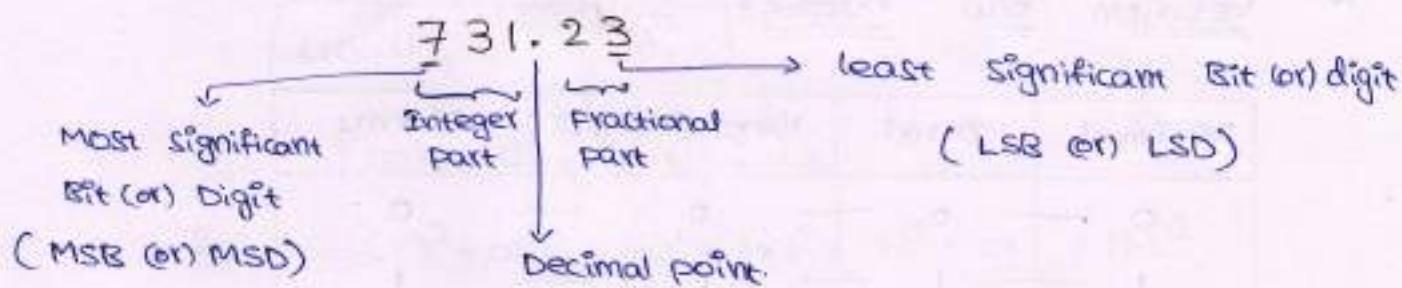
1

# I. NUMBER SYSTEMS AND BOOLEAN ALGEBRA

## 1. Decimal number System:-

0, 1, 2, 3, 4, 5, 6, 7, 8, 9 (10 digits)

Ex: 73<sub>10</sub>, 48.92<sub>10</sub> → Base (or) Radix (or) Index.



$10^2 \ 10^1 \ 10^0 \ 10^{-1} \ 10^{-2}$  → Positional values (or) weights.  
7 3 1 . 2 3

## 2. Binary number System:-

0, 1 (2 digits)

Ex: 1101<sub>2</sub>, 110010<sub>2</sub>

$2^3 \ 2^2 \ 2^1 \ 2^0 \ 2^{-1} \ 2^{-2}$  → Positional values (or) weights  
1 0 1 . 0 1  
Binary Point

## 3. Octal number System:-

0, 1, 2, 3, 4, 5, 6, 7 (8 digits)

Ex: 73<sub>8</sub>, 4072<sub>8</sub>, 123.65<sub>8</sub>

$8^2 \ 8^1 \ 8^0 \ 8^{-1} \ 8^{-2}$  → weights (or) positional values  
1 2 3 . 6 5  
Octal Point

#### 4. Hexa decimal number system:-

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F (16 digits)

Ex:-  $B73_{16}$ ,  $57A.C3_{16}$

$16^2 \ 16^1 \ 16^0 \ 16^1 \ 16^0$  → weights (or) positional values

1    2    A    .    1    3



Hexa decimal point.

#### \* Relation b/w number systems:-

Decimal	Octal	Hexadecimal	Binary
0	0	0	0
1	1	1	1
2	2	2	10
3	3	3	11
4	4	4	100
5	5	5	101
6	6	6	110
7	7	7	111
8	10	8	1000
9	11	9	1001
10	12	A	1010
11	13	B	1011
12	14	C	1100
13	15	D	1101
14	16	E	1110
15	17	F	1111
16	20	10	10000
17	21	11	10001

## Number Base conversion :-

### 1. Binary to Decimal conversion :-

Ex:- ①  $(101.11)_2 = (?)_{10}$

Sol:- 
$$\begin{array}{r} \frac{1}{2^2} \frac{0}{2^1} \frac{1}{2^0} \frac{1}{2^{-1}} \frac{1}{2^{-2}} \\ 1 \ 0 \ 1 \cdot 1 \ 1 \end{array}$$

$$= 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2}$$

$$= 4 + 0 + 1 + 0.5 + 0.25 = 5.75$$

$(101.11)_2 = (5.75)_{10}$

②  $(1011.101)_2 = (?)_{10}$

Sol:- 
$$\begin{aligned} & 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} \\ & = 8 + 2 + 1 + \frac{1}{2} + \frac{1}{8} \\ & = (11.625)_{10} \end{aligned}$$

$(1011.101)_2 = (11.625)_{10}$

### 2. Octal to Decimal conversion :-

Ex:- ①  $(75.3)_8 = (?)_{10}$

Sol:- 
$$\begin{aligned} & = 7 \times 8^1 + 5 \times 8^0 + 3 \times 8^{-1} \\ & = 56 + 5 + \frac{3}{8} = (61.375)_{10} \end{aligned}$$

$(75.3)_8 = (61.375)_{10}$

②  $(624.712)_8 = (?)_{10}$

Sol:- 
$$\begin{aligned} & 6 \times 8^2 + 2 \times 8^1 + 4 \times 8^0 + 7 \times 8^{-1} + 1 \times 8^{-2} + 2 \times 8^{-3} \\ & = 6 \times 64 + 16 + 4 + \frac{7}{8} + \frac{1}{64} + \frac{2}{64 \times 8} \\ & = 384 + 20 + 0.875 + 0.015625 + 0.00390625 \\ & = (404.894)_{10} \end{aligned}$$

$(624.712)_8 = (404.894)_{10}$

$$\underline{\text{Ex:}} \quad \textcircled{3} \quad (482.31)_8 = (?)_{10}$$

It is not an octal number.  
So, conversion is not possible.

### (3) Hexadecimal to decimal conversion :-

$$\underline{\text{Ex:}} \quad \textcircled{1} \quad (7A2.C9)_{16} = (?)_{10}$$

$$\begin{aligned}\underline{\text{Sol:}} \quad & 7 \times 16^2 + A \times 16^1 + 2 \times 16^0 + C \times 16^{-1} + 9 \times 16^{-2} \\ & = 7 \times 256 + 10 \times 16 + 2 \times 1 + \frac{12}{16} + \frac{9}{256} \\ & = 1792 + 160 + 2 + \frac{3}{4} + 0.3515625 \\ & = (1954.78)_{10}\end{aligned}$$

$$(7A2.C9)_{16} = (1954.78)_{10}$$

$$\textcircled{2} \quad (CD3.B7)_{16} = (?)_{10}$$

$$\begin{aligned}\underline{\text{Sol:}} \quad & C \times 16^2 + D \times 16^1 + 3 \times 16^0 + B \times 16^{-1} + 7 \times 16^{-2} \\ & = 12 \times 256 + 13 \times 16 + 3 \times \frac{11}{16} + \frac{7}{256} \\ & = (3283.715)_{10}\end{aligned}$$

$$(CD3.B7)_{16} = (3283.715)_{10}$$

### (4) Base 5 no. to decimal conversion:-

$$\underline{\text{Ex:}} \quad \textcircled{1} \quad (431.23)_5 = (116.52)_{10}$$

$$\begin{aligned}\underline{\text{Sol:}} \quad & 4 \times 5^2 + 3 \times 5^1 + 1 \times 5^0 + 2 \times 5^{-1} + 3 \times 5^{-2} \\ & = 100 + 15 + 1 + \frac{2}{5} + \frac{3}{25} \\ & = (116.52)_{10}\end{aligned}$$

$$(431.23)_5 = (116.52)_{10}$$

$$\textcircled{5} \quad (231.12)_3 = (?)_{10}$$

Sol. It is not a Base 3 no.

Base 3 no. digits are 0, 1, 2.

\textcircled{6} Decimal to Binary conversion :-

$$\text{Ex:- } \textcircled{1} \quad (13.25)_{10} = (?)_2$$

Sol. Integer part conversion      Fractional part conversion

$$\begin{array}{r} 13 \\ 2 \overline{) 6 - 1} \\ 2 \overline{) 3 - 0} \\ 2 \overline{) 1 - 1} \\ 0 - 1 \end{array}$$

↑  
all remainders  
from bottom  
to top

$$0.25 \times 2 = 0.5$$

$$0.5 \times 2 = 1.0$$



Integer

parts of all  
product terms  
from top to  
bottom.

$$0.25_{10} = 0.01_2$$

$$(13.25)_{10} = (1101.01)_2$$

$$\textcircled{2} \quad (5.6)_{10} = (?)_2$$

Sol: I.P.C.

$$\begin{array}{r} 5 \\ 2 \overline{) 2 - 1} \\ 2 \overline{) 0 - 0} \\ 0 - 1 \end{array}$$

↑

$$(5)_{10} = (101)_2$$

F.P.C

$$0.6 \times 2 = 1.2$$

$$0.2 \times 2 = 0.4$$

$$0.4 \times 2 = 0.8$$

$$0.8 \times 2 = 1.6$$

$$0.6 \times 2 = 1.2$$

$$0.2 \times 2 = 0.4$$

$$0.4 \times 2 = 0.8$$

$$0.8 \times 2 = 1.6$$

↓  
 $0.\overline{1001}$

$$(5.6)_{10} = (101.\overline{1001})_2$$

Note :- If there is no repetition in that case the process of multiplication is to be stopped after 4 or 5 places.

10/07/2014

### 7. Decimal to Octal conversion :-

Ex:- (i)  $(73.625)_{10} = (?)_8$

Sol:- Integer conversion

$$\begin{array}{r} 73 \\ 8 \overline{) 9 - 1} \\ 8 \overline{) 1 - 1} \\ 0 - 1 \end{array}$$

↑  
↑

$$(73)_{10} = (111)_8$$

Fractional part conversion

$$0.625 \times 8 = 5.000 \quad \downarrow$$

$$(0.625)_{10} = (0.5)_8$$

$$\therefore (73.625)_{10} = (111.5)_8$$

(ii)  $(296.198)_{10} = (?)_8$

Sol:- Integer part conversion

$$\begin{array}{r} 296 \\ 8 \overline{) 37 - 0} \\ 8 \overline{) 4 - 5} \\ 0 - 4 \end{array}$$

↑  
↑

$$(296)_{10} = (450)_8$$

Fractional part conversion

$$0.198 \times 8 = 1.584$$

$$0.584 \times 8 = 4.672$$

$$0.672 \times 8 = 5.392$$

$$0.392 \times 8 = 3.008$$

$$0.008 \times 8 = 0.064$$

$$0.064 \times 8 = 0.512$$

$$(0.198)_{10} = (0.145300)_8$$

$$(296.198)_{10} = (450.145300)_8$$

### 8. Decimal to Hexa Decimal :-

Ex:- (i)  $(1954.785)_{10} = (?)_{16}$

Sol:- Integer part conversion

$$\begin{array}{r} 1954 \\ 16 \overline{) 122 - 2} \\ 16 \overline{) 7 - 10A} \\ 0 - 7 \end{array}$$

↑  
↑

$$(1954)_{10} = (7A2)_{16}$$

Fractional Part conversion

$$0.785 \times 16 = 12.56 \quad C$$

$$0.56 \times 16 = 8.96 \quad 8$$

$$0.96 \times 16 = 15.36 \quad F$$

$$0.36 \times 16 = 5.76 \quad 5$$

$$0.76 \times 16 = 12.16 \quad C$$

$$(0.785)_{10} = (0.C8F5C)_{16}$$

$$\therefore (1954.785)_{10} = (7A2.C8F5C)_{16}$$

(ii)  $(3283.715)_{10} = (?)_{16}$

Sol: Integer part conversion

$$\begin{array}{r} 16 \mid 3283 \\ 16 \boxed{205} - 3 \\ 16 \boxed{12} - \textcircled{13} \text{ D} \\ 0 - \textcircled{12} \text{ C} \end{array}$$

$$(3283)_{10} = (CD3)_{16}$$

Fractional part conversion

$$\begin{array}{l} 0.715 \times 16 = 11.44 \\ 0.44 \times 16 = 7.04 \\ 0.04 \times 16 = 0.64 \\ 0.64 \times 16 = 10.24 \end{array}$$

B  
7  
0  
A

$$(0.715)_{10} = (0.B70A)_{16}$$

$$\therefore (3283.715)_{10} = (CD3.B70A)_{16}$$

9. Base 3 to Base 5 :-

Ex:- (i)  $(21.1)_3 = (?)_5$

Sol:- Base 3  $\rightarrow$  decimal  $\rightarrow$  Base 5

$$2 \times 3^1 + 1 \times 3^0 + 1 \times 3^{-1} = 7.333$$

$$\therefore (21.1)_3 = (7.333)_{10}$$

$$(7.333)_{10} = (?)_5$$

$$\begin{array}{r} 5 \mid 7 \\ 5 \boxed{1-2} \uparrow \\ 0-1 \end{array}$$

$$(7)_{10} = (12)_5$$

$$\begin{array}{l} 0.333 \times 5 = 1.665 \\ 1.665 \times 5 = 3.325 \\ 0.325 \times 5 = 1.625 \\ 0.625 \times 5 = 3.125 \\ 0.125 \times 5 = 0.725 \\ 0.725 \times 5 = 3.625 \end{array}$$

$$(0.333)_{10} = (0.131\overline{30})_5$$

$$\therefore (7.333)_{10} = (12.131\overline{30})_5 = (21.1)_3$$

10. Octal to Binary conversion :-

Ex:- (i)  $(73.2)_8 = (?)_2$

Sol:-

7	3	.	2
↓	↓	↓	
111	011		010

$$\therefore (73.2)_8 = (111011.010)_2$$

$$(ii) (23.46)_8 = (?)_2$$

Sol:-

2	3	.	4	6
↓	↓		↓	↓
010	011		100	110

$$\therefore (23.46)_8 = (10011.10011)_2$$

### 11. Hexadecimal to Binary conversion :-

Ex:- (i)  $(7A.2C)_{16} = (?)_2$

Sol:-

7	A	.	2	C
↓	↓		↓	↓
0111	1010		0010	1100

$$(7A.2C)_{16} = (1111010.00101100)_2$$

(ii)  $(D2.E9)_{16} = (?)_2$

Sol:-

D	2	.	E	9
↓	↓		↓	↓
1101	0010		1110	1001

$$(D2.E9)_{16} = (11010010.11101001)_2$$

### 12. Binary to octal conversion :-

16 8 4 2 1

Ex:- (i)  $(10111.101)_2 = (?)_8$

Sol:-  $\underbrace{1}_{2} \underbrace{0}_{2} \underbrace{1}_{2} \underbrace{1}_{2} \underbrace{1}_{2} \underbrace{.}_{3} \underbrace{1}_{4} \underbrace{0}_{4} = (27.5)_8$

$$\therefore (10111.101)_2 = (27.5)_8$$

(ii)  $(1011.1)_2 = (?)_8$

Sol:-  $\underbrace{0}_{1} \underbrace{0}_{2} \underbrace{1}_{3} \underbrace{0}_{3} \underbrace{1}_{4} \underbrace{1}_{4} = (13.4)_8$

$$\therefore (011011.100)_2 = (13.4)_8$$

13. Binary to Hexadecimal :-

Ex: (i)  $(01110111.1010)_2 = (?)_{16}$

Sol:  $\underbrace{0111}_{7} \underbrace{0111}_{7} \cdot \underbrace{1010}_{A} = (77.A)_{16}$

$\therefore (01110111.1010)_2 = (77.A)_{16}$

(ii)  $(10111.1000)_2 = (?)_{16}$

Sol:  $\underbrace{0001}_{1} \underbrace{0111}_{7} \cdot \underbrace{1000}_{8} = (17.8)_{16}$

$\therefore (10111.1)_2 = (17.8)_{16}$

14. Octal to Hexadecimal :-

Ex: (i)  $(35.7)_8 = (?)_{16}$

Sol: Octal  $\rightarrow$  Binary  $\rightarrow$  Hexadecimal.

$$\begin{array}{ccc} 3 & 5 & 7 \\ \downarrow & \downarrow & \downarrow \\ 011 & 101 & 111 \end{array}$$

$(35.7)_8 = (\underbrace{011101}_{1D} \cdot \underbrace{111}_{E})_2$

$\therefore (35.7)_8 = (1D.E)_{16}$

(ii)  $(73.2)_8 = (?)_{16}$

Sol:  $\begin{array}{ccc} 7 & 3 & 2 \\ 111 & 011 & 010 \end{array}$

$(73.2)_8 = (\underbrace{111}_{3} \underbrace{011}_{B} \cdot \underbrace{010}_{4})_2$

$\therefore (73.2)_8 = (3B.4)_{16}$

15. Hexadecimal to octal :-

Hexadecimal  $\rightarrow$  Binary  $\rightarrow$  octal

Ex:- ①  $(1D.E)_{16} = (?)_8$ .

Soln:-  
1 D . E  
0001 1101 1110

$$\begin{aligned}\therefore (1D.E)_{16} &= ( \underbrace{0111}_{3} \underbrace{01}_{5} \underbrace{111}_{7} )_2 \\ &= (35.7)_8\end{aligned}$$

$$\therefore \boxed{(1D.E)_{16} = (35.7)_8}$$

②  $(3B.4)_{16} = (?)_8$

Soln:-  
3 B . 4  
0011 1011 0100  
 $= (\underbrace{111}_{7} \underbrace{011}_{3} \underbrace{010}_{2})_2$

$$\therefore \boxed{(3B.4)_{16} = (73.2)_8}$$

16/07/2014

### \* Binary addition

#### Rules

	Sum	Carry	Result
0+0	0	0	0
0+1	1	0	1
1+0	1	0	1
1+1	0	1	10

$+1$   
10  
 $+1$   
11  
 $+1$   
100  
 $+1$   
101

Ex:- (i) perform  $1011_2 + 1111_2$

Soln:-  
$$\begin{array}{r} 1011 \\ + 1111 \\ \hline \end{array}$$
  
Carry Sum

$$1011_2 + 1111_2 = 11010_2$$

(ii) perform  $101_2 + 110_2 + 111_2$ 

$$\begin{array}{r} *1 \ 0 \ 1 \ 1 \\ & 1 \ 1 \ 0 \ 1 \\ + & 1 \ 1 \ 1 \ 0 \\ \hline 1 0 0 \ 1 \ 1 \ 0 \end{array}$$

Carry      Sum

$$\therefore 101_2 + 110_2 + 111_2 = 100110_2$$

(iii)  $1001 \cdot 1_2 + 1101 \cdot 01_2$ 

$$\begin{array}{r} 1 \ 0 \ 0 \ 1 \cdot 1 \ 0 \\ 1 \ 1 \ 0 \ 1 \cdot 0 \ 1 \\ \hline 1 \ 0 \ 1 \ 1 \ 0 \cdot 1 \ 1 \\ \text{Carry} \quad \text{Sum} \end{array}$$

$$\therefore 1001 \cdot 1_2 + 1101 \cdot 01_2 = 10110 \cdot 1_2$$

### \* Binary multiplication :-

Ex: (i)  $101_2 \times 111_2$ 

$$\begin{array}{r} 101 \\ \times 111 \\ \hline 101 \\ 101 \\ \hline 100011 \end{array}$$

$$\therefore 101_2 \times 111_2 = 100011_2$$

(ii)  $1010 \cdot 1_2 \times 1011 \cdot 1_2$ 

$$\begin{array}{r} 1010 \cdot 1 \times 1011 \cdot 1 \\ \hline 10101 \\ 10101 \\ 10101 \\ 00000 \\ \hline 10101 \\ \hline 111100011 \end{array}$$

$$\therefore 1010 \cdot 1_2 \times 1011 \cdot 1_2 = 111100011$$

### \* Binary division :-

Ex: (i)  $111101_2 \div 100_2$ 

$$\begin{array}{r} 100 ) 111101 ( 1111 \\ 100 \\ \hline 111 \\ 100 \\ \hline 110 \\ 100 \\ \hline 101 \\ 100 \\ \hline 1 \end{array}$$

$$\therefore 111101_2 \div 100_2 = 1111 \cdot 1$$

Ex: (ii)  $1010 \cdot 1 \div 101 \cdot 01$ 

$$\begin{array}{r} 1010 \cdot 1 \\ \hline 101 \cdot 01 \\ \hline 10101 \\ \hline 10101 \\ \hline 10 \end{array}$$

$$= 10$$

## \* Binary subtraction & complements :-

Note:-

$A - B = A + \text{X}'\text{s complement}$  of  $B$  (Subtraction using radix complement)

$A - B = A + (\text{X}-1)\text{'s complement}$  of  $B + 1$  (Subtraction using diminished radix complement)

$\text{X} \rightarrow$  radix (or) Base of  $A + B$  no's

If  $A + B$  are Binary no.'s then

$A - B = A + 1\text{'s comp. of } B + 1$  where

$= A + 2\text{'s comp. of } B.$

$A \rightarrow$  Minuend

$B \rightarrow$  Subtrahend

### 1's complement:-

1's complement of any number can be obtained by changing each '0' to '1' and '1' to '0'.

∴ 1's complement of 1101001 is 0010110.

### 1's complement subtraction:-

#### Rules:-

- (i) Add minuend and 1's complement of subtrahend.
- (ii) If there is carry after addition, then the result is '+ve'. else result is '+ve'. else result is '-ve'
- (iii) If result is '+ve', add carry to sum to get the actual value.
- (iv) If result is '-ve' take 1's complement of sum to get the actual value.

Ex:- (i) Perform unsigned binary subtraction using 1's complement.

(i)  $1101_2 - 1001_2$

Sol:-  $1101 \rightarrow$  minuend

$1001 \rightarrow$  Subtrahend

$0110 \rightarrow$  1's complement of Subtrahend.

$$\begin{array}{r}
 1101 \rightarrow \text{minuend} \\
 0110 \rightarrow \text{i's comp. of sub} \\
 \hline
 \begin{array}{c}
 10011 \\
 \text{carry sum}
 \end{array}
 \end{array}$$

$\therefore$  there is a carry.  $\therefore$  Result is '+ve'.

To get the actual value (difference) add carry to sum

$$\begin{array}{r}
 0011 \\
 + 1 \\
 \hline
 0100 \rightarrow \text{difference}
 \end{array}$$

$$\therefore 1101_2 - 1001_2 = 0100_2$$

$$(ii) 1001_2 - 1101_2$$

Sol:  $1001 \rightarrow \text{minuend}$   
 $1101 \rightarrow \text{Subtrahend}$ .

$$\begin{array}{r}
 1001 \rightarrow \text{minuend} \\
 0010 \rightarrow \text{i's complement of Subtrahend} \\
 \hline
 \begin{array}{c}
 1011 \\
 \text{sum}
 \end{array}
 \end{array}$$

there is no carry.  $\therefore$  Result is '-ve'.

To get the actual value take i's complement of sum.

i's complement of 1011 is 0100.

$$\therefore 1001_2 - 1101_2 = -0100_2$$

2's complement:-

2's complement of A = i's complement of A + 1

Ex: Find 2's complement of 1010.

Sol:  $= \text{i's complement of } 1010 + 1$

$$= 0101 + 1$$

$$= \underline{\underline{0110}}$$

$$\begin{array}{r}
 0101 \\
 + 1 \\
 \hline
 0110
 \end{array}$$

$$\therefore 2\text{'s complement of } 1010 = 0110$$

2's comp of 101100 is 010100 (write no. as it is from right to left until u found 1. if u found 1 then write that first 1 as it is. then change the remaining nos from 0's to 1's and 1's to 0's)

## 2's complement subtraction rules :-

- (i) Add minuend and 2's complement of subtractend.
- (ii) After addition if there is carry then the result is '+ve', else '-ve'.
- (iii) If the result is '+ve' to get the actual value discard the carry.
- (iv) If the result is '-ve' to get the actual value take 2's complement of sum.

Ex: i. perform unsigned binary subtraction using 2's complement.

$$\& \text{ (i) } 1101_2 - 1010_2$$

Sol:  $1101 \rightarrow$  minuend  
 $1010 \rightarrow$  subtractend.

$$\begin{array}{r} 0101 \\ + 110 \\ \hline 0110 \end{array}$$

$$2\text{'s comp. of subtractend (1010)} = 0101 + 1 = 0110.$$

$$\begin{array}{r} 1101 \rightarrow \text{minuend} \\ 0110 \rightarrow 2\text{'s comp. of sub.} \\ \hline \begin{array}{c} 10011 \\ \text{~~~~~} \\ \text{carry sum} \end{array} \end{array}$$

$\therefore$  It has carry.  $\therefore$  Result is '+ve'.

$\therefore$  To get actual value discard the carry.

$$= 0011_2$$

$$\therefore 1101_2 - 1010_2 = 0011_2$$

$$\text{(ii) } 1010_2 - 1101_2$$

Sol:  $1010 \rightarrow$  min.  
 $1101 \rightarrow$  sub.

$$\begin{array}{r} 0011 \rightarrow 2\text{'s comp. of sub.} \\ 1010 \rightarrow \text{min} \\ \hline \begin{array}{c} 1101 \\ \text{~~~~~} \\ \text{sum.} \end{array} \end{array}$$

$\therefore$  It has no carry.  $\therefore$  Result is '-ve'.

To get actual value take 2's comp. of sum.

$$2\text{'s comp. of } 1101 = 0010 + 1 = 0011_2$$

$$\therefore 1010_2 - 1101_2 = -0011_2$$

### 9's complement:-

a's complement of a decimal number can be obtained by the subtracting each digit from 9.

Ex: a's comp. of  $73_{10}$  is

$$\begin{array}{r} 99 \\ - 73 \\ \hline 26 \end{array}$$

$\therefore$  9's comp. of  $73_{10}$  is 26

### 9's complement subtraction:-

Rules same as 1's complement but instead of 1's complement take 9's complement.

Ex: (i)  $73_{10} - 26_{10}$

Sol:  $73 \rightarrow$  min

$26 \rightarrow$  sub.

$73 \rightarrow$  min.

$73 \rightarrow$  9's comp. of sub.

$$\begin{array}{r} 146 \\ \hline \text{min} \\ \text{comp. sum} \end{array}$$

$$\begin{array}{r} 99 \\ - 26 \\ \hline 73 \end{array}$$

$\therefore$  It has carry.  $\therefore$  Result is '+ve'.

To get actual value add carry to sum.

$$\begin{array}{r} 46 \\ - 1 \\ \hline 47_{10} \end{array}$$

$$\therefore 96_{10} - 26_{10} \therefore 73_{10} - 26_{10} = 47_{10}$$

(ii)  $26_{10} - 73_{10}$

Sol:  $26 \rightarrow$  min.

$73 \rightarrow$  sub.

$$\begin{array}{r} 99 \\ - 73 \\ \hline 26 \end{array} \rightarrow$$
 9's comp. of sub.

$$\begin{array}{r} 26 \\ - 26 \\ \hline 52 \end{array}$$

It has no carry. Result is 've'.

To get actual values Take 9's comp. of sum.

$$\begin{array}{r} 99 \\ - 52 \\ \hline 47 \end{array} \rightarrow 9\text{'s comp. of } 52.$$

$$\therefore 26_{10} - 73_{10} = -47_{10}$$

### 10's complement :-

10's comp. of A = 9's comp. of A + 1

Ex: Find 10's comp. of  $23_{10}$ .

Sol: 10's comp. of  $23 = 9\text{'s comp. of } 23 + 1$

$$\begin{array}{r} 99 \\ - 23 \\ \hline 76 \end{array}$$
  
$$= 76 + 1$$
  
$$= 77$$

$$\therefore 10\text{'s comp. of } 23 = 77$$

### 10's complement subtraction :-

Rules same as 2's subtraction complement  
but take 10's complement instead of 2's comp.

Ex:  $45_{10} - 16_{10}$

Sol:  $45 \rightarrow \text{min.}$

$16 \rightarrow \text{Sub.}$

$$\begin{array}{r} 99 \\ - 16 \\ \hline 83 \end{array}$$
  
$$10\text{'s comp. of } 16 = 9\text{'s comp. of } 16 + 1$$
  
$$= 83 + 1 = 84$$

$\therefore 84 \rightarrow 10\text{'s comp. of sub.}$

$45 \rightarrow \text{min.}$

129  
min  
Carry Sum

It has carry.  $\therefore$  Result is 've'.  
To get actual value discard the carry.

$$\therefore 45_{10} - 16_{10} = 29_{10}$$

Ex: (ii)  $16_{10} - 45_{10}$

Sol:-  $16 \rightarrow$  min.  
 $45 \rightarrow$  sub.

$$\begin{array}{r} 10\text{'s comp. of } 45 = 9\text{'s comp. of } 45 + 1 \\ = 54 + 1 = 55. \end{array} \quad \begin{array}{r} 99 \\ - 45 \\ \hline 54 \end{array}$$

$$55 \rightarrow 10\text{'s comp. of } 45$$

$16 \rightarrow$  min

71  
Sum

It has no carry.  $\therefore$  Result is '-ve'.

To get actual value take 10's comp. of sum.

$$\begin{array}{r} 10\text{'s comp. of } 71 \text{ is } = 9\text{'s comp. of } 71 + 1 \\ = 28 + 1 = 29 \end{array} \quad \begin{array}{r} 99 \\ - 71 \\ \hline 28 \end{array}$$

$$\therefore 16_{10} - 45_{10} = -29_{10}$$

### Octal subtraction :-

If A & B are octal no.'s then

$$\begin{aligned} A - B &= A + 8\text{'s comp. of } B \\ &= A + 7\text{'s comp. of } B + 1 \end{aligned}$$

$$7\text{'s comp. of } 23_8 \text{ is } \begin{array}{r} 77 \\ - 23 \\ \hline 54 \end{array}$$

### Hexadecimal subtraction :-

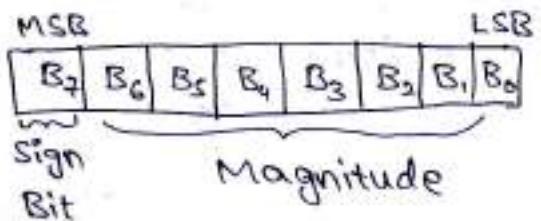
If A & B are octal no.'s then,

$$\begin{aligned} A - B &= A + 16\text{'s comp. of } B \\ &= A + 15\text{'s comp. of } B + 1 \end{aligned}$$

15's comp. of  $A2_{16}$  is

$$\begin{array}{r} FF \\ - A2 \\ \hline 5D \end{array}$$

1/03/2014  
Signed Binary numbers :-



sign bit = 0  $\rightarrow$  +ve binary number

sign bit = 1  $\rightarrow$  -ve binary number.

Ex:- Represent  $-5_{10}$ ,  $-9_{10}$ ,  $+17_{10}$ . in signed magnitude form.

Sol:-

$$5 \rightarrow 101$$

$$-5_{10} = (\underline{1} \underline{0} \underline{1})_{\substack{\text{Signed} \\ \text{binary}}}$$

( $\because$  add '1' in MSB place)

$$9 \rightarrow 1001$$

$$-9_{10} = (\underline{1} \underline{0} \underline{0} \underline{1})_{\substack{\text{Signed} \\ \text{binary}}}$$

$$+17_{10} = \underline{0} \underline{1} \underline{0} \underline{0} \underline{1}$$

$$+9_{10} = (\underline{0} \underline{1} \underline{0} \underline{0} \underline{1})_{\substack{\text{Signed} \\ \text{binary}}}.$$

Ex:- Represent  $-5_{10}$ ,  $-9_{10}$ ,  $+17_{10}$  in 8 bit signed magnitude form.

Sol:-

$$-5_{10} = (\underline{1} \underline{0} \underline{0} \underline{0} \underline{0} \underline{1} \underline{1} \underline{0} \underline{1})$$

$$-5_{10} = (\underline{1} \underline{0} \underline{0} \underline{0} \underline{0} \underline{1} \underline{0} \underline{1})_{\substack{\text{Signed} \\ \text{Binary (2)}}}$$

$$-9_{10} = (\underline{1} \underline{0} \underline{0} \underline{0} \underline{1} \underline{0} \underline{0} \underline{1})_{\substack{\text{Signed} \\ \text{binary}}}$$

$$+17_{10} = (\underline{0} \underline{0} \underline{0} \underline{1} \underline{0} \underline{0} \underline{0} \underline{1})_{\substack{\text{Signed} \\ \text{binary}}}.$$

Ex:- Represent  $+7$ ,  $-7$  in  $8\text{bit}$  signed binary, Signed is complement, signed 2's complement form.

Sol: (i)  $+7 = 0111$  (signed binary representation).

$+7 = 0111$  (signed is complement " )

$+7 = 0111$  (signed 2's complement " )

$+7 = 00000111$  (8-bit signed binary " )

$+7 = 00000111$  (signed is comp. " )

$+7 = 00000111$  (signed 2's " " ).

(ii)  $-7 = 1111$  (signed magnitude Representation)

$-7 = 1000$  (signed i's comp. " ) ( $+7$ :complement)

$-7 = 1001$  (" 2's " " ) ( $+2$ :2's complement  
= i's comp + 1)

$-7 = 10000111$  (8-Bit Signed magnitude rep)

$-7 = 11111000$  (signed i's comp. rep)

$-7 = 11111001$  (" 2's comp. " )

Ex:- In signed is complement ' $-7$ ' is obtained by complementing all the bits of  $+7$ , including the sign bit.

In signed 2's complement ' $-7$ ' is obtained by taking 2's complement of  $+7$ .

\* Arthametic Addition :-

Ex:- Perform the following using 8-Bit Binary arthametic.

$$(i) (+6) + (+13) \quad (ii) (-6) + (+13) \quad (iii) (+6) + (-13)$$

$$(iv) (-6) + (-13).$$

(i) sol:-

$$\begin{array}{r} +6 \rightarrow 00000110 \\ +13 \rightarrow 00001101 \end{array}$$

$$\begin{array}{r} 00000110 \\ 00001101 \\ \hline \underbrace{0010011}_{\text{Signed } 19} \end{array}$$

Signed bit is '0'.

∴ Result is 'tve' number.

$$(+6) + (+13) \Rightarrow +19$$

(ii) sol:-

$$\begin{array}{r} +6 \rightarrow 00000110 \\ (-6 \rightarrow 10000110) \times \end{array}$$

Represent 've' no's in signed 2's comp. form

$$-6 = (\text{2's comp. of } +6)$$

$$-6 = 11111010$$

$$\begin{array}{r} +13 = 00001101 \\ -1111 \\ \hline \underbrace{100000111}_{\text{Carry } 7} \end{array}$$

Sign bit = 0, ∴ Result is 'tve' no.

If carry is generated, discard that carry.

$$(-6) + (+13) = +7.$$

$$(iii) \quad +6 \rightarrow 00000110$$

$$-13 \rightarrow 11110011 \quad (\text{2's comp. of } -13)$$

$$\begin{array}{r} 00000110 \\ 11110011 \\ \hline 11111001 \end{array}$$

Sign bit

Sign bit = 1,  $\therefore$  Result is 've' no.

've' results are already in 2's comp. form.

To get result take 2's comp. of sum including sign bit.

$$2\text{'s comp. of } 11111001 \text{ is } \underbrace{00000111}_7$$

$$\therefore (+6) + (-13) = -7$$

$$(iv) \quad -6 \rightarrow 11111010$$

$$-13 \rightarrow 11110011$$

$$\begin{array}{r} 11111010 \quad (\text{2's comp of } -6) \\ 11110011 \quad (\text{2's comp of } -13) \\ \hline 111101101 \end{array}$$

carry  
Sign bit

carry is discard

Sign bit = 1  $\therefore$  Result is 've' no.

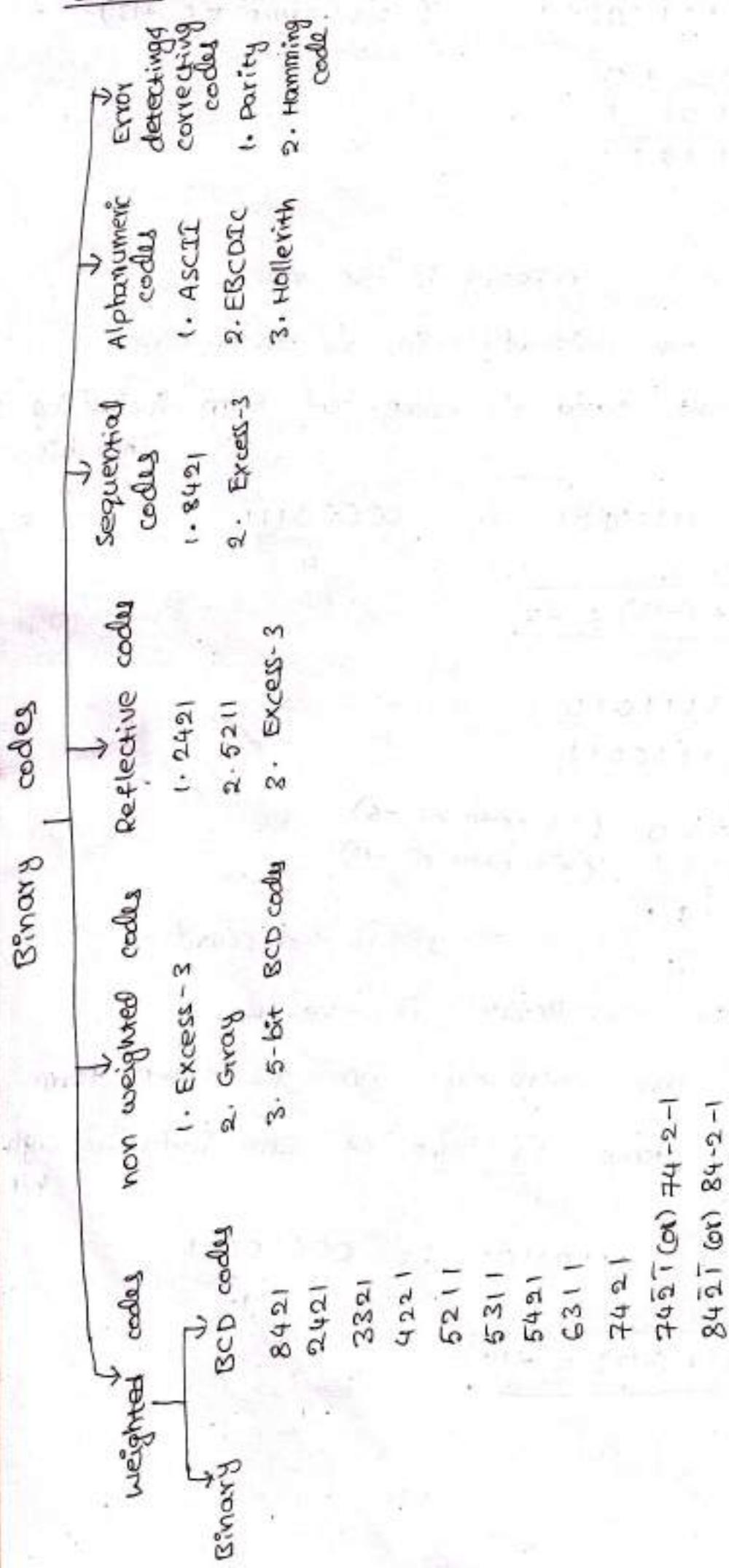
've' results are already in 2's comp. form.

To get result take 2's comp. of sum including sign bit.

$$2\text{'s comp. of } 111101101 \text{ is } \underbrace{00010011}_{19}$$

$$\therefore (-6) + (-13) = -19$$

## Binary codes :-



Error detecting & correcting codes

Alphanumeric codes

Sequential codes

Reflective codes

non weighted codes

Weighted codes

BCD codes

1. Excess-3  
2. Gray  
3. 5-bit BCD codes

8421  
2421

3321

4221

5211

5311

5421

6311

7421

7451 (OR) 74-2-1

8421 (OR) 84-2-1

1. ASCII  
2. EBCDIC

1. Parity  
2. Hamming code

1. Hollerith  
2. EBCDIC

1. Parity  
2. Hamming code

1. ASCII  
2. EBCDIC

1. Parity  
2. Hamming code

1. Hollerith  
2. EBCDIC

1. Parity  
2. Hamming code

## Weighted codes :-

In weighted codes each digit position of the number representing a specific weight.

### 1. Straight Binary code

$\begin{array}{ccccccc} & 4 & 3 & 2 & 1 & 0 \\ \dots & 2 & 2 & 2 & 2 & 2 \\ \hline \text{MSB} & & & & & \text{LSB} \end{array}$

↳ depends upon no. of digits.

### 2. Binary coded decimal (BCD)

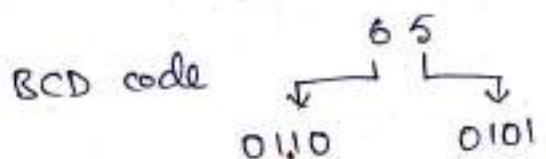
BCD is a numeric code in which each digit of a decimal number is represented by a separate group of bits.

The most common BCD code is 8421.

<u>Decimal</u>	<u>BCD codes</u>		
8421	4221	7421	0011
0	0000	0000	0000
1	0001	0001	0101
2	0010	0100 0010	0110
3	0011	0011 000101	0101
4	0100	1000 000110	1011
5	0101	1001 000111	1010
6	0110	1100 0011010	1001
7	0111	1101 0011011	1000
8	1000	1110	1100
9	1001	1111	1110

Ex:- Represent  $65_{10}$  in straight binary & BCD code

Sol:- S.B  $65_{10} = (10000001)_2$



$$65_{10} = (01100101)_{\text{BCD}} \quad (8421)$$

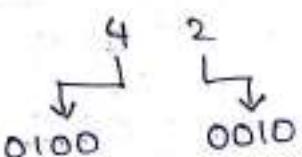
$$\begin{array}{r} 65 \\ 2 | \\ 32 - 1 \\ 2 | 16 - 0 \\ 2 | 8 - 0 \quad \uparrow \\ 2 | 4 - 0 \\ 2 | 2 - 0 \\ 2 | 1 - 0 \\ 0 - 1 \end{array}$$

Ex:- ① convert  $2A_{16}$  to BCD code.

Sol:- Hexa decimal  $\rightarrow$  decimal  $\rightarrow$  BCD.

$$\begin{aligned} &= 2 \times 16^1 + A \times 16^0 \\ &= 32 + 10 = 42 \end{aligned}$$

$$(2A)_{16} = (42)_{10}$$



$$(2A)_{16} = (42)_{10} = (01000010)_{\text{BCD}} \quad 8421$$

② convert  $(01000010)_{\text{BCD}} = (?)_{16}$

BCD  $\rightarrow$  decimal  $\rightarrow$  Hexadecimal

Sol:-  $\underline{\underline{01000010}}$

$$(01000010)_{\text{BCD}} = (42)_{10}$$

$$\begin{array}{r} 42 \\ 26 | \\ 2 - 10A \quad \uparrow \\ 0 - 2 \end{array}$$

$$(42)_{10} = (2A)_{16}$$

$$\therefore (01000010)_{\text{BCD}} = (42)_{10} = (2A)_{16}$$

22/07/2014

E

## BCD Addition rules :-

- Add two BCD numbers using binary addition rules
- If a 4 bit sum is greater than 9 or if a carry is generated from a 4 bit sum, the sum is invalid
- Add 6 to the 4 bit sum in order to skip the invalid states.

Ex:- Perform the following using BCD arithmetic.

(i)  $1273_{10} + 9587_{10}$

Sol:-  $1273_{10} \rightarrow \underline{\underline{0001}} \underline{\underline{0010}} \underline{\underline{0111}} \underline{\underline{0011}}_{BCD}$

$9587_{10} \rightarrow \underline{\underline{1001}} \underline{\underline{0101}} \underline{\underline{1000}} \underline{\underline{0111}}_{BCD}$

$$\begin{array}{r}
 & 00010010011001 \\
 + & 1001010110000111 \\
 \hline
 & 10100111111010 \\
 & \swarrow \quad \swarrow \quad \swarrow \quad \swarrow \\
 1010 & 1111 & 1111 & 1010 \\
 0110 & & & 0110 0110 \\
 \hline
 10000 & 1000 & 0110 & 0000 \\
 \swarrow & \swarrow & \swarrow & \swarrow \\
 1 & 0 & 8 & 6 & 0
 \end{array}$$

$$1273_{10} + 9587_{10} = 10860_{10}$$

} in the first level of addition add 0110(6) to the groups which are  $>9$ (1001)  
on carry generated out of 4-bit group

From ~~second~~ second level onwards add 6 to the groups which are  $>9$

(ii)  $999_{10} + 989_{10}$

Sol:-  $999_{10} \rightarrow 1001 1001 1001_{BCD}$

$989_{10} \rightarrow 1001 1000 1001_{BCD}$

$$\begin{array}{r}
 1001 1001 1001 \\
 1001 1000 1001 \\
 \hline
 1001100100010
 \end{array}$$

carry →

$$\begin{array}{r}
 0110 0110 0110 \\
 \hline
 11001 1000 1000
 \end{array}$$

↓      ↓      ↓

1      9      B      8

In the first level of addition carry is generated so, add 6(0110) to all 4-bit groups.

$$(iii) \quad 7762_{10} + 3838_{10}$$

Sol:  $7762_{10} \rightarrow 0111 \ 0111 \ 0110 \ 0010$  BCD  
 $3838_{10} \rightarrow 0011 \ 1000 \ 0011 \ 1000$  BCD.

$$\begin{array}{r}
 0111 & 0111 & 0110 & 0010 \\
 0011 & 1000 & 0011 & 1000 \\
 \hline
 1010 & 1111 & 1000 & 1010 \\
 0110 & 0110 & & 0110 \\
 \hline
 10001 & 0101 & 1000 & 0000 \\
 & 0110 & & \\
 \hline
 10001 & 0110 & 0000 & 0000
 \end{array}$$

In the first level of addition  
 add 0110 to the groups which  
 are  $> 1001$  (carry generated  
 out of 4-bit group)

} from 2<sup>nd</sup> level onwards add  
 6 to the groups which are  $> 9$   
 only

$$7762_{10} + 3838_{10} = 11600_{10}$$

\* Non-weighted or unweighted codes :-

Un weighted codes are not assigned  
 with any weight to each digit position

Ex:- EXCESS-3, GRAY codes.

EXCESS-3 code :- It can be derived from the  
 natural BCD code by adding '3' to each coded  
 number.

Decimal	BCD	Excess-3 (BCD+0011)
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

Ex. Find EXCESS-3 code for  $59_{10}$ .

Sol:  $\begin{array}{ccc} 5 & 9 & 2_{10} \\ \downarrow & \downarrow & \downarrow \\ 1000 & 1100 & 0101 \end{array} \rightarrow \text{EXCESS-3 form.}$

$$59_{10} = 100011000101_{\text{EXCESS-3}}$$

→ Binary to Gray code conversion :-

rules:-

$B_1 \ B_2 \ B_3 \ B_4 \dots$  Binary no.

$G_1 \ G_2 \ G_3 \ G_4 \dots$  Gray code no.

$$G_1 = B_1$$

$$G_2 = B_1 \oplus B_2 = B_1 \bar{B}_2 + \bar{B}_1 B_2$$

$$G_3 = B_2 \oplus B_3$$

$$G_4 = B_3 \oplus B_4$$

⋮

⋮

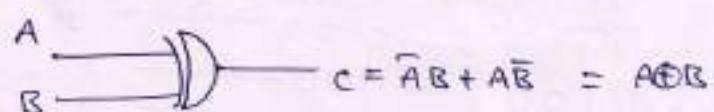
$$\underline{\text{Ex:}} \quad (1100)_2 = (?)_{\text{Gray}}$$

Sol: MSB is same.

remaining digits are replaced with opposite digits  
 $0 \mapsto 0's$   
 $1 \mapsto 1's$

Remaining are as below  
 Table.

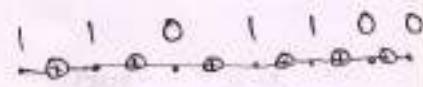
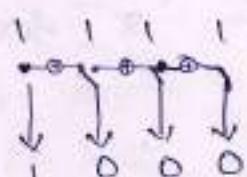
EXOR (or) XOR (or) Exclusive OR gate.



A	B	C = A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

$$\underline{\text{Ex:}} \quad (\text{i}) \quad (1111)_2 = (?)_{\text{Gray}}$$

Sol:-



$$(1111)_2 = (1000)_{\text{Gray}}$$

$$\text{(ii)} \quad (101010)_2 = (?)_{\text{Gray}}$$

$$\underline{\text{Sol:}} \quad (101010)_2 = (110101)_{\text{Gray}}$$

## Gray to Binary :-

rules:-

$$\begin{array}{llll} G_1 & G_2 & G_3 & \dots \text{ Gray code no.} \\ B_1 & B_2 & B_3 & \dots \text{ Binary code no.} \end{array}$$

$$B_1 = G_1$$

$$B_2 = B_1 \oplus G_2$$

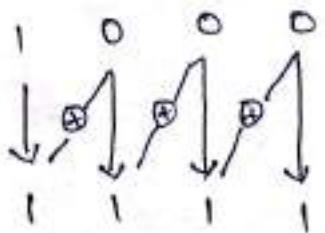
$$B_3 = B_2 \oplus G_3$$

$$B_4 = B_3 \oplus G_4$$

⋮

Ex:- (i)  $(1000)_{\text{Gray}} = (?)_2$

Sol:-



$$(1000)_{\text{Gray}} = (111)_2$$

## Reflective codes :-

A code is said to be reflective when the code for '9' is the complement for the code for '0', '8' for '1', '7' for '2', '6' for '3', '5' for '4'.

Ex:- 2421, 5211, EXCESS-3.

### sequential codes :-

In sequential codes each succeeding code is one binary no. greater than it's preceding code.

### Alphanumeric codes :-

The codes which represent the alphabetic characters , numbers are called alphanumeric codes .

EX:- ASCII . ( American standard code for Information Interchange )

EBCDIC ( Extended Binary coded Decimal Interchange code )

### → Binary to Gray code :-

unit distance code is gray code .

<u>decimal</u>	<u>Binary</u>	<u>Gray</u>
0	0000	0000
1	0001	0001
2	0010	0001
3	0011	0011
4	0100	0010
5	0101	0110
6	0110	0111
7	0111	0101
8	1000	0100
9	1001	1100
10	1010	1101
11	1011	1111
12	1100	1110
13	1101	1010
14	1110	1011
15	1111	1000

Boolean algebra is used to simplify boolean equations, to make simple logic circuits.

Representation of Boolean function (or) switching function of logical function :-

Ex:-  $y = ab + bc$

→ a, b, c are called variables (or) literals, which are inputs for the logic gates.

→ To implement a logic function with less no. of gates we have to minimise variables & the number of terms.

Boolean algebra rules :-

(i) commutative law

$$A+B = B+A$$

$$A \cdot B = B \cdot A$$

(ii) Associative law

$$A+(B+C) = (A+B)+C$$

$$(AB)C = A(BC)$$

(iii) Distributive law

$$A(B+C) = AB+AC$$

$$A+BC = (A+B)(A+C)$$

(iv) Additive law

$$A+0 = A$$

$$A+A = A$$

$$A+1 = 1$$

$$A+\bar{A} = 1$$

## (v) multiplication Property.

$$A \cdot 0 = 0$$

$$A \cdot 1 = A$$

$$A \cdot A = A$$

$$A \cdot \bar{A} = 0$$

## (vi) Double negation.

$$\overline{\overline{A}} = A$$

## (vii) Absorption law

$$A(A+B) = A$$

$$A + AB = A$$

$$A(\bar{A}+B) = AB$$

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

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

(viii) Duality property.  $AB + \bar{A}\bar{B} = (A+B)(\bar{A}+\bar{B})$ (ix) DeMorgan's theorem.  $\overline{A \cdot B} = \bar{A} + \bar{B}$ 

$$\overline{A+B} = \bar{A} \cdot \bar{B}$$

(x) Consensus theorem.  $AB + \bar{A}C + BC = AB + \bar{A}C$ .① Prove that  $(A+B)(A+C) = A+BC$ .

Proof: L.H.S =  $(A+B)(A+C)$

$$= AA + AC + BA + BC \quad (\because A \cdot A = A)$$

$$= A + AC + BA + BC$$

$$= BC + (1 + C + B)A \quad (\because 1 + A = 1)$$

$$= A + BC$$

$$= R.H.S.$$

(1 + Anything = 1 in logical gates).

$\therefore A+BC = (A+B)(A+C)$

Ex: Simplify following boolean expressions to a min. (12)  
no. of variables or literals.

$$\text{Q} \quad ① (A+B)(A+\bar{B})$$

$$\begin{aligned}\text{Sol: } (A+B)(A+\bar{B}) &= A \cdot A + A \cdot \bar{B} + B \cdot A + B \cdot \bar{B} \\ &= A + A \cdot \bar{B} + B \cdot A + 0 \\ &= A(1 + B + \bar{B}) \\ &= A(1 + 1) \quad (\because 1 + \text{anything} = 1) \\ &= A\end{aligned}$$

$$\text{Q} \quad xy + xy\bar{z} + xy\bar{z} + \bar{x}y\bar{z}$$

$$\begin{aligned}\text{Sol: } xy + xy\bar{z} + xy\bar{z} + \bar{x}y\bar{z} &= xy[1 + \bar{z} + \bar{\bar{z}}] + \bar{x}y\bar{z} \quad (\because 1 + \text{anything} = 1) \\ &= xy \cdot 1 + \bar{x}y\bar{z} \\ &= \bar{x}y\bar{z} + xy \\ &= y[\bar{x}\bar{z} + x] = y[(x + \bar{x}) + (x + \bar{z})] \\ &= \bar{y}[x + z]\end{aligned}$$

23/07/2014

$$\text{Q} \quad x'y + xy' + xy + x'y'$$

$$\begin{aligned}\text{Sol: } x'[y+y'] + x[y+y'] &= x'[1] + x[1] = x + x' = 1\end{aligned}$$

$$\text{Q} \quad \bar{A} + AB + A\bar{C} + A\bar{B}\bar{C}$$

$\begin{aligned}\text{Sol: } \bar{A} + AB + A\bar{C}(A + A\bar{B}) &= \cancel{\bar{A} + AB + A\bar{C}(1)} \\ &= \cancel{\bar{A} + A(B + \bar{C})} \\ &= 1 - A + AB + A\bar{C}(1 - C) \\ &= 1 - A + AB + A\bar{C} \\ &= 1 - A + AB + A - AC \\ &= 1 - A + AB + A - AC \\ &= 1 - A + AB + A - AC\end{aligned}$	$\begin{aligned}\bar{A} + AB + A\bar{C}(1 + \bar{B}) &= \bar{A} + AB + A\bar{C} \\ &= (A + \bar{A})(\bar{A} + B) + A\bar{C} \\ &= \bar{A} + B + A\bar{C} \quad (\because \text{distributive law}) \\ &= \bar{A} + A\bar{C} + B \\ &= (\bar{A} + A)(\bar{A} + \bar{C}) + B \\ &= \bar{A} + \bar{C} + B\end{aligned}$
--	---

$$⑤ \quad ABC + \bar{A}B + AB\bar{C}$$

Sol: 
$$\begin{aligned} &= ABC + AB\bar{C} + \bar{A}B \\ &= AB(C + \bar{C}) + \bar{A}B \\ &= AB + \bar{A}B \\ &= B(A + \bar{A}) = B \end{aligned}$$

$$⑥ \quad x'y' + xy + x'y$$

Sol: 
$$\begin{aligned} &= x'(y' + y) + xy \\ &= x' + xy \\ &= (x' + x)(x' + y) \quad (\because \text{distributive law}) \\ &= x' + y \end{aligned}$$

$$⑦ \quad a + ab + a'b'c + a'b'c'd + \dots$$

Sol: 
$$a + a'[b + b'c + b'c'd + \dots] \quad (\text{absorption law})$$

$$(\because A + \bar{A}B = A + B)$$

$$= a + b + b'c' + b'c'd + \dots$$

$$= a + b + b' [c + c'd + c'd'e + \dots] \quad (\because \text{absorption law})$$

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

$$= a + b + c + c'd + c'd'e + \dots$$

⋮

$$= a + b + c + d + e + \dots$$

$$⑧ \quad \text{prove that } A(\bar{A} + C)(\bar{A}B + \bar{C}) = 0$$

Sol: 
$$(A \cdot \bar{A} + AC)(\bar{A}B + \bar{C})$$

$$= AC(\bar{A}B + \bar{C})$$

$$= AC \cdot \bar{A}B + AC \cdot \bar{C} \quad (\because A \cdot \bar{A} = 0)$$

$$= (A \cdot \bar{A})BC + A(0)$$

$$= 0 + 0 = 0.$$

$$Q) P.T. (A+C)(A+D)(B+C)(B+D) = AB + CD.$$

$$\text{Sol: } [A \cdot A + AD + CA + CD][B \cdot B + BD + CB + CD]$$

$$= [A + AD + CA + CD][B + BD + CB + CD] \quad x$$

$$(A+C)(A+D)(B+C)(B+D)$$

$$[\because A+BC = (A+B)(A+C)]$$

$$= (A+CD)(B+CD) = A$$

$$= CD + AB$$

$$10) P.T. AB + \bar{A}C + A\bar{B}C (AB + C) = 1$$

$$\text{Sol: } AB + \bar{A} + \bar{C} + \underbrace{A\bar{B}C \cdot AB}_{0} + A\bar{B}C \cdot C \Leftarrow$$

$$C = AB + \bar{A}C + A\bar{B}C \quad (\because B \cdot \bar{B} = 0)$$

$$= (A+C)(\bar{A} + \bar{C}) + A\bar{B}C \quad x$$

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

$$= A(B + \bar{B}C) + \bar{A}C$$

$$(\because A+BC = (A+B)(A+C))$$

$$= A[(B+\bar{B}) + (B+C)] + \bar{A}C$$

$$(\because B+\bar{B} = 1)$$

$$= AB + AC + \bar{A}C$$

$$= 1 + AB \quad (\because 1 + \text{anything} = 1)$$

$$= 1$$

$$11) \text{ PROVE THAT } \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + A\bar{B}\bar{C} + A\bar{B}C = \bar{C}$$

$$\text{Sol: } \bar{A}\bar{C}[B + \bar{B}] + A\bar{C}[B + \bar{B}]$$

$$(\because A + \bar{A} = 1)$$

$$= \bar{A}\bar{C} + A\bar{C}$$

$$= \bar{C}[A + \bar{A}] = \bar{C}$$

Ex: Reduce the following Boolean expression to indicated no. of variables or literals.

(12)  $A\bar{C} + AB\bar{C} + A\bar{C}$  to 3 literals.

Sol:-  $(\bar{A}+A)\bar{C} + ABC$

$$= \bar{C} + ABC.$$

$$= \bar{C} + C \cdot AB$$

$$(\because A + BC = (A+B)(A+C))$$

$$= (\bar{C} + C)(\bar{C} + AB)$$

$$(\bar{A} + A = 1)$$

$$= \bar{C} + AB$$

(13)  $(\overline{xy} + z) + \bar{z} + xy + wz$  to 3 variables.

Sol:-  $(\overline{xy}) + \bar{z} + \bar{z} + xy + wz$  ( $\because \overline{A+B} = \bar{A} \cdot \bar{B}$ )

$$xy\bar{z} + \bar{z} + xy + wz$$

$$= \bar{z} + \bar{z} \cdot xy + xy + wz$$

$$(\bar{A} + BC = (\bar{A} + B)(\bar{A} + C))$$

$$= (\bar{z} + \bar{z})(\bar{z} + xy) + xy + wz$$

$$= xy(1 + \bar{z}) + \bar{z}(1 + w)$$

$$= \bar{z} + xy + wz$$

$$= \bar{z} + xy \quad (\because 1 + anything = 1)$$

$$= \bar{z} + xy$$

(13)(i)  $(\overline{xy} + z) + \bar{z} + xy + wz$  to 3 variables.

Sol:-  $\overline{\bar{x} \cdot \bar{y}} \cdot \bar{z} + \bar{z} + xy + wz$

$$(\overline{\bar{x} + \bar{y}}) \bar{z} + \bar{z}(1+w) + xy$$

$$= xy\bar{z} + xy +$$

$$= x\bar{z} + y\bar{z} + \bar{z} + xy$$

$$= \bar{z} + x\bar{z} + y\bar{z} + xy$$

$$= (\bar{z} + x)(\bar{z} + \bar{y}) + y\bar{z} + xy$$

$$= x + \bar{z} + xy + \bar{y}\bar{z} - x(1+y) + (z+y)(z+\bar{z})$$

$$= x + y + z \quad (\because 1 + \text{anything} = 1 \\ A + \bar{A} = 1).$$

(14)  $A'B(C'D + D') + B(A + \bar{A}CD) \rightarrow 1 \text{ (literal)}$

Sol:  $A'BC'D + A'BD' + AB + A'BCD$

$$= A'BD(C + C') + AB + A'BD' \quad (\because A + BC = (A + B)(A + C))$$

$$= A'BD + AB + A'BD' \quad (A + \bar{A} = 1)$$

$$= A'B(D + D') + AB$$

$$= A'B + AB = B(A + A') = B$$

~~Ex:~~ Find the complement of the Boolean function and reducing to a minimum number of variables.

~~Sol:~~ (15)  $(B\bar{C} + \bar{A}D)(A\bar{B} + C\bar{D})$

~~Sol:~~ after taking complement

$$\overline{(B\bar{C} + \bar{A}D)(A\bar{B} + C\bar{D})}$$

$$= \overline{(AB\bar{B}\bar{C} + B\bar{C}C\bar{D} + A\bar{A}D\bar{B} + \bar{A}DC\bar{D})} \quad (\because A \cdot \bar{A} = 0)$$

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

(16)  $x'(y' + z')(x + y + z')$

~~Sol:~~  $\overline{x'(y' + z')(x + y + z')}$

$$= \overline{(x'y' + x'z')(x + y + z')}$$

$$= \overline{(x'y'x + x'z'x + x'y'y + x'y'z' + x'z'y + x'z'z)}$$

$$= \overline{x'y'z' + x'z'y + x'z'z}$$

$$= \overline{x'z'(y+y')} + \overline{x'z'z} = \overline{x'z' + x'z'} \quad (C = \overline{x(y+z)})$$

$$= \overline{x'z'(1+1)} = \overline{x'z'}$$

$$= \overline{\overline{xz}} = x_0$$

$$= \overline{(\bar{x})} + \overline{(\bar{z})} = x + z$$

(17) Find dual of  $F = \bar{A}B + B\bar{C} + A\bar{C}$

Sol:- Duality of  $F = (\bar{A}+B)(B+C)(A+\bar{C})$

Sub.  $x's \rightarrow x'$   
 $+s \rightarrow x's$   
 $0's \rightarrow 1's$   
 $1's \rightarrow 0's$

(18) Find dual of  $xy + x'z = 0$

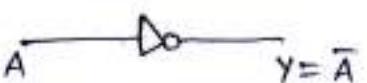
Sol:- Duality  $\Rightarrow (x+y)(x'+z) = 1$

## LOGIC GATES :-

1. Inverter (or) NOT gate :-

'NOT' Gate performs inversion operation.

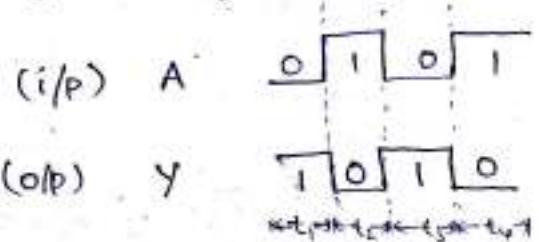
It has 1 input and one output.

Logic Symbol :- 

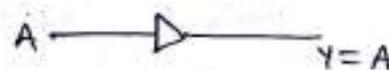
Truth Table :- Truth table indicates outputs for different possibilities of input.

i/p	o/p
A	$y = \bar{A}$
0	1
1	0

Timing diagram :- The input & output wave forms showing time relationship is called timing diagram.



## 2. Buffer gate :-



output is same as the i/p

i/p = o/p.

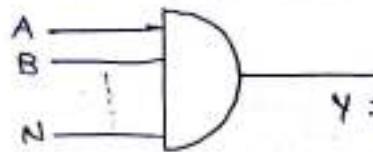
It used for time delay.

## 3. AND gate :-

'AND' gate performs logical multiplication.

It has two or more i/p's and one output.

Logic symbol :-



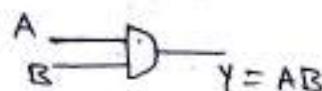
$$Y = A \text{ and } B \text{ and } \dots \text{ and } N$$

$$= A \cdot B \cdot \dots \cdot N$$

$$= AB \dots N$$

## 2 i/p AND gate :-

Logic symbol :-

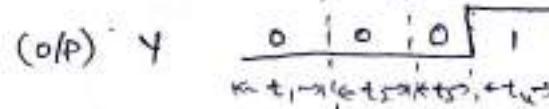
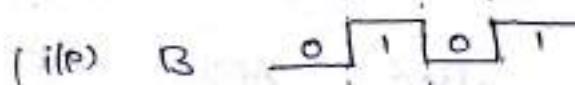
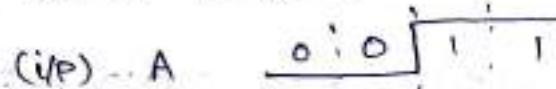


Truth table :-

i/p's		o/p
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

AND gate o/p is high when all i/p's are high.

Timing diagram :-



at, after, etc., etc.

## 1. i/p AND gate :-

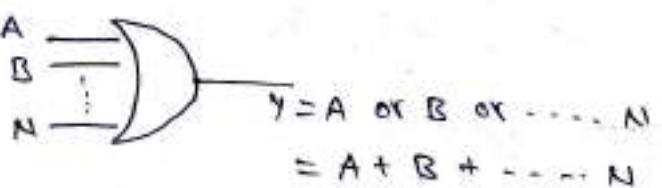


#### ④ OR gate :-

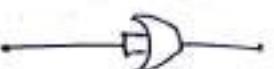
(PQ)

OR gate performs logical Addition. It has two or more inputs & one output.

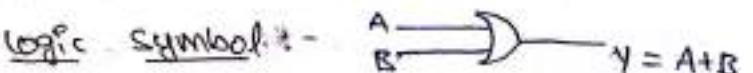
Logic symbol :-



1 i/p OR gate :-



2 i/p OR gate :-

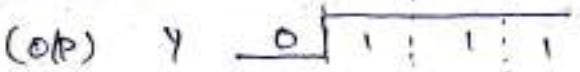
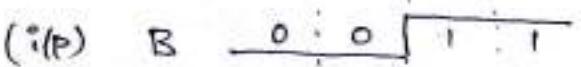
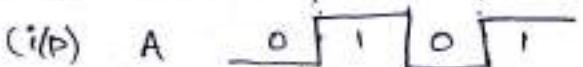


Truth table :-

i/p's		O/P
A	B	Y
0	0	0
1	0	1
0	1	1
1	1	1

OR gate o/p is low when all the inputs are low.

Timing diagram :-



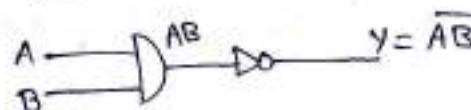
Note:- logic circuits which use AND, OR, NOT gates only are called AOI logic circuits

## Universal gates :-

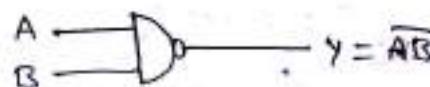
NAND & NOR are called universal gates because by using either NAND or NOR gates we can implement any logic circuit.

NAND gate:- The term NAND is a combination of AND & NOT gates.

Logic symbol:-



2 i/p NAND gate:-



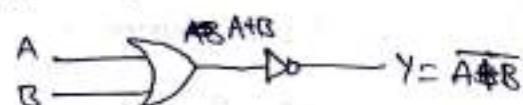
Truth table:-

i/p's		o/p
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

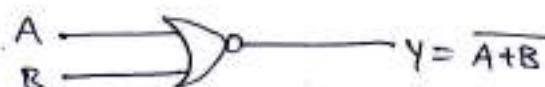
NAND gate o/p is low when all the i/p's are high.

NOR gate:- The term NOR is a combination of OR & NOT gates.

Logic symbol:-



2 i/p's NOR gate:-



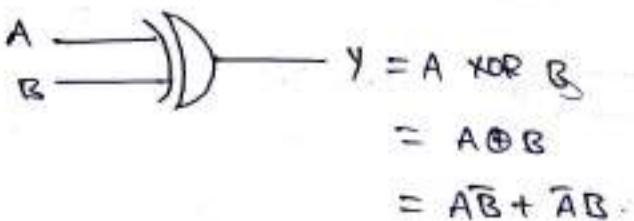
Truth table :-

i/p's		O/P
A	B	y
0	0	1
1	0	0
0	1	0
1	1	0

NOR gate O/P is high when all i/p's are low.

EXOR (or) XOR (or) EXCLUSIVE-OR gate :-

Logic symbol :-



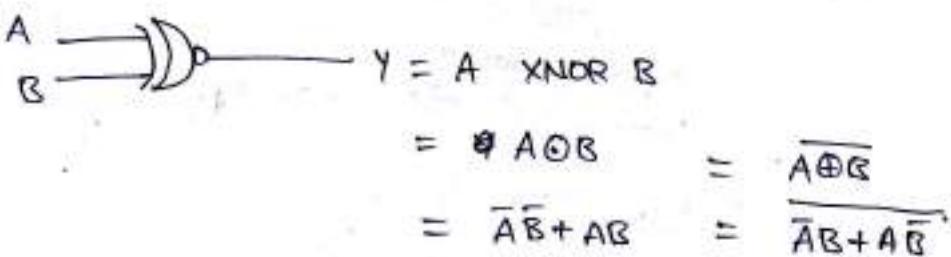
Truth table :-

i/p's		O/P
A	B	y
0	0	0
0	1	1
1	0	1
1	1	0

XOR O/P is low when all i/p's are same.

EXNOR (or) XNOR (or) EXCLUSIVE NOR gate :-

Logic symbol :-



Truth table:-

i/p's		o/p
A	B	y
0	0	1
0	1	0
1	0	0
1	1	1

EXNOR o/p is high when all i/p's are same.

Note:- dual of XOR gate is XNOR.

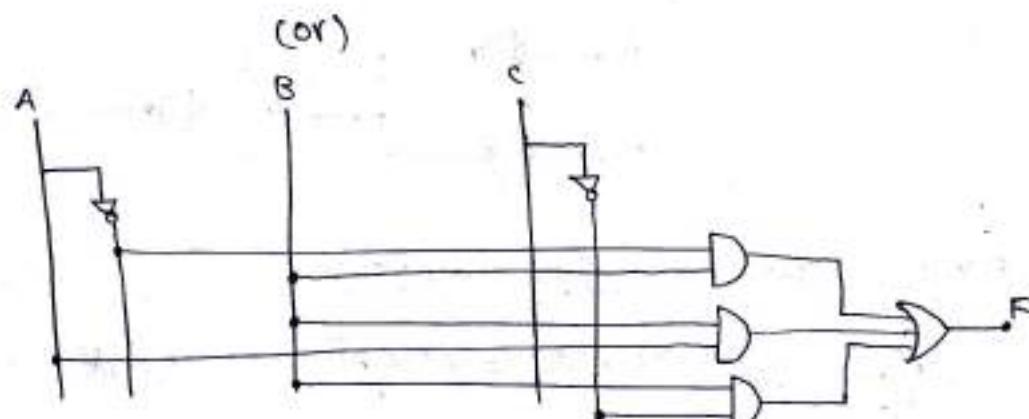
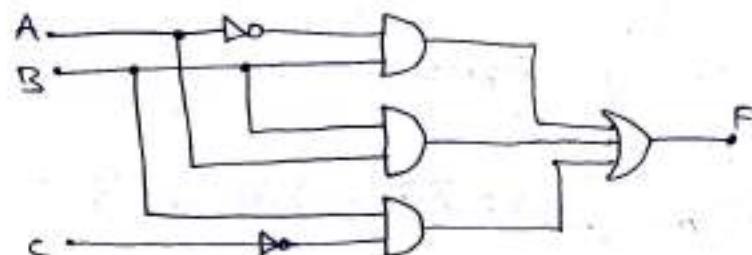
30/07/2014

\* Dual of XOR gate is XNOR.

$$\begin{aligned}
 \text{dual of } \bar{A}B + A\bar{B} &= (\bar{A}+B)(A+\bar{B}) \\
 &= \bar{A}\cdot A + \bar{A}\cdot\bar{B} + B\cdot A + B\cdot\bar{B} \\
 &= \bar{A}\bar{B} + AB
 \end{aligned}$$

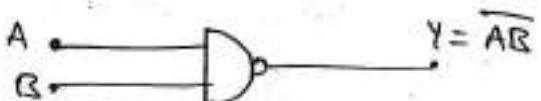
Ex: Implement  $F = \bar{A}B + AB\bar{C} + BC$  using basic logic gates.

Sol:-



Ex: Implement NOT, AND, OR, NOR, EXNOR, EXOR gates using NAND gate.

Sol:- NAND gate :-



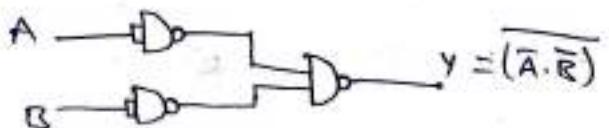
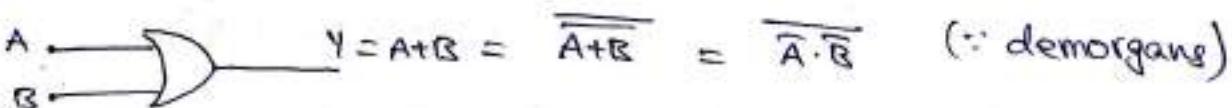
NOT gate using NAND gate :-



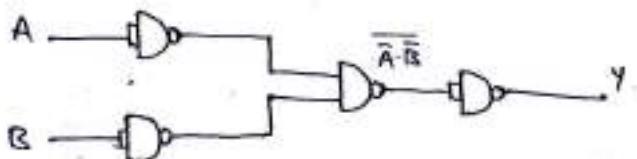
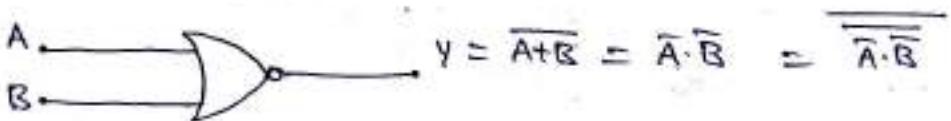
AND gate using NAND :-



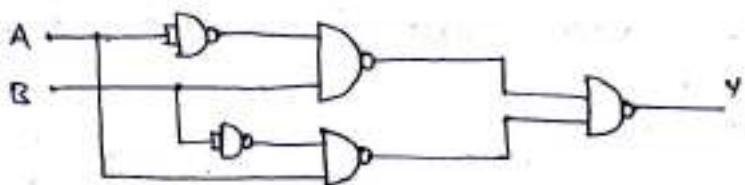
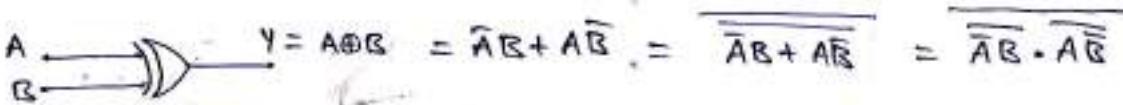
OR gate using NAND :-



NOR gate using NAND :-

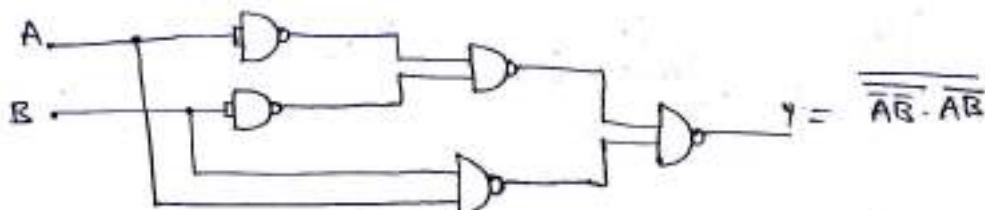


EXOR gate using NAND :-



EXNOR gate using NAND:-

$$\text{A} \quad \text{B} \quad Y = A \otimes B = \overline{\overline{A}\overline{B}} + \overline{A}\overline{B} = \overline{\overline{A}\overline{B} + A\overline{B}} = \overline{\overline{A}\overline{B} \cdot \overline{A}\overline{B}}$$



Ex:- Implement all logic gates except NOR gate using NOR gate.

Sol:- NOR gate :-

$$\text{A} \quad \text{B} \quad Y = \overline{A+B}$$

NOT gate using NOR gate :-

$$\text{A} \quad Y = \overline{A} \quad = \quad \text{A} \quad Y = \overline{\overline{A}}$$

AND gate using NOR gate :-

$$\text{A} \quad \text{B} \quad Y = AB = \overline{\overline{A}\overline{B}} = \overline{\overline{A}+\overline{B}}$$

$$\text{A} \quad \text{B} \quad Y = \overline{\overline{A}+\overline{B}}$$

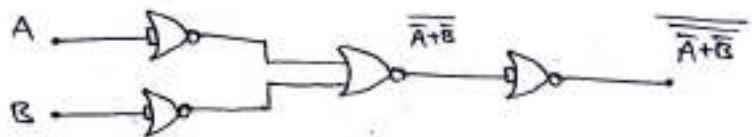
OR gate using NOR gate :-

$$\text{A} \quad \text{B} \quad Y = A+B = \overline{\overline{A}+\overline{B}}$$

$$\text{A} \quad \text{B} \quad \overline{\overline{(A+B)}}$$

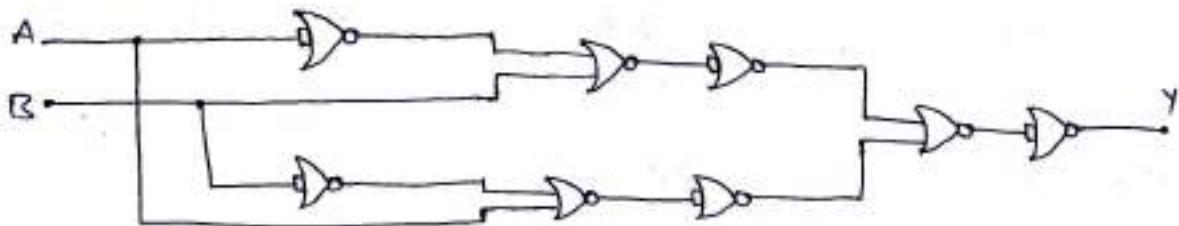
NAND gate using NOR gate :-

$$\text{A} \quad \text{B} \quad \text{NOR gate} \quad \overline{AB} = \overline{A} + \overline{B} = \overline{\overline{A} + \overline{B}}$$



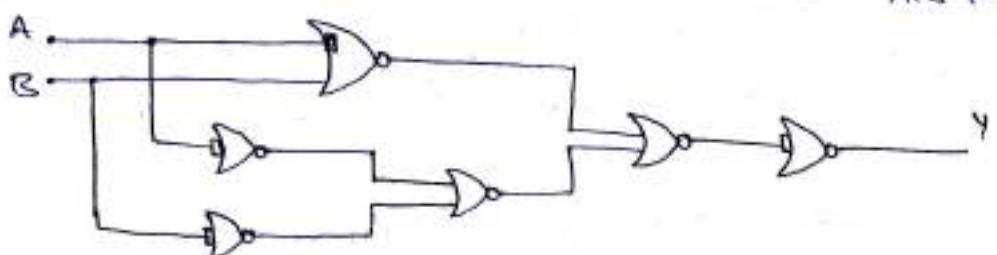
EXOR gate using NOR gate :-

$$\text{A} \quad \text{B} \quad \text{NOR gate} \quad Y = A \oplus B = \overline{AB} + A\overline{B} = \overline{\overline{AB}} + \overline{A\overline{B}} \\ = \overline{\overline{AB} + A\overline{B}}$$



EXNOR gate using NOR gate :-

$$\text{A} \quad \text{B} \quad \text{NOR gate} \quad Y = A \otimes B = \overline{AB} + AB = \overline{\overline{AB} + AB} = \overline{\overline{AB}} + \overline{AB} \\ = \overline{\overline{A} + \overline{B}} + \overline{A + B} = \overline{\overline{A}\overline{B}} \cdot \overline{AB} \quad \overline{A + B} + \overline{A + B} \\ = \overline{\overline{A} + \overline{B}}$$



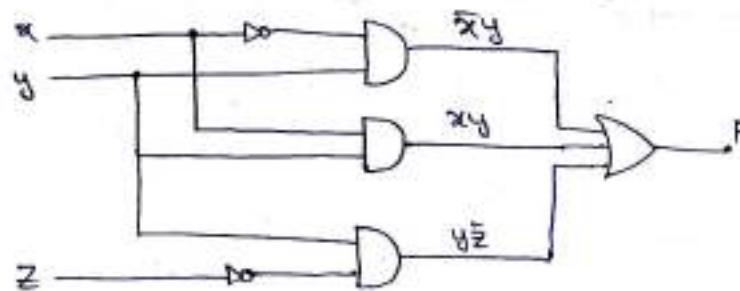
Ex:- Implement  $F = \overline{xy} + \overline{xz} + y\overline{z}$  using (i) Basic gates

(ii) AND & NOT gates (iii) OR & NOT gates

(iv) only NAND gate (iv) only NOR gate.

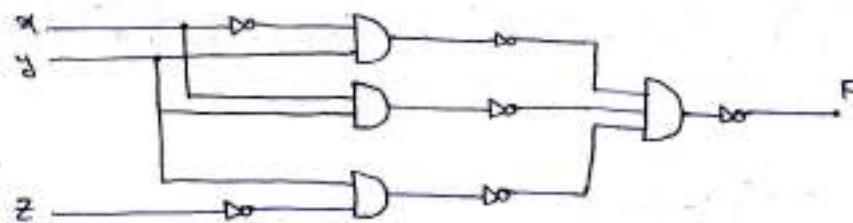
Sol:-  $F = \bar{x}y + xy + y\bar{z}$

(i) Using Basic gates.



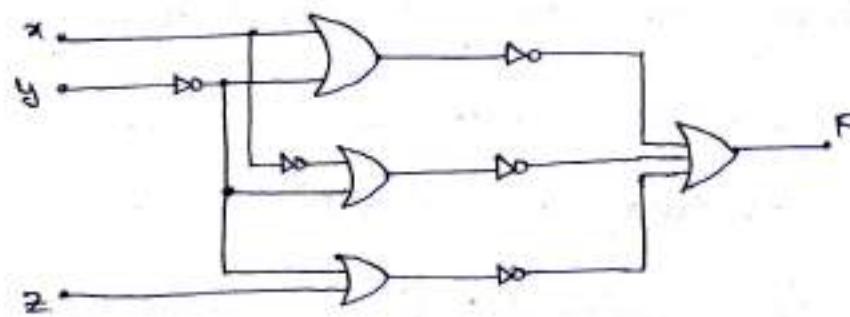
(ii) Using AND & NOT gates

$$F = \bar{x}y + xy + y\bar{z} = \overline{\bar{x}y + xy + y\bar{z}} = \overline{\overline{\bar{x}y} \cdot \overline{xy} \cdot \overline{y\bar{z}}}$$



(iii) Using OR & NOT gates.

$$\begin{aligned} F &= \bar{x}y + xy + y\bar{z} \\ &= \overline{\bar{x}y} + \overline{xy} + \overline{y\bar{z}} \\ &= \overline{\bar{x} + y} + \overline{\bar{x} + y} + \overline{(y + \bar{z})} \\ &= \overline{x + \bar{y}} + \overline{\bar{x} + y} + \overline{y + \bar{z}} \end{aligned}$$

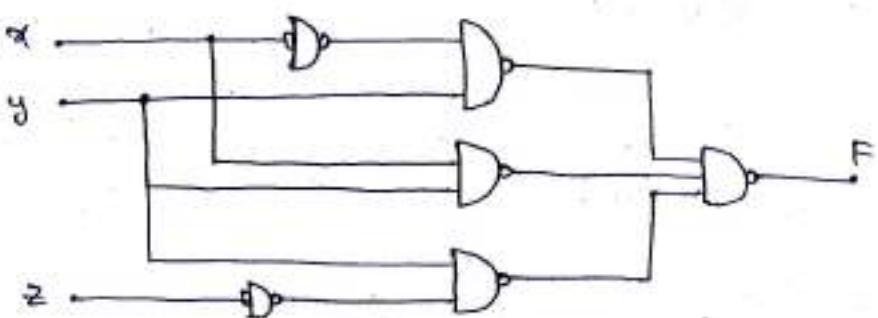


(iv) Using NAND gate

$$F = \bar{x}y + xy + y\bar{z}$$

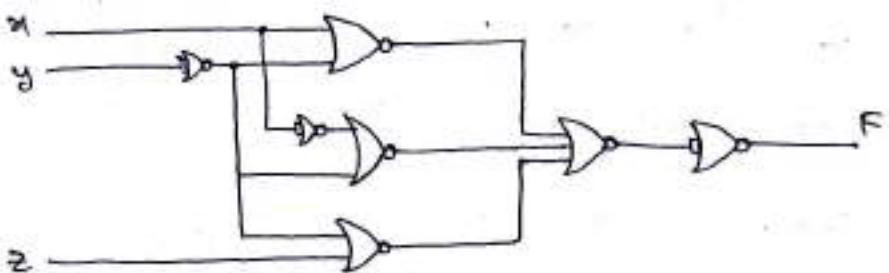
$$= \overline{\overline{\bar{x}y + xy + y\bar{z}}} \quad (\text{de Morgan's})$$

$$= \overline{\overline{\bar{x}y} \cdot \overline{xy} \cdot \overline{y\bar{z}}}$$



(v) using NOR gates.

$$\begin{aligned}
 F &= \bar{x}y + xy + y\bar{z} \\
 &= \overline{\bar{x}y} + \overline{xy} + \overline{y\bar{z}} \\
 &= \overline{\bar{x}+\bar{y}} + \overline{\bar{x}+y} + \overline{y+\bar{z}} \\
 &= \overline{x+\bar{y}} + \overline{\bar{x}+\bar{y}} + \overline{\bar{y}+z} = \overline{\overline{x+y} + \overline{\bar{x}+\bar{y}} + \overline{y+z}}
 \end{aligned}$$



\* Representation of switching functions (or) logic functions

(or) Boolean function :-

there are two standard forms in which logic functions can be expressed.

- (i) Sum of products form (SOP)
- (ii) Product of sum's (POS).

(i) SOP :- SOP is a group of product terms ORed together.

Ex :-  $F = \bar{x}y + xy + yz$

(ii) POS: POS is a group of sum terms ANDed together. 25

Ex:  $F = (\bar{x} + \bar{y})(x + y)(y + z)$

Standard & canonical SOP & POS forms :-

In the SOP & POS forms all the individual terms do not contain all variables. If each term is SOP & POS form contains all variables then it is canonical SOP & POS forms.

Ex. of CSOP is,

$$F(A, B, C) = \bar{A}BC + ABC + A\bar{B}C$$

Ex. of CPOS is,

$$F(A, B, C) = (\bar{A} + B + C)(A + B + C)(A + \bar{B} + C)$$

Ex:- convert the given expression is into canonical  
standard SOP form.

(i)  $y(A, B, C) = \bar{A}B + A\bar{B}C$ .

Sol:  $= \bar{A}B(c + \bar{c}) + A\bar{B}C \quad (\because c + \bar{c} = 1)$   
 $= \bar{A}Bc + \bar{A}B\bar{c} + A\bar{B}C$

(ii)  $y(A, B, C) = AB + BC + CA$

Sol:  $= AB(c + \bar{c}) + BC(A + \bar{A}) + CA(\bar{C} + \bar{B})$   
 $= ABC + ABC\bar{c} + ABC + \bar{ABC} + ABC + A\bar{B}C$   
 $= ABC(1 + 1 + 1) + ABC\bar{c} + A\bar{B}C + \bar{ABC} \quad (\because 1 + \text{anything} = 1)$   
 $= ABC + ABC\bar{c} + A\bar{B}C + \bar{ABC}$

(iii)  $A(x, y, z) = x + xy + xyz$

Sol:  $= x(y + \bar{y})(z + \bar{z}) + xy(z + \bar{z}) + xyz$   
 $= x(yz + y\bar{z} + \bar{y}z + \bar{y}\bar{z}) + xy(z + \bar{z}) + xyz$   
 $= xyz + xy\bar{z} + x\bar{y}z + x\bar{y}\bar{z} + xy\bar{z} + xy\bar{z} + xyz$   
 $= xyz(1 + 1 + 1) + xy\bar{z} + x\bar{y}z + x\bar{y}\bar{z} + xy\bar{z}$   
 $= xyz + xy\bar{z} + x\bar{y}z + x\bar{y}\bar{z} + xyz$   
 $= xyz + xy\bar{z} + x\bar{y}z + x\bar{y}\bar{z} + xyz$

05/08/2014

Ex:- Convert the given expression into canonical POS form.

(i)  $F(A, B, C) = (A+B)(B+C)(A+C)$

Sol:-  $= (\underline{A+B} + \underline{C}\bar{C})(B+C+A\bar{A})(A+C+B\bar{B})$  ( $\because A+\bar{A}C = (A+B)(A+C)$ )  
 $= (A+B+C)(A+B+\bar{C})(B+C+A)(B+C+\bar{A})(A+C+B)(A+\bar{B}+C)$   
 $= (A+B+C)(\bar{A}+\bar{B}+C)(A+\bar{B}+C)(A+\bar{B}+\bar{C})$

(ii)  $F(x, y, z) = \bar{x}(x+y)(x+y+\bar{z})$

Sol:-  $= (\bar{x} + y\bar{y} + z\bar{z})(x+y+z\bar{z})(x+y+\bar{z})$   
 $= (x+y\bar{y}+z)(x+y\bar{y}+\bar{z})(x+y+z)(x+y+\bar{z})(x+y+\bar{z})$   
 $= (x+y+z)(x+\bar{y}+z)(x+y+\bar{z})(x+\bar{y}+\bar{z})(x+y+\bar{z})$   
 $= (x+y+z)(x+\bar{y}+z)(x+y+\bar{z})(x+\bar{y}+\bar{z})$

#### \* Minterms and Maxterms :-

→ Each individual term in CSOP is called Minterm.

→ Each individual term in CPoS is called Maxterm.

\* For 'n' variable logic function there are  $2^n$  minterms and  $2^n$  maxterms.

#### Minterms and Maxterms for 3 Binary Variables :-

Variables

A B C

0 0 0

0 0 1

0 1 0

0 1 1

1 0 0

1 0 1

1 1 0

1 1 1

Minterms ( $m_i$ )

Term designation

$\bar{A}\bar{B}\bar{C}$  —  $m_0$

$\bar{A}\bar{B}C$  —  $m_1$

$\bar{A}BC$  —  $m_2$

$\bar{A}B\bar{C}$  —  $m_3$

$\bar{A}B\bar{C}$  —  $m_4$

$\bar{A}B\bar{C}$  —  $m_5$

$\bar{A}B\bar{C}$  —  $m_6$

$\bar{A}B\bar{C}$  —  $m_7$

Maxterm ( $M_i$ )

Term designation

$A+B+C$  —  $M_0$

$A+B+\bar{C}$  —  $M_1$

$A+\bar{B}+C$  —  $M_2$

$A+\bar{B}+\bar{C}$  —  $M_3$

$\bar{A}+B+C$  —  $M_4$

$\bar{A}+B+\bar{C}$  —  $M_5$

$\bar{A}+\bar{B}+C$  —  $M_6$

$\bar{A}+\bar{B}+\bar{C}$  —  $M_7$

- the minterm is represented by  $m$ , Maxterm by  $M$
- the subscript 'i' is the decimal number equivalent of binary input.
- \* with these short hand notations, logic function can be represented as follows.

$$\text{Ex:- (i) } Y = \overline{A}BC + \overline{\overline{A}}\overline{B}C + \overline{A}\overline{\overline{B}}C + A\overline{\overline{B}}\overline{C}$$

$$= m_7 + m_3 + m_5 + m_4$$

$$= \sum m(3, 4, 5, 7)$$

$$= \sum (3, 4, 5, 7)$$

$\Sigma$  denotes SOP

$$\text{(ii) } Y = (\overline{A} + \overline{B} + C)(\overline{\overline{A}} + \overline{B} + \overline{C})(\overline{\overline{A}} + \overline{B} + \overline{C})$$

$$= M_0 \cdot M_5 \cdot M_7$$

$$= \prod M(0, 5, 7)$$

$$= \prod (0, 5, 7)$$

$\prod$  denotes POS

Note: If o/p func is '1' then it corresponds to minterm.  
 If o/p func is '0' then it corresponds to maxterm.  
Ex:- Find SOP and POS expressions from given truthtable

i/p's	o/p
A B C	y
0 0 0	1
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	0
1 1 0	0
1 1 1	1

Note:-

O/P is '1' → minterm

O/P is '0' → maxterm.

Sol:- SOP

$$Y(A, B, C) = \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C + \overline{A}B\overline{C} + ABC$$

$$= m_0 + m_2 + m_3 + m_7 = \sum m(0, 2, 3, 7)$$

POS

$$Y(A, B, C) = (A + B + C)(\overline{A} + B + C)(\overline{A} + \overline{B} + C)(\overline{A} + \overline{B} + \overline{C})$$

$$= M_1 \cdot M_4 \cdot M_5 \cdot M_6 = \prod M(1, 4, 5, 6)$$

\* From the above expressions, there is a complementary

Ex: Simplify the following 3 variable expressions using boolean algebra.

(i)  $Y = \sum m(1, 3, 5, 7)$  considering A, B, C as 3 input variables.

Sol: 
$$\begin{aligned} Y &= m_1 + m_3 + m_5 + m_7 \\ &= \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}C + ABC \\ &= \bar{A}C(B + \bar{B}) + AC(B + \bar{B}) \\ &= \bar{A}C + AC \\ &= C(A + \bar{A}) = C \end{aligned}$$

(ii)  $Y = \prod M(3, 7)$

Sol: 
$$\begin{aligned} Y &= M_3 \cdot M_7 \\ &= (A + \bar{B} + \bar{C})(\bar{A} + \bar{B} + \bar{C}) \\ &= A\bar{A} + A\bar{B} + A\bar{C} + \bar{A}\bar{B} + \bar{B}\bar{B} + \bar{B}\bar{C} + \bar{C}\bar{A} + \bar{B}\bar{C} + \bar{C}\bar{C} \\ &= A\bar{B} + A\bar{C} + \bar{A}\bar{B} + \bar{B}\bar{C} + \bar{C}\bar{A} + \bar{C} \\ &= \bar{B}(A + \bar{A}) + \bar{C}(A + \bar{A}) + \bar{B}(1 + \bar{C}) + \bar{C} \\ &= \bar{B} + \bar{C} + \bar{B} + \bar{C} \\ &= \bar{B} + \bar{C}. \quad (\because 1 + \text{anything} = 1) \end{aligned}$$

Ex: Convert the given expression into Minterms using complementary property and simplify the expression.

(i)  $F = \prod M(3, 5, 7)$

Sol:  $F = M_3 \cdot M_5 \cdot M_7 \rightarrow$  This is POS form.  
 $\therefore$  SOP form is.

$$\begin{aligned} F(A, B, C) &= \sum m(0, 1, 2, 4, 6) \\ &= m_0 + m_1 + m_2 + m_4 + m_6 \\ &= \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + \bar{A}B\bar{C} + AB\bar{C} + ABC \\ &= \bar{A}\bar{B}(C + \bar{C}) + \bar{A}B\bar{C} + AC(B + \bar{B}) \\ &= \bar{A}\bar{B} + \bar{A}B\bar{C} + AC \end{aligned}$$

$$\begin{aligned}
 &= \bar{A}\bar{B} + \bar{C}(\bar{A}\bar{B} + A) \\
 &= (\bar{A}\bar{B} + \bar{C})(\bar{A}\bar{B} + \bar{A}\bar{B} + A) \\
 &= (\bar{A}\bar{B} + \bar{C})(\bar{A}(B + \bar{B}) + A) \\
 &= (\bar{A}\bar{B} + \bar{C})(A + \bar{A}) = \bar{A}\bar{B} + \bar{C}.
 \end{aligned}$$

(∴ distributive law)

Ex: Express the following functions in sum of minterms (canonical SOP) and product of max. terms (canonical POS) forms.

$$(i) F(w, x, y, z) = w'y + xy' + yz + xyz.$$

Sol:w'y

w	x	y	z	
0	0	1	0	-m <sub>2</sub>
0	0	1	1	-m <sub>3</sub>
0	1	1	0	-m <sub>6</sub>
0	1	1	1	-m <sub>7</sub>

xy

w	x	y'	z	
0	1	0	0	-m <sub>4</sub>
0	1	0	1	-m <sub>5</sub>
1	1	0	0	-m <sub>12</sub>
1	1	0	1	-m <sub>13</sub>

yz

w	x	y	z	
0	0	1	1	-m <sub>3</sub>
0	1	1	1	-m <sub>7</sub>
1	0	1	1	-m <sub>11</sub>
1	1	1	1	-m <sub>15</sub>

xyz

w	x	y	z	
0	1	1	1	-m <sub>7</sub>
1	1	1	1	-m <sub>15</sub>

CSOP (or) Sum of minterms form,

$$\begin{aligned}
 F(w, x, y, z) &= m_2 + m_3 + m_4 + m_5 + m_6 + m_7 + m_{11} + m_{12} + \\
 &\quad m_{13} + m_{15} \\
 &= \sum m(2, 3, 4, 5, 6, 7, 11, 12, 13, 15)
 \end{aligned}$$

Remaining terms are Maxterms.

CPoS (or) Product of Maxterms form.

$$\begin{aligned}
 F(w, x, y, z) &= M_0 \cdot M_1 \cdot M_8 \cdot M_9 \cdot M_{10} \cdot M_{14} \\
 &= \prod M(0, 1, 8, 9, 10, 14)
 \end{aligned}$$

$$② F(A, B, C) = (AB + C)(AC + B)$$

Sol:-  $F(A, B, C) = ABC + AB + AC + BC$

<u>ABC</u>	<u>AB</u>	<u>BC</u>	<u>AC</u>
A B C 1 1 1 → m <sub>7</sub>	A B C 1 1 0 → m <sub>6</sub> 1 1 1 → m <sub>7</sub>	A B C 0 1 1 → m <sub>3</sub> 1 1 1 → m <sub>7</sub>	A B C 1 0 1 → m <sub>5</sub> 1 1 1 → m <sub>7</sub>

Sum of minterms  $F = \sum m(3, 5, 6, 7)$ .

Product of maxterms  $F = \prod M(0, 1, 2, 4)$

06/08/2014

### \* Karnaugh Map (K-Map) :-

K-Map gives a systematic approach for simplifying a boolean expression. The basis of this method is a graphical chart known as "K-Map". It contains boxes called cells. Each of the cell represents one of the  $2^n$  possible products that can be formed from 'n' variables.

#### 2-variable K-Map

##### (i) Representation using

A\B	B	$\bar{B}$
$\bar{A}$	$\bar{A}\bar{B}$	$\bar{A}B$
A	$A\bar{B}$	AB

##### 2-variable K-Map Minterm representation using minterms

A\B	0	1
0	$m_0$ 00	$m_1$ 01
1	$m_2$ 10	$m_3$ 11

##### (ii) Representation using maxterms

A\B	B	$\bar{B}$
A	$A+B$	$A+\bar{B}$
$\bar{A}$	$\bar{A}+B$	$\bar{A}+\bar{B}$

A\B	0	1
0	$M_0$ 00	$M_1$ 01
1	$M_2$ 10	$M_3$ 11

Ex: (i) Simplify  $F(A, B) = \sum m(0, 1)$  using K-Map.

Sol:

A	B	$\bar{B}$	B
$\bar{A}$	0	1	1
A	1	2	3

$$F(A, B) = \sum m(0, 1)$$

$$= m_0 + m_1$$

Fill minterms with '1'.

$$\therefore F(A, B) = \bar{A}$$

$$(\because F(A, B) = \bar{A}\bar{B} + \bar{A}B$$

$$= \bar{A}(B + \bar{B}) = \bar{A})$$

(ii) Simplify  $F(A, B) = \sum m(0, 2)$  using K-Map.

Sol:

A	B	$\bar{B}$	B
$\bar{A}$	0	1	1
A	1	2	3

$$F(A, B) = \sum m(0, 2)$$

$$= m_0 + m_2$$

$$\therefore F(A, B) = \bar{B}$$

(iii) Simplify  $F(A, B) = \sum m(0, 3)$  using K-Map.

Sol:

A	B	$\bar{B}$	B
$\bar{A}$	0	1	1
A	1	2	3

$$F(A, B) = \sum m(0, 3)$$

$$= m_0 + m_3$$

$$\therefore F(A, B) = \bar{A}\bar{B} + AB$$

(iv) Simplify  $F(A, B) = \prod M(0, 3)$  using K-Map.

Sol:

A	B	$\bar{B}$	B
$\bar{A}$	0	0	1
A	1	2	3

$$F(A, B) = \prod M(0, 3)$$

$$= M_0 \cdot M_3$$

Fill maxterms with '0's.

$$\therefore F(A, B) = \bar{B}$$

(v) Simplify  $F(x, y) = \prod M(1, 2)$  using K-Map.

Sol:

x	y	$\bar{y}$
$\bar{x}$	0	0
x	1	0

$$F(x, y) = \prod M(1, 2)$$

$$= M_1 \cdot M_2$$

Fill maxterms with '0's.

$$\therefore F(x, y) = (x + \bar{y})(\bar{x} + y)$$

### 3 - Variable K-Map :-

Representation using Minterms

A	BC	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$	$\bar{A}\bar{B}\bar{C}$	$\bar{A}\bar{B}C$	$\bar{A}B\bar{C}$	$\bar{A}B\bar{C}$	$A\bar{B}\bar{C}$
A	$A\bar{B}\bar{C}$	$A\bar{B}C$	$AB\bar{C}$	$ABC$	$A\bar{B}\bar{C}$

A	BC	00	01	11	10
0	$m_0$	$m_1$	$m_3$	$m_2$	
1	$m_4$	$m_5$	$m_7$	$m_6$	

Representation using Maxterms

A	BC	$B+C$	$B+\bar{C}$	$\bar{B}+\bar{C}$	$\bar{B}+C$
A	$A+B+C$	$A+B+\bar{C}$	$A+\bar{B}+\bar{C}$	$A+\bar{B}+C$	
$\bar{A}$	$\bar{A}+B+C$	$\bar{A}+B+\bar{C}$	$\bar{A}+\bar{B}+\bar{C}$	$\bar{A}+\bar{B}+C$	

A	BC	00	01	11	10
0	$M_0$	$M_1$	$M_3$	$M_2$	
1	$M_4$	$M_5$	$M_7$	$M_6$	

Ex-1 Simplify  $F(A, B, C) = \sum m(1, 5, 2, 6)$  using K-Map.

Sol:-

A	BC	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
0	00	01	11	10	
1	0	1	1	0	

$$F(A, B, C) = \sum m(1, 5, 2, 6)$$

$$= m_1 + m_2 + m_5 + m_6.$$

Fill minterms with '1's.

$$\therefore F(A, B, C) = \bar{B}C + BC$$

$$\therefore F(A, B, C) = B \oplus C$$

$$\begin{aligned} \because F(A, B, C) &= \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + A\bar{B}\bar{C} + AB\bar{C} \\ &= \bar{B}\bar{C}(A + \bar{A}) + B\bar{C}(A + \bar{A}) \\ &= \bar{B}C + B\bar{C} \\ &= B \oplus C. \end{aligned}$$

(ii) Simplify  $F(A, B, C) = \sum m(1, 3, 5, 7, 6)$  using K-Map.

Sol:-

A	BC	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
0	00	01	11	10	
1	0	1	1	0	

$$F(A, B, C) = \sum m(1, 3, 5, 7, 6)$$

$$= m_1 + m_3 + m_5 + m_6 + m_7$$

Fill minterms with '1's.

$$\therefore F(A, B, C) = C + BA$$

$$(iii) F(A, B, C) = \sum(0, 1, 4, 5, 2, 6).$$

Sol:-

	$\bar{A} \bar{B} \bar{C}$	$\bar{B} \bar{C}$	$\bar{B} C$	$B \bar{C}$
$\bar{A}$	0	0	1	1
A	1	1	0	0
	0	1	3	2

$$F(A, B, C) = m_0 + m_1 + m_2 + m_4 + m_5 + m_6.$$

$$\therefore F(A, B, C) = \bar{B} + \bar{C}.$$

$$(iv) f(x, y, z) = \sum m(0, 1, 2, 3, 5).$$

Sol:-

	$\bar{x}yz\bar{z}$	$\bar{y}z$	$y\bar{z}$	$\bar{y}\bar{z}$
x	0	0	1	1
0	1	0	1	1
1	0	1	0	0

$$\therefore f(x, y, z) = \bar{x} + \bar{y}z$$

$$(v) f(x, y, z) = \sum (1, 4, 5, 6).$$

Sol:-

	$\bar{x}yz$	$\bar{y}z$	$\bar{y}z$	$y\bar{z}$	$\bar{y}\bar{z}$
$\bar{x}$	0	0	1	1	2
1	1	0	1	1	0

$$\therefore f(x, y, z) = \bar{y}z + x\bar{z}$$

don't care condition

$x \rightarrow$  don't care symbol, it can be 0 or 1.

$$(vi) y(A, B, C) = \sum m(1, 3, 5) + \sum d(6, 7)$$

Sol:-

	$\bar{A} \bar{B} \bar{C}$	$\bar{B} \bar{C}$	$\bar{B} C$	$B \bar{C}$
$\bar{A}$	0	0	1	1
A	1	1	0	0
	0	1	1	2

$$y(A, B, C) = m_1 + m_3 + m_5 + d_6 + d_7$$

Fill minterms with '1'

Fill don't terms with 'x'

It may be '0' or '1'  
consider  $d_7 = 1$ .

$$\therefore y(A, B, C) = C$$

Ex:- Simplify the following boolean expression and draw logic circuit.

$$(vii) Y(A, B, C) = \Sigma(0, 1, 2, 6)$$

Sol:-

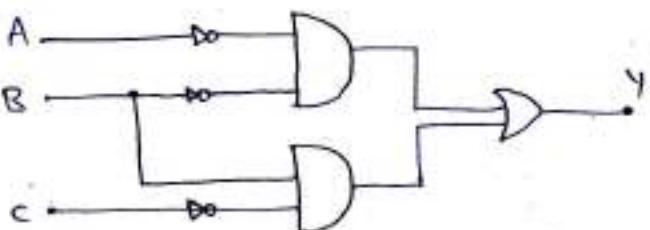
	BC	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
A	00	01	11	10	
$\bar{A}$	0	1		3	1
A	1		4	5	6

$$Y(A, B, C) = \Sigma(0, 1, 2, 6)$$

$$= m_0 + m_1 + m_2 + m_6$$

$$\therefore Y(A, B, C) = \bar{A}\bar{B} + BC.$$

Logic circuit :-



Ex:- Simplify the following boolean expression and draw logic circuits using NAND gate only.

$$(viii) F = \Sigma(0, 1, 5, 6) + \Sigma d(4, 3)$$

Sol:-

	BC	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
A	00	01	11	10	
$\bar{A}$	0	1		3	2
A	1	X	1	5	4

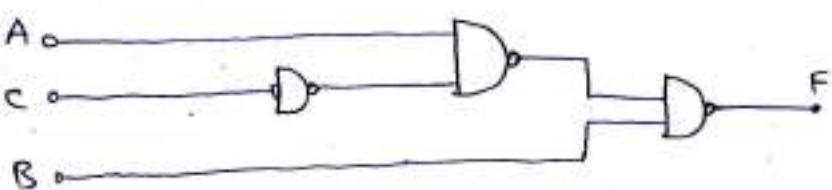
$$F = m_0 + m_1 + m_5 + m_6 + d_3 + d_4$$

Fill minterms with 1's

Fill don't care terms with X's.

$$F = \bar{B} + \text{X} + A\bar{C} = A\bar{C} + \bar{B}$$

Logic circuit:  $F = A\bar{C} + \bar{B} = \overline{\overline{A\bar{C}} + \bar{B}} = \overline{\overline{B} \cdot \overline{A\bar{C}}}$



07/08/2014

(ix)  $F(A, B, C) = \pi M(0, 1, 3, 4, 7)$

Sol:

		$\bar{B}C$	$B\bar{C}$	$\bar{B}\bar{C}$	$B\bar{C}$
		00	01	11	10
A	0	0	0	0	1
	1	0	X	0	0

$F(A, B, C) = \pi M(0, 1, 3, 4, 7)$

$= M_0 \cdot M_1 \cdot M_3 \cdot M_4 \cdot M_7$

Fill Maxterms with 0's.

$\therefore F(A, B, C) = (B+C)(\bar{B}+\bar{C})(A+\bar{C})$

(or)

$(B+C)(\bar{B}+\bar{C})(A+B)$

(x)  $F = \pi M(0, 1, 3, 7) + \pi d(2, 5)$

Sol:

		$\bar{B}C$	$B\bar{C}$	$\bar{B}\bar{C}$	$B\bar{C}$
		00	01	11	10
A	0	0	0	0	X
	1	0	X	0	0

$F = \pi M(0, 1, 3, 7) + \pi d(2, 5)$

$= M_0 \cdot M_1 \cdot M_3 \cdot M_7 + d_2 \cdot d_5$

Fill Maxterms with 0's.

Fill don't terms with X's

consider  $d_2, d_5$  as 0's.

(xi) Given  $F = \sum(0, 3, 4, 7)$ . Find (a) F (b) dual of F (c)  $F'$ .

Sol:

		$\bar{B}C$	$\bar{B}\bar{C}$	$B\bar{C}$	$BC$
		00	01	11	10
A	0	0	1	1	0
	1	1	0	0	1

$F = \sum m(0, 3, 4, 7)$

$= m_0 + m_3 + m_4 + m_7$

Fill minterms with 1's.

 $F \rightarrow \text{POS SOP}$ 

(i)  $F(A, B, C) = \bar{B}\bar{C} + BC = B\oplus C$

(ii) Dual of F is  $(\bar{B}+\bar{C})(B+C)$

$= B \cdot \bar{B} + C \cdot \bar{B} + \bar{C} \cdot B + C \cdot \bar{C}$

$= \bar{B}\bar{C} + B\bar{C} = B\oplus C$

$\therefore \boxed{\text{Dual of } F = \bar{B}\bar{C} + B\bar{C} = B\oplus C}$

(iii)  $F' \rightarrow \text{SOP}$

$\therefore F' = \pi(1, 2, 5, 6)$

$\therefore F' = (\bar{B}+C)(B+C')$

 $(\because F' = \text{dual of } F)$

#### 4-variable K-Map :-

Representation using Minterms

AB\CD	$\bar{C}\bar{D}$	$\bar{C}D$	$C\bar{D}$	$CD$
$\bar{A}\bar{B}$	$\bar{A}\bar{B}\bar{C}\bar{D}$	$\bar{A}\bar{B}\bar{C}D$	$\bar{A}\bar{B}CD$	$\bar{A}\bar{B}C\bar{D}$
$\bar{A}B$	$\bar{A}\bar{B}\bar{C}\bar{D}$	$\bar{A}\bar{B}\bar{C}D$	$\bar{A}\bar{B}CD$	$\bar{A}\bar{B}C\bar{D}$
$A\bar{B}$	$A\bar{B}\bar{C}\bar{D}$	$A\bar{B}\bar{C}D$	$A\bar{B}CD$	$A\bar{B}C\bar{D}$
$AB$	$A\bar{B}\bar{C}\bar{D}$	$A\bar{B}\bar{C}D$	$A\bar{B}CD$	$A\bar{B}C\bar{D}$

AB\CD	00	01	11	10
00	$m_0$	$m_1$	$m_3$	$m_2$
01	$m_4$	$m_5$	$m_7$	$m_6$
11	$m_{12}$	$m_{13}$	$m_{15}$	$m_{14}$
10	$m_8$	$m_9$	$m_{11}$	$m_{10}$

Representation using Maxterms

AB\CD	$C+D$	$C+\bar{D}$	$\bar{C}+\bar{D}$	$\bar{C}+D$
$\bar{A}+B$	$A+B+C+D$	$A+B+C+\bar{D}$	$A+B+\bar{C}+\bar{D}$	$A+B+\bar{C}+D$
$A+\bar{B}$	$A+\bar{B}+C+D$	$A+\bar{B}+C+\bar{D}$	$A+\bar{B}+\bar{C}+\bar{D}$	$A+\bar{B}+\bar{C}+D$
$\bar{A}+\bar{B}$	$\bar{A}+\bar{B}+C+D$	$\bar{A}+\bar{B}+C+\bar{D}$	$\bar{A}+\bar{B}+\bar{C}+\bar{D}$	$\bar{A}+\bar{B}+\bar{C}+D$
$\bar{A}+B$	$\bar{A}+B+C+D$	$\bar{A}+B+C+\bar{D}$	$\bar{A}+B+\bar{C}+\bar{D}$	$\bar{A}+B+\bar{C}+D$

AB\CD	00	01	11	10
00	$M_0$	$M_1$	$M_3$	$M_2$
01	$M_4$	$M_5$	$M_7$	$M_6$
11	$M_{12}$	$M_{13}$	$M_{15}$	$M_{14}$
10	$M_8$	$M_9$	$M_{11}$	$M_{10}$

Ex:- (i)  $F(A, B, C, D) = \sum m(2, 4, 5, 9, 10, 12, 13)$

Sol:-

AB\CD	$\bar{C}\bar{D}$	$\bar{C}D$	$C\bar{D}$	$CD$
$\bar{A}\bar{B}$	00	01	11	10
$\bar{A}B$	0	1	3	2
$A\bar{B}$	4	1	5	6
$AB$	12	1	13	15
$A\bar{B}$	8	1	11	14

$$F(A, B, C, D) = \sum m(2, 4, 5, 9, 10, 12, 13)$$

$$= m_2 + m_4 + m_5 + m_9 + m_{10} + m_{12} + m_{13}$$

Fill Minterms with 1's.

$$\therefore F(A, B, C, D) = BC + A\bar{C}D + \bar{B}CD$$

$$(ii) F(w, x, y, z) = \sum m(1, 3, 5, 7, 8, 9, 10, 11, 12, 14).$$

Sol:

$wx \backslash yz$	$\bar{y}\bar{z}$	$\bar{y}z$	$y\bar{z}$	$yz$
$\bar{w}x$	00	01	11	10
$\bar{w}x$	00	01	11	10
$\bar{w}x$	00	01	11	10
$\bar{w}x$	00	01	11	10
$\bar{w}x$	00	01	11	10
$\bar{w}x$	00	01	11	10

$$\begin{aligned} F(w, x, y, z) &= m_1 + m_3 + m_5 + m_7 + \\ &m_8 + m_9 + m_{10} + m_{11} + \\ &m_{12} + m_{14} \end{aligned}$$

fill minterms with 1's.

$$\begin{aligned} F(\bar{w}, x, y, z) &= z\bar{w} + w\bar{z} + w\bar{x} \\ &= w\bar{x} + w\bar{z} + \bar{w}z \end{aligned}$$

$$(iii) F(A, B, C, D) = \sum (0, 1, 2, 3, 5, 7, 8, 9, 10, 11, 13, 15).$$

Sol:

$AB \backslash CD$	$\bar{C}\bar{D}$	$\bar{C}D$	$CD$	$C\bar{D}$
$\bar{A}\bar{B}$	00	01	11	10
$\bar{A}\bar{B}$	00	01	11	10
$\bar{A}\bar{B}$	00	01	11	10
$\bar{A}\bar{B}$	00	01	11	10
$\bar{A}\bar{B}$	00	01	11	10
$\bar{A}\bar{B}$	00	01	11	10

$$\begin{aligned} F(A, B, C, D) &= m_0 + m_1 + m_2 + m_3 + m_5 + \\ &m_7 + m_8 + m_9 + m_{10} + m_{11} + \\ &m_{13} + m_{15} \end{aligned}$$

fill minterms with 1's.

$$F(A, B, C, D) = D + \bar{B}$$

$$(iv) F = \sum (0, 5, 7, 8, 10, 15) + \sum d(2, 6, 13).$$

Sol:

$AB \backslash CD$	$\bar{C}\bar{D}$	$\bar{C}D$	$CD$	$C\bar{D}$
$\bar{A}\bar{B}$	00	01	11	10
$\bar{A}\bar{B}$	00	01	11	10
$\bar{A}\bar{B}$	00	01	11	10
$\bar{A}\bar{B}$	00	01	11	10
$\bar{A}\bar{B}$	00	01	11	10
$\bar{A}\bar{B}$	00	01	11	10

$$\begin{aligned} F &= \sum (0, 5, 7, 8, 10, 15) + \sum d(2, 6, 13) \\ &= m_0 + m_5 + m_7 + m_8 + m_{10} + m_{15} + \\ &d_2 + d_6 + d_{13}. \end{aligned}$$

fill minterms with 1's.

fill don't terms with X's.

consider  $d_{12}, d_2$  as 1's.

$$\therefore F(A, B, C, D) = BD + \bar{B}\bar{D} = B \oplus D.$$

$$(v) F = \sum (0, 1, 2, 4, 5, 6, 7, 13, 15) + \sum \phi(3, 12, 14, 9, 11).$$

$$\begin{aligned} \text{Sol: } F &= m_0 + m_1 + m_2 + m_4 + m_5 + m_6 + m_7 + m_{13} + m_{15} + \phi_3 + \phi_{12} + \\ &\phi_{14} + \phi_9 + \phi_{11} \end{aligned}$$

fill minterms with 1's.

fill  $\phi$ 's considering  $\phi_2, \phi_{11}$  as 1's.

$AB \backslash CD$	$\bar{C}\bar{D}$	$\bar{C}D$	$C\bar{D}$	$CD$
$\bar{A}\bar{B}$	00	01	11	10
$\bar{A}B$	01	11	X <sub>3</sub>	1 <sub>2</sub>
$A\bar{B}$	11	X <sub>4</sub>	1 <sub>5</sub>	1 <sub>6</sub>
$AB$	10	X <sub>8</sub>	X <sub>9</sub>	X <sub>10</sub>

$$\therefore F(A, B, C, D) = \bar{A} + D.$$

(or)

$$\bar{A} + B.$$

$$(vi) F = \sum m(1, 5, 8, 15, 13, 14) + \sum d(6, 7).$$

Sol:-

$AB \backslash CD$	$\bar{C}\bar{D}$	$\bar{C}D$	$C\bar{D}$	$CD$
$\bar{A}\bar{B}$	00	01	11	10
$\bar{A}B$	01	11	X <sub>3</sub>	1 <sub>2</sub>
$A\bar{B}$	11	X <sub>4</sub>	X <sub>5</sub>	X <sub>6</sub>
$AB$	10	1 <sub>8</sub>	1 <sub>9</sub>	1 <sub>10</sub>

$$F = m_1 + m_5 + m_8 + m_{15} + m_{13} + m_{14} + d_6 + d_7$$

Fill 'd' terms with x's  
consider  $d_6, d_7$  as 1s.

$$\therefore F(A, B, C, D) = BD + BC + \bar{A}\bar{C}D + A\bar{B}\bar{C}\bar{D}$$

$$(vii) F = \pi M(0, 4, 7, 10) + \pi d(3, 13)$$

Sol:-

$AB \backslash CD$	$C\bar{D}$	$C\bar{D}$	$\bar{C}\bar{D}$	$\bar{C}D$
$A+B$	00	01	11	10
$A+\bar{B}$	01	0 <sub>4</sub>	0 <sub>7</sub>	1 <sub>10</sub>
$\bar{A}+\bar{B}$	11	X <sub>12</sub>	1 <sub>15</sub>	1 <sub>14</sub>
$\bar{A}+B$	10	1 <sub>8</sub>	1 <sub>9</sub>	1 <sub>10</sub>

$$F = M_0 + M_4 + M_7 + M_{10} + d_3 \cdot d_{13}$$

consider  $d_3$  as 0.

$$\therefore F(A, B, C, D) = (A + C + D)(A + \bar{C} + \bar{D})(\bar{A} + B + \bar{C} + D).$$

(viii)  $F(w,x,y,z) = \pi(0,3,4,7,8,11,12,14) + \pi d(2,6)$ .

Sol:-

	$wz$	$y_2 z$	$y_2 \bar{z}$	$y_1 z$	$y_1 \bar{z}$	$y_0 z$	$y_0 \bar{z}$
$w+x$	00	00	01	11	10	01	10
$w+\bar{x}$	01	01	10	11	11	00	00
$\bar{w}+x$	11	01	10	10	00	00	00
$\bar{w}+x+z$	10	00	00	01	11	10	10

$$F(w,x,y,z) = M_0 \cdot M_3 \cdot M_4 \cdot M_7 \cdot M_8 \cdot M_{11} \\ M_{12} \cdot M_{14} + \phi_2 \cdot \phi_6$$

Fill maxterms with 0's

Fill  $\phi$  terms with X's.

Consider  $\phi_2, \phi_6$  as 0's.

$$\therefore F(w,x,y,z) = (y_2 z)(\bar{x} + \bar{y} + z)(x + \bar{y} + \bar{z})$$

Ques 10/10/14

### 5-variable K-Map :-

Representing using minterms & Maxterms

AB\CDE	000	001	011	010	110	111	101	100
00	0	1	3	2	6	7	5	4
01	8	9	11	10	14	15	13	12
11	24	25	23	26	30	33	29	28
10	16	17	19	18	22	23	21	20

Ex:- Simplify the following boolean expressions using K-map.

(i)  $F = \pi M(0,1,4,5,9,11,12,15,16,17,25,27,28,29,31) + \pi d(20,21,22,30)$

Sol:-

AB\CDE	000	001	011	010	110	111	101	100
AB	00	0	0		3	2	6	7
A+B	01	8	0	0		14	0	15
$\bar{A}+\bar{B}$	11	24	0	0	X	30	0	29
$\bar{A}+B$	10	16	0	19	18	22	X	20

$$\therefore F = (\bar{B} + \bar{E})(B + D)(\bar{A} + \bar{C} + D)$$

(ii)  $F = \sum m(0,2,4,6,9,11,13,15,17,21,25,27,29,31)$

Sol:-

AB		CDE							
		000	001	011	010	110	111	101	100
AB	00	1	0	1	2	1	2	5	1
$\bar{A}B$	01	1	1	11	10	14	15	12	13
AB	11	1	1	24	25	26	27	21	28
$\bar{A}\bar{B}$	10	1	13	14	16	22	23	21	20

DE

$$\therefore F = BE + A\bar{D}E + \bar{A}\bar{B}\bar{D}\bar{E}$$

Note: (i) In K-map pair is a group of two adjacent cells which cancels one variable.

(ii) Quad is a group of 4 adjacent cells which cancels two variables.

(iii) Octet is a group of 8 adjacent cells which cancels three variables.

Ex:- ① Using demorgan's law simplify the following.

$$(i) Y_1 = ((a+b')(c'+d))'$$

$$\begin{aligned} \text{Sol: } Y_1 &= (\overline{(a+b)}(\overline{c}+d))' && (\because \text{demorgan's law} \\ &= (\overline{a+b})+(\overline{c}+d) && \overline{x+y} = \overline{x}\cdot\overline{y} \\ &= (\overline{a}\cdot\overline{b})+(\overline{c}\cdot\overline{d}) && \overline{xy} = \overline{x}+\overline{y} \\ &= (\overline{a}\cdot b)+(c\cdot\overline{d}) \end{aligned}$$

$$\therefore Y_1 = \overline{a}\cdot b + c\cdot\overline{d}$$

$$(ii) Y_2 = (a+b)'$$

$$\begin{aligned} \text{Sol: } Y_2 &= (a+b)' \\ &= \overline{a}\cdot\overline{b} = \overline{a}\cdot b \end{aligned}$$

$$Y_2 = \overline{a}\cdot b$$

33

Ex: ② Simplify the following expressions and implement using 2-level NAND gate.

$$(i) AB' + ABD + ABD' + A'c'd' + A'b'c'$$

$$(\because A+BC = (A+B)(A+C))$$

Sol:  $AB' + AB(D+D') + A'c'd' + A'b'c'$

$$= AB' + AB + A'c'd' + A'b'c'$$

$$= A + A'c'd' + A'b'c' = (A+A')(A+c'd') + A'b'c'$$

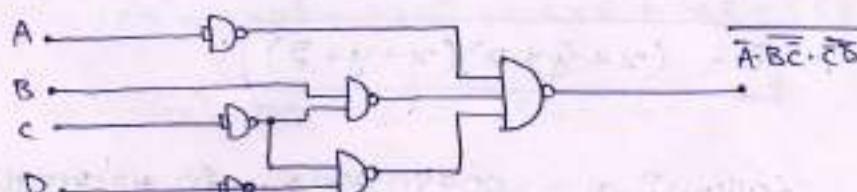
$$= (A+A'c'd' + A'b'c') = A + c'd' + A'b'c'$$

$$= (A+A')(A+b'c') + c'd'$$

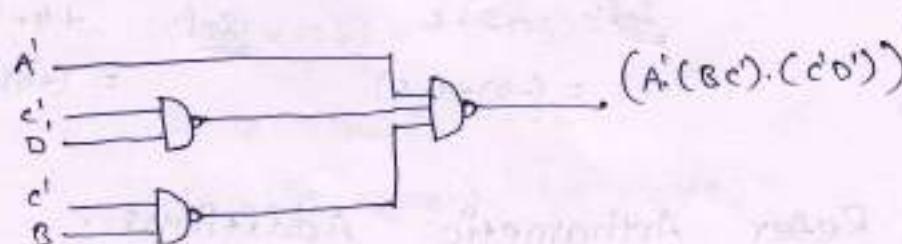
$$= A + b'c' + c'd'$$

$$= ((A+b'c'+c'd'))'$$

$$= \overline{\overline{A} \cdot \overline{B} \overline{c} \cdot \overline{c} \overline{b}}$$



(or)



Note:-

If variables - n

$\therefore$  Total number of Maxterm =  $2^n$

" " " Minterms =  $2^n$

" " " Boolean expressions =  $2^n$ .

Ex-③ The minimum number of NAND gates required to implement  $A + AB' + ABC'$ .

Sol:  $A(1 + \bar{B} + B\bar{C})$       ( $\because 1 + \text{anything} = 1$ )  
 $= A$        $\overbrace{\quad \quad \quad}^A$

$\therefore$  the no. of NAND gates required  $\approx 0$ .

④ find complement of  $F = \bar{x}y\bar{z} + \bar{x}\bar{y}z$

Sol:  $F = \bar{x}y\bar{z} + \bar{x}\bar{y}z$   
complement of  $F$  is  $\bar{F} = \overline{\bar{x}y\bar{z} + \bar{x}\bar{y}z}$   
 $= \overline{\bar{x}y\bar{z}} \cdot \overline{\bar{x}\bar{y}z}$   
 $= (\bar{\bar{x}} + \bar{y} + \bar{z})(\bar{\bar{x}} + \bar{y} + \bar{z})$   
 $= (x + \bar{y} + z)(x + y + \bar{z})$   
 $\therefore \boxed{\bar{F} = (x + \bar{y} + z)(x + y + \bar{z})}$

⑤ Perform the following operations in 2's compliment form.

(i) +6 -3

Sol: +6 -3  
 $= (+6) + (-3)$

arithmetic.

(ii) -2 -6

Sol: -2 -6  
 $= (-2) + (-6)$

(iii) +4 -7

Sol: +4 -7  
 $= (+4) + (-7)$

Refer Arithmetic Additions.

Page no: - 11

⑥ Determine the value of Base  $x$ .

34

(i)  $(211)_x = (152)_8$  convert both numbers into decimal.

Sol:  $2x^2 + 1x^1 + 1x^0 = 1 \times 8^2 + 5 \times 8^1 + 2 \times 8^0$

$$\Rightarrow 2x^2 + x + 1 = 64 + 40 + 2$$

$$\therefore 2x^2 + x - 105 = 0$$

$$2x^2 + 15x - 14x - 105 = 0$$

$$2x(x-7) + 15(x-7) = 0$$

$$(x-7)(2x+15) = 0$$

$$\therefore x = 7 \text{ (or)} \frac{-15}{2}$$

It can't be '-ve'

$$\therefore \boxed{x = 7}$$

(ii)  $(193)_x = (623)_8$

Sol:  $1x^2 + 9x^1 + 3x^0 = 6 \times 8^2 + 2 \times 8^1 + 3 \times 8^0$

$$\Rightarrow x^2 + 9x + 3 = 384 + 16 + 3$$

$$\therefore x^2 + 9x - 400 = 0$$

$$x^2 + 25x - 16x - 400 = 0$$

$$x(x+25) - 16(x+25) = 0$$

$$\therefore x = 16 \text{ (or)} -25$$

It can't be '-ve'

$$\therefore \boxed{x = 16}$$

(iii)  $(\sqrt{41})_x = 5_{10}$

Sol:  $(\sqrt{41})_x = 5$

$$(\sqrt{41})^2 = 5^2$$

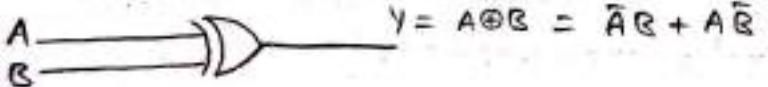
$$\therefore (41)_x = 25$$

$$4x^2 + 1x^0 = 25$$

$$4x = 24$$

$$\therefore \boxed{x = 6}$$

## EXOR function :-



A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

## EXOR properties :-

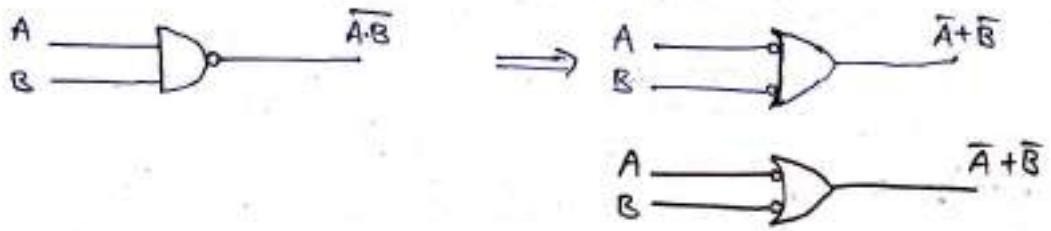
1.  $A \oplus A = 0$
2.  $A \oplus \bar{A} = 1$
3.  $A \oplus 1 = \bar{A}$  (Inverter)
4.  $A \oplus 0 = A$  (Non-Inverter)

## Alternative gate representation :-

### 1. NOT Gate :-



### 2. NAND Gate



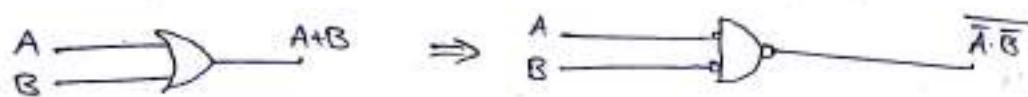
### 3. NOR Gate



### 4. ANd Gate



### 5. OR gate

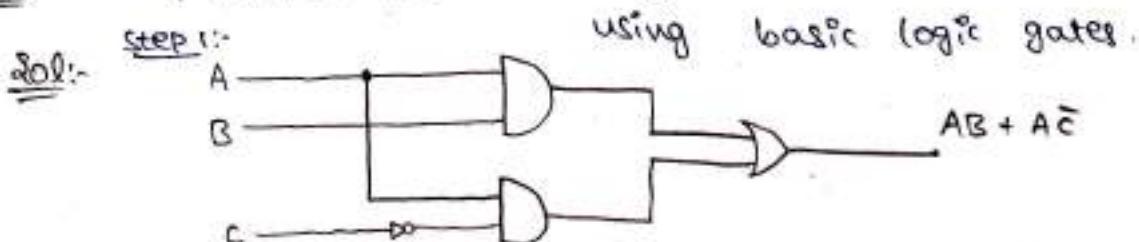


conversion of AND, OR, NOT logic to NAND logic  
using graphical procedure :-

Rules:-

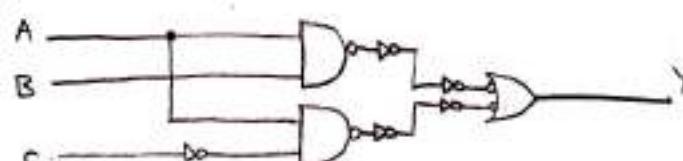
- (i) Draw logic circuit using logic circuit w/ basic gates.
- (ii) If it is an AND gate place a bubble at output side.
- (iii) If it is an OR gate place a bubble at input side.
- (iv) Place a NOT gate where we placed the bubble.
- (v) Replace bubbled ~~AND~~ by using NAND.
- (vi) Eliminate double inverters.

Ex:-  $y = AB + AC$  using NAND logic.

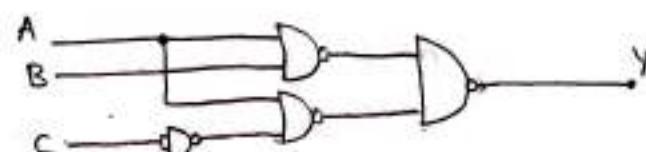


Graphical method.

Step 2:-



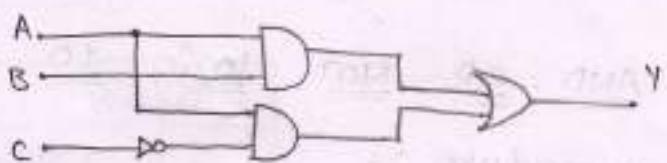
Step 3:-



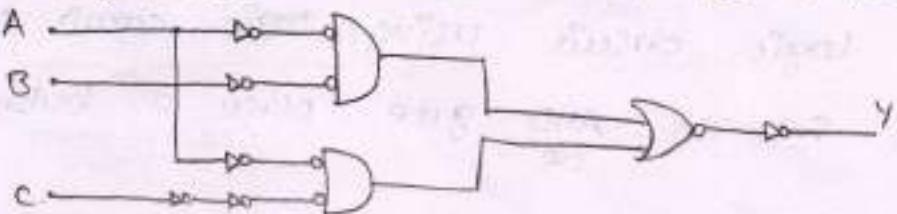
Ex:-  $Y = AB + A\bar{C}$  using NOR gate.

Sol:-

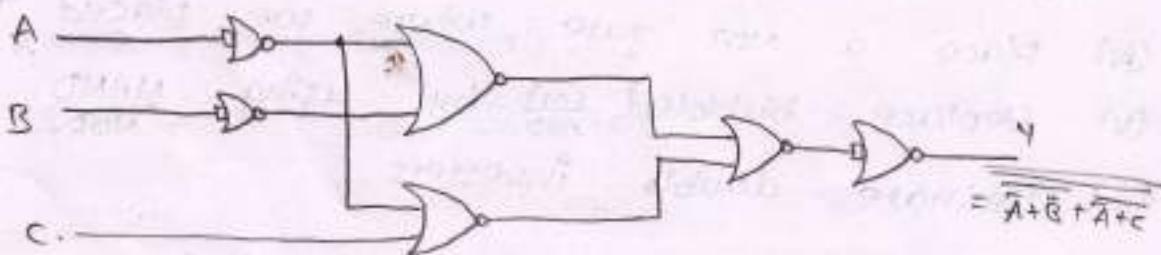
Step 1:-



Step 2:- In OR gate place a bubble at o/p side & place a NOT gate.  
In AND gate place a bubble at i/p side & place a NOT gate.



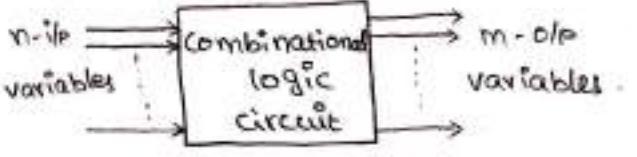
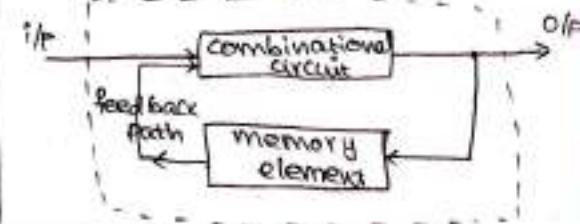
Step 3:- Replace bubble AND with by using NOR.



COMBINATIONAL LOGIC

there are two types of digital logic circuits.

- combinational circuit
- sequential circuit

combinational circuits	sequential circuit
<ul style="list-style-type: none"> <li>* o/p depends on present i/p's.</li> <li>* contains no internal memory</li> <li>* It consists of logic gates</li> <li>* Block diagram.</li> </ul> 	<ul style="list-style-type: none"> <li>* o/p depends on present i/p's &amp; past o/p's.</li> <li>* contains internal memory</li> <li>* It consists of logic gates &amp; memory elements</li> <li>* Block diagram</li> </ul> 

### Design procedure of a combinational circuit :-

Step 1:- From the specifications of the circuit, determine the required number of i/p's & o/p's and assign a symbol to each.

Step 2:- Derive the truth table that defines the required relation b/w i/p's & o/p's.

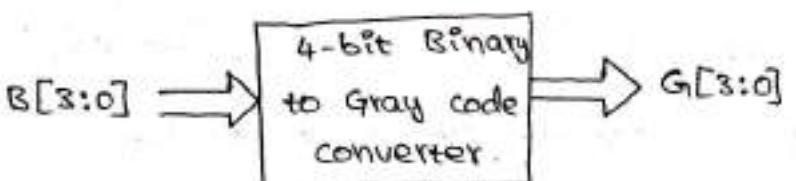
Step 3:- Obtain the simplified boolean functions for each o/p as a function of the i/p variables.

Step 4:- Draw the logic diagram and verify the correctness of the design.

\* code converters :-

\* construct 4-bit binary to Gray code converter

4-bit Binary to Gray code converter converts  
4-bit binary to 4 bit Gray code.



$B[3:0] \rightarrow$  4-bit Binary number

$$B = B_3 B_2 B_1 B_0$$

$G[3:0] \rightarrow$  4-bit Gray number

$$G = G_3 G_2 G_1 G_0$$

Truth Table :-

	i/p (Binary)				o/p (Gray)			
	$B_3$	$B_2$	$B_1$	$B_0$	$G_3$	$G_2$	$G_1$	$G_0$
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	1	1	0
12	1	1	0	0	1	0	1	0
13	1	1	0	1	1	0	1	1
14	1	1	1	0	1	0	0	1
15	1	1	1	1	1	0	0	0

But In Gray numbers, 1's are minterms, 0's are Maxterms

$$G_0 = \Sigma m(1, 2, 5, 6, 9, 10, 13, 14)$$

$$G_1 = \Sigma m(2, 3, 4, 5, 10, 11, 12, 13)$$

$$G_2 = \Sigma m(4, 5, 6, 7, 8, 9, 10, 11)$$

$$G_3 = \Sigma m(8, 9, 10, 11, 12, 13, 14, 15)$$

K-Map Simplification :-

For  $G_0$  :-

		$B_3 B_2 \backslash B_1 B_0$		00	01	11	10
		00	01	11	10		
00	0	1		3	1		
	1	4	5	7	6		
01	12	13	15	14			
	8	9	11	10			

$$\therefore G_0 = \bar{B}_1 B + B_1 \bar{B}_0 = B_1 \oplus B_0$$

For  $G_1$  :-

		$B_3 B_2 \backslash B_1 B_0$		00	01	11	10
		00	01	11	10		
00			1	1			
	1	1					
01	1	1					
11							
10			1	1			

$$\therefore G_1 = B_1 \bar{B}_2 + \bar{B}_1 B_2 = B_1 \oplus B_2$$

For  $G_2$  :-

		$B_3 B_2 \backslash B_1 B_0$		00	01	11	10
		00	01	11	10		
00							
	1	1	1	1			
01	1	1	1	1			
11							
10	1	1	1	1			

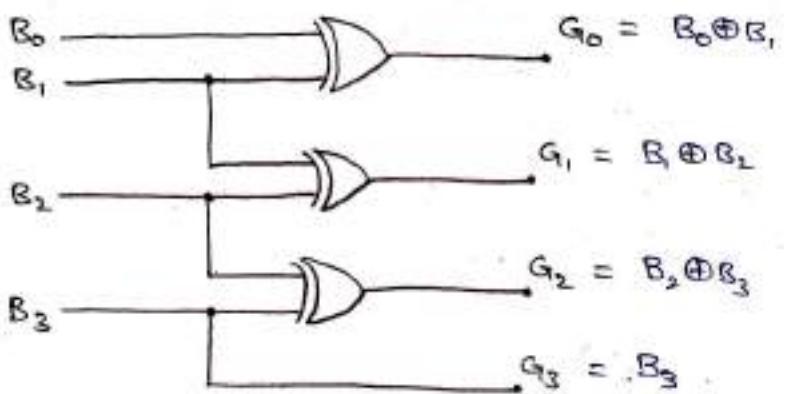
$$\therefore G_2 = B_3 \bar{B}_2 + \bar{B}_3 B_2 = B_3 \oplus B_2$$

For  $G_3$  :-

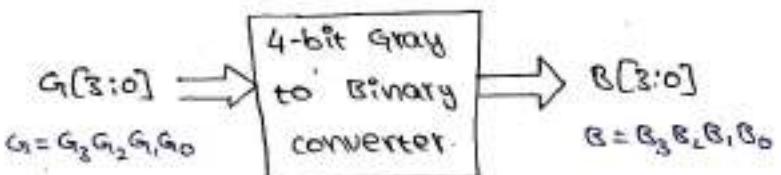
		$B_3 B_2 \backslash B_1 B_0$		00	01	11	10
		00	01	11	10		
00							
	1	1	1	1			
01	1	1	1	1			
11	1	1	1	1			
10	1	1	1	1			

$$\therefore G_3 = 1$$

## logic circuits :-



\* Gray to Binary converter :-



Truth table :-

	i/p (Gray)	o/p (Binary)
	$G_3\ G_2\ G_1\ G_0$	$B_3\ B_2\ B_1\ B_0$
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 0
3	0 0 1 1	0 0 1 1
4	0 1 1 0	0 1 0 0
5	0 1 1 1	0 1 0 1
6	0 1 0 1	0 1 1 0
7	0 1 0 0	0 1 1 1
8	1 1 0 0	1 0 0 0
9	1 1 0 1	1 0 0 1
10	1 1 1 1	1 0 1 0
11	1 1 1 0	1 0 1 1
12	1 0 1 0	1 1 0 0
13	1 0 1 1	1 1 0 1
14	1 0 0 1	1 1 1 0
15	1 0 0 0	1 1 1 1

	i/p (Gray)	o/p (Binary)
	$G_3\ G_2\ G_1\ G_0$	$B_3\ B_2\ B_1\ B_0$
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 1
5	0 1 0 1	0 1 1 0
6	0 1 1 0	0 1 0 0
7	0 1 1 1	0 1 0 1
8	1 0 0 0	1 1 1 1
9	1 0 0 1	1 1 1 0
10	1 0 1 0	1 1 0 0
11	1 0 1 1	1 1 0 1
12	1 1 0 0	1 0 0 0
13	1 1 0 1	1 0 0 1
14	1 1 1 0	1 0 1 1
15	1 1 1 1	1 0 1 0

$$B_0 = \sum m(1, 3, 5, 7, 9, 11, 13, 15) = \sum m(1, 2, 4, 7, 8, 11, 13, 14)$$

$$B_1 = \sum m(2, 3, 6, 7, 10, 11, 14, 15) = \sum m(2, 3, 4, 5, 8, 9, 14, 15)$$

$$B_2 = \sum m(4, 5, 6, 7, 12, 13, 14, 15) = \sum m(4, 5, 6, 7, 8, 9, 10, 11)$$

$$B_3 = \sum m(8, 9, 10, 11, 12, 13, 14, 15) = \sum m(8, 9, 10, 11, 12, 13, 14, 15)$$

K-Map Simplification:-

For  $B_0$  :-

$G_3 G_2$	$G_1 G_0$	$\bar{G}_1 \bar{G}_0$	$G_1 \bar{G}_0$	$\bar{G}_1 G_0$
00	0	1	2	3
01	4	5	6	7
11	8	9	10	11
10	12	13	14	15

$$\therefore B_0 = (G_0 \oplus G_1) \oplus (G_2 \oplus G_3)$$

For  $B_1$  :-

$G_3 G_2$	$G_1 G_0$	00	01	11	10
00				(1)	(1)
01		(1)	(1)		
11				(1)	(1)
10		(1)	(1)		

$$\therefore B_1 = G_1 \oplus G_2 \oplus G_3$$

For  $B_2$  :-

$G_3 G_2$	$G_1 G_0$	00	01	11	10
00					
01		(1)	(1)	(1)	(1)
11					
10		(1)	(1)	(1)	(1)

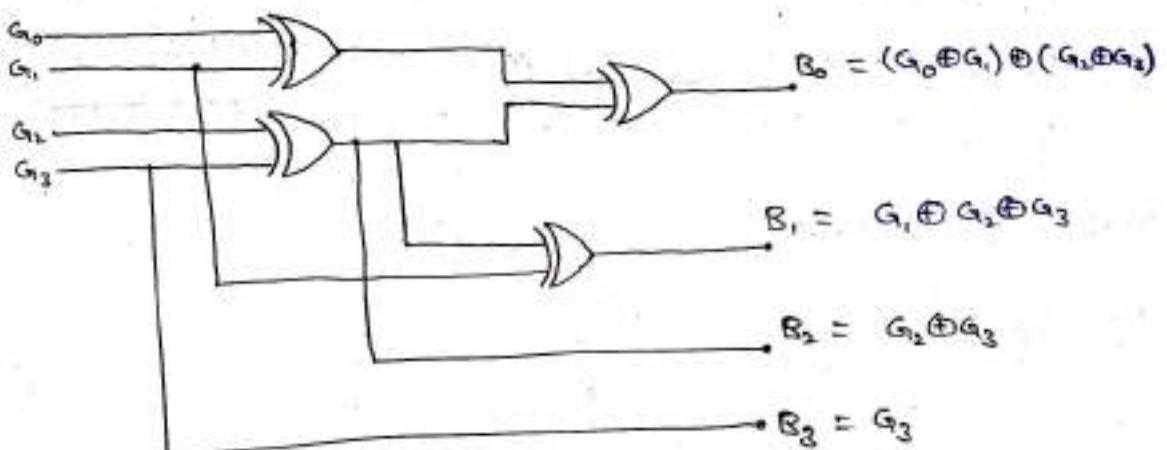
$$\therefore B_2 = G_2 \bar{G}_3 + \bar{G}_2 G_3$$

For  $B_3$  :-

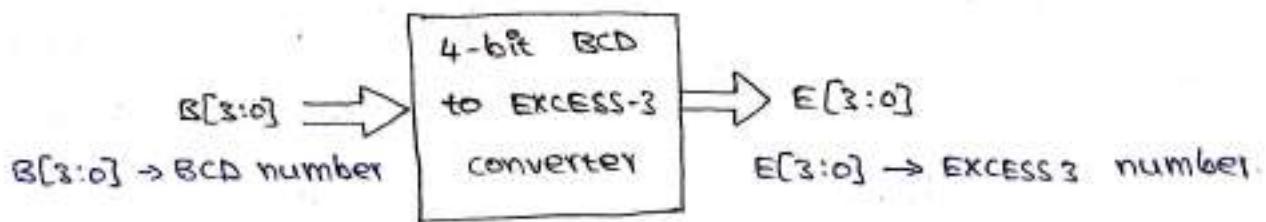
$G_3 G_2$	$G_1 G_0$	00	01	11	10
00					
01					
11		(1)	(1)	(1)	(1)
10		(1)	(1)	(1)	(1)

$$\therefore B_3 = G_3$$

logic circuit :-



\* BCD to EXCESS-3 converter :-



Truth table :-

i/p (BCD)	o/p (EXCESS-3)			
B <sub>3</sub> B <sub>2</sub> B <sub>1</sub> B <sub>0</sub>	E <sub>3</sub>	E <sub>2</sub>	E <sub>1</sub>	E <sub>0</sub>
0 0 0 0	0	0	1	1
0 0 0 1	0	1	0	0
0 0 1 0	0	1	0	1
0 0 1 1	0	1	1	0
0 1 0 0	0	1	1	1
0 1 0 1	1	0	0	0
0 1 1 0	1	0	0	1
0 1 1 1	1	0	1	0
1 0 0 0	1	0	1	1
1 0 0 1	1	1	0	0

BCD numbers are 0-9

∴ consider 10-15 as don't care.

$$E_0 = \sum m(0, 2, 4, 6, 8) + \sum d(10, 11, 12, 13, 14, 15)$$

$$E_1 = \sum m(0, 3, 4, 7, 8) + \sum d(10, 11, 12, 13, 14, 15)$$

$$E_2 = \sum m(1, 2, 3, 4, 9) + \sum d(10, 11, 12, 13, 14, 15)$$

$$E_3 = \sum m(5, 6, 7, 8, 9) + \sum d(10, 11, 12, 13, 14, 15)$$

K-map Simplification :-

For  $E_0$  :-

$B_3 B_2 \backslash B_1 B_0$	00	01	11	10
$\bar{B}_3 \bar{B}_2$	1			1
$\bar{B}_3 B_2$	1			1
$B_3 B_2$	X	X	X	X
$B_3 \bar{B}_2$	1		X	X

$$\therefore E_0 = \bar{B}_0$$

For  $E_1$  :-

$B_3 B_2 \backslash B_1 B_0$	00	01	11	10
00	1		1	
01	1		1	
11	X	X	X	X
10	1		X	X

$$\therefore E_1 = B_1 \oplus B_0 = \bar{B}_0 \bar{B}_1 + B_0 B_1$$

For  $E_2$  :-

$B_3 B_2 \backslash B_1 B_0$	00	01	11	10
00		1	1	1
01	1			
11	X	X	X	X
10		1	X	X

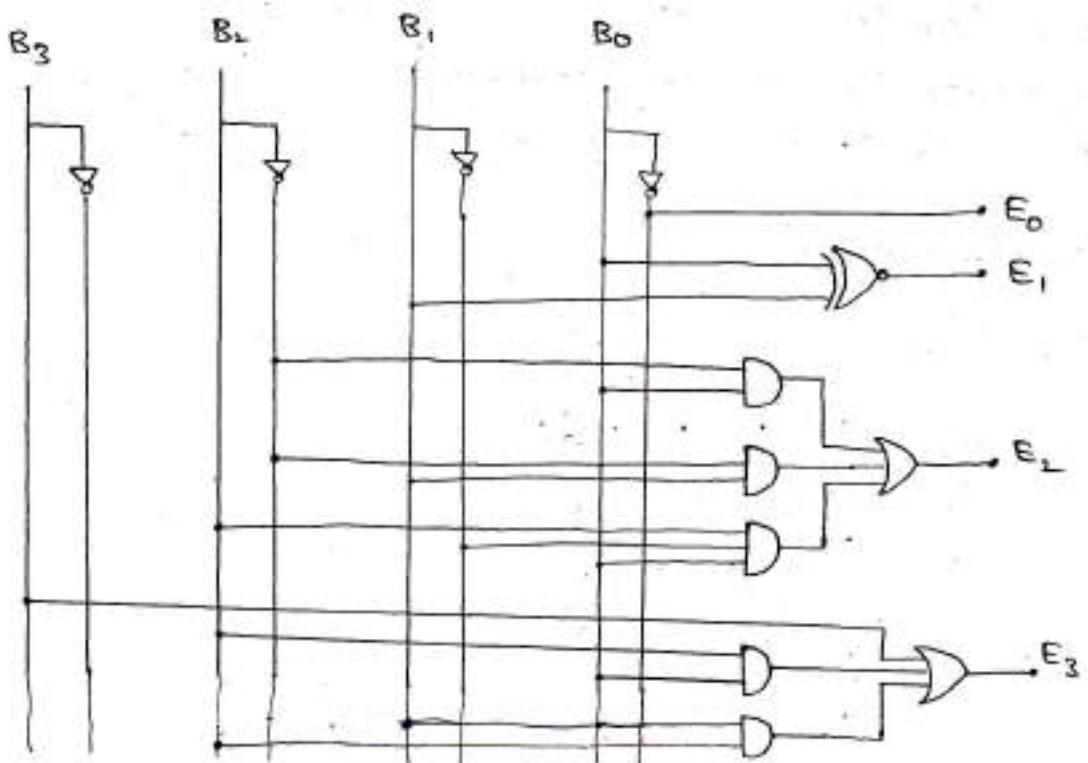
$$\therefore E_2 = \bar{B}_2 B_0 + \bar{B}_2 B_1 + B_2 \bar{B}_1 \bar{B}_0$$

For  $E_3$  :-

$B_3 B_2 \backslash B_1 B_0$	00	01	11	10
00				
01		1	1	1
11	X	X	X	X
10	1	1	X	X

$$\therefore E_3 = B_3 + B_2 B_0 + B_2 B_1$$

logic circuit :-



2d08/2014

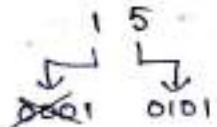
\* Excess-3 to BCD converter :-

Truth table :-

I/P (Excess-3)				O/P (BCD)			
$E_3$	$E_2$	$E_1$	$E_0$	$B_3$	$B_2$	$B_1$	$B_0$
0	0	0	0	x	x	x	x
0	0	0	1	x	x	x	x
0	0	1	0	x	x	x	x
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	1	0	0	0	1	1
0	1	1	1	0	1	0	0
1	0	0	0	0	1	0	1
1	0	0	1	0	1	1	0
1	0	1	0	0	1	1	1
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	1
1	1	0	1	x	x	x	x
1	1	1	0	x	x	x	x
1	1	1	1	x	x	x	x

Binary to BCD converter :-

Ex:-



for 10-15 no.'s 3 MSB's are 0's.

∴ neglect 3 MSB's.

Truth table :-

I/P (Binary)	O/P (BCD)
B <sub>3</sub> B <sub>2</sub> B <sub>1</sub> B <sub>0</sub>	D <sub>4</sub> D <sub>3</sub> D <sub>2</sub> D <sub>1</sub> D <sub>0</sub>
0 0 0 0	0 0 0 0 0
0 0 0 1	0 0 0 0 1
0 0 1 0	0 0 0 1 0
0 0 1 1	0 0 0 1 1
0 1 0 0	0 0 1 0 0
0 1 0 1	0 0 1 0 1
0 1 1 0	0 0 1 1 0
0 1 1 1	0 0 1 1 1
1 0 0 0	0 1 0 0 0
1 0 0 1	0 1 0 0 1
1 0 1 0	1 0 0 0 0
1 0 1 1	1 0 0 0 1
1 1 0 0	1 0 0 1 0
1 1 0 1	1 0 0 1 1
1 1 1 0	1 0 1 0 0
1 1 1 1	1 0 1 0 1

BCD to Binary converter :-

input (BCD)	O/P (Binary)
D <sub>4</sub> D <sub>3</sub> D <sub>2</sub> D <sub>1</sub> D <sub>0</sub>	B <sub>3</sub> B <sub>2</sub> B <sub>1</sub> B <sub>0</sub>
0 0 0 0 0	0 0 0 0
0 0 0 0 1	0 0 0 1
0 0 0 1 0	0 0 1 0
0 0 0 1 1	0 0 1 1
0 0 1 0 0	0 1 0 0
0 0 1 0 1	0 1 0 1
0 0 1 1 0	0 1 1 0

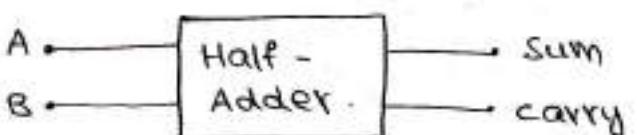
D <sub>4</sub> D <sub>3</sub> D <sub>2</sub> D <sub>1</sub> D <sub>0</sub>	B <sub>3</sub> B <sub>2</sub> B <sub>1</sub> B <sub>0</sub>
0 1 0 0 0	1 0 0 0
0 1 0 0 1	1 0 0 1
1 0 0 0 0	1 0 1 0
1 0 0 0 1	1 0 1 1
1 0 0 1 0	1 1 0 0
1 0 0 1 1	1 1 0 1
1 0 1 0 0	1 1 1 0
1 0 1 0 1	1 1 1 1

\* ADDERS:- Adder is a digital circuit which performs addition operations.

### 1. Half Adders:-

The logic circuit which performs addition of 2 bits is called a "Half Adder". It contains two binary i/p (Augend & Addend) and two binary o/p (sum & carry).

Block diagram:-

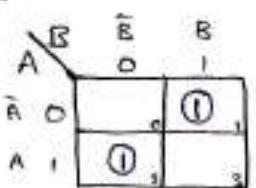


Truth table:-

i/p's		o/p's	
A	B	sum	carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

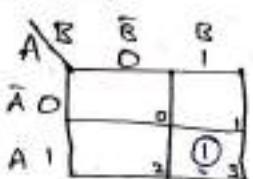
K-Map Simplification:-

For sum:-



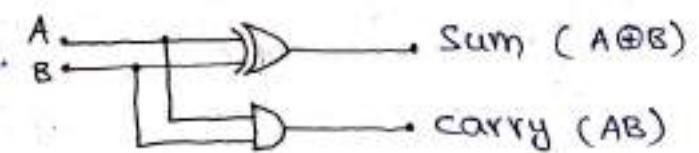
$$\begin{aligned}\therefore \text{sum} &= \bar{A}\bar{B} + A\bar{B} \\ &= A \oplus B\end{aligned}$$

For carry:-



$$\therefore \text{carry} = AB$$

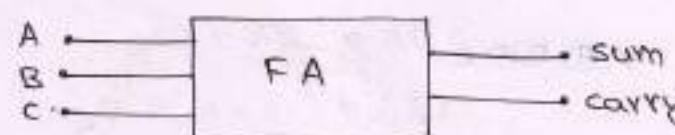
Logic circuit:-



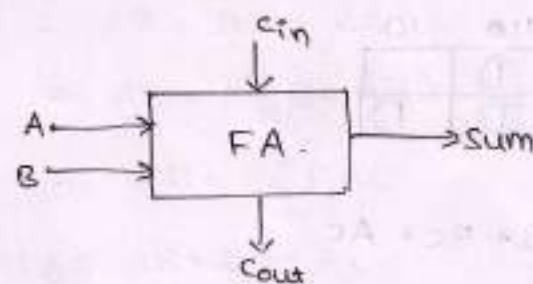
## 2. Full Adder :-

Full Adder is a combinational circuits that forms the arithmetic sum of 3-bit input bits. It consists of 3 i/p's & 2 o/p. The third input is a  $C_{in}$  which represents the carry from the previous significant position.

### Block diagram :-



(or)



### Truth table :-

i/p's			o/p's	
A	B	C	Sum	carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

## K-Map simplifications:-

For sum

A \ BC	00	01	11	10
0	0	1	0	1
1	1	0	0	0

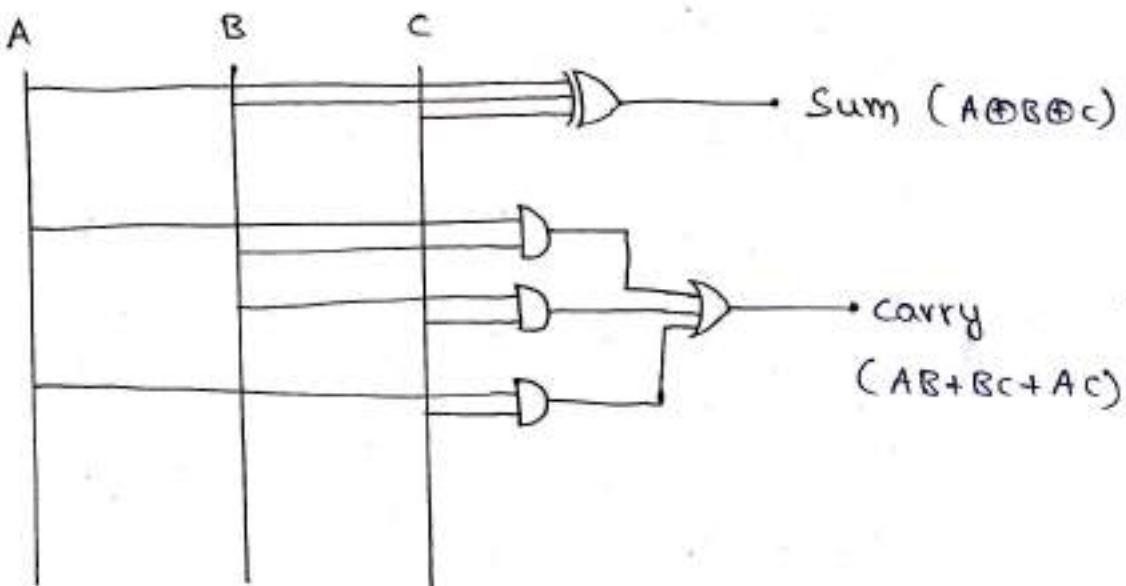
$$\begin{aligned}
 \text{sum} &= \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC \\
 &= \bar{A}(\bar{B}C + B\bar{C}) + A(\bar{B}\bar{C} + BC) \\
 &= \bar{A}(B \oplus C) + A(\overline{B \oplus C}) \quad (\because \bar{x}y + x\bar{y} = x \oplus y) \\
 &= A \oplus B \oplus C.
 \end{aligned}$$

For carry

A \ BC	00	01	11	10
0	0	0	1	0
1	0	1	0	1

$$\therefore \text{carry} = AB + BC + AC.$$

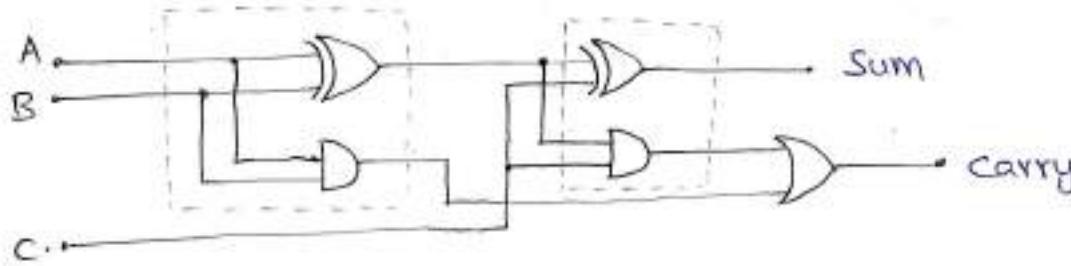
## Logic circuit:-



Note:- A full adder can also be implemented by using two half-adders and 1 OR gate.

## Full Adder using Half Adders:-

7



$$\therefore \text{Sum} = A \oplus B \oplus C.$$

$$\text{Carry} = AB + (A \oplus B)C$$

$$= AB + (\bar{A}B + A\bar{B})C$$

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

$$= B(A + \bar{A}C) + A\bar{B}C$$

$$= B((A + \bar{A})(A + C)) + A\bar{B}C$$

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

$$= AB + C(B + A\bar{B}) = AB + C((B+A)(B+\bar{B}))$$

$$= AB + BC + AC.$$

$$\therefore \text{Carry} = AB + BC + AC.$$

## 3. Parallel Adder (Binary Adder):-

It is a combinational circuit which performs addition of  $2, n$ -bit numbers.

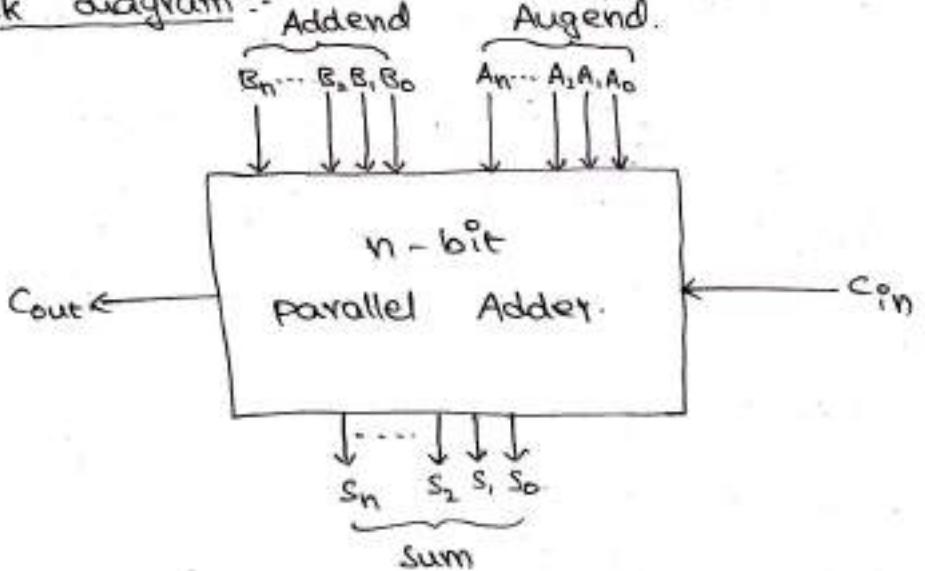
A & B are  $n+1$  bit numbers.

$$A \rightarrow A_n \dots A_3 A_2 A_1 A_0$$

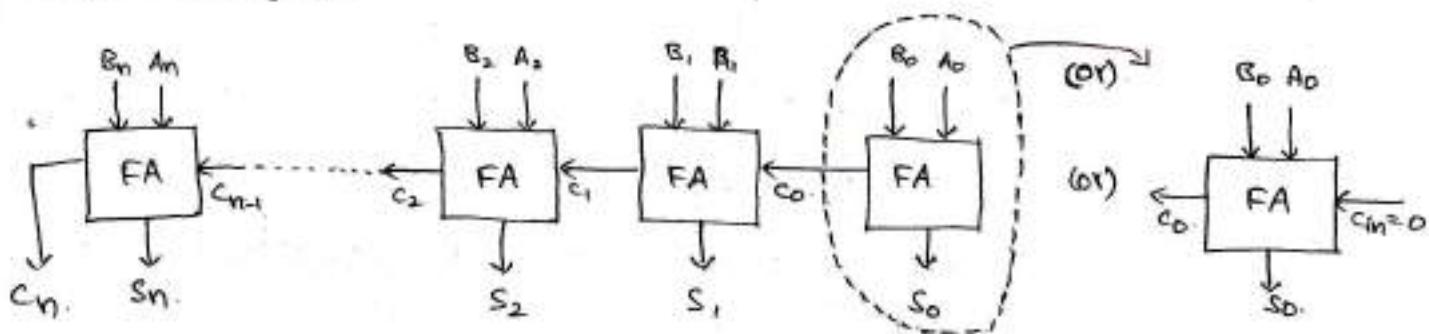
$$B \rightarrow B_n \dots B_3 B_2 B_1 B_0$$

$$\begin{array}{r}
 c_{n-1} \quad c_2 \quad c_1 \quad c_0 \\
 A_n \dots A_2 \quad A_1 \quad A_0 \rightarrow \text{Augend} \\
 + B_n \dots B_2 \quad B_1 \quad B_0 \rightarrow \text{Addend} \\
 \hline
 c_n \quad s_n \quad s_2 \quad s_1 \quad s_0 \rightarrow \text{Sum}
 \end{array}$$

Block diagram :-



Logic diagram :-



26/08/2014

Ex:- Design 4-bit parallel adder using full-adders.

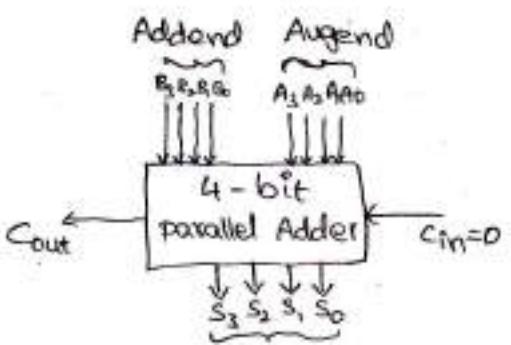
Sol:- A, B  $\rightarrow$  4-bit binary numbers.

$$A = A_3 A_2 A_1 A_0$$

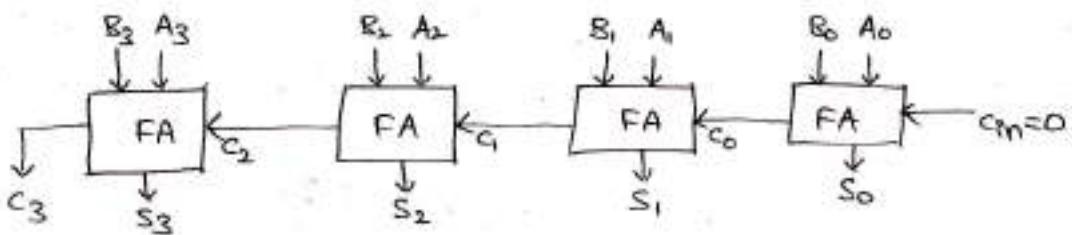
$$B = B_3 B_2 B_1 B_0$$

$$\begin{array}{cccc}
 c_2 & c_1 & c_0 & c_{in} \xrightarrow{=0} \\
 A_3 & A_2 & A_1 & A_0 \longrightarrow \text{Augend} \\
 B_3 & B_2 & B_1 & B_0 \longrightarrow \text{Addend} \\
 \hline
 c_3 & S_3 & S_2 & S_1 & S_0 \longrightarrow \text{Result.}
 \end{array}$$

Block diagram:-



Logic diagram :-

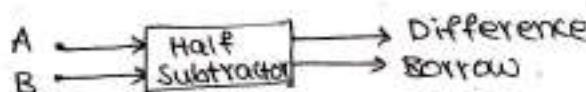


- \* Subtractors:- Subtractor is a digital circuit which performs subtraction operations.

### 1. Half Subtractor :-

It is a combinational circuit which performs subtraction of two binary bits. It's has two inputs (minuend & subtrahend) and two outputs (difference & borrow).

Block diagram :-



Truth table :-

i/p's		o/p's	
A	B	Diff.	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

K-Map Simplification :-

for Difference :-

	B	B	B
A	0	0	1
Ā	0	1	0

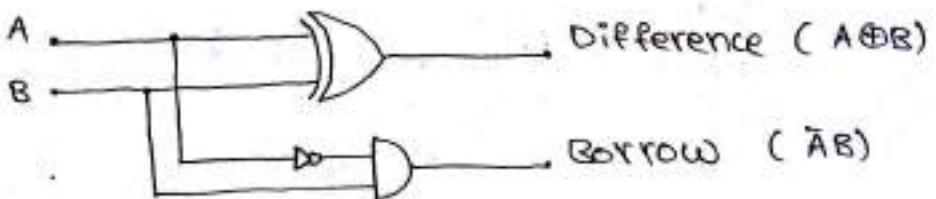
$$\therefore \text{Diff.} = \overline{AB} + \overline{A}\overline{B}$$

for Borrow :-

	B	B
A	0	0
Ā	1	0

$$\therefore \text{Borrow} = \overline{AB}$$

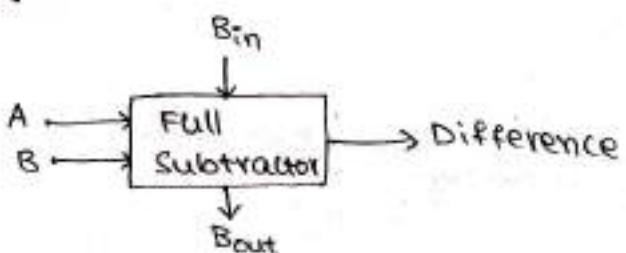
## Logic circuit :-



## 2. Full Subtractor :-

Full Subtractor is a combinational circuit which performs subtraction of 2 binary bits by considering borrow of the previous stage. It has 3 inputs and 2 outputs.

### Block diagram:-



### Truth table :-

i/p's			o/p's	
A	B	B <sub>in</sub>	Diff	B <sub>out</sub>
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

## K-Map Simplifications :-

For Difference :-

		B Bin	$\bar{B}$ Bin	$\bar{B} \bar{B}_{in}$	$B \bar{B}_{in}$	$BB_{in}$
		00	01	11	10	
		A 0	0	1	1	0
		A 1	1	0	0	1

$$\therefore \text{Difference} = A \oplus B \oplus B_{in}$$

$$\begin{aligned}
 \therefore \text{Difference} &= \bar{A} \bar{B} \bar{B}_{in} + \bar{A} B \bar{B}_{in} + A \bar{B} \bar{B}_{in} + A B \bar{B}_{in} \\
 &= \bar{A} (\bar{B} \bar{B}_{in} + B \bar{B}_{in}) + A (\bar{B} \bar{B}_{in} + B \bar{B}_{in}) \\
 &= \bar{A} (B \oplus B_{in}) + A (B \oplus B_{in}) = \bar{A} (B \oplus B_{in}) + A (\bar{B} \oplus \bar{B}_{in}) \\
 &= A \oplus B \oplus B_{in} = A \oplus B \oplus B_{in}.
 \end{aligned}$$

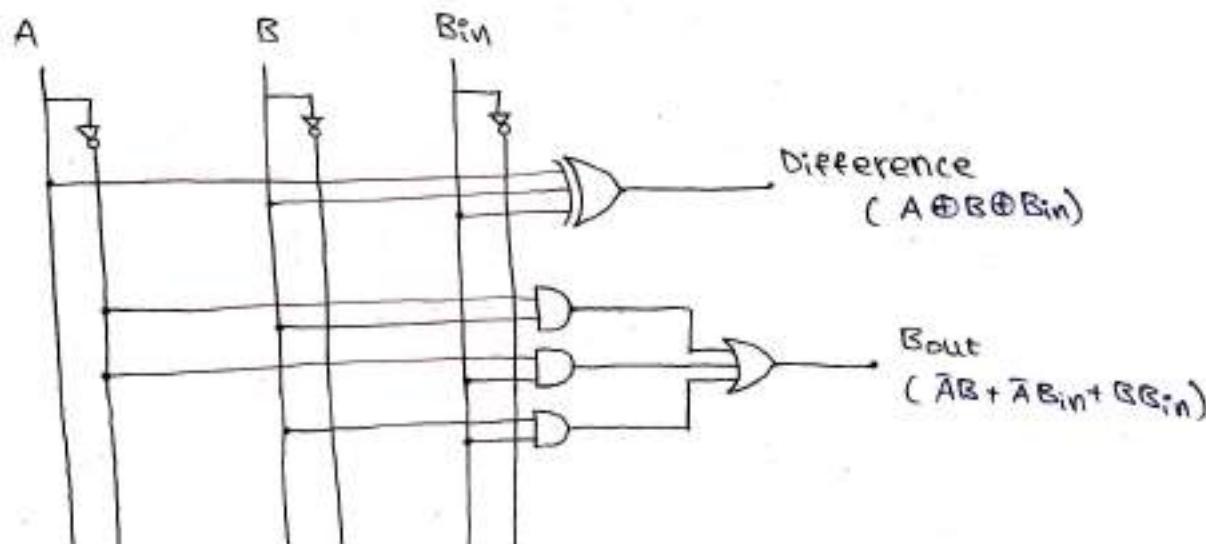
For Bout :-

		B Bin	$\bar{B}$ Bin	$\bar{B} \bar{B}_{in}$	$B \bar{B}_{in}$	$BB_{in}$
		00	01	11	10	
		A 0	0	1	0	1
		A 1			1	

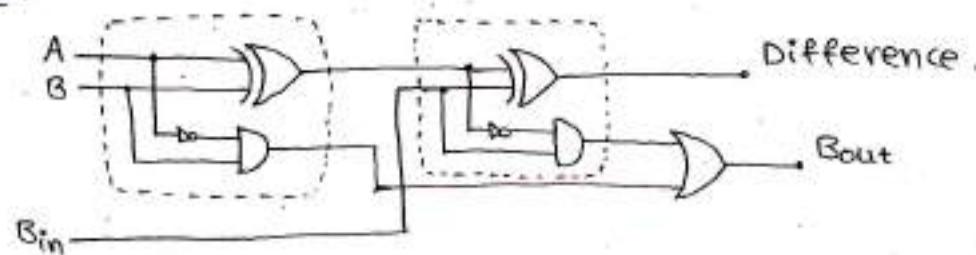
$$\therefore \text{Bout} = \bar{A} B + \bar{A} B_{in} + B \bar{B}_{in}$$

$$\therefore \text{Bout} = \bar{A} B + \bar{A} B_{in} + B \bar{B}_{in}$$

Logic circuit :-



Note :- A full subtractor can also be implemented with two half-subtractors and 1 OR gate.



$\therefore$  Difference =  $A \oplus B \oplus B_{in}$ .

$$\begin{aligned}\therefore B_{out} &= (\overline{A \oplus B})_{Bin} + \overline{AB} \\ &= (\overline{A}\overline{B} + AB)_{Bin} + \overline{AB} \\ &= \overline{A}\overline{B}_{Bin} + AB_{Bin} + \overline{AB} \\ &= \overline{A}\overline{B}_{Bin} + B(A\overline{B} + \overline{A}) \quad (\because A+BC = (A+B)(A+C)) \\ &= \overline{A}\overline{B}_{Bin} + B((A+\overline{A})(B+\overline{A})) \quad (\because A+\overline{A} = 1) \\ &= \overline{A}\overline{B}_{Bin} + BB_{Bin} + B\overline{A} \\ &= B_{Bin}(\overline{A}\overline{B} + B) + B\overline{A} \quad (\because A+BC = (A+B)(A+C)) \\ &= B_{Bin}((\overline{A}+B)(B+\overline{A})) + B\overline{A} \quad (\because B+\overline{B} = 1) \\ &= \overline{A}B_{Bin} + BB_{Bin} + B\overline{A},\end{aligned}$$

Ex: Design 4-bit parallel subtractor using full adders.

Sol:  $A, B \rightarrow$  4-bit binary no.'s.

$$A = A_3 \ A_2 \ A_1 \ A_0$$

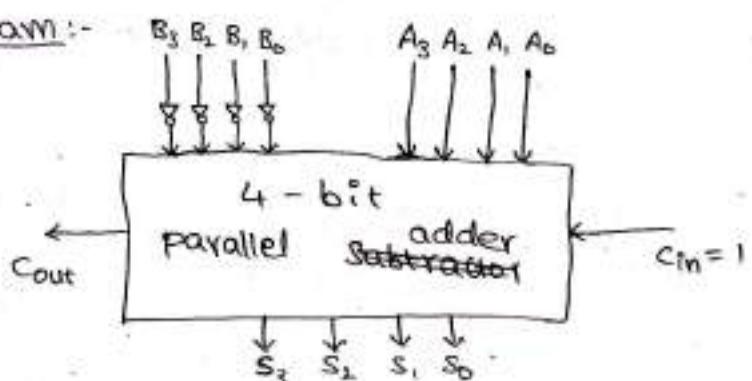
$$B = B_3 \ B_2 \ B_1 \ B_0$$

$$A - B = A + 2's \text{ complement of } B$$

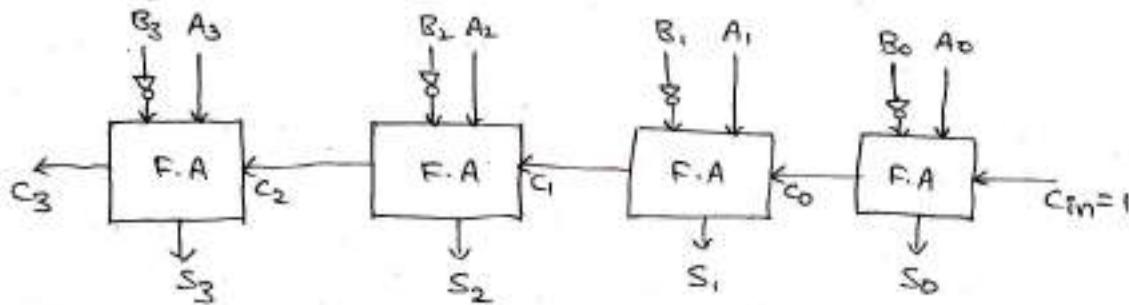
$$= A + 1's \text{ comp. of } B + 1$$

$$\begin{array}{ccccccc} c_2 & c_1 & c_0 & & & & \\ A_3 & A_2 & A_1 & A_0 & \longrightarrow & \text{minuend} \\ \overline{B}_3 & \overline{B}_2 & \overline{B}_1 & \overline{B}_0 & \longrightarrow & 1's \text{ comp. of } B \\ + & & & & 1 & \longrightarrow & 1 \\ \hline c_3 & s_3 & s_2 & s_1 & s_0 & & \end{array}$$

Block diagram:-



Logic diagram :-



Parallel adder / subtractor (or) Binary adder / subtractor :-

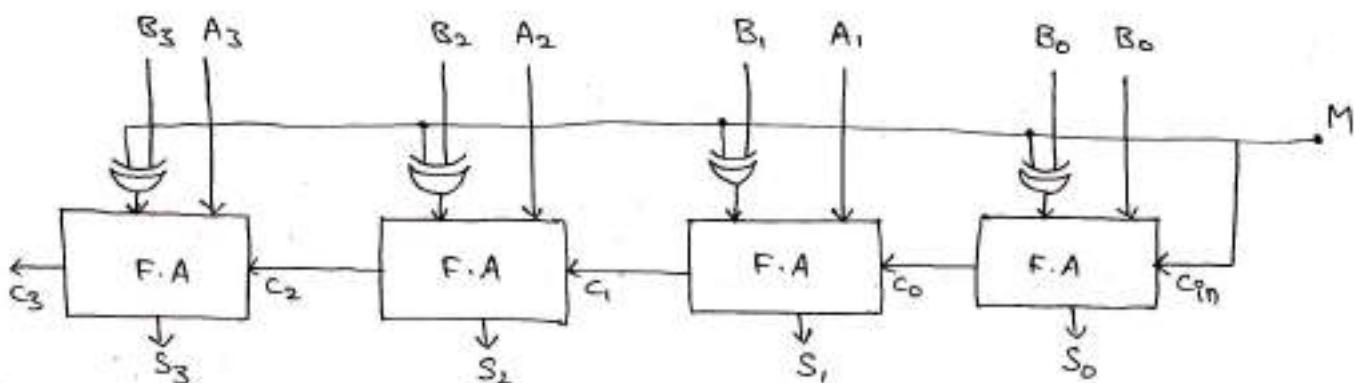


Fig:- 4-bit parallel adder / subtractor.

where M - controlling input.

$$\begin{aligned} \text{if } M=0 \text{ then output} &= A_3 A_2 A_1 A_0 + B_3 B_2 B_1 B_0 + \text{Cin} \quad (\because A \oplus 0 = A) \\ &= A_3 A_2 A_1 A_0 + B_3 B_2 B_1 B_0 \quad (\because \text{Cin} = 0) \\ &= A + B. \end{aligned}$$

$\therefore$  (i.e) This circuit act as adder.

$$\begin{aligned} \text{if } M=1 \text{ then output} &= A_3 A_2 A_1 A_0 + \bar{B}_3 \bar{B}_2 \bar{B}_1 \bar{B}_0 + \text{Cin} \quad (\because A \oplus 1 = \bar{A}) \\ &= A_3 A_2 A_1 A_0 + \bar{B}_3 \bar{B}_2 \bar{B}_1 \bar{B}_0 + 1 \quad (\because M=1 \Rightarrow \text{Cin}=0) \\ &= A - B \end{aligned}$$

$\therefore$  (i.e) This circuit act as subtractor.

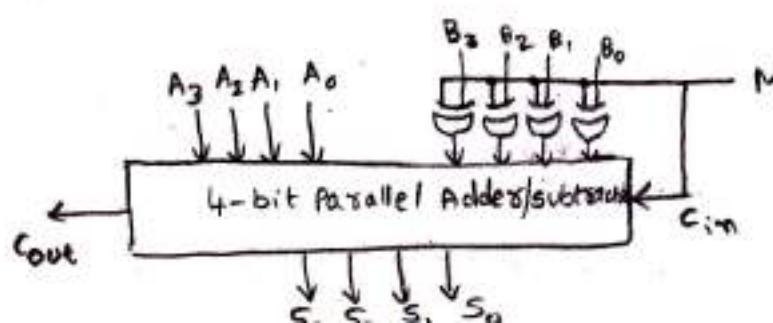


fig: Block diag

## ~~Look~~ ahead carry generator :-

\* The sum and carry outputs of any stage of n-bit parallel adder cannot be produced until the input carry occurs. This leads to time delay in addition process. This delay is known as "carry propagation delay."

\* LACG is used to eliminate the carry propagation delay.

\* LAC addition uses two functions.

- (i) carry generate ( $G_i$ )
- (ii) carry propagate ( $P_i$ )

$$\begin{array}{r}
 c_2 \quad c_1 \quad c_0 \\
 A_2 \quad A_1 \quad A_0 \\
 + B_2 \quad B_1 \quad B_0 \\
 \hline
 c_3 \quad s_2 \quad s_1 \quad s_0
 \end{array}$$

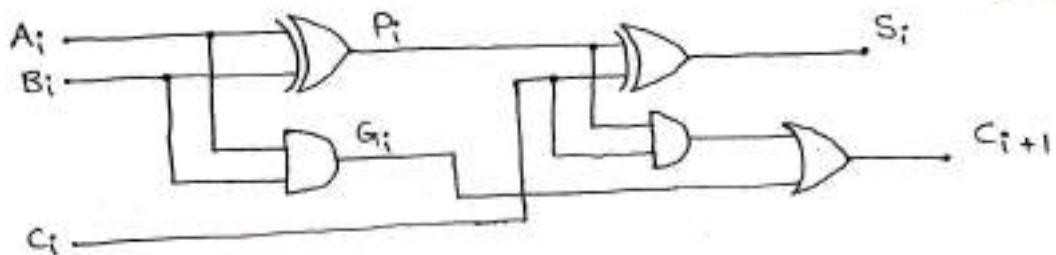


Fig: FA using 2 HAs and OR gate with P & G shown.

$$\therefore P_i = A_i \oplus B_i$$

$$G_i = A_i \cdot B_i$$

using P & G, sum & carry can be expressed as

$$S_i = P_i \oplus C_i$$

$$C_{i+1} = G_i + (P_i \cdot C_i)$$

$\therefore C_0$  = i/p carry.

$$C_1 = G_0 + P_0 \cdot C_0$$

$$C_2 = G_1 + P_1 \cdot C_1 = G_1 + P_1(G_0 + P_0 \cdot C_0)$$

$$= G_1 + P_1 \cdot G_0 + P_1 \cdot P_0 \cdot C_0$$

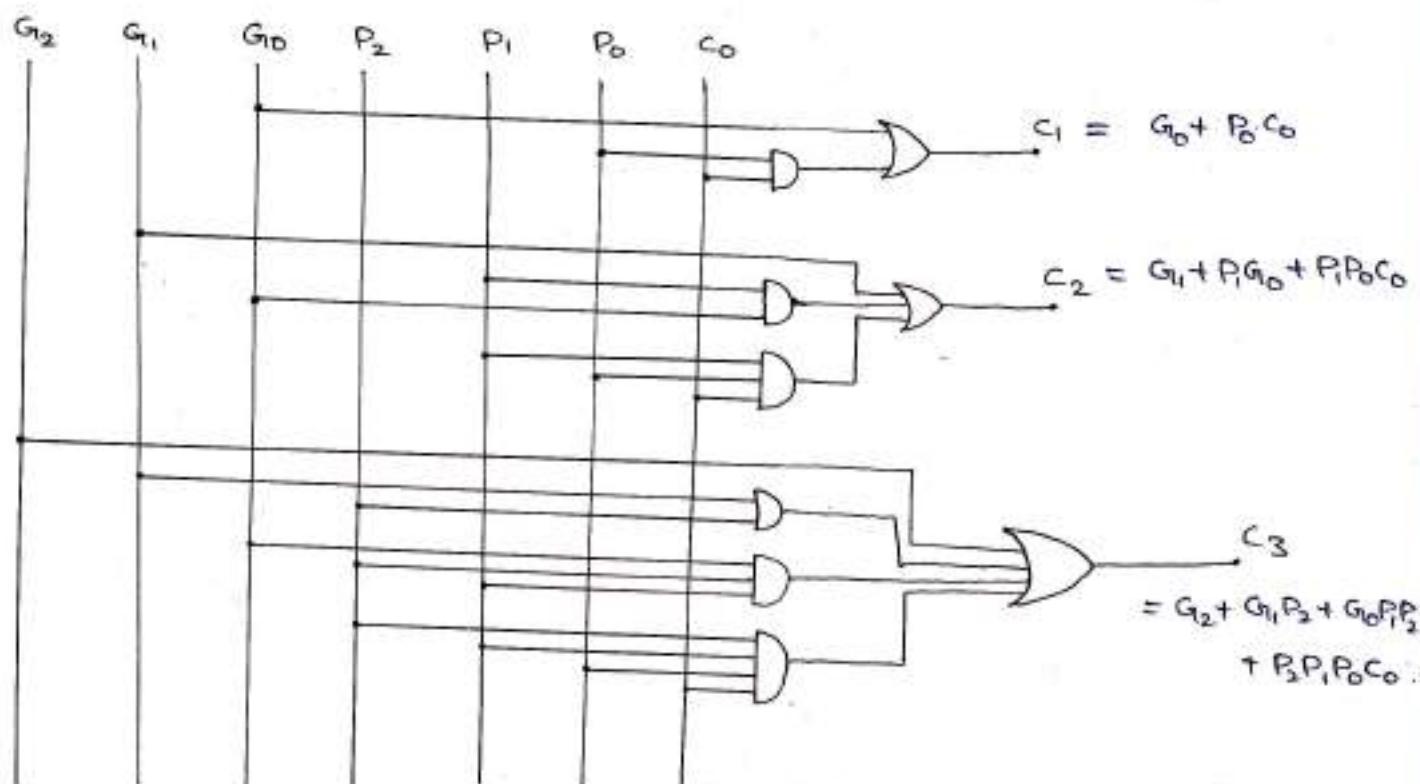
$$\therefore C_3 = G_2 + P_2 C_2$$

$$= G_2 + P_2 (G_1 + P_1 G_0 + P_1 P_0 C_0)$$

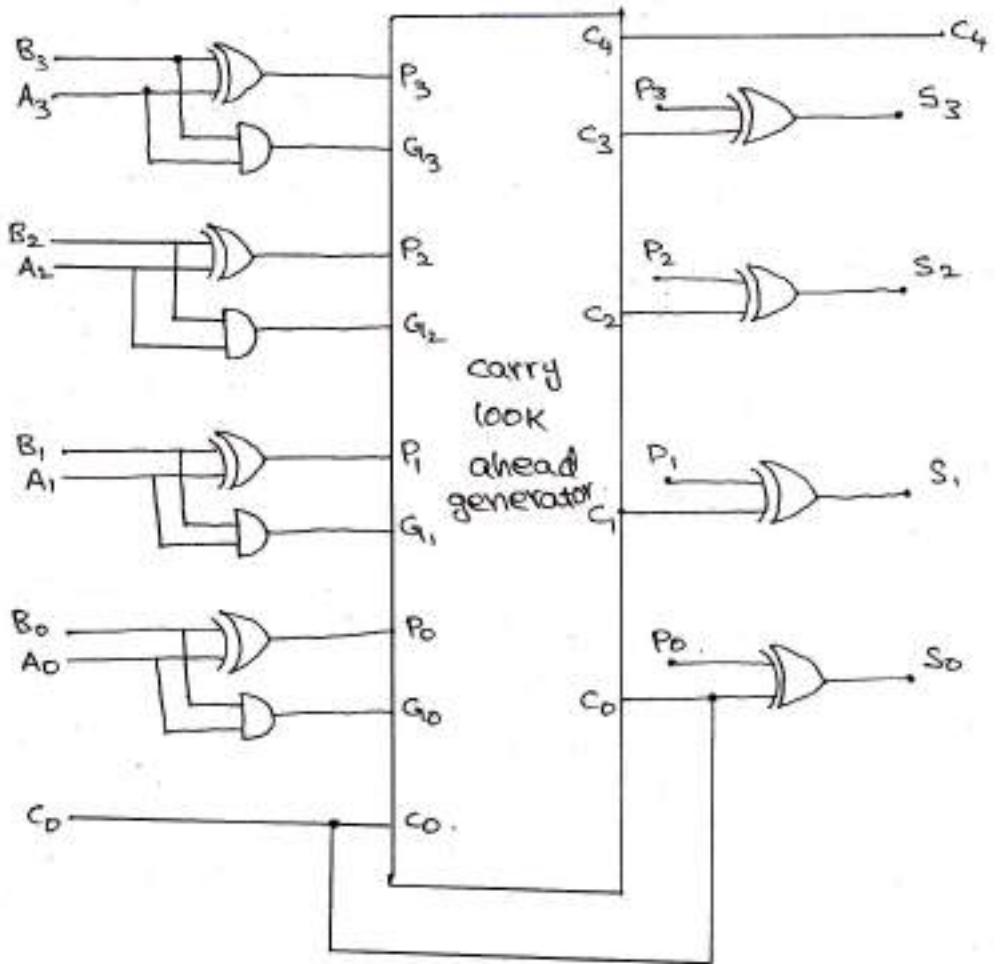
$$= G_2 + G_1 P_2 + G_0 P_1 P_2 + P_2 P_1 P_0 C_0$$

From the above boolean equations (Expressions) it can be seen that  $C_1, C_2, C_3$  propagate at the same time and depends only on  $C_0$ .

3-bit look ahead carry generator logic diagram :-



## 4-bit adder with carry look ahead :-



## \* BCD adder / Decimal adder :-

\* the digital systems handles the decimal no. in the form of BCD number.

\* A BCD adder is a circuit that adds two BCD digits and produces a sum digit also in BCD.

\* To implement BCD adder we required,

- (i) 4-bit binary adder for initial Addition.
- (ii) logic circuit to detect sum is greater than 9 (or) not.
- (iii) 1 more 4-bit binary adder to add  $(0110)_2$ .

to the sum if sum is  $>9$  or carry is 1. 12

Truth table to implement logic circuit to detect sum is  $>9$  or not :-

i/p's				o/p
$S_3$	$S_2$	$S_1, S_0$		$Y$
0	0	00		0
0	0	01		0
0	0	10		0
0	0	11		0
0	1	00		0
0	1	01		0
0	1	10		0
0	1	11		0
1	0	00		0
1	0	01		0
1	0	10		1
1	0	11		1
1	1	00		1
1	1	01		1
1	1	10		1
1	1	11		1

o/p is '1' when i/p is  $>9$ .

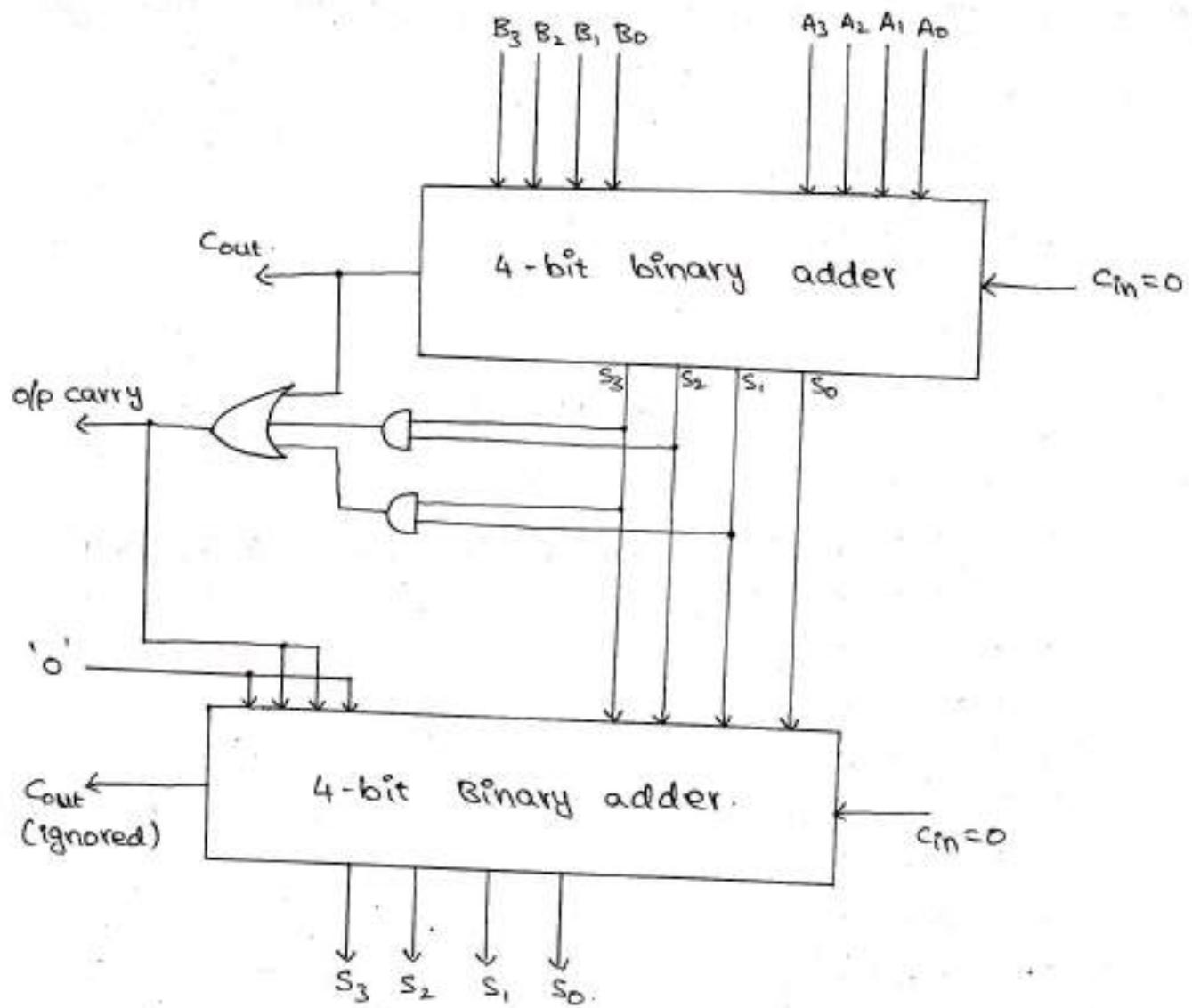
K-Map Simplification :-

for  $y$  :-

$S_3 S_2$	$S_3 S_2 \bar{S}_1 \bar{S}_0$	$\bar{S}_3 S_2 \bar{S}_1 \bar{S}_0$	$\bar{S}_3 S_2 S_1 \bar{S}_0$	$S_3 S_2 S_1 \bar{S}_0$
$\bar{S}_3 \bar{S}_2$	00	01	11	10
00	0	1	1	1
01	4	5	2	6
11	(1) 12 13	1 14 15	(1) 16 17	1 18 19
10	8	9	1 11 12	1 13 14

$$\therefore Y = S_3 S_2 + S_3 S_1$$

Logic diagram :-

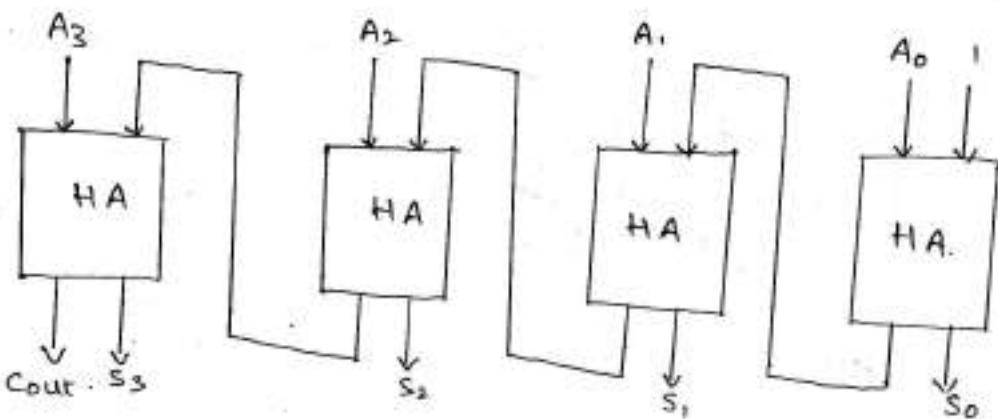


\* 4-bit Binary Incrementer :-

$$A = A_3 A_2 A_1 A_0 \quad \text{4-bit binary no.}$$

$$A + 1 = A_3 A_2 A_1 A_0 + 1$$

$$\begin{array}{r}
 & c_2 & c_1 & c_0 \\
 & A_3 & A_2 & A_1 & A_0 \\
 + & & & & 1 \\
 \hline
 \text{Cout} & S_3 & S_2 & S_1 & S_0
 \end{array}$$



\* 4-bit Binary Decrementer :-

$A = A_3 A_2 A_1 A_0$  4-bit binary number.

$$\begin{aligned}A - 1 &= A_3 A_2 A_1 A_0 - 0001 \\&= A_3 A_2 A_1 A_0 + 2\text{'s comp. of } 0001 \\&= A_3 A_2 A_1 A_0 + 1\text{'s comp. of } 0001 + 1 \\&= A_3 A_2 A_1 A_0 + 1110 + 1\end{aligned}$$

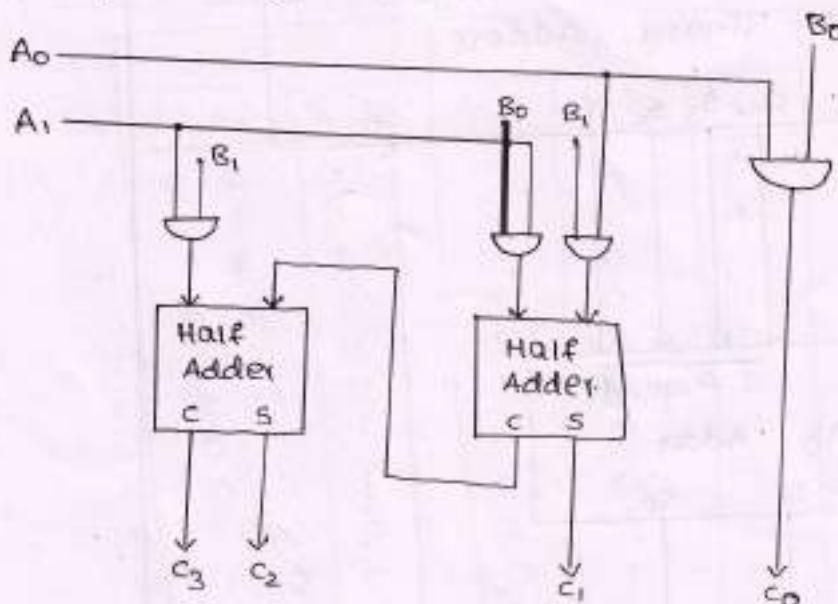
\*\* Binary Multiplier :-

(i) 2x2 Binary Multiplier :-

$B = B_1 B_0$        $A = A_1 A_0$  2-bit binary no.'s.

$$\begin{array}{l}B \times A \Rightarrow \begin{array}{l}B_1 \ B_0 \rightarrow \text{Multiplicand} \\ \times A_1 \ A_0 \rightarrow \text{Multiplier}\end{array} \quad (B \times A = \text{simple multiplication}) \\ \begin{array}{c} A_0 B_1 \ A_0 B_0 \\ \hline A_1 B_1 \ A_1 B_0 \end{array} \quad \left. \begin{array}{l} \text{Partial products} \\ \hline \end{array} \right. \\ \begin{array}{c} c_3 \ c_2 \ c_1 \ c_0 \\ \hline \end{array} \quad \rightarrow \text{Product}\end{array}$$

Logic diagram :-

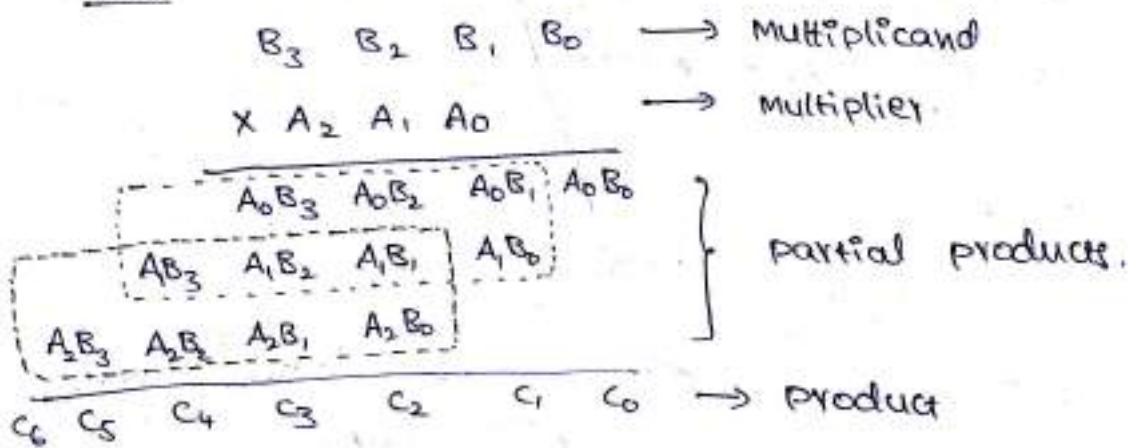


(ii)  $4 \times 3$  Binary Multiplier :-

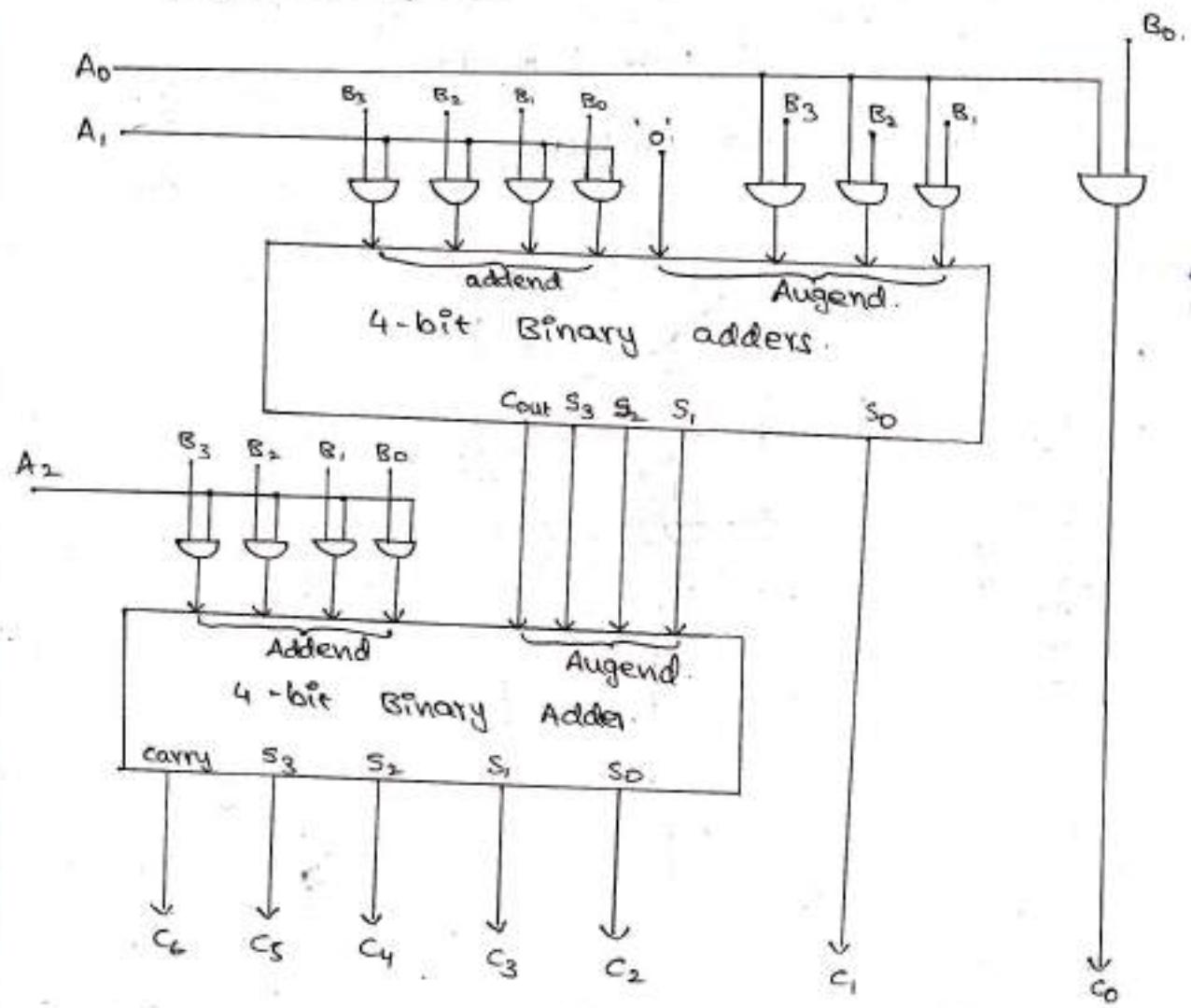
$B = B_3 \ B_2 \ B_1 \ B_0 \rightarrow 4\text{-bit Binary no.}$

$A = A_2 \ A_1 \ A_0 \rightarrow 3\text{-bit Binary no.}$

$B \times A$



Logic diagram :-



16/08/2014

### \* Magnitude comparator (Digital comparator) :-

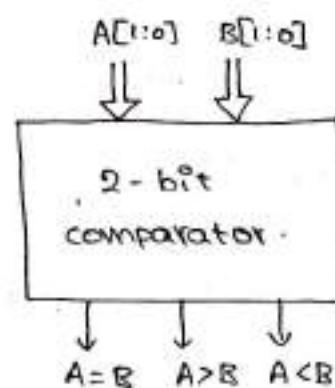
\* A comparator is a special combinational circuit designed primarily to compare the relative magnitudes of two binary numbers.

\* It receives two n-bit numbers, A & B are inputs and produces 3 outputs  $A > B$ ,  $A < B$ ,  $A = B$ .

Design 2-bit comparator using logic gates.

2-bit comparator compares 2, 2-bit binary numbers.

Block diagram:-



A, B are 2 bit Binary numbers.

Truth table:-

i/p's		o/p's				
A <sub>1</sub>	A <sub>0</sub>	B <sub>1</sub>	B <sub>0</sub>	A = B	A > B	A < B
0	0	0	0	1	0	0
1	0	0	1	0	0	1
2	0	0	10	0	0	1
3	0	0	11	0	0	1
4	0	1	00	0	1	0
5	0	1	01	1	0	0
6	0	1	10	0	0	1
7	0	1	11	0	0	1
8	1	0	00	0	1	0
9	1	0	01	0	1	0
10	1	0	10	1	0	0
11	1	0	11	0	0	1
12	1	1	00	0	1	0
13	1	1	01	0	1	0
14	1	1	10	0	1	0
15	1	1	11	1	0	0

$$\begin{aligned}(A=B) &= \sum m(0, 5, 10, 15) \\ (A>B) &= \sum m(4, 8, 9, 12, 13, 14) \\ (A<B) &= \sum m(1, 2, 3, 6, 7, 11)\end{aligned}$$

### K-Map Simplification :-

For  $A=B$  :-

$$(A=B) = \sum m(0, 5, 10, 15)$$

		$A_1 A_0 \setminus B_1 B_0$	00	01	11	10
		00	1			
		01		1		
		11			1	
		10				1

$$\therefore (A=B) = (A_0 \oplus B_0)(A_1 \oplus B_1)$$

$$\begin{aligned}(A=B) &= \bar{A}_1 A_0 \bar{B}_1 \bar{B}_0 + \bar{A}_1 A_0 \bar{B}_1 B_0 + \\ A_1 A_0 B_1 B_0 + A_1 \bar{A}_0 B_1 \bar{B}_0 \\ &= \bar{A}_0 \bar{B}_0 (\bar{A}_1 \bar{B}_1 + A_1 B_1) + A_0 B_0 (\bar{A}_1 \bar{B}_1 + A_1 B_1) \\ &= (\bar{A}_1 \bar{B}_1 + A_1 B_1)(\bar{A}_0 \bar{B}_0 + A_0 B_0).\end{aligned}$$

For  $(A>B)$  :-

$$(A>B) = \sum m(4, 8, 9, 12, 13, 14)$$

		$A_1 A_0 \setminus B_1 B_0$	00	01	11	10
		00				
		01	1			
		11	1	1		1
		10	1	1		

$$(A>B) = A_1 \bar{B}_1 + \bar{B}_1 B_0 A_0 + A_1 A_0 \bar{B}_0$$

$$\therefore (A>B) = A_1 \bar{B}_1 + A_1 A_0 \bar{B}_0 + A_0 \bar{B}_1 \bar{B}_0$$

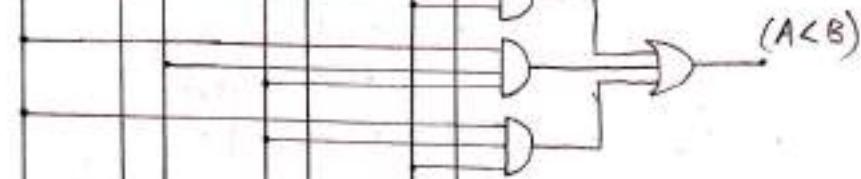
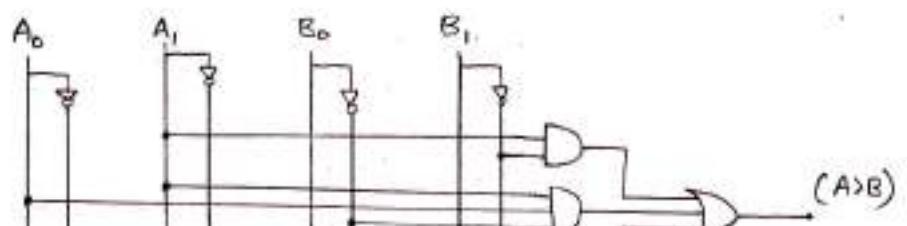
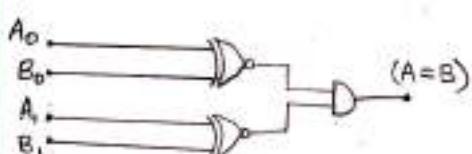
$$(A<B) = \sum m(1, 2, 3, 6, 7, 11)$$

		$A_1 A_0 \setminus B_1 B_0$	00	01	11	10
		00		1	1	1
		01			1	1
		11				
		10				1

$$(A<B) = \bar{A}_1 B_1 + B_0 \bar{A}_1 \bar{A}_0 + B_1 B_0 \bar{A}_0$$

$$\therefore (A<B) = \bar{A}_1 B_1 + \bar{A}_1 \bar{A}_0 B_0 + \bar{A}_0 B_1 B_0$$

### Logic Circuit :-



26/08/2014

\* Decoder :- A Decoder is a multiple input, multiple o/p circuit, which converts coded inputs into coded outputs, where the i/p & o/p codes are different.

\* The i/p code generally has fewer bits than their o/p codes.

Logic diagram :-

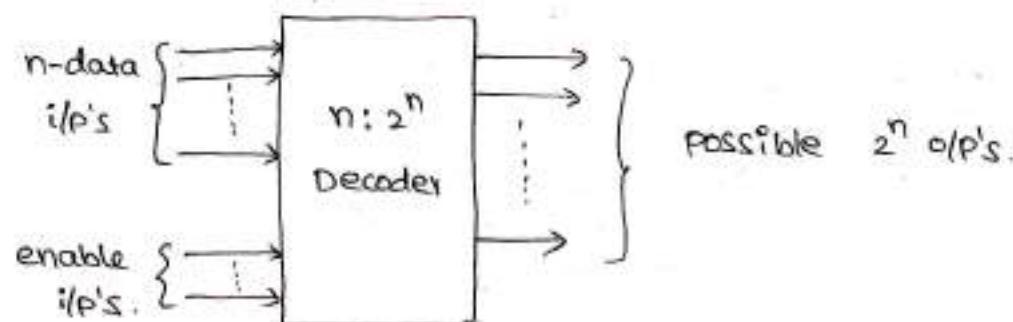
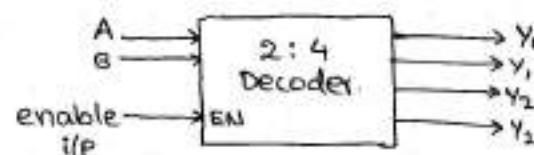


Fig: General Structure of Decoder.

(1) 2 - to - 4 (2:4) Binary Decoder :-

Block diagram :-



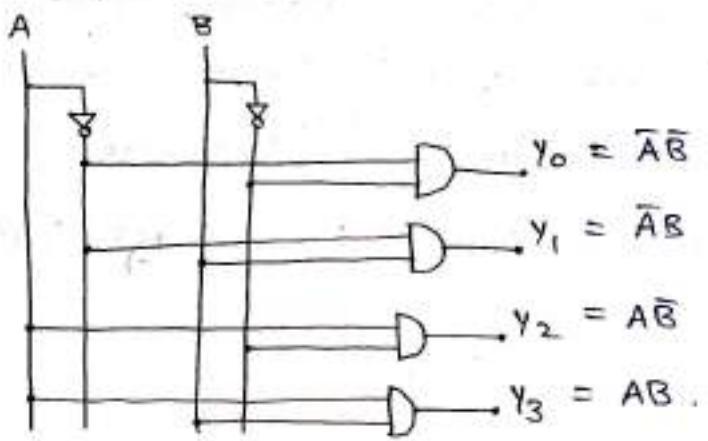
Truth table :-

i/p's			o/p's			
EN	A	B	$y_0$	$y_1$	$y_2$	$y_3$
0	X	X	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

(∴ enable i/p is '0' then there is no i/p's.)

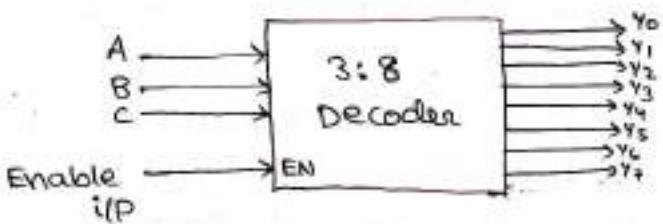
From the above truth table,  $y_0 = \bar{A}\bar{B}$ ,  $y_1 = \bar{A}B$  ( $\because$  K-map),  $y_2 = A\bar{B}$ ,  $y_3 = AB$  Simplification

Logic diagram :-



(ii) 3 - to - 8 (3:8) Binary decoder :-

Block diagram :-



Truth table :-

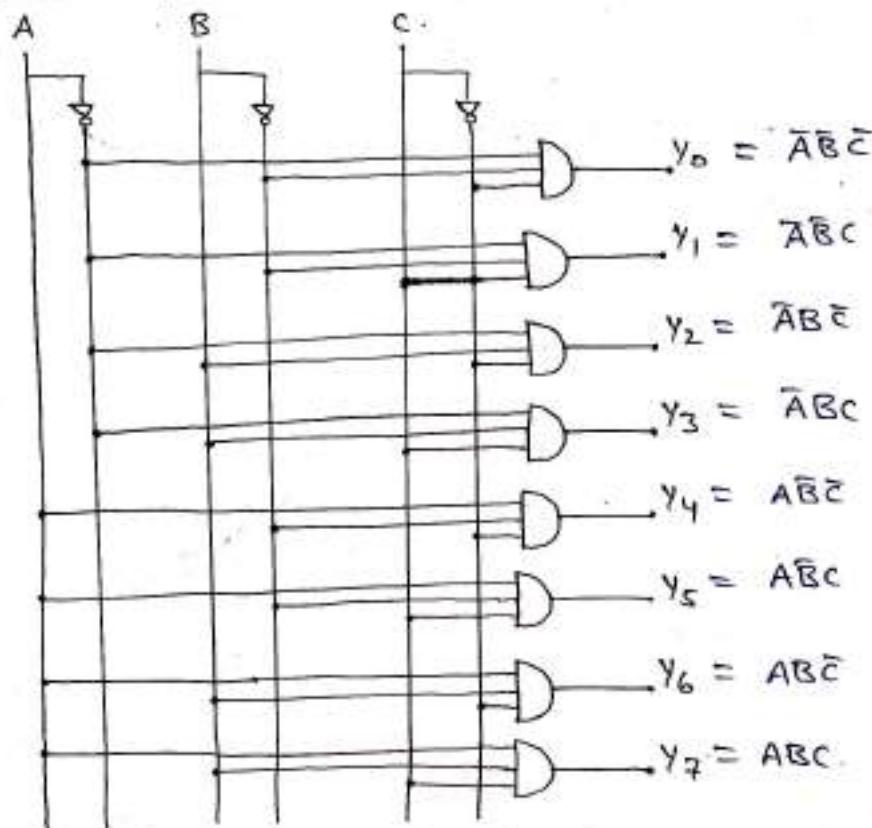
i/p's				o/p's							
EN	A	B	C	$y_0$	$y_1$	$y_2$	$y_3$	$y_4$	$y_5$	$y_6$	$y_7$
0	x	x	x	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	0
1	0	1	0	0	0	1	0	0	0	0	0
1	0	1	1	0	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	1	0	0	0
1	1	0	1	0	0	0	0	0	1	0	0
1	1	1	0	0	0	0	0	0	0	1	0
1	1	1	1	0	0	0	0	0	0	0	1

From the above truth table (using k-map simplification)

$$y_0 = \bar{A}\bar{B}\bar{C}, \quad y_1 = \bar{A}\bar{B}C, \quad y_2 = \bar{A}BC, \quad y_3 = \bar{A}B\bar{C}$$

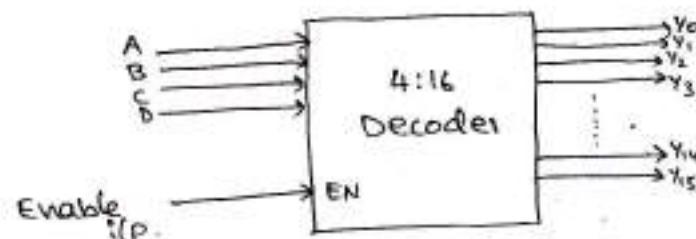
$$y_4 = A\bar{B}\bar{C}, \quad y_5 = A\bar{B}C, \quad y_6 = AB\bar{C}, \quad y_7 = ABC$$

logic diagram :-



Ex:- construct 4:16 decoders using 3:8 decoders.

Sol:- Block diagram of 4:16 decoder :-



Truth table of 4:16 decoder:-

t/p's				o/p
A	B	C	D	$y(\text{high})$
0	0	0	0	$y_0$
0	0	0	1	$y_1$
0	0	1	0	$y_2$
0	0	1	1	$y_3$
0	1	0	0	$y_4$
0	1	0	1	$y_5$
0	1	1	0	$y_6$
0	1	1	1	$y_7$
1	0	0	0	$y_8$
1	0	0	1	$y_9$
1	0	1	0	$y_{10}$
1	0	1	1	$y_{11}$
1	1	0	0	$\vdots$

4:16 Decoder using 3:8 Decoders :-

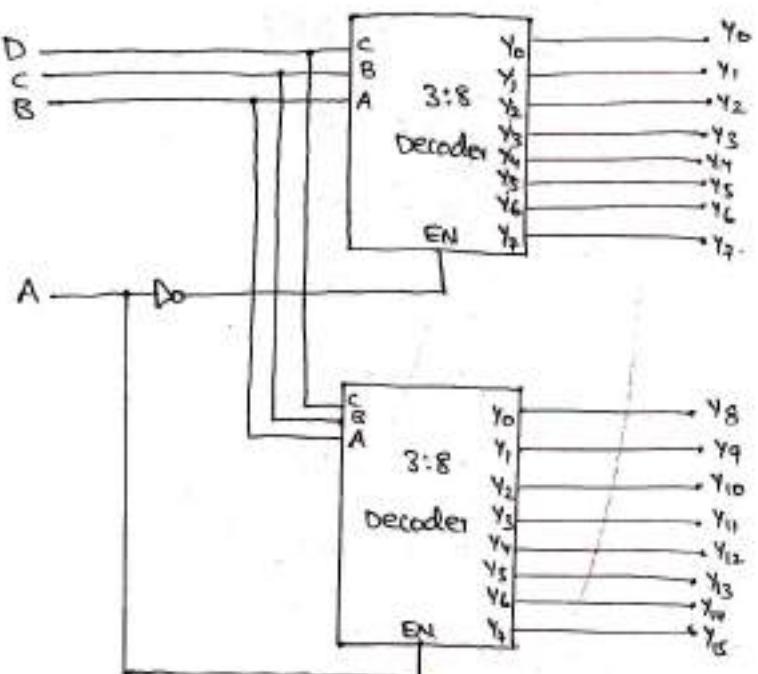
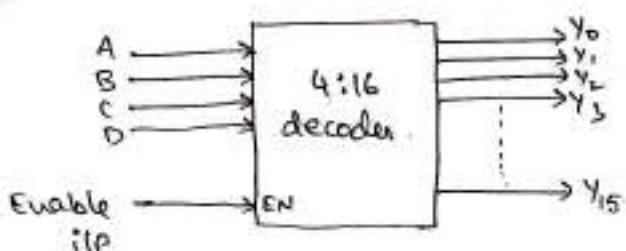


Fig: 4:16 decoder using 3:8 decoders.

Ex:- Construct 4:16 Decoder using 2:4 decoder.

Sol:-

Block diagram of 4:16 decoder :-



Truth table of 4:16 decoder :-

i/p's				o/p
A	B	C	D	Y(High)
0	0	0	0	Y <sub>0</sub>
0	0	0	1	Y <sub>1</sub>
0	0	1	0	Y <sub>2</sub>
0	0	1	1	Y <sub>3</sub>
0	1	0	0	Y <sub>4</sub>
0	1	0	1	Y <sub>5</sub>
⋮	⋮	⋮	⋮	⋮
1	1	1	0	Y <sub>14</sub>
1	1	1	1	Y <sub>15</sub>

4:16 decoder using 2:4 decoders :-

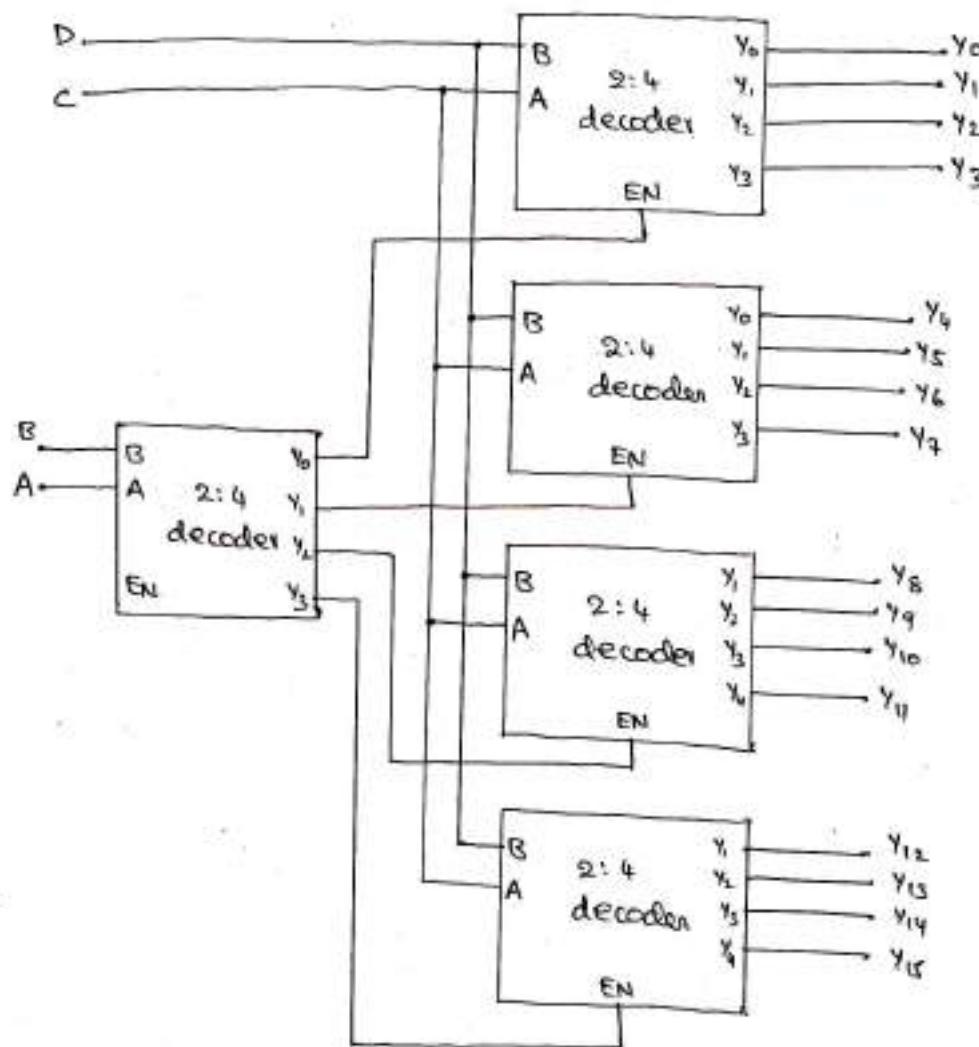
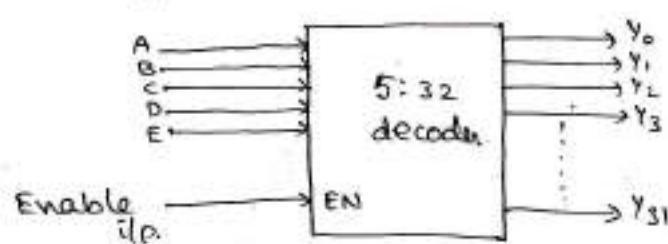


Fig: 4:16 decoder using 2:4 decoders.

Ex:- construct 5:32 decoder using 4, 3:8 decoders & 1, 2:4 decoders

Sol:- Block diagram of 5:32 decoder:-



Truth table of 5:32 decoder:

ip's					op
A	B	C	D	E	Y (High)
0	0	0	0	0	Y <sub>0</sub>
0	0	0	0	1	Y <sub>1</sub>
0	0	0	1	0	Y <sub>2</sub>
⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	1	0	Y <sub>30</sub>
⋮	⋮	⋮	⋮	⋮	⋮

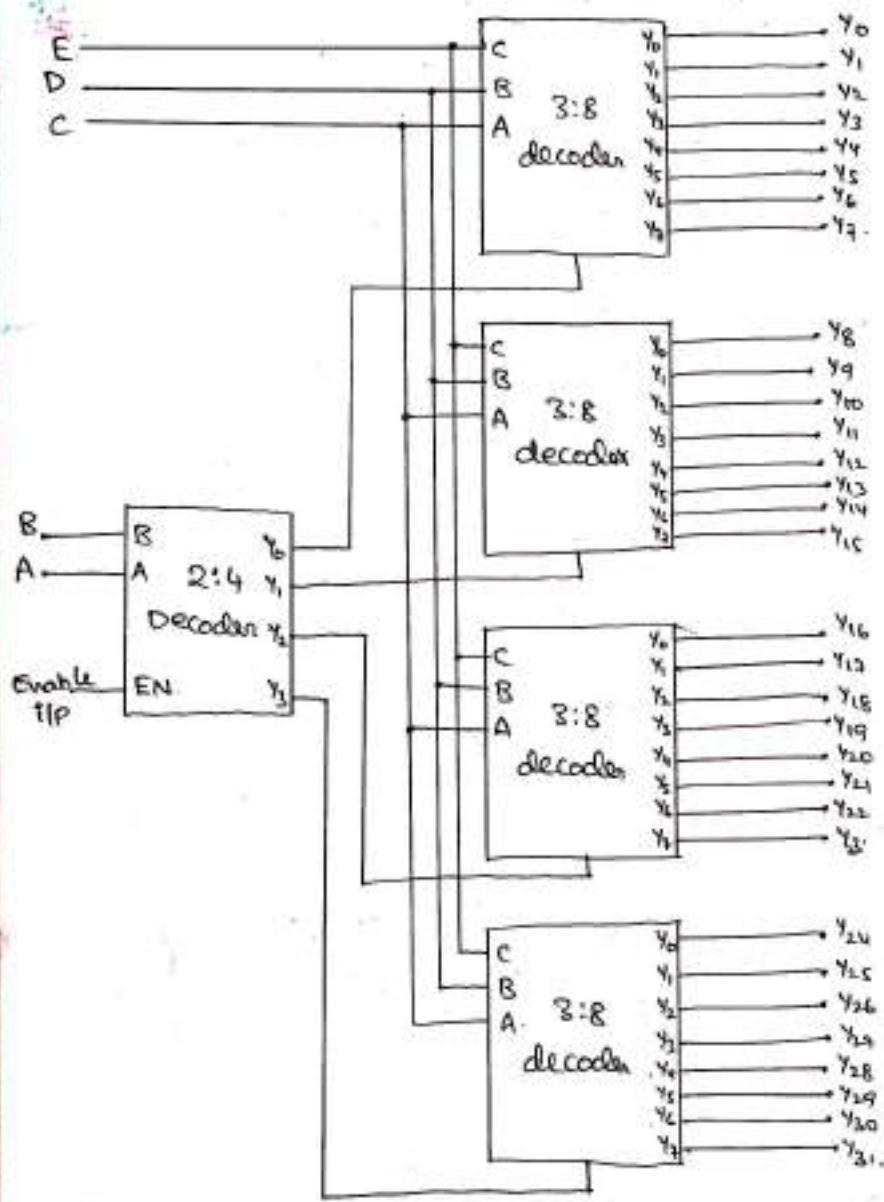


Fig: 5:32 decoder using 4, 3:8 decoders &  
1, 2:4 decoders.

\* Realization of Boolean Expressions using Binary decoder:-

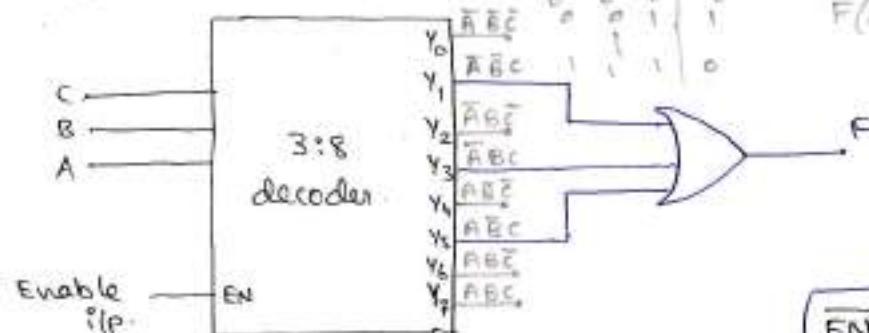
→ the combination of decoder and external logic gates can be used to implement single or multiple output functions.

→ The decoder generates minterms for input variables. Thus by logically ORing specified minterms we can implement the given function.

Ex:-  $F(A, B, C) = \sum m(1, 3, 5)$  Implement by using decoder 18

for 3M write TRUTH table

Sol:-



A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

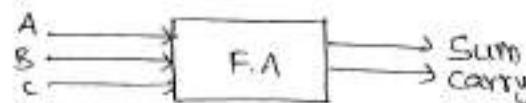
$$F(A, B, C) = \overline{A} \overline{B} \overline{C} + \overline{A} \overline{B} C + \overline{A} B \overline{C} + A \overline{B} \overline{C}$$

$\overline{EN} = 0 \rightarrow$  Active

$\overline{EN} = 1 \rightarrow$  disable

Ex:- Implement full adder circuit using decoder.

Sol:- Block diagram of full adder :-



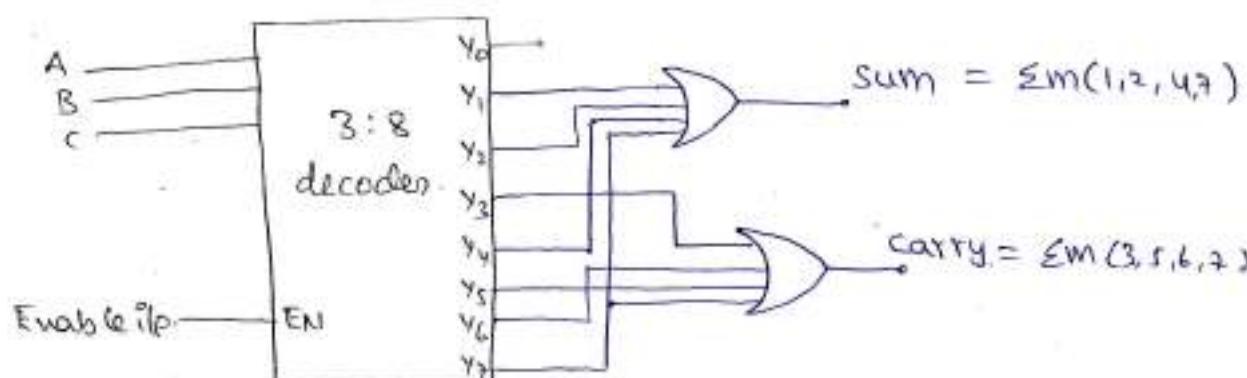
Truthtable of full adder:-

I/P's			O/P's	
A	B	C	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$\therefore \text{Sum} = \sum m(1, 2, 4, 7)$$

$$\text{Carry} = \sum m(3, 5, 6, 7) = \sum m(3, 5, 6, 7)$$

Full adder using Decoder & external gates:-

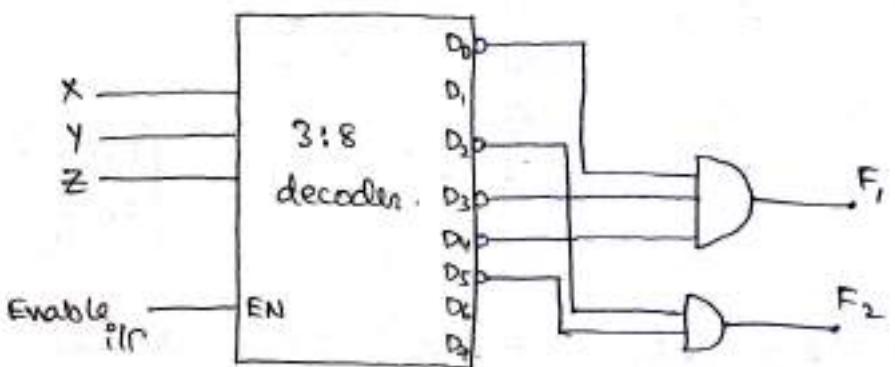


Ex: Implement the following boolean expressions using decoder.

Sol:  $F_1(x,y,z) = \pi M(0,3,4)$

$F_2(x,y,z) = \pi M(2,5)$

Sol:-



from the above circuit,

$$F_1 = \pi M(0,3,4)$$

$$F_2 = \pi M(2,5).$$

02/09/2014

\* BCD to 7-segment display decoder :-

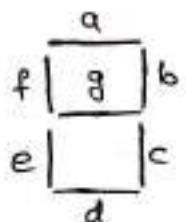


fig: 7-segment display.

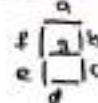
→ there are two types of 7-segment displays.

- (i) common anode → segment i/p must be active low to glow the segment.
- (ii) common cathode → segment i/p must be active high to glow the segment.

→ most commonly used one is common Cathode type.

19

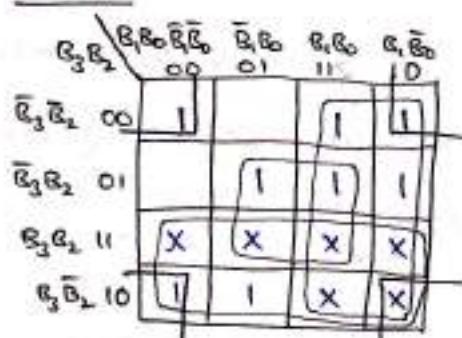
Truth table of common cathode 7-segment display decoder.



decimal	i/p (BCD)				o/p			
	B <sub>3</sub>	B <sub>2</sub>	B <sub>1</sub>	B <sub>0</sub>	a	b	c	d
0	0	0	0	0	1	1	1	1
1	0	0	0	1	0	1	1	0
2	0	0	1	0	1	1	0	1
3	0	0	1	1	1	1	1	0
4	0	1	0	0	1	1	0	1
5	0	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1
7	0	1	1	1	1	1	0	0
8	1	0	0	0	1	1	1	1
9	1	0	0	1	1	1	1	0

K-Map Simplification

for a :-



$$\therefore a = B_3 + B_2B_0 + B_1 + \bar{B}_2\bar{B}_0$$

Similarly, b =

$$c =$$

$$d =$$

$$e =$$

$$f =$$

$$g =$$

Logic diagram

## \* ENCODER :-

An encoder is a digital circuit that performs the inverse operation of the decoder. If it has " $2^n$ "(or) fewer input lines and 'n' output lines, the o/p lines generate the binary code corresponding to the i/p value.

### Block diagram :-

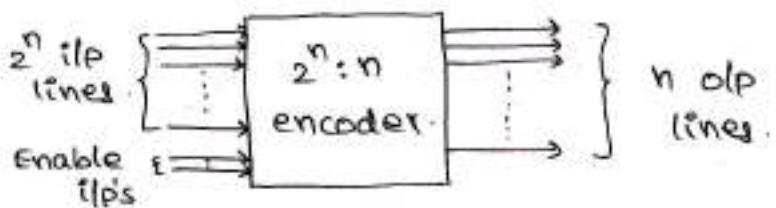
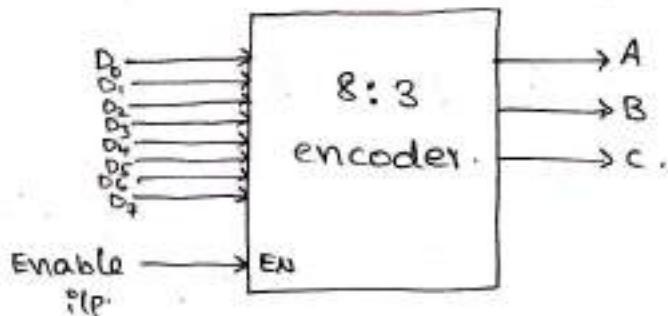


Fig: General structure of Encoder.

### Octal to Binary Encoder :- (8-to-3) (8:3 Encoder)

- It has '8' inputs. (i.e) 1 for each octal digit and 3 o/p's that generate the corresponding binary code.
- In Encoders it is assumed that only one i/p has a value of '1' at any given time, otherwise the circuit is meaningless.

### Block diagram:-



Truth table (or) Function table :-

20

i/p's								o/p's		
D <sub>0</sub>	D <sub>1</sub>	D <sub>2</sub>	D <sub>3</sub>	D <sub>4</sub>	D <sub>5</sub>	D <sub>6</sub>	D <sub>7</sub>	A	B	C
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	0	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1
EN = 0	0	0	0	0	0	0	0	X	X	X

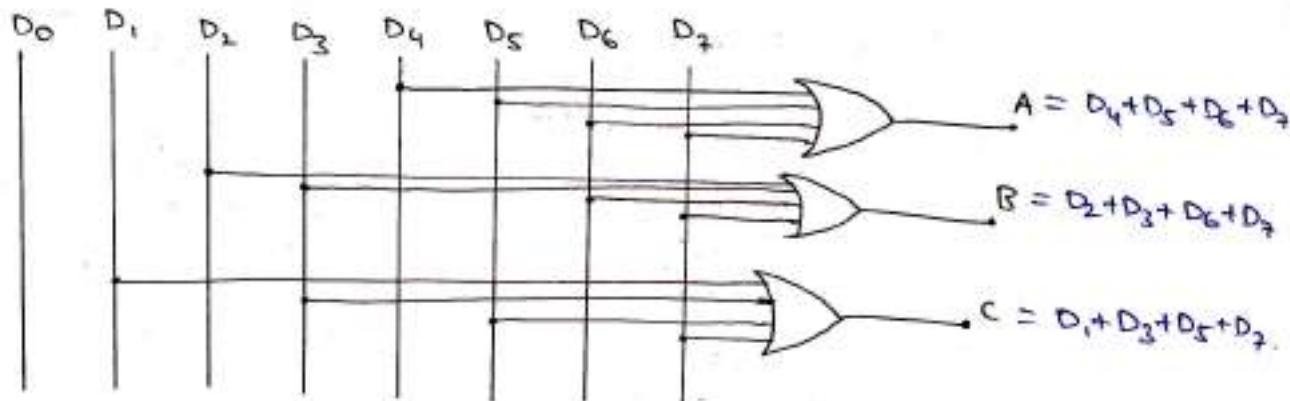
→ if Enable i/p is '0',  
there is no o/p.

$$\therefore A = D_4 + D_5 + D_6 + D_7$$

$$B = D_2 + D_3 + D_6 + D_7$$

$$C = D_1 + D_3 + D_5 + D_7$$

Logic diagram:-



\* Priority Encoder:- ( 4-bit priority encoder )

A priority encoder is a encoder circuit that includes the priority function. In priority encoder, if two or more i/p's are equal to '1' at the same time, the i/p having the highest priority will take precedence.

### Truth table:-

i/p's				o/p's		
$D_0$	$D_1$	$D_2$	$D_3$	$Y_1$	$Y_0$	$V$
0	0	0	0	X	X	0
1	0	0	0	0	0	1
0	1	0	0	0	1	1
0	0	1	0	1	0	1
1	0	0	1	1	1	1

→  $D_3$  i/p has highest priority and  $D_0$  i/p has lowest priority. When  $D_3$  i/p is high, regardless of other i/p's output is 1.

→ 'V' is a valid bit indicator, that is set to 1 when 1 or more i/p's are equal to 1.

### K-Map Simplifications:-

for  $Y_1$  :-

$D_0\bar{D}_3$	$\bar{D}_2\bar{D}_3$	$\bar{D}_2D_3$	$D_2\bar{D}_3$	$D_2D_3$
$\bar{D}_0\bar{D}_1$	00	01	11	10
X	1	1	1	1
00	1	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

$$\therefore Y_1 = D_2 + D_3$$

for  $Y_0$  :-

$D_0\bar{D}_3$	00	01	11	10
$\bar{D}_0D_1$	00	X	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

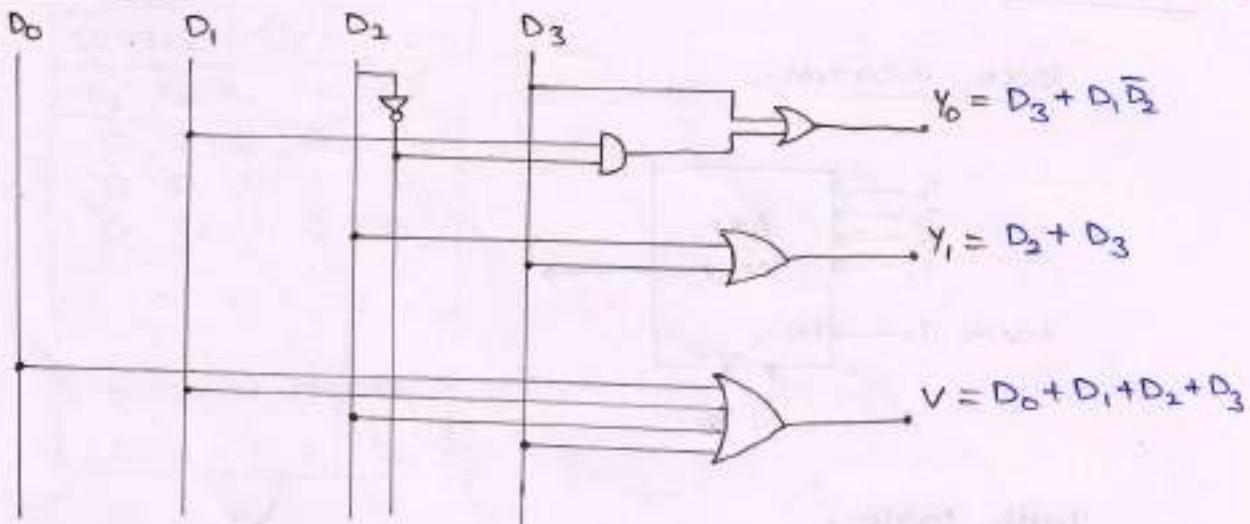
$$\therefore Y_0 = D_3 + D_1\bar{D}_2$$

for  $V$  :-

$D_0\bar{D}_3$	00	01	11	10
$\bar{D}_0D_1$	00	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

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

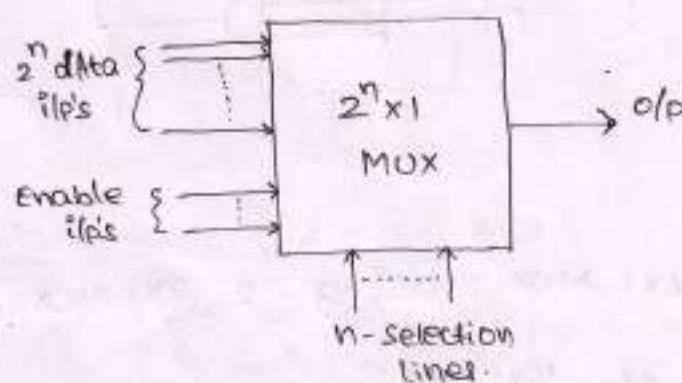
logic circuit :-



\* Multiplexer (or) Data selector :-

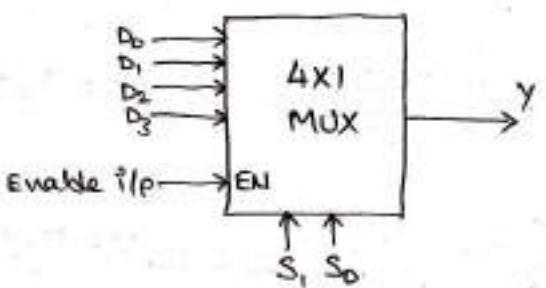
- Multiplexer is a combinational logic circuit that selects binary information from one of many input lines and directs it to a single o/p line.
- the selection of particular i/p line is controlled by a set of selection lines.
- MUX has  $2^n$  i/p lines, n selection lines & one o/p.

Block diagram:-



03/01/2014  
4x1 MUX :-

Block diagram :-

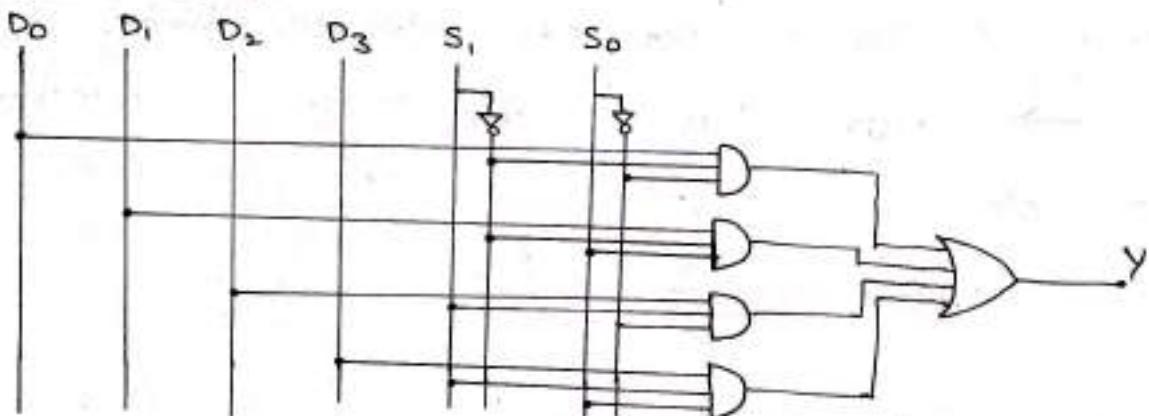


Truth table :-

i/p's			o/p
EN	S <sub>1</sub>	S <sub>0</sub>	Y
0	X	X	X
1	0	0	D <sub>0</sub>
1	0	1	D <sub>1</sub>
1	1	0	D <sub>2</sub>
1	1	1	D <sub>3</sub>

$$\therefore Y = D_0 \cdot \bar{S}_1 \cdot \bar{S}_0 + D_1 \cdot \bar{S}_1 \cdot S_0 + D_2 \cdot S_1 \cdot \bar{S}_0 + D_3 \cdot S_1 \cdot S_0$$

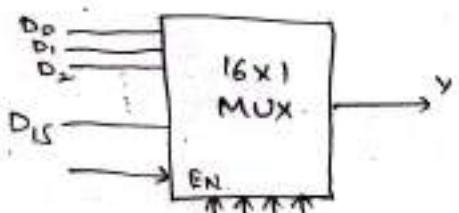
Logic circuit :-



Ex:- Construct a 16x1 MUX using 2, 8x1 MUX & 1, 2x1 MUX.

Sol:-

Block diagram of 16x1 MUX :-



Truth table :-

selection if(p's)				o/p
$S_3$	$S_2$	$S_1$	$S_0$	$y$
0	0	0	0	$D_0$
0	0	0	1	$D_1$
0	0	1	0	$D_2$
				.
				.
1	1	1	0	$D_{14}$
1	1	1	1	$D_{15}$

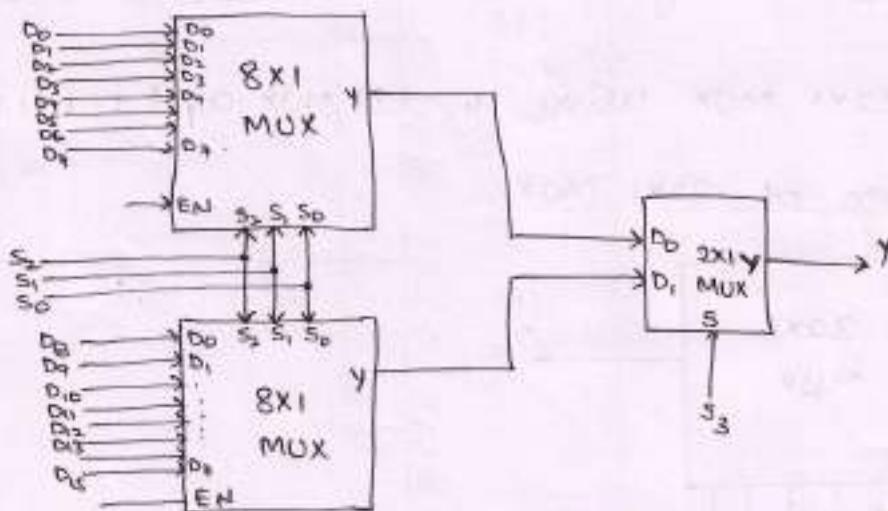
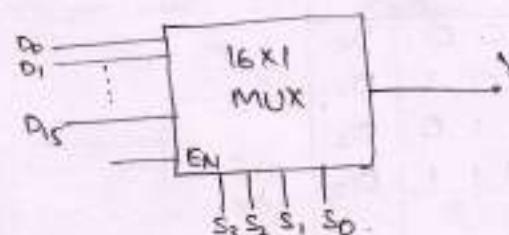


Fig:- 16x1 MUX using two 8x1 MUX & one 2x1 MUX.

Ex:- construct a 16x1 MUX using two 8x1 MUX and OR gate.

Sol:- Block diagram of 16x1 MUX:-



Truth table of 16x1 MUX :-

selection if(p's)				o/p
$S_3$	$S_2$	$S_1$	$S_0$	$y$
0	0	0	0	$D_0$
0	0	0	1	$D_1$
				.
				.
1	1	1	0	$D_{14}$

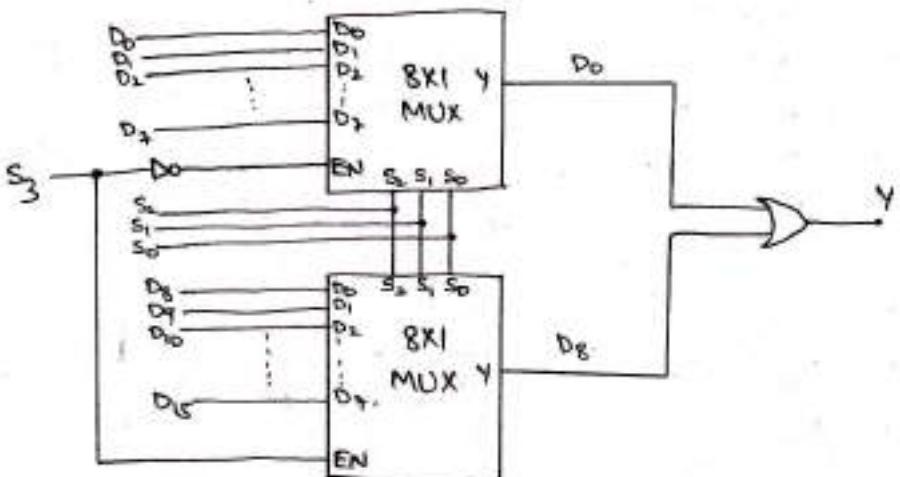
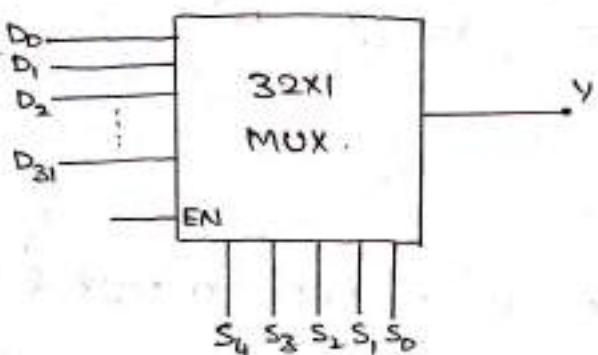


Fig:- 16x1 MUX using two 8x1 MUX and one OR gate.

Ex:- CONSTRUCT A 32x1 MUX USING 4, 8x1 MUX AND 1, 4x1 MUX.

Sol:- Block diagram of 32x1 MUX:-



Truth table of 32x1 MUX:-

Selection bits					out.
\$S_4	\$S_3	\$S_2	\$S_1	\$S_0	\$Y
0	0	0	0	0	\$D_0\$
0	0	0	0	1	\$D_1\$
0	0	0	1	0	\$D_2\$
0	0	0	1	1	\$D_3\$
.	.	.	.	.	.
1	1	1	0	1	\$D_{29}\$
1	1	1	1	0	\$D_{30}\$
1	1	1	1	1	\$D_{31}\$

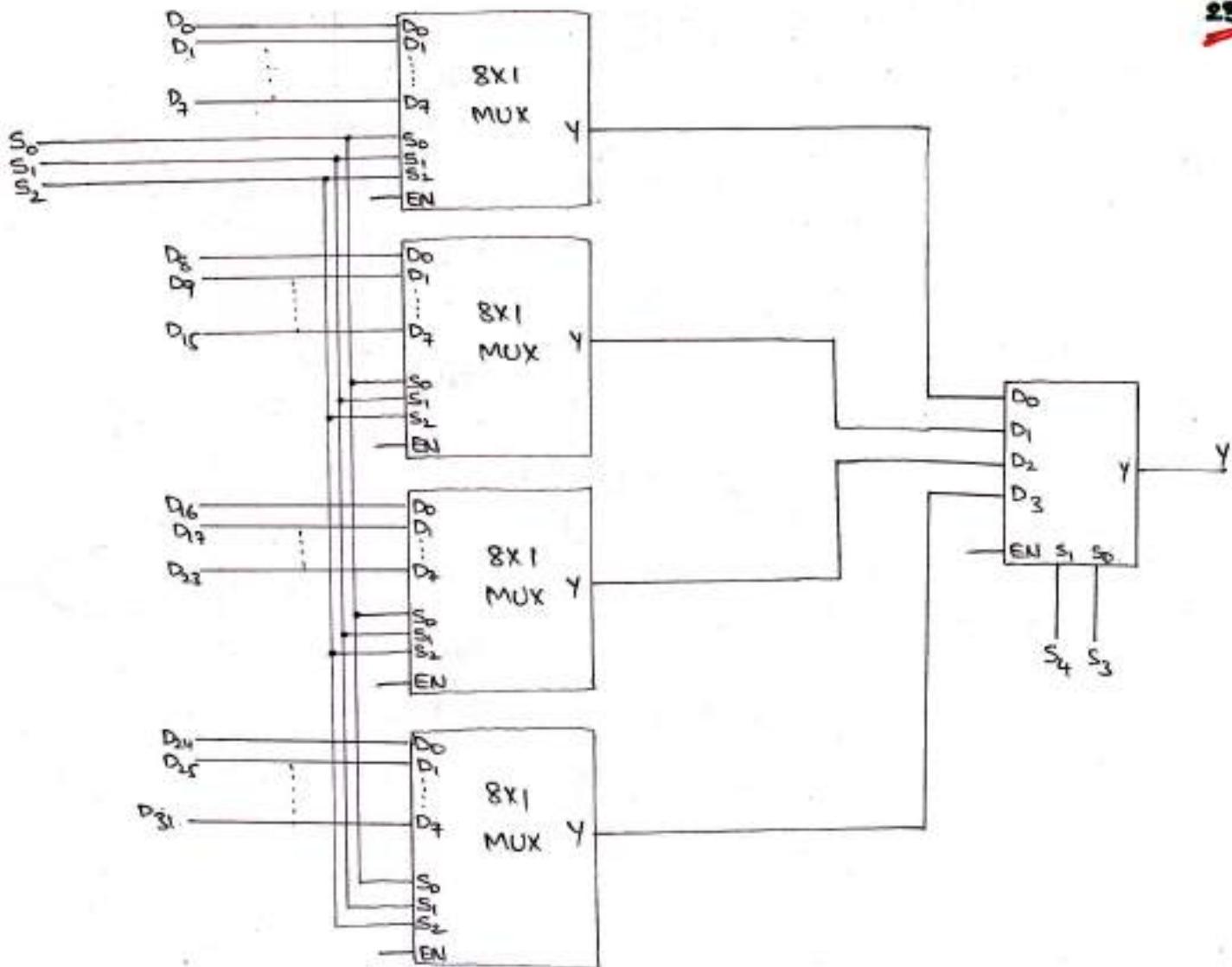
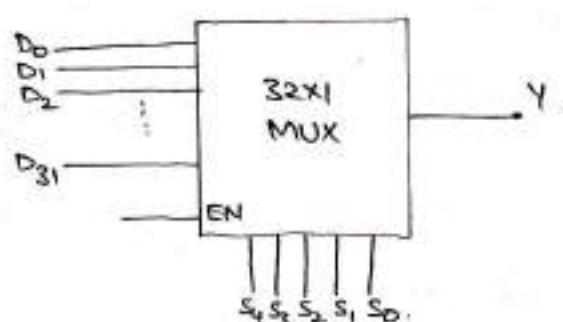


Fig: 32x1 MUX using four 8x1 MUX & one 4x1 MUX.

Ex:- Construct 32x1 MUX using 4, 8x1 MUX & 1, 2-to-4 decoder, 1 OR-gate.

Sol:- Block diagram of 32x1 MUX :-



Truth table of 32x1 MUX :-

selection i/p's	o/p
S <sub>4</sub> S <sub>3</sub> S <sub>2</sub> S <sub>1</sub> S <sub>0</sub>	Y
0 0 0 0 0	D <sub>0</sub>
0 0 0 0 1	D <sub>1</sub>
⋮	⋮
1 1 1 1 0	D <sub>30</sub>
1 1 1 1 1	D <sub>31</sub>

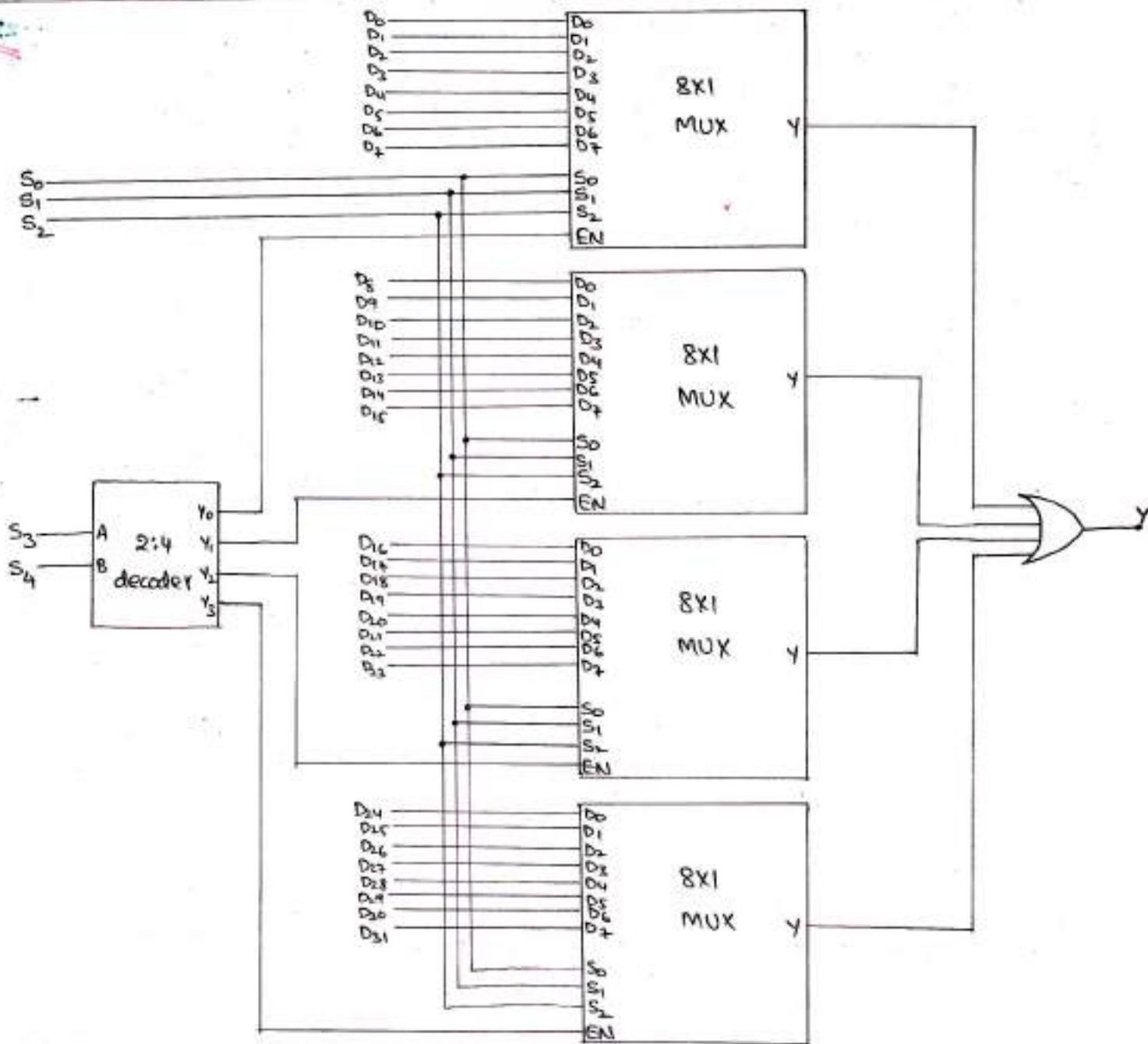


Fig:- 32x1 MUX using four 8x1 MUX, one 2:4 decoder & one OR-gate.

\* Implementation of combinational logic using multiplexer:-

Ex:- Implement the following boolean function using

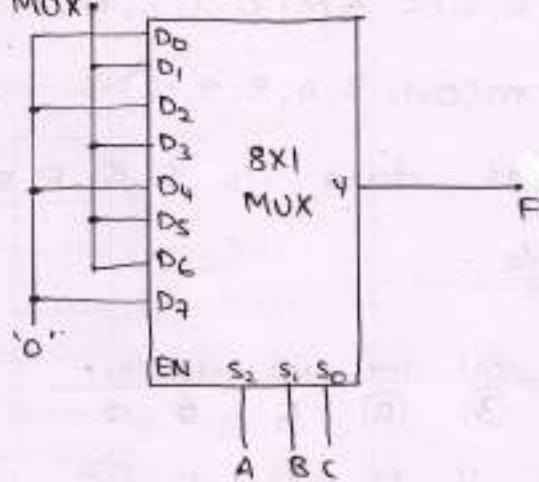
(a) 8x1 MUX (b) 4x1 MUX

$$(i) F(A, B, C) = \sum m(1, 3, 5, 6)$$

Sol:- Truth table:-

IP's			OP
A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

(i) using 8x1 MUX



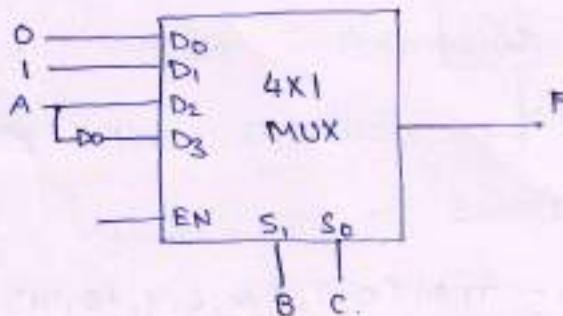
(ii) Using 4x1 MUX

→ considering 'A' as data i/p & B,C as selection i/p's.

implementation table :-

A	BC		D <sub>0</sub>	D <sub>1</sub>	D <sub>2</sub>	D <sub>3</sub>
	B	C	B̄C	B̄C	BC	BC
Ā	0	00	0	1	2	3
	1	01	4	5	6	7

O I A Ā

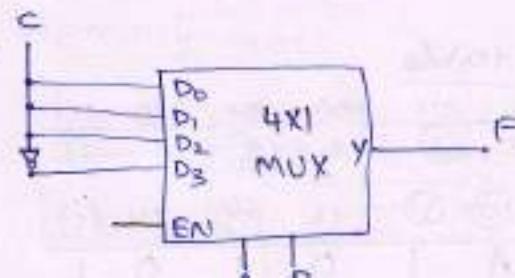


→ considering 'c' as data i/p & A,B as selection i/p's.

implementation table :-

C	AB		D <sub>0</sub>	D <sub>1</sub>	D <sub>2</sub>	D <sub>3</sub>
	A	B	ĀB	AB	ĀB	AB
C̄	0	00	0	2	4	6
	1	01	①	③	⑤	7

C C C C̄



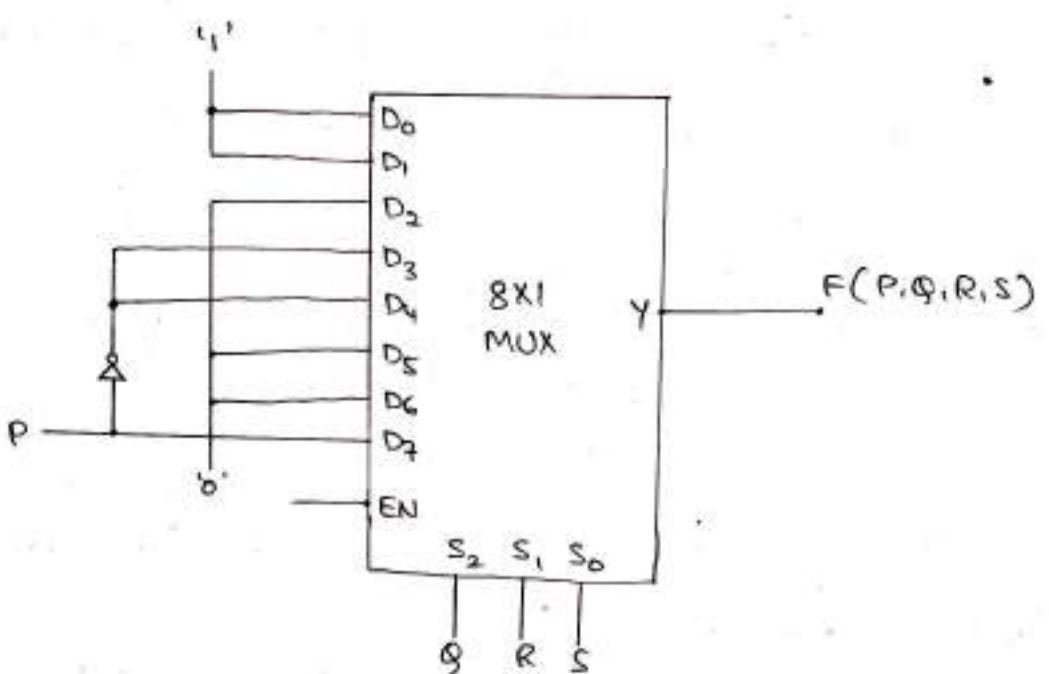
Ex. Implement  $F(P, Q, R, S) = \sum m(0, 1, 3, 4, 8, 9, 15)$  using 8x1 MUX.

Sol.  $F(P, Q, R, S) = \sum m(0, 1, 3, 4, 8, 9, 15)$ .

considering 'P' as data i/p & Q, R, S as selection i/p's.

Implementation table :-

P \ QRS	000	001	010	011	100	101	110	111
$\bar{P}$ 0	①	②	2	③	④	5	6	7
P 1	⑧	⑨	10	11	12	13	14	⑯
	1	1	0	$\bar{P}$	$\bar{P}$	0	0	P



Ex. Implement  $F(A, B, C, D) = \prod M(0, 1, 2, 4, 6, 9, 12, 14)$  using 8x1 MUX.

Sol.  $F(A, B, C, D) = \prod M(0, 1, 2, 4, 6, 9, 12, 14)$ .

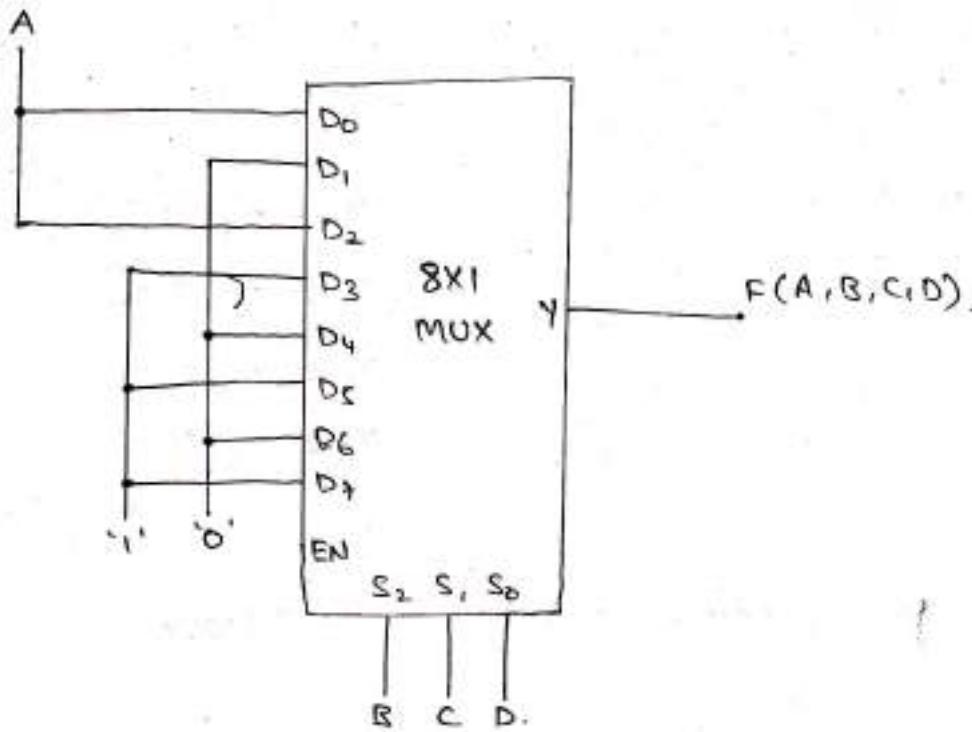
considering 'A' as data i/p & B, C, D as selection i/p's.  
Here instead of minterms, Maxterms are given.

Specified thus we have to circle Maxterms which are not included in the boolean function.

$$\therefore F(A, B, C, D) = \prod M(0, 1, 2, 4, 6, 9, 12, 14) = \sum m(3, 5, 7, 8, 10, 11, 13, 15)$$

Implementation table :-

A \ BCD	000	001	010	011	100	101	110	111
$\bar{A}$ 0	0	1	2	③	4	⑤	6	⑦
A 1	⑧	9	⑩	⑪	12	⑬	14	⑯
	A	0	A	1	0	1	0	1

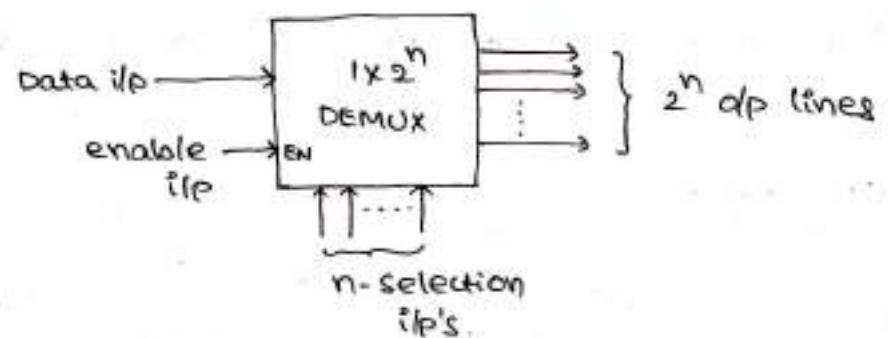


05log12014

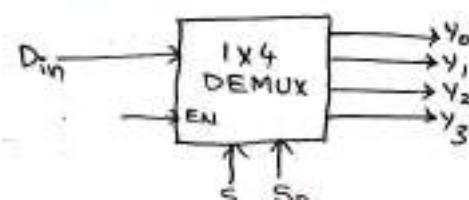
### Demultiplexer (or) Data distributor :-

A demultiplexer is a combinational circuit that receives information on a single i/p line and transmits that information on one of  $2^n$  possible o/p lines. The selection of specific o/p line is controlled by the values of 'n' selection lines.

#### Block diagram:-



#### (i) $1 \times 4$ demultiplexer



Truth table:

i/p's			o/p's			
EN	S <sub>1</sub>	S <sub>0</sub>	Y <sub>3</sub>	Y <sub>2</sub>	Y <sub>1</sub>	Y <sub>0</sub>
0	X	X	0	0	0	0
1	0	0	0	0	0	D <sub>in</sub>
1	0	1	0	0	D <sub>in</sub>	0
1	1	0	0	D <sub>in</sub>	0	0
1	1	1	D <sub>in</sub>	0	0	0

$$\therefore Y_0 = D_{in} \cdot \bar{S}_1 \bar{S}_0$$

$$Y_1 = D_{in} \cdot \bar{S}_1 S_0$$

$$Y_2 = D_{in} S_1 \bar{S}_0$$

$$Y_3 = D_{in} \cdot S_1 S_0$$

Ex:- Construct 1x4 DEMUX using 2, 1x2 DEMUX.

Sol:-

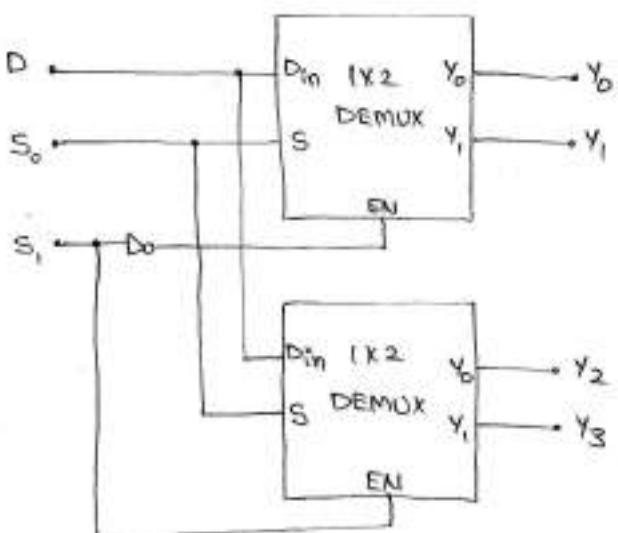


fig:- 1x4 DEMUX using two 1x2 demux.

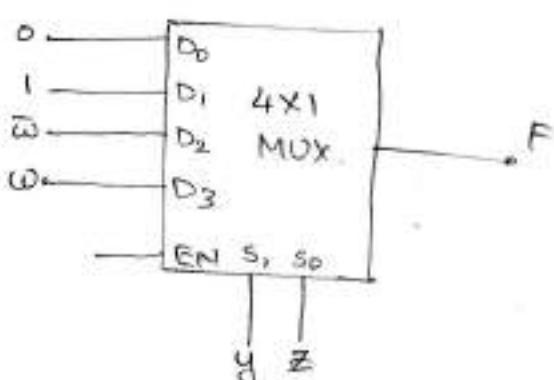
09/10/2014

Ex:- Implement  $F(w, x, y, z) = \sum m(1, 2, 5, 6, 9, 11, 13, 15)$  using 4x1 MUX.

Sol:- Considering w, x as data i/p's & y, z as section i/p's of 4x1 MUX.

Implementation table

w\z	y\z	00	01	10	11
00	0	①	②	3	
01	4	⑤	⑥	7	
w\z	10	8	⑨	10 - 11	
w\z	11	12	⑬	14 - 15	
0	1	$\bar{w}\bar{x}$	$\bar{w}\bar{x} + w\bar{x}$	$w\bar{x}$	$w$



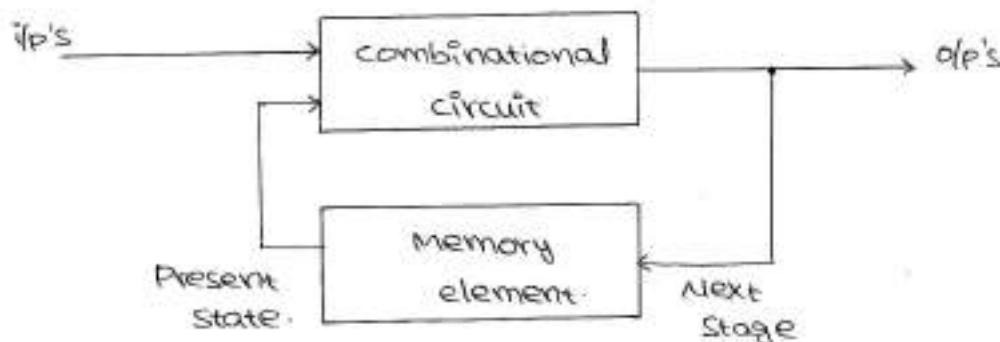
SEQUENTIAL LOGIC

09/09/2014

Differences b/w combinational circuit and sequential circuit:

combinational circuit	sequential circuit
(i) Output depends on only present inputs.	(i) Output depends on present i/p's and past outputs.
(ii) Memory unit is not required.	(ii) Memory unit is required to store the past history of o/p's.
(iii) Faster in speed.	(iii) slower than combinational circuits
(iv) Easy to design.	(iv) compared to combinational circuits harder to design.
(v) Ex:- Encoders, Decoders, MUX, DEMUX.	(v) Ex:- flip-flops, Registers, counters.

Block diagram of sequential circuit:-



- \* One of the most fundamental sequential circuit is latch / flip-flop.
- \* A latch / flip-flop is a bistable multivibrator, that can store 1 bit of binary information (either 0 or 1).
- \* Because of their ability to retain a given state, these elements are useful as storage elements.

## \* LATCHES :-

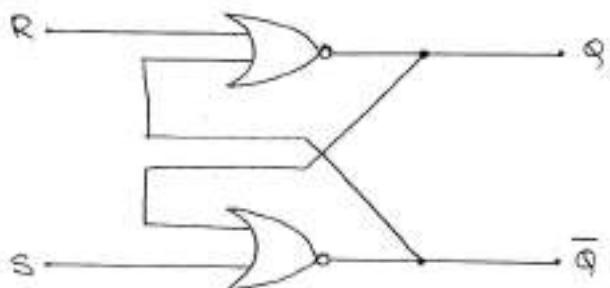
### 1. RS latch / SR latch / Set-Reset latch / Reset-Set latch :-

- \* The most fundamental type of storage element is an SR latch.
- \* SR latch has two i/p's (S and R), two o/p's ( $Q$  &  $\bar{Q}$ ) which are complemented to each other.
- \* The SR latch can be constructed from either NAND (or) NOR gates.

### Logic symbol of SR latch :-



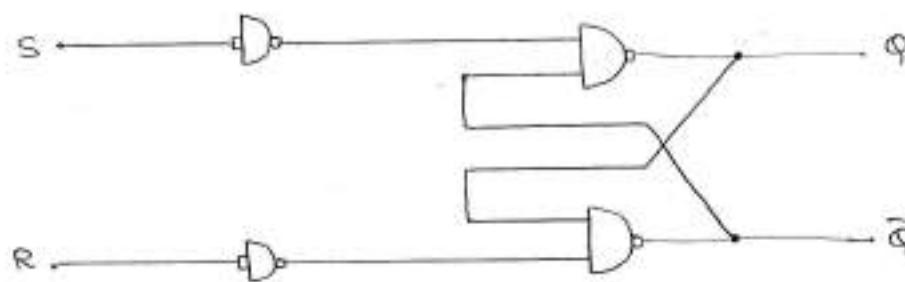
### Logic circuit of RS latch using NOR gates :-



### Truth table / Function table :-

i/p's	o/p			State
R	S	$Q_{in}$	$Q_{out}$	
0	0	0	0	No change
0	0	1	1	
0	1	0	1	Set
0	1	1	1	
1	0	0	0	Reset
1	0	1	0	
1	1	0	X	Indeter- minate.
1	1	1	X	

Logic circuit of RS latch using NAND Gate :-



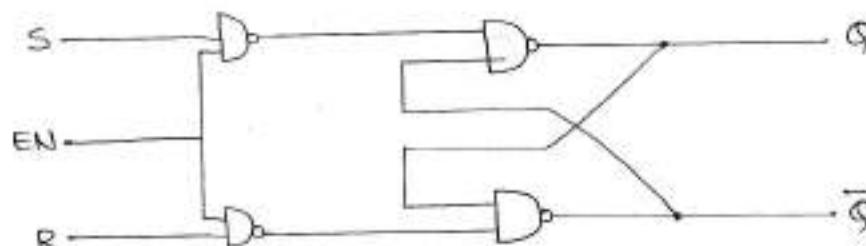
2. Gated SR latch :-

SR latch with enable input is known as "Gated SR latch".

Block diagram / logic symbol :-



Logic circuit:-



Truth table / Function table :-

EN	I/P's			O/P	State
	R	S	$Q_{in}$	$Q_{out}$	
1	0	0	0	0	No change
1	0	0	1	1	
1	0	1	0	1	Set
1	0	1	1	1	
1	1	0	0	0	Reset
1	1	0	1	0	
1	1	1	0	X	Indeter- minate
1	1	1	1	X	
0	X	X	0	0	No change
0	X	X	1	1	

### 3. D latch / Data latch :-

(4)

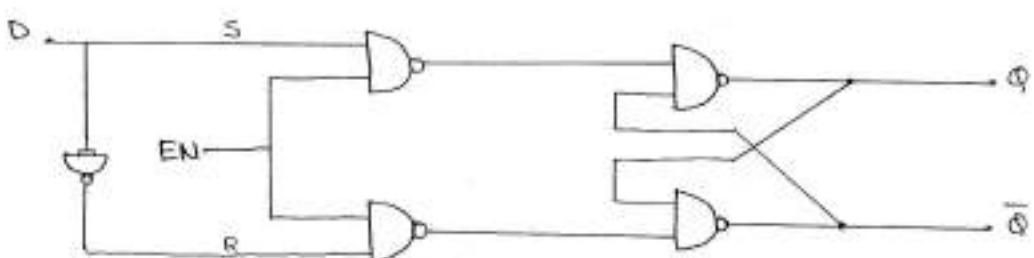
- \* From the truth table of SR latch it is clear that when both i/p's are same, the o/p either doesn't change or it is invalid.
- \* In many practical applications these i/p conditions are not required.
- \* These i/p conditions can be avoided by making i/p's complement to each other.
- \* This modified SR latch is known as "D latch".

### Block diagram / logic symbol :-



EN → Enable i/p.

### Logic circuit :-

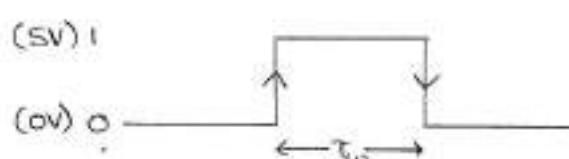


### Truth table / Function table :-

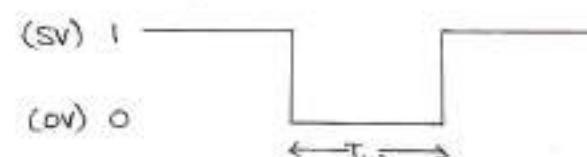
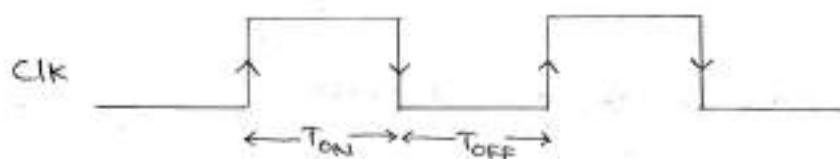
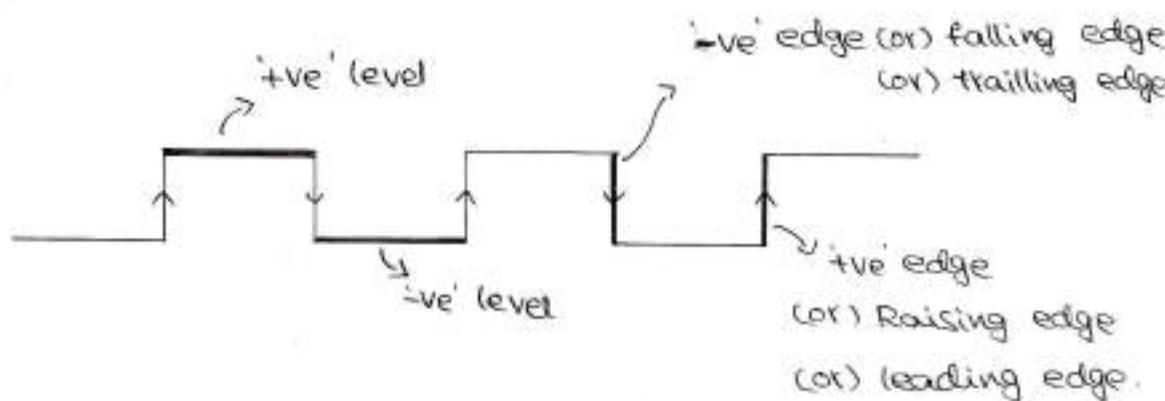
i/p's		o/p		State
EN	D	$Q_{in}$	$Q_{out}$	
0	x	0	0	No change
0	x	1	1	change
1	0	0	0	Reset
1	0	1	0	Set
1	1	0	1	
1	1	1	1	

10/09/2014

Positive pulse (0 to 1)



Negative pulse (1 to 0)

 $T_W \rightarrow$  pulse widthclk  $\rightarrow$  clock is a periodic pulse trainTime period of clk signal  $T = T_{ON} + T_{OFF}$ frequency of clk ( $f$ ) =  $\frac{1}{T}$ 

#### \* Types of triggerings :-

##### 1. Level triggering.

(a) '+ve' level triggering

(b) '-ve' level triggering

##### 2. Edge triggering

(a) '+ve' edge triggering

(b) '-ve' edge triggering

$\rightarrow$  '-ve' edge triggered  
circuit responds to edges only  
at '-ve' edges of clk signal.

\* Latches uses level triggering.

\* flip-flops uses edge triggering.

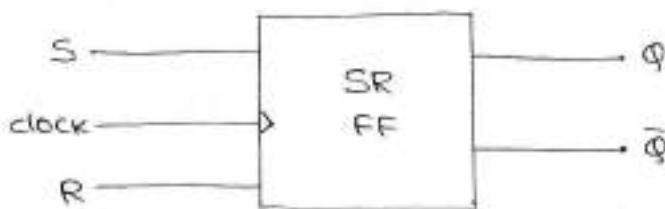
## \* Flip-flop :-

(6)

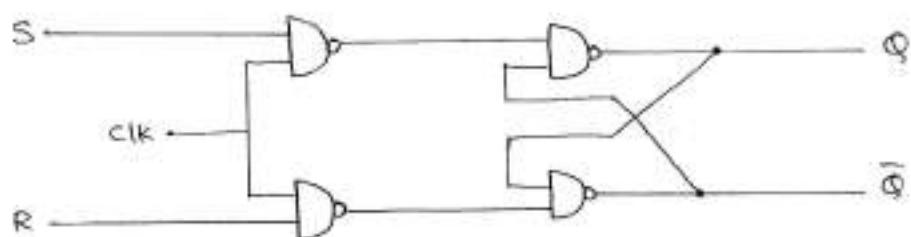
- \* Flip-flop is a memory element stores 1 bit of binary information.
- \* flip-flop has two stable states (0 or 1).
- \* State of flip-flop i.e. bit stored is indicated by Q output.
- \* Other names of flip-flop are 1 bit memory cell, Bistable circuit, Bistable multivibrator.
- \* On power, FF can be at 0 or 1 state

### 1. Clocked SR flip-flop :-

(i) Logic symbol of 'tve' edge triggered SR flip-flop:



Logic circuit:-



Truth table:-

I/P's				O/P	State
CLK	S	R	$Q_n$	$Q_{n+1}$	
For ↑	0	0	0	0	No change
For ↑	0	0	1	1	Set
↑	0	1	0	0	Reset
↑	0	1	1	0	
↑	1	0	0	1	
↑	1	0	1	1	
↑	1	1	0	X	Indeterminate
↑	1	1	1	X	Indeterminate
↑	X	X	0	0	No change
↑	X	X	1	1	No change

$Q_n \rightarrow$  State of FF before applying clk pulse

$Q_{n+1} \rightarrow$  State of FF after applying clk

Characteristic table

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

characteristic equation of SR flip-flop :-

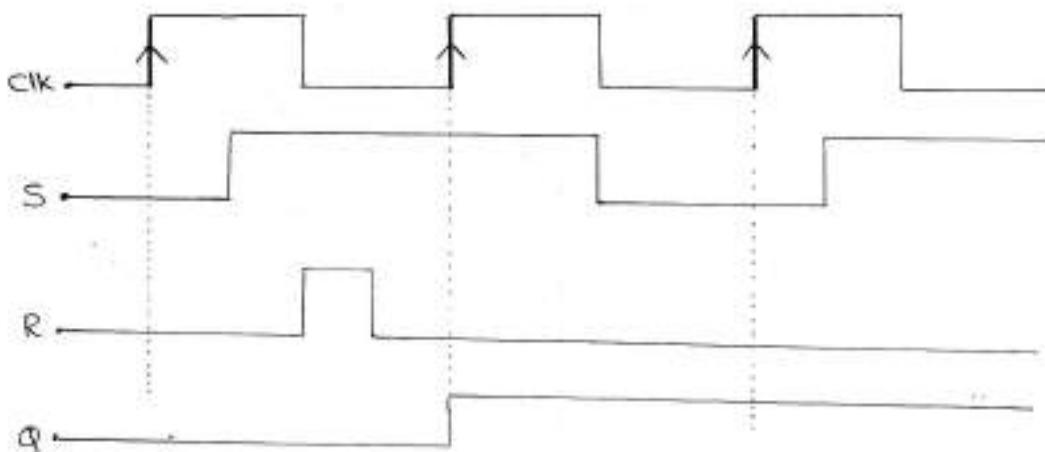
K-map for  $Q_{n+1}$

S/R	Q <sub>n</sub>	00	01	11	10
0		1			
1		1	X	X	

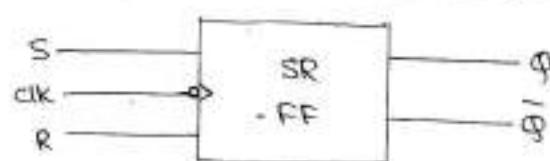
$$\therefore Q_{n+1} = S + \bar{R}Q_n$$

characteristic equation of SR flip-flop is  $Q_{n+1} = S + \bar{R}Q_n$

i/p & o/p waveforms of 'tve' edge triggered SR flip-flop:-

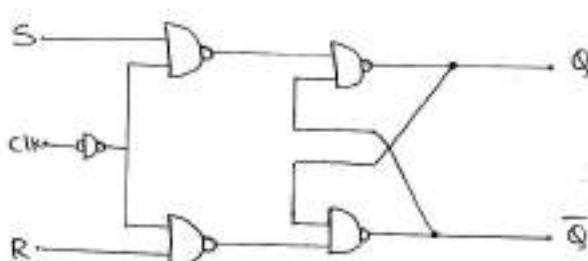


(ii) logic symbol of 'n'e' edge triggered SR flip-flop :-



clock signal symbol  
 $\text{F}_{(n)} \downarrow$

logic circuit:-

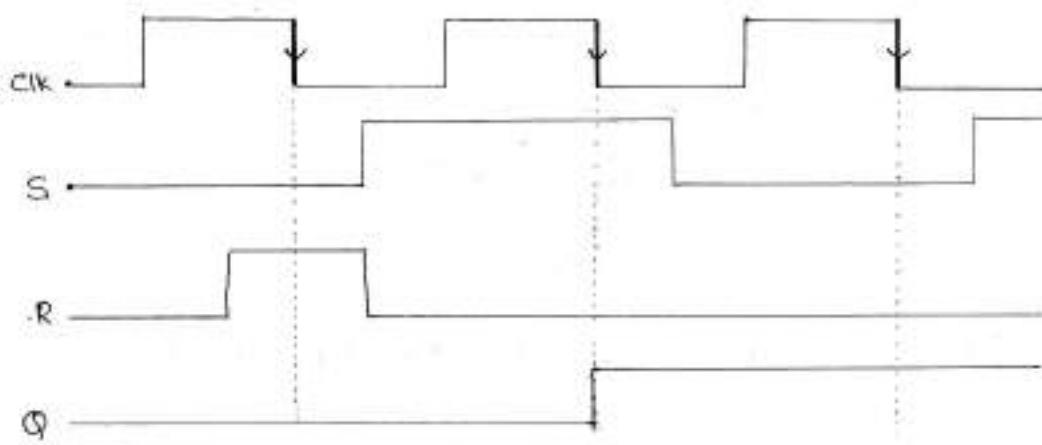


Truth table:-

clk	i/p's			o/p		state
	S	R	$Q_n$	$Q_{n+1}$	$\bar{Q}_{n+1}$	
$\text{F}_{(n)} \downarrow$	0	0	0	0	1	No change
$\text{F}_{(n)} \downarrow$	0	0	1	1	0	
$\text{F}_{(n)} \downarrow$	0	1	0	0	1	Set
$\text{F}_{(n)} \downarrow$	0	1	1	0	1	Reset
$\downarrow$	1	0	0	1	0	
$\downarrow$	1	0	1	1	0	
$\downarrow$	1	1	0	X	X	Indeterminate
$\downarrow$	1	1	1	X	X	-infinite
$\downarrow$	X	X	0	0	1	No change
$\downarrow$	X	X	1	1	0	

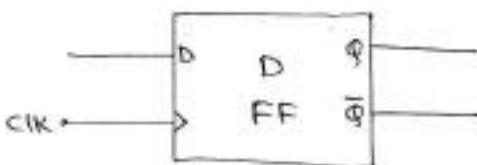
I/p & O/p waveforms of '-ve' edge triggered SR flip-flop: -

(8)

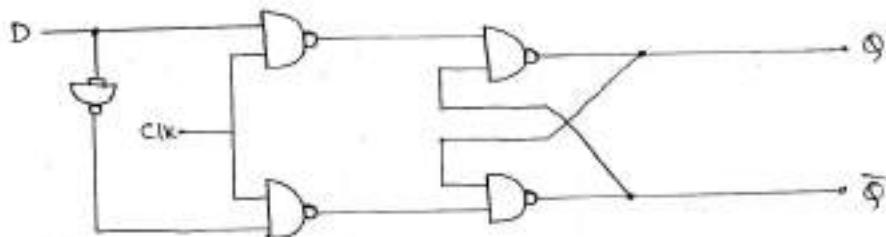


2. Clocked D flip-flop :- / Data flip-flop / Delay flip-flop :-

(i) Logic symbol of 'tve' edge triggered D flip-flop :-



Logic circuit:-



Truth table:-

I/p's		O/p		State
clk	D	$Q_{in}$	$Q_{out}$	
↑	0	0	0	Reset
↑	0	1	0	
↑	1	0	1	Set
↑	1	1	1	
0	x	0	0	No change
0	x	1	1	

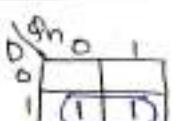
Characteristic table:-

D	$Q_{out}$
0	0
1	1

=

Characteristic equation of D flip-flop:-

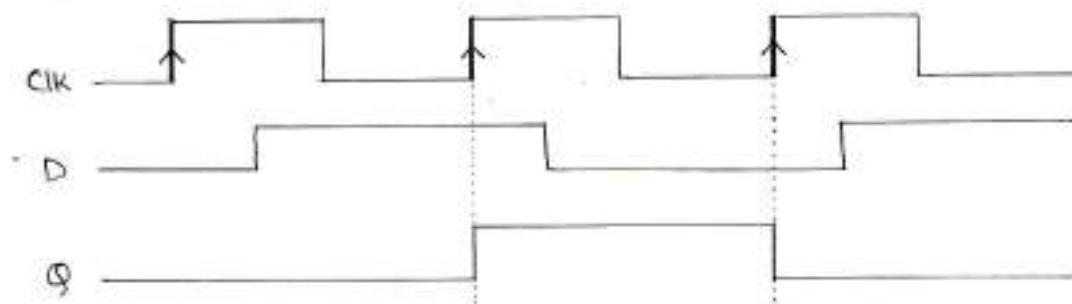
K-map for  $Q_{out}$



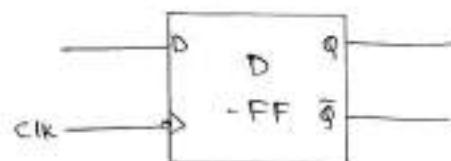
$$\therefore Q_{out} = D$$

∴ characteristic equation of D flip-flop is  $Q_{out} = D$

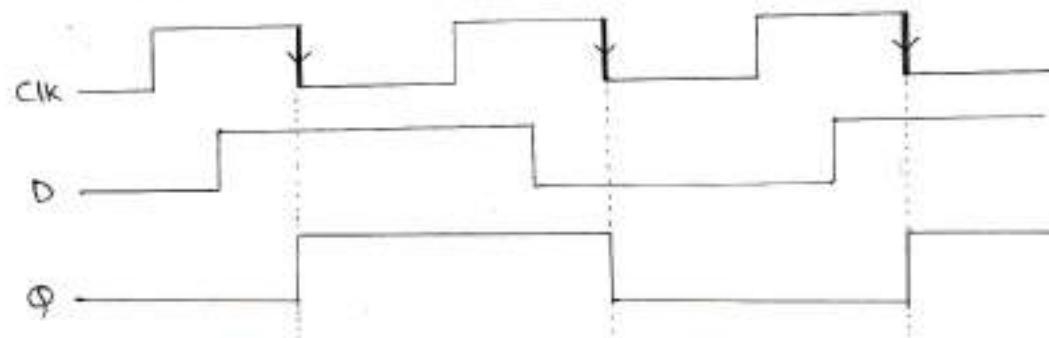
i/p & o/p waveforms of 'tve' edge triggered D flip-flop:-



(ii) logic symbol of '-ve' edge triggered D flip-flop :-



i/p & o/p waveforms of '-ve' edge triggered D flip-flop:-

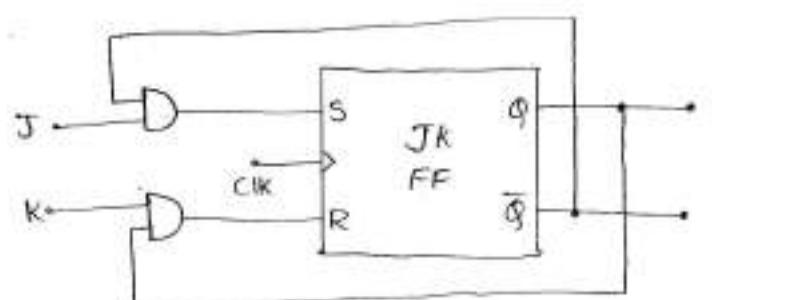


11/09/2014

3. clocked JK flip-flop :-

the uncertainty in the state of an SR FF when  $S=R=1$  can be eliminated by converting it into a JK flip-flop

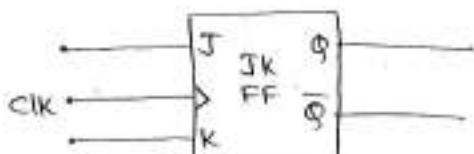
JK flip flop using SR flip flop :-



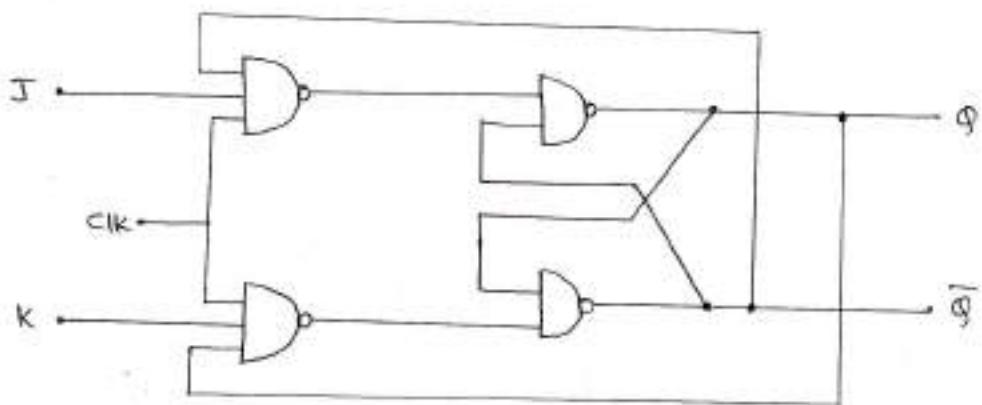
$$S = J\bar{Q}$$

$$R = KQ$$

Logic symbol of JK flip-flop :-



Logic circuit of JK flip-flop using NAND gates :-



Truth table :-

i/p's			o/p	
clk	J	K	$Q_n$	$Q_{n+1}$
0	x	x	0	0
0	x	x	1	1
↑	0	0	0	0
↑	0	0	1	1
↑	0	1	0	0
↑	0	1	1	0
↑	1	0	0	1
↑	1	0	1	1
↑	1	1	0	1
↑	1	1	1	0

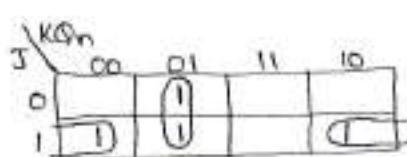
Characteristic table :-

≡

J	K	$Q_{n+1}$
0	0	$Q_n$
0	1	0
1	0	1
1	1	$\bar{Q}_n$

Characteristic equation of JK flip-flop :-

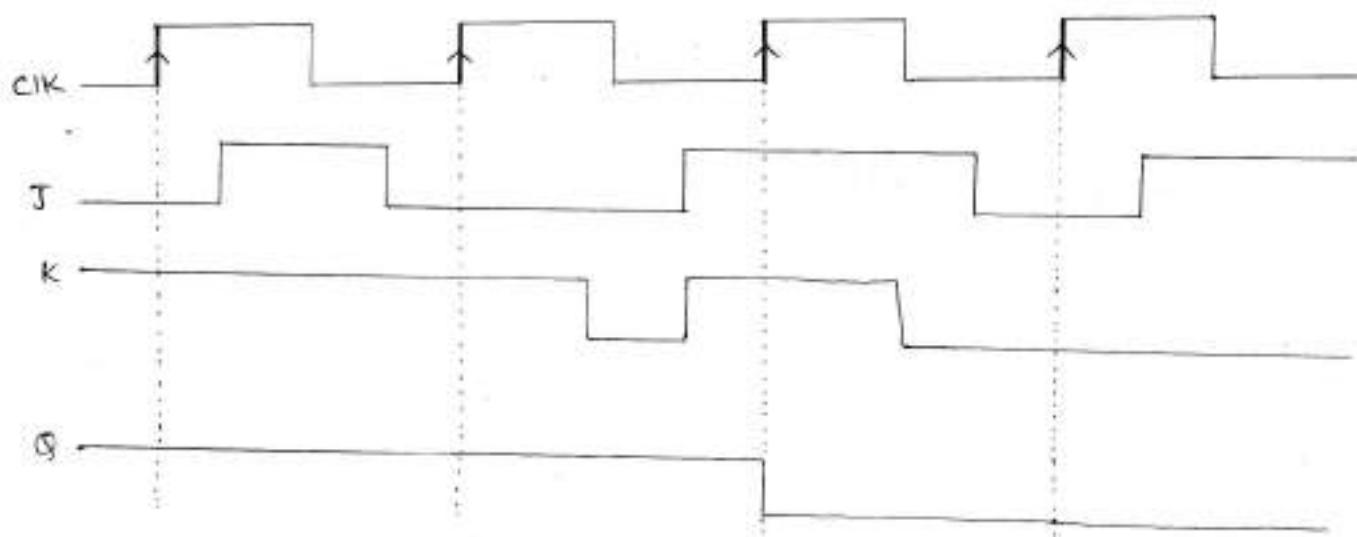
K-map for  $Q_{n+1}$



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

Characteristic equation of JK flip-flop is  $Q_{n+1} = J\bar{Q}_n + \bar{K}Q_n$

i/p & o/p waveforms of 've' edge triggered JK flip-flop:-



16/09/2014

\* Race around condition :-

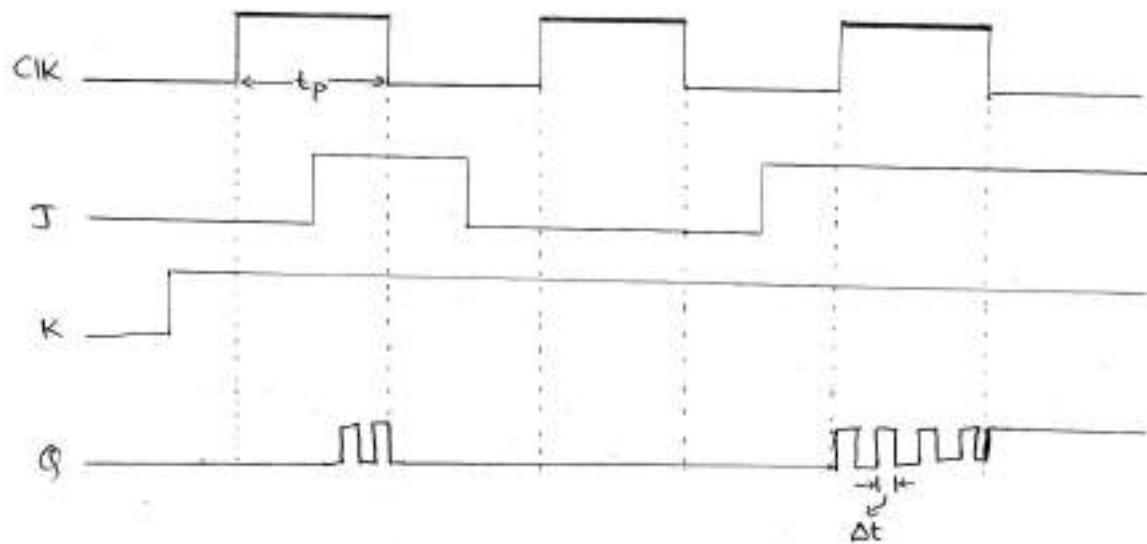


fig: 've level triggered JK FF i/p & o/p waveforms

$t_p \rightarrow$  pulse width

$\Delta t \rightarrow$  propagation delay of 2 level NAND gates  
(considering JK FF using NAND gates circuit)

- \* In JK flip-flop, when  $J = K = 1$ , the o/p continuously toggles in that region ( o/p changes either from 0 to 1 (or) from 1 to 0 ) . which creates disturbance in the o/p . this situation is referred to as the

Race around condition.

\* There are 3 solutions for reducing race around condition.

(i) Use of edge triggering.

(ii) RAC exists when  $t_p \geq \Delta t$ . Thus by keeping  $t_p < \Delta t$ , we can avoid RAC.

(iii) Use of Master slave flip-flop configuration.

\* Master - slave JK flip-flop :-

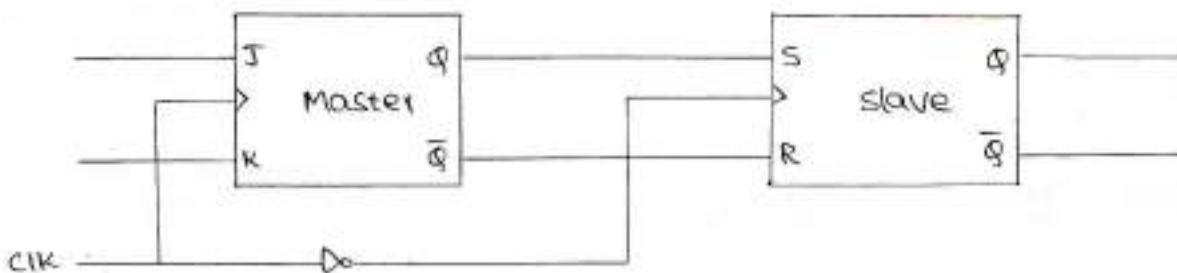


Fig: logic diagram of master - slave JK flipflop

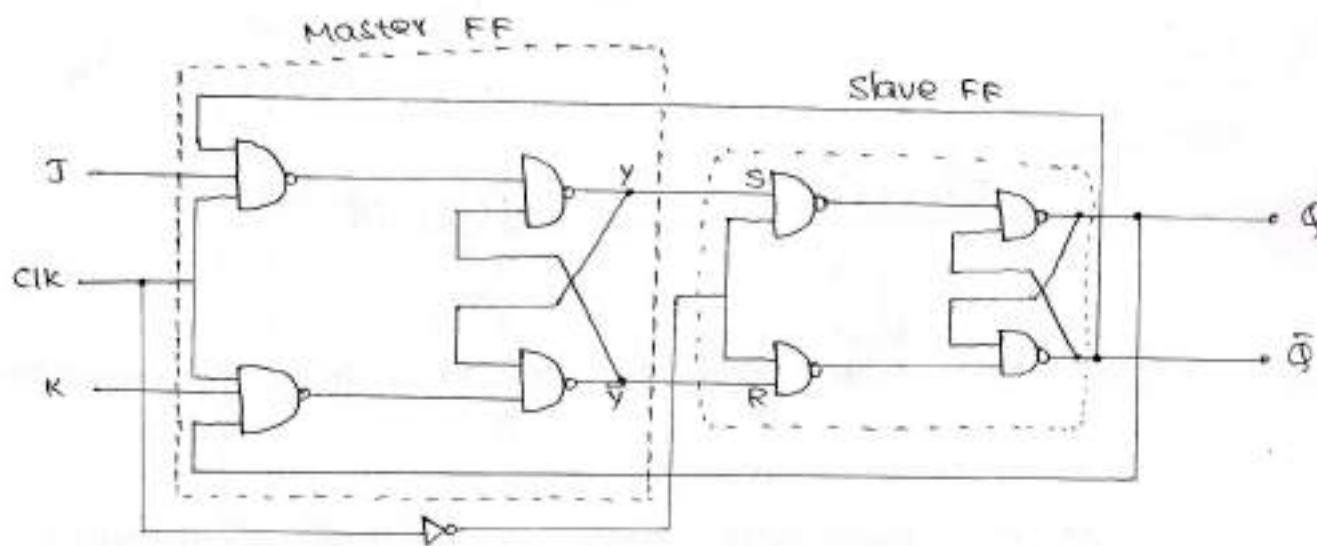


Fig: Logic circuit of master slave JK FF.

\* Master slave JK flip-flop consists of clocked JK flip-flop as the master and clocked SR flip-flop as a slave.

- \* the o/p of the Master flip-flop is fed as an i/p to the slave FF.
  - \* clock signal is connected directly to the Master flip-flop but it is connected through inverter to the slave flip-flop.
- ∴ the information present at the J4 K i/p's is transmitted to the o/p of master flip flop on the '+ve' clock pulse & it is held there until the '-ve' clock pulse occurs. After which it is allowed to pass through to the o/p of slave FF. The o/p of the slave FF is connected to the third i/p of the 3<sup>r</sup> Master JK FF.

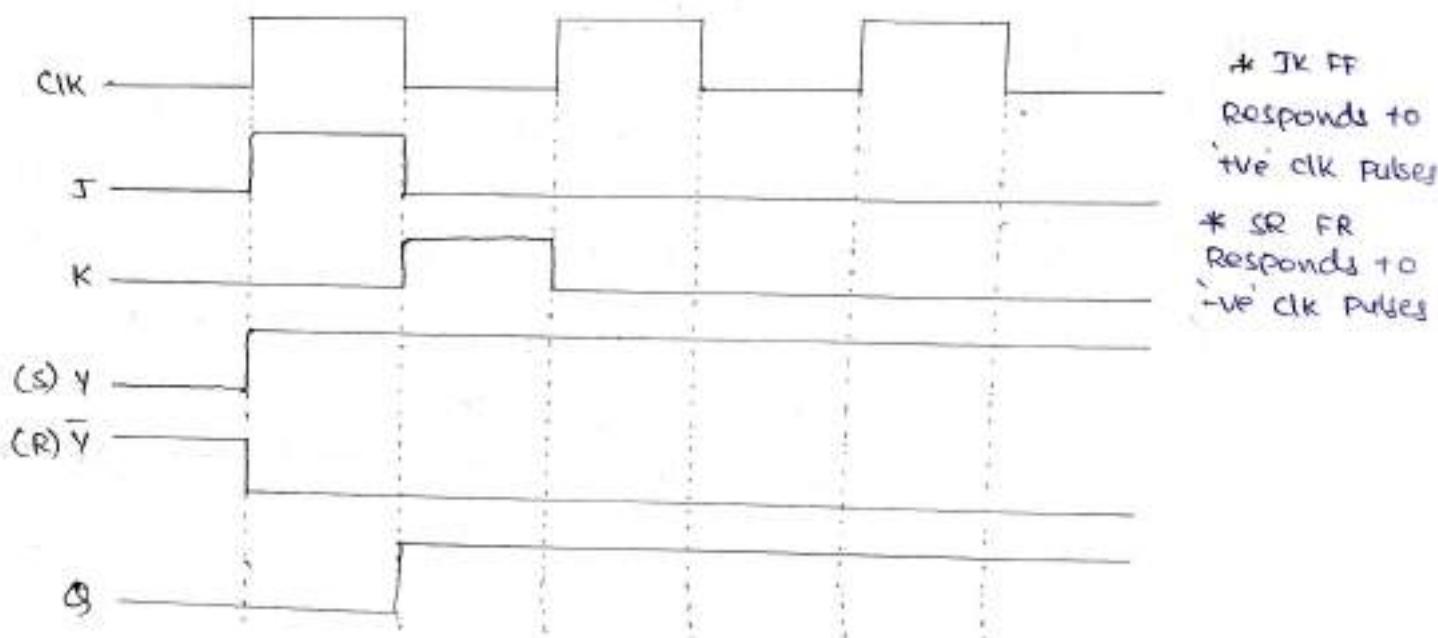
Truth table:-

CLK	J	K	$Q_n$	Master FF o/p Y	Slave FF o/p $Q_{n+1}$
↑	0	0	0	0	No change
↓	0	0	0	NC	0
↑	0	0	1	1	NC
↓	0	0	1	No change	1
↑	0	1	0	0	NC
↓	0	1	0	NC	0
↑	0	1	1	0	NC
↓	0	1	1	NC	0
↑	1	0	0	1	NC
↓	1	0	0	NC	1
↑	1	0	1	1	NC
↓	1	0	1	NC	1
↑	1	1	0	1	NC
↓	1	1	0	NC	1
↑	1	1	1	0	NC
↓	1	1	1	NC	0

\* When  $J=1$  &  $K=1$  master flip-flop toggles on '+ve' clock and slave then copies the o/p of master on the '-ve' clock. At this instant, feedback i/p's to the master flip-flop are complemented but as it is '-ve' half of the clock pulse master FF is inactive.

This prevents race around condition.

i/p & o/p waveforms of Master-slave JK FF :-



Differences b/w latch & flip-flop :-

latch

flip-flop

\* A latch checks all its i/p's continuously & changes its o/p's accordingly at any time.

\* No clock is used.

\* flip-flop samples its i/p's if changes change its o/p's only at a time as determine by a clock signal.

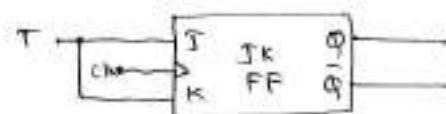
\* A clock is used.

12/09/2014

## Clocked T Flip-flop / Toggle FF :-

- \* The T flip-flop is a modification of the JK flip-flop.
- \* T flip-flop is obtained from a JK flip-flop by connecting J & K inputs together.

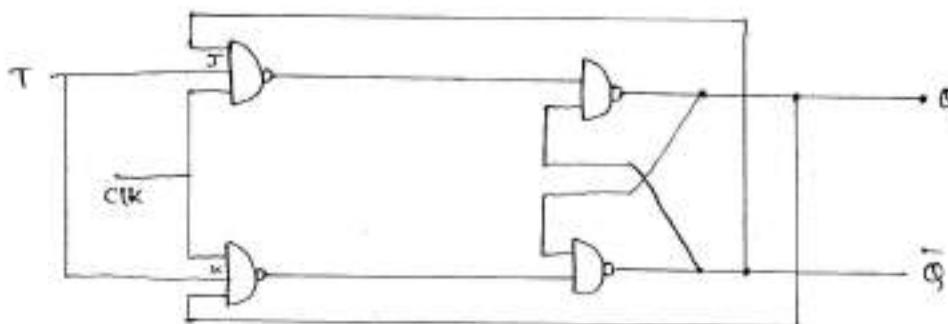
Logic diagram of T FF using JKFF :-



Logic diagram of T FF :-



Logic circuit:-



Truth table:-

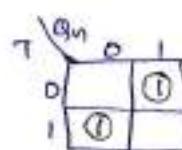
T	$Q_n$	$Q_{n+1}$
0	0	0
0	1	1
1	0	1
1	1	0

Characteristic table :-

T	$Q_{n+1}$
0	$Q_n$
1	$\bar{Q}_n$

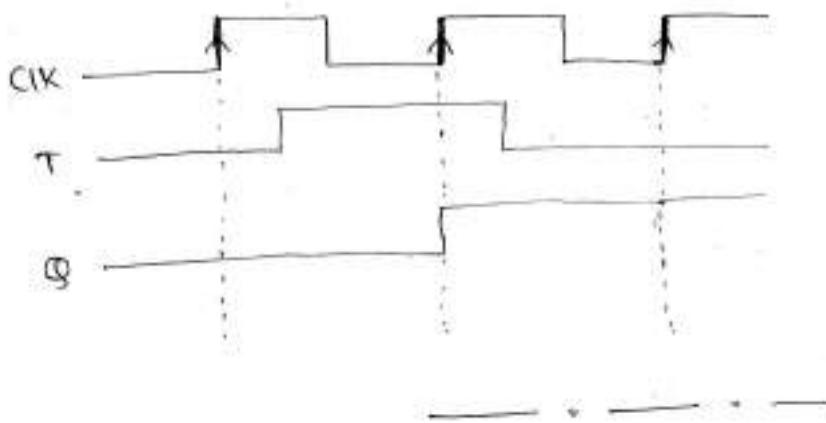
Characteristic equation of T flip-flop:-

K Map for  $Q_{n+1}$



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

## S/I/P and O/I/P waveforms of +ve edge triggered T-FF:



### \* flip-flop excitation tables:-

#### 1. SR flip-flop:-

characteristic table:-

S	R	$Q_{n+1}$
0	0	$Q_n$
0	1	0
1	0	1
1	1	*

→ Indeterminate

Excitation table:-

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

#### 2. D flip-flop:-

characteristic table:-

D	$Q_{n+1}$
0	0
1	1

Excitation table:-

$Q_n$	$Q_{n+1}$	D
0	0	0
0	1	1
1	0	0
1	1	1

#### 3. JK flip-flop:-

characteristic table:-

J	K	$Q_{n+1}$
0	0	$Q_n$
0	1	0
1	0	1
1	1	$\bar{Q}_n$

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

#### 4. T - flip-flop :-

Characteristic table :-

T	$Q_{n+1}$
0	$Q_n$
1	$\bar{Q}_n$

Excitation table :-

S	$Q_{n+1}$	T
0	0	0
0	1	1
1	0	1
1	1	0

Realisation of one flip-flop using other flip-flop:-

Ex:- convert SR FF to D FF.

Sol:- Excitation table for SR to D FF conversion:-

I/P	Present state	New state	FF I/P's	
			S	R
0	0	0	0	X
0	1	0	D	1
1	0	1	1	0
1	1	1	X	0

K-map simplification:-

for S

D	$Q_n$	0	1
0	0	0	0
1	1	1	0

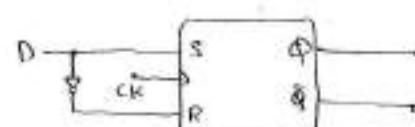
$$\therefore S = D$$

for R

D	$Q_n$	0	1
0	0	X	1
1	1	0	0

$$\therefore R = \bar{D}$$

Logic diagram of DFF Using SR FF :-



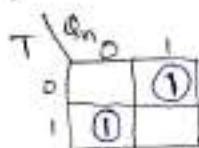
2. Convert D FF to T FF :-

Excitation table for D to T FF conversion :-

i/p	Present State	Next State	FF i/p
T	$Q_n$	$Q_{n+1}$	D
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

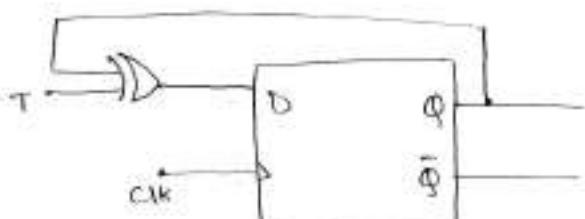
K-Map Simplification :-

for D :-



$$\therefore D = \bar{T}Q_n + T\bar{Q}_n = T \oplus Q_n$$

Logic diagram of T FF using D FF :-



3. Convert SR FF to JK FF.

Excitation table for SR FF to JK FF conversion:-

i/p		Present State	Next State	FF i/p	
J	K	$Q_n$	$Q_{n+1}$	S	R
0	0	0	0	0	x
0	0	1	1	x	0
0	1	0	0	0	x
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	1	x	0
1	1	0	1	1	0
1	1	1	0	0	1

## K-MAP Simplification :-

for S

J\KQ <sub>n</sub>		00	01	11	10
0	X				
1	D	K			

$$\therefore S = J\bar{Q}_n$$

for R

J\KQ <sub>n</sub>		00	01	11	10
0	X	*	1	0	X
1					

$$\therefore R = KQ_n$$

Logic diagram of JK flip-flop using SR FF :-

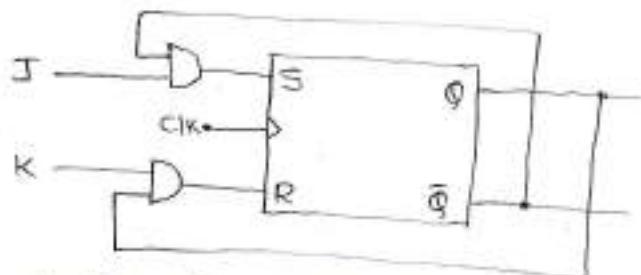


Fig:- logic diagram of JK FF using SR FF.

21/10/2014

Analysis and design of Synchronous (or) Clocked Sequential Circuits

The synchronous or clocked sequential circuits are represented by two models.

- (ii) Moore circuit (or) Moore model (or) Moore Machine
- (iii) Mealy circuit.

1. Moore circuit :-

- (i) Moore circuit o/p depends only on the present state of FF's.
- (ii) I/p changes doesn't effect the o/p.
- (iii) It requires more no. of states for implementing function which is implemented using Mealy circuit.

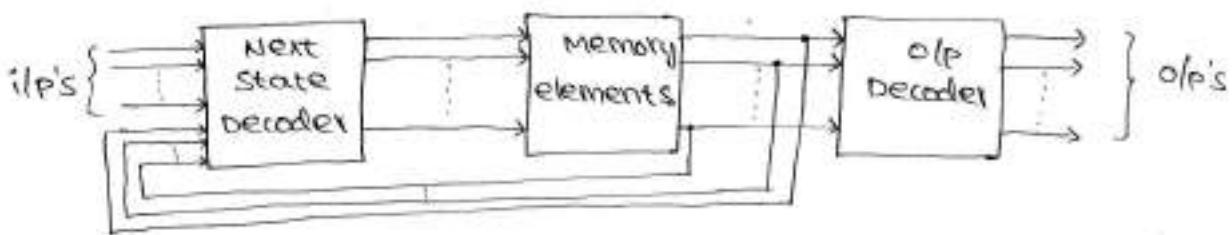


Fig: B.D. of Moore circuit.

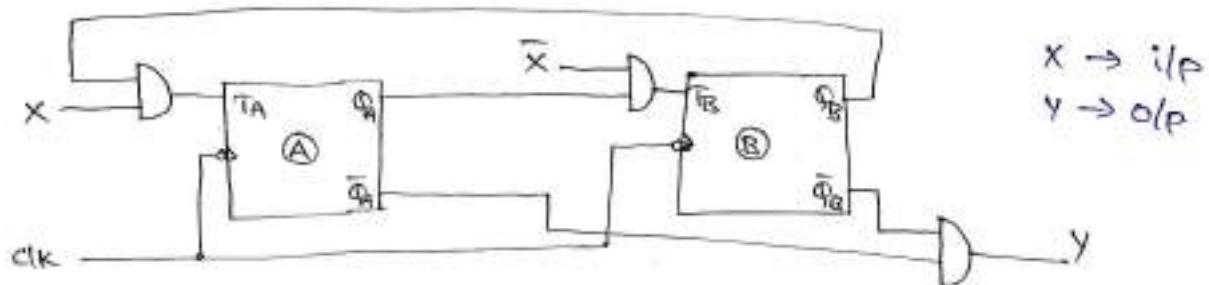


Fig: Example of moore circuit.

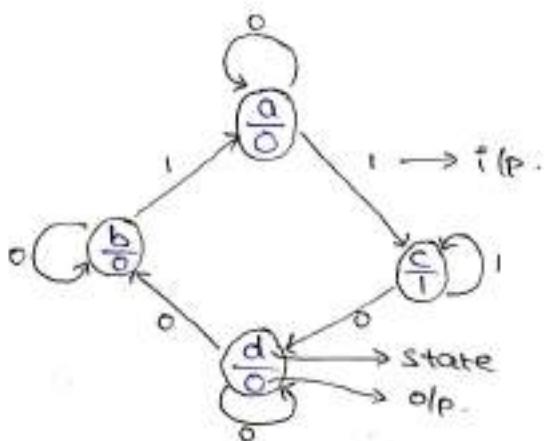


Fig: Example state diag. for moore circuit.

## 2. Mealy circuit:

- (i) the o/p depends on both the present state of the FF's and on the i/p's.
- (ii) i/p changes may effect the o/p of the ckt.
- (iii) it requires less no. of states for implementing the function which is implemented using Moore circuit.

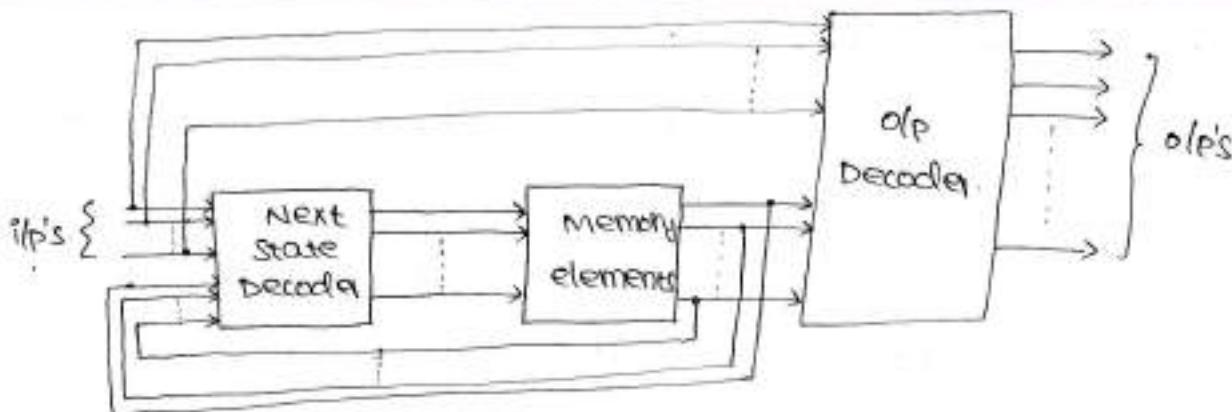


Fig:- B.D. of Mealy circuit.

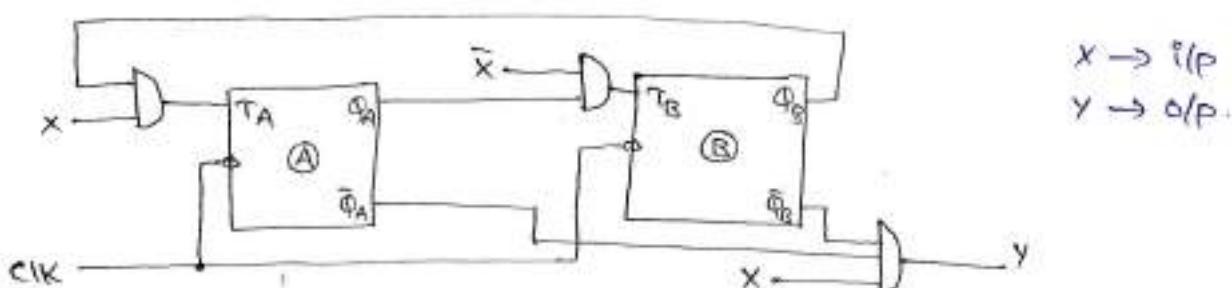


Fig:- Example of mealy circuit

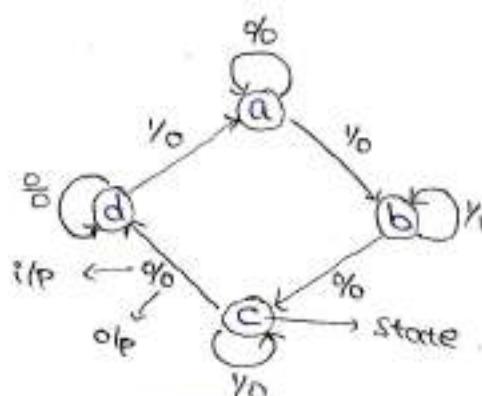


Fig:- example diag. for mealy circuit.

Steps for the design of a synchronous (or) clocked sequential circuits:-

Step 1: Obtain the state table from the given circuit information such as state diagram.

Step 2: Reduce no. of states if possible by state reduction technique.

Step 3: Assign binary values to each state in the state table

Step 4: Determine the no. of FF's needed and

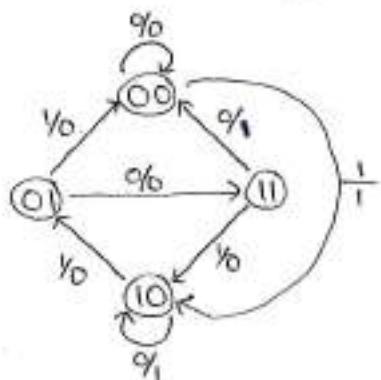
choose the type of FF to be used.

Step 5: From the state table derive the circuit excitation & o/p tables.

Step 6: Using K-map, derive the circuit o/p functions and the FF i/p functions.

Step 7: Draw the logic diagram.

Q:- A sequential circuit has 1 i/p & 1 o/p. The ckt state diagram is shown below. Design the sequential circuit with D-FF.



Sol: Step 1: state diagram

The given state diagram has 4 states with one i/p and one o/p. Considering  $X \rightarrow \text{i/p}$   $Y \rightarrow \text{o/p}$

Present State	Next State				o/p (Y)	
	X=0		X=1		X=0	X=1
	Q <sub>A</sub>	Q <sub>B</sub>	Q <sub>A+1</sub>	Q <sub>B+1</sub>	Y	Y
0 0	0	0	1	0	0	1
0 1	1	1	0	0	0	0
1 0	1	0	0	1	1	0
1 1	0	0	1	0	1	0

Step 2: State Reduction and State Assignment

Here binary values are already assigned to states and it is not possible to reduce no. of states because no two present states, next states and o/p's are same.

Step 3: Determine no. of FF's required.

Determine no. of FF's required using eq.,

$$2^n \geq N \quad n \rightarrow \text{no. of FF's}$$

$$2^n \geq 4 \quad N \rightarrow \text{no. of states}$$

$$\therefore n=2$$

FF's required  $\rightarrow 2$   
 FF's using  $\rightarrow D$  FF's

Step 4: Excitation table

Present State	i/p	Next State		FF i/p's	o/p
		$Q_A$	$Q_B$		
$Q_A$	$Q_B$	X		$D_A$	$D_B$
0	0	0	0	0	0
0	0	1	1	1	0
0	1	0	1	1	0
0	1	1	0	0	0
1	0	0	1	0	1
1	0	1	0	0	0
1	1	0	0	0	1
1	1	1	1	1	0

( $\because$  D-FF  
excitation table)

$Q_n$	$Q_{n+1}$	D
0	0	0
0	1	1
1	0	0
1	1	1

Step 5: K-Map Simplification.

K-map simplification for FF i/p functions & sequential circuit o/p function.

for  $D_A$

$Q_A$	$Q_B$	X	00	01	11	10
0			0	1	1	1
1			1	1	0	0

$$\therefore D_A = Q_A \oplus Q_B \oplus X$$

for  $D_B$

$Q_A$	$Q_B$	X	00	01	11	10
0			0			1
1			1			1

$$\therefore D_B = \bar{Q}_A Q_B \bar{X} + Q_A \bar{Q}_B \bar{X}$$

for Y

$Q_A$	$Q_B$	X	00	01	11	10
0			0	1		
1			1	1	1	1

$$\therefore Y = \bar{Q}_A \bar{Q}_B X + \bar{Q}_A X$$

Step 6: logic diagram

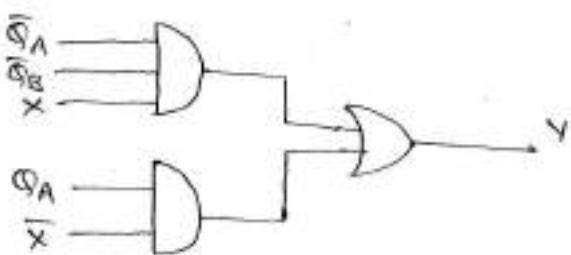
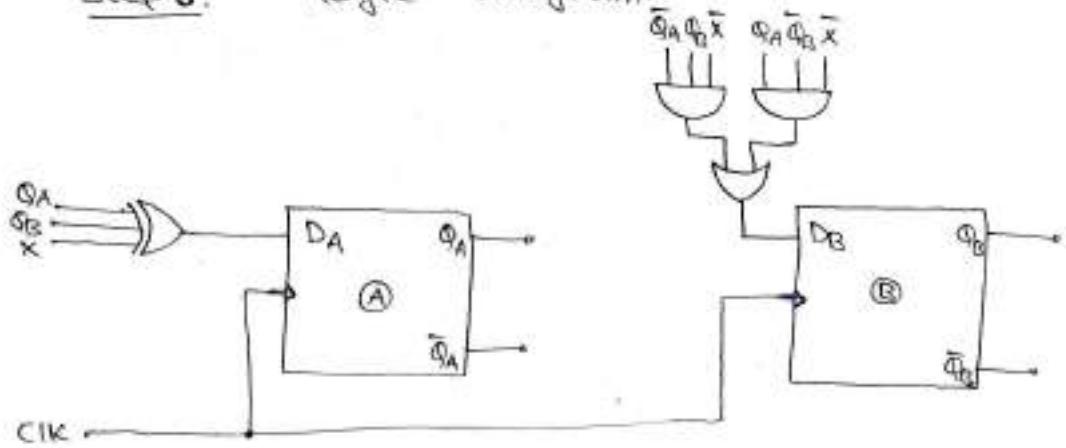
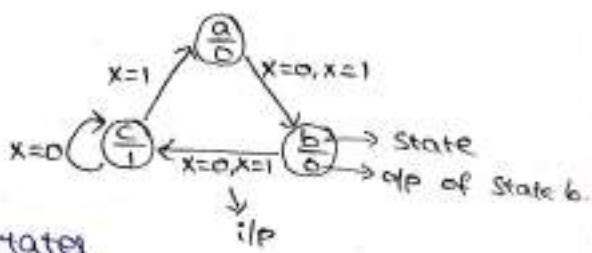


Fig: logic diagram of mealy circuit for given state diagram.

22/10/2014

Q: Realize the sequential circuit for the state diagram shown below.



Sol: Given state diag. has 3 states, one i/p, one o/p.

considering  $x \rightarrow \text{i/p}$   
 $y \rightarrow \text{o/p}$

Step 1: State table

Present state	Next state		Output y
	$x=0$	$x=1$	
a	b	b	0
b	c	c	0
c	c	a	1

### Step 2: State Reduction

there is no possibility of State Reduction since no two states, Next states & o/p's are same.

### Step 3: No. of FF's Required

Determine no. of FF's required using eq.

$$2^n \geq N \quad n \rightarrow \text{no. of FF's}$$

N → no. of States

$$2^n \geq 3$$

$$\therefore n = 2$$

$$\boxed{\text{FF's required} = 2}$$

$$\boxed{\text{FF's using } \rightarrow \text{T FF's}}$$

### Step 4: State Assignment

Assign Binary values to States.

$$a = 00, b = 01, c = 10$$

### Step 5: Excitation table

Present state	i/p	Next state	FF i/p's	o/p
$Q_A$ $Q_B$	X	$Q_{A+1}$ $Q_{B+1}$	$T_A$ $T_B$	Y
0 0	0	0 1	0 1	0
0 0	1	0 1	0 1	0
0 1	0	1 0	1 1	0
0 1	1	1 0	1 1	0
1 0	0	1 0	0 0	1
1 0	1	0 0	1 0	1

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

State 11 is unused state.  $\therefore$  Treat that state as don't care.

### Step 6: K-Map simplification

K-Map simplification for FF's i/p functions and sequential circuit o/p function.

for  $T_A$

$Q_A$	$Q_B$	00	01	11	10
0		1	1		
1		1	X	X	

$$\therefore T_A = Q_A X + Q_B$$

For  $T_B$

$Q_A$	$Q_B$	00	01	11	10
0		1	1	1	1
1			X	X	

$$\therefore T_B = \bar{Q}_A$$

for Y

$Q_A$	$Q_B$	00	01	11	10
0					
1		1	1	X	X

$$\therefore Y = Q_A$$

Step 6: LOGIC DIAGRAM

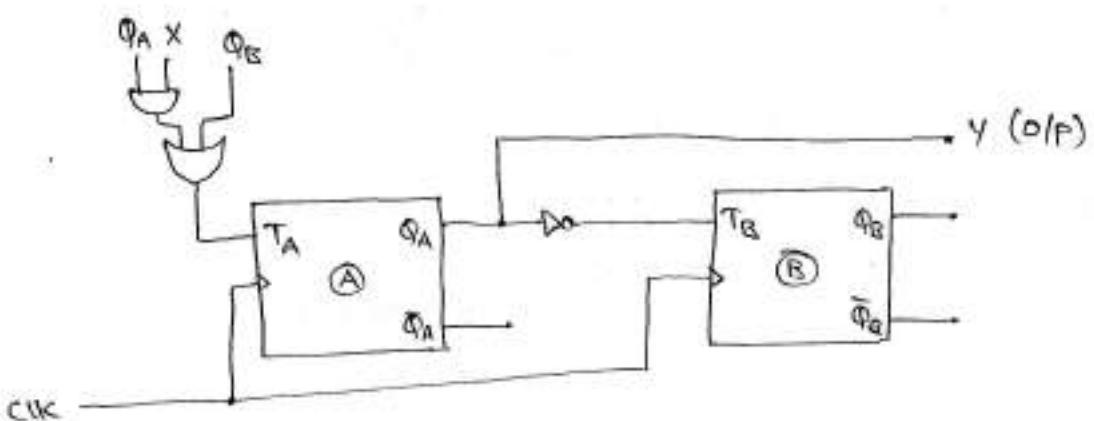


Fig:- logic diagram of moore sequential ckt for given state diagram.

Q:- Design a clocked sequential circuit for State diagram shown below.

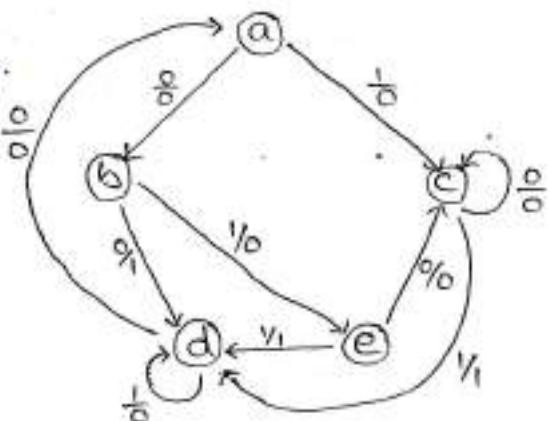
Sol:- Given state diag.

has 5 states with  
one i/p and one o/p.

considering,

$x \rightarrow$  i/p

$y \rightarrow$  o/p



Step 1: State table

Present state	Next state		o/p(y)	
	$x=0$	$x=1$	$x=0$	$x=1$
a	b	c	0	0
b	d	c	1	0
c	d	c	0	1
d	a	d	0	0
e	c	d	0	1

c & e states next states and o/p's are same.  
 $\therefore$  we can cancel either c or e state. if e is cancelled then replace e with c.

Reduced state table

Present State	Next State		o/p(y)	
	$x=0$	$x=1$	$x=0$	$x=1$
a	b	c	0	0
b	d	c	1	0
c	c	d	0	1
d	a	d	0	0

Step 2: Determine no. of FF's required.

No. of FF's required to design seq. ckt can be determined by the eq.,

$$2^n \geq N \quad n \rightarrow \text{no. of FF's}$$

$N \rightarrow \text{no. of States.}$

$$2^n \geq 4 \quad (\because \text{After state reduction } N=4) \\ \therefore n=2$$

$\therefore$  no. of FF's required  $\approx 2$

FF's using  $\rightarrow$  D-FF's.

Step 3: State Assignment

Assign binary values to states

$$a=00, b=01, c=10, d=11$$

Step 4: Excitation table

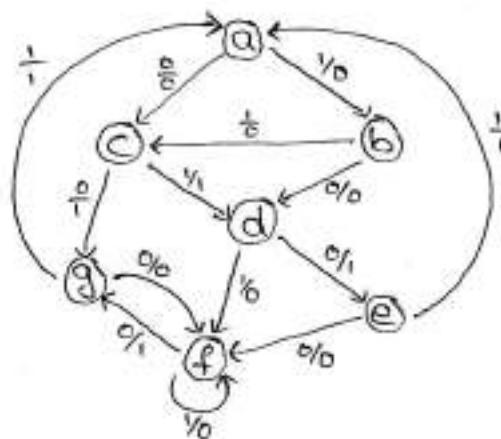
(for procedure refer  
previous problem)

Step 5: K-map simplification

Step 6: Logic diagram

Q: Design a clocked seq. circuit for state diagram shown below.

Sol: The given state diag. has 7 states with one I/P & one O/P.  
considering  $X \rightarrow I/P$   
 $Y \rightarrow O/P$ .



Step1: State table

Present state	Next state		O/P(Y)	
	$X=0$	$X=1$	$X=0$	$X=1$
a	c	b	0	0
b	d	c	0	0
c	g/e	d	1	1
d	e	f	1	0
e	f	a	0	1
f	g/e	f	1	0
x/g	f	a	0	1

e & g states next states and O/P's are same.  
 $\therefore$  we can cancel either e or g state. If 'g' is cancelled then replace g with e.

reduced state table

Present state	Next state		O/P(Y)	
	$X=0$	$X=1$	$X=0$	$X=1$
a	c	b	0	0
b	d	c	0	0
c	e	d	1	1
d	e/f/d	f/d	1	0
e	f/d	a	0	1
x/f	e	f	1	0

d and f states next states and O/P's are same.  
∴ we can cancel either d or f state. If 'f' is cancelled then replace 'f' with 'd'

#### final reduced state table

Present State	Next State		O/P (y)	
	$x=0$	$x=1$	$x=0$	$x=1$
a	c	b	0	0
b	d	c	0	0
c	e	d	1	1
d	e	d	1	0
e	d	a	0	1

#### Step 2:

no.' of FF's required = 3 ( $\because 2^n \geq 5$   
 $n=3$ )

FF's using  $\rightarrow$  D FF's.

#### Step 3: State Assignment

Assign binary values to states

a = 000, b = 001, c = 010, d = 011, e = 100

101, 110, 111 are unused states.

∴ treat them as don't cares.

(i.e) for 101, 110, 111 states, Next states and i/p and o/p's are don't cares.

#### Step 4: Excitation table

#### Step 5: K-map simplification

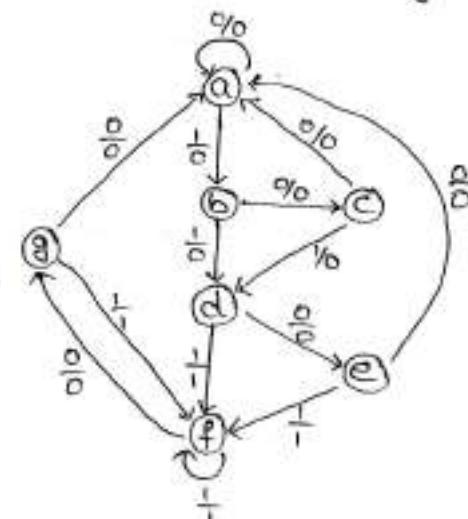
#### Step 6: Logic diagram

25/10/2014

Q. :- obtain the reduced state table and reduced state diagram for a sequential ckt whose state diagram shown below.

Sol:- The give state diagram has 7 states, one i/p & 1 o/p. considering,

$$x \rightarrow \text{i/p}$$



Step 1: state table.

Present state	Next state		o/p	
	$x=0$	$x=1$	$x=0$	$x=1$
a	a b	b a	0 0	0 0
b	c d	d c	0 0	0 0
c	a d	d a	0 0	0 0
d	e f	f e	0 0	1 1
e	a f	f e	0 0	1 1
f	g e	f g	0 0	1 1
x	g a	a f	0 0	1 1
g				

$\therefore$  e & g states next states and o/p's are same  
 $\therefore$  we can cancel either e or g state. If 'g' is cancelled then replace g with e.

reduced state table

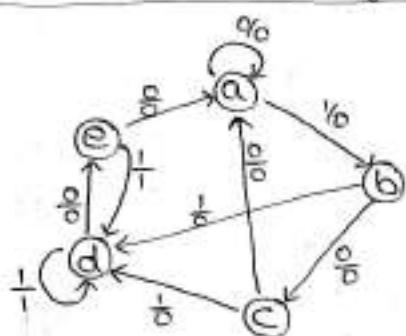
Present state	Next state		o/p	
	$x=0$	$x=1$	$x=0$	$x=1$
a	a b	b a	0 0	0 0
b	c d	d c	0 0	0 0
c	a d	d a	0 0	0 0
d	e f d	f e d	0 0 1	1 1 0
e	a f d	f g d	0 0 1	1 1 0
x	f	e f	0 0	1 1

- $\therefore$  d & f states next states and o/p's are same.  
 $\therefore$  we can cancel either d (or) f state. If 'f' is cancelled then replace 'f' with 'd'.

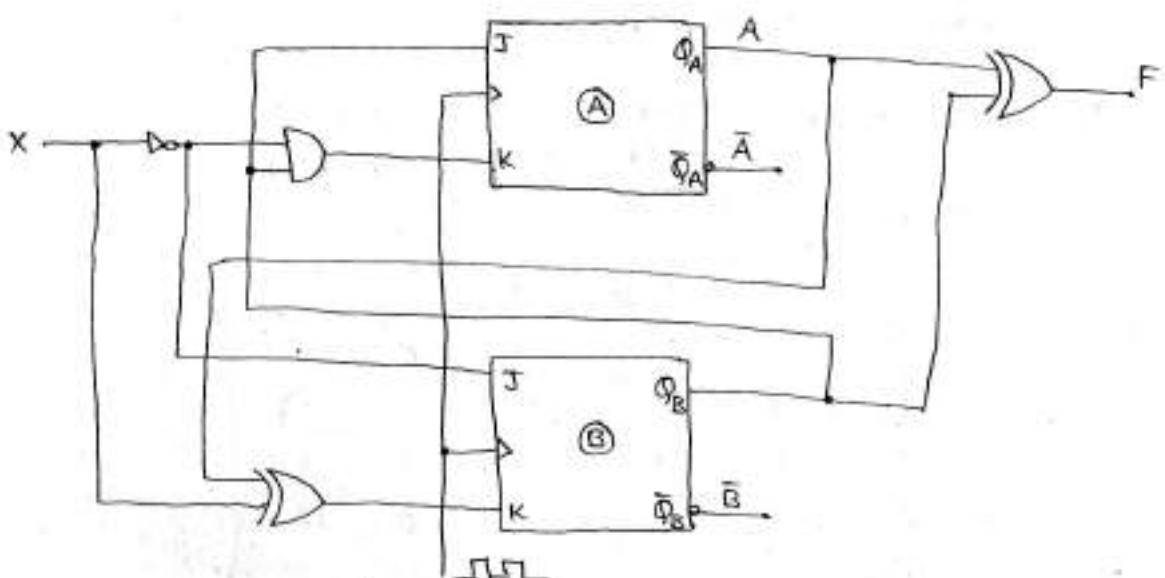
Final reduced state table

Present state	Next state		O/P	
	$x=0$	$x=1$	$x=0$	$x=1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	d	0	1
e	a	d	0	1

Reduced state diagram:



Q:- Derive the transition table, state table & state diagram for the Moore Sequential circuit given below.



Sol: Step 1: Determine the FF ilp Eq's & the o/p eq's of sequential circuit from the above seq. ckt.

FF ilp eq's

$$J_A = Q_B$$

$$K_A = \bar{X}Q_B$$

$$J_B = \bar{X}$$

$$K_B = Q_A \oplus X$$

seq. circuit o/p

$$F = Q_A \oplus Q_B$$

Step 2: Derive the transition eq's.

The transition eq's for JK FF's can be derived from the characteristic eq. of JK FF.

char. eq. of JK FF is  $Q_{n+1} = J\bar{Q}_n + \bar{K}Q_n$

$$Q_{A+1} = J_A \bar{Q}_A + \bar{K}_A Q_A$$

After substituting  $J_A$  &  $K_A$  eqn's,

Above eqn. becomes,

$$\therefore Q_{A+1} = Q_B \bar{Q}_A + (\bar{X}Q_B) Q_A$$

$$Q_{B+1} = J_B \bar{Q}_B + \bar{K}_B Q_B$$

After substituting  $J_B$  &  $K_B$  eqn's,

Above eqn. becomes,

$$\therefore Q_{B+1} = \bar{X} \bar{Q}_B + (Q_A \oplus \bar{X}) Q_B$$

Step 3: Transition table

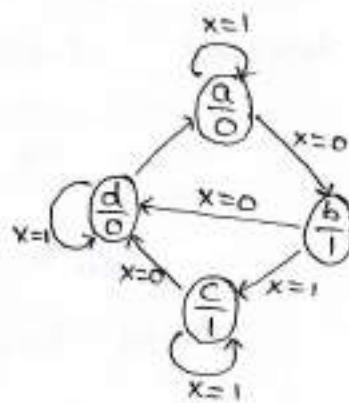
Present State	Next state				Output F
	$x=0$		$x=1$		
$Q_A$	$Q_B$	$Q_{A+1}$	$Q_{B+1}$	$Q_{A+1}$	$Q_{B+1}$
0	0	0	1	0	0
0	1	1	1	1	1
1	0	1	1	1	0
1	1	0	0	1	1

Step 4: State table.

Considering  $a=00$ ,  $b=01$ ,  $c=10$ ,  $d=11$ .

Present State	Next State		O/P (F)
	$x=0$	$x=1$	
a	b	a	0
b	d	c	1
c	d	c	1
d	a	d	0

Step 5: State diagram



$A \text{ (or) } Q_A$

$A^+ \text{ (or) } Q_{A+1} \text{ (or) } A(t+1) \text{ (or) } Q_A^+$

} Different representations

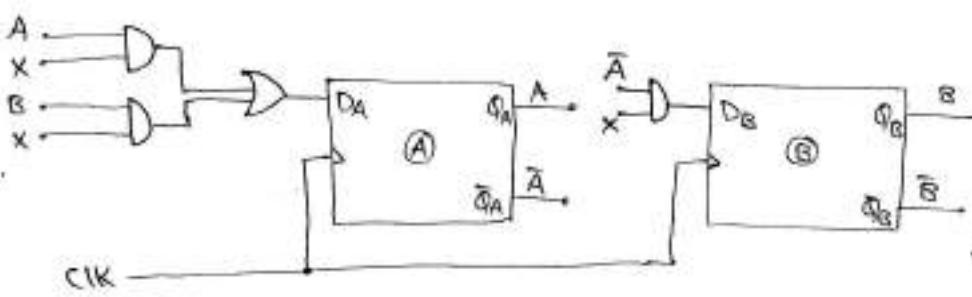
Q. A sequential circuit with two D-FF's A & B and ifp x & o/p y. is specified by the following Next state and o/p equations.

$$A(t+1) = Ax + Bx$$

$$B(t+1) = A'x \quad ; \quad Y = (A+B)x'$$

- (i) Draw the logic diagram of the ckt.
- (ii) Derive the State table
- (iii) Derive the State diagram.

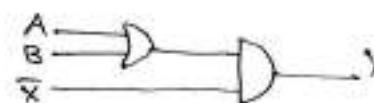
Sol:- (i) logic diagram



for D-FF,  
 $D(p) = \bar{D}(p)$ .

$$\therefore D_A = A(t+k)$$

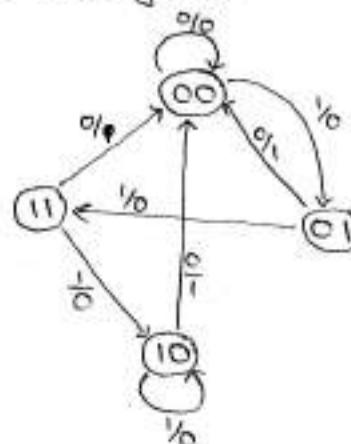
$$D_B = B(t+k)$$



(ii) State table

Present state A B	Next state				o/p (Y) x=0 x=1	
	x=0		x=1			
	A(t+k)	B(t+k)	A(t+k)	B(t+k)		
0 0	0 0	0 0	0 1	0 0	0 0	
0 1	0 0	0 0	1 1	1 0	1 0	
1 0	0 0	0 0	1 0	1 0	1 0	
1 1	0 0	0 0	1 0	1 0	1 0	

(iii) state diagram

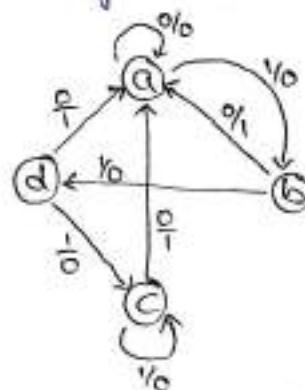


considering  $a=00, b=01$   
 $c=10, d=11$

Present state	Next state		o/p(y) x=0 x=1
	x=0	x=1	
a	a	b	0 0
b	a	d	1 0
c	a	c	1 0
d	a	c	1 0

(or)

considering  $a=00, b=01, c=10, d=11$

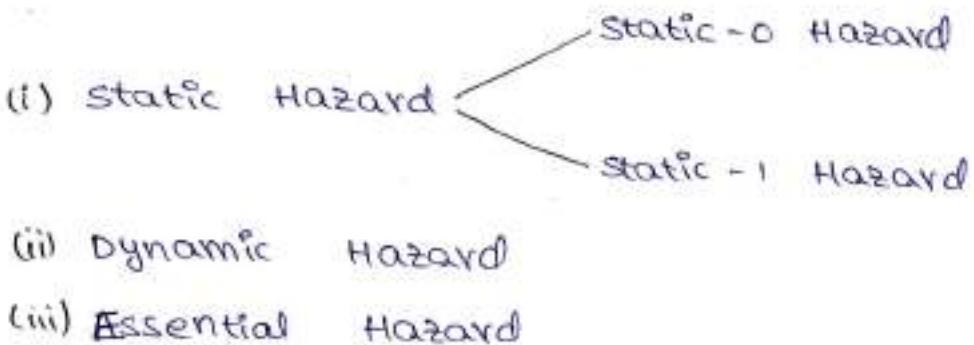


## \* Hazards :-

- Hazards are unwanted switching transients that may appear at the o/p of a circuit because different paths exhibit different propagation delays.
- Hazards occur in combinational circuits, where they may cause a temporary false o/p value. When this condition occurs in sequential circuits, it may result in a transition to a wrong stable state.

## Classification of Hazards :-

There are 3 different types of Hazards.



### 1. (i) static - 1 Hazard :-

When a circuit output goes to momentarily '0', where it has to remain at constant '1', then we say that circuit has "static-1 Hazard".



Fig:- static-1 hazard.

### 1. (ii) static - 0 Hazard :-

When a circuit output goes to momentarily '1', where it has to remain at constant '0',

then we say that circuit has "Static-0 Hazard".

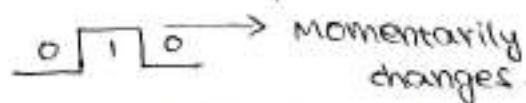


Fig:- static-0 Hazard.

## 2. Dynamic Hazard :-

when the o/p is supposed to change from '0' to '1' (or '1' to '0'), if o/p changes more no. of times then this transient is called "Dynamic Hazard".

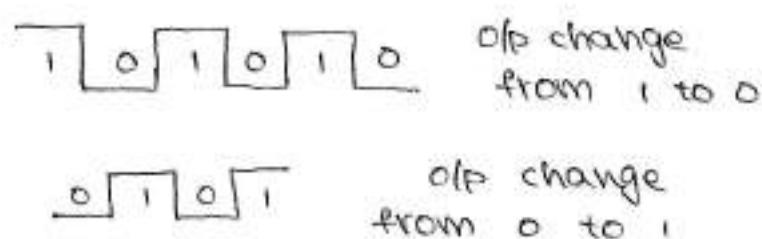
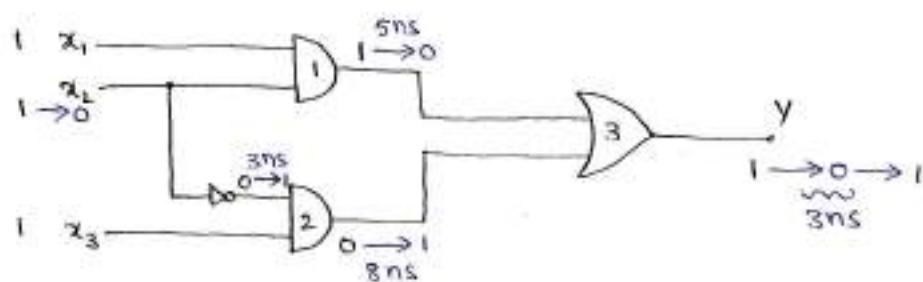


Fig:- Dynamic Hazard.

## combinational circuit with static-1 Hazard



propagation delay of AND gate = 5 ns

NOT gate = 3 ns

∴ propagation delay of 1<sup>st</sup> path is 5 ns

2<sup>nd</sup> path is 8 ns.

This circuit displays wrong o/p '0' for 3ns when o/p changes from 111 to 101.

## 3. Essential Hazards:-

→ the static and dynamic Hazards can occur in combinational as well as sequential logic circuits.

→ Essential Hazards occur in sequential circuits only.

→ An essential hazard is caused by unequal delays along two or more paths that originate from the same input.

Q.:- Realize the boolean function  $T(x,y,z) = \Sigma(1,3,4,5)$  using logic gates for hazard free.

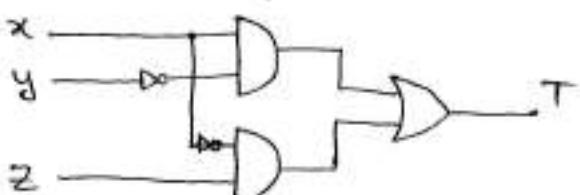
Sol:- K-map simplification 4 logic circuit of above function T with hazard are :-

K-map

	00	01	11	10
0	1	1		
1	1	1		

$$\therefore T = x\bar{y} + \bar{x}z$$

logic circuit



For designing Hazard free circuit :-

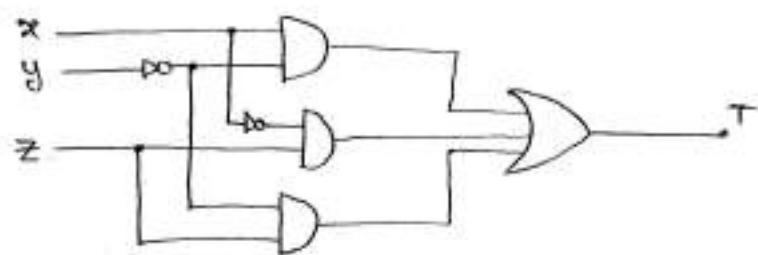
K-map

	00	01	11	10
0	1	0	1	
1	1	0		

(∴ combined all the possible groups)

$$\therefore T = x\bar{y} + \bar{x}z + \bar{y}z$$

Hazard free logic circuit:



22/09/2014

UNIT-IV

1

REGISTERS AND COUNTERSApplications of flip-flops :-

- It can be used as a memory element.
- It can be used to eliminate key debounce.
- It is used as a basic building block in sequential circuits such as counters and registers.
- It can be used as a delay element.
- (E)

REGISTERS :-

- Register is a group of flip-flops.
- n-bit Register consists of 'n' number of flip-flops and it stores 'n' bit binary information.

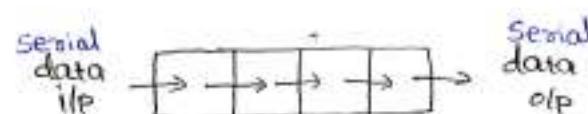
Types of shift Registers :-1. Serial In serial Out shift register (SISO) :-

Fig: Data flow in SISO shift right register

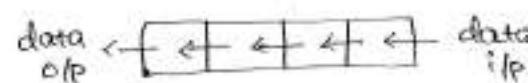


Fig: Data flow in SISO shift left register.

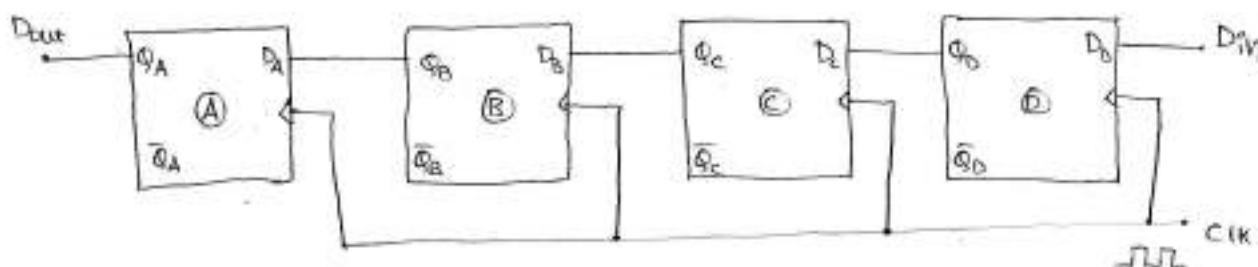


Fig: 4-bit SISO shift left register.

i/p to reg 1011

Clk	$Q_A$	$Q_B$	$Q_c$	$Q_D$	reg. o/p (Q <sub>out</sub> )
0	0	0	0	0	
$\uparrow$	0	0	0	1	
$\uparrow$	0	0	1	0	
$\uparrow$	0	1	0	1	
$\uparrow$	1	0	1	1	→ i/p data stored in register.
$\uparrow$	0	1	1	0	
$\uparrow$	1	1	0	0	
$\uparrow$	1	0	0	0	→ all 4 data i/p bits are transferred to o/p side
$\uparrow$	0	0	0	0	→ reg. cleared.

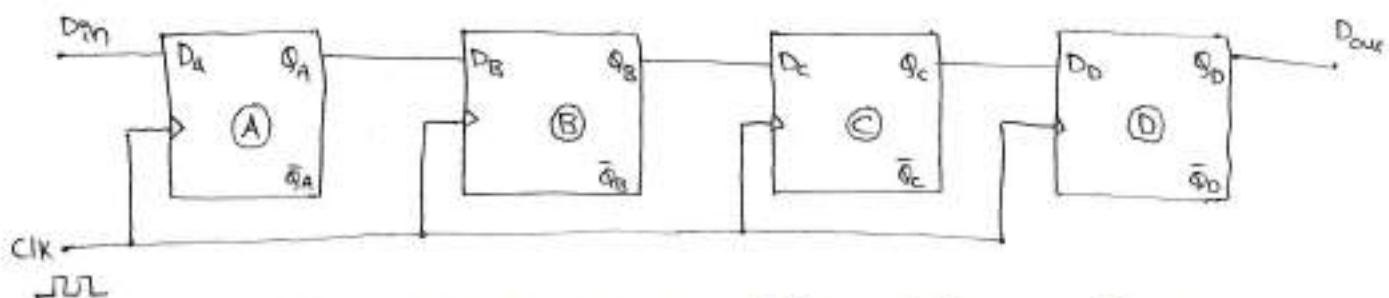


Fig:- 4-bit SISO shift right register

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

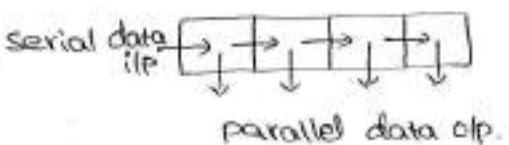


Fig:- data flow in SIPO shift right register

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

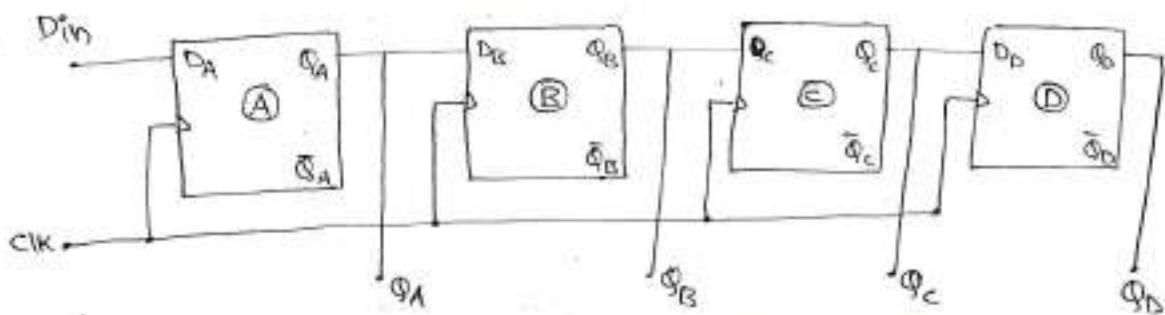


Fig: 4-bit SIPO shift right register.

$D_{in}$  → 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 referred to as loading Register.
- \* If all the bits of Register are loaded simultaneously with a common clock pulse, we say that the loading is done in parallel.

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

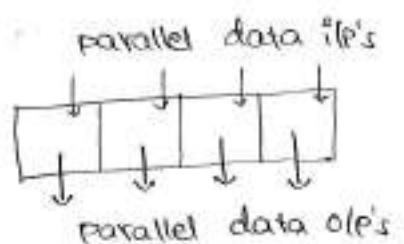


Fig: data flow in PIPO shift register.

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

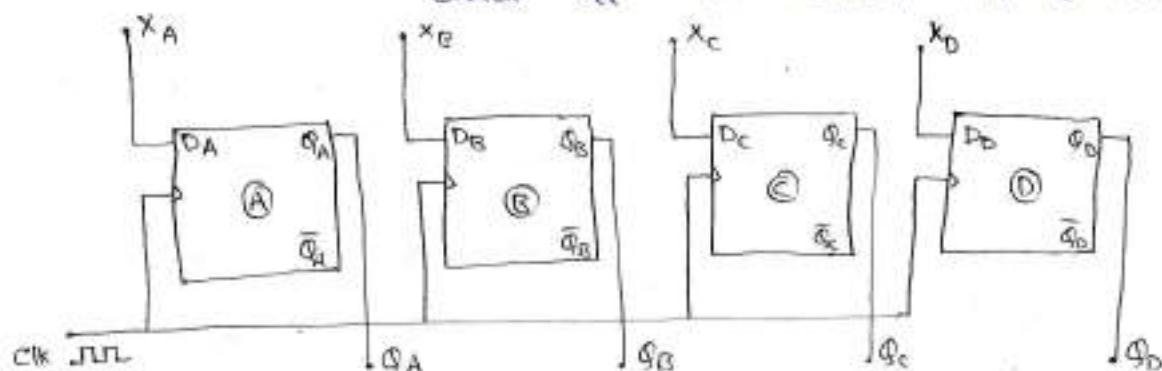


Fig: 4-bit PIPO shift register.

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

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

03/10/2014

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

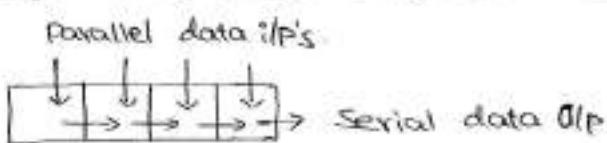


Fig: Dataflow in PISO shift right register.

I/P data entered into register parallelly but o/p is taken

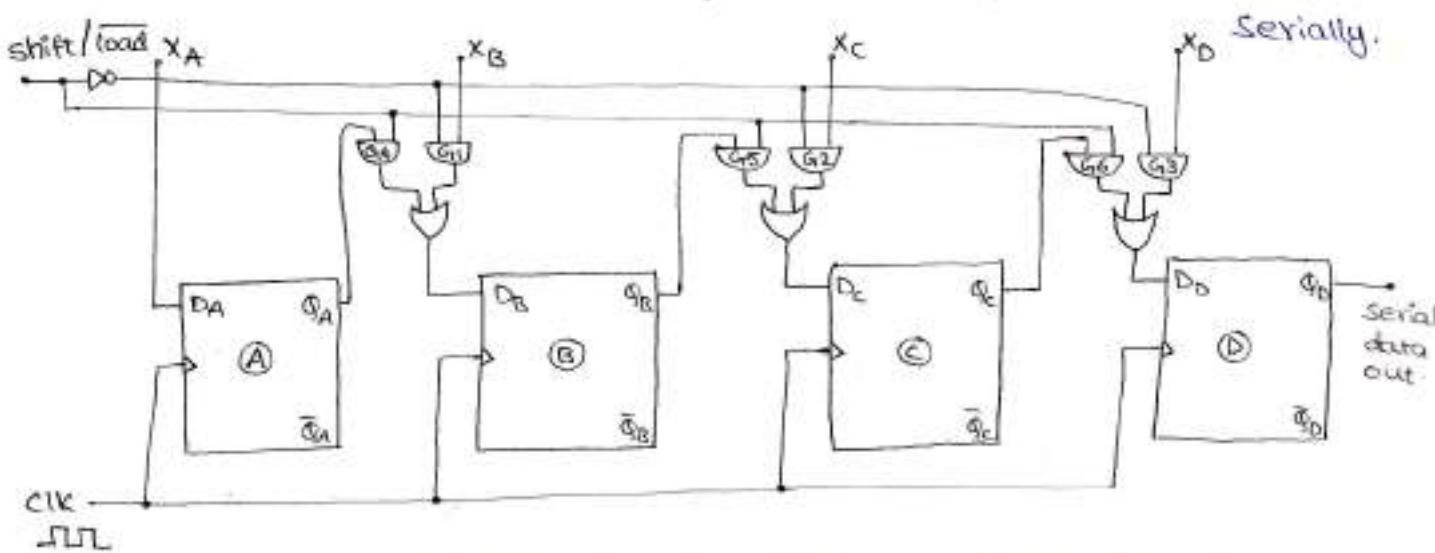


Fig: 4-bit PISO shift right register.

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

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

when  $\overline{\text{shift}}/\overline{\text{load}} = 1$ , 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  $\Rightarrow$  flip-flop will set

If set is low  $\Rightarrow$  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.

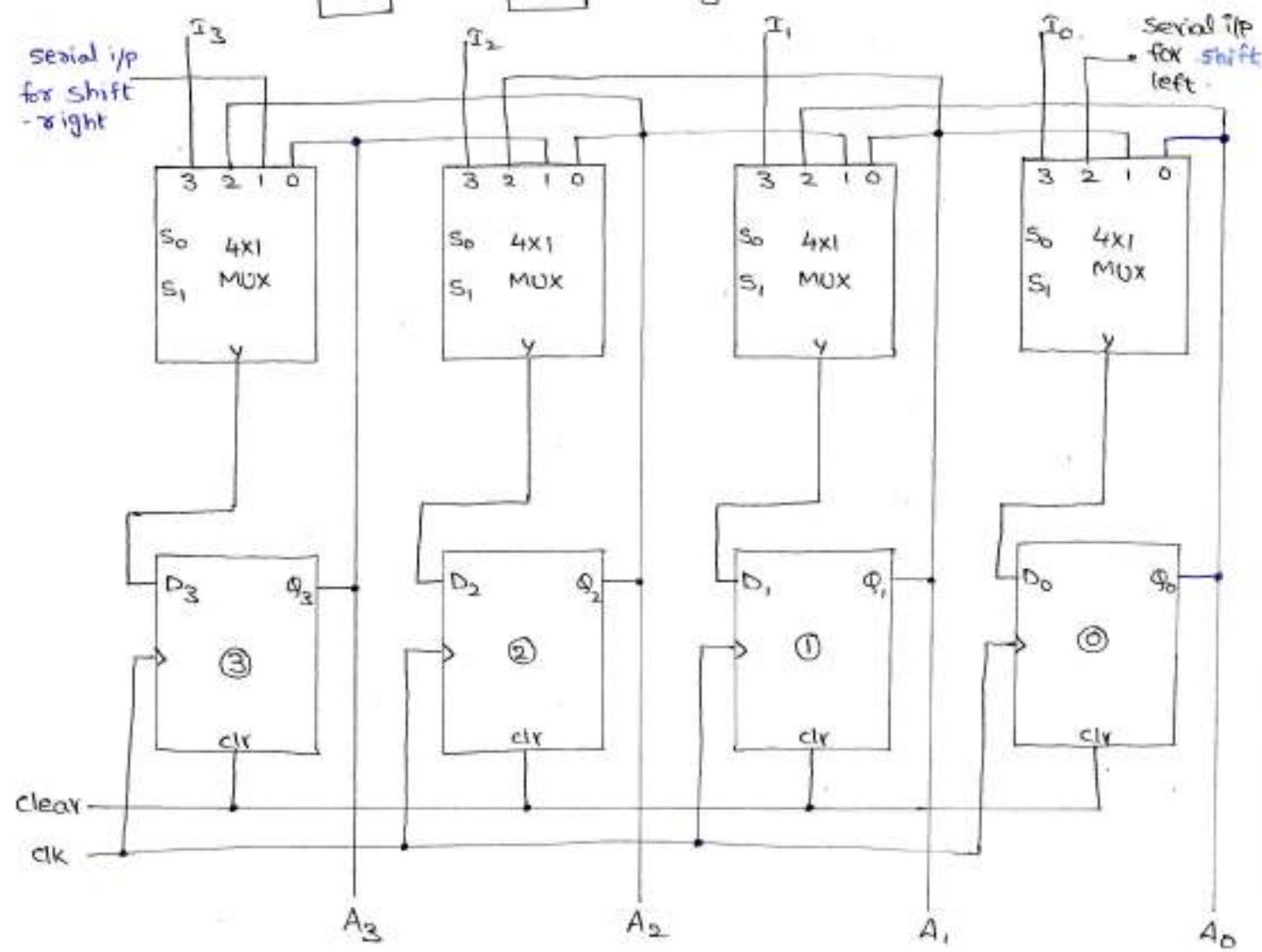
\* Universal shift register :-

If the register has both <sup>shift</sup> (left & Right) and parallel load capabilities, it is referred 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 .



$S_0, S_1 \rightarrow$  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 <sub>1</sub> S <sub>0</sub>	Reg. operation
0    0	NO change
0    1	Shift right
1    0	Shift left
1    1	Parallel load.

I<sub>3</sub> I<sub>2</sub> I<sub>1</sub> I<sub>0</sub> → parallel i/p's  
A<sub>3</sub> A<sub>2</sub> A<sub>1</sub> A<sub>0</sub> → parallel o/p's

for '00' we observe all '0' connections in MUX,  
gives o/p to D<sub>1</sub>, D<sub>2</sub>, D<sub>3</sub>, D<sub>0</sub>.

Similarly, for '11' we observe all '1' connections in MUX,  
gives o/p to D<sub>0</sub>, D<sub>1</sub>, D<sub>2</sub>, D<sub>3</sub>.

### \* Serial Adder :-

It performs serial addition.

Block diagram:-

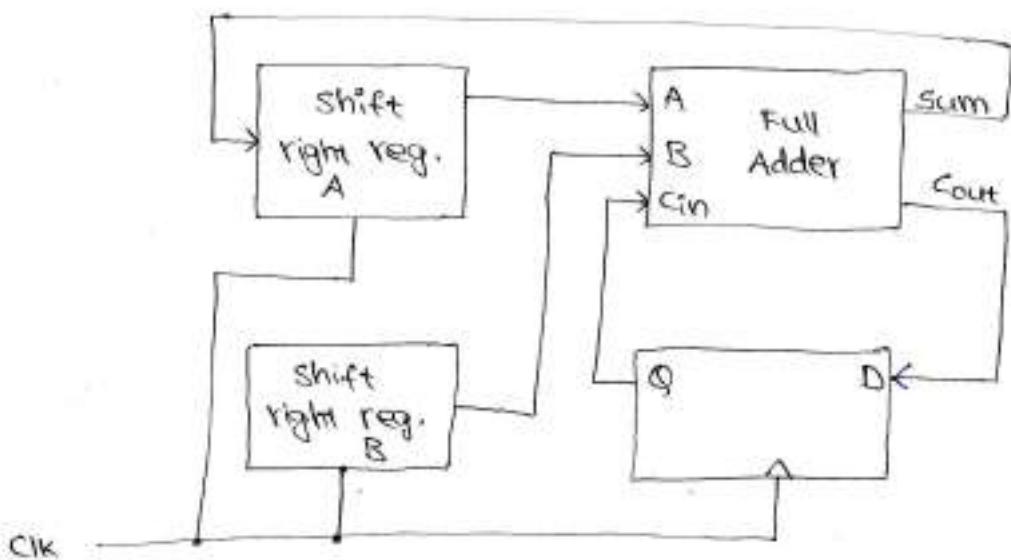


Fig: Serial Adder

let us consider two <sup>4-bit</sup> numbers →  $\begin{array}{r} 1011 \\ + 1001 \\ \hline \end{array}$  → Intermediate carry  
considering A & B are 4 bit registers  
( SISO shift right reg )  
A reg. content is 1011.  
B reg. content is 1001.

$$\begin{array}{r}
 1011 \\
 + 1001 \\
 \hline
 \underbrace{1010}_\text{end carry} \quad \underbrace{0}_\text{sum}
 \end{array}$$

serial adder performs operation.

$$A \leftarrow A + B$$

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, Cin bits and transfers sum to A reg. & Cout to D-FF.

After performing serial addition contents of

reg A  $\rightarrow$  0100 (sum)

B  $\rightarrow$  0000

FF D  $\rightarrow$  1 (end carry)

### Serial Transfer :-

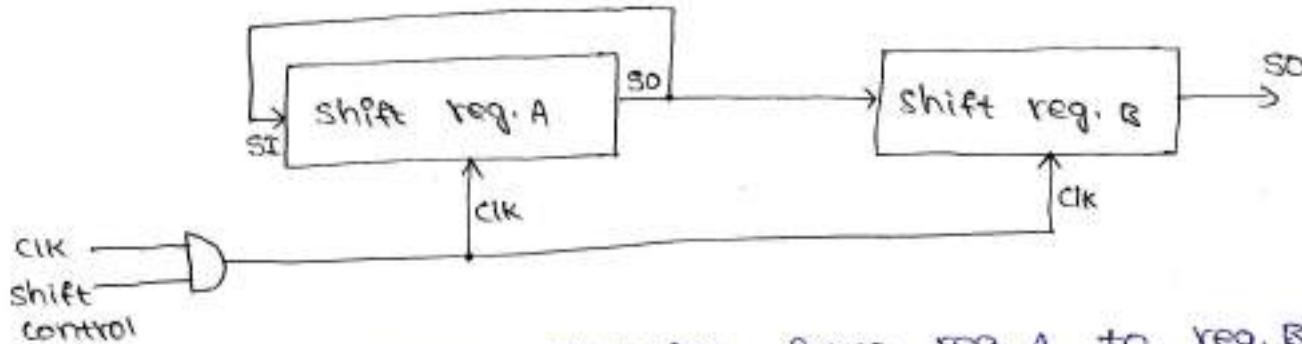


Fig:- (a) serial transfer from reg. A to reg. B.

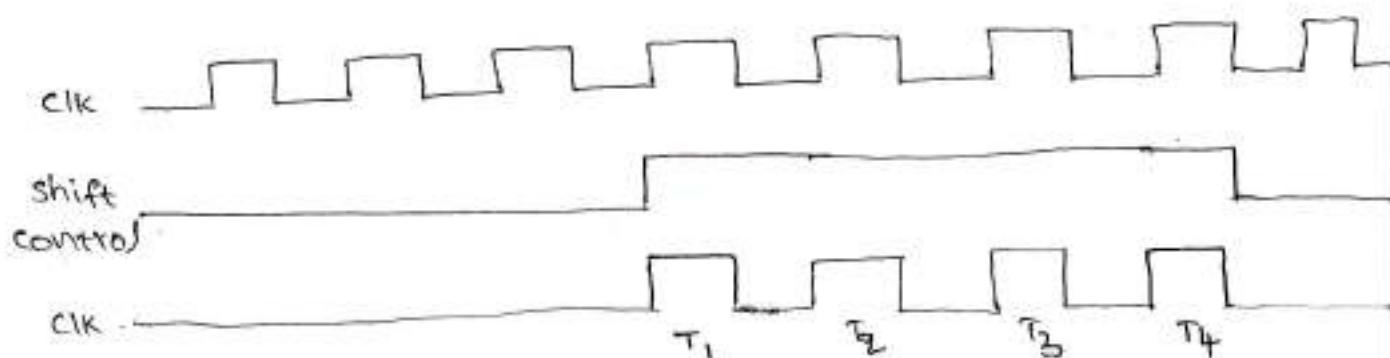


Fig:- (b) Timing diagram

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

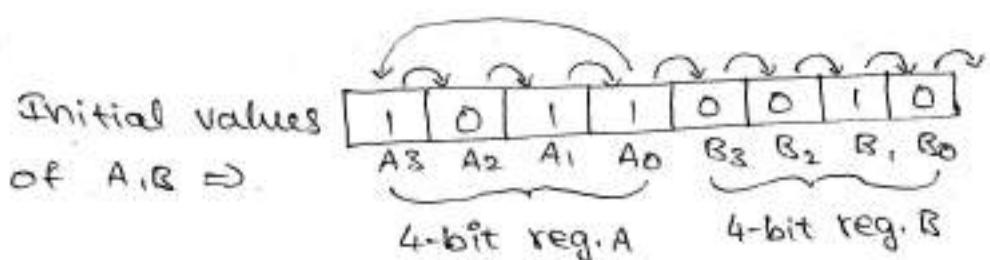
SO  $\rightarrow$  serial out ; SI  $\rightarrow$  serial in.

The above ckt performs operation

$B \leftarrow A$  or  $A \rightarrow B$  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 if/p only when shift control = 1.



### serial transfer

Timing pulse	Shift reg. A	Shift reg. B
Initial value of reg.	1011	0010
After $T_1$	1101	1001
$T_2$	1110	1100
$T_3$	0111	0110
$T_4$	1011	1011

After  $T_4$ , both A & B reg.'s Content is same.

(i.e)  $A=B$ .

## COUNTERS:-

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

(or)

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

\* counters are available in two categories.

1. Asynchronous counters.
2. Synchronous counters.

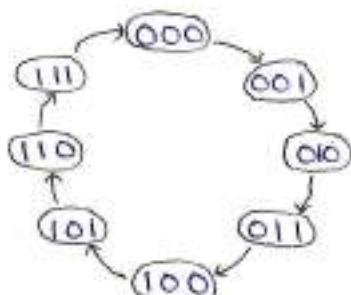
Asynchronous counters	Synchronous counters
<ul style="list-style-type: none"> <li>1. In this type of counter, flip-flops are connected in such a way that o/p of 1<sup>st</sup> FF drives the clk for the next FF.</li> <li>2. All the FF are not clocked simultaneously.</li> <li>3. logic circuit is very simple even for more number of states</li> <li>4. Main drawback of these counters is their low speed as the clk is propagated through no. of FF before it reaches last FF.</li> </ul>	<ul style="list-style-type: none"> <li>1. In this type, there is no connection b/w o/p of 1<sup>st</sup> FF and clk i/p of the next FF.</li> <li>2. All the FF are clocked simultaneously,</li> <li>3. Design involves complex logic circuits as no. of states increased.</li> <li>4. As the clk is simultaneously given to all FF's there is no problem of propagation delay. Hence they are high speed counters.</li> </ul>

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 :-



## High/Logic'1'

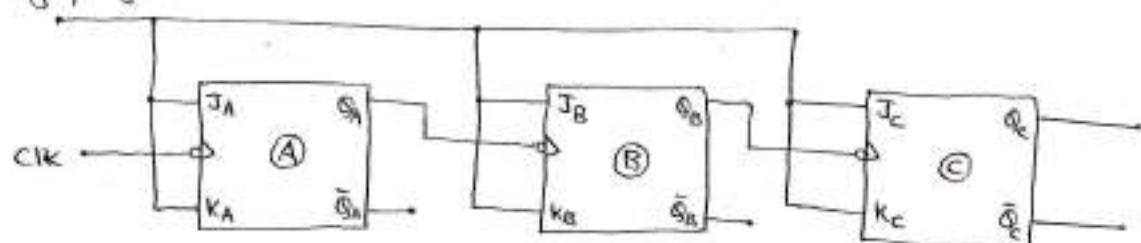
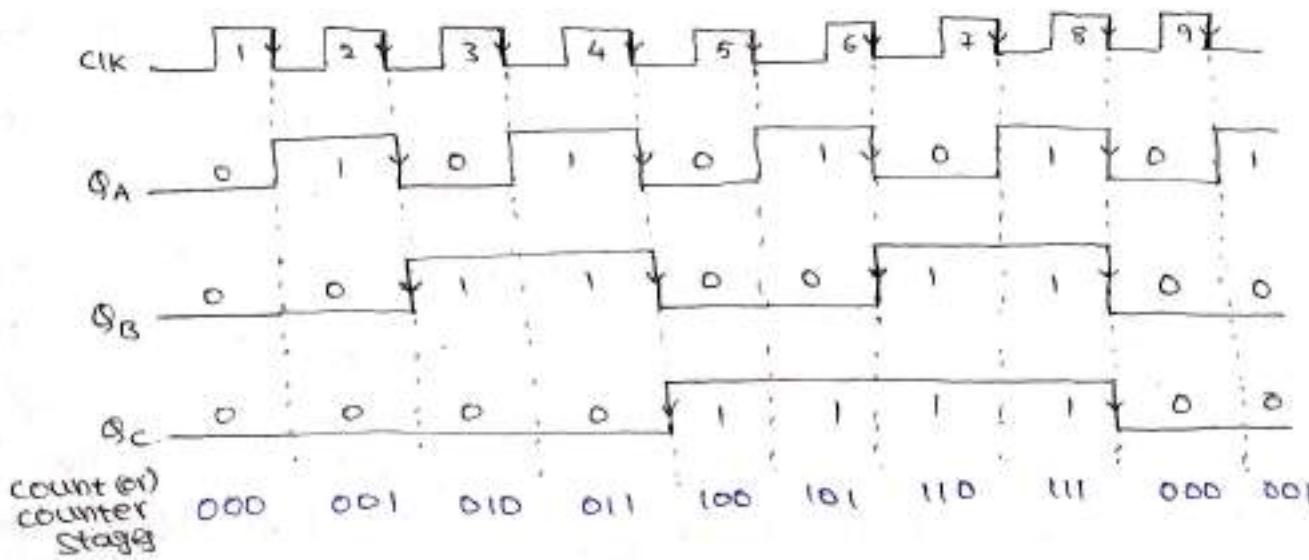


Fig. 3-bit ripple up counter

using 've' edge triggered JK FF's.

counter o/p  $\rightarrow$   $Q_C$   $Q_B$   $Q_A$   
 MSB                    LSB

## Q/P wave forms



\* 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

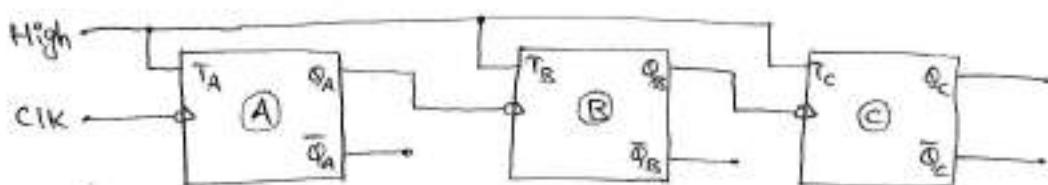


Fig: 3-bit Asynchronous up counter using  
'-ve' edge triggered T-FF's.

counter o/p  $\rightarrow Q_c \ Q_B \ Q_A$   
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.

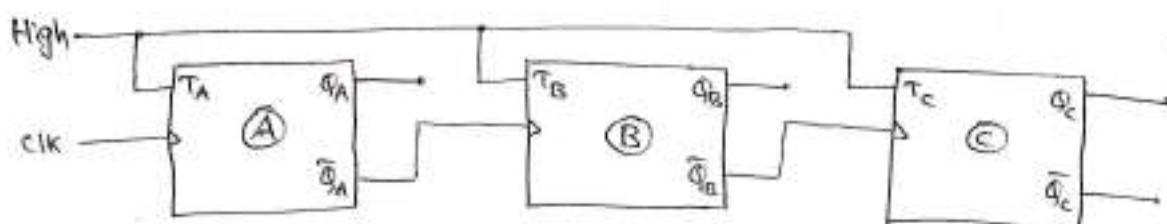


Fig: 3-bit ripple up counter using '+ve' edge triggered T-FF's.

counter o/p  $\rightarrow Q_c \ Q_B \ Q_A$   
MSB            LSB

## 2. 3-bit Asynchronous down counter :-

Down counter counts in downward direction. (ie)  $n, n-1, n-2, \dots, 2, 1, 0$ .

State diagram:-

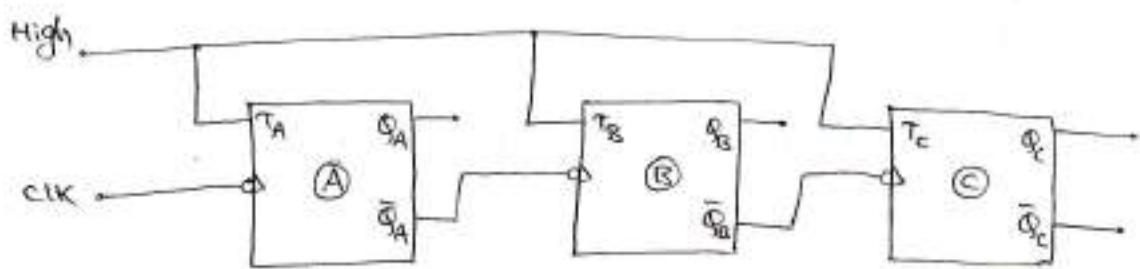
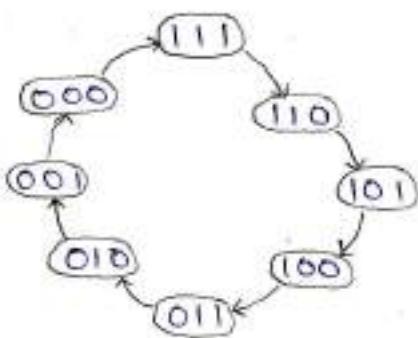
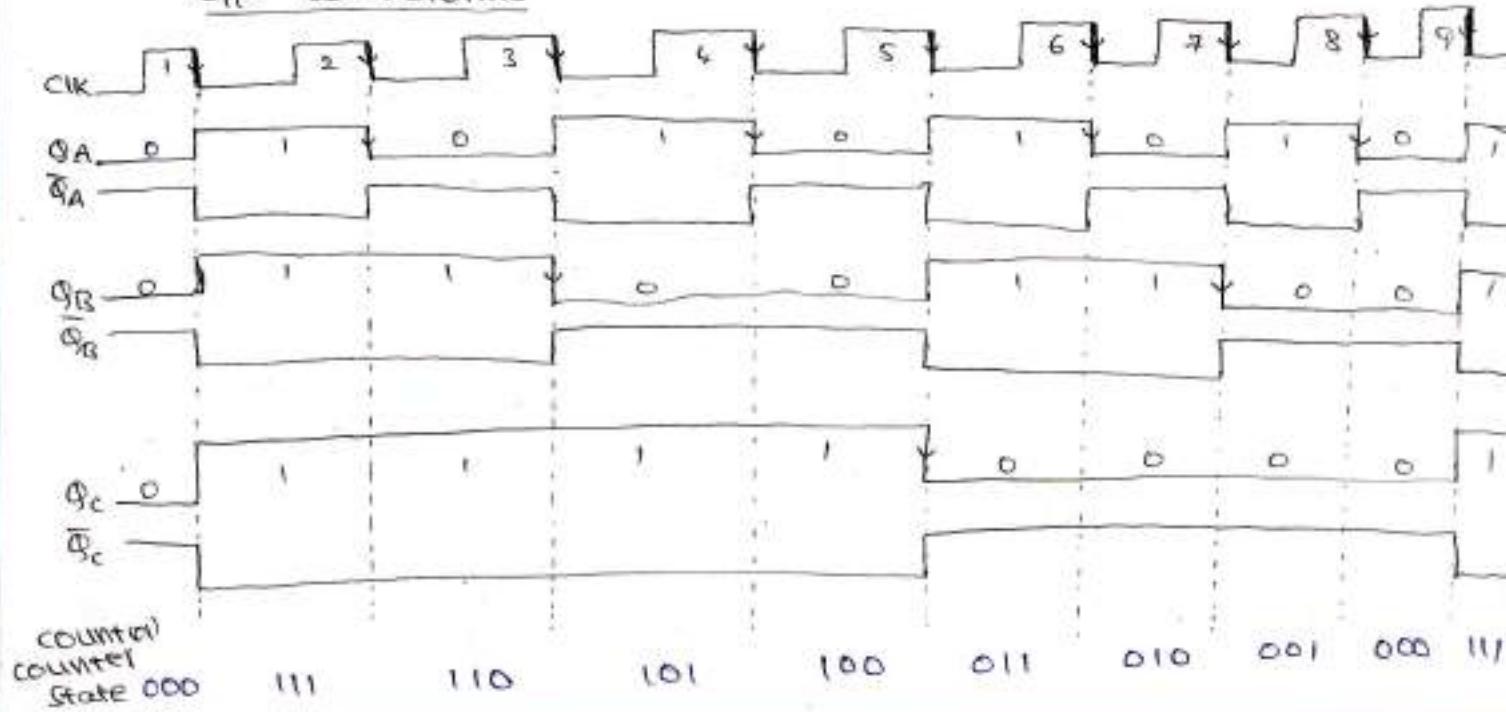


Fig: 3-bit ripple down counter using -ve edge triggered T-FF's.

counter o/p  $\rightarrow Q_c \ Q_b \ Q_a$   
MSB      LSB

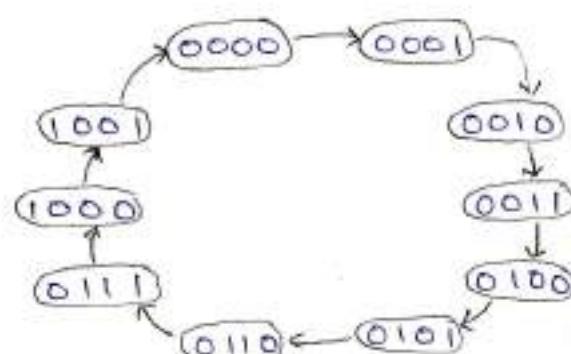
O/P waveforms



### 3. Asynchronous Mod-10 counter (or) Decade counter (or)

BCD counter :-

state diagram :-



for NAND gate  
draw bubbles  
for its i/p

for AND gate  
Don't draw bubbles

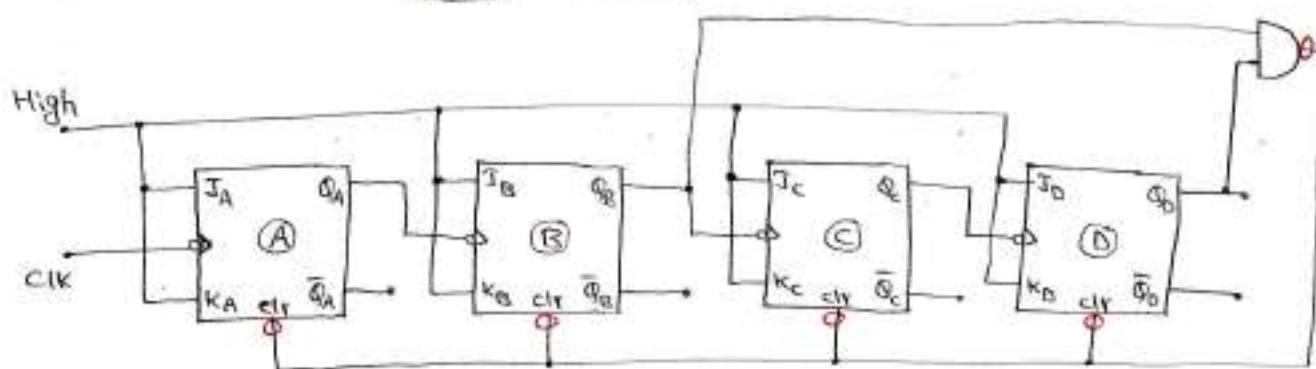


Fig: MOD-10 ripple counter

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

\* When counter o/p is 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

CLK	$Q_D$	$Q_C$	$Q_B$	$Q_A$
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:-

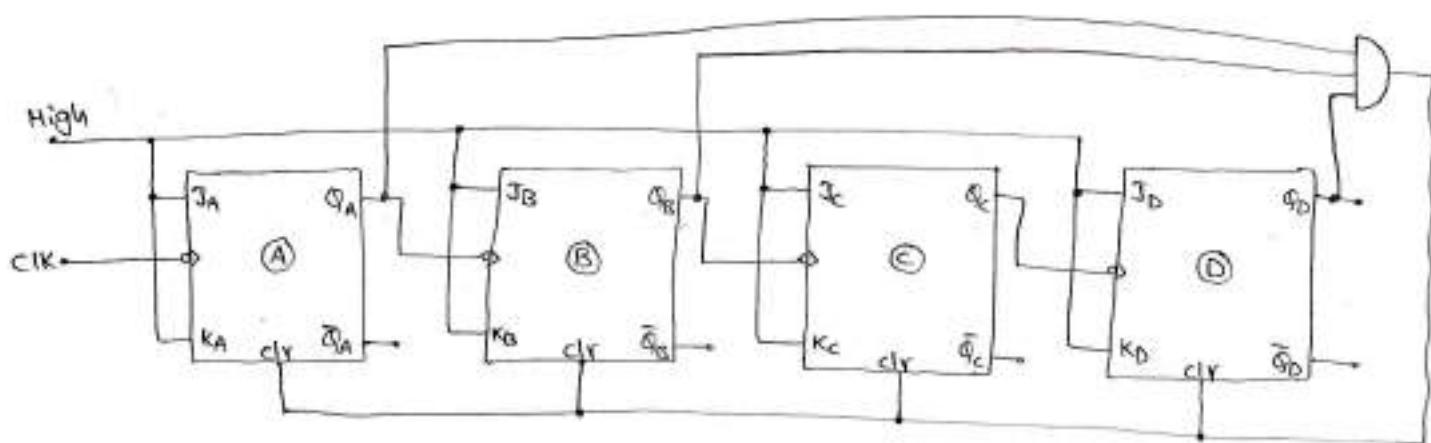
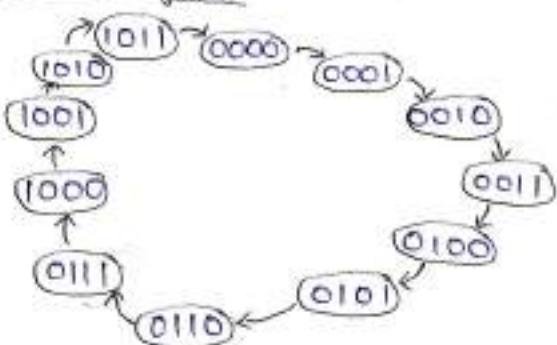


Fig: MOD-12 ripple counter

counter o/p  $\rightarrow Q_D \bar{Q}_C \bar{Q}_B \bar{Q}_A$   
MSB            LSB

\* When counter o/p is 1011, o/p of And gate becomes 1. Since o/p of And gate is connected to reset (clr) 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 :-

8

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

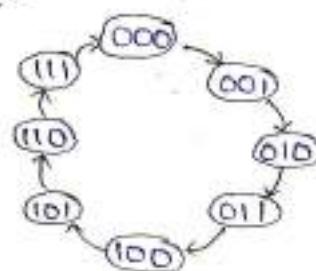
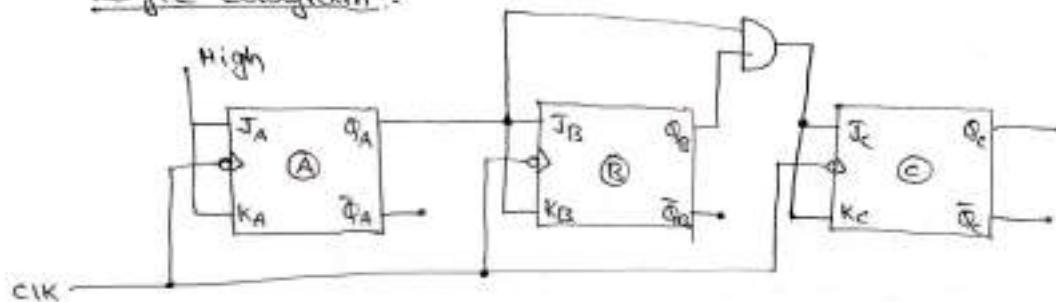
state diagram:-Logic diagram:-

Fig.: MOD-8 Sync. counter

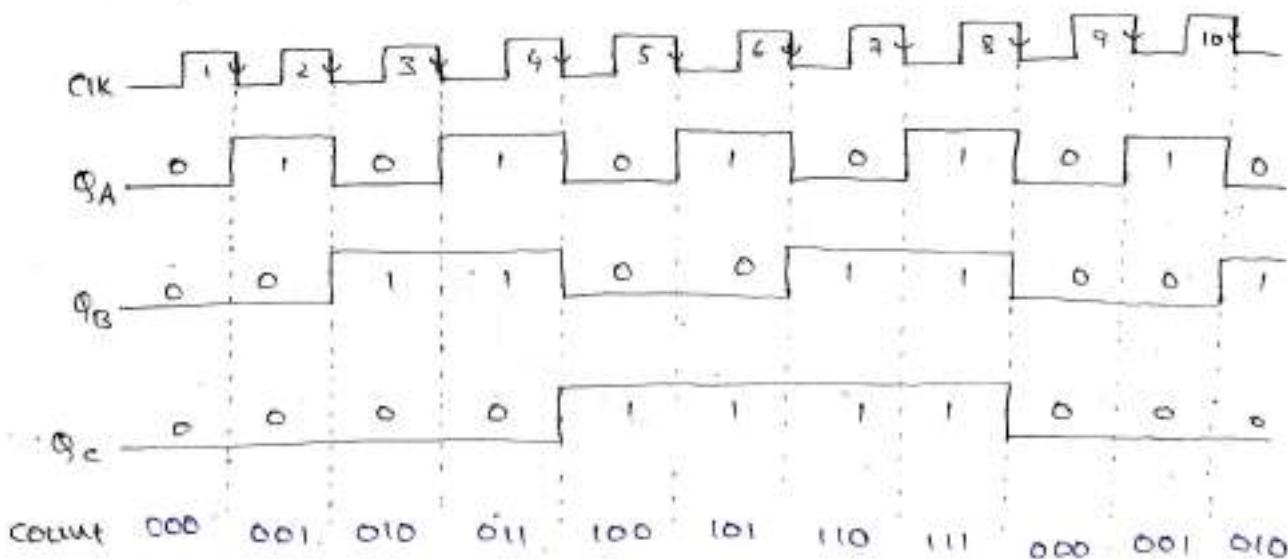
FF P/P's

$$J_A = K_A = 1$$

$$J_B = K_B = Q_A$$

$$J_C = K_C = Q_A \cdot Q_B$$

counter o/p  $\rightarrow Q_C \ Q_B \ Q_A$   
MSB            LSB

output waveforms:-

2. 4-bit synchronous binary up counter (or) MOD-16 counter:-

Block diagram:-

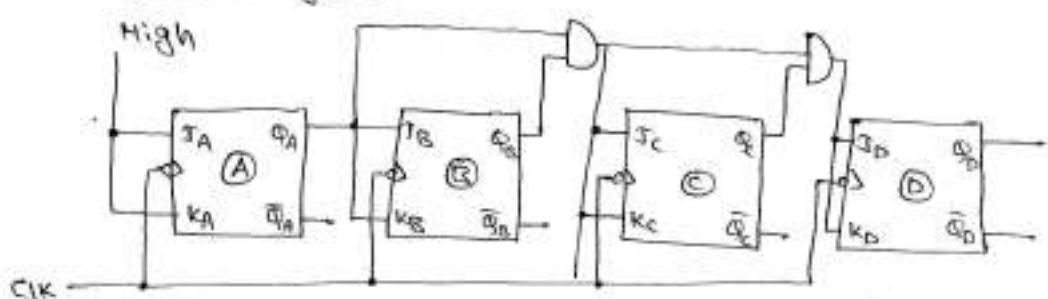


Fig: MOD-16 Binary up counter

FF i/p's

$$J_A = K_A = 1$$

$$J_B = K_B = Q_A$$

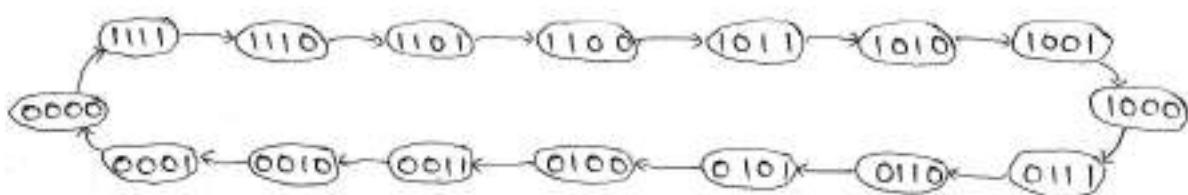
$$J_C = K_C = Q_A \cdot Q_B$$

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

Counter o/p  $\rightarrow Q_D \ Q_C \ Q_B \ Q_A$   
 MSB                    LSB

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

State diagram:-



Logic diagram:-

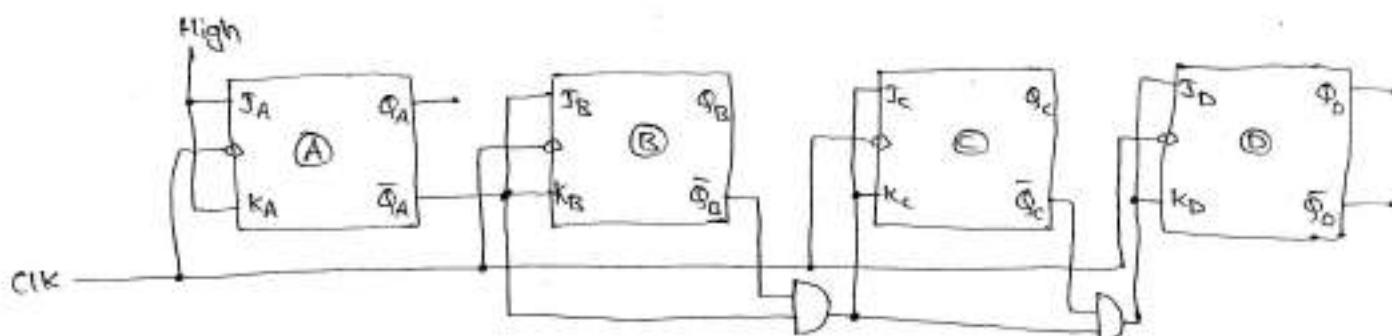


Fig: MOD-16 Down counter

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  $\rightarrow Q_D \ Q_C \ Q_B \ Q_A$   
MSB                    LSB

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	0	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 Synchronous counters :-

Q. Design synchronous 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.

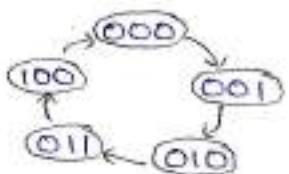
The no. of FF's required to design MOD-5 synchronous up counter can be determine by the equation,  $2^n \geq N$  where  $n \rightarrow$  no. of FF's  
 $N \rightarrow$  MOD no.

The possible value of 'n' which satisfies 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 ilp'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

∴ SR 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

<u>for <math>J_C</math></u>		
$Q_C$	$Q_B Q_A$	
0	00 01 11 10	
1	X X   X X	

$$\therefore J_C = Q_B Q_A$$

<u>for <math>J_B</math></u>		
$Q_C$	$Q_B Q_A$	
0	00 01 11 10	
1	X   X X X	

$$\therefore J_B = Q_A$$

<u>for <math>J_A</math></u>		
$Q_C$	$Q_B Q_A$	
0	00 01 11 10	
1	X   X X X	

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

<u>for <math>K_C</math></u>		
$Q_C$	$Q_B Q_A$	
0	00 01 11 10	
1	X X   X X	

$$\therefore K_C = 1$$

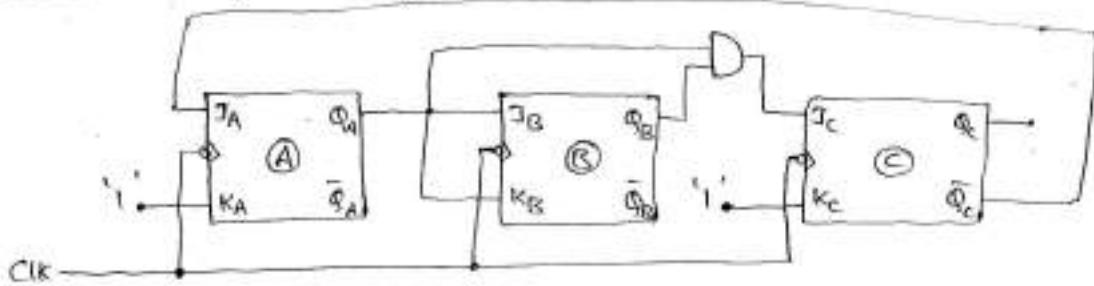
<u>for <math>K_B</math></u>		
$Q_C$	$Q_B Q_A$	
0	00 01 11 10	
1	X   X X X	

$$\therefore K_B = Q_A$$

<u>for <math>K_A</math></u>		
$Q_C$	$Q_B Q_A$	
0	00 01 11 10	
1	X   X X X	

$$\therefore K_A = 1$$

### Step 5 :- Logic diagram



10

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.

$$2^n \geq N \quad \text{where} \quad N \rightarrow \text{Mod No.}$$

$$n \rightarrow \text{no. of FF's.}$$

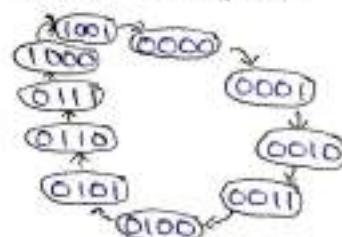
$$2^n \geq 10$$

$$n = 4$$

$$\therefore \text{No. of FF's required} = 4$$

FF TYPE  $\rightarrow$  T-FF

Step 2: State diagram



Step 3: Excitation table

Present stage Q <sub>B</sub> Q <sub>C</sub> Q <sub>D</sub> Q <sub>A</sub>	Next stage Q <sub>B+1</sub> Q <sub>C+1</sub> Q <sub>D+1</sub> Q <sub>A+1</sub>	FF i/p's			
		T <sub>D</sub>	T <sub>C</sub>	T <sub>B</sub>	T <sub>A</sub>
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 <sub>n</sub>	Q <sub>n+1</sub>	T
0	0	0
0	1	1
1	0	1
1	1	0

Step 4: K-Map simplification

for  $T_D$

		Q_B Q_A	Q_B Q_A	Q_B Q_A	Q_B Q_A
		00	01	11	10
		00			
		01		1	
		11	X	X	X
		10	1	X	X

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

for  $T_C$

		Q_B Q_A	Q_B Q_A	Q_B Q_A	Q_B Q_A
		00	01	11	10
		00			
		01		1	
		11	X	X	X
		10		X	X

$$\therefore T_C = Q_B Q_A$$

for  $T_B$

		Q_B Q_A	Q_B Q_A	Q_B Q_A	Q_B Q_A
		00	01	11	10
		00			
		01	1	1	
		11	X	X	X
		10		X	X

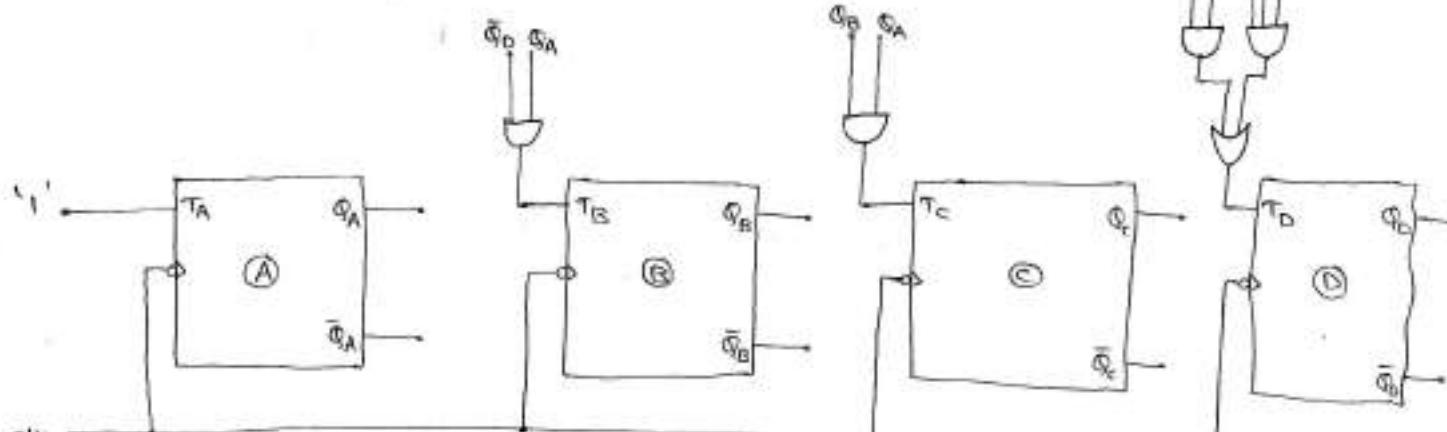
$$\therefore T_B = \bar{Q}_D Q_A$$

for  $T_A$

		Q_B Q_A	Q_B Q_A	Q_B Q_A	Q_B Q_A
		00	01	11	10
		00			
		01	1	1	1
		11	X	X	X
		10	1	X	X

$$\therefore T_A = 1$$

Step 5: logic diagram



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

## II

### Counter with unused states :-

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

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

$$2^n \geq N \quad \text{where } n \rightarrow \text{no. of FF's}$$

$$N \rightarrow \text{Mod no.}$$

$$\therefore 2^n \geq 5$$

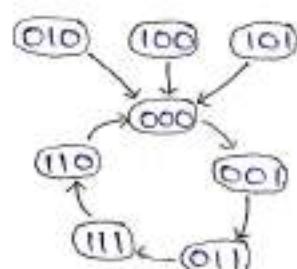
$$\therefore n=3$$

i. no. of FF's required  $\rightarrow 3$   
 FF's using  $\rightarrow$  T FF's

Step 2: State diagram

Used states  $\rightarrow 0, 1, 3, 7, 6$

Unused states  $\rightarrow 2, 4, 5$



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

Step 3: Excitation table

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

$\therefore$  T-FF excitation table

$Q_m$	$Q_{m+1}$	T
0	0	0
0	1	1
1	0	1
1	1	0

Step 4: K-Map simplification

for  $T_2$

		Q <sub>2</sub> Q <sub>1</sub>	Q <sub>2</sub> Q <sub>0</sub>	Q <sub>1</sub> Q <sub>0</sub>	Q <sub>2</sub> Q <sub>1</sub> Q <sub>0</sub>
		00	01	11	10
Q <sub>0</sub>	0	1	1	1	1
	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 <sub>2</sub> Q <sub>1</sub>	Q <sub>2</sub> Q <sub>0</sub>	Q <sub>1</sub> Q <sub>0</sub>	Q <sub>2</sub> Q <sub>1</sub> Q <sub>0</sub>
		00	01	11	10
Q <sub>0</sub>	0	1	1	1	1
	1	1	1	1	1

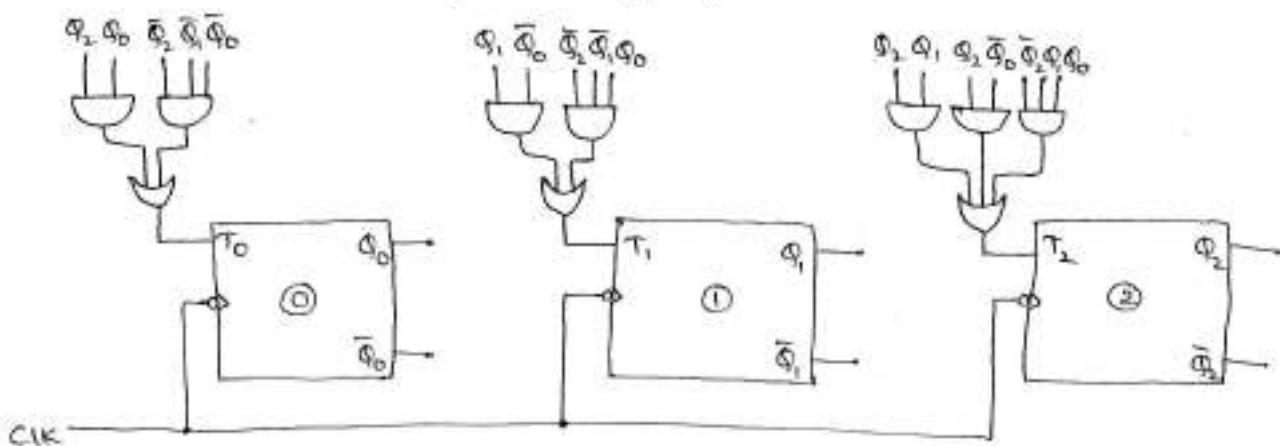
$$\therefore T_1 = Q_1 \bar{Q}_0 + \bar{Q}_2 \bar{Q}_1 Q_0$$

for  $T_0$

		Q <sub>2</sub> Q <sub>1</sub>	Q <sub>2</sub> Q <sub>0</sub>	Q <sub>1</sub> Q <sub>0</sub>	Q <sub>2</sub> Q <sub>1</sub> Q <sub>0</sub>
		00	01	11	10
Q <sub>0</sub>	0	1	1	1	1
	1	1	1	1	1

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

Step 5: logic diagram

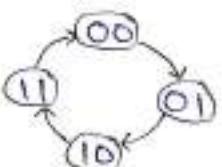


counter o/p  $\rightarrow Q_2 \ Q_1 \ Q_0$   
MSB            LSB

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, ...  
 $\therefore$  It is MOD-4 counter.

State diagram:-



Q. Design synchronous counter using JK FF. to count 12  
the following sequence.

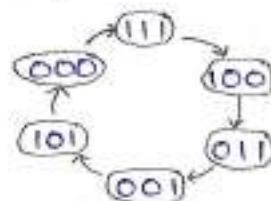
7, 4, 3, 1, 5, 0, 7, ...

Sol.: Total no. of states  $\rightarrow$  6

$\therefore$  It is MOD-6 counter.

Used States  $\rightarrow$  7, 4, 3, 1, 5, 0.

State diagram:-



Unused states  $\rightarrow$  2, 6.

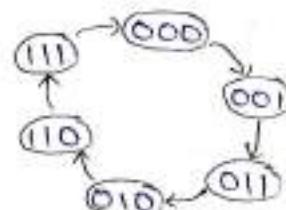
Treat unused states as don't cares.

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.

$\therefore$  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/down (M) signal.

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

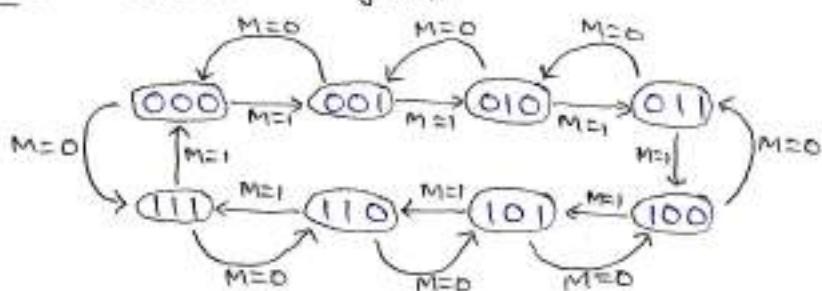
→ When  $M=0$ , the counter goes through down sequence ( $n, n-1, \dots, 2, 1, 0$ ).

Step 1 :

No. of FF's required  $\rightarrow 3$  ( $\because$  it is 3-bit counter)

FF's using  $\rightarrow T$  FF's.

Step 2 : State diagram



Step 3: Excitation table

control i/p UP/down (M)	present state $Q_C$ $Q_B$ $Q_A$	Next Stage			FF i/p's		
		$Q_{C1}$	$Q_{B1}$	$Q_{A1}$	$T_C$	$T_B$	$T_A$
Down	0 0 0	1	1	1	1	1	1
	0 0 1	0	0	0	0	0	1
	0 1 0	0	0	1	0	1	1
	0 1 1	0	1	0	0	0	1
	1 0 0	0	1	1	1	1	1
	1 0 1	1	0	0	0	0	1
	1 1 0	1	0	1	0	1	1
	1 1 1	1	1	0	0	0	1
UP	0 0 0	0	0	1	0	0	1
	0 0 1	0	1	0	0	1	1
	0 1 0	0	1	1	0	0	1
	0 1 1	1	0	0	1	1	1
	1 0 0	1	0	1	0	0	1
	1 0 1	1	1	0	0	1	1
	1 1 0	1	1	1	1	0	0
	1 1 1	0	0	0	0	1	1

$\therefore T$ -FF  
excitable  
table

$Q_{in}$	$Q_{out}$	T
0 0	0	0
0 1	1	
1 0	1	
1 1	0	

Step 4: K-map simplification

		$M\bar{Q}_C\bar{Q}_A$			
		00	01	11	10
<u>for <math>T_C</math></u>	00	1			
	01	1			
	11			1	
	10				1

$$\therefore T_C = \bar{M}\bar{Q}_C\bar{Q}_A + M\bar{Q}_B\bar{Q}_A \\ = M\odot Q_B \ominus Q_A$$

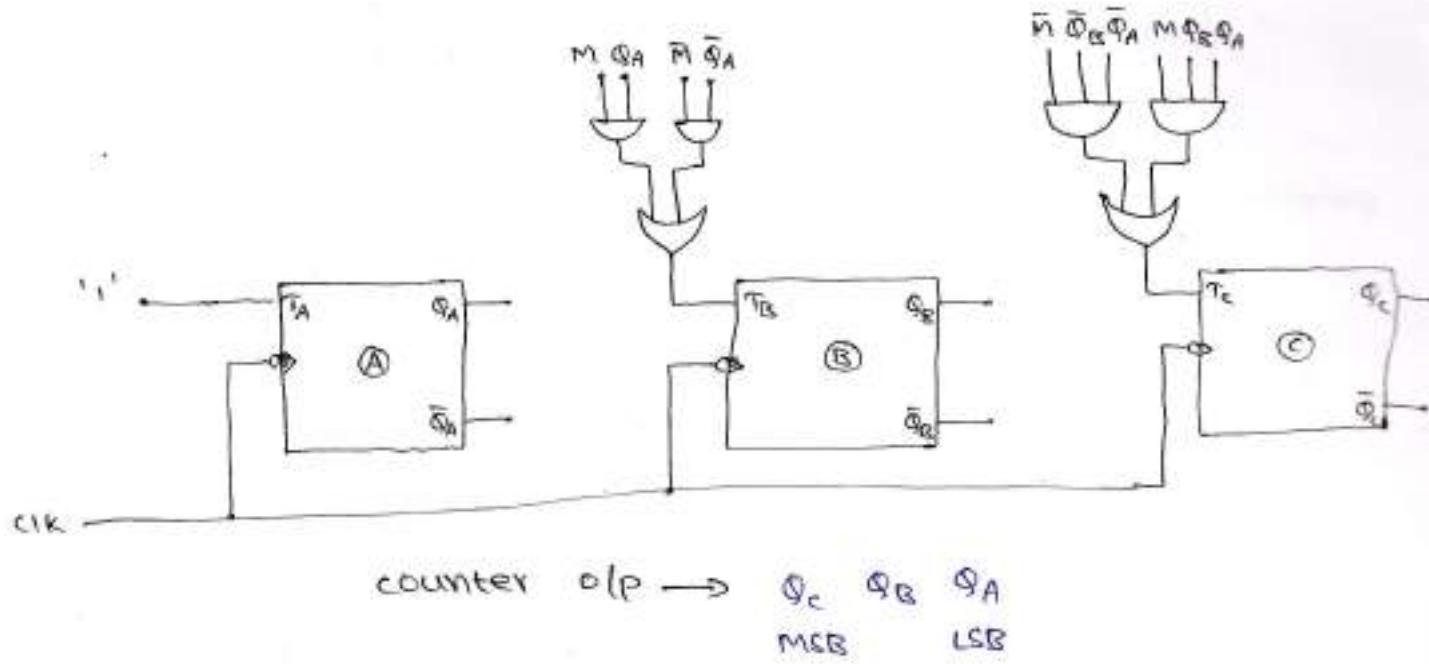
		$M\bar{Q}_C\bar{Q}_A$			
		00	01	11	10
<u>for <math>T_B</math></u>	00	1			1
	01	1			1
	11		1	1	
	10		1	1	

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

		$M\bar{Q}_C\bar{Q}_A$			
		00	01	11	10
<u>for <math>T_A</math></u>	00	1	1	1	1
	01	1	1	1	1
	11	1	1	1	1
	10	1	1	1	1

$$\therefore T_A = 1$$

logic diagram :-



16/10/2014

\* Frequency dividers :-

2-bit Asynchronous counter (or) MOD-4 counter (or)

Divide-by-4 counter :-

logic diagram :-

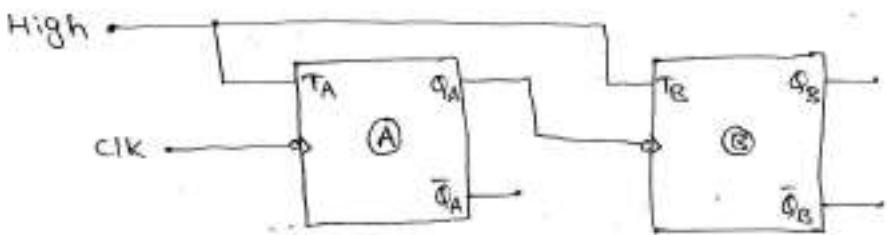
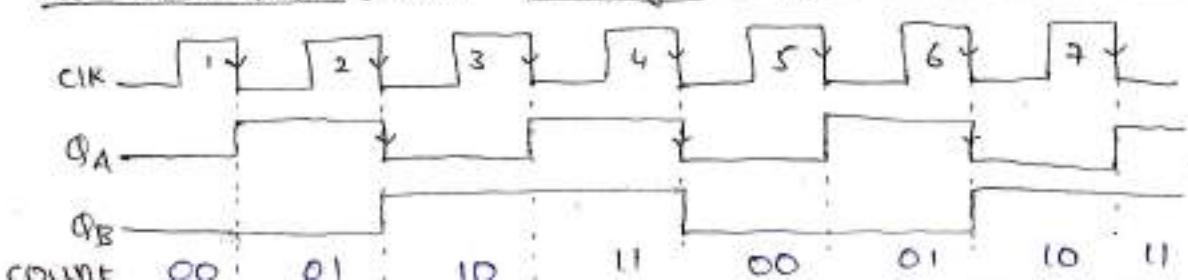


fig:- 2-bit ripple counter.

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  $Q_B$  is one fourth of the clk frequency.

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

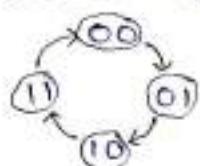
Note:- Each FF acts as a divide by 2 ( $\div 2$ ) frequency divider. (i.e) Each FF divides the incoming clk signal frequency by 2.

3-bit counter $\rightarrow$	MOD-8 counter $\rightarrow$	Divide-by-8 counter
4-bit counter $\rightarrow$	MOD-16 counter $\rightarrow$	Divide-by-16 counter
MOD-5 counter $\rightarrow$ Divide-by-5 counter		
MOD-6 counter $\rightarrow$ 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 "busless circuit"

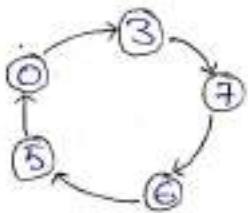


Fig: a) Desired sequence

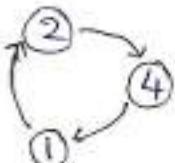


Fig: b) Unused states forming lockout.

To avoid lockout condition, the unused states are introduced in front of the used states.

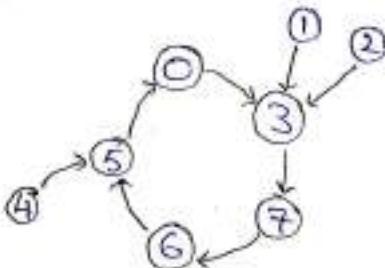
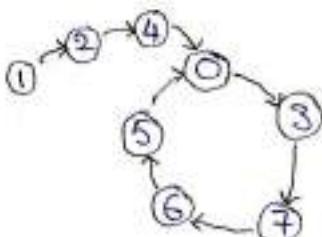
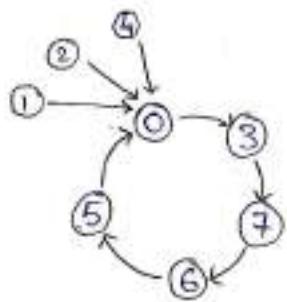
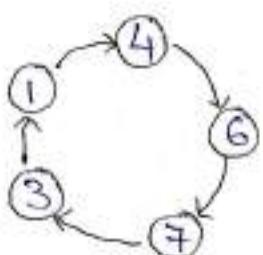


Fig: state diagrams for removing lockout.

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

Sol: state diagram:-



Used states  $\rightarrow 1, 3, 4, 6, 7$

Unused states  $\rightarrow 0, 2, 5$

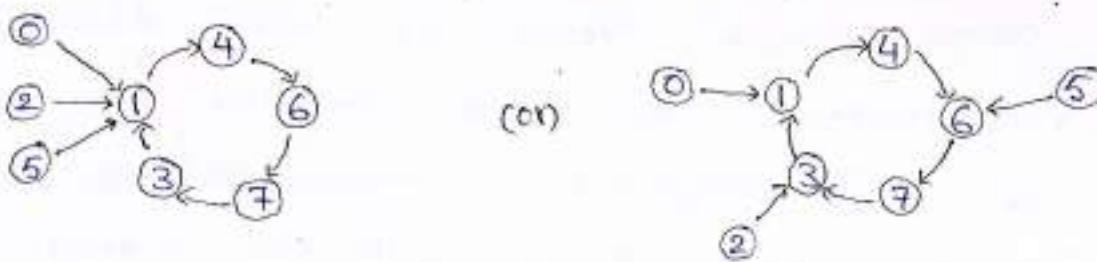


Fig.: state diagrams for Avoiding lockout condition.

### \* Shift Register counters :-

A shift Register counter is basically a shift register with 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.

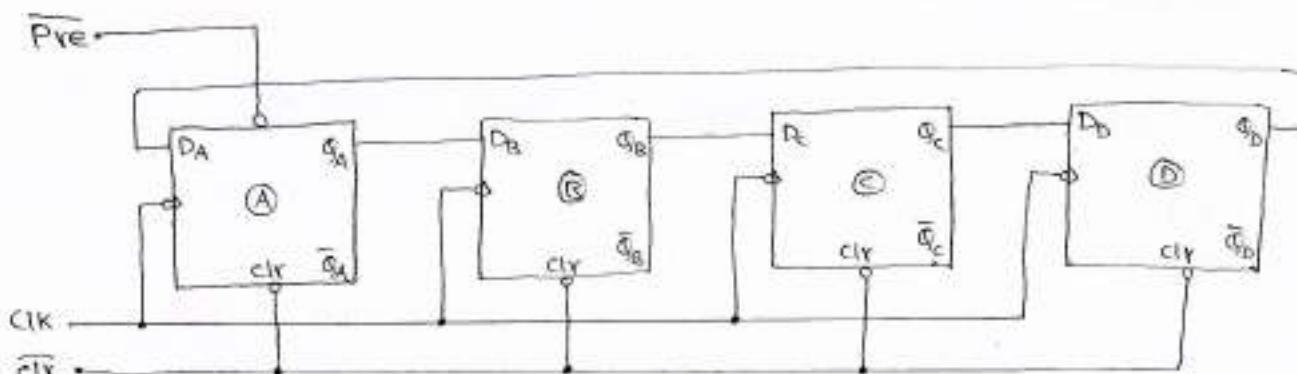
Two of the most common types of shift Register counters are

- (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  $\rightarrow Q_D \ Q_C \ Q_B \ Q_A$   
MSB LSB

Fig.: 4-bit ring counter.

clr  $\rightarrow$  clear (or)  
Reset i/p  
Pre  $\rightarrow$  Set (or)  
Pre  $\rightarrow$  i/p

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 fed back to the D<sub>0</sub> of first stage. The CLR & PRE inputs 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 <sub>A</sub>	Q <sub>B</sub>	Q <sub>C</sub>	Q <sub>D</sub>
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.

Solt:-

21/10/2014

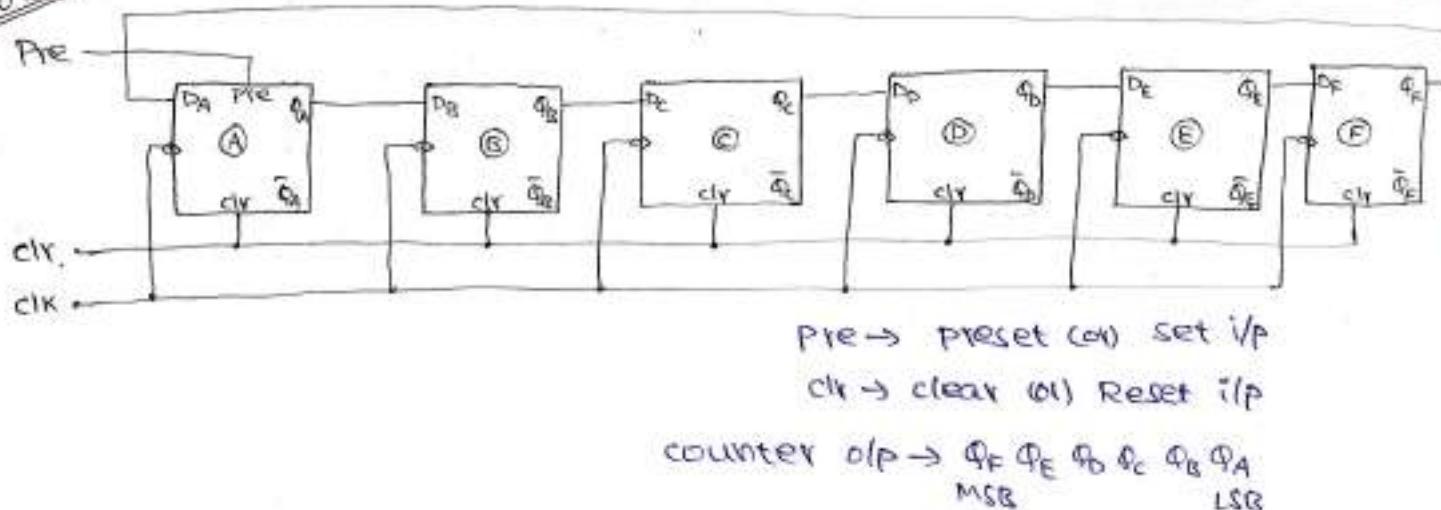
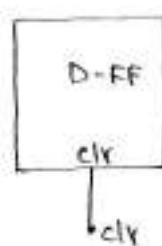


Fig: MOD-6 Ring counter

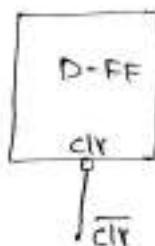
Truth table:

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

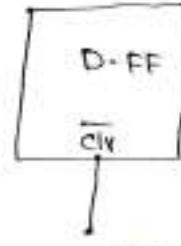
\*



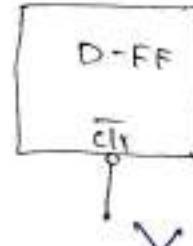
If  $\overline{clr} = 1$ ,  
then FF clears.



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



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



X

\* 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  $2^n$ , 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.

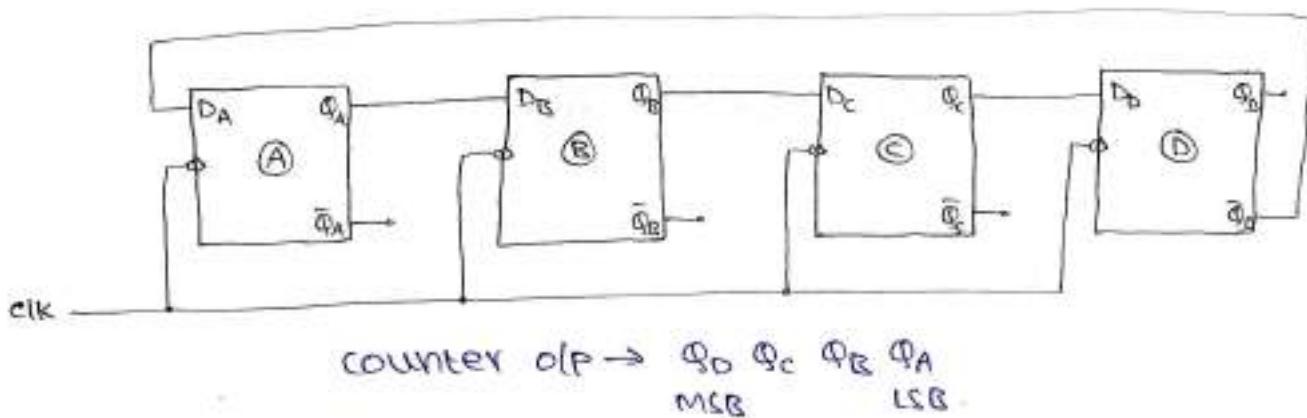


Fig:- 4-bit Johnson ring counter.

Truth table on State Sequence :-

CLK	Q <sub>A</sub>	Q <sub>B</sub>	Q <sub>C</sub>	Q <sub>D</sub>
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

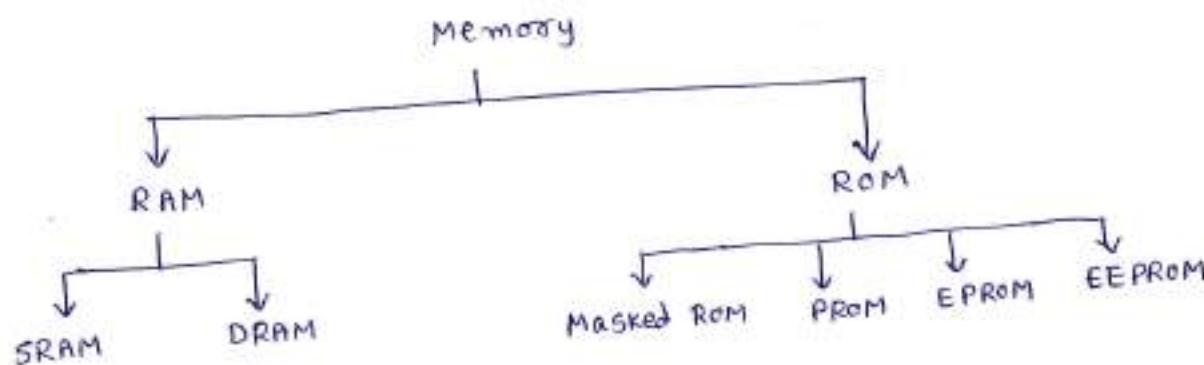
## UNIT - 5

### 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

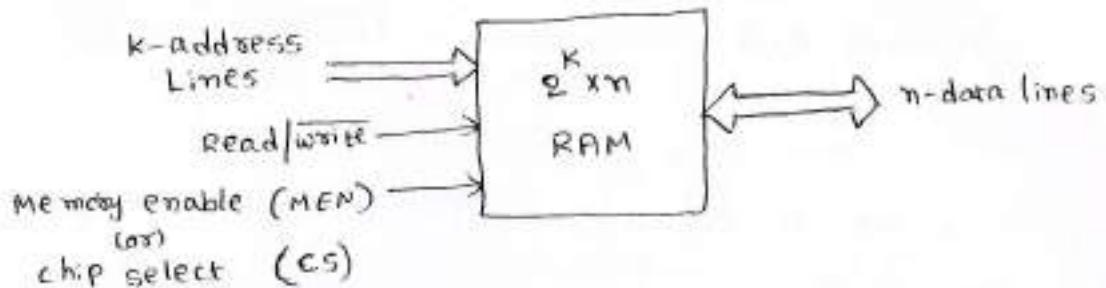


fig: B-D of  $2^K \times n$  RAM

- \* In the above fig, size of RAM is  $2^K \times n$ , it means RAM consists of  $2^K$  memory locations and each memory location size is  $n$ -bits.
- \* The communication between a memory and other devices can be achieved through data lines. Each data line carries one bit of binary information.
- \* Data lines are bidirectional, but at any time they act as either i/p or o/p lines. During memory write operation, data lines act as i/p lines.
- \* Address lines carries desired memory location address for memory read (or) write operation.
- \* Read/write is control signal. It is used to select either read or write operation. If Read/write = 0, the RAM performs write operation only.
- \* RAM enables and performs either Read (or) write operation only when its CS (or) MEN i/p is high. If CS=0, RAM disables.

### Static RAM (SRAM)

SRAM consists of flip-flops to store the binary information.

### Dynamic RAM (DRAM)

- \* DRAM consists of CMOS transistors and capacitors.
- \* DRAM stores the binary information in the form of electric charges on capacitors.

## Read operation

The process of transferring the stored information out of memory is referred to as a memory read operation.

### Memory Read operation steps

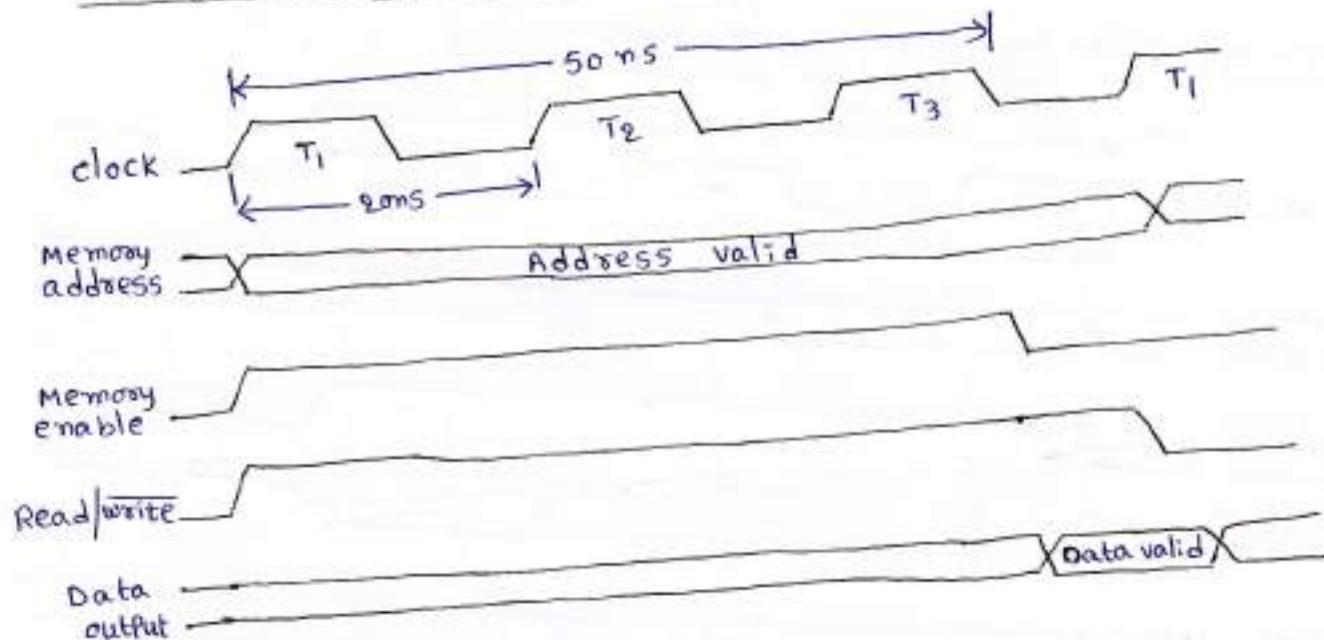
Step 1 : Apply the binary address of the desired word to the address lines.

Step 2 : Activate the read input.

The memory unit will then take the bits from the word that has been selected by the address and apply them to the output data lines.

The content of the selected word does not change after reading.

### Memory Read cycle Timing diagram



$$\text{clock time period} = 20\text{ns}$$

$$\text{Access time} = 50\text{ns}$$

### Access time

It is a memory device operating speed. It is defined as the time required to perform the read operation. (or) write

## Write operation

The process of storing new information into memory is referred to as a memory write operation.

## Memory write operation steps

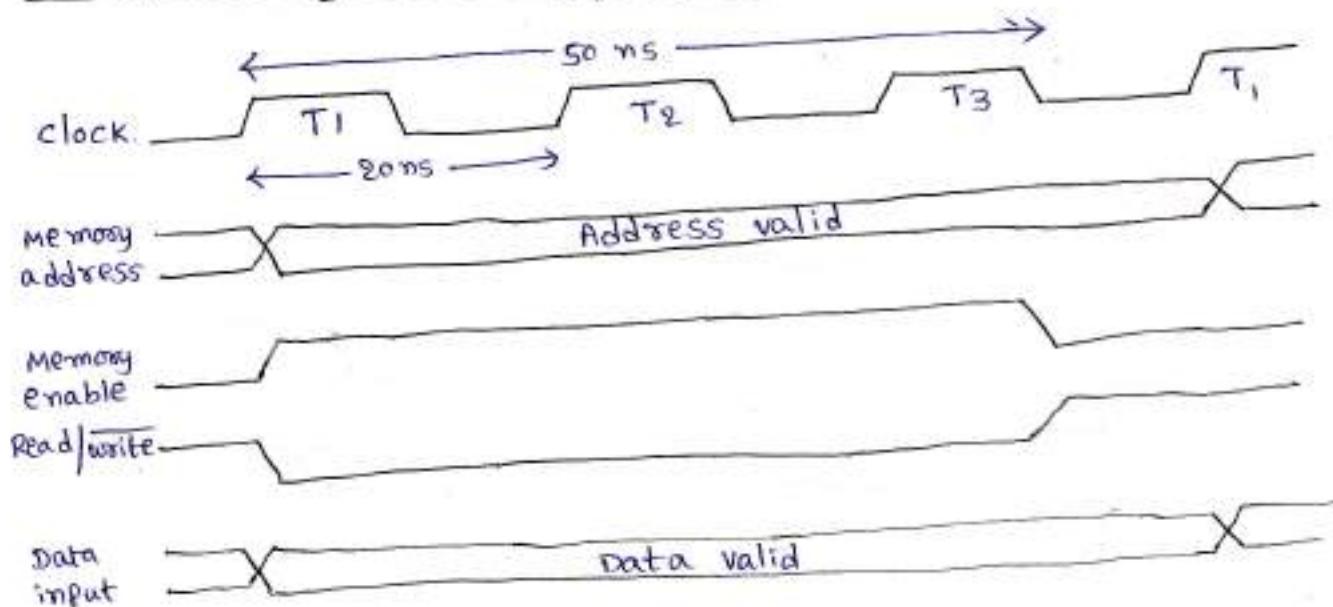
Step 1: Apply the binary address of the desired word to the address lines.

Step 2: Apply the data bits that must be stored in memory to the data input lines.

Step 3: Activate the write input.

The memory unit will then take the bits from the input data lines and store them in the word specified by the address lines.

## Memory write cycle timing diagram



## Read only Memory (ROM)

\* ROM performs only Read operation

\* ROM is non-volatile memory i.e. stored information will not get erased even after power is turned off. That's why we store permanent programs (e.g. operating system programs) in ROM. ∴ ROM is called as Program memory.

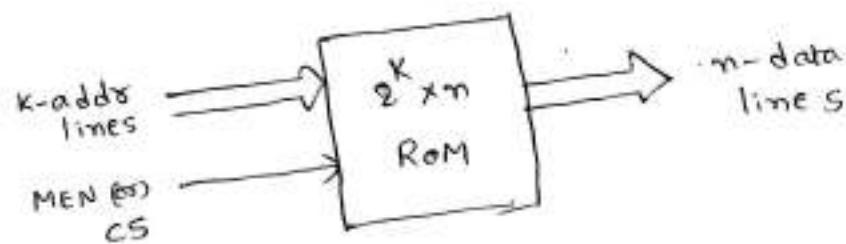


fig: B.T of  $2^K \times n$  ROM

- \* for ROM data lines are always o/p lines because it performs only Read operation
- \* CS (or) MEN i/p is used to select ROM IC. if CS=0, then n-data lines will be in high impedance state.

### Types of ROM

Masked ROM (or) Simple ROM

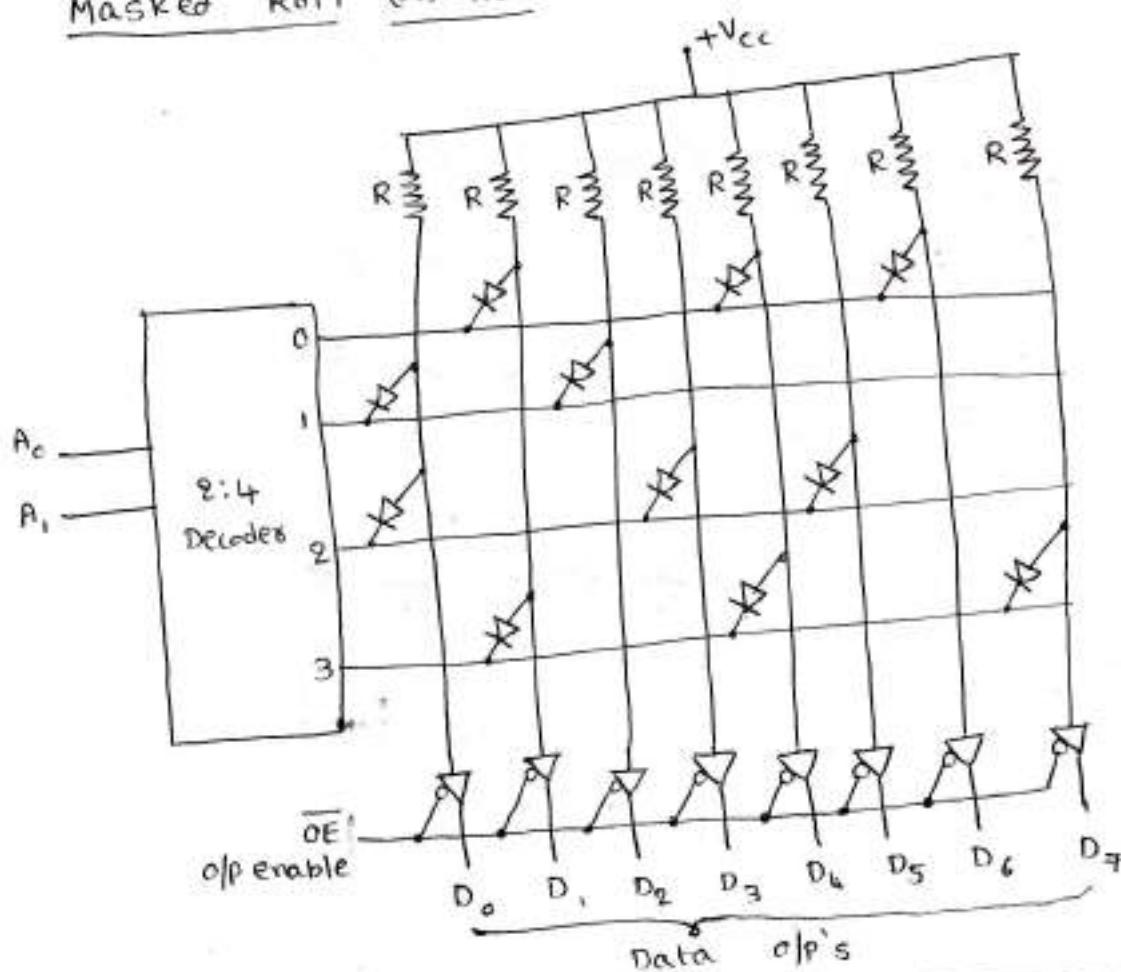


fig: simple  
4 byte  
diode ROM

Address in Binary	Binary Data							
	D <sub>0</sub>	D <sub>1</sub>	D <sub>2</sub>	D <sub>3</sub>	D <sub>4</sub>	D <sub>5</sub>	D <sub>6</sub>	D <sub>7</sub>
00	1	0	1	1	0	1	0	1
01	0	1	0	1	1	1	1	1
10	0	1	1	0	1	0	1	1
11	1	0	1	1	0	1	1	0

fig: contents  
of ROM

Diode ROM consists of only diodes and a decoder. as shown in the above fig. address lines  $A_0$  and  $A_1$  are decoded by 2:4 decoder and used to select one of the 4 rows. Data is available on the o/p data lines only when output enable ( $\overline{OE}$ ) signal is Low.

Masked ROMs are used in microprocessor based toys, TV games, home computers.

### PROM (Programmable Read only Memory or field PROM)

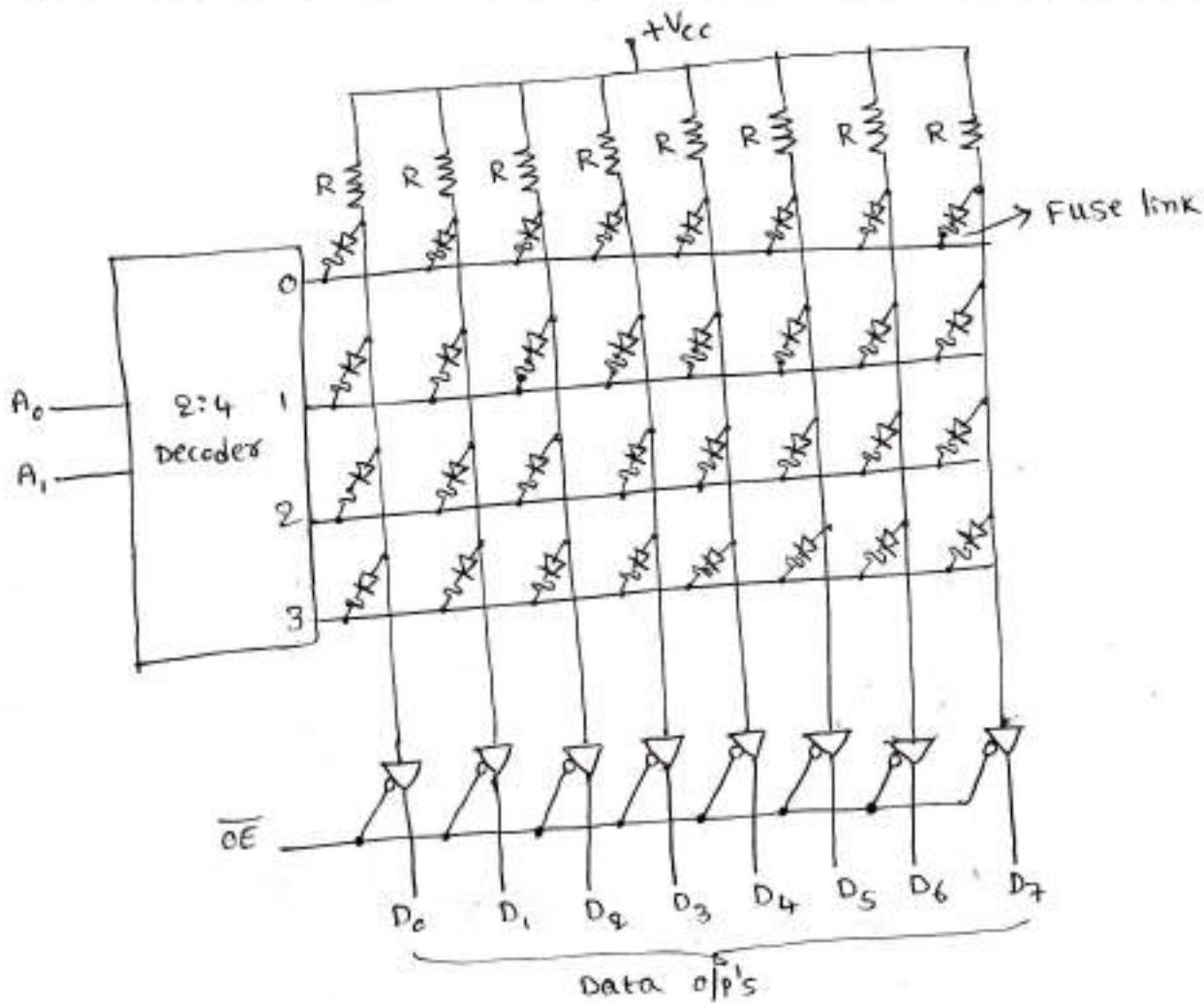


fig: 4-byte PROM

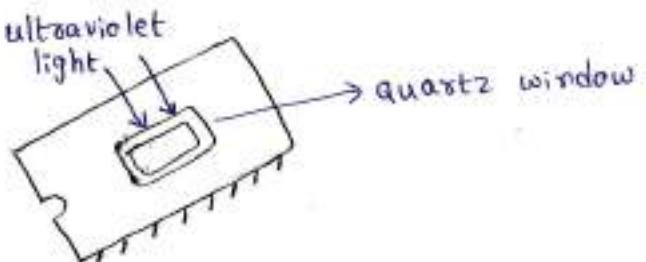
The PROM chip comes 'blank' and can be programmed only once by a user

above fig shows 4-byte PROM. It has diodes in every bit position; therefore, the o/p is initially all 0's. Each diode, however has a fusible link in series with it. By addressing bit and

applying proper current pulse at the corresponding address, we can blow out the fuse, storing logic 1 at that bit position.

The PROMs are one time programmable. Once programmed, the information stored is permanent.

### EPROM (Erasable programmable Read only Memory)



\* EPROMs use MOS circuitry. They store 1's and 0's as a packet of charge in a buried layer of the IC chip.

\* EPROM chips can be erased and rewritten a no. of times.

\* Stored data in the EPROM can be erased by exposing the chip to ultraviolet light through its quartz window for 15 to 20 minutes. (before erasing the data we have to remove the IC from its socket)

\* It is not possible to erase selective information, when erased the entire information is lost. The chip can be reprogrammed.

\* EPROM is ideally suitable for product development, experimental projects and college laboratories, since this chip can be reused many times.

\* In EPROM, it is possible to program any location at any time - either individually, sequentially, or at random.

### EEPROM (Electrically Erasable programmable ROM)

\* EEPROM is like the EPROM except that the previously programmed connections can be erased with an electrical signal instead of ultraviolet light

\* The advantage of EEPROM over EPROM is that the device can be erased without removing it from its socket.

30/10/2014

- \* Logic devices :- Logic devices can be broadly categorized as:
  - (i) Fixed logic devices
  - (ii) Programmable logic devices

### 1. Fixed logic devices :- (FLD's)

As the name indicates, the circuits in the FLD are permanent, they perform one function or set of functions. Once manufactured, they cannot be erased.

### 2. Programmable logic devices (PLD's) :-

\* PLD is an IC that contains large no. of gates, FF's and registers that are inter connected on the chip.

\* PLDs can be reconfigured to perform any no. of functions at any time.

\* Types of PLD's are (i) simple PLD's / sequential PLD's.

(a) PROM

(b) PLA

(c) PAL

(ii) complex PLD's (CPLD's)

(iii) Field programmable gate Arrays (FPGA).

### (i) PROM ( Programmable Read only Memory) :-

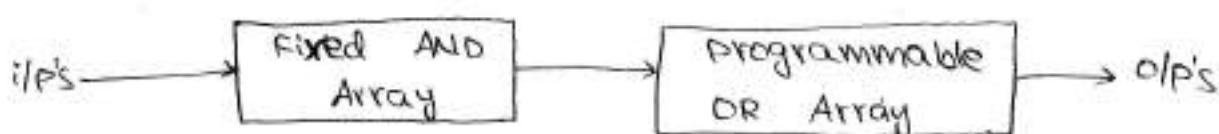
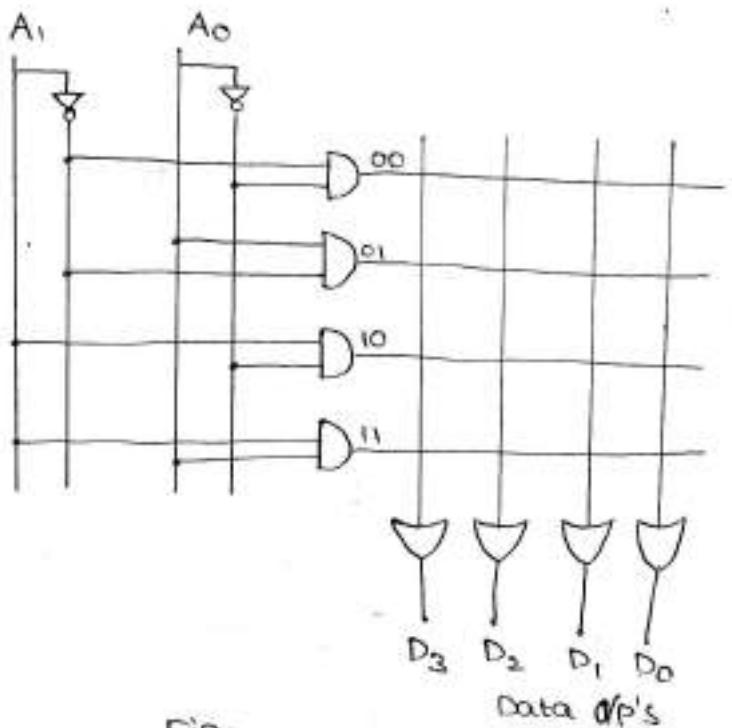


Fig: Basic configuration of PROM



PROM has  
fixed AND Array &  
Programmable OR Array

Fig: Internal logic of 4x4 ROM implementation of 4x4 ROM  
(a) AND-OR

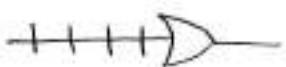


Fig: Array logic symbol  
for OR gate

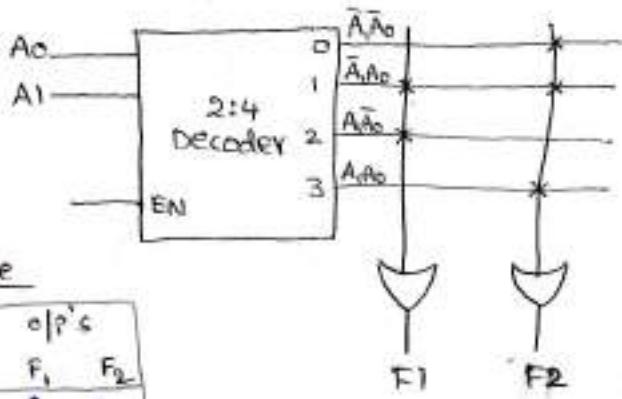


Fig: conventional logic symbol  
for OR gate.

Q. :- Implement the following boolean functions  
using PROM.

$$F_1(A_1, A_0) = \sum m(1, 2) ; F_2(A_1, A_0) = \sum m(0, 1, 3)$$

Sol: Logic diagram variables are  $A_1, A_0$  (2 variables)  
 $\therefore$  USE 2:4 Decoder.



$$\begin{aligned} F_1(A_1, A_0) &= \sum m(1, 2) \\ &= \bar{A}_1 A_0 + A_1 \bar{A}_0 \\ F_2(A_1, A_0) &= \sum m(0, 1, 3) \\ &= \bar{A}_1 \bar{A}_0 + \bar{A}_1 A_0 + A_1 A_0 \end{aligned}$$

fig: Implementation of F<sub>1</sub>, F<sub>2</sub> using PROM

Q.:- Implement full adder using PROM.

Sol: Full adder :-

Block diagram :



Truth table :

i/p's			o/p's	
A	B	C	sum	carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

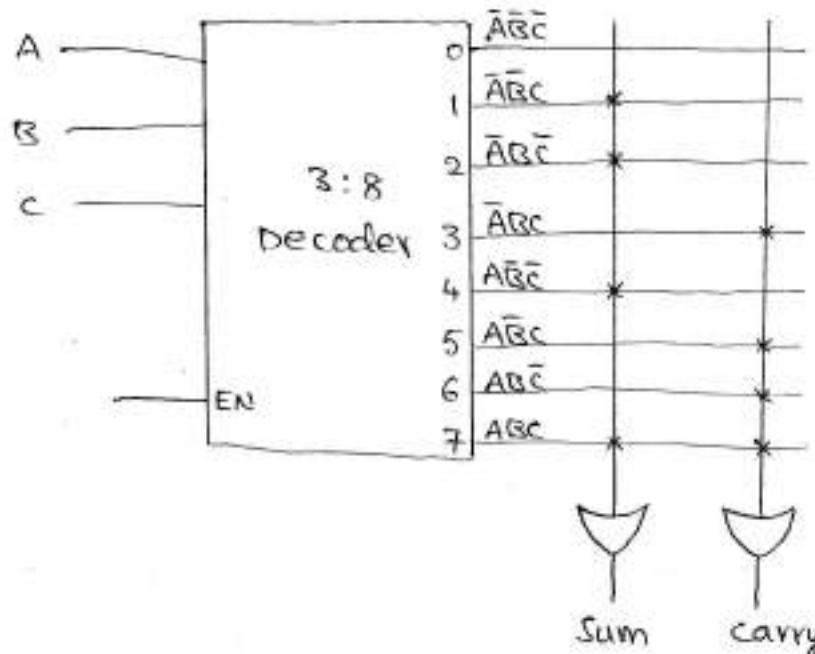
$$\therefore \text{Sum} = \Sigma m(1, 2, 4, 7)$$

$$\therefore \text{carry} = \Sigma m(3, 5, 6, 7)$$

Logic Diagram

i/p variables = A, B, C (3 variables)

$\therefore$  use 3:8 decoder.



$$\therefore \text{Sum} = \Sigma m(1, 2, 4, 7)$$

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

$$\therefore \text{carry} = \Sigma m(3, 5, 6, 7)$$

$$= \bar{A}BC + A\bar{B}C + \\ AB\bar{C} + ABC$$

fig: Full adder using PROM

Q.:- Design a combinational circuit using ROM. The circuit accepts 3-bit binary no. and generates its equivalent EXCESS-3 code.

Sol: EXCESS-3 :- truth table :-

I/P (3 bit binary no.)	O/P (EXCESS-3 code of I/P)
$B_2 \ B_1 \ B_0$	$E_3 \ E_2 \ E_1 \ E_0$
0 0 0	0 0 1 1
0 0 1	0 1 0 0
0 1 0	0 1 0 1
0 1 1	0 1 1 0
1 0 0	0 1 1 1
1 0 1	1 0 0 0
1 1 0	1 0 0 1
1 1 1	1 0 1 0

$$\therefore E_0(B_2, B_1, B_0) = \sum m(0, 2, 4, 6)$$

$$E_1(B_2, B_1, B_0) = \sum m(0, 3, 4, 7)$$

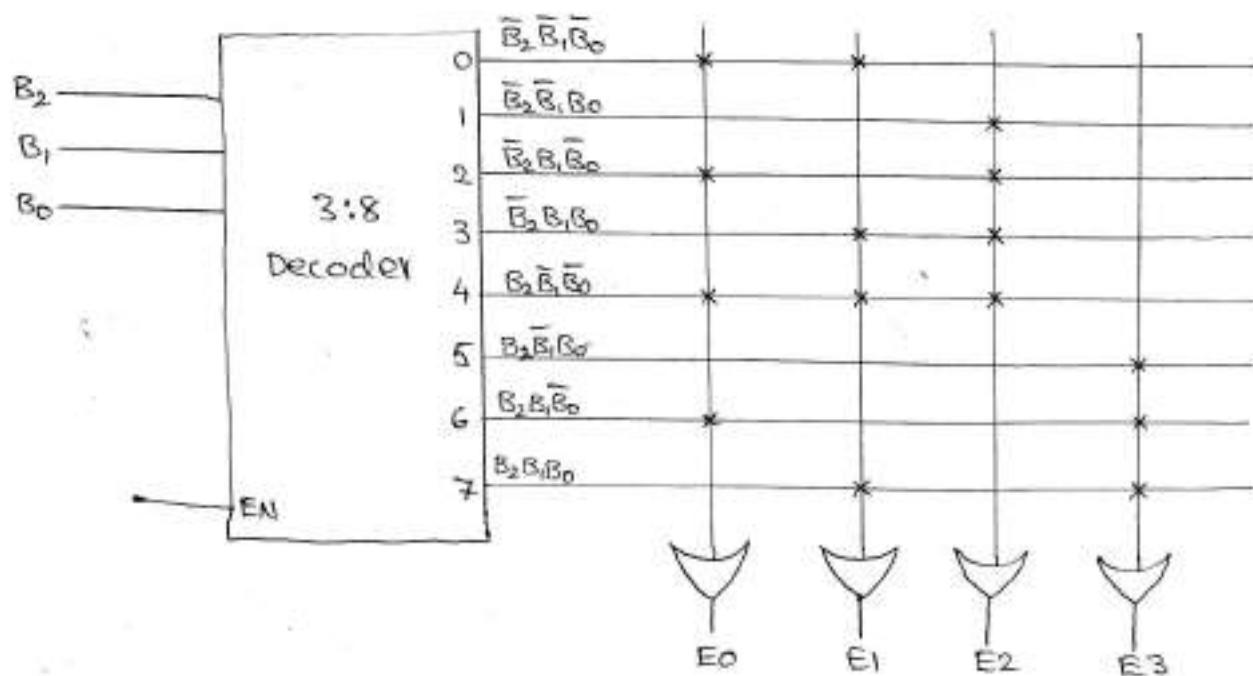
$$E_2(B_2, B_1, B_0) = \sum m(1, 2, 3, 4)$$

$$E_3(B_2, B_1, B_0) = \sum m(5, 6, 7)$$

Logic diagram

i. I/P Variables =  $B_2, B_1, B_0$  (3 variables)

ii. use 3:8 decoder.



Q. :- Design a combinational circuit using ROM. The circuit accepts a 3-bit number and generates an o/p binary no. equal to the square of the i/p no.

Sol: Truth table

i/p (3-bit binary no.)			o/p (square of i/p binary no.)						decimal no.
$B_2$	$B_1$	$B_0$	$S_5$	$S_4$	$S_3$	$S_2$	$S_1$	$S_0$	
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1	1
0	1	0	0	0	1	0	0	0	4
0	1	1	0	0	1	0	0	1	9
1	0	0	0	1	0	0	0	0	16
1	0	1	0	1	1	0	0	1	25
1	1	0	1	0	0	1	0	0	36
1	1	1	1	1	0	0	0	1	49

$$S_0(B_2, B_1, B_0) = \sum m(1, 3, 5, 7)$$

$$S_1(B_2, B_1, B_0) = \sum m(0)$$

$$S_2(B_2, B_1, B_0) = \sum m(2, 6)$$

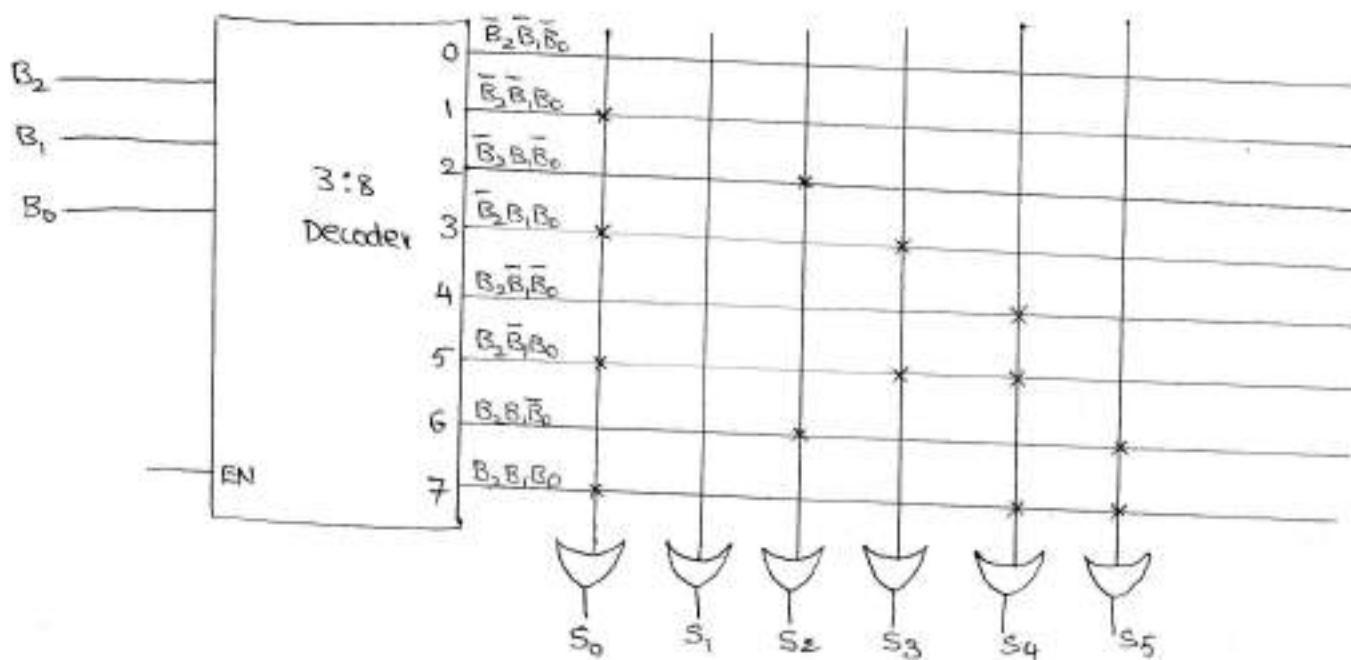
$$S_3(B_2, B_1, B_0) = \sum m(3, 5)$$

$$S_4(B_2, B_1, B_0) = \sum m(4, 5, 6)$$

$$S_5(B_2, B_1, B_0) = \sum m(6, 7)$$

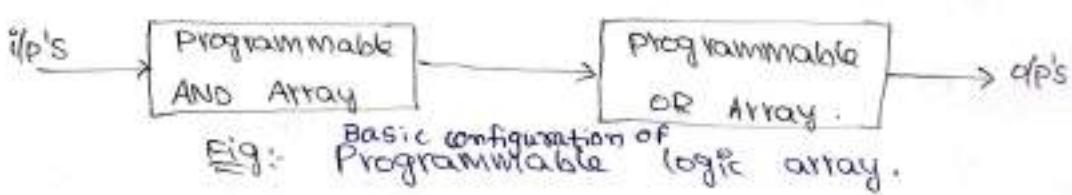
Logic Diagram: I/P variables =  $B_2, B_1, B_0$  (3 variables).

∴ Use 3:8 decoder.



05/11/2014

## Programmable logic Array (PLA) :-



Q.:- A combinational circuit is defined by the function  
 $F_1 = \sum m(3,5,7)$ ,  $F_2 = \sum m(4,5,7)$ . Implement the circuit  
 with a PLA having 3 I/P's, 3 product terms and  
 two O/P's.

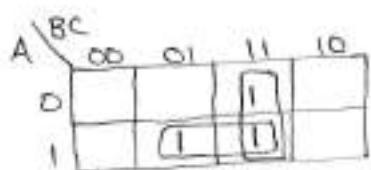
Sol:- Given,  $F_1 = \sum m(3,5,7)$   
 $F_2 = \sum m(4,5,7)$

Truth table for boolean functions  $F_1$  &  $F_2$ ,

I/P's			O/P's	
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	1	0
1	0	0	0	1
1	0	1	1	1
1	1	0	0	0
1	1	1	1	1

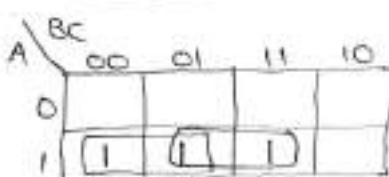
### K-Map simplification :-

for  $F_1$



$$\therefore F_1 = AC + BC$$

for  $F_2$



$$\therefore F_2 = AB + AC$$

PLA Program table :-

Product term	I/P's			O/P's		no. of I/P's $\rightarrow 3$	no. of product terms $\rightarrow 3$
	A	B	C	F <sub>1</sub>	F <sub>2</sub>		
1	A	C	-	1	1		
2	B	C	-	1	-		
3	A	$\bar{B}$	-	1	0		

no. of O/P's  $\rightarrow 2$

Logic diagram:-

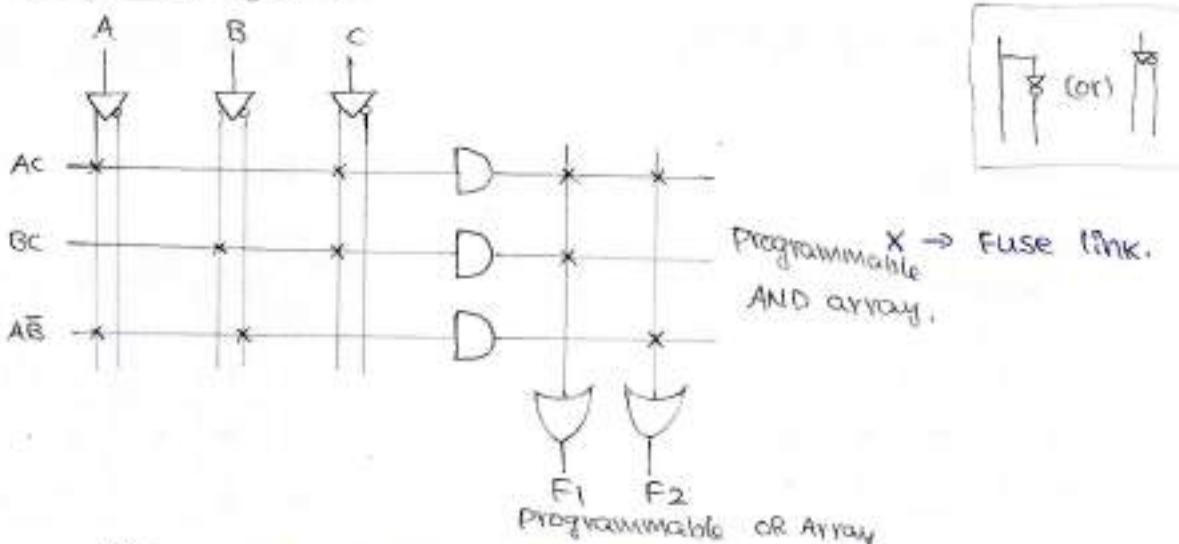


Fig:- Implementation of F<sub>1</sub>, F<sub>2</sub> using PLA

Q:- Design a BCD to EXCESS-3 code converter and implement using suitable PLA.

Sol:- Truth table for BCD to EXCESS-3 code converter:-

B <sub>3</sub>	B <sub>2</sub>	B <sub>1</sub>	B <sub>0</sub>	I/P's		O/P's			
				E <sub>3</sub>	E <sub>2</sub>	E <sub>1</sub>	E <sub>0</sub>		
0	0	0	0	0	0	1	1		
1	0	0	0	1	0	0	0		
2	0	0	1	0	0	1	0		
3	0	0	1	1	0	1	0		
4	0	1	0	0	0	1	1		
5	0	1	0	1	0	0	0		
6	0	1	1	0	1	0	1		
7	0	1	1	1	0	1	0		
8	1	0	0	0	1	0	1		
9	1	0	0	1	1	1	0		

( $\because$  BCD digits are 0-9, so consider 10-15 as don't care)

$$E_3(B_3, B_2, B_1, B_0) = \Sigma m(5, 6, 7, 8, 9)$$

$$E_2(B_3, B_2, B_1, B_0) = \Sigma m(1, 2, 3, 4, 9)$$

$$E_1(B_3, B_2, B_1, B_0) = \Sigma m(0, 3, 4, 7, 8)$$

$$E_0(B_3, B_2, B_1, B_0) = \Sigma m(0, 2, 4, 6, 8)$$

K-map simplification :-

for  $E_3$

$B_3B_2$ \ $B_1B_0$	00	01	11	10
00				
01	1	1	1	
11	X	X	X	X
10	1	1	X	X

for  $E_2$

$B_3B_2$ \ $B_1B_0$	00	01	11	10
00				
01	1			
11	X		X	X
10		1	X	X

$$\therefore E_3 = B_3 + B_2B_0 + B_2B_1$$

$$\therefore E_2 = B_2\bar{B}_1\bar{B}_0 + \bar{B}_2B_0 + \bar{B}_2B_1$$

for  $E_1$

$B_3B_2$ \ $B_1B_0$	00	01	11	10
00	1		1	
01	1		1	
11	X	X	X	X
10	1		X	X

$$\therefore E_1 = \bar{B}_1\bar{B}_0 + B_1B_0$$

for  $E_0$

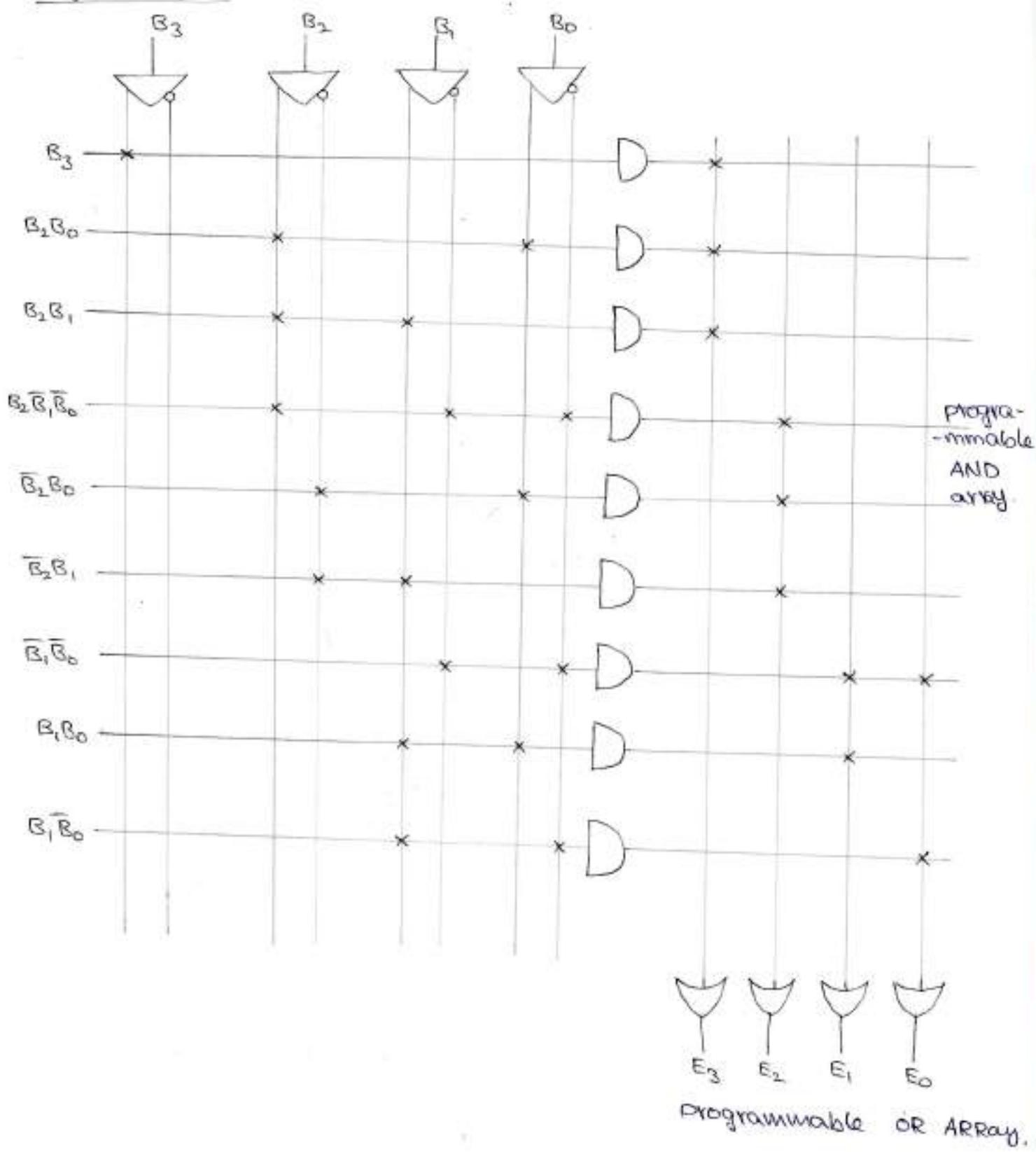
$B_3B_2$ \ $B_1B_0$	00	01	11	10
00	1			1
01	1			1
11	X	X	X	X
10	1		X	X

$$\therefore E_0 = \bar{B}_1\bar{B}_0 + B_1\bar{B}_0$$

PLA program table :-

Product terms	i(p's)				o(p's)			
	$B_3$	$B_2$	$B_1$	$B_0$	$E_3$	$E_2$	$E_1$	$E_0$
$B_3$	1	-	-	-	1	-	-	-
$B_2B_0$	-	1	-	1	1	-	-	-
$B_2B_1$	-	1	1	-	1	-	-	-
$B_2\bar{B}_1\bar{B}_0$	-	1	0	0	-	1	-	-
$\bar{B}_2B_0$	-	0	-	1	-	1	-	-
$\bar{B}_2B_1$	-	0	1	-	-	1	-	-
$\bar{B}_1\bar{B}_0$	-	-	0	0	-	-	1	1
$B_1B_0$	-	-	1	1	-	-	1	-
$\bar{B}_1\bar{B}_0$	-	-	1	0	-	-	-	1

logic diagram :-



\* Programmable Array Logic (PAL) :-



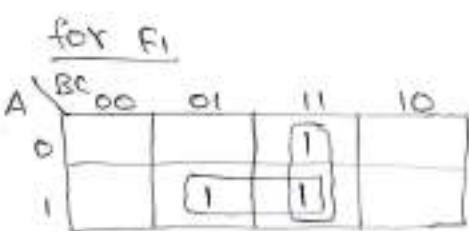
Fig:- Basic configuration of PAL.

- Q.:- Implement the following boolean expressions using PAL.
- $F_1 = \Sigma m(3, 5, 7)$ ,  $F_2 = \Sigma m(4, 5, 7)$ .
- Sol:- Given  $F_1 = \Sigma m(3, 5, 7)$
- $F_2 = \Sigma m(4, 5, 7)$

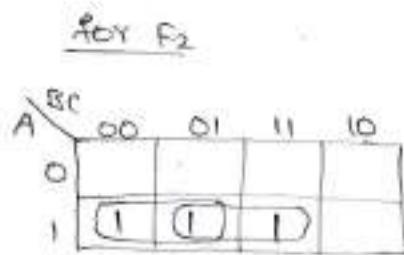
Truth table for boolean functions  $F_1$  &  $F_2$ .

i/p's			o/p's	
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	1	0
1	0	0	0	1
1	0	1	1	1
1	1	0	0	0
1	1	1	1	1

K-map simplification:-



$$\therefore F_1 = AC + BC$$

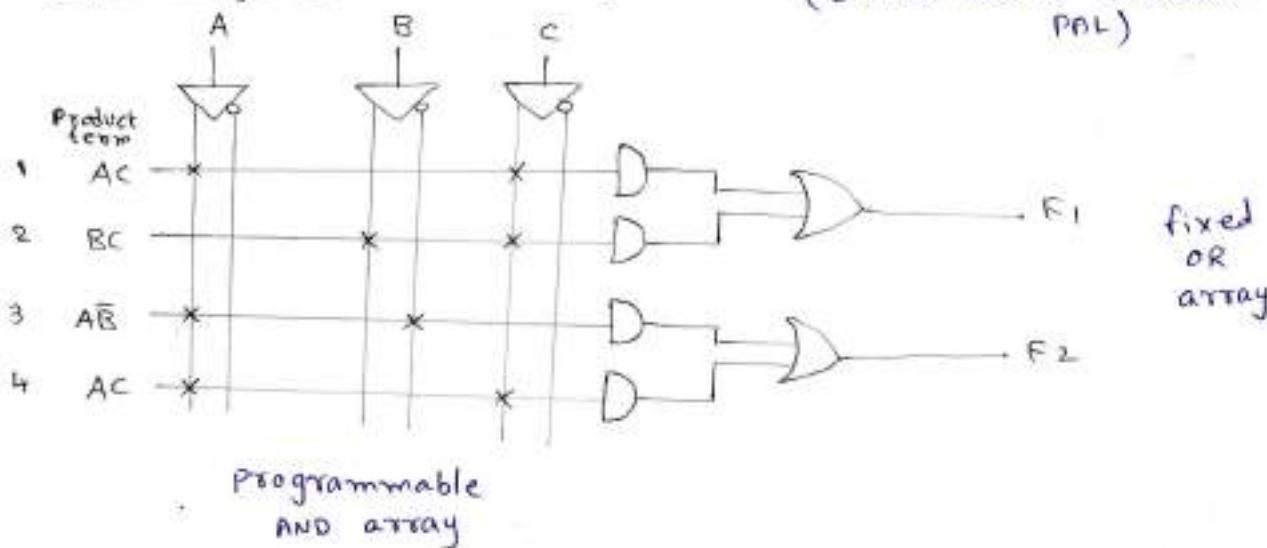


$$\therefore F_2 = A\bar{B} + AC$$

PAL Program table :-

Product term	AND i/p's			o/p's	
	A	B	C		
1	AC	1	-	1	
2	BC	-	1	1	$F_1 = AC + BC$
3	$A\bar{B}$	1	0	-	
4	AC	1	-	1	$F_2 = AC + A\bar{B}$

Logic diagram:-



Q1 :- Implement the following Boolean expressions using PAL

$$A(x,y,z) = \sum m(1,2,4,6)$$

$$B(x,y,z) = \sum m(0,1,6,7)$$

$$C(x,y,z) = \sum m(2,6)$$

$$D(x,y,z) = \sum m(1,2,3,5,7)$$

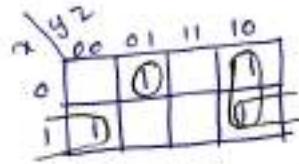
Solu

Truth table for Boolean functions A, B, C, D

i/p's			o/p's			
x	y	z	A	B	C	D
0	0	0	0	1	0	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	0	0	1
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	1	1	1	0
1	1	1	0	1	0	1

K-map simplification

for A



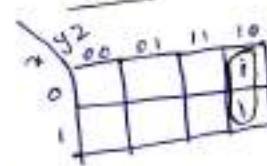
$$A = xz + yz + xy$$

for B



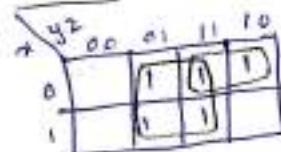
$$B = xz + xy$$

for C



$$C = yz$$

for D

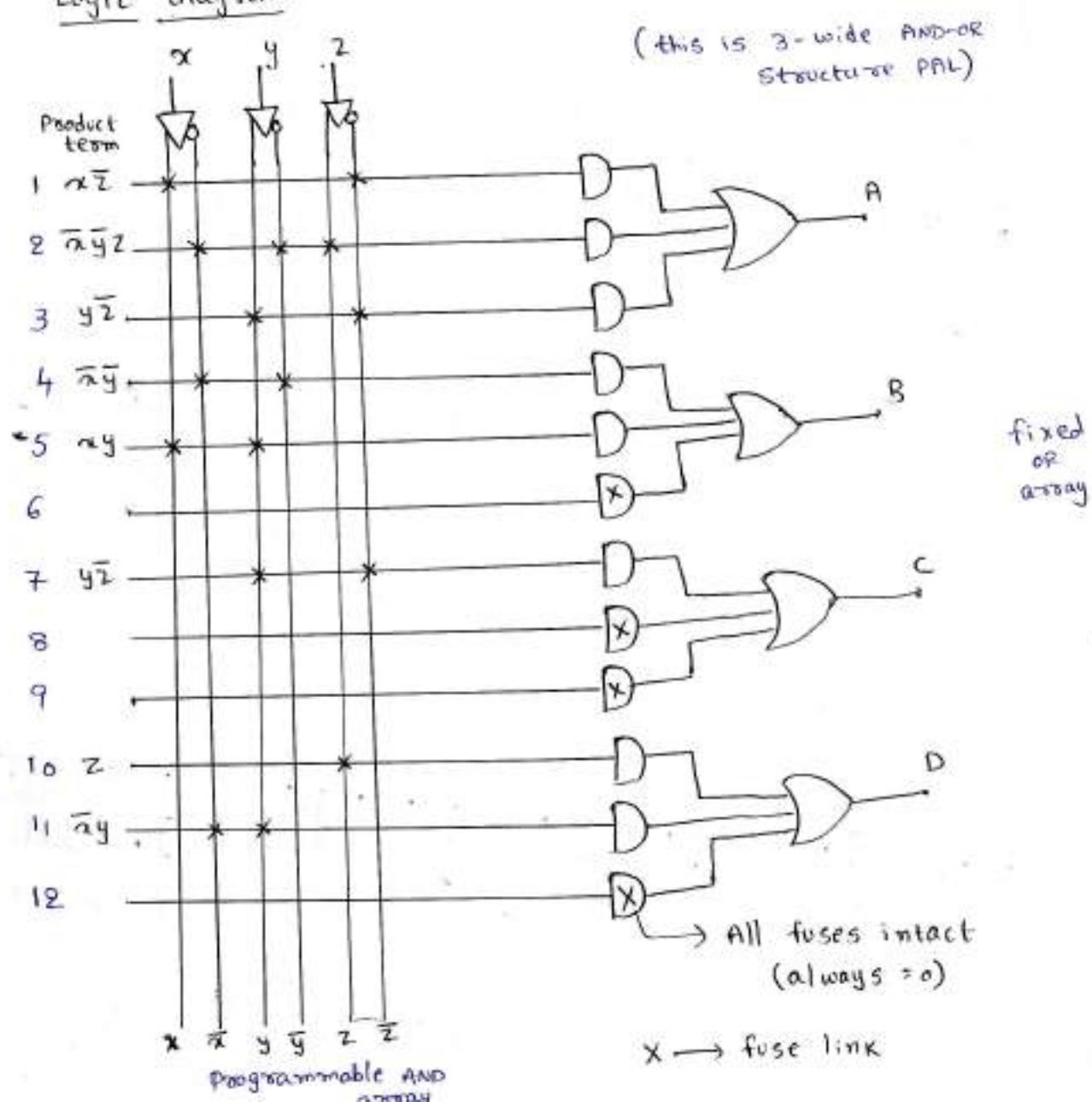


$$D = z + x'y$$

PAL Program Table

Product terms	AND inputs			outputs
	x	y	z	
1 $x\bar{z}$	1	-	0	
2 $\bar{x}\bar{y}z$	0	0	1	$A = x\bar{z} + \bar{x}\bar{y}z + y\bar{z}$
3 $y\bar{z}$	-	1	0	
4 $\bar{x}\bar{y}$	0	0	-	$B = \bar{x}\bar{y} + xy$
5 $\bar{x}y$	1	1	-	
6 $y\bar{z}$	-	1	0	$C = y\bar{z}$
7 $z$	-	-	1	
8 $\bar{x}y$	0	1	-	$D = z + \bar{x}y$
9				
10				
11				
12				

Logic diagram



- Q Implement the following Boolean functions using 3 wide AND-OR structure  
 $w(A, B, C, D) = \sum m(0, 2, 6, 7, 8, 9, 12, 13)$  PAL  
 $x(A, B, C, D) = \sum m(0, 2, 6, 7, 8, 9, 12, 13, 14)$  PAL  
 $y(A, B, C, D) = \sum m(2, 3, 8, 9, 10, 12, 13)$   
 $z(A, B, C, D) = \sum m(1, 3, 4, 6, 9, 12, 14)$

Solu

### K-map Simplification

for  $w$

AB\CD	00	01	11	10
00	0	0	1	1
01	0	1	1	0
11	1	1	0	0
10	1	1	0	0

$w = \overline{A} \overline{B} \overline{D} + \overline{A} B C + A \overline{C}$

for  $x$

AB\CD	00	01	11	10
00	1	0	0	1
01	0	1	0	0
11	1	1	0	0
10	1	1	0	0

$$\begin{aligned}x &= \overline{A} \overline{B} \overline{D} + \overline{A} B C + A \overline{C} + B C \overline{D} \\&= w + B C \overline{D}\end{aligned}$$

for  $y$

AB\CD	00	01	11	10
00	0	0	0	0
01	0	1	1	0
11	1	1	0	0
10	1	1	0	0

$y = \overline{A} B C + \overline{B} C \overline{D} + A \overline{C}$

for  $z$

AB\CD	00	01	11	10
00	0	0	1	0
01	1	0	0	0
11	1	1	0	0
10	1	1	0	0

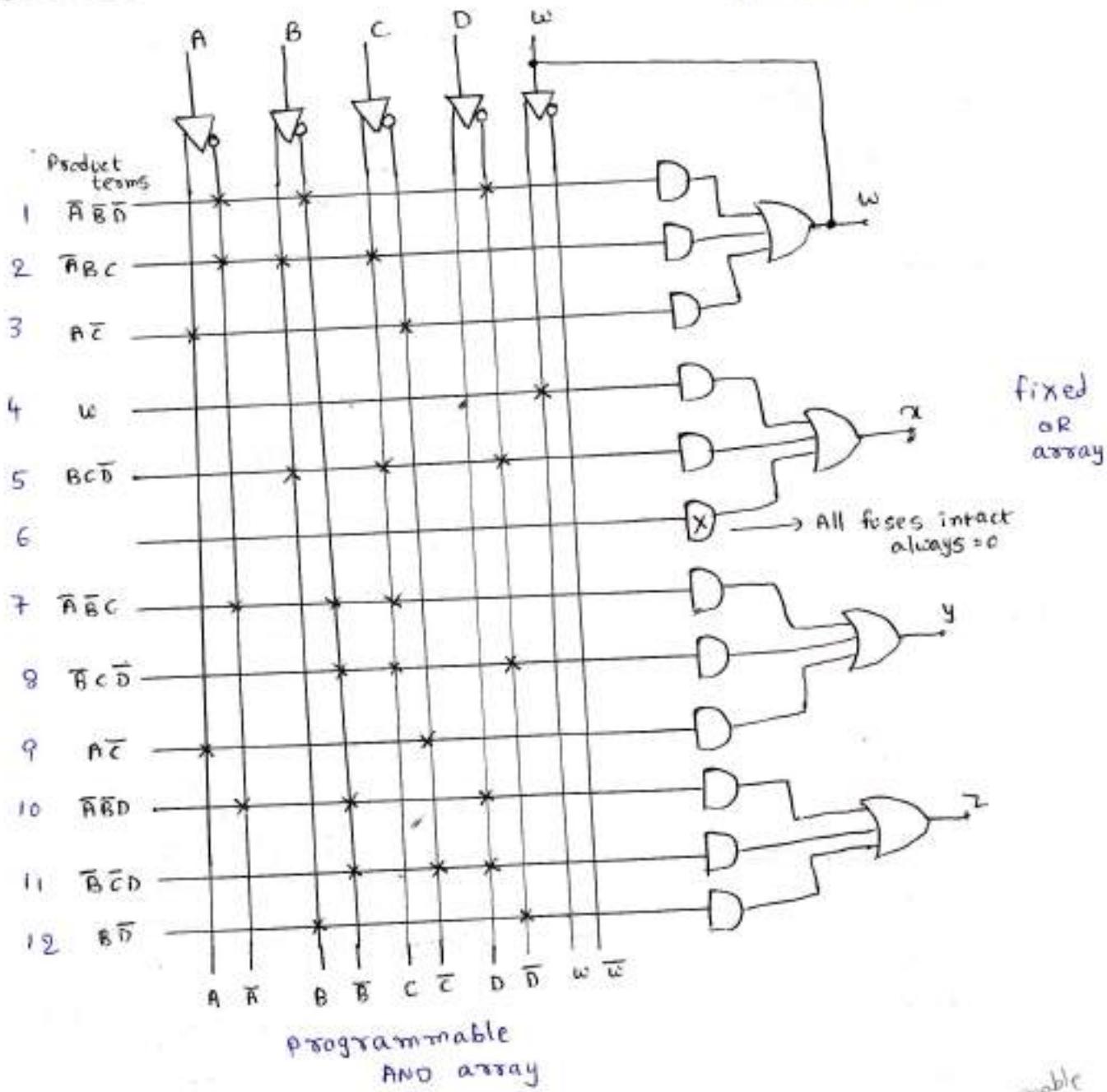
$z = \overline{A} \overline{B} D + \overline{B} C \overline{D} + B \overline{D}$

### PAL Program Table

Product terms	AND inputs				outputs	
	A	B	C	D		
$\overline{A} \overline{B} \overline{D}$	0	0	-	0	-	$w = \overline{A} \overline{B} \overline{D} + \overline{A} B C + A \overline{C}$
$\overline{A} B C$	0	1	1	-	-	
$A \overline{C}$	1	-	0	-	-	
$w$	-	-	-	-	1	$x = w + B C \overline{D}$
$B C \overline{D}$	-	1	1	0	-	
$\overline{B} B C$	0	0	1	-	-	$y = \overline{A} B C + \overline{B} C \overline{D} + A \overline{C}$
$\overline{B} C \overline{D}$	-	0	1	0	-	
$A \overline{C}$	1	-	0	-	-	
$\overline{A} \overline{B} D$	0	0	-	1	-	$z = \overline{A} \overline{B} D + \overline{B} C \overline{D} + B \overline{D}$
$\overline{B} C \overline{D}$	-	0	0	1	-	
$\overline{C} \overline{D}$	-	1	-	0	-	

## Logic diagram

This is 3-wide AND-OR structured PAL



## Comparison Between PROM, PLA and PAL

	PROM Programmable AND Fixed OR	PLA Both AND and OR arrays are programmable	PAL Programmable AND Fixed OR
1	AND array is fixed and OR array is programmable	Both AND and OR arrays are programmable	OR array is fixed and AND array is programmable
2	cheaper and simple to use	costliest and complex than PAL and PROMs	cheaper and simpler
3	All minterms are decoded	AND array can be programmed to get desired minterms	AND array can be programmed to get desired minterms
4	only Boolean functions in standard SOP form can be implemented using PROM	Any Boolean functions in SOP form can be implemented using PLA	Any Boolean function in SOP form can be implemented using PAL

## Error Detecting and correcting codes

when the digital information in the binary form is transmitted from one ckt to another ckt an error may occur. This means signal corresponding to 0 may change to 1 (or) vice versa due to presence of noise.

To maintain the data integrity between Transmitter and Receiver, extra bit or more than one bit are added in the data. These extra bits allow the detection and sometimes correction of error in the data. The data along with the extra bit/bits forms the code.

Codes which allow only error detection are called error detecting codes.

Codes which allow error detection and correction are called error detecting and corresponding codes.

Example of error detecting code is parity code.  
and error detecting and correction code is Hamming code.

### Hamming code

Hamming code not only provides the detection of a bit error, but also identifies which bit is in error so that it can be corrected. Thus Hamming code is called error detecting and correcting code.

## Hamming code

Q: encode the binary word 11000100 into even parity Hamming code

Solu: H.C comprises of certain parity bits. The no. of parity bits used by the H.C depends on the no. of data bits. Thus, we can derive the no. of parity bits used by the H.C for the given no. of data bits from the eqn given below

$$2^P \geq n+P+1$$

where  $P \rightarrow$  no. of parity bits       $n \rightarrow$  no. of data bits

given binary word = 11000100

$$n=8, \therefore P=4$$

$$\therefore \text{total no. of bits in H.C} = 8+4=12$$

place 4 parity bits in positions  $2^0, 2^1, 2^2, 2^3$  respectively  
 $(1, 2, 4, 8)$

so, the H.C is of the form

bit position:	1	2	3	4	5	6	7	8	9	10	11	12
	$P_0$	$P_1$	1	$P_2$	1	0	0	$P_3$	0	1	0	0

To determine parity bit  $P_0$ , consider 1, 3, 5, 7, 9, 11 bits

$$\text{for } P_0 \rightarrow 1\ 3\ 5\ 7\ 9\ 11 \rightarrow P_0\ 11000$$

we are using even parity.  $\therefore$  To make no. of 1's in the

above group even  $P_0$  must be '0'.

$$\therefore P_0 = 0$$

$$\text{for } P_1 \rightarrow 2\ 3\ 6\ 7\ 10\ 11 \rightarrow P_1\ 10010$$

$$\therefore P_1 = 0$$

for  $P_2 \rightarrow 4\ 5\ 6 + 12 \rightarrow P_2 = 1\ 0\ 0\ 0$

$P_3$	$P_2$	$P_1$	$P_0$
0 - 0	0	0	0
1 - 0	0	0	1
2 - 0	0	1	0
3 - 0	0	1	1
4 - 0	1	0	0
5 - 0	1	0	1
6 - 0	1	1	0
7 - 0	1	1	1
8 - 1	0	0	0
9 - 1	0	0	1
10 - 1	0	1	0
11 - 1	0	1	1
12 - 1	1	0	0
13 - 1	1	0	1
14 - 1	1	1	0
15 - 1	1	1	1

for  $P_3 \rightarrow 8\ 9\ 10\ 11\ 12 \rightarrow P_3 = 0\ 1\ 0\ 0$

$$\therefore P_3 = 1$$

$\therefore$  12-bit even parity H.C for 8-bit data word  $11\ 000\ 100$  is  $\rightarrow$

$$00\ 111\ 0\ 0\ 10\ 100$$

Q single error detection  
Detect and correct errors in the Hamming code  
received as  $000\ 110010100$  with even parity

Consider 12-bit H.C shown above, but  $3^{\text{rd}}$  bit error. we have to detect this

Solve

Received even parity H.C is  $000\ 110010100$

bit position	1	2	3	4	5	6	7	8	9	10	11	12
	0	0	0	1	1	0	0	1	0	1	0	0

To detect error bit, find error word  $c = c_3\ c_2\ c_1\ c_0$

if  $c = 0000$ , that indicates no error has occurred.

if  $c \neq 0$ , then the 4-bit binary no. gives the position of the erroneous bit

total no. of bits in error word depends on formula

$$2^c \geq n+p \quad \text{where } n+p = \text{no. of bits in H.C}$$

$c = \text{no. of bits in error word}$

$$n+p = 12, \therefore c=4 \text{ i.e. error word has 4-bits}$$

$$c = c_3\ c_2\ c_1\ c_0$$

To determine  $c_0$ , consider 1, 3, 5, 7, 9, 11 bits

$$c_0 \rightarrow 1 \ 3 \ 5 \ 7 \ 9 \ 11 \rightarrow 0 \ 0 \ 1 \ 0 \ 0 \ 0$$

We are using even parity H.C  
but, total no. of ones in above group are odd. . . there  
is an error bit in the above group.

$$\therefore c_0 = 1$$

for

$$c_1 \rightarrow 2 \ 3 \ 6 \ 7 \ 10 \ 11 \rightarrow 0 \ 0 \ 0 \ 0 \ 1 \ 0$$

There is no even no. of 1's  
∴ parity checks for even parity is wrong.

$$\therefore c_1 = 1$$

for  $c_2 \rightarrow 4 \ 5 \ 6 \ 7 \ 12 \rightarrow 1 \ 1 \ 0 \ 0 \ 0$

There is even no. of 1's. correct parity

$$\therefore c_2 = 0$$

for  $c_3 \rightarrow 8 \ 9 \ 10 \ 11 \ 12 \rightarrow 1 \ 0 \ 1 \ 0 \ 0$

correct parity

$$\therefore c_3 = 0$$

Now,  $C = c_3 \ c_2 \ c_1 \ c_0 = 0 \ 0 \ 1 \ 1$

∴ 3<sup>rd</sup> bit in the received message is wrong.

This bit then should be complemented, so that the correct code will be 00110010100

Note: for even parity, we use XOR gates for generating parity bits  
for odd " " XNOR " "  
i.e. for  $P_0 \rightarrow$  XOR of bits (1, 3, 5, 7, 9, 11)

## Double-Error Detection

The Hamming code can detect and correct only a single error. Multiple errors are not detected.

By adding another parity bit to the coded word, the Hamming code can be used to correct a single error and detect double errors.

If we include this additional parity bit, then the previous 12-bit Hamming coded word becomes  $001110010100 P_{13}$ .

where  $P_{13}$  makes no. of 1's in the message even if we are using even parity H.C

$001110010100 P_{13}$

(for even parity)  $P_{13} = 1$

$\therefore$  13-bit H.C word =  $0011100101001$

Transmitter then sends this 13-bit word to receiver. After receiving this 13-bit word, receiver calculates error word ( $C$ ) and also calculates the parity ( $P$ ) over the entire 13-bits.

After calculating  $C \& P$  four cases can occur:

If  $C=0$  and  $P=0$ , no error occurred

If  $C \neq 0$  and  $P=1$ , a single error occurred that can be & corrected

If  $C \neq 0$  and  $P=0$ , a double error occurred that is detected but that cannot be corrected.

If  $C=0$  and  $P=1$ , an error occurred in the  $P_{13}$  bit.