RS-(60)/-

Syllabus (For Gate) :

(a) Number systems.

(i) Conversions.

(ii) Signed representation

(b) Boolean algebra.

(c) Logic gates.

(d) K-map

(e) Combinational circuits.

(f) Sequential circuits → synchronous.
→ asynchronous.

(g) A to D and D to A converters.

Textbooks :

(i) Switching theory & logic design — KOHAVI.

(ii) Model digital ckts → R.P. JAIN
(for K-MAP)

(iii) (for sequential circuits) → V.K PURI.

(iv) Wakerley.

14/9/14 E.K.D.C.V.V

NUMBER SYSTEMS

$d_n\ d_{n-1} \ldots d_3 d_2 d_1 d_0 \cdot d_{-1} d_{-2} d_{-3} \ldots d_{-m}$

Integer part — radix point — fraction part

$r^n\ r^{n-1} \ldots r^2\ r^1\ r^0 \cdot r^{-1}\ r^{-2} \ldots r^{-m}$

$r^n \longrightarrow$ weight of the position.

$d_n r^n \longrightarrow$ weighted value of digit in that position.

Base (or) radix :-

The no. of different symbols used to represent a number in that number system is called base (or) radix of the number system.

Every number consists of two parts integer & fraction separated by radix point.

Conversions :- (Base conversions)

(a) Decimal to any base :-

→ To get integer part take successive divisions of decimal integer part with the required base and accumulate remainders in reverse order.

→ To get fraction part take successive multiplications of decimal fraction part with the required base & accumulate integers in original order.

Ex:- Convert to base 2, 8, 16, 4 for $(108.6875)_{10}$.

(108)

Sol:-

2 | 108	
2 | 54 – 0	
2 | 27 – 0	
2 | 13 – 1	
2 | 6 – 1	
2 | 3 – 0	
1 – 1	

$(108)_{10} \longrightarrow (1101100)_2$

$0.6875 \times 2 \longrightarrow 1.375 \longrightarrow 1$ ↓

$0.375 \times 2 \longrightarrow 0.75 \longrightarrow 0$ ↓

$0.75 \times 2 \longrightarrow 1.5 \longrightarrow 1$ ↓

$0.5 \times 2 \longrightarrow 1.0 \longrightarrow 1$.

$\therefore (108.6875)_{10} \longrightarrow (1101100.1011)_2$.

$$\begin{array}{r|l} 8 & 108 \\ 8 & 13-4 \\ & 1-5 \end{array} \qquad (108)_{10} \longrightarrow (154)_8 .$$

$0.6875 \times 8 \longrightarrow 5.5 \longrightarrow 5$

$0.5 \times 8 \longrightarrow 4 \longrightarrow 4$

$$(0.6875)_{10} \longrightarrow (54)_8 .$$

$$\therefore (108.6875)_{10} \longrightarrow (154.54)_8 .$$

$$\begin{array}{r|l} 16 & 108 \\ 16 & 6-C \\ & 6-0 \end{array} \qquad (108) \qquad \begin{array}{r|l} 16 & 108 \\ & 6-C \end{array}$$

$$(108)_{10} \longrightarrow (6C)_{16} .$$

$0.6875 \times 16 \longrightarrow 11 \longrightarrow B$

$$(0.6875)_{10} \longrightarrow (B)_{16}$$

$$\therefore (108.6875)_{10} \longrightarrow (6C.B)_{16} .$$

$$\begin{array}{r|l} 4 & 108 \\ 4 & 27-0 \\ 4 & 6-3 \\ & 1-2 \end{array} \qquad (108)_{10} \longrightarrow (1230)_4 .$$

$0.6875 \times 4 \longrightarrow 2.75 \longrightarrow 2$

$0.75 \times 4 \longrightarrow 3 \longrightarrow 3$

$$(0.6875)_{10} \longrightarrow (.23)_4 .$$

$$\therefore (108.6875)_{10} \longrightarrow (1230.23)_4 .$$

(b) Any base to decimal :-

To get decimal equivalent of a number in base 'r' take summation of weighted values of all digits.

Ex:- $(10110.011)_2$

$$1\times2^4 + 0\times2^3 + 1\times2^2 + 1\times2^1 + 0\times2^0 + 0\times2^{-1} + 1\times2^{-2} + 1\times2^{-3}$$

$$\Rightarrow 16 + 4 + 2 + 0.25 + 0.125$$

$$\Rightarrow (22.375)_{10} .$$

Ex:- $(627.24)_8$

$\Rightarrow 6\times8^2 + 2\times8 + 7\times8^0 + 2\times8^{-1} + 4\times8^{-2}$

$\Rightarrow (407.3125)_{10}$.

Ex:- $(3AF.C)_{16}$

$\Rightarrow 3\times16^2 + 10\times16 + 15 + 12\times16^{-1}$

$\Rightarrow (943.75)_{10}$.

Ex:- $(608.74)_9$.

Find the eq. number in base 3

$\Rightarrow 6\times9^2 + 8 + 7\times9^{-1} + 4\times9^{-2}$

$\Rightarrow (494.827160\overline{5})_{10}$

3 | 494	
3 | 164 - 2	$(494)_{10} \Rightarrow$
3 | 54 - 2	$(200022)_3$.
3 | 18 - 0	
3 | 6 - 0	
2 - 0	

$0.8271605 \times 3 \rightarrow 0.827 \times 3 \rightarrow 2.481$

$0.481\times3 \rightarrow 1.443$.

$0.443\times3 \rightarrow 1.329$

$0.329\times3 \rightarrow 0.987$

$0.987\times3 \rightarrow 2.961$

$0.961\times3 \rightarrow 2.883$.

$(608.74)_9 \cong (200022.211022)_3$.

© Converting base (b^n) to base (b) :-

Each digit in base b^n can be represented using 'n' number of digits in base 'b'.

Ex:- $(608.74)_9 \longrightarrow (\quad)_3$.

$9 \rightarrow 3^2$ to 3.

(9)	(3)
0	00
1	01
2	02
3	10
4	11
5	12
6	20
7	21
8	22

$9 = 3^2$

$11 \rightarrow 1\times3^1 + 1\times3^0$

$\therefore (608.74)_9 \longrightarrow (200022.2111)$

ⓓ Converting hexadecimal to binary.

base 16 → 2

$4^2 \rightarrow 4$

(16)	(2)	(4)
0	0000	00
1	0001	01
2	0010	02
3	0011	03
4	0100	10
5	0101	11
6	0110	201
7	0111	211
8	1000	20
9	1001	21
A	1010	22
B	1011	23
C	1100	30
D	1101	31
E	1110	32
F	1111	33

F

Ex:- $(704.25)_8 \longrightarrow (\quad)_2$

$8 = 2^3 \rightarrow 2$

$\Rightarrow (111000100.010101)_2$.

Ex:- $(3AF.C)_{16} \longrightarrow (\quad)_2$.

$\Rightarrow (0011\,1010\,1111.1100)_2$.

$\Rightarrow (032233.30)_4$.

(e) Converting base (b) to base (b^n):-

Ex:- find base 2 to base 8, 16.

$(1011110.11011)_2$

$\Rightarrow 001\,011\,110.110\,110$

$\Rightarrow (136.66)_8$.

$\Rightarrow \overset{8\ 4\ 2\ 1}{0101\,1110.1101\,1000}$

$\Rightarrow (5E.D8)_{16}$.

Ex:- $22 + 23 = 100$. Then what is the base value of the numbers.

$(22)_x + (23)_x = (100)_x$.

$(2x+2)_{10} + (2x+3)_{10} = (x^2)_{10}$

$2x+2+2x+3 = x^2$

$x^2 - 4x - 5 = 0$.

$x^2 - 5x + x - 5 = 0$

$x = 5, -1$.

As base value cant be -ve, base is 5.

Ex:- $\sqrt{41} = 5$. What can be the possible base value.

$\sqrt{41} = 5$.

$\Rightarrow \sqrt{(41)_x} = (5)_x$.

$\Rightarrow 4x+1 = (25)_x$

$4x = 24$

$x = 16$.

$\Rightarrow 4x+1 \neq 2x+9$,

$\Rightarrow \sqrt{4x+1} = 5 \times x^0 = 5$.

$4x+1 = 25$

$4x = 24$

$x = 6$.

Ex:- find the base value which satisfies the equation $\frac{55}{5} = 11$.

$\frac{5x+5}{5} = x+1$.

$5x+5 = 5x+5$

equation is valid for any value of x.

$\therefore \frac{55}{5} = 11$ is valid for any base value.

but base ≥ 6, as 5 is used in the number ; so base ≥ 6.

Ex:- find the base values which satisfies the equation,

$\frac{302}{20} = 12.1$.

$\frac{3x^2+2}{2x} = x+2+\frac{1}{x}$.

$\frac{3x^2+2}{2x} = \frac{x^2+2x+1}{x}$

$x^2 - 4x = 0$.

$x = 0, 4$.

∴ Base is 4 as base value cannot be equal to zero.

Ex:- find base value which satisfies the following two equations simultaneously

(i) $2+3=5$ and $2\times3=10$

⇒ $2+3=5$

Base value ≥ 6.

$2\times3=10$

$2\times3 = x$

$x = 6$.

∴ Base is 6.

(ii) $4+5=9$ and $4\times2=10$.

$4+5=4$

∴ Base $\not\leq 6$.

$4\times2=10$.

Base $\not\leq 5$

$4+5=9$.

Base ≥ 10.

$4\times2=10$. These two equations cannot be satisfied simultaneously with any base.

$8 = x$

Ex:- find the base values which satisfy the equation $\sqrt{144}=12$.

$\sqrt{x^2+4x+4} = x+2$.

$x^2+4x+4 = (x+2)^2$.

$= x^2+4x+4$

equation is valid for any base value ≥ 5.

* Find the number of possible val for 'x' which satisfy the equation

$(43)_8 = (x0)_y$.

Sol:- $(43)_8 = (x0)_y$

$4\times8+3 = xy$.

$32+3 = xy$.

$35 = xy$.

$x = \frac{35}{y}$.

$y = \frac{35}{x}$.

x	y	
1	35	✓
35	1	×
5	7	✓
7	5	×

Two possible values of x.

Pgno.20 Q.7

* Find number of possible solutions

$(123)_5 = (x8)_y$.

Sol:- $1\times25 + 2\times5 + 3 = xy + 8$.

$25+10+3 = xy+8$.

$38 = xy+8$.

$xy = 38-8 = 30$.

$y \geq 9$.

x	y	
1	30	✓
2	15	✓
3	10	✓
5	6	×
6	5	×
10	3	×
15	2	×
30	1	×

3 possible solutions.

* Pgno.19 Q.1.

Sol:- $(135)_x + (144)_x = (323)_x$

$\Rightarrow x^2+3x+5+x^2+4x+4 = 3x^2+2x+3$

$\Rightarrow x^2-5x-6=0$

$x = 6, -1.$

$\therefore$ Base $x = 6$.

* Pgno.19 Q.4

Sol:- $(657)_8$

$\Rightarrow (110101111)_2$

8421

0001 | 1010 | 1111

1 | A | F

$\Rightarrow (1AF)_H$.

* Pgno.20 Q.6

Sol:- $\frac{312}{20} = 13.1.$

$\frac{3x^2+x+2}{2x} = x+3+\frac{1}{x}.$

$\frac{3x^2+x+2}{2x} = \frac{x^2+3x+1}{x}$

$3x^2+x+2 = 2x^2+6x+2$

$x^2-5x=0$

$x = 0, 5$

$\therefore$ Base $x = 5$.

15/9/14

Addition in Other base values :-

Ex:-
$$\begin{pmatrix} 2\ 6\ 5\ 7\ 8 \\ (+)\ 3\ 5\ 7\ 6\ 4 \\ \hline 6\ 3\ 4\ 5\ 3 \end{pmatrix}_9$$

→ add every two digits in decimal. If sum is < base, keep the sum as it is.

→ If sum ≥ base, then subtract the sum and base and put carry '1' to the previous digits.

[This procedure only for adding two numbers]

*
$$\begin{array}{r} {\scriptstyle 1\ 1\ 1\ 1} \\ 2\ 6\ 5\ 4\ 3 \\ +\ 3\ 5\ 2\ 6\ 4 \\ \hline 6\ 5\ 1\ 4\ 0 \end{array}_7$$

*
$$\begin{array}{r} {\scriptstyle 1\ \ 0\ 1} \\ 4\ 6\ 2\ 4\ 7 \\ +\ 5\ 7\ 3\ 2\ 4 \\ \hline 1\ 2\ 5\ 5\ 0\ 3 \end{array}_8$$

Subtraction :-

*
$$\begin{pmatrix} 6\ 4\ 3\ 7\ 5 \\ -\ 2\ 7\ 8\ 4\ 6 \\ \hline 3\ 5\ 4\ 2\ 8 \end{pmatrix}_9$$

*
$$\begin{pmatrix} 6\ 4\ 0\ 2\ 5 \\ -\ 3\ 6\ 5\ 4\ 3 \\ \hline 2\ 4\ 1\ 5\ 2 \end{pmatrix}_7$$

Ex:-
$$\begin{pmatrix} 6\ 2\ 0\ 4\ 7 \\ -\ 3\ 7\ 6\ 4\ 5 \\ \hline 2\ 2\ 2\ 0\ 2 \end{pmatrix}_8$$

*
$(1.2)_4 + (2.3)_4 = (y)_4.$

$$\left(\begin{array}{r} {\scriptstyle 1} \\ 1.2 \\ +\ 2.3 \\ \hline 10.1 \end{array}\right)_4$$

$\Rightarrow$
$$\begin{array}{r} {\scriptstyle 1\ 1} \\ 01.10. \\ +\ 10.11 \\ \hline 100.01 \end{array}$$

$\Rightarrow (10.1)_4.$

COMPLEMENTS

(i) Radix complements [r's].

(ii) Diminished radix complements [(r-1)'s].

$(N)_r = d_{n-1} d_{n-2} \ldots d_1 d_0 \cdot d_{-1} d_{-2} \ldots d_m$

r's complement :-

Assume a number(N_r) has 'n' number of digits in integer part and 'm' number of digits in fractional part, then

r's complement $= r^n - N \ (\forall N \neq 0)$

$= 0 \ (\forall N = 0)$.

(r-1)'s complement :-

(r-1)'s complement $= r^n - r^{-m} - N$.

Ex:- find 10's & 9's complement of $(0030700)_{10}$.

10's comp $\longrightarrow 10^7 - 0030700$

$$\begin{array}{r} 10000000 \\ \Rightarrow \quad 0030700 \\ \hline 9969300 \end{array}$$

9's complements $\longrightarrow 10^7 - 10^{-0} - (0030700)$.

$$\Rightarrow \begin{array}{r} 10000000 \\ 0030700 \\ \hline 9969299 \end{array}$$

Ex:- $(0607.89000)_{10}$

10's complement $\longrightarrow$

$$\begin{array}{r} 10000.00000 \\ \cdot 0607.89000 \\ \hline 9392.11000 \end{array}$$

9's complement $\longrightarrow$

$10^4 - 10^{-5} - (0607.89000$

$10000 - 0.00001 - 0607.8900$

$$\Rightarrow \begin{array}{r} 10000.00000 \\ 0607.89001 \\ \hline 9392.10999 \end{array}$$

Ex:- $(00307.20600)_8$.

8's complement $\longrightarrow 8^5 = 4096$

4096/0606 0
0307.2060 0

32768.00000
00307.20600
/ / /200/

$(8^5)_8 = 100000.00000$.

$$\begin{array}{r} 100000.00000 \\ 00307.20600 \\ \hline 77470.57200 \end{array}$$

7's complement $\longrightarrow (8^5)_8 - (8^{-5}) - (00307.20600$

$$\Rightarrow \begin{array}{r} 100000.00000 \\ 00307.20601 \\ \hline 77470.57177 \end{array}$$

Ex:- $(0101001.10100)_2$.

2's complement ⟶

```
 10000000.00000
-0101001.10100
---------------
 1010110.01100
---------------
```

1's complement ⟶

```
 10000000.00000
-0101001.10101
---------------
 1010110.01011
---------------
```

Ex:- find 9

Note:- To get r's complement of a number in base r, leave least significant zeros as it is, substract first non-zero least significant digit from 'r' and substract remaining digits each from (r-1).

To get (r-1)'s complement of a number in base r, substract each digit from (r-1).

Ex:- 01001.10100

2's comp ⟶ 10110.01100

1's comp ⟶ 10110.01011.

Signed representation of binary numbers :-

(i) Signed magnitude.

(ii) 1's complement.

(iii) 2's complement.

Signed magnitude representation :-

In signed magnitude representation, MSB is used explicitly for sign which doesnot have any magnitude and the remaining bits are used for magnitude.

If MSB = 0, number is positive and if MSB = 1, number is negative.

Ex:- 0 000 ⟶ +0.

1 000 ⟶ -0.

0 001 ⟶ +1

0 100 ⟶ +4

1 101 ⟶ -5

The range of 4 bit is

-7 to +7

$-(2^{4-1}-1)$ to $(2^{4-1}-1)$.

The range of n-bit is

$$\boxed{-(2^{n-1}-1) \text{ to } (2^{n-1}-1)}.$$

The range of decimal integers can be represented in n-bits in signed magnitude representation is as above.

In this representation zero has two representations +0 (0000) and −0 (1000).

1's complement representation :-

$-(2^{n-1}-1)\ 2^{n-2}\ 2^{n-3} \ldots\ldots 2^1\ 2^0$

MSB

1000 → −7.
1001 → −6.
1100 → −3
1111 → −0
0001 → +1
0101 → +5
0111 → +7
0000 → +0

The range is −7 to +7.

In 1's complement representation, the only bit in MSB has −ve weight and the weight is $-(2^{n-1}-1)$ and the remaining bits have +ve weights.

The range of decimal integers can be represented using 'n' bits in 1's complement representation is

$$-(2^{n-1}-1) \text{ to } (2^{n-1}-1)$$

In this representation, zero has two representations +0 (0000) and −0 (1111)

2's complement representation :-

$-(2^{n-1})\ 2^{n-2}\ 2^{n-3} \ldots\ldots 2^1\ 2^0$.

1000 → −8
1001 → −7
1100 → −4
1111 → −1
0000 → 0
0001 → +1
0101 → +5
0111 → +7.

range is (−8 to +7).

In this representation, the only bit in MSB has −ve weight and the weight is $-(2^{n-1})$ and the remaining bits have positive weights.

The range is

$$-2^{n-1} \text{ to } (2^{n-1}-1)$$

The range of decimal integers can be represented using n-bits in 2's complement representation is as above.

In this representation, zero has only one representation (0000).

Ex:- 1's comp

1's 1001 ⇒ −6
0110 ⇒ +6.

1's 0101 ⇒ +5
1010 ⇒ −5

2's comp

2's 1011 ⇒ −5.
0101 ⇒ +5

2's 0110 ⇒ +6
1010 ⇒ −6.

Ex:- 1000 ⇒ -8
2's → 1000 ⇒ -8
not giving complement form beoz of range.

Overflow :-

If the number of bits is not sufficient to represent the result, then we say there is overflow.

1st &

1's complement

1010 ⟶ -5
11010 ⟶ -5
111010 ⟶ -5
11111010 ⟶ -5

0100 ⟶ +4
00100 ⟶ +4
00000100 ⟶ +4.

2's complement

1001 ⟶ -7
111001 ⟶ -7
11111001 ⟶ -7

0101 ⟶ +5
000101 ⟶ +5
00000101 ⟶ +5.

In 1's comp and 2's comp representation, the extension of sign bit doesnot change the decimal equivalent value.

Ex:- 2's comp
11000 ⟶ -8
2's → 01000 ⟶ +8.

Ex:- find signed magnitude, 1's comp and 2's comp of -27 using 8 bits.

16 8 4 2 1
⇒ (+27) ⟶ 1 1 0 1 1

1st comp of -27 ⟶

11100100

2's comp of -27 ⟶

11100101

method 2 :-

2 | 27
2 | 13 - 1
2 | 6 - 1
2 | 3 - 0
1 - 1

$(27)_{10} \longrightarrow (11011)_2$.

+27 ⟶ 00011011.
-27 ⟶ 10011011 ↷ signed mag. representation.

-31+4 = -27.
= 100100.
= 11100100 ↷ 1's comp. representation, of -27

-27 = 100100.

-32+5 = -27
= 100101 (-32, +5)
= 11100101 ↷ 2's comp representation of -27.

Method 3 :-

+27 → 00011011
-27 → 11100100 } 1's comp.

2's comp of +27 → 11100101
⇒ -27

Ex:- Find signed magnitude, one's complement and 2's complement representations of +28 using 8 bits.

Sol:- signed mag → 00011100.

2|28
2|14-0
2|7-0
2|3-1
1-1

1's comp → 00011100.
⇒ 11100011
2's comp → 00011100.
⇒

For +ve integers signed magnitude, 1's comp and 2's comp representations are same.

Substraction with complements :-

(i) 1's complement.
(ii) 2's complement.

(i) 2's complement substraction :-

Add 'A' to the 2's complement of 'B'. Observe the result of obtained in step1.

(a) If A ≥ B, sum produces an end carry. Discard the carry, sum is the desired result.

(b) If A < B, sum fr doesnot produce an end carry. Take 2's complement of sum and produce -ve sign infront to get the desired result.

(ii) 1's complement method :-

→ Add 'A' to the 1's complement of B.

→ Observe the result obtained in step 1.

(a) If A > B, sum produces an end carry, add '1' to the LSB to get the desired result

(b) If A < B, sum doesnot produ an end carry, Take 1's complement of sum and produce -ve sign infront get the desired result.

Ex:- substract using 1's comp and 2's comp method.

$(1001)_2 - (100)_2$

Sol:- 1's comp :-

1001
0100
⇒ 1001
1011
① 0100

0100
1
0101 ✓

2's comp :-

1001
0100

⇒ 1001
1100
① 0101

⇒ 0101 ✓

Ex:- $(0101)_2 - (1100)_2$:

Sol:-

$$\begin{array}{r} 0101 \\ -\ 1100 \end{array}$$

1's comp $\Rightarrow$

$$\begin{array}{r} 0101 \\ +\ 0011 \\ \hline 1000 \end{array}$$

1's comp $\rightarrow 0111$

$\Rightarrow -0111$ ✓

2's comp $\Rightarrow$

$$\begin{array}{r} 0101 \\ +\ 0100 \\ \hline 1001 \end{array}$$

2's comp $\rightarrow 0111$

$\Rightarrow -0111$ ✓

Ex:- Two 2's complement numbers $X = 01100$ and $Y = 11011$ are added. $X+Y$ in 2's comp format using 6 bits.

(a) 100111 (b) 000111

(c) 0110001 (d) 111001.

Sol:-

$X = 01100$

$Y = 11011$

$\Rightarrow X = 001100$

$Y = 111011$

$X+Y = 1000111$

$X+Y = 000111.$

BOOLEAN ALGEBRA

OR (+)	AND (·)	INV('or)$^-$ (NOT)
$0+0=0$	$0\cdot0=0$	$\bar{0}=1$
$0+1=1$	$0\cdot1=0$	$\bar{1}=0$.
$1+0=1$	$1\cdot0=0$	
$1+1=1$	$1\cdot1=1$.	

Dual of a function:-

To get dual of a function replace logical OR with logical AND and viceversa and replace logic 0 with logic 1 and viceversa.

Note:-

If a function is True then dual of that function is also true.

$x+0=x$	$x\cdot1=x$
$x+1=1$	$x\cdot0=0$.
$x+x=x$	$x\cdot x=x$
$x+\bar{x}=1$	$x\cdot\bar{x}=0$
$\bar{\bar{x}}=x$	$\bar{\bar{x}}=x$.
$x+y=y+x$	$x\cdot y=y\cdot x$.
$x+(y+z)=(x+y)+z$	$x\cdot(y\cdot z)=(x\cdot y)\cdot z$.
$x\cdot(y+z)=x\cdot y+x\cdot z$	$x+(y\cdot z)=(x+y)\cdot(x+z)$.
$x+xy=x$	$x\cdot(x+y)=x$.
$x+\bar{x}y=x+y$	$x\cdot(\bar{x}+y)=x\cdot y$
$\overline{x+xy}=\bar{x}+y$	$\overline{x\cdot(x+y)}=\bar{x}\cdot y$
$\overline{(x+y)}=\bar{x}\cdot\bar{y}$	$\overline{(x\cdot y)}=\bar{x}+\bar{y}$.

$\overline{x+y+z} = \bar{x}\cdot\bar{y}\cdot\bar{z}$ $\quad$ $\overline{x\cdot y\cdot z} = \bar{x}+\bar{y}+\bar{z}$.

$xy + \bar{x}z + yz = xy + \bar{x}z$ (Consensus theorem)

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

Ex:-

	A	B	C	f(A,B,C)
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0.

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

$= m_1 + m_2 + m_5 + m_6$

$= \sum m(1,2,5,6)$

$AB\bar{C}$ → canonical SOP

minterms.

$\bar{f} = \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}\bar{C} + ABC.$

$f = \overline{\bar{f}} = \overline{\bar{A}\bar{B}C + \bar{A}BC + A\bar{B}\bar{C} + ABC}.$

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

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

$= \prod M(0,3,4,7)$

canonical POS.

maxterms

In canonical representation, all the variables must be existed in complemented or uncomplemented form in all terms.

17/8/14

* pgno.19 Q.2

Sol:- −127 in 2's comp ⟶ 10000001.

−127 in 1's comp ⟶ 10000000.

m:n = 2:1.

* Q.3

sol:- 4-bit 2's comp ⟶ $x_3x_2x_1x$

8 bit ⟶ $x_3x_3x_3x_3x_3x_2x_1$

sign expansion.

* Q.5

Sol:- $(-539)_{10}$

⇒ +539 = $(01000011011)_2$

−539 = 2's comp of +539

= 1011110 0101

= $(DE5)_H$.

2	539
2	269 – 1
2	134 – 1
2	67 – 0
2	33 – 1
2	16 – 1
2	8 – 0
2	4 – 0
2	2 – 0
	1 – 0

8421.

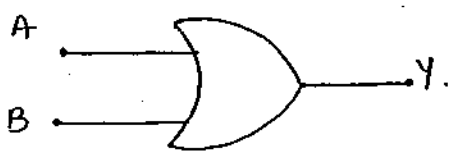
LOGIC GATES

OR GATE :-

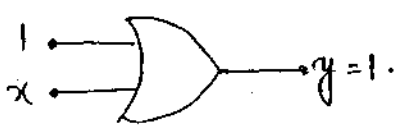
A B inputs.

OR $\Rightarrow$ A+B=Y.

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



If one i/p of OR gate is connected to zero, then the o/p depends on other i/p. So zero is called Enabled i/p.



If one i/p is one, then irrespective of other i/p, the o/p is high. So logic '1' is called disabled i/p for other i/p.

Enabled input :-

The i/p which is allowing other i/p's to pass through the gate is called Enabled i/p.

Disabled input :-

The i/p which not allowing other inputs to pass through the gat is disabled input.

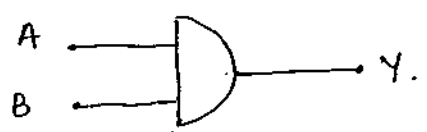
The three i/p OR gate,

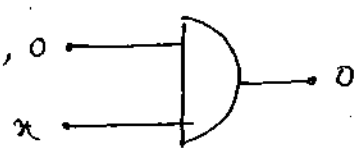
A	B	C	Y=A+B+C
0	0	0	$\longrightarrow$ 0
0	0	1	$\longrightarrow$ 1
0	1	0	$\longrightarrow$ 1
0	1	1	$\longrightarrow$ 1
1	0	0	$\longrightarrow$ 1
1	0	1	$\longrightarrow$ 1
1	1	0	$\longrightarrow$ 1
1	1	1	$\longrightarrow$ 1.

OR gate o/p is zero when all i/p's are zeros.

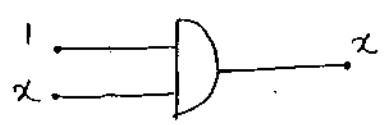
AND GATE :-

A	B	Y = A·B
0	0	0
0	1	0
1	0	0
1	1	1



for AND gate, 

$\therefore$ '0' is disabled i/p.



'1' is enabled i/p.

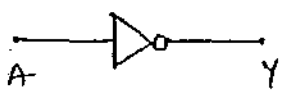
The three i/p AND gate

A	B	C	Y=ABC
0	0	0	→ 0
0	0	1	→ 0
0	1	0	→ 0
0	1	1	→ 0
1	0	0	→ 0
1	0	1	→ 0
1	1	0	→ 0
1	1	1	→ 1.

AND gate o/p is '1' only when all inputs are ones.

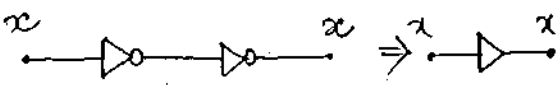
NOT gate :- A → i/p.

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



Enabled and disabled i/p are not possible for NOT gate.

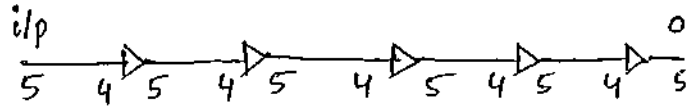
If two not gates are cascaded,



It acts as a buffer.

Buffers helps in transmitting the voltages with less noise, as it introduces noise margin.

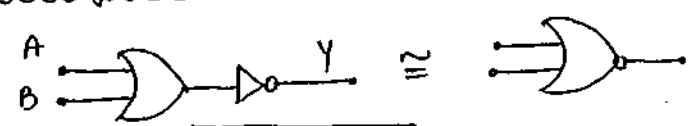
Ex:- If 1cm of wire has 0.5V loss, by introducing buffers in wire, the effect can be reduced.



The AND, OR, NOT are called basic gates as these are sufficient to implement any function.

DERIVED GATES :-

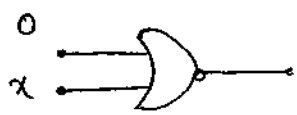
NOR gate :-



$\boxed{Y=\overline{A+B}}$.

By Demorg Law

A	B	$Y=\overline{A+B}=\overline{A}\cdot\overline{B}$
0	0	1
0	1	0
1	0	0
1	1	0

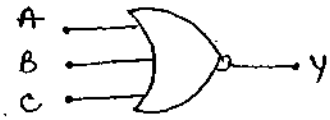


'0' is enabled i/p.

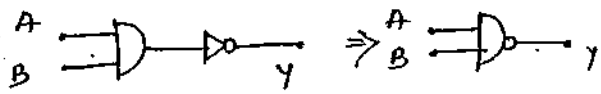
'1' is disabled i/p.

The 3-i/p NOR gate is

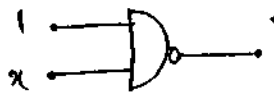
A	B	C	$Y=\overline{A+B+C}=\overline{A}\cdot\overline{B}\cdot\overline{C}$
0	0	0	→ 1
0	0	1	→ 0
0	1	0	→ 0
0	1	1	→ 0
1	0	0	→ 0
1	0	1	→ 0
1	1	0	→ 0
1	1	1	→ 0.



NAND gate :-

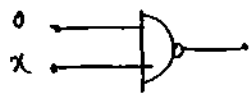


A	B	$Y = \overline{A \cdot B} = \bar{A} + \bar{B}$
0	0	$\rightarrow 1$
0	1	$\rightarrow 1$
1	0	$\rightarrow 1$
1	1	$\rightarrow 0$



$\overline{1 \cdot x} = \bar{x} = y.$

'1' is enabled i/p.



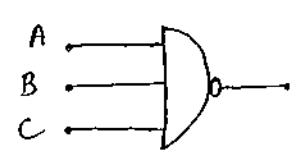
$y = \overline{0 \cdot x} = 1.$

'0' is disabled i/p.

The 3-i/p NAND gate is

A	B	C	$Y = \overline{ABC} = \bar{A} + \bar{B} + \bar{C}$
0	0	0	$\rightarrow 1$
0	0	1	$\rightarrow 1$
0	1	0	$\rightarrow 1$
0	1	1	$\rightarrow 1$
1	0	0	$\rightarrow 1$
1	0	1	$\rightarrow 1$
1	1	0	$\rightarrow 1$
1	1	1	$\rightarrow 0.$

NAND Gate O/p is zero only when all i/p's are 1's.



NAND, NOR and multiplexer are universal gates.

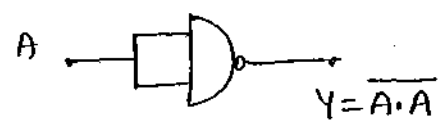
If it is possible to implement any function using a gates, then that gate is universal gate.

So, if basic gates are able to be implemented using universal gate then all the functions can be implemented using universal gate.

To prove this it must be possible to implement three basic gates (AND, OR, NOT) using only the gates then it is universal gate.

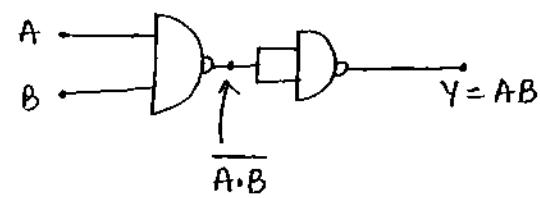
NAND gate as universal gate :-

(i) Implementation of NOT gate.

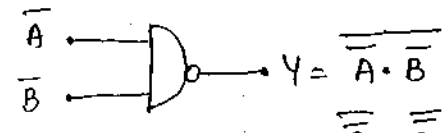


$Y = \bar{A}.$

(ii) Implementation of AND gate.

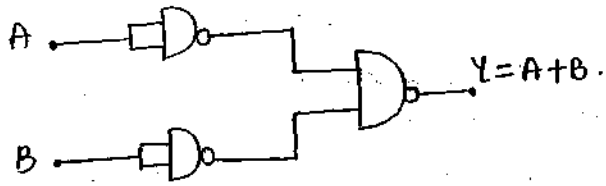


(iii) Implementation of OR gate.



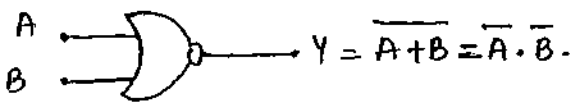
$= \bar{\bar{A}} + \bar{\bar{B}}$

$= A + B.$

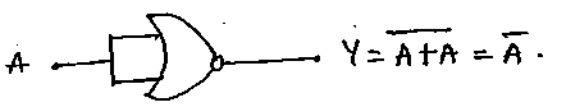


∴ NAND gate is universal gate.

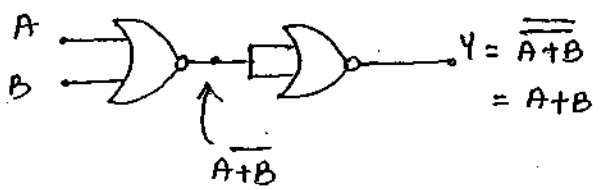
NOR gate as universal gate :-



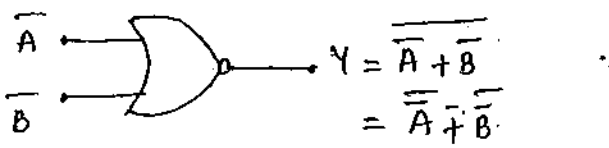
(i) NOT gate implementation.

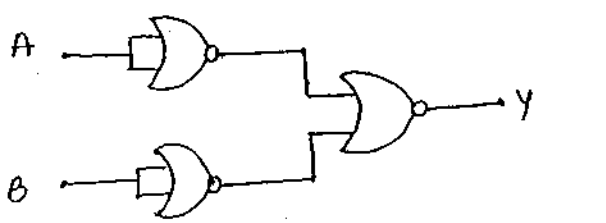


(ii) OR gate implementation.

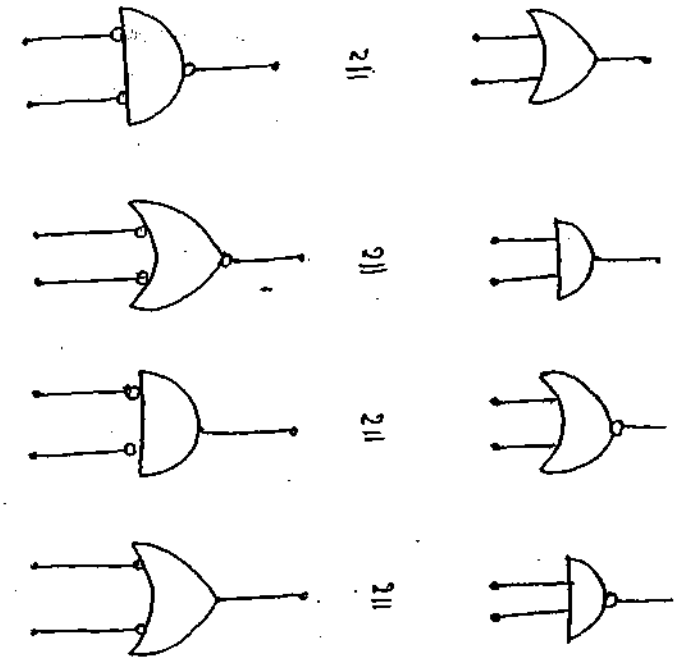


(iii) AND gate implementation.

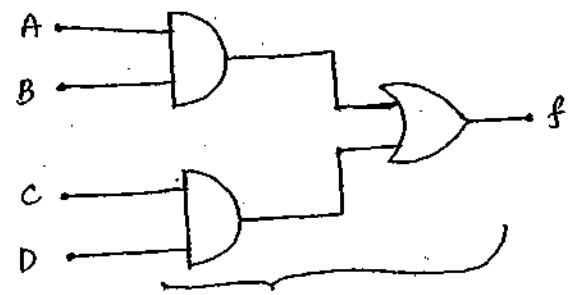




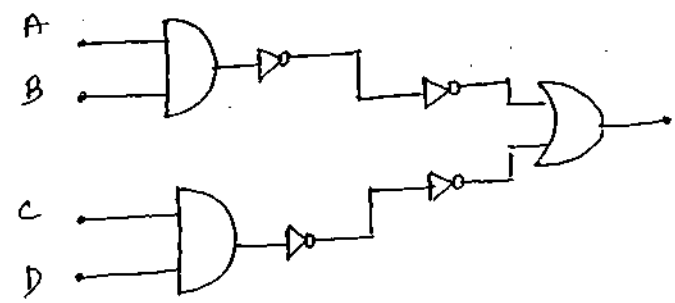
∴ NOR gate is a universal gate.

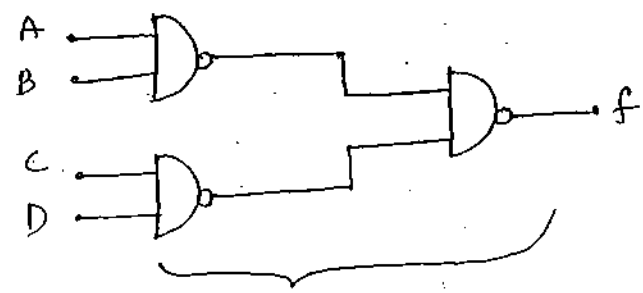


Ex:- AB + CD = f. Implement f using logic gates.



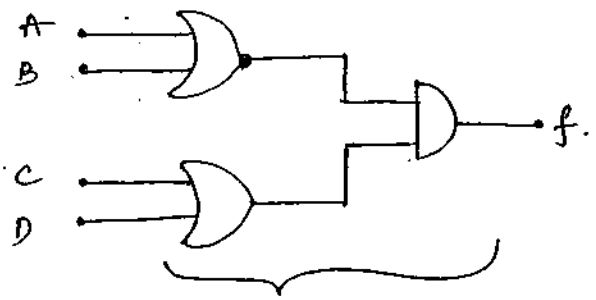
Two-level
AND-OR implementation



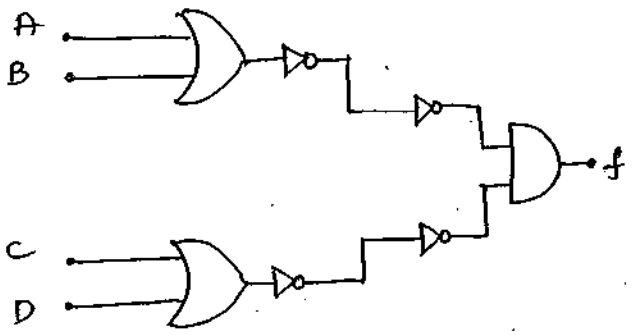


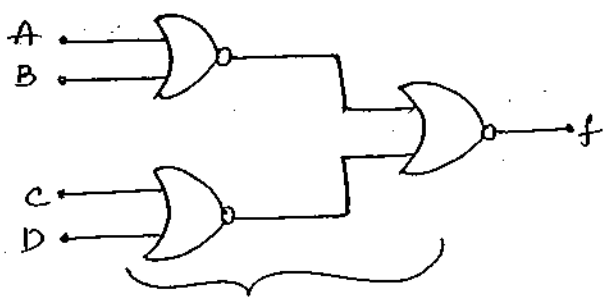
two-level
NAND-NAND implementation.

Ex:- Implement $f = (A+B)(C+D)$.



2-level
OR-AND implementation

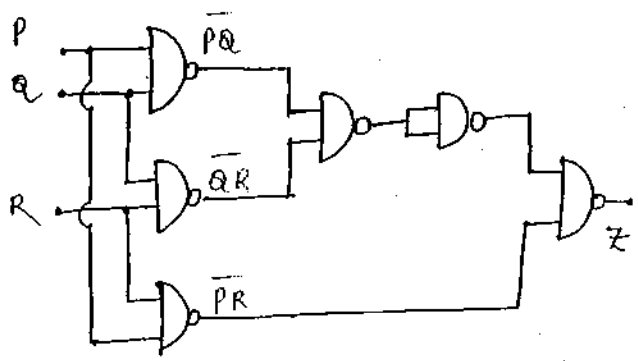




2-level
NOR-NOR implementation

∴ SOP form can be implemented with 2-level NAND-NAND.

POS form can be implemented with 2-level NOR-NOR.

Ex:- 

The o/p Z will be 1 in the ckt for

(a) two or more of the i/p P, Q, R are zero.

(b) two or more of the i/p's of P, Q, R are one.

(c) odd no. of i/p's of P, Q, R is zero.

(d) odd no. of i/p's of P, Q, R is one.

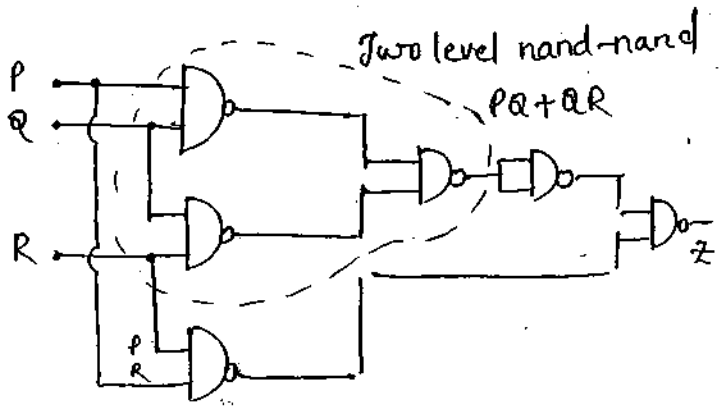
Sol:-

$$Z = \overline{\overline{PQ} \cdot \overline{QR} \cdot \overline{PR}}$$

$$Z = \overline{\overline{PQ} \cdot \overline{QR} \cdot \overline{PR}}$$

$$Z = PQ + QR + PR.$$

$$1 = PQ + QR + PR.$$



$Z = PQ + QR + PR.$

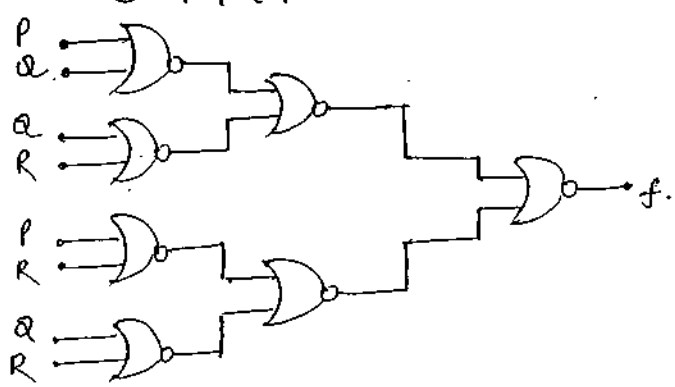
Ex:- The o/p 'f' in the ckt

(a) $\overline{P+Q+R}$

(b) $\overline{Q+R}$

(c) $\overline{P+R}$

(d) $\overline{P} + \overline{Q} + \overline{R}$.



$$f = \overline{(P+Q)(Q+R) + (P+R)(Q+R)}$$

Two-level NOR-NOR given ⇓ OR-AND.

$$f = \left(\overline{P+Q} + \overline{Q+R}\right) \cdot \left(\overline{P+R} + \overline{Q+R}\right).$$

$$f = \left(\overline{P+Q} \cdot \overline{P+R}\right) + \overline{Q+R}.$$

$$f = \left(\overline{P} \cdot \overline{Q} \cdot \overline{P} \cdot \overline{R}\right) + \overline{Q} \cdot \overline{R}.$$

$$f = \overline{Q} \cdot \overline{R}\left[\overline{P}+1\right] = \overline{Q} \cdot \overline{R} = \overline{Q+R}.$$

(or)

$$f = \overline{(P+Q)(Q+R) + (P+R)(Q+R)} = \overline{(Q+R)(P+Q+R)}$$

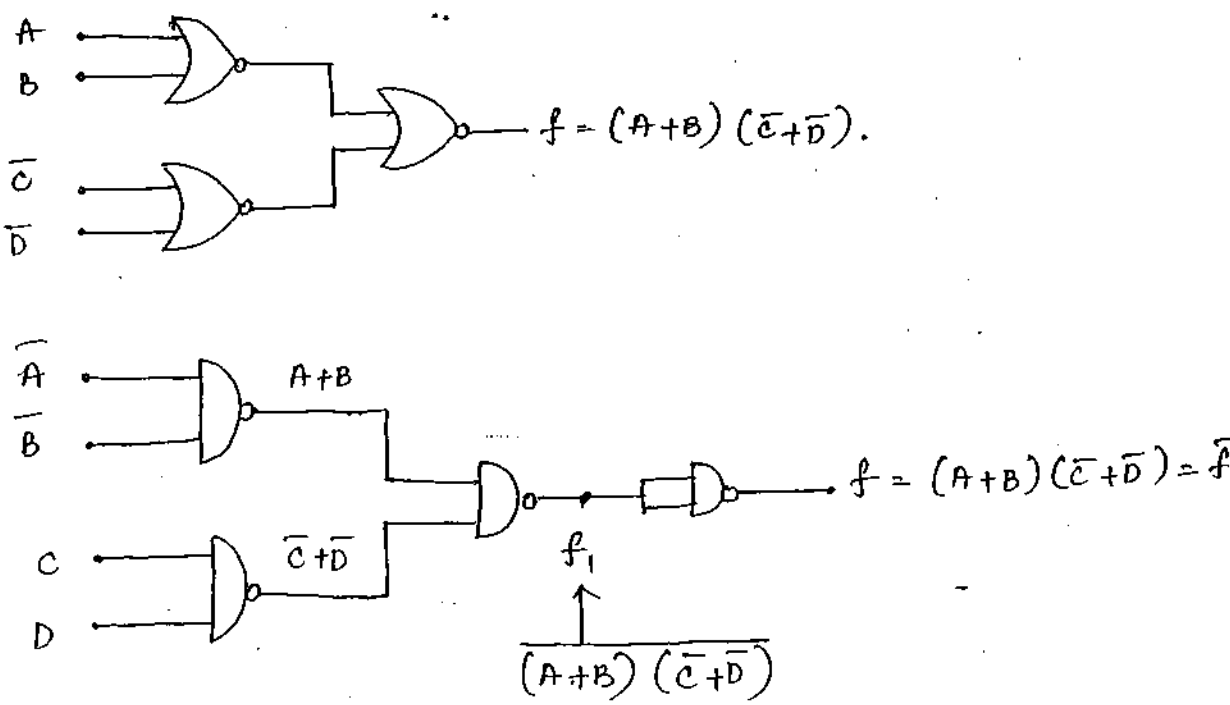
$$= \overline{Q+R} \quad \because x(x+y) = x$$

Ex:- Find minimum no. of 2-i/p NAND gates required to implement the function $(A+B)(\overline{C}+\overline{D})$.

[Assume that variables are available in both complemented and uncomplemented form] → literals $(A, \overline{A}, B, \overline{B}, C, \overline{C})$ but variables are A, B, C only

$f = (A+B)(\overline{C}+\overline{D})$.

→ two i/p nor-nor implementation.

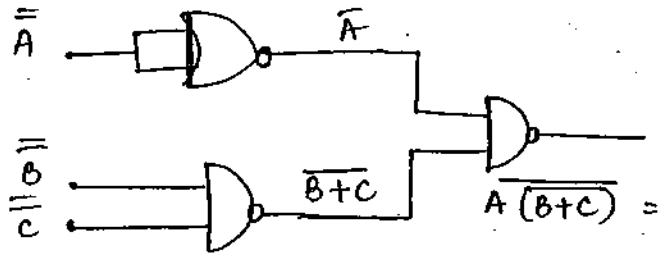


∴ Minimum 4 nand-gates required.

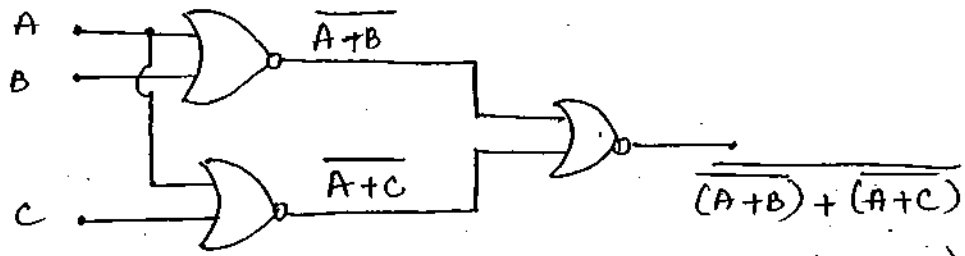
Ex:- min. no. of 2-ilp nor gates required to implement $f = A + BC$.

Sol:- $f = A + BC$.

$f = AA + BC$.



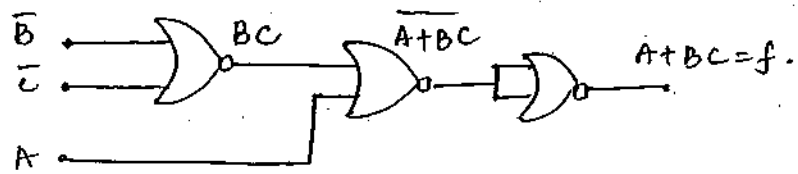
$f = AA + BC$.



$\Rightarrow (A+B)(A+C)$

$\Rightarrow A + BC$.

(or)



Ex:- find min no. of NAND gates require to implement $A + AB + A\overline{B}C$.

$f = A + AB + A\overline{B}C$.

$f = A(1 + B + \overline{B}C) = A$.

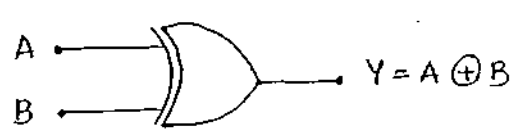
$\therefore$ No NAND gate is required to implement A.

Ex:- 2-ilp

EX-OR gate :-

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

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



$y = \bar{0}\cdot x + 0\cdot\bar{x}$

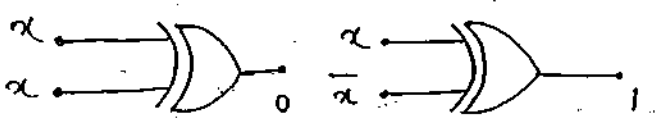
$y = x.$

$y = \bar{1}\cdot x + 1\cdot\bar{x}$

$y = \bar{x}.$

∴ for Ex-or there is no disable i/p. Both '0' and '1' are enable i/p. So Ex-OR gate cannot be used for enabling a circuit or gate unlike basic gates AND and OR.

2-i/p Ex-OR gate can be used to check unequality of inputs.



3-i/p Ex-or gate :-

	A	B	C	$Y = A \oplus B \oplus C$
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1

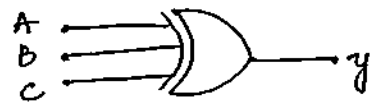
The o/p of gate will be 1, when the i/p is having odd number of one's.

$y = \sum m(1,2,4,7) = \pi M(0,3,5,6)$

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

↳ SOP minterms.

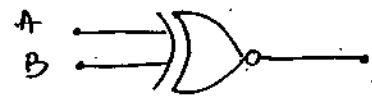
$Y = (A+B+C)(A+\bar{B}+\bar{C}) + (\bar{A}+B+\bar{C})(\bar{A}+\bar{B}+C).$



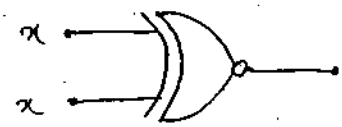
EX-NOR GATE :-

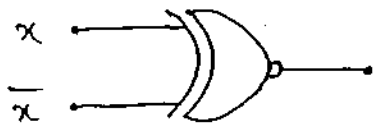
A	B	$Y = A\odot B = \bar{A}\bar{B} + AB$
0	0	1
0	1	0
1	0	0
1	1	1

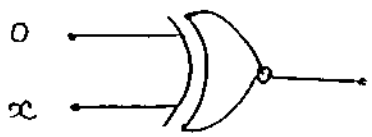
$= (\bar{A}+B)(A+\bar{B})$

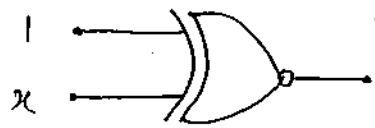


2-i/p EX-NOR gate can be used to check equality of inputs.









Ex-NOR gate cannot be used for enabling and disabling the block.

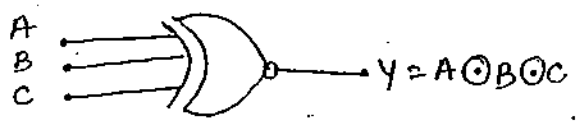
3 i/p Ex-NOR gate :-

	A	B	C	Y = A⊙B
0	0	0	0	→ 1
1	0	0	1	→ 0
2	0	1	0	→ 0
3	0	1	1	→ 1
4	1	0	0	→ 0
5	1	0	1	→ 1
6	1	1	0	→ 1
7	1	1	1	→ 0

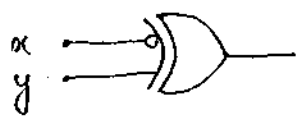
$Y = \sum m(0,3,5,6) = \pi M(1,2,4,7)$

$Y = \bar{A}\bar{B}\bar{C} + \bar{A}BC + A\bar{B}C + AB\bar{C}$.

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



Multi-i/p Ex-NOR gate o/p is 1 when even no. of one's present in i/p.

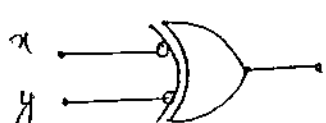


$y = \bar{A}B + A\bar{B}$

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

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

$y = x \odot y$.

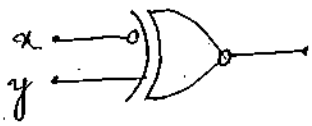


$y = \bar{A}B + A\bar{B}$

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

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

$= x \oplus y$.

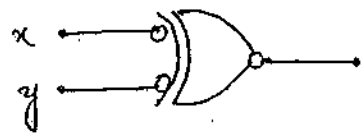


$y = \bar{A}B + A\bar{B}$

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

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

$= x \oplus y$.



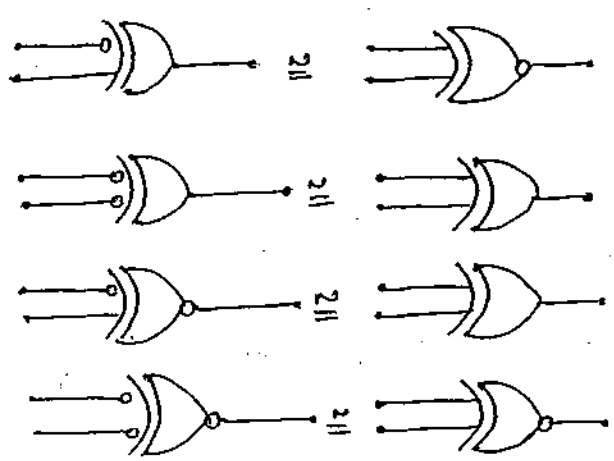
$y = \bar{A}\bar{B} + AB$

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

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

$= x \odot y$.

$\therefore$



$A \oplus B = \bar{A}B + A\bar{B} = \overline{A \odot B} = \overline{\bar{A}\bar{B} + AB}$.

$A \odot B = \bar{A}\bar{B} + AB = \overline{A \oplus B} = \overline{\bar{A}B + A\bar{B}}$.

$A \oplus B \oplus C = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC$.

$A \odot B \odot C = \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}C + AB\bar{C}$.

$(A \odot B) \odot C = \underbrace{(\bar{A}\bar{B} + AB)}_{P} \odot C$.

$= P \odot C = \bar{P}\bar{C} + PC$

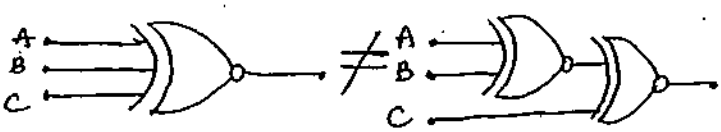
$= \overline{\bar{A}\bar{B} + AB}\,\bar{C} + (\bar{A}\bar{B} + AB)C$

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

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

$= A \oplus B \oplus C$

$\neq A \odot B \odot C.$



$(A \oplus B) \oplus C = \underbrace{(\bar{A}B + A\bar{B})}_{P} \oplus C$

$= P \oplus C$

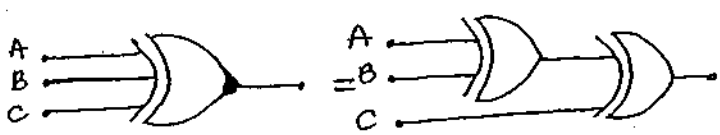
$= \bar{P}C + P\bar{C}$

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

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

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

$= A \oplus B \oplus C$



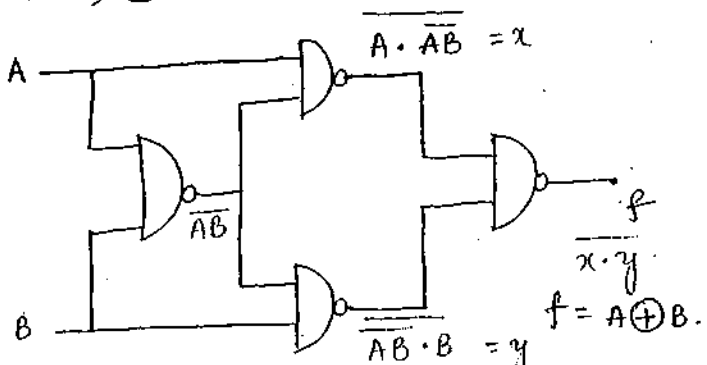
$(A \oplus B) \oplus A = A \oplus B \oplus A.$

$= A \oplus A \oplus B.$

$= 0 \oplus B$

$= B.$

$(A \oplus B) \oplus B = A.$



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

$= \bar{A} + AB$

$= \bar{A} + B.$

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

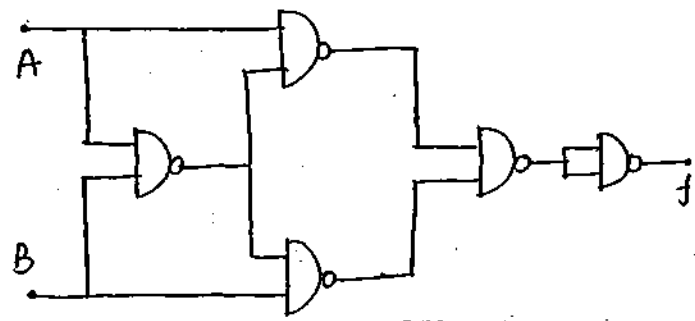
$= AB + \bar{B}$

$= A + \bar{B}.$

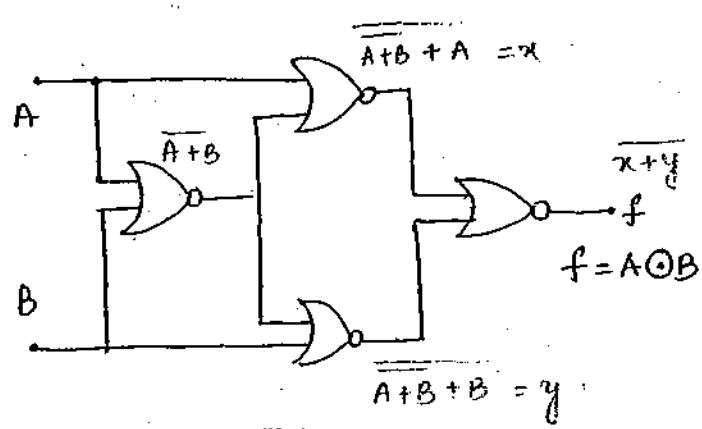
$f = \overline{x \cdot y} = \overline{(\bar{A} + B)(A + \bar{B})}$

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

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



$f = \overline{A \oplus B} = A \odot B.$



$x = \overline{\overline{A+B} + A}$

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

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

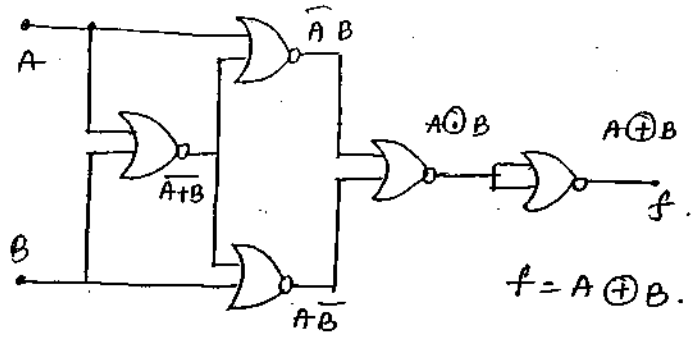
$y = \overline{\overline{A+B} + B}$

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

$f = \overline{x+y} = \overline{\bar{A}B + A\bar{B}}$

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

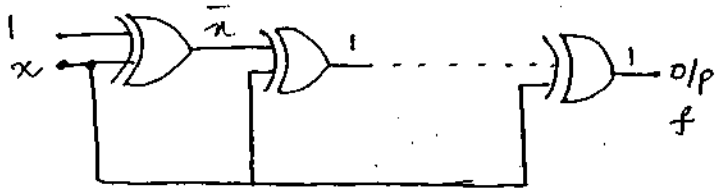
$= A \odot B$



$f = A \oplus B$.

Ex:- If 20 EX-OR gates are connected in cascade as shown in figure. Then o/p 'f' is ___.

(a) 0 (b) 1 (c) x (d) $\bar{x}$.



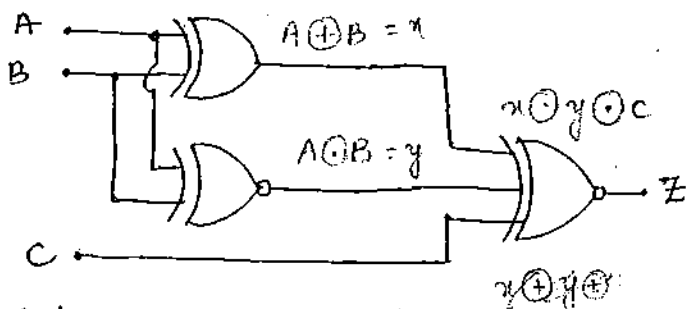
$1 \oplus x = \bar{1}x + 1\bar{x} = \bar{x}$.

$\bar{x} \oplus x = \bar{x}\bar{x} + \bar{\bar{x}}x = 1$.

$\therefore f = 1 \longrightarrow$ as there are even number of EX-OR gates.

Ex:- The i/p combinations A,B,C for which o/p z = 1

(a) 0 1 0 (b) 1 0 0 (c) 1 1 0 (d) 0 0 1.



Sol:- $Z = x \odot y \odot C$

$1 = (A \oplus B) \odot (A \odot B) \odot C$.

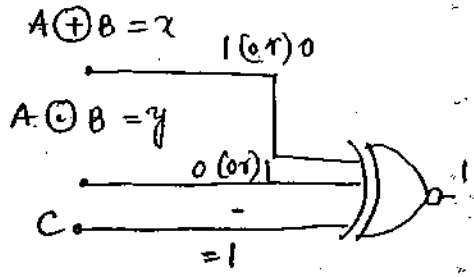
x for A.

$1 = \left[(\bar{A}B + A\bar{B}) \odot (A \odot B) \odot C\right]$

for $A \odot B \odot C$.

$(1 = \bar{A}\bar{B}\bar{C} + \bar{A}BC + A\bar{B}C + ABC)$

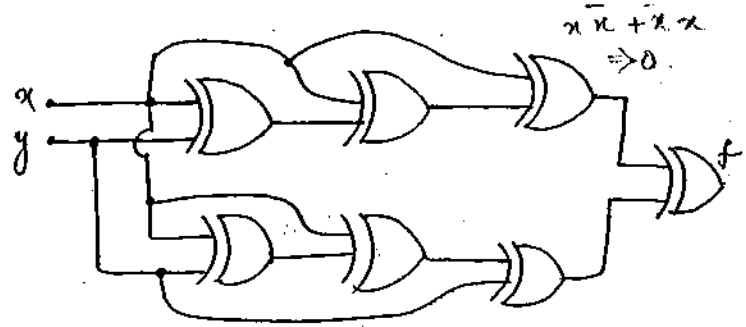
for



$\therefore ABC = (101)$ or (011)

Ex:- The o/p 'f' in the circuit $0\bar{y} + 1y$

(a) 0 (b) 1 (c) $x \oplus y$ (d) $x \odot y$.



Sol:-

$x \oplus y \oplus x \oplus x \oplus x \oplus y \oplus x \oplus y$

$\Rightarrow x \oplus y \oplus 0 \oplus x \oplus y \oplus y \oplus x$

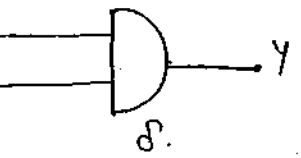
$\Rightarrow x \oplus y \oplus 0$

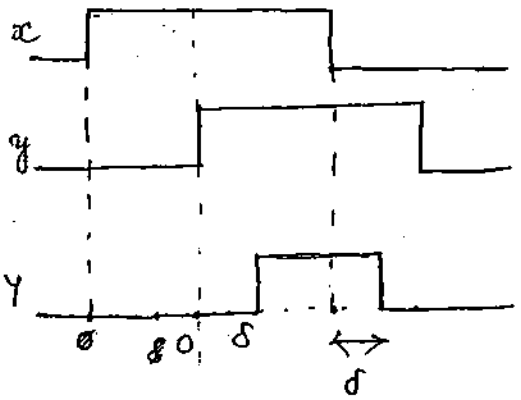
$\Rightarrow x \oplus y$.

PROPAGATION DELAY OF LOGIC GATES :-

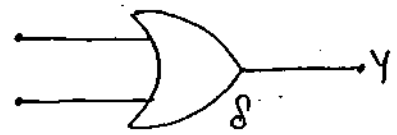
If there is a change in o/p due to changes in the i/p, the change will occur in the o/p after propagation delay time from the point inputs changed.

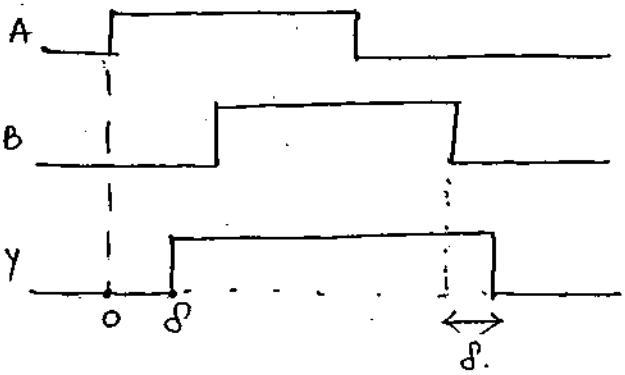
Ex:- x
=
y





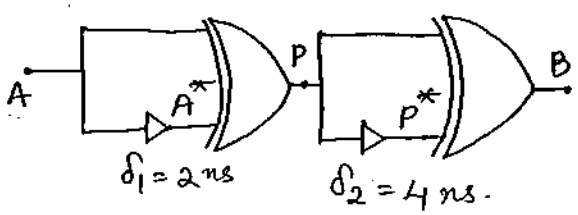
Ex:-

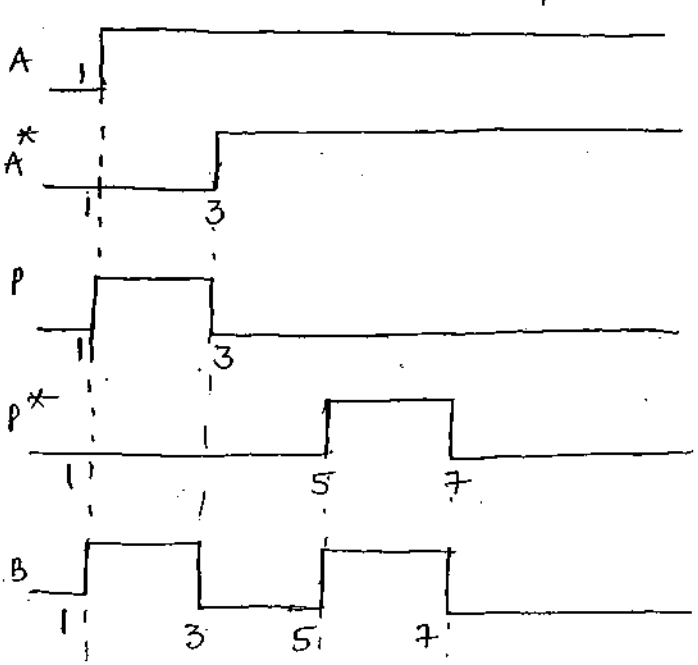




* Q.5 pgno.35

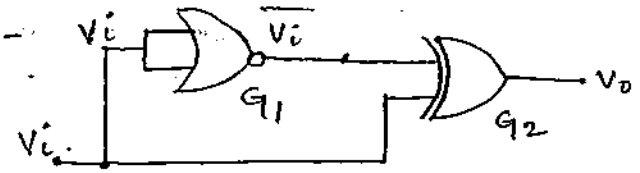
Sol:-



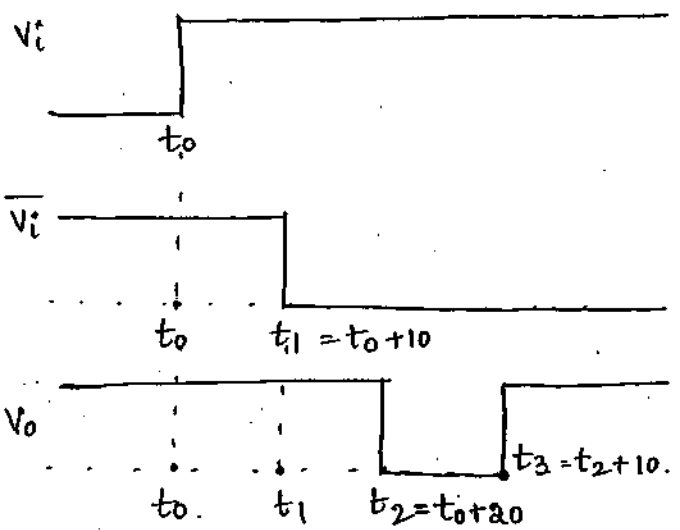


* Q.37 level-1 pg-35

Sol:-



G1 → 10nsec and G2 → 20nsec

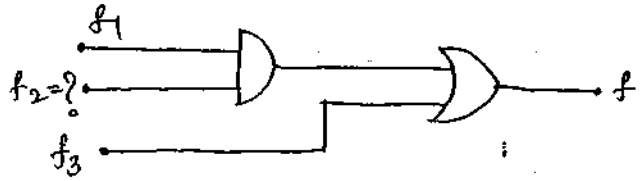


* Q.4 pgno.35 level2

Sol:- $f_1 = \sum m\ (4,5,6,7,8)$

$f_3 = \sum m\ (1,6,15)$

$f = \sum m\ (1,6,8,15)$.



f_2 contain must 8 and also may contain (1,6,15). It should not contain 4.

∴ from options $f_2 = \sum m(6,8$

* Q.1 pgno.35 1-2

Sol:- By assuming the result is also in 2's complement form,

x, y → sign bits of inputs.

z → sign bit of output.

Overflow doesn't occur if the two numbers are in the range (-8 to 7) during addition (if one is -ve and other is +ve).

If one number is +ve and other number is -ve, then overflow doesn't occur.

If two +ve numbers or two -ve numbers, there is a possibility to occur overflow.

x	y	z	f
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

$$-3 + (-5) = -8$$

x: 1 0 1 1
y: 1 0 1 1
1 0 0 0

$\Rightarrow$ 1 0 0 0

z

$$-4 + (-5) = -9$$

x: 1 1 0 0
y: 1 0 1 1
0 1 1 1

z $\rightarrow$ overflow.

$$+3 + (+4) = +7$$

x: 0 0 1 1
y: 0 1 0 0
0 1 1 1

z

$$+4 + (+5) = +9$$

x: 0 1 0 0
y: 0 1 0 1
1 0 0 1

z $\rightarrow$ overflow

$\therefore f = xy\bar{z} + \bar{x}\bar{y}z.$

K-MAP

→ using K-MAP, the no. of logic gates used decreases and no. of inputs required may also decrease.

Ex:- $A\bar{B}C + ABC + A\bar{B}\bar{C} + \bar{A}BC + AB\bar{C}$.

$\Rightarrow AC(\bar{B}+B) + A\bar{C}(\bar{B}+B) + \bar{A}BC$.

$\Rightarrow AC + A\bar{C} + \bar{A}BC$

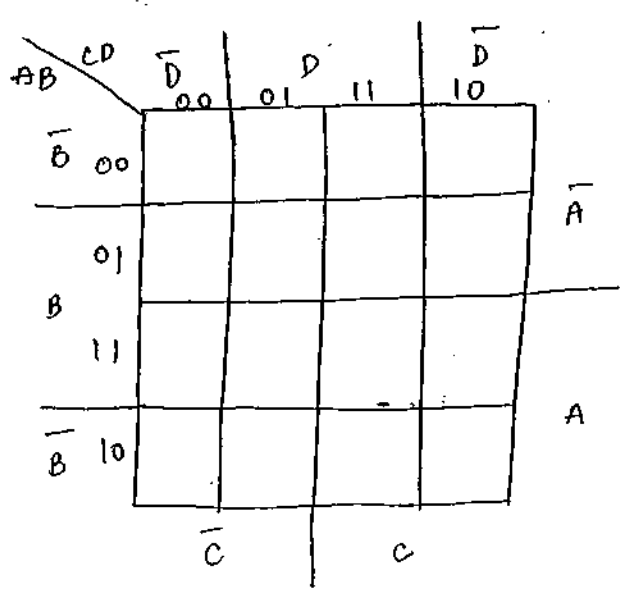
$\Rightarrow A(C+\bar{C}) + \bar{A}BC = A + \bar{A}BC$

$= A + BC$.

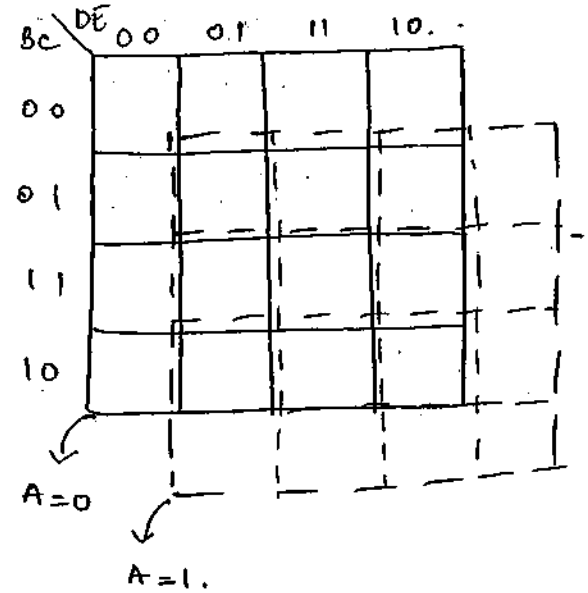
→ In any number of variables, EX-OR and EX-NOR cannot be minimised.

→ In 'n' variable K-map grouping 2^m number of adjacent cubes can eliminate n-variables and the resultant term will be of (n-m) variables.

4-Variable K-map :-



5-Variable K-map :-



Prime implicant :-

The smallest possible product term which cannot be reduced further is called prime implicant.

The smallest possible gate count is called minimal expression.

gate count = Total no. of gates use
Total no. of inputs required.

Ex:- $AB + BC + \bar{A}\bar{C}$

↓ ↓ ↓

2 i/ps 2 i/ps 2 i/ps.

AND AND AND

OR gate with 3 i/p's.

gate count = 4+9 = 13.

Essential prime implicant :-

If a prime implicant coverir atleast one minterm which is not covered by other implicants, ther that prime implicant is called

Essential prime implicant.

Redundant prime implicant :-

The prime implicant which is not present in the minimal expression is called redundant prime implicant.

Ex:- $\bar{A}\bar{B}C + \bar{A}BC + ABC + AB\bar{C}$.

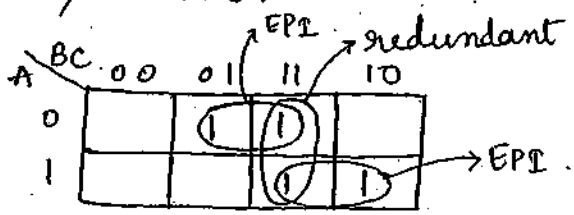
1 3 7 6

$\bar{A}C$ (1,3) → EPI

BC (3,7) → NEPI

AB (6,7) → EPI.

$\Rightarrow \bar{A}C + AB$.

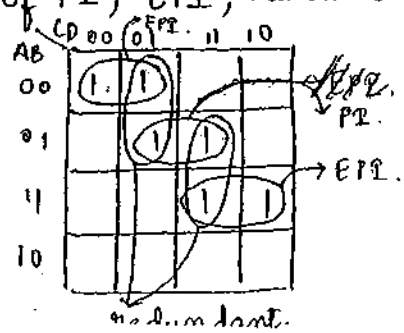


In K-Map grouping, for prime implicants in a grouping if there is atleast one cube which is not covered by any other groupings. Then it is called Essential prime implicant.

Ex:- Consensus theorem,

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

Ex:- Find no. of PI, EPI, minimal expression.



(a) PI → 5

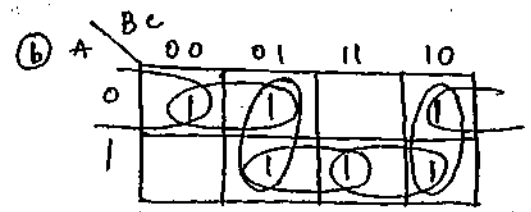
EPI → 2

minimal expression →

$\bar{A}\bar{B}\bar{C} + \bar{A}BD + ABC$.

To cover minterms which are not covered by essential prime implicants we need to consider some of non-essential PI in the minimal expression with minimum count.

(b)



P.I → 6.

E.P.I → 0.

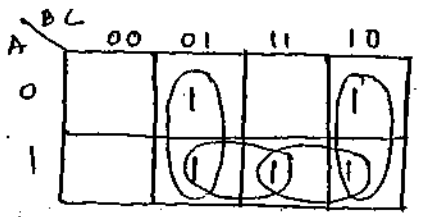
minimal expression →

$\bar{A}\bar{B} + AC + AB\bar{C}$.

(or)

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

(c) Find PI, EPI, ME.



PI → 4.

EPI → 2.

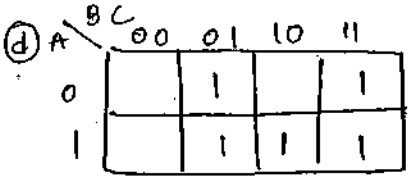
minimal expression :-

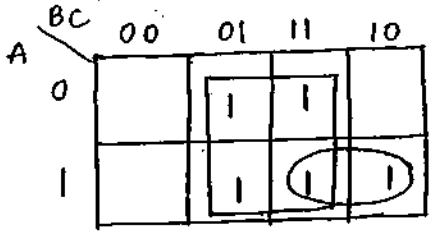
$\bar{B}C + AC + B\bar{C}$

(or)

$\bar{B}C + AB + B\bar{C}$.

(d)



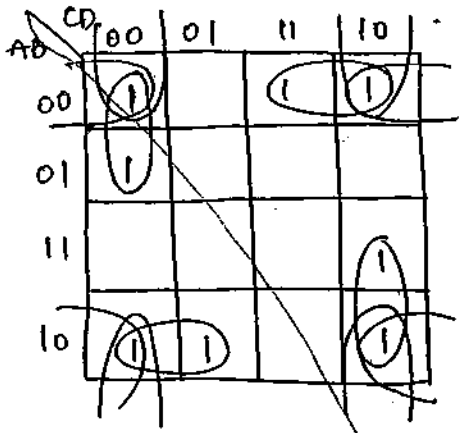


PI → 2.

EPI → 2.

minimal expression → C + AB.

(e)

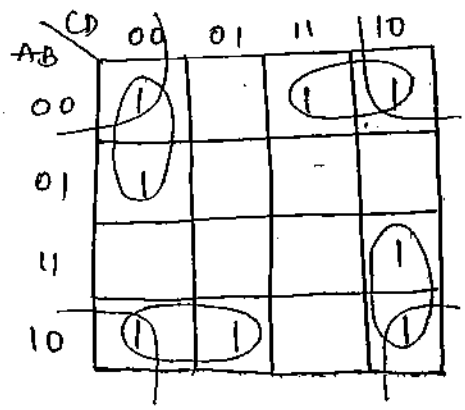


P.I → 8

EPI → 4

minimal expression →

$\bar{A}\bar{C}\bar{D} + \bar{A}\bar{B}C + A\bar{B}\bar{C} + AC\bar{D}$.



This quad cannot be a essential prime implicant as there is possibility.

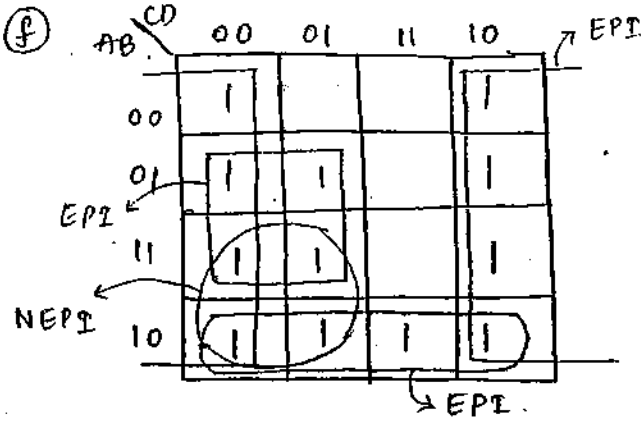
P.I → 5

E.P.I → 4.

minimal expression →

$\bar{A}\bar{C}\bar{D} + A\bar{B}\bar{C} + AC\bar{D} + \bar{A}\bar{B}C$

(f)



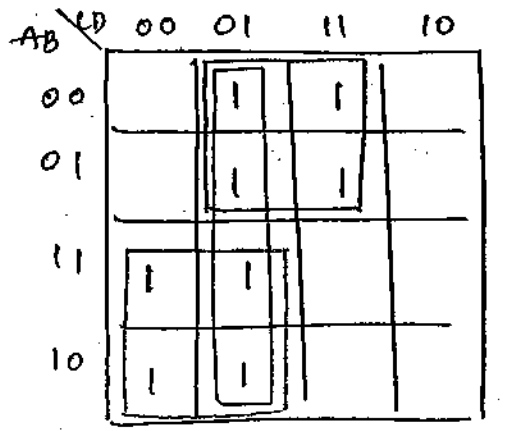
P.I → 3+1 = 4.

EPI → 3

Minimal expression →

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

(g) $f(A,B,C,D) = \sum m(1, 3, 5, 7, 8, 9, 12, 1$



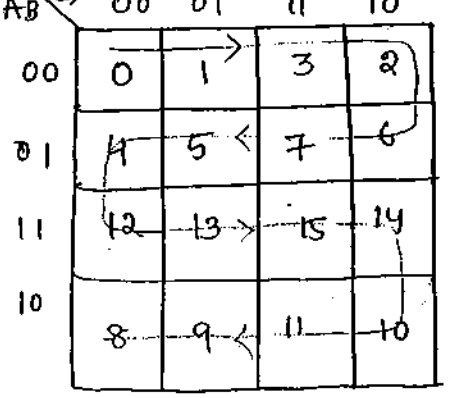
P.I → 3

EPI → 2.

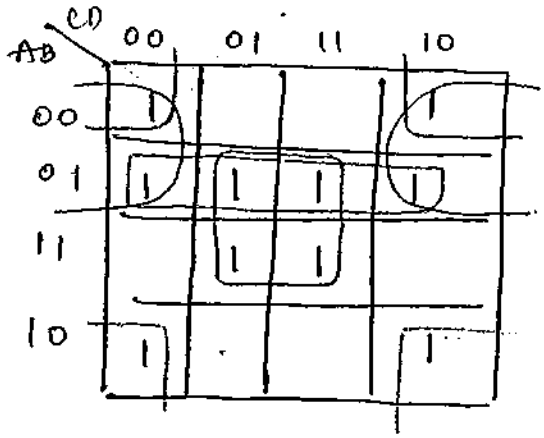
Minimal expression →

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

The code for gray codes is



(h) $f(A,B,C,D) = \{0,2,4,5,6,7,8,10,13,15\}$.



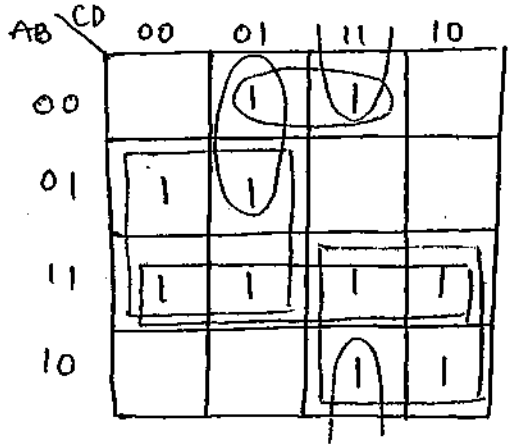
PI → 3+1 = 4.

EPI → 2.

Minimal expression →

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

(i) $f(A,B,C,D) = \sum m(1,3,4,5,10,11,12,13,14,15)$.



P.I → 6

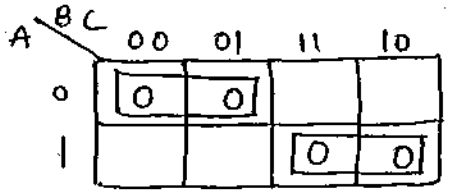
EPI → 2

Minimal expression →

$B\bar{C} + AC + \bar{A}\bar{B}D$.

Minimal expression in POS :-

Ex:- Find minimal expression.

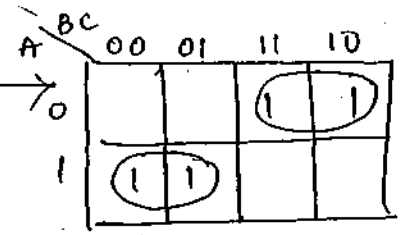


minimal expression in POS →

$f = (A+B)(\bar{A}+\bar{B}) = 0 + A\bar{B} + \bar{A}B + 0$

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

Minimal expression in SOP →

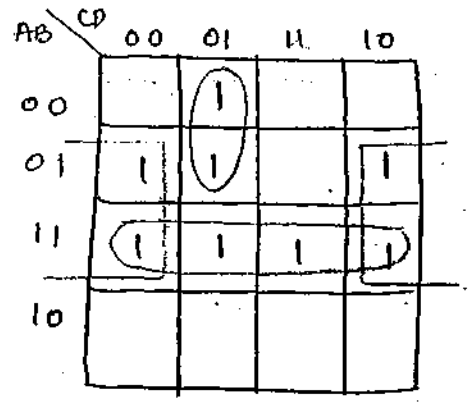


$f = A\bar{B} + \bar{A}B$.

SOP ≅ POS.

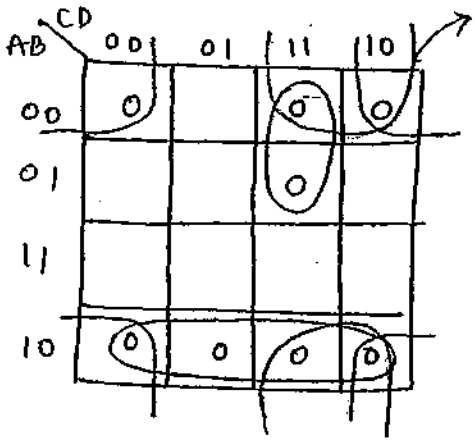
Ex:- Find minimal expression in SOP and POS for the fn

$f(A,B,C,D) = \sum m(1,4,5,6,12,13,14,15)$.



minimal expression in SOP →

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



minimal expression in POS →

$\bar{B}\bar{D} + \bar{A}CD + A\bar{B}$,

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

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

$= \bar{A}B\bar{C} + \underline{\bar{A}B\bar{D}} + \bar{A}\bar{C}D + \underline{AB} + \underline{B\bar{C}} + \underline{B\bar{D}} + \underline{ADB} + \underline{B\bar{C}}D$.

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

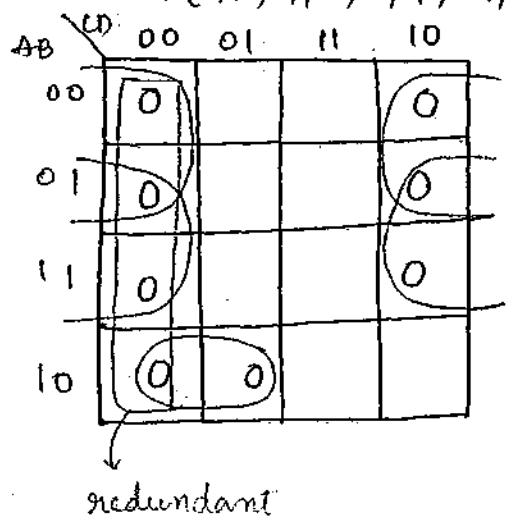
$= B(A+\bar{C}) + B\bar{C} + B\bar{D} + \bar{A}\bar{C}D$.

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

($B\bar{C}$ ↓ redundant)

Ex:- Find minimal expression for SOP and POS.

$\sum M(0,2,4,6,8,9,12,14)$.



minimal expression in POS →

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

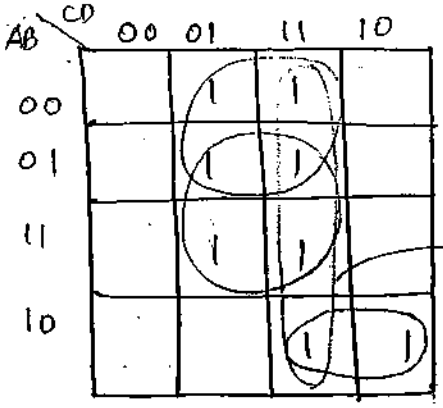
$\Rightarrow (A\bar{B} + AD + \bar{B}D + D)(\bar{A}+B+C)$

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

$\Rightarrow \bar{A}D + BD + A\bar{B}C + CD$.

(CD ↓ redundant.)

$\Rightarrow$

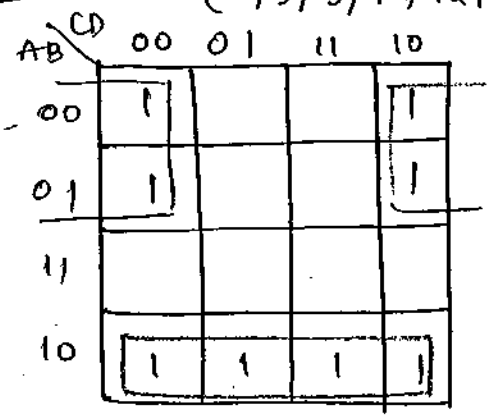


The minimal expression is

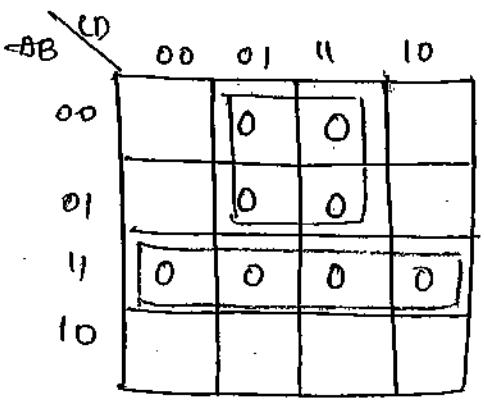
$(\bar{A}D) + BD + A\bar{B}C$.

→ redundant

Ex:- $\pi M(1,3,5,7,12,13,14,15)$.



SOP → $\bar{A}\bar{D} + A\bar{B}$



POS → $(\bar{A}+\bar{B})(A+\bar{D})$.

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

$= \bar{A}\bar{D} + A\bar{B}$ → Consensus theorem.

Incompletely specified function (Don't care) :-

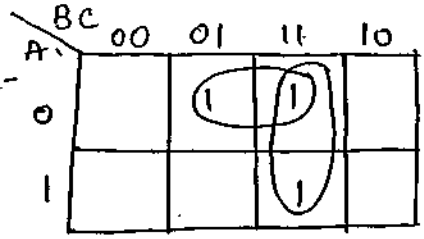
Function is defined as one for some combinations, defined as zero for some other combinations, for remaining combinations of i/p, o/p is not specified, means we are not expecting the o/p for these i/p combinations. We dont care what system gives the o/p for these combinations which are called don't cares. If dont cares are helping in minimising the expression, take the help of don't cares else don't care about don'tcares. We dont need to cover all the don't cares.

NOTE :-

If a grouping is cover in K-MAP is covering extra only

dontcares compare to other groupings which we are considering i the minimal expression then we don't need to consider this grouping in the minimal expression.

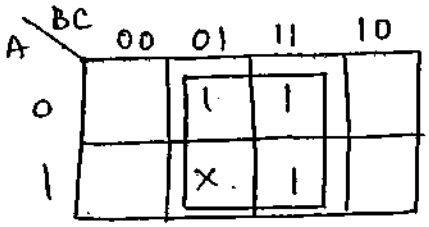
Ex:-



$f = \bar{A}C + BC.$

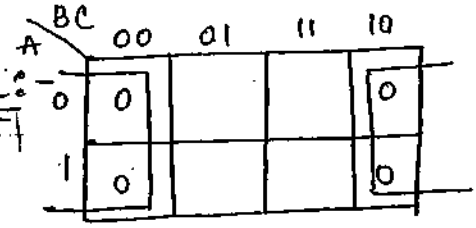
↓ function is reduced.

$f = C.$

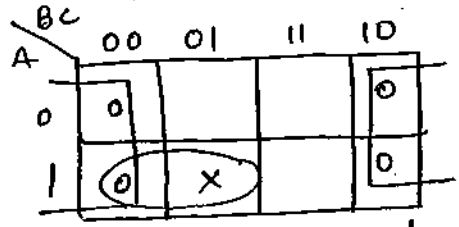


In this example, by considering the dontcare, the function is minimised. So consideration of dontcare is necessary.

Ex:-



$f = C.$



If we consider dontcare

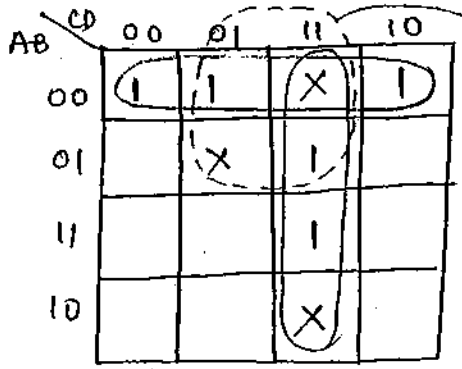
$f = C(\bar{A}+B).$

After considering the don't care, the function expanded. So dontcare should not be considered in this case.

Ex:- Find the minimal expression in SOP and POS for the function

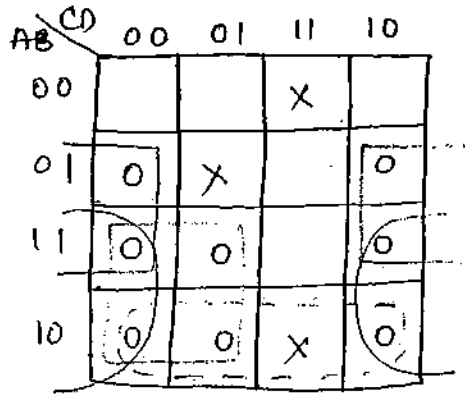
$F(A,B,C,D) = \sum m(0,1,2,7,15) + X(3,5,11).$

Sol:-



This not required as it is not involving extra minterms.

$F = \bar{A}\bar{B} + CD.$



By not considering any dontcares,

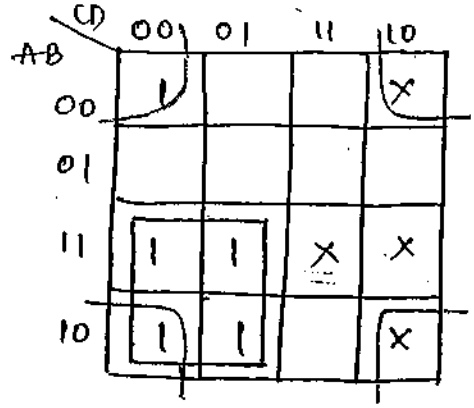
$F = (\bar{A}+C)(\bar{B}+D)(\bar{A}+D).$

By considering dontcare,

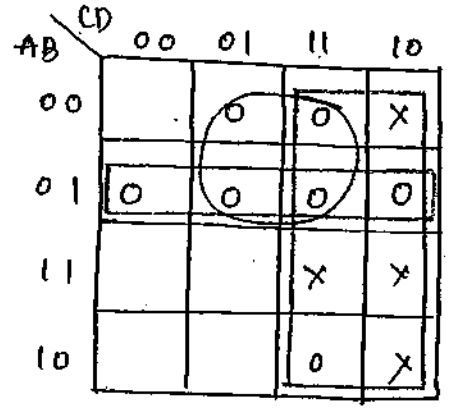
$F = (\bar{B}+D)(\bar{A}+C)(\bar{A}+B).$

Ex:- $F(A,B,C,D) = \sum m(0,8,9,12,13) + X(2,10,14,15)$.

Sol:-



$F = \bar{B}\bar{D} + A\bar{C}$.



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

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

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

$= A\bar{C} + \bar{B}\bar{C}\bar{D}$.

Functionality SOP may not be equal to POS in the presence of dontcares. If dontcares all are used in one form and not used in other form, then functionality will be equal.

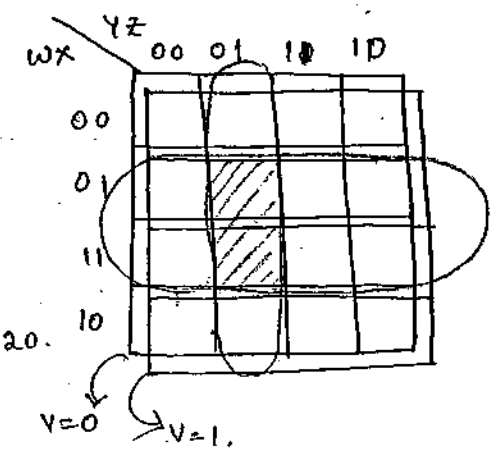
* Q.3 Pgno.35

Sol:- $f(v,w,x,y,z) = x + \bar{y}z$.

$= 2^4 + 2^3$

$= 16 + 8 = 24 - 4 = 20$.

(or) $2^4 + 2^3 - 2^2 = 16 + 8 - 4 = 20$.



No. of minterms excluding redundant terms is

Ex:- $f(v,w,x,y,z) = w + x + \bar{y}z$

$= 2^4 + 2^4 + 2^3 - 2^3 - 2^2 - 2^2 + 2^1$

$= 16 + 16 + 8 - 8 - 4 - 4 + 2$

$= 34 - 8$

$= 24$.

Number of minterms excluding redundant terms is,

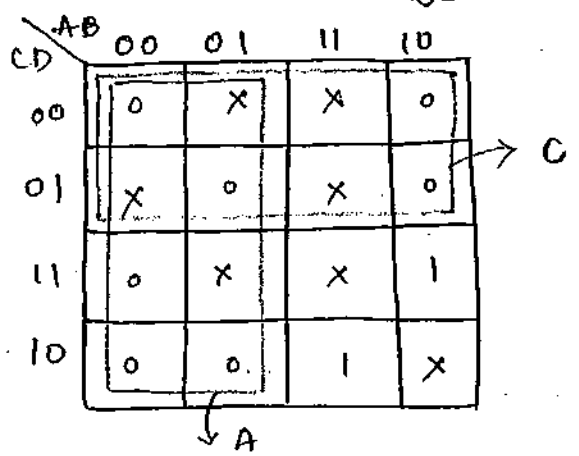
$$n(A \cup B) = n(A) + n(B) - n(A \cap B).$$

$$n(A \cup B \cup C) = n(A) + n(B) + n(C) - n(A \cap B) - n(B \cap C) - n(C \cap A) \; n(A \cap B \cap C).$$

Ex:- Find minimal expression in POS for F implemented in the K-map shown

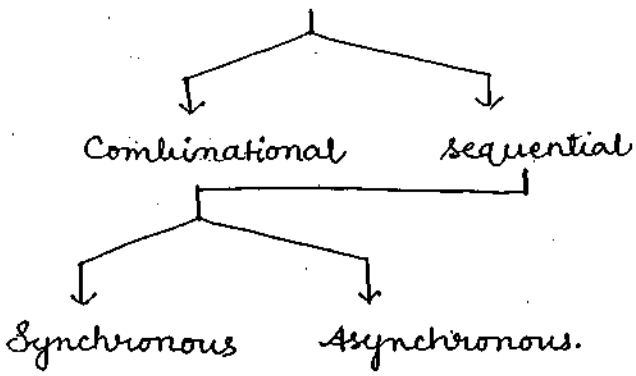
Sol:- $F(A, B, C, D) = ?$

(a) $A + C$ (b) $A \cdot C$ (c) $\overline{A} + \overline{C}$ (d) $\overline{A} \cdot \overline{C}$.



$F = A \cdot C$

DIGITAL CIRCUITS



The o/p depends on the combination of present inputs is called combinational circuit.

The o/p depends on the sequence of combination of i/p's. For sequence we need a memory element. It is called sequential circuit.

COMBINATIONAL CIRCUITS

It doesnot needs any memory element.

HALF ADDER :-

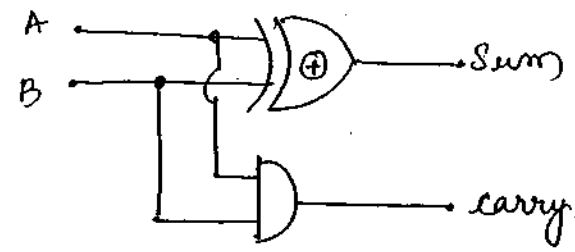
	A	B	S	Co
0	0	0	0	0
1	0	1	1	0
2	1	0	1	0
3	1	1	0	1

$S = \sum m(1,2)$

$C_0 = \sum m(3)$.

$S = \bar{A}B + A\bar{B} = A \oplus B$.

$C_0 = AB$.

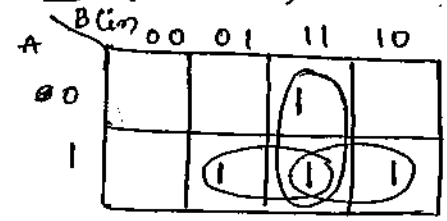


FULL ADDER :-

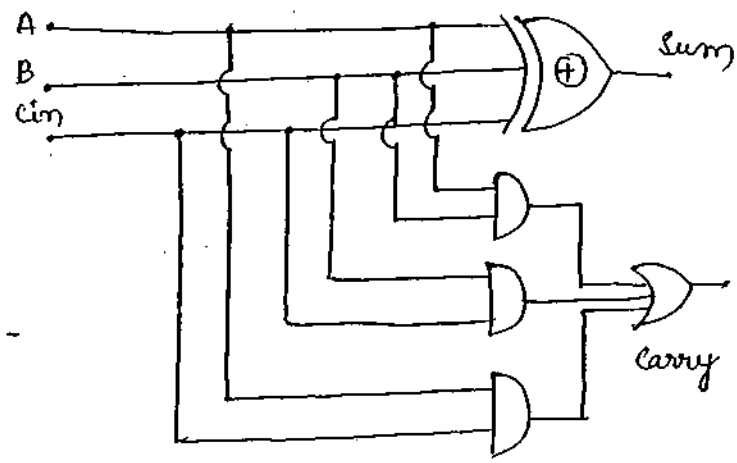
	A	B	Ci	S	Co
0	0	0	0	0	0
1	0	0	1	1	0
2	0	1	0	1	0
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	1
6	1	1	0	0	1
7	1	1	1	1	1

$S = \sum m(1,2,4,7) = A \oplus B \oplus Cin$.

$C_0 = \sum m(3,5,6,7)$.



$C_0 = ACin + AB + BCin$.

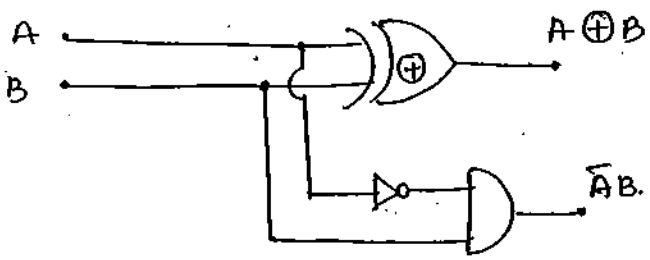


HALF SUBTRACTOR :-

	A	B	D	B_0
0	0	0	0	0
1	0	1	1	1
2	1	0	1	0
3	1	1	0	0

$D = \sum m(1,2) = A \oplus B$

$B_0 = \sum m(1) = \bar{A}B$.

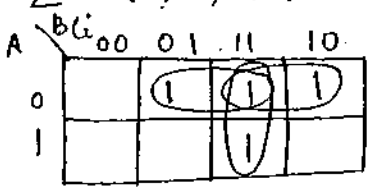


FULL SUBTRACTOR :-

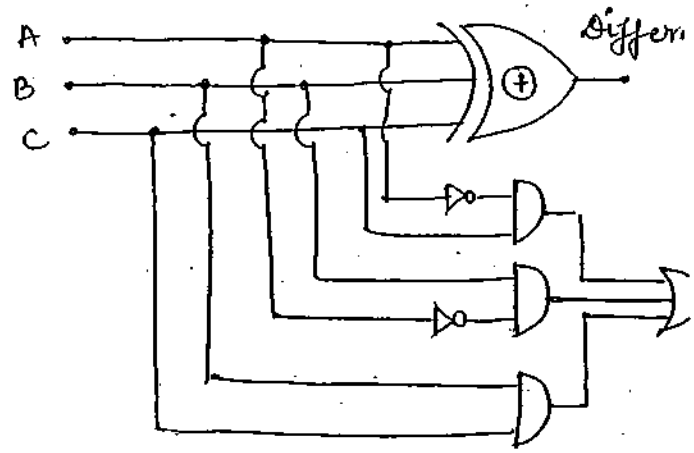
	A	B	Ci	D	B_0
0	0	0	0	0	0
1	0	0	1	1	1
2	0	1	0	1	1
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	0
6	1	1	0	0	0
7	1	1	1	1	1

$D = \sum m(1,2,4,7) = A \oplus B \oplus Ci$

$B_0 = \sum m(1,2,3,7)$.



$B_0 = \bar{A}Ci + \bar{A}B + BCi$.



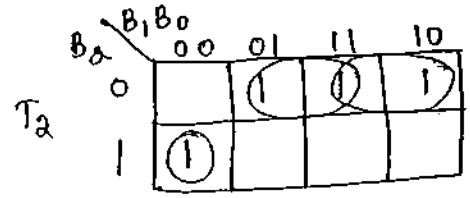
Ex:- Design a combinational ckt which is finding 8's complement of 3-bit binary.

	B_2	B_1	B_0	T_2	T_1	T_0
0	0	0	0	0	0	0
1	0	0	1	1	1	1
2	0	1	0	1	1	0
3	0	1	1	1	0	1
4	1	0	0	1	0	0
5	1	0	1	0	1	1
6	1	1	0	0	1	0
7	1	1	1	0	0	1.

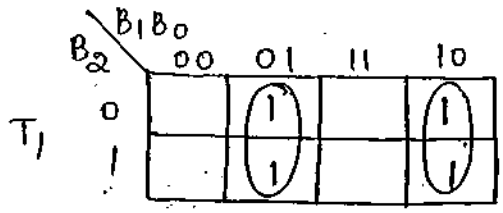
$T_2 = \sum m(1,2,3,4)$.

$T_1 = \sum m(1,2,5,6)$.

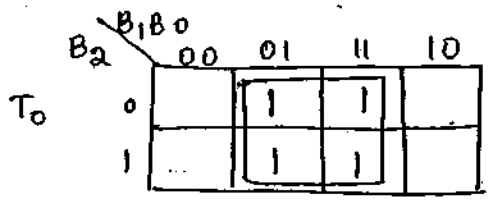
$T_0 = \sum m(1,3,5,7)$.



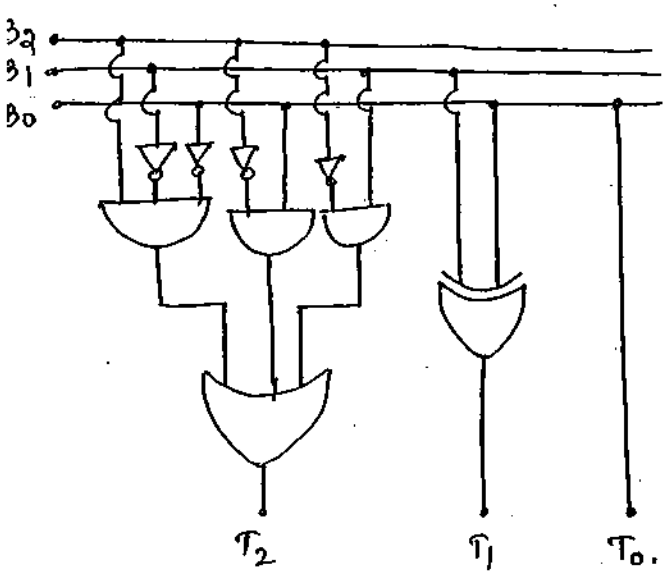
$T_2 = B_2\bar{B}_1\bar{B}_0 + \bar{B}_2B_0 + \bar{B}_2B_1$.



$T_1 = \bar{B_1} B_0 + B_1 \bar{B_0} = B_1 \oplus B_0$



$T_0 = B_0$.



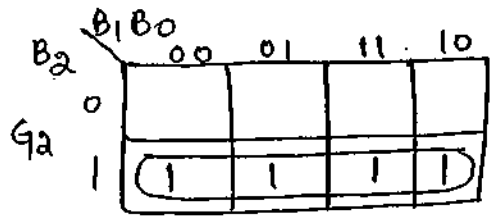
Eg:- Design a combinational ckt which converts 3-bit binary to gray.

	B_2	B_1	B_0	G_2	G_1	G_0	
0	0	0	0	0	0	0	0
1	0	0	1	0	0	1	1
2	0	1	0	0	1	1	3
3	0	1	1	0	1	0	2
4	1	0	0	1	1	0	6
5	1	0	1	1	1	1	7
6	1	1	0	1	0	1	5
7	1	1	1	1	0	0	4

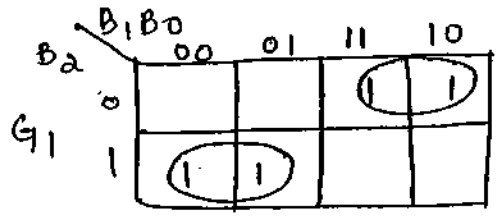
$G_2 = \sum m\,(4,5,6,7)$

$G_1 = \sum m\,(2,3,4,5)$:

$G_0 = \sum m\,(1,2,5,6)$.

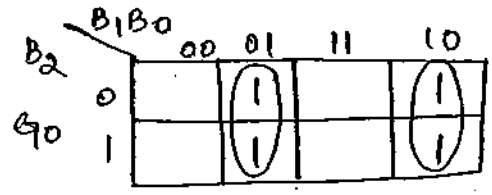


$G_2 = B_2$.



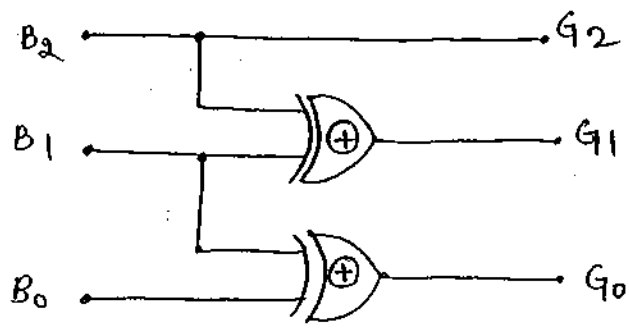
$G_1 = \bar{B_2} B_1 + B_2 \bar{B_1}$

$G_1 = B_2 \oplus B_1$.



$G_0 = \bar{B_1} B_0 + B_1 \bar{B_0}$

$G_0 = B_1 \oplus B_0$.



Ex:- Conversion of Gray to binary.

	G_2	G_1	G_0	B_2	B_1	B_0
0	0	0	0	0	0	0
1	0	0	1	0	0	1
3	0	1	1	0	1	0
2	0	1	0	0	1	1
6	1	1	0	1	0	0
7	1	1	1	1	0	1
5	1	0	1	1	1	0
4	1	0	0	1	1	1

$B_2 = \sum m\,(6,7,5,4)$.

$B_1 = \sum m\,(3,2,5,4)$.

$B_0 = \sum m\,(1,2,4,7)$.

$B_0 = A$

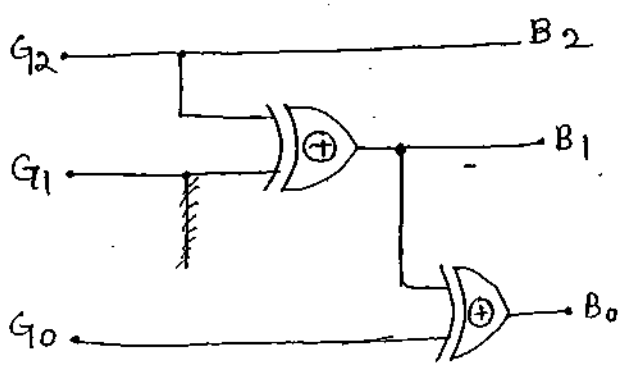
$B_2 = G_2$.

$B_1 = G_2 \oplus G_1$.

$B_0 = G_1 \oplus G_0$

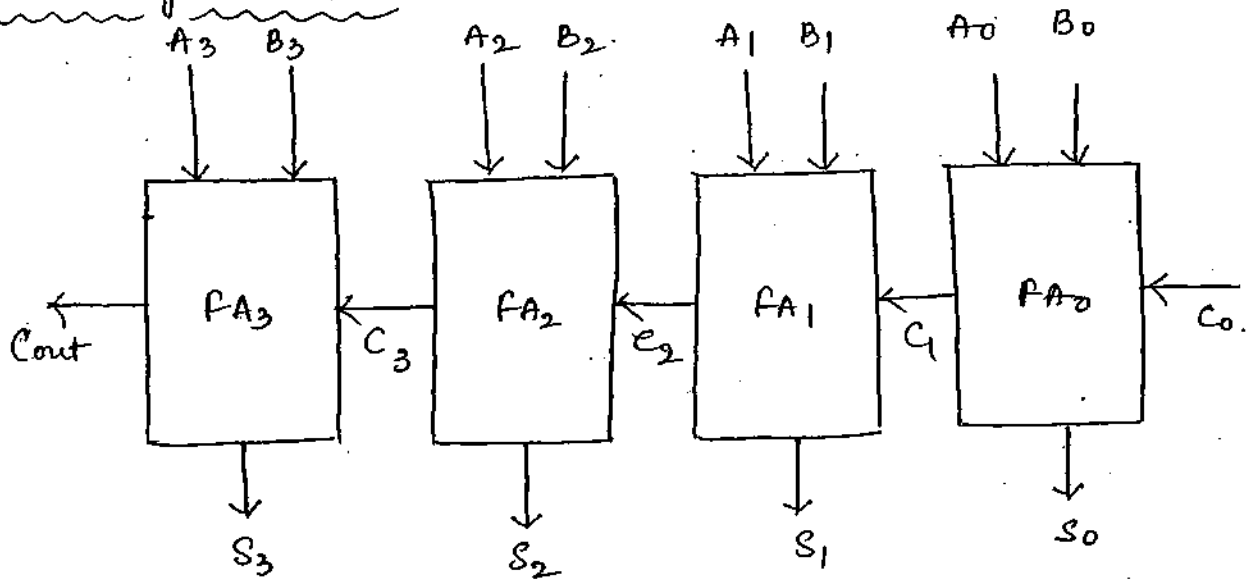
G_2 \ G_1G_0	00	01	11	10
0		1		1
1	1		1	

$B_0 = G_2 \oplus G_1 \oplus G_0$.



N-BIT ADDER :-

Ripple carry adder :-



For a 4-bit ripple carry adder, the time taken to give the sum output is $(3t_c + t_s)$ and for carry output is $(4t_c)$.

For a n-bit ripple carry adder, time taken to give o/p of

* sum ⟶ $(n-1)t_c + t_s$. [for $t_s \geq t_c$]

(expecting only sum) $(n-1)t_c + t_s$ ⟷ $(n)t_c$ [Expecting sum & final carry].

Eg:- 16 bit addition, $t_s = 15$ ns and $t_c = 12$ ns.

$\frac{16 \times 12}{32}$
$\frac{16}{192}$

sum ⟶ $15 \times 12 \times 10^{-9} + 15 \times 10^{-9} = 180 + 15 = 195$ ns.

carry ⟶ $16 \times 12 \times 10^{-9} = 192$ ns.

Eg:- 4 bit addition, $t_c = 1.5$ ns, $t_s = 1$ ns.

sum ⟶ $3 \times 1.5 + 1 \rightarrow 5.5$ ns.

CARRY LOOK AHEAD ADDER :-

$$C_0 = PC_i + G.$$

$$C_n = P_{n+1}C_{i_{n+1}} + G_{n-1}.$$

$$C_1 = P_0 C_0 + G_0$$

$$= C_0(A_0 \oplus B_0) + G_0.$$

$$= C_0 (A_0 \oplus B_0) + A_0 B_0.$$

	A	B	C_i	C_o	S
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	0	1
3	0	1	1	1	0
4	1	0	0	0	1
5	1	0	1	1	0
6	1	1	0	1	0
7	1	1	1	1	1

$C_o = 0$

$C_o = C_i$ } propagation of i/p carry to o/p,
$P = A \oplus B$.

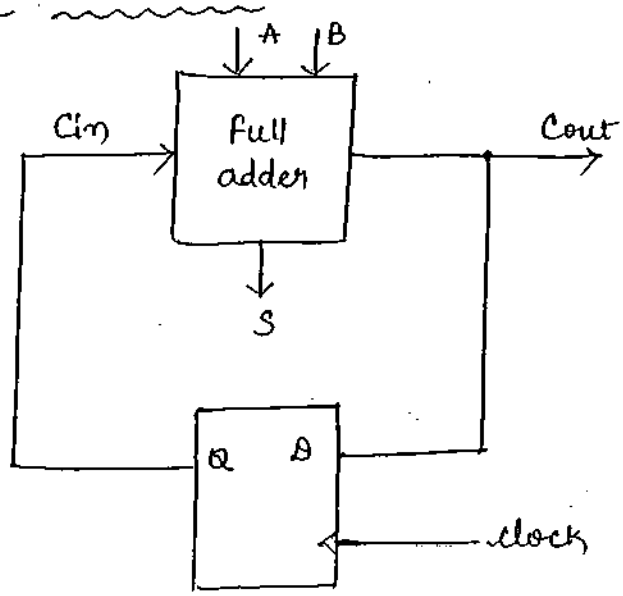
$C_o = 1$. } carry is generated
$G = AB$.

$$C_2 = P_1 C_1 + G_1$$
$$= (A_1 \oplus B_1)(C_0 (A_0 \oplus B_0) + A_0 B_0) + A_1 B_1.$$

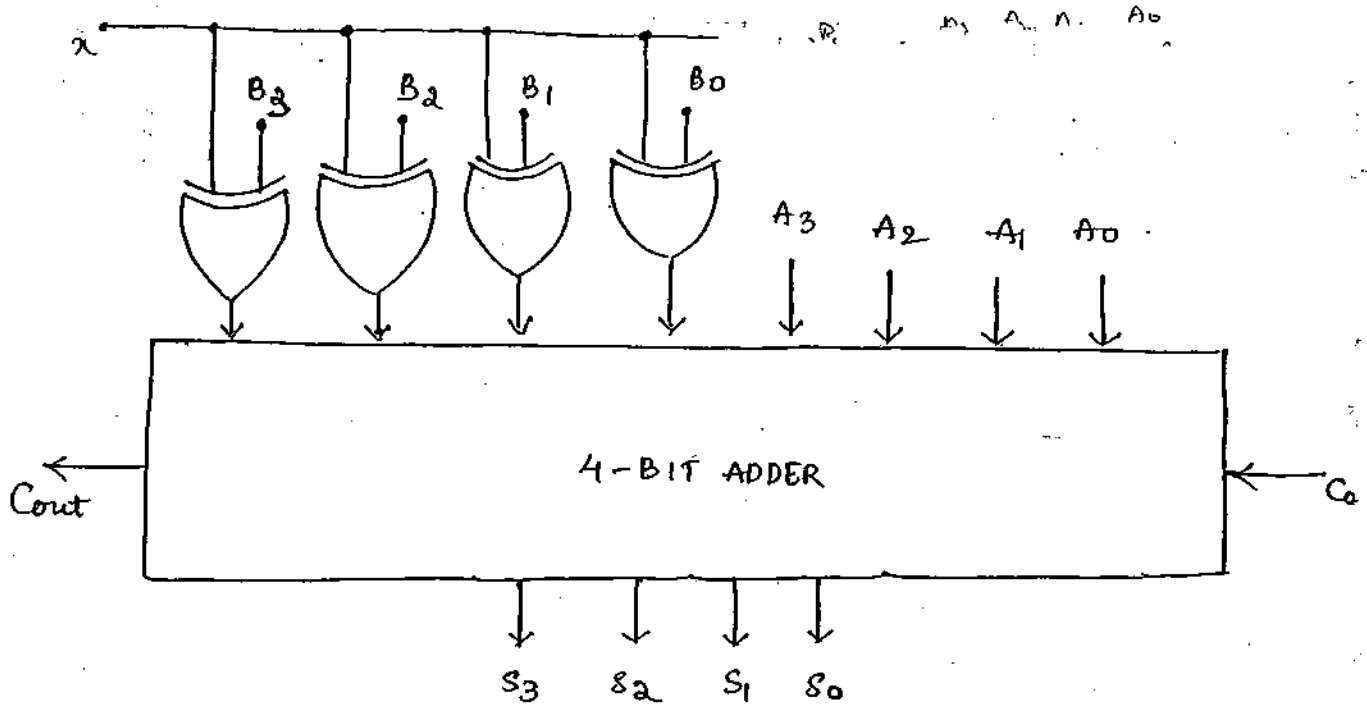
$$C_3 = P_2 C_2 + G_2$$
$$= (A_2 \oplus B_2)\{(A_1 \oplus B_1)\{[C_0 (A_0 \oplus B_0) + A_0 B_0]\} + A_1 B_1\} + A_2 B_2.$$

The carry generation itself taking more time. So carry look ahead adder cannot be used for more number of bits.

CARRY SAVE ADDER :-



clock period must be greater than or equal to full adder delay.



for $x=0$ & $C_0=0$

$A+B$.

$A+2^{1}s$ complement of B is $(A-B)$.

for $x=0$ & $C_0=1$

$A+B+1$

for $x=1$ & $C_0=0$

$A+\overline{B}$

for $x=1$ & $C_0=1$

$A+\overline{B}+1$.

$A+2^{1}s$

$A-B$

COMPARATOR :-

A	B	A>B	A<B	A=B
0	0	0	0	1
0	1	0	1	0
1	0	1	0	0
1	1	0	0	1

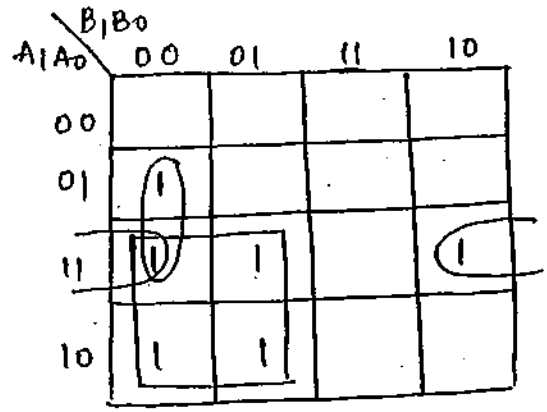
$A > B \Rightarrow A\bar{B}$.

$A < B \Rightarrow \bar{A}B$.

$A = B \Rightarrow \bar{A}\bar{B} + AB \Rightarrow A \odot B$.

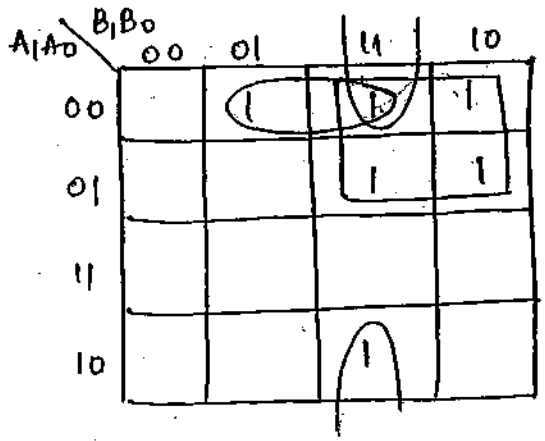
	A_1	A_0	B_1	B_0	A>B	A<B	A=B
0	0	0	0	0	0	0	1
1	0	0	0	1	0	1	0
2	0	0	1	0	0	1	0
3	0	0	1	1	0	1	0
4	0	1	0	0	1	0	0
5	0	1	0	1	0	0	1
6	0	1	1	0	0	1	0
7	0	1	1	1	0	1	0
8	1	0	0	0	1	0	0
9	1	0	0	1	1	0	0
10	1	0	1	0	0	0	1
11	1	0	1	1	0	1	0
12	1	1	0	0	1	0	0
13	1	1	0	1	1	0	0
14	1	1	1	0	1	0	0
15	1	1	1	1	0	0	1

$f(A>B) = \sum m(4,8,9,12,13,14)$.



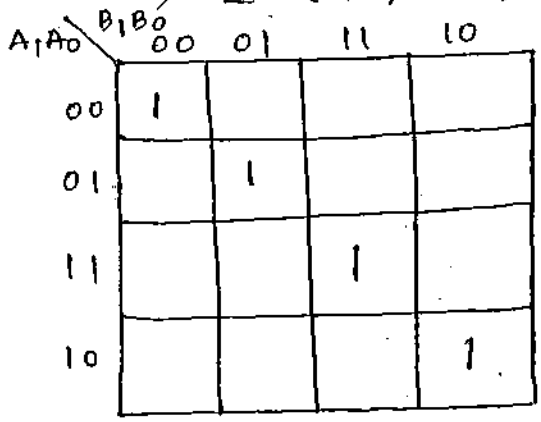
$f(A>B) = A_1\bar{B}_1 + A_0\bar{B}_1\bar{B}_0 + A_1A_0\bar{B}_0$

$f(A<B) = \sum m(1,2,3,6,9,11)$.



$f(A<B) = \bar{A}_1\bar{A}_0B_0 + \bar{A}_0B_1B_0 + \bar{A}_1B_1$.

$f(A=B) = \sum m(0,5,10,15)$.



$f(A=B) = \bar{A}_1\bar{A}_0\bar{B}_1\bar{B}_0 + \bar{A}_1A_0\bar{B}_1B_0 + A_1A_0B_1B_0 + A_1\bar{A}_0B_1\bar{B}_0$.

$A > B \Rightarrow$

$$A_1\overline{B_1} + (A_1 \odot B_1) A_0 \overline{B_0}.$$

$A < B \Rightarrow$

$$\overline{A_1} B_1 + (A_1 \odot B_1) \overline{A_0} B_0.$$

$A = B \Rightarrow$

$$(A_1 \odot B_1)(A_0 \odot B_1).$$

comparator o/p for 4 bit inputs;

$A < B \Rightarrow$

$$\{\overline{A_3} B_3 + (A_3 \odot B_3) \overline{A_2} B_2 + (A_3 \odot B_3)(A_2 \odot B_2) \overline{A_1} B_1 + (A_3 \odot B_3)(A_2 \odot B_2)(A_1 \odot B_1) \overline{A_0} B_0.\}$$

$A > B \Rightarrow$

$$\{A_3 \overline{B_3} + (A_3 \odot B_3) A_2 \overline{B_2} + (A_3 \odot B_3)(A_2 \odot B_2) A_1 \overline{B_1} + (A_3 \odot B_3)(A_2 \odot B_2)(A_1 \odot B_1) A_0 \overline{B_0}\}$$

$A = B \Rightarrow$

$$\{(A_3 \odot B_3)(A_2 \odot B_2)(A_1 \odot B_1)(A_0 \odot B_0)\}$$

BCD to Excess - 3 :-

	BCD ($B_3 B_2 B_1 B_0$)
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001.

Ex:- BCD representation of 9025 is

1001 0000 0010 0101

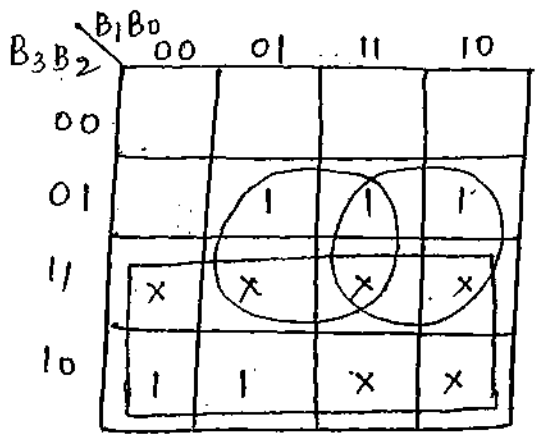
	B_3 B_2 B_1 B_0	E_3 E_2 E_1 E_0
0	0 0 0 0	0 0 1 1
1	0 0 0 1	0 1 0 0
2	0 0 1 0	0 1 0 1
3	0 0 1 1	0 1 1 0
4	0 1 0 0	0 1 1 1
5	0 1 0 1	1 0 0 0
6	0 1 1 0	1 0 0 1
7	0 1 1 1	1 0 1 0
8	1 0 0 0	1 0 1 1
9	1 0 0 1.	1 1 0 0

$$E_3 = \sum m(5,6,7,8,9) + d(10,11,12,13,14,15).$$

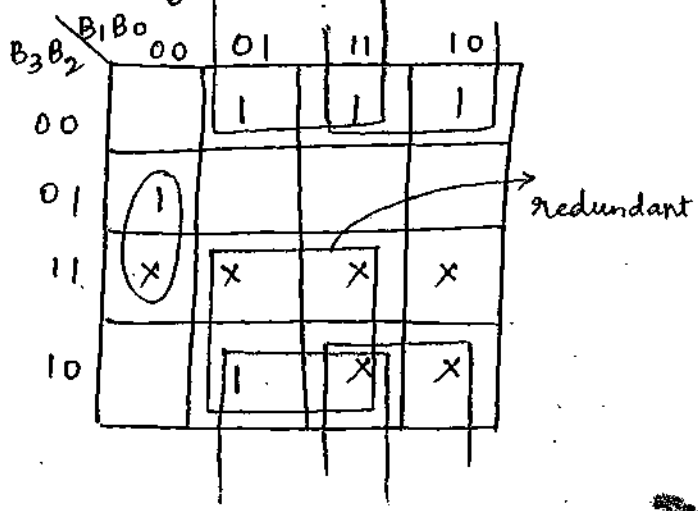
$$E_2 = \sum m(1,2,3,4,9) + d(10,11,12,13,14,15)$$

$$E_1 = \sum m(0,3,4,7,8) + d(10,11,12,13,14,15)$$

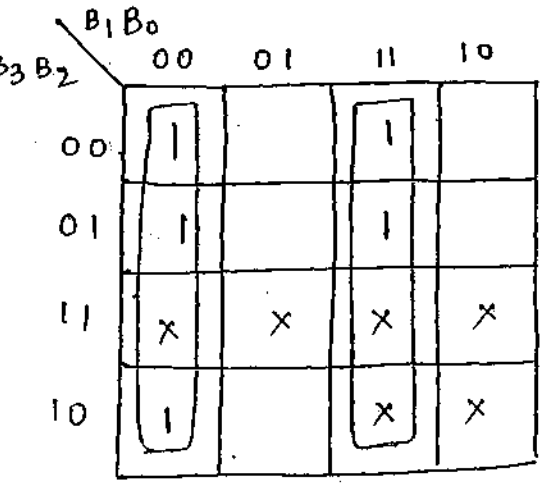
$$E_0 = \sum m(0,2,4,6,8) + d(10,11,12,13,14,15).$$



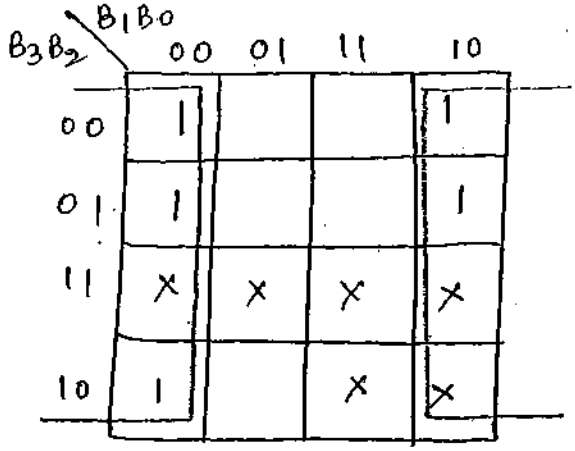
$E_3 = B_3 + B_2B_0 + B_2B_1$.



$E_2 = \overline{B_2}B_0 + \overline{B_2}B_1 + B_2\overline{B_1}\,\overline{B_0}$



$E_1 = \overline{B_1}\,\overline{B_0} + B_0B_1$.



$E_0 = \overline{B_0}$.

Excess 3 to BCD :-

	E_3	E_2	E_1	E_0	B_3	B_2	B_1	B_0
3+0	0	0	1	1	0	0	0	0
3+1	0	1	0	0	0	0	0	1
3+2	0	1	0	1	0	0	1	0
3+3	0	1	1	0	0	0	1	1
3+4	0	1	1	1	0	1	0	0
3+5	1	0	0	0	0	1	0	1
3+6	1	0	0	1	0	1	1	0
3+7	1	0	1	0	0	1	1	1
3+8	1	0	1	1	1	0	0	0
3+9	1	1	0	0	1	0	0	1

$B_3 = \sum m(11,12) + d(0,1,2,13,14,$

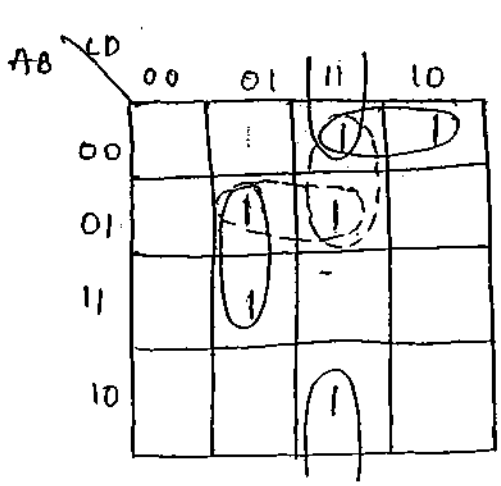
$B_2 = \sum m(7,8,9,10) +$
$d(0,1,2,13,14,1$

$B_1 = \sum m(5,6,9,10) +$
$d(0,1,2,13,14,1$

$B_0 = \sum m(4,6,8,10,12) +$
$d(0,1,2,13,14,15$

Eg:- Design a combinational ckt giving o/p, $f(A,B,C,D) = 1$, when decimal eqv of i/p binary decimal combination is prime number.

	A	B	C	D	P
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	0

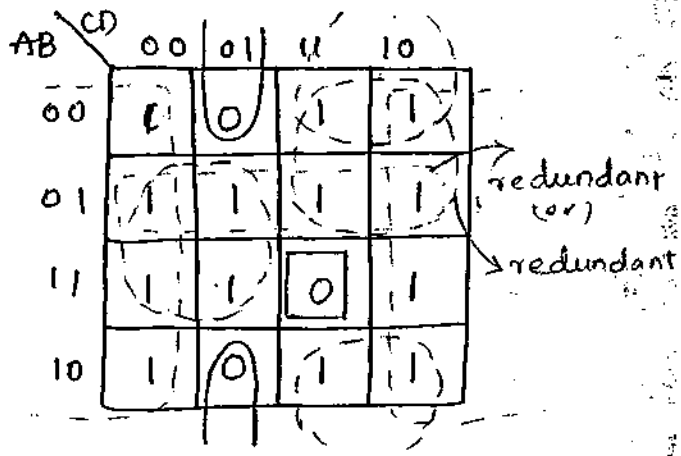


$P = B\bar{C}D + \bar{A}\bar{B}C + \bar{B}CD + \bar{A}BD$

(or)

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

Eg:- Design a combinational ckt. which is giving o/p one if the decimal equivalent of 4-bit binary is even (or) prime.



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

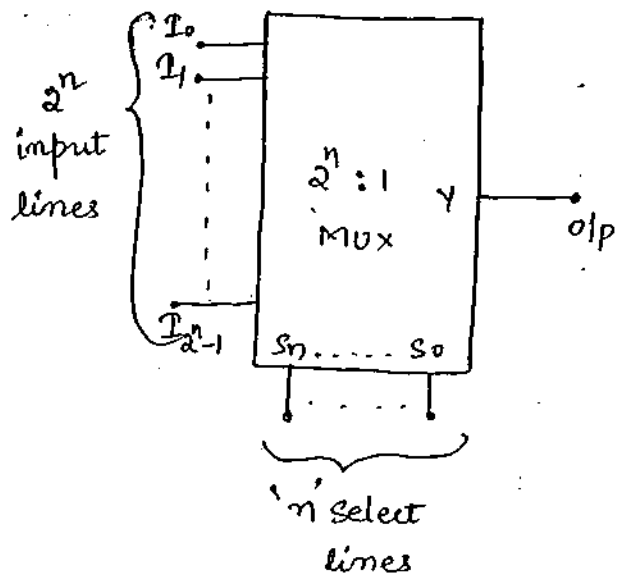
(or)

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

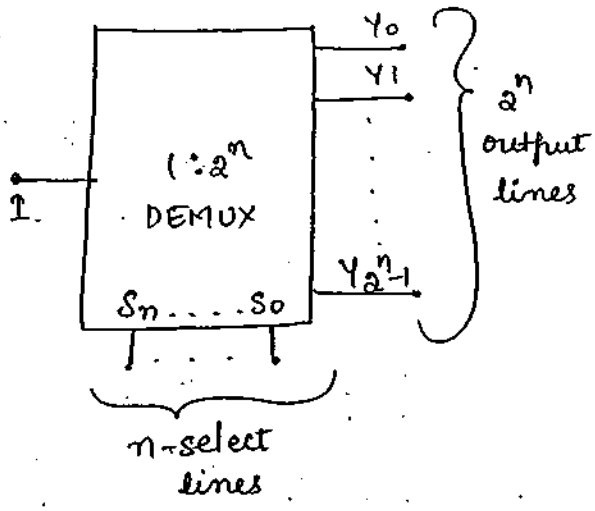
(or)

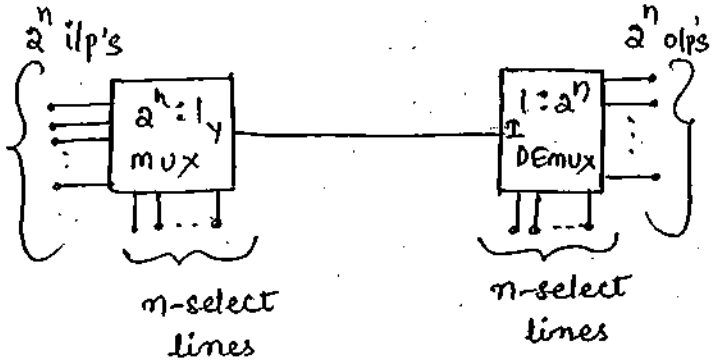
$\bar{A}C$.

MULTIPLEXER :-

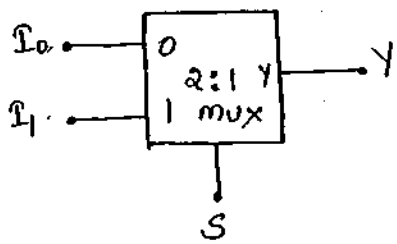


DEMULTIPLEXER :-





2:1 Multiplexer :-



2:1 ⇒ 2^1 : 1

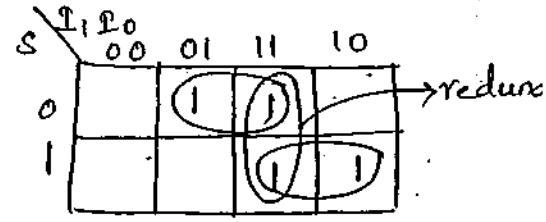
1 select line.

S	Y
0	I_0
1	I_1

$\therefore Y = \bar{S} I_0 + S I_1$

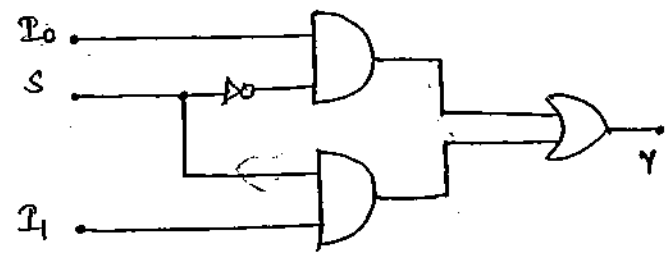
2:1 Multiplexer :-

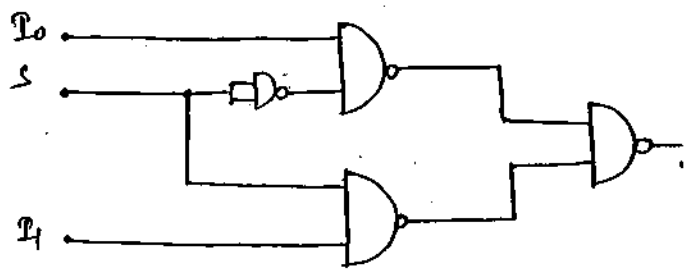
S	I_1	I_0	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1



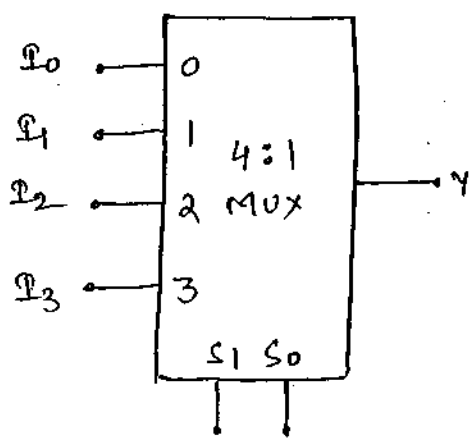
$Y = \bar{S} I_0 + S I_1$.

Implementation of 2:1 MUX using logic gate :-

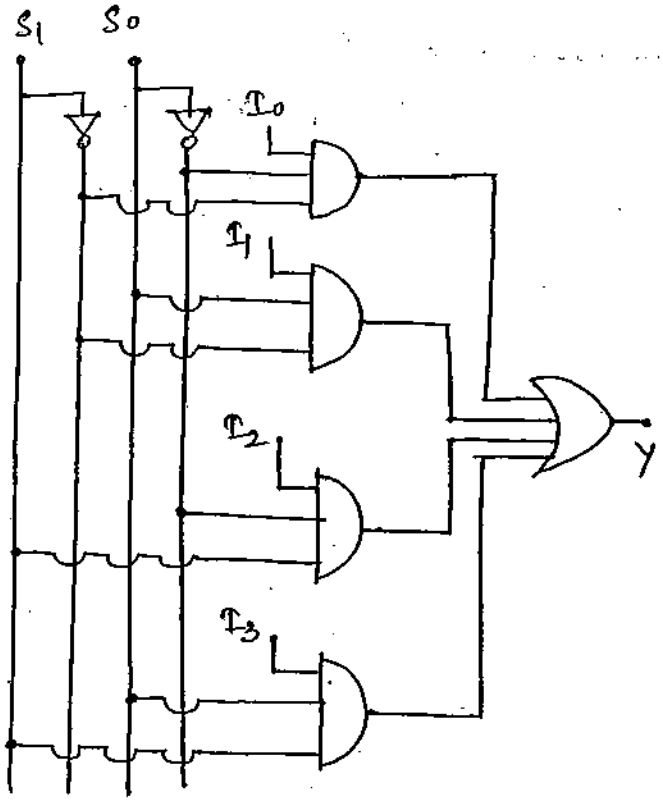




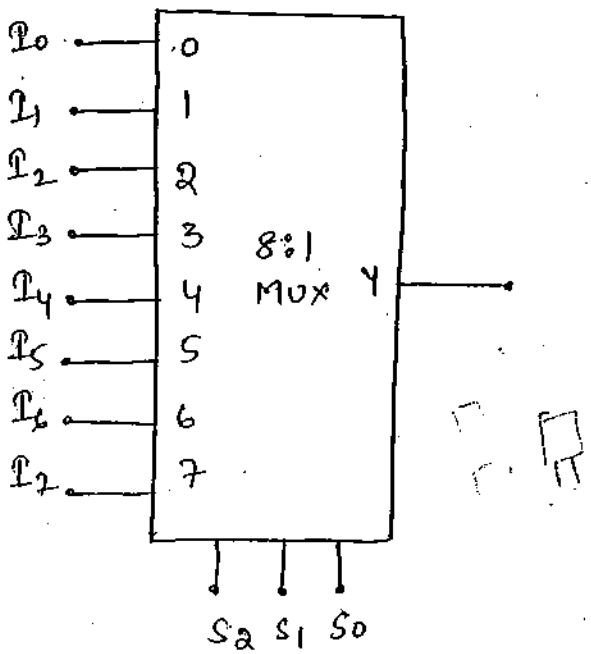
4:1 Multiplexer :-



$Y = \bar{S_1}\bar{S_0} I_0 + \bar{S_1} S_0 I_1 + S_1 \bar{S_0} I_2 + S_1 S_0 I_3$.



8:1 Multiplexer :-



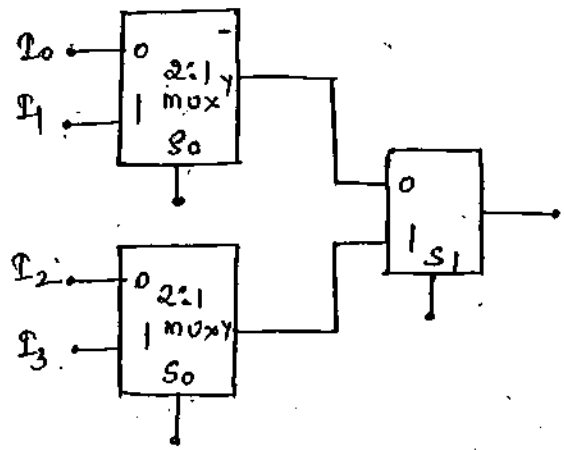
$$Y = \overline{S_2}\,\overline{S_1}\,\overline{S_0}\,I_0 + \overline{S_2}\,\overline{S_1}\,S_0\,I_1 + \overline{S_2}\,S_1\,\overline{S_0}\,I_2 + \overline{S_2}\,S_1\,S_0\,I_3 + S_2\,\overline{S_1}\,\overline{S_0}\,I_4 + S_2\,\overline{S_1}\,S_0\,I_5 + S_2\,S_1\,\overline{S_0}\,I_6 + S_2\,S_1\,S_0\,I_7.$$

Eg :- Design a 4:1 MUX using minimum no. of 2:1 multiplexers.

4:1

$\frac{4}{2} + 1 = 2 + 1 = 3.$

∴ 3 2:1 mux are required to represent 4:1 mux.



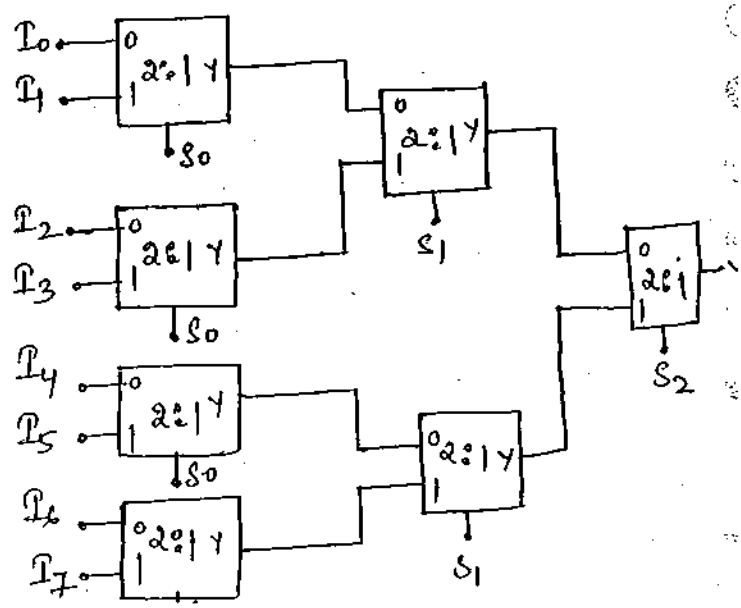
When a higher order mux is represented using a lower order mux, then the LSB select line is represented to the first range mux.

Ex :- Design a 8:1 MUX using minimum no. of 2:1 MUX.

$\frac{8}{2} + \frac{4}{2} + 1$

$\Rightarrow 4 + 2 + 1$

$\Rightarrow$ 7 muxes are required.

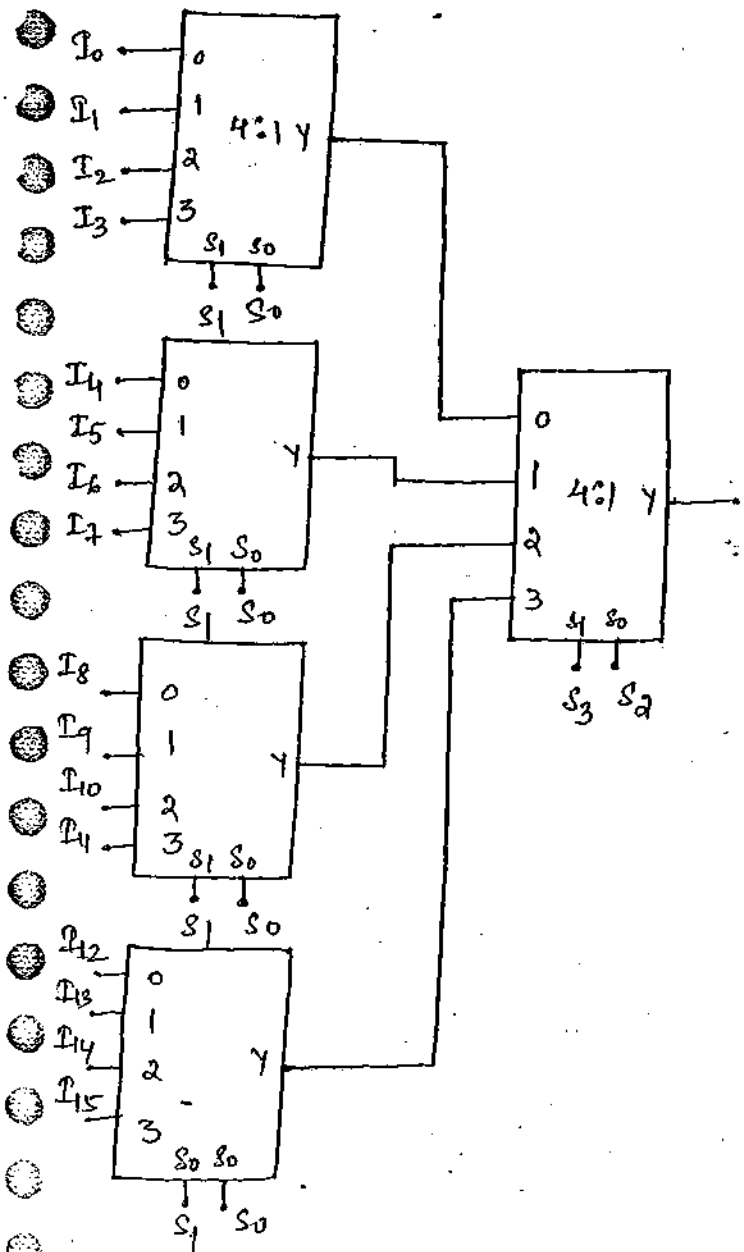


NOTE :-

To construct a 2^n : 1 MUX, using min no. of 2:1 MUX, then minimum of $(2^n - 1)$ multiplexers are required.

Eg :- Design 16:1 Multiplexer with min no. of 4:1 MUX.

$$\frac{16}{4} + \frac{4}{4} \Rightarrow 4 + 1 = 5.$$



Eg :- Find min. no. of 4:1 MUX required to construct 1024 : 1 Mux.

$$\frac{2^{10}}{4} + \frac{2^8}{4} + \frac{2^6}{4} + \frac{2^4}{4} + \frac{2^2}{4}$$

$$\Rightarrow 256 + 64 + 16 + 4 + 1$$

$$\Rightarrow 341.$$

Eg :- Find min no. of 4 : 1 MUX required to construct 4096 : 1 and find min no. of 8 : 1 MUX required to construct 4096 : 1

Sol :- 4 : 1 MUX $\longrightarrow$

$$\frac{4096}{4} + \frac{1024}{4} + \frac{256}{4} + \frac{64}{4}$$

$$\frac{16}{4} + \frac{4}{4}$$

$$\Rightarrow 1024 + 256 + 64 + 16 + 4 + 1$$

$$\Rightarrow 1024 + 341$$

$$\Rightarrow 1365.$$

8 : 1 MUX $\longrightarrow$

$$\frac{4096}{8} + \frac{512}{8} + \frac{64}{8} + \frac{8}{8}$$

$$\Rightarrow 512 + 64 + 8 + 1$$

$$\Rightarrow 585.$$

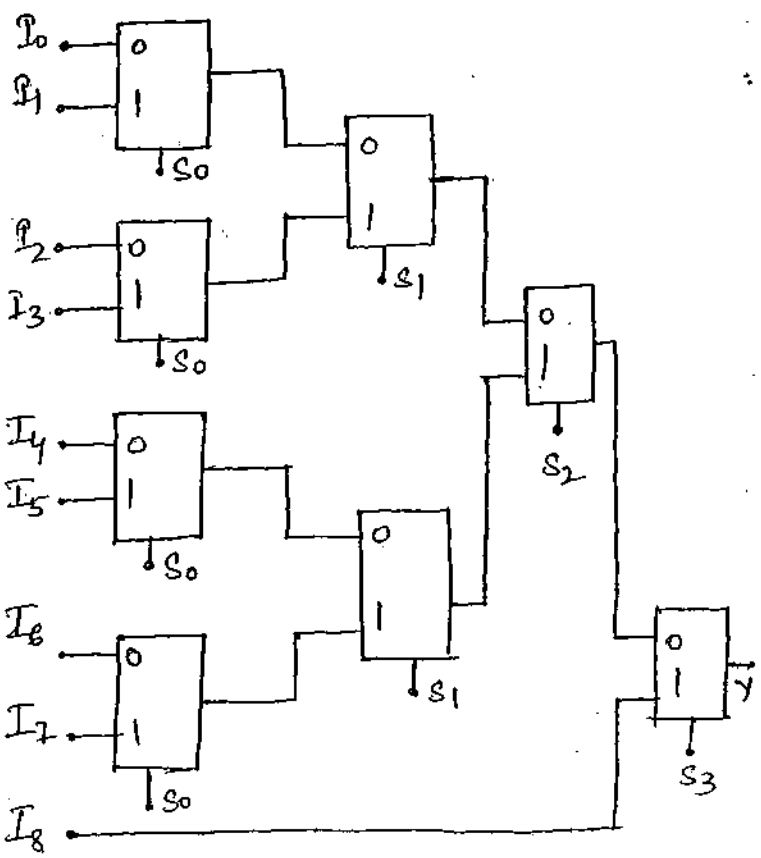
Eg :- Find min. no. of 2:1 MUX required to construct 9:1 MUX

Sol :-

S_3	S_2	S_1	S_0	Y
0	0	0	0	I_0
0	0	0	1	I_1
0	0	1	0	I_2
0	0	1	1	I_3
0	1	0	0	I_4
0	1	0	1	I_5
0	1	1	0	I_6
0	1	1	1	I_7
1	0	0	0	I_8

When S_3

When $S_3 = 0$, I_0 to I_7 is transferred to the o/p, when $S_3 = 1$, I_8 is passed to the o/p directly.



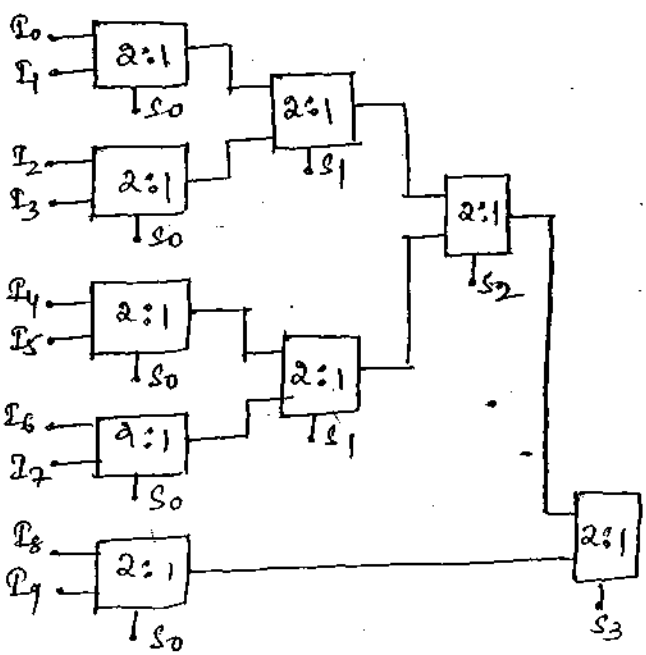
Eg: Design a 10:1 mux using with min no. of 2:1.

S_3	S_2	S_1	S_0	Y
0	0	0	0	I_0
0	0	0	1	I_1
0	0	1	0	I_2
0	0	1	1	I_3
0	1	0	0	I_4
0	1	0	1	I_5
0	1	1	0	I_6
0	1	1	1	I_7
1	0	0	0	I_8
1	0	0	1	I_9

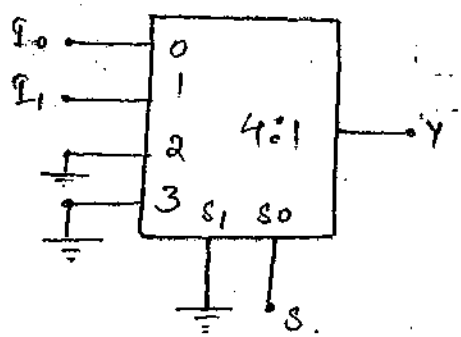
When $S_3 = 0$, I_0 to I_7 is transferred to o/p.

$$\frac{10}{2} + \frac{5}{2} + 1$$

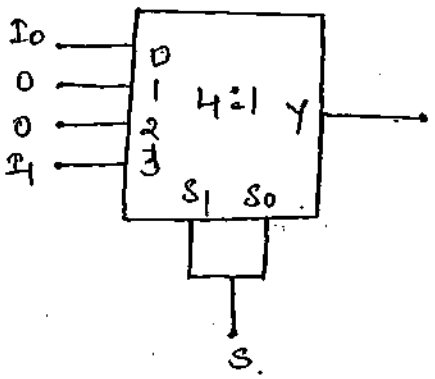
$$\Rightarrow 5 + 2 + 1 + \underline{1} = 9.$$

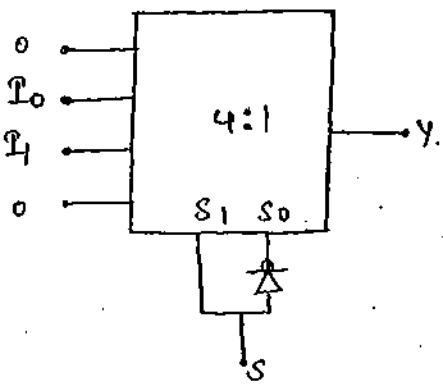


Eg:- Design 2:1 MUX using 4:1 MUX.



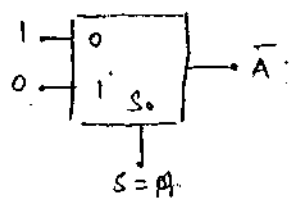
Sol:- There are many possibilities for this implementation.

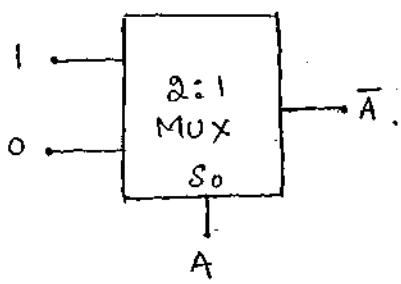




NOT gate implementation using multiplexer :-

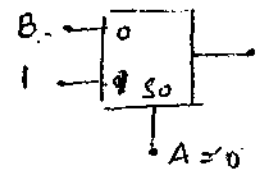
A	$Y = \bar{A}$
0	1
1	0

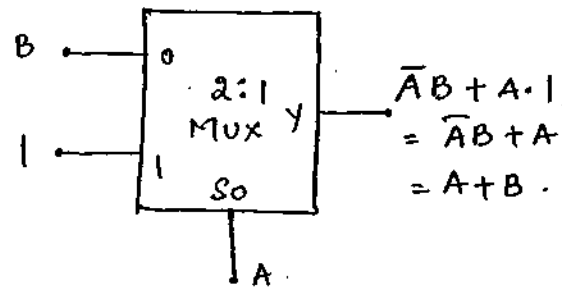




OR gate using MUX :-

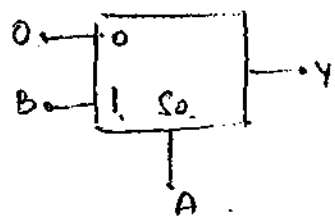
A	B	Y = A+B
0	0	0
0	1	1
1	1	1
1	0	1

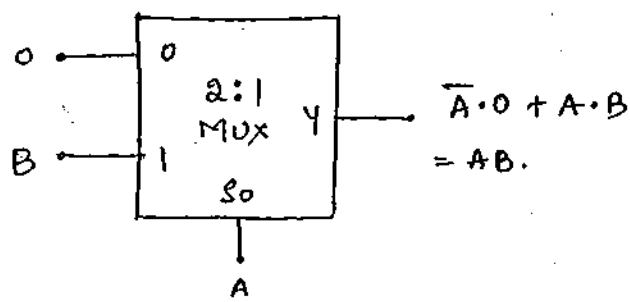




AND gate using MUX :-

A	B	Y = AB
0	0	0
0	1	0
1	0	0
1	1	1

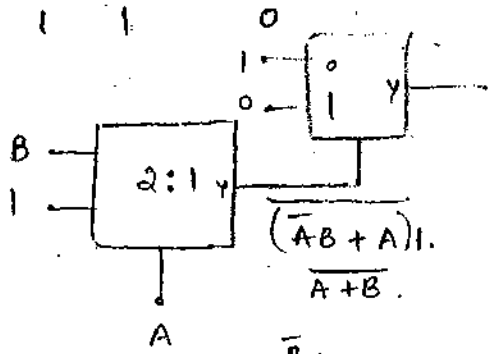


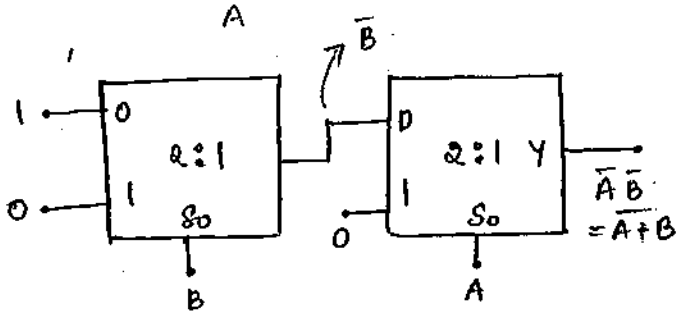


As it is possible to implement three basic gates using MUX, so we can say that MUX is an universal gate.

NOR gate using MUX :-

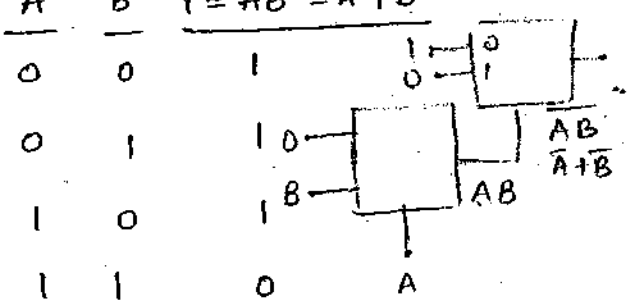
A	B	$Y = \overline{A+B} = \overline{A} \cdot \overline{B}$
0	0	1
0	1	0
1	0	0
1	1	0

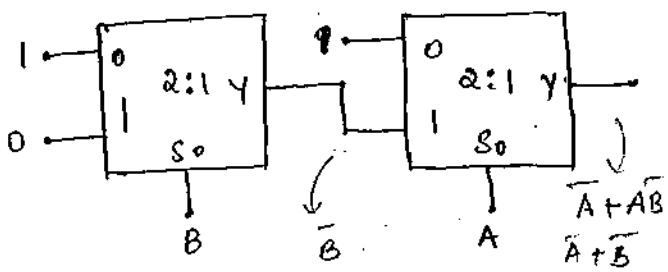




NAND gate using MUX :-

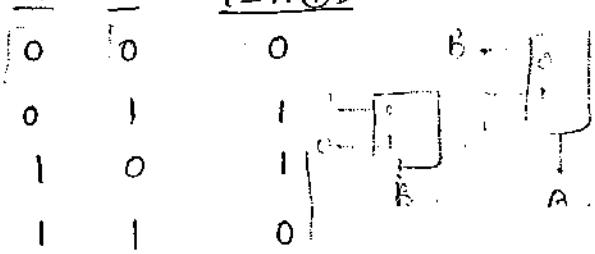
A	B	$Y = \overline{AB} = \overline{A} + \overline{B}$
0	0	1
0	1	1
1	0	1
1	1	0

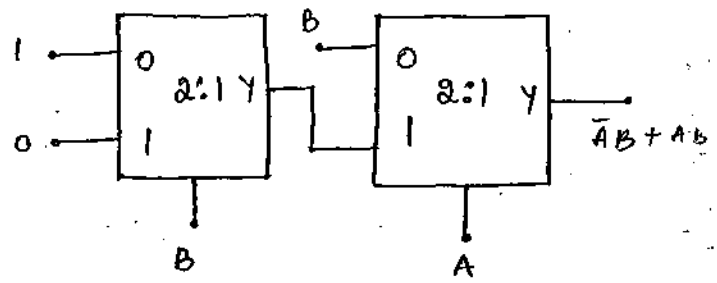




2i/p Ex-OR gate using MUX :-

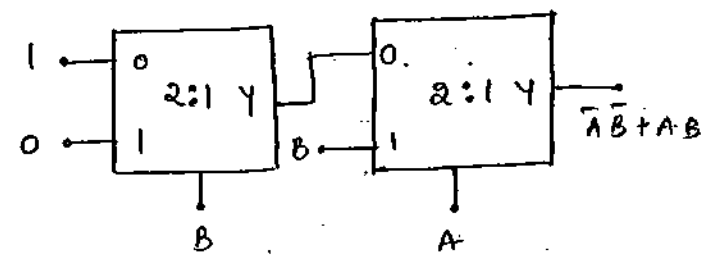
A	B	$Y = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0





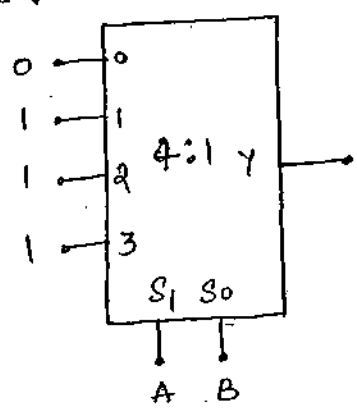
2 i/p Ex-NOR Gate using MUX :-

A	B	$Y = A \odot B$
0	0	1
0	1	0
1	0	0
1	1	1

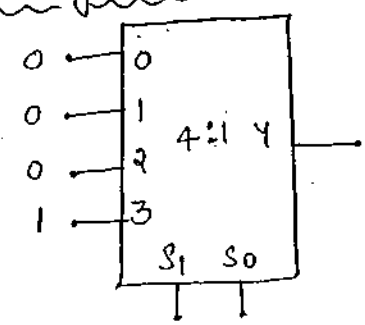


Implementation of logic gates using 4:1 MUX :-

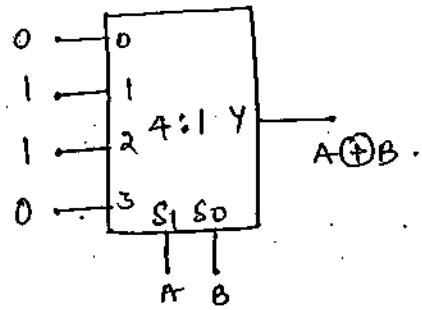
(i) OR gate :-



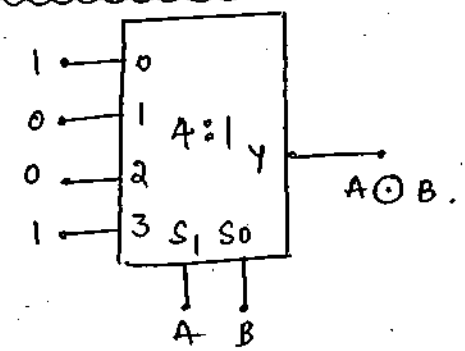
(ii) AND gate :-



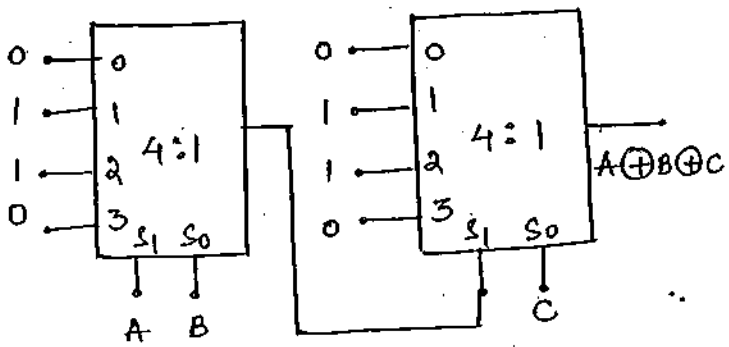
(iii) 2 i/p EXOR :-



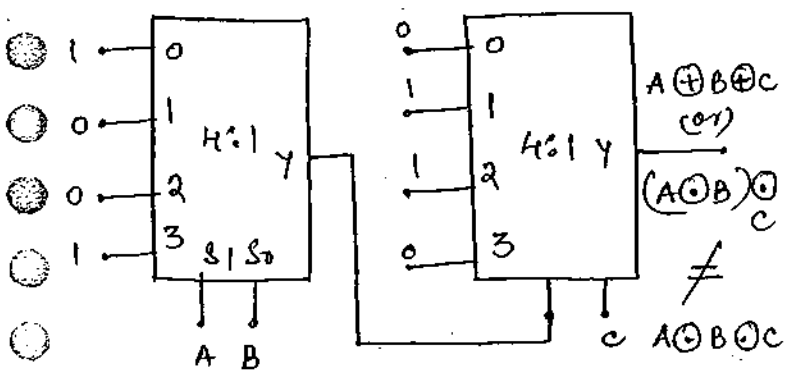
(iv) 2 i/p EX-NOR :-



(v) 3 i/p EX-OR :-



(vi) 3 i/p EX-NOR :-



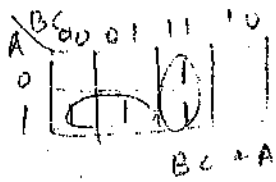
Eg :- Implement

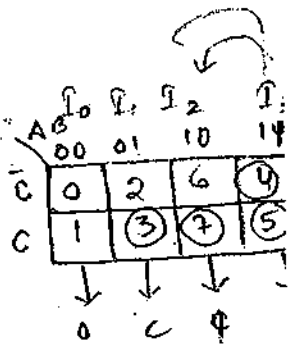
$f(A,B,C) = \sum m(3,4,5,7)$ using

(i) @ 8:1 MUX

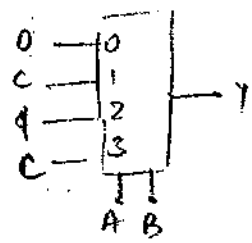
(b) 4:1 MUX and some

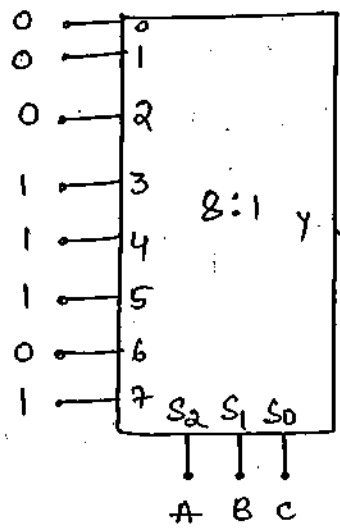
	A	B	C	Y
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1





By using 8:1 MUX ;





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

$A\bar{B}C$

$\bar{A}BC +$

$A(\bar{B}+C$

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

$A($

$C'(A+B) +$

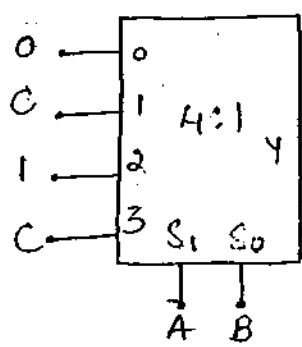
$A\bar{B}$

consensus $AC + BC'$

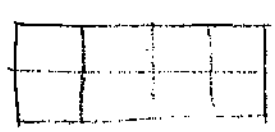
$A($

$= A\bar{B} + BC$

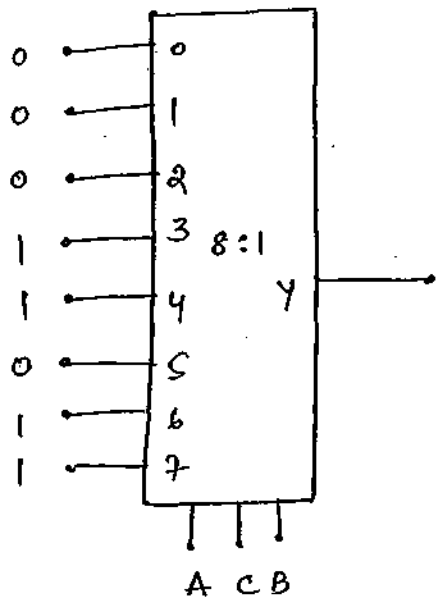
By using 4:1 MUX and some combinational gates,



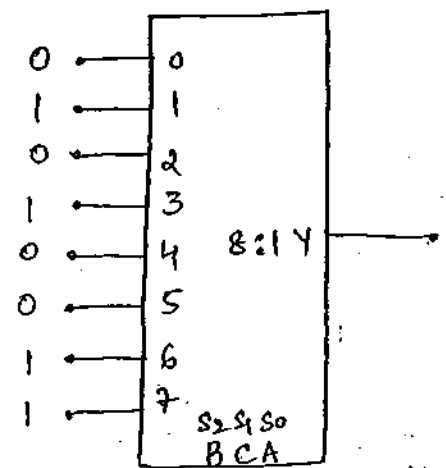
A	B	C	f
0	0	0	0
0	0	1	C
0	1	0	1
0	1	1	C
1	0	0	1
1	0	1	C
1	1	0	0
1	1	1	C



Using 8:1 MUX by interchanging select lines,



ACB	X0	X1	X2	X3	X4	X5	X6	X7
	000	001	010	011	100	101	110	111
	0	2	1	(3)	(4)	6	(5)	(7)



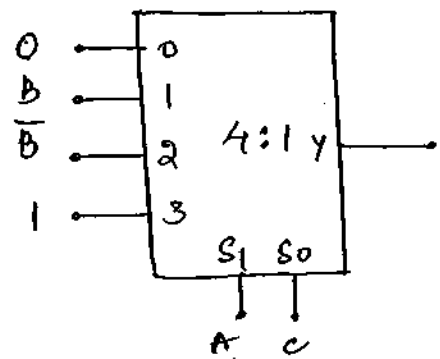
CBA / BCA	X0	X1	X2	X3	X4	X5	X6	X7
	000	001	010	011	100	101	110	111
	0	(4)	2	6	1	(5)	(3)	(7)

BCA	X0	X1	X2	X3	X4	X5	X6	X7
	000	001	010	011	100	101	110	111
	0	(4)	1	(5)	2	6	(3)	(7)

using 4:1 MUX using 'A' and 'c' as select lines.

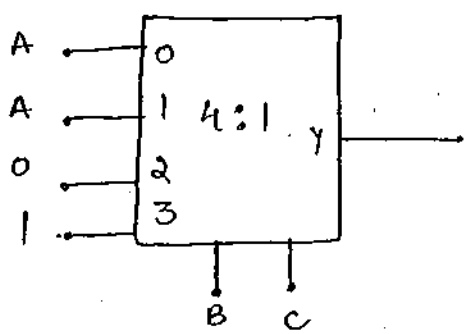
$\bar{B}$ / B \ AC	00	01	10	11
$\bar{B}$	0	1	(4)	(5)
B	2	(3)	6	(7)

	A	B	C	f
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1



using 4:1 MUX using 'B' and 'C' as select lines,

$\bar{A}$ / A \ BC	00	01	10	11
$\bar{A}$	0	1	2	(3)
A	(4)	(5)	6	(7)
	A	A	0	1



using 4:1 MUX using 'C' and 'A' are select lines,

B / $\bar{B}$ \ CA	00	01	10	11
$\bar{B}$	0	(4)	1	(5)
B	2	6	(3)	(7)
	↓	↓	↓	↓

using 'C' and 'B' as select lines,

A \ CB	00	01	10	11
$\bar{A}$	0	2	1	(3)
A	(4)	6	(5)	(7)
	↓ A	↓ 0	↓ A	↓ 1

Eg:- Implement the function

$f(A,B,C,D)$

Eg:- Using only one 4:1 MUX we can implement

(a) All 3 variable and 2-variable functions.

(b) Only 2 variable functions.

(c) ✓ All 2-variable functions and some of 3-variable functions

(d) none.

Eg:- Implement the function

$f(A,B,C,D) = \sum m(1,2,3,4,6,7,8,11,13,14)$

using (a) 8:1 MUX and some combinational gates. Use A,C,D as select lines.

(b) 4:1 MUX and some combinational gates.

Sol:-

(a) $f(A,B,C,D) =$

$\sum m(1,2,3,4,6,7,8,11,13,14)$

	A	B	C	D	F
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	1

8421

ACD	000	001	010	011	100	101	110	111
$\bar{B}$	0	(1)	(2)	(3)	(8)	9	10	(11)
B	(4)	5	(6)	(7)	12	(13)	(14)	15
	↓	↓	↓	↓	↓	↓	↓	↓
	B	$\bar{B}$	1	1	$\bar{B}$	B	B	$\bar{B}$
	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7

8421

By using DBA as select lines,

DBA	000	001	010	011	100	101	110	111
$\bar{C}$	0	(8)	(4)	12	(1)	9	5	(13)
C	(2)	10	(6)	(14)	(3)	(11)	(7)	15
	↓	↓	↓	↓	↓	↓	↓	↓
	C	$\bar{C}$	1	C	1	C	C	$\bar{C}$
	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7

E.K.D.C.V.V

$f(A,B,C,D) = \sum m(1,2,3,4,6,7,8,11,13,14)$.

AB	00	01	10	11
$\bar{C}\bar{D}$ (00)	0	(4)	(8)	12
$\bar{C}D$ (01)	(1)	5	9	(13)
$C\bar{D}$ (10)	(2)	(6)	10	(14)
CD (11)	(3)	(7)	(11)	15

$\bar{C}D + C\bar{D} + CD = C + D$

$\bar{C}\bar{D} + C\bar{D} + CD = C + \bar{D}$

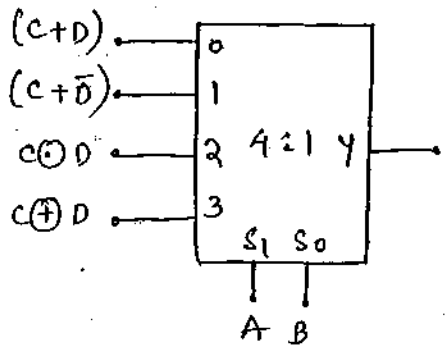
$\bar{C}D + C\bar{D}$.

$I_0 \rightarrow (C+D)$.

$I_1 \rightarrow (C+\bar{D})$.

$I_2 \rightarrow C \odot D$.

$I_3 \rightarrow C \oplus D$.



By considering B,D as select lines,

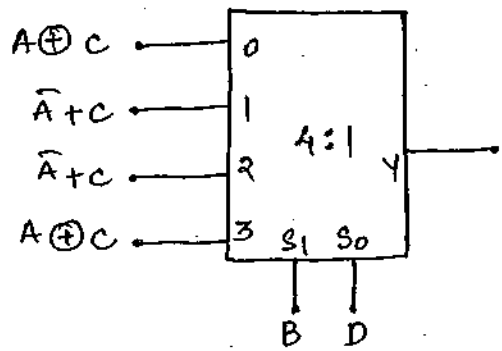
BD	00	01	10	11
$\bar{A}\bar{C}$ (00)	0	(1)	(4)	5
$\bar{A}C$ (01)	(2)	(3)	(6)	(7)
$A\bar{C}$ (10)	(8)	9	12	(13)
AC (11)	10	(11)	(14)	15

$I_0 \rightarrow \bar{A}C + A\bar{C} \rightarrow A \oplus C$.

$I_1 \rightarrow \bar{A} + C$

$I_2 \rightarrow \bar{A} + C$

$I_3 \rightarrow \bar{A}C + A\bar{C} \rightarrow A \oplus C$.



Now considering C,A are select lines

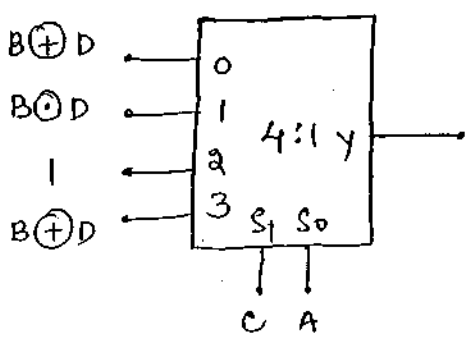
CA	00	01	10	11
$\bar{B}\bar{D}$ (00)	0	(8)	(2)	10
$\bar{B}D$ (01)	(1)	9	(3)	(11)
$B\bar{D}$ (10)	(4)	12	(6)	(14)
BD (11)	5	(13)	(7)	15

$I_0 \rightarrow \bar{B}D + B\bar{D} \rightarrow B \oplus D$.

$I_1 \rightarrow \bar{B}\bar{D} + BD \rightarrow B \odot D$

$I_2 \rightarrow 1$

$I_3 \rightarrow \bar{B}D + B\bar{D} \rightarrow B \oplus D$.



By considering D, A as select lines,

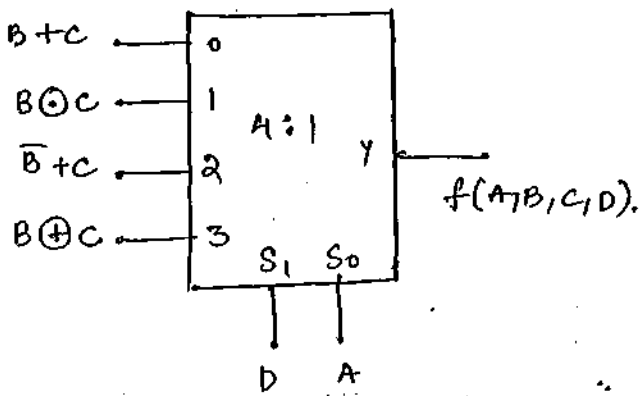
DA	00	01	10	11
$\bar{B}\bar{C}$ (00)	0	(8)	(1)	9
$\bar{B}C$ (01)	(2)	10	(3)	(11)
$B\bar{C}$ (10)	(4)	12	5	(13)
BC (11)	(6)	(14)	(7)	15

$I_0 \longrightarrow B \phi C \longrightarrow B+C$

$I_1 \longrightarrow B \odot C$

$I_2 \longrightarrow \bar{B}+C$

$I_3 \longrightarrow B \oplus C$.



Implement the sum of full adder using 4:1 MUX :-

$\sum m$

Sum $= f(A,B,C_i) = \{1,2,4,7\}$

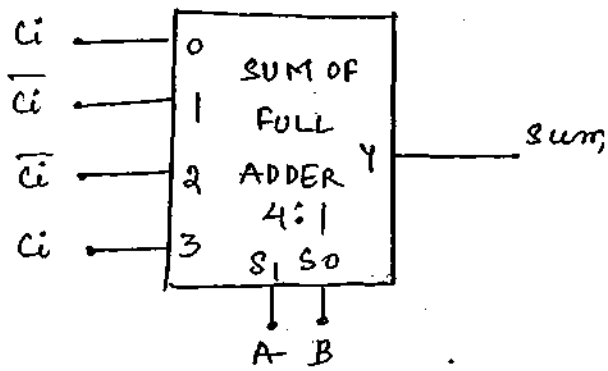
$= A \oplus B \oplus C_i$

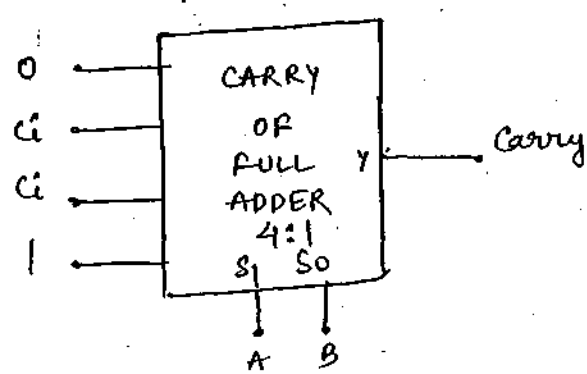
By considering A, B as select lines

AB	00	01	10	11
$\bar{C}_i$ (0)	0	(2)	(4)	6
C_i (1)	(1)	3	5	(7)
	↓	↓	↓	↓
	C_i	$\bar{C}_i$	$\bar{C}_i$	C_i
	I_0	I_1	I_2	I_3.

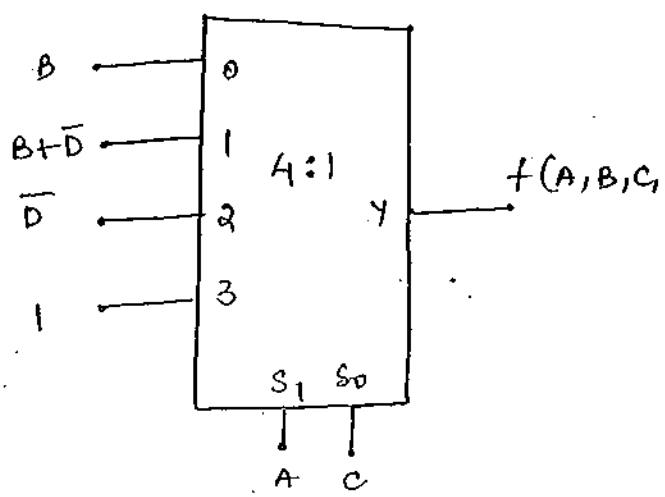
Carry $= F(A,B,C_i) = \sum m(3,5,6,7)$.

AB	00	01	10	11
$\bar{C}_i$ (0)	0	2	4	(6)
C_i (1)	1	(3)	(5)	(7)
	↓	↓	↓	↓
	0	C_i	C_i	1
	I_0	I_1	I_2	I_3.





Eg:- Find minimal expression of fn $f(A,B,C,D)$ implemented by # multiplexer ckt shown in the figure in SOP and POS.



$f(A,B,C,D) =$

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

$A\bar{C}(\bar{D}) + AC.$

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

AC

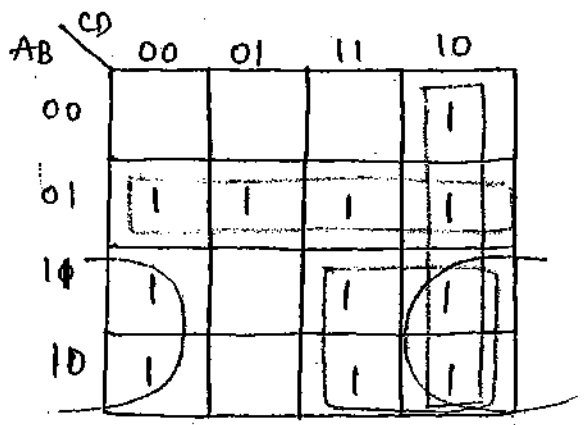
$\Rightarrow \bar{A}B + \bar{A}C\bar{D} + A(C+\bar{C}\bar{D})$

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

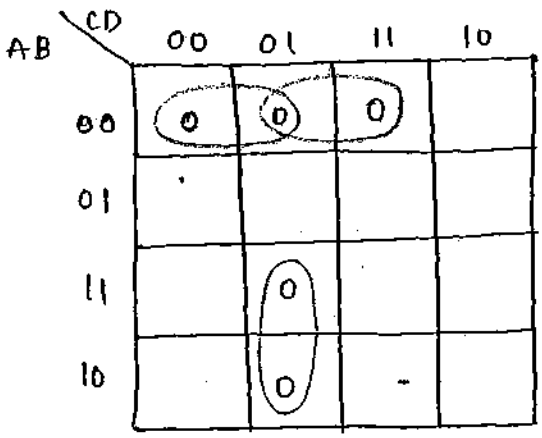
$\Rightarrow \bar{A}B + \bar{A}C\bar{D} + AC + A\bar{D}$

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

$\Rightarrow \bar{A}B + AC + A\bar{D} + C\bar{D}.$



SOP form $\Rightarrow \bar{A}B + C\bar{D} + AC + A\bar{D}$



POS form $\Rightarrow (A+B+C)\cdot(A+B+\bar{D})\cdot(\bar{A}+C+\bar{D}).$

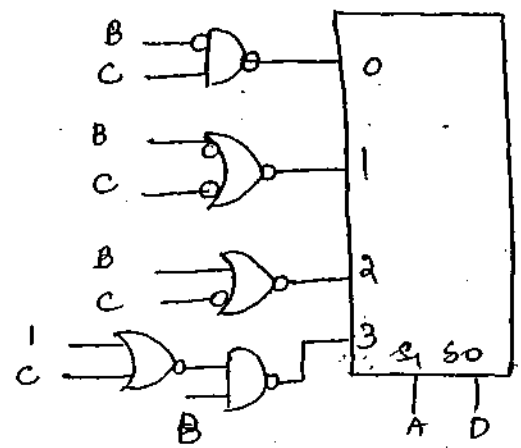
$f(A,B,C,D) =$

$\sum m(2,4,5,6,7,8,10,11,12,14,15).$

To know the minterms,

AC	00	01	11	10
$\bar{B}\bar{D}$ (00)	0	(2)	(8)	(10)
$\bar{B}D$ (01)	1	3	9	(11)
$B\bar{D}$ (10)	(4)	(6)	(12)	(14)
BD (11)	(5)	(7)	13	(15)

Eg:- Find no. of P.I and E.P.I for fn $f(A,B,C,D)$ implemented by the circuit shown in the figure.



Sol:- $I_0 \longrightarrow \overline{\bar{B}\cdot C} = B+\bar{C}$

$I_1 \longrightarrow \overline{B+\bar{C}} = B\cdot C$

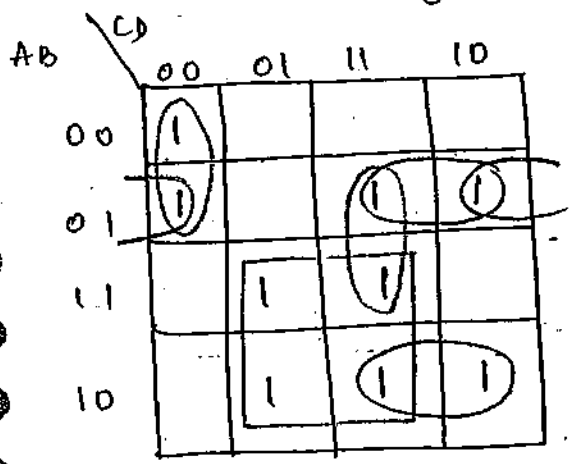
$I_2 \longrightarrow \overline{B+\bar{C}} = \bar{B}\cdot C.$

$I_3 \longrightarrow \overline{(\overline{1+C})\cdot B}$

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

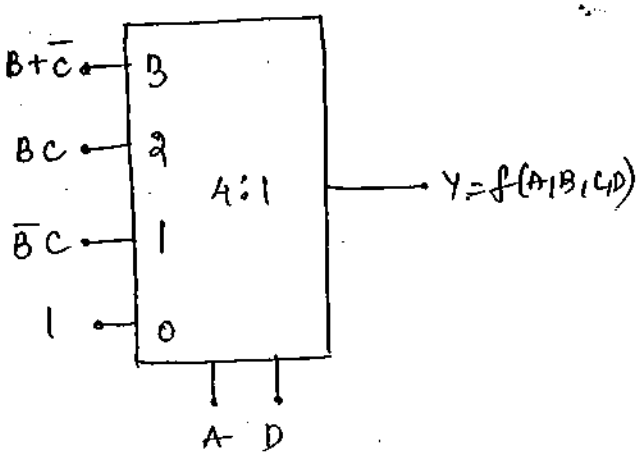
AD	00	01	10	11
$\bar{B}\bar{C}$ (00)	(0)	1	8	(9)
$\bar{B}C$ (01)	2	3	(10)	(11)
$B\bar{C}$ (10)	(4)	5	12	(13)
BC (11)	(6)	(7)	14	(15)
	$B+\bar{C}$	BC	$\bar{B}C$	1

$f = \sum m\{0,4,6,7,9,10,11,13,15\}$.



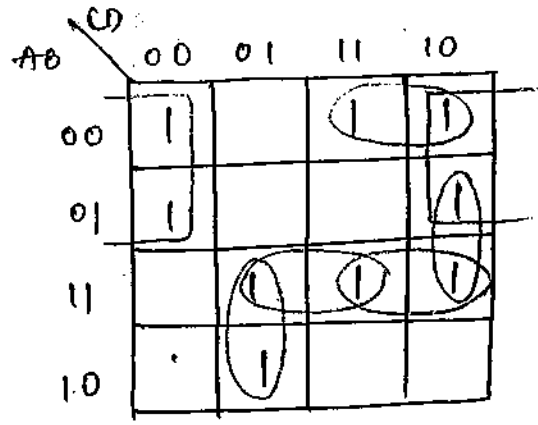
P.I → 6.

EPI → 3



AD	00	01	10	11
$\bar{B}\bar{C}$ (00)	(0)	1	8	(9)
$\bar{B}C$ (01)	(2)	(3)	10	11
$B\bar{C}$ (10)	(4)	5	12	(13)
BC (11)	(6)	7	(14)	(15)
	$B+\bar{C}$	BC	$\bar{B}C$	1.

$f = \sum m(0,2,3,4,6,9,13,14,15)$.

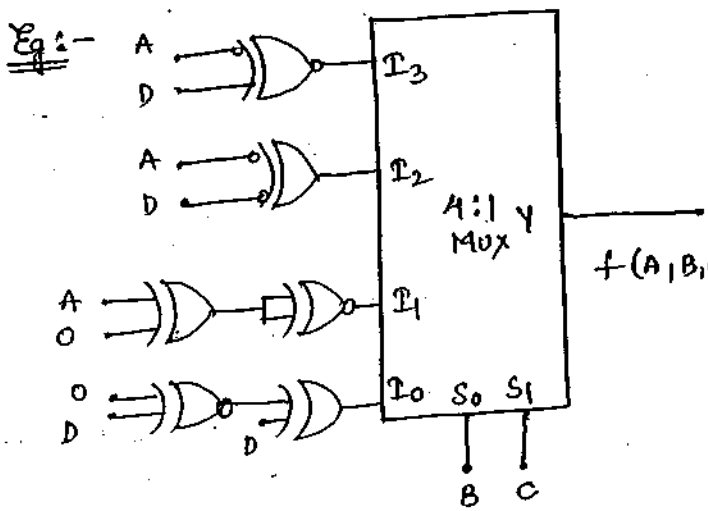


P.I → 6

E.P.I → 3.

$f(A,B,C,D) = \bar{A}\bar{D} + A\bar{C}D + \bar{A}\bar{B}C + ABC$.

Eg:-



$I_0 \rightarrow \overline{(0 \oplus D)} \oplus D$

$(\bar{D} \oplus D) \rightarrow 1$

$I_1 \rightarrow (A \oplus 0) \odot (A \oplus 0)$

$A \odot A \rightarrow 1$

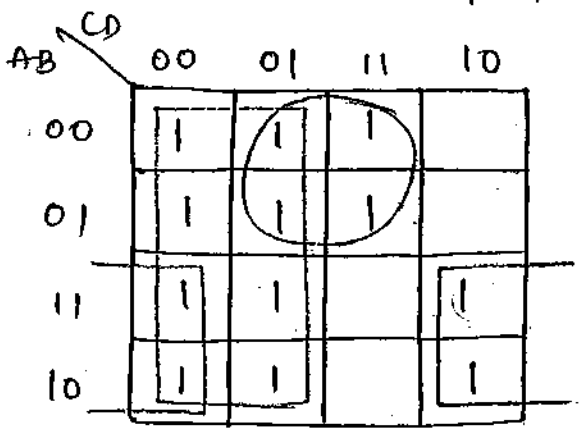
$I_2 \rightarrow (\bar{A} \oplus \bar{D})$.

$I_3 \rightarrow \overline{(\bar{A} \oplus D)} \rightarrow \bar{A} \odot D$

$\rightarrow A \oplus D$

C B	00	01	10	11
$\bar{A}\bar{D}$ (00)	(0)	(4)	2	6
$\bar{A}D$ (01)	(1)	(5)	(3)	(7)
$A\bar{D}$ (10)	(8)	(12)	(10)	(14)
AD (11)	(9)	(13)	11	15
	(1)	(1)	$(\bar{A}\oplus\bar{D})$	$(A\oplus D)$

$f(A,B,C,D) = \sum m(^{0,1,}_{3,4,5,7,10,}{}^{8,9.}_{}$
$12,13,14).$



P.I → 3

E.P.I → 3

Minimal expression →

$\bar{C} + \bar{A}D + A\bar{D}.$

* Pg.55 Q.4

Sol:-

$I_0 \rightarrow P+\bar{Q}$

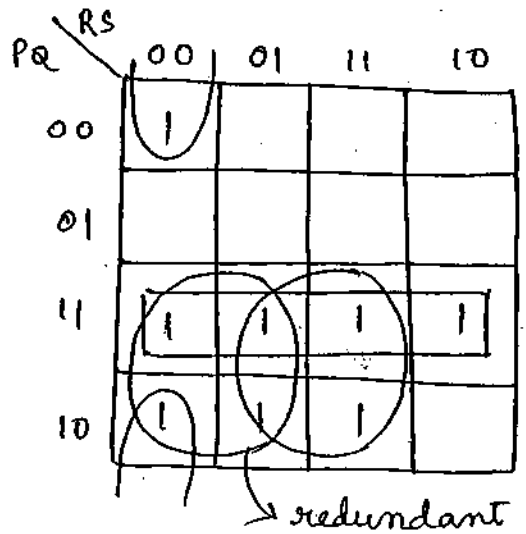
$I_1 \rightarrow P$

$I_2 \rightarrow PQ$

$I_3 \rightarrow P.$

RS	00	01	10	11
$\bar{P}\bar{Q}$ (00)	(0)	1	2	3
$\bar{P}Q$ (01)	4	5	6	7
$P\bar{Q}$ (10)	(8)	(9)	10	(11)
PQ (11)	(12)	(13)	(14)	(5)
	↓	↓	↓	↓
	$P+\bar{Q}$	P	PQ	P.

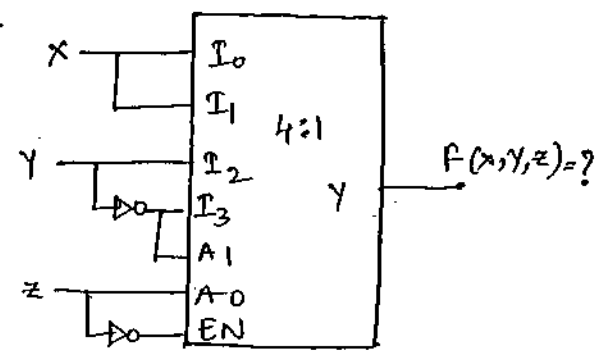
$f(P,Q,R,S) = \sum m\{0,8,9,11,12,13,14,15\}$



$f(P,Q,R,S) = PQ + P\bar{R}S + \bar{Q}\bar{R}\bar{S}.$

* Q.9 Pgno.56

Sol:-



For enabling MUX, EN = 1 must.

$\therefore Z = 0.$

$A_0 = 0.$

A_1	A_0
0	0
1	0.

$f \neq (x, \bar{z})$.

$F(X,Y,Z) = \bar{A_1}\,\bar{A_0}\,I_0 + \bar{A}_1 A_0 I_1 + A_1 \bar{A}_0 I_2 + A_1 A_0 I_3$.

$= \bar{A_1} I_0 + 0 + A_1 I_2 + 0$

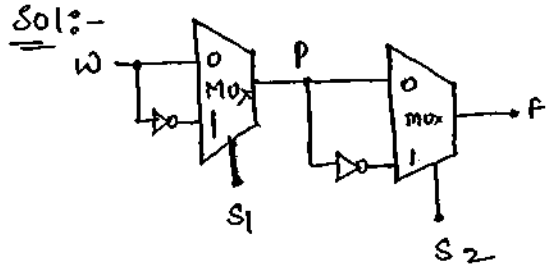
$= (\bar{A_1} I_0 + A_1 I_2)\bar{z}$

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

$= xy\bar{z}$.

* Q.1 pgno.57

Sol:-



$P = \bar{S_1} w + S_1 \bar{w} = S_1 \oplus w$

$F = \bar{S_2} P + S_2 \bar{P} = S_2 \oplus P$

$= S_2 \oplus S_1 \oplus w$.

* Q.2 pgno.58

Sol:- $T_C = 12\,ns$.

$T_S = 15\,ns$.

Worst-case delay is

$(n-1)\,T_C + T_S$.

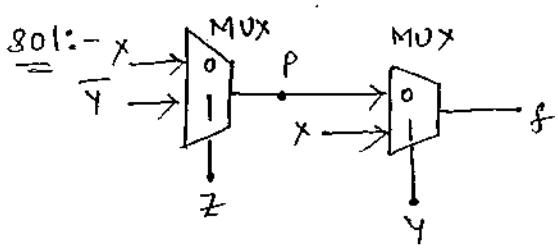
$\Rightarrow (16-1)\,T_C + T_S$

$\Rightarrow 15 \times 12 + 15$

$\Rightarrow 195\,ns$.

* Q.2 pgno.54

Sol:-



$p = \bar{z}x + z\bar{y}$.

$f = \bar{y}p + yx$.

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

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

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

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

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

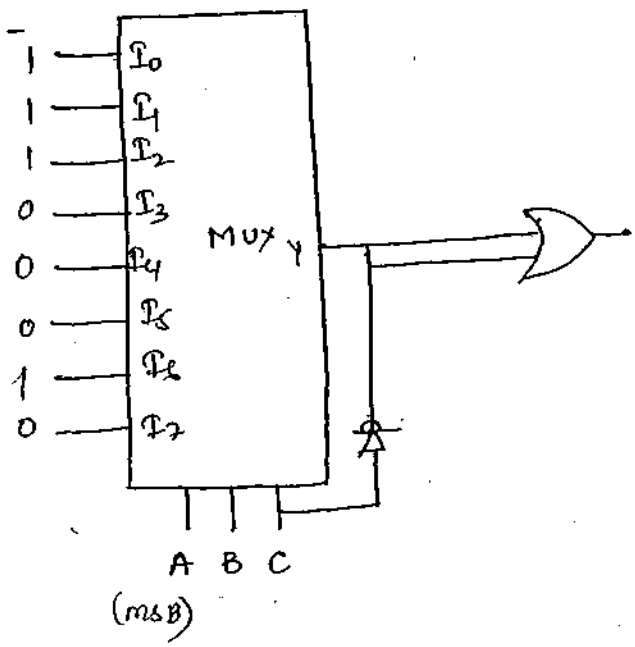
Q.3 pgno.58

Sol:-

S_1 A_2	S_0 A_1	A_0 ($\bar{E}$).	output
0	0	0	I_0
0	0	1	1
0	1	1	1
0	1	0	I_1
1	1	0	I_3
1	1	1	1
P	0	1	1
1	0	0	I_2.

Q.5 pgno.58

Sol:-



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

$Z = Y + \bar{C}$

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

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

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

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

floating inputs in the TTL logic family are considered as logic 1.

C	B	A	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

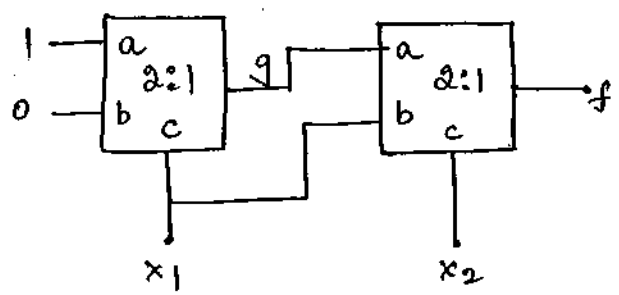
$\therefore Y = \bar{B}\bar{A}(0) + \bar{B}A(1) + B\bar{A}(1) + BA(0)$

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

$Y = A \oplus B$.

Q.6 Pgno.58

Sol:-



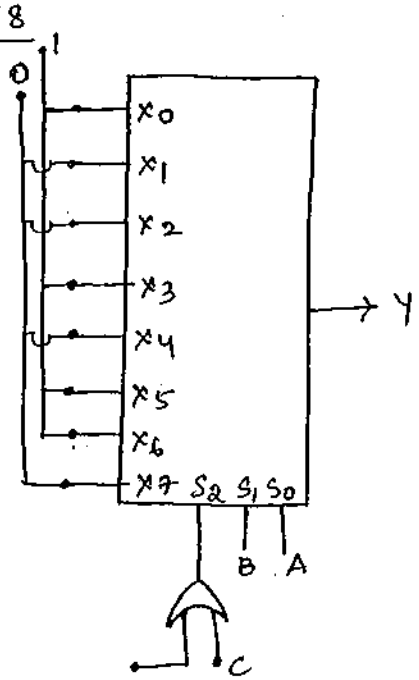
$f = a\bar{c} + bc$.

$g = 1\bar{x_1} + 0x_1 = \bar{x_1}$.

$f = \bar{x_2}g + x_2x_1$.

$= \bar{x_2}\bar{x_1} + x_1x_2$.

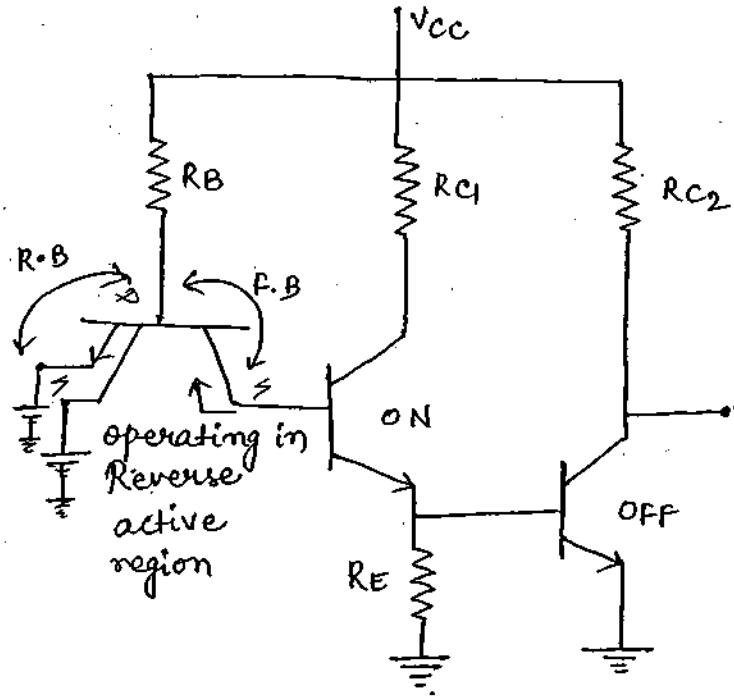
Q.7 Pgno.58



floating logic '1'.

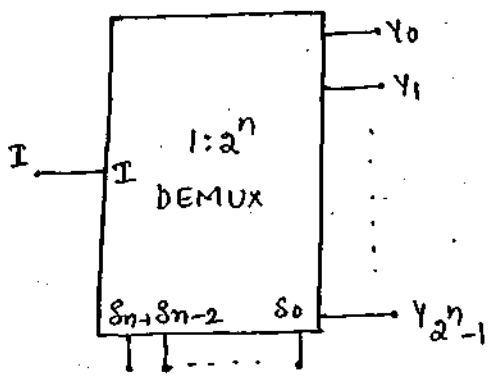
$\therefore S_2 = 1$ always.

Basic TTL logic gate :-

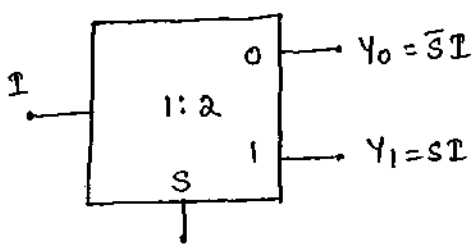


If one of the i/p's is floating then also logic '0' is o/p. So the floating i/p in TTL gate considered as logic '1'.

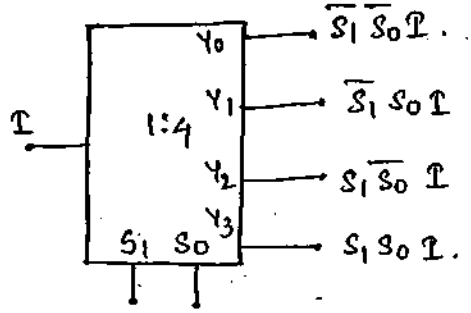
DEMULTIPLEXER :-



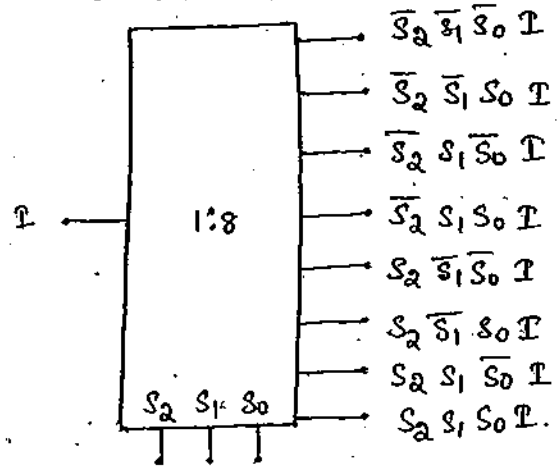
1:2 DEMUX :-



1:4 DEMUX :-

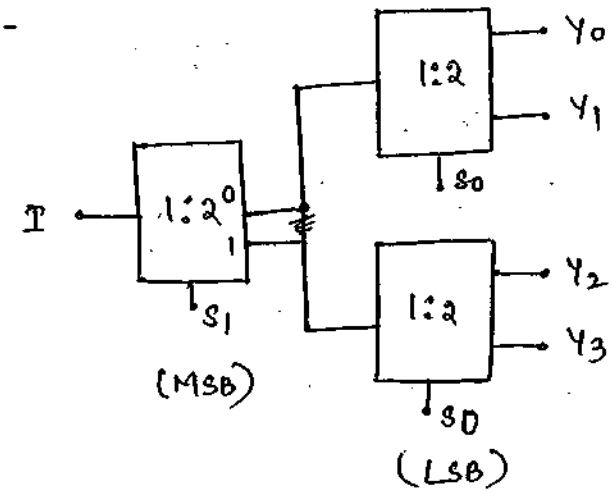


1:8 DEMUX :-



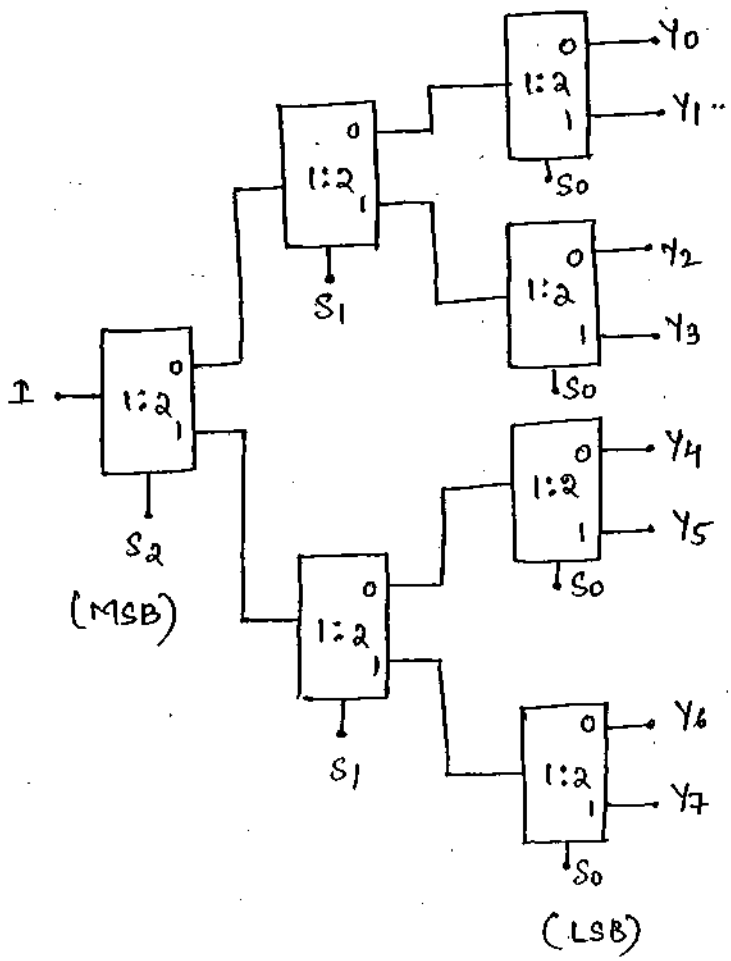
Eg:- Implement 1:4 DEMUX using minimum number of 1:2 DEMUX.

Sol:-

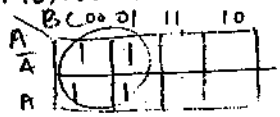


S_1	S_0	
0	0	$Y_0 = I$
0	1	$Y_1 = I$
1	0	$Y_2 = I$
1	1	$Y_3 = I$

Eg:- Implement 1:8 DEMUX using minimum number of 1:2 DEMUX.



Eg:- Implement the function $f(A,B,C) = \sum m(0,1,4,5)$ using 1:8 DEMUX and some combinational gates.

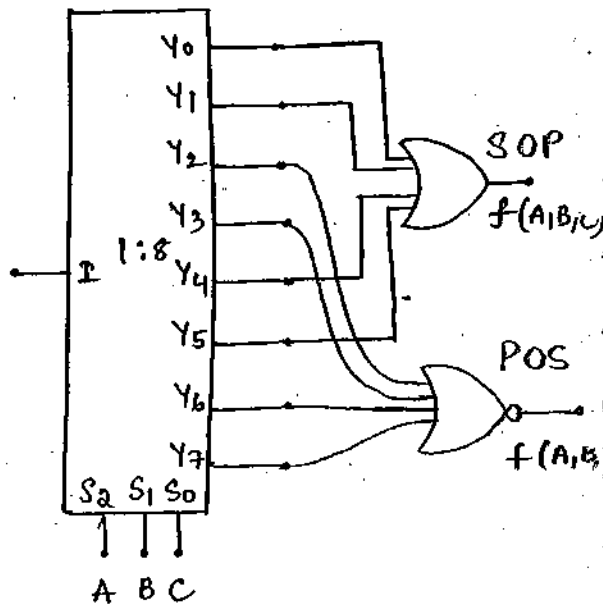


$$f(A,B,C) = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + \bar{A}B\cdot A\bar{B}\bar{C} + A\bar{B}C.$$

$$= \pi M(2,3,6,7).$$

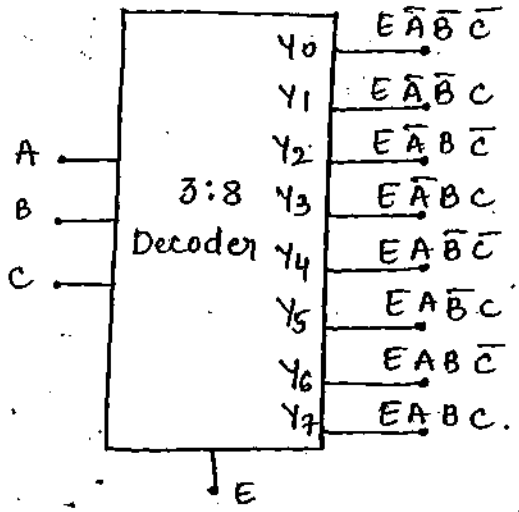
$$= (A+\bar{B}+C)(A+\bar{B}+\bar{C})(\bar{A}+\bar{B}+C)(\bar{A}+\bar{B}+\bar{C}).$$

$$= \overline{\bar{A}B\bar{C} + \bar{A}BC + AB\bar{C} + ABC}$$



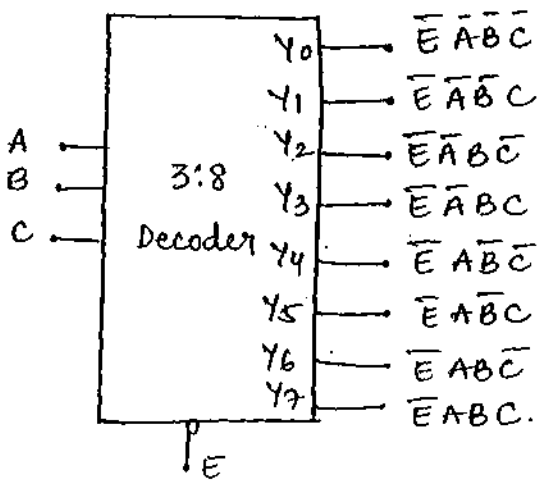
DECODER :- $(n \times 2^n)$.

3:8 Decoder :-



E	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

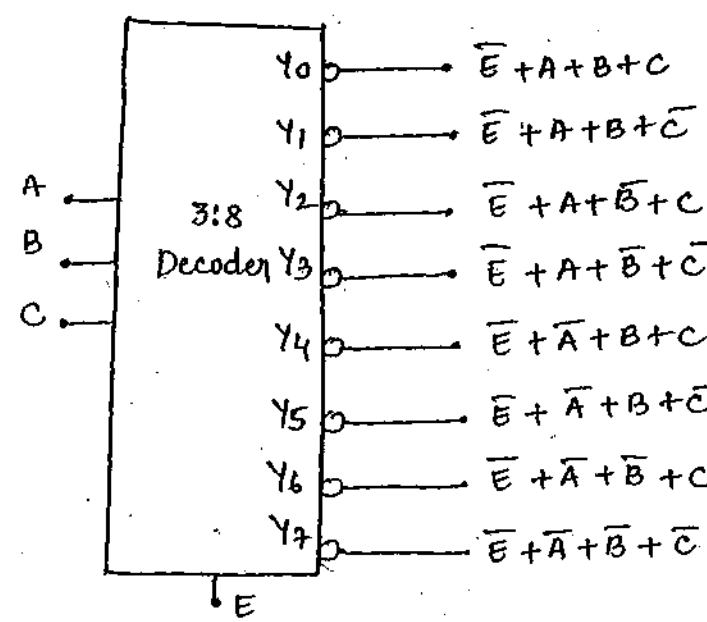
This is for logic high enable decoder block.



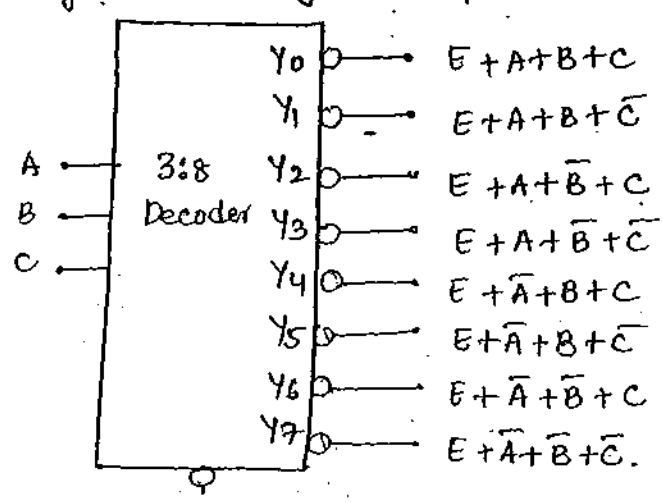
This is logic '0' enable and logic '1' outputs 3x8 decoder.

The truth table for logic '1' enable and logic '0' outputs 3x8 decoder is

E	A	B	C	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
0	x	x	x	1	1	1	1	1	1	1	1
1	0	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	0	1	1	1	1	1	1
1	0	1	0	1	1	0	1	1	1	1	1
1	0	1	1	1	1	1	0	1	1	1	1
1	1	0	0	1	1	1	1	0	1	1	1
1	1	0	1	1	1	1	1	1	0	1	1
1	1	1	0	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	1	0



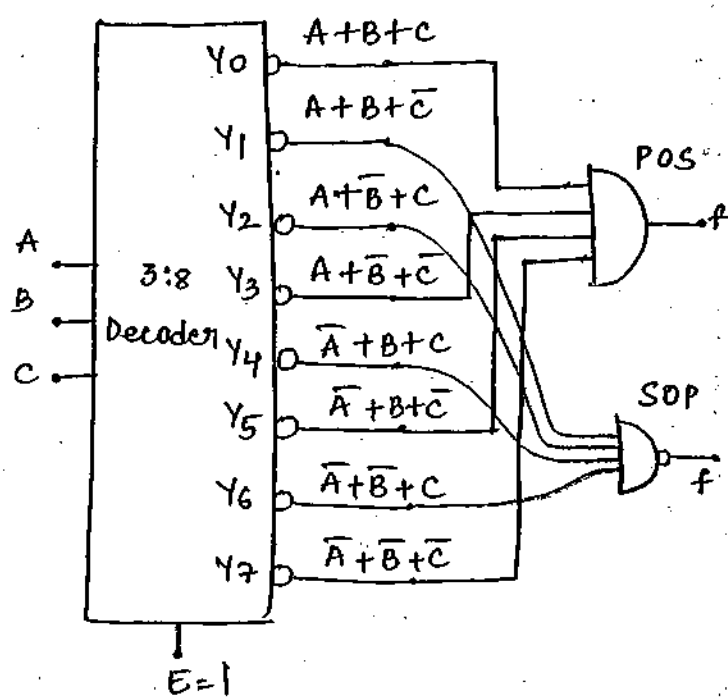
The truthtable and pindiagram for logic '0' and logic '1' o/p 3x8 decoder



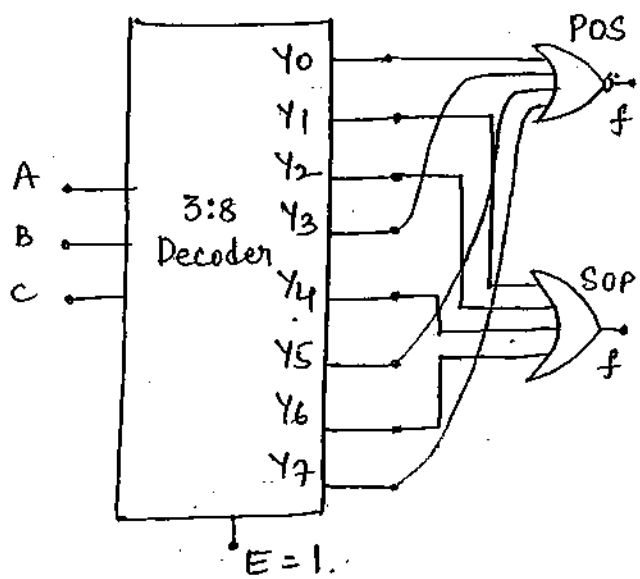
E	A B C	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
1	× × ×	1	1	1	1	1	1	1	1
0	0 0 0	0	1	1	1	1	1	1	1
1	0 0 1	1	0	1	1	1	1	1	1
2	0 1 0	1	1	0	1	1	1	1	1
3	0 1 1	1	1	1	0	1	1	1	1
4	1 0 0	1	1	1	1	0	1	1	1
5	1 0 1	1	1	1	1	1	0	1	1
6	1 1 0	1	1	1	1	1	1	0	1
7	1 1 1	1	1	1	1	1	1	1	0

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

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



Eg:- Implement the function $f(A,B,C) = \sum m(1,2,4,6)$ using 3:8 Decoder and some combinational gates.



$$f(A,B,C) = \sum m(1,2,4,6)$$

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

$$= \pi M(0,3,5,7)$$

$$= \overline{\bar{A}\bar{B}\bar{C} + \bar{A}BC + A\bar{B}C + ABC}$$

ENCODER $(2^n : n)$:-

Mutually exclusive 2^n i/ps to 'n' o/p bits conversion.

I_0	I_1	I_2	I_3	Y_1	Y_0
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

PRIORITY ENCODER :-

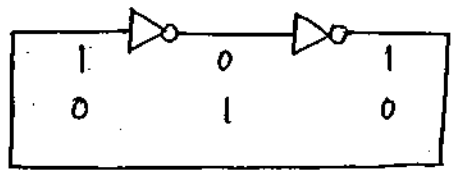
Predefined priority $I_0 > I_1 > I_2 > I_3$

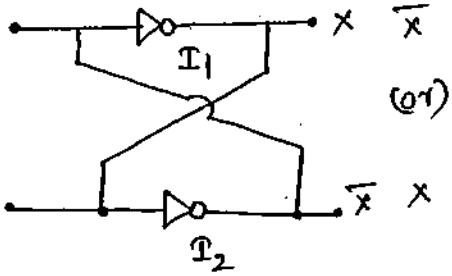
I_0	I_1	I_2	I_3	Y_1	Y_0
1	x	x	x	0	0
0	1	x	x	0	1
0	0	1	x	1	0
0	0	0	1	1	1

For priority $I_0 > I_1 > I_3 > I_2$

I_0	I_1	I_2	I_3	Y_1	Y_0
1	x	x	x	0	0
0	1	x	x	0	1
0	0	x	1	1	1
0	0	1	0	1	0

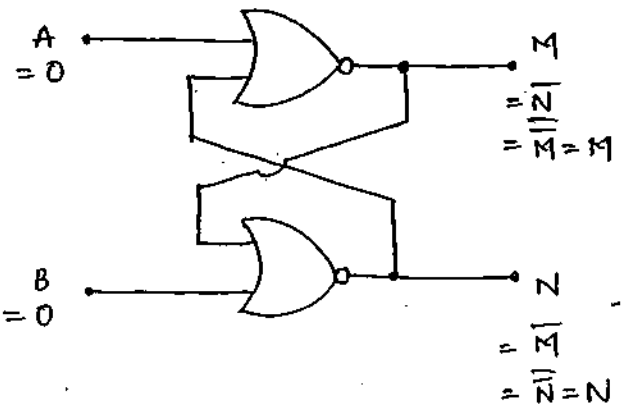
SEQUENTIAL CIRCUITS



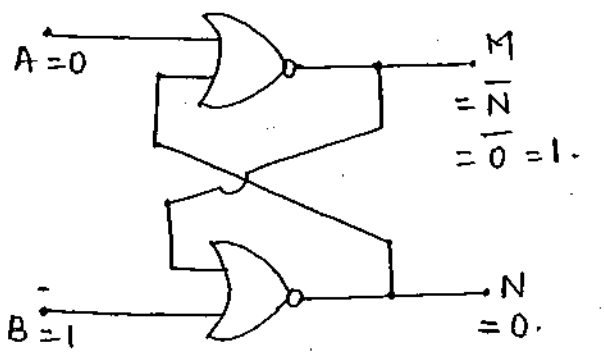


These are basic memory storage elements.

Using NOR gates :-

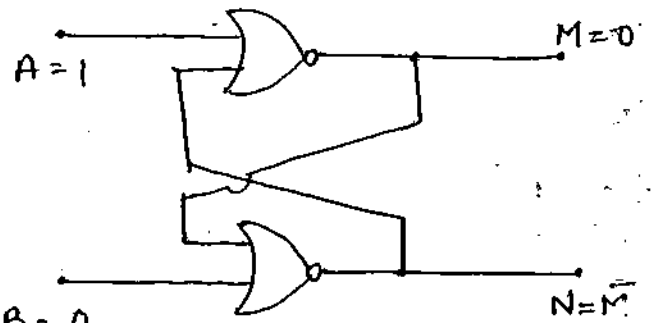


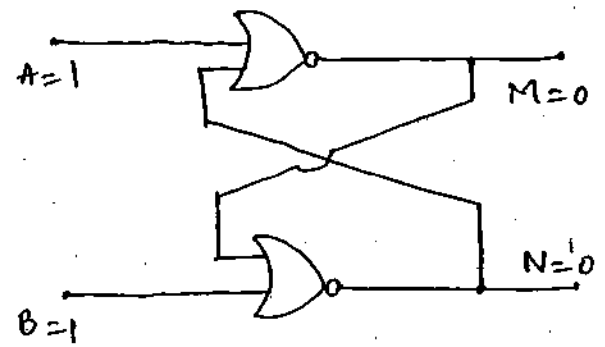
for i/p's A=B=0, the o/p's remains unchanged i.e., stays in previous state.



Disable i/p

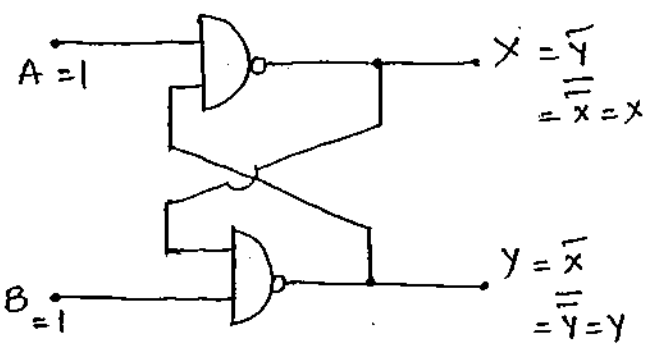
A	B	M	N
0	0	previous	
0	1	1	0
1	0	0	1
1	1	0	0.

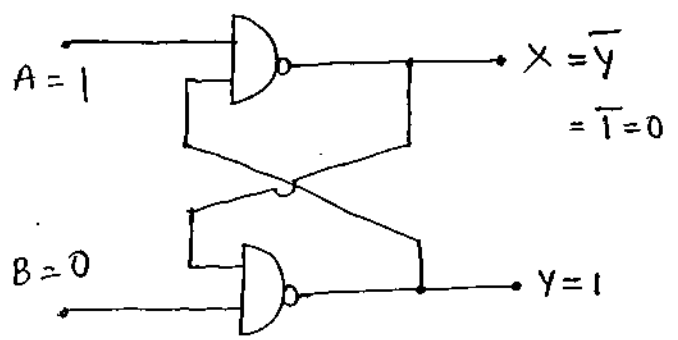


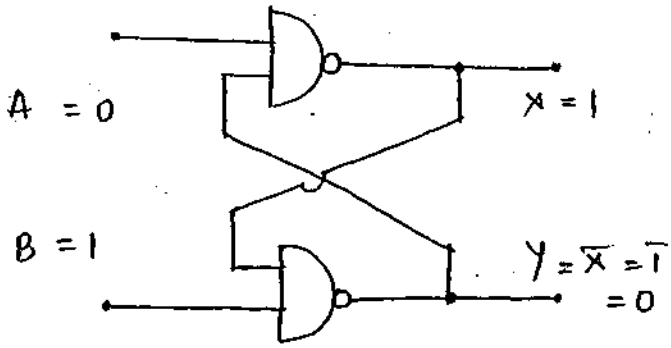


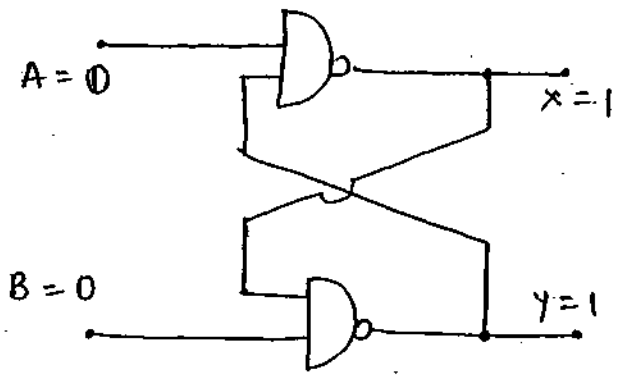
M=0 and N=0 is an indeterminate states. These indeterminate state will create a problem for analysing the outputs of next inputs applied as the output depend on previous outputs.

Using NAND gates :-



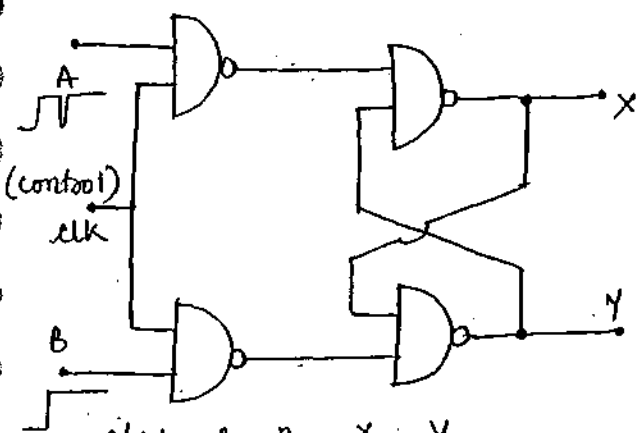






A	B	X	Y
1	1	previous	
0	1	1	0
1	0	0	1
0	0	1	1 → Indeterminate state.

To avoid this effect of noise ~~indeterminate state~~, two more NAND gates are required.



ctrl	A	B	X	Y
0	✗	✗	previous states.	
1	0	0	previous states.	
1	0	1	0	1
1	1	0	1	0
1	1	1	1	1 → Indetermi-

for set, Q = 1

Reset, Q = 0.

Assume X = Q, Y = $\bar{Q}$.

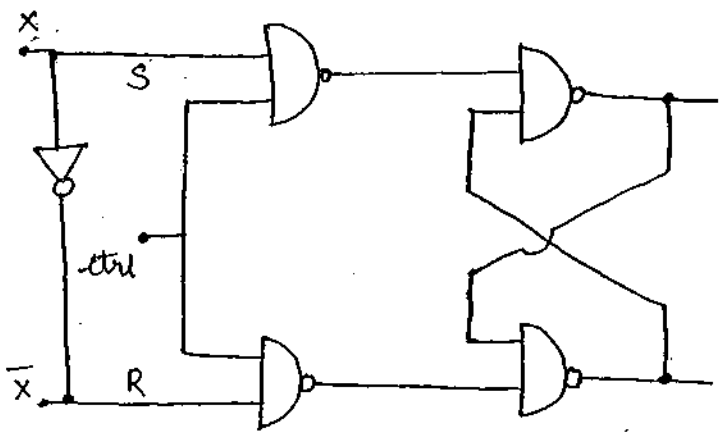
ctrl	S	R	Q	$\bar{Q}$
0	×	×	previous	
1	0	0	previous	
1	0	1	0	1
1	1	0	1	0
1	1	1	1	1

This is the truthtable for S-R latc

Q(n) ⟶ present state

Q(n+1) ⟶ next state.

ctrl	S	R	Q(n+1) / Q(n)
0	×	×	Q(n)
1	0	0	Q(n)
1	0	1	0
1	1	0	1
1	1	1	Indeterminal state.



X = D	Q(n+1)
0	0
1	1

This function of D-latch or D-flipflop.

ctrl	D	Q(n+1)
0	×	Q(n)
1	0	0
1	1	1

The indeterminate state in S-R flipflop is eliminated in J-K flipflop.

J	K	Q(n+1)
0	0	Q(n)
0	1	0
1	0	1
1	1	$\overline{Q(n)}$.

The truthtable for Toggle flipflop is

T	Q(n+1)
0	Q(n)
1	$\overline{Q(n)}$.

Truthtable of S-R flipflop :-

S	R	Q(n)	Q(n+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	×

Excitation table of S-R flipflop :-

Q(n)	Q(n+1)	S	R
0	0	0	×
0	1	1	0
1	0	0	1
1	1	×	0

Characteristic table of D-flipflop:-

D	Q(n+1)
0	0
1	1

Truthtable of D-flipflop :-

D	Q(n)	Q(n+1)
0	0	0
0	1	0
1	0	1
1	1	1

Excitation table of D-flipflop :-

Q(n)	Q(n+1)	D
0	0	0
0	1	1
1	0	0
1	1	1

Characteristic table of J-K flipflop:-

J	K	Q(n+1)
0	0	Q(n)
0	1	0
1	0	1
1	1	$\overline{Q(n)}$.

Truthtable of J-K flipflop :-

J	K	Q(n)	Q(n+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

Excitation ~~characteristic~~ table of J-K flipflop :-

Q(n)	Q(n+1)	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

characteristic table of T-flipflop :-

T	Q(n+1)
0	Q(n)
1	$\overline{Q(n)}$

Truthtable of T-flipflop :-

T	Q(n)	Q(n+1)
0	0	0
0	1	1
1	0	1
1	1	0

Excitation table of T-flipflop :-

Q(n)	Q(n+1)	T
0	0	0
0	1	1
1	0	1
1	1	0

FLIP-FLOP CONVERSIONS :-

ⓐ Write the Truth table for required flipflop. Table must contain column for given flipflop inputs.

ⓑ Flipflop inputs column must b filled with the help of excitation table of given flipflop.

ⓒ Find the expression for given flipfl inputs interms of required flipflop i/p and present state Q(n

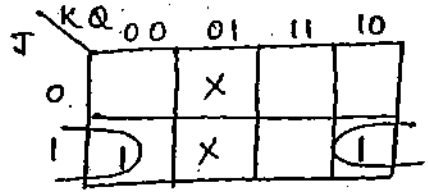
ⓓ Draw the logic diagram.

Conversion of S-R to J-K :-

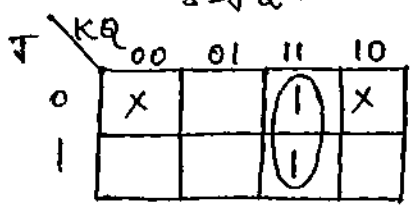
	J	K	Q(n)	Q(n+1)	S	R
0	0	0	0	0	0	x
1	0	0	1	1	x	0
2	0	1	0	0	0	x
3	0	1	1	0	0	1
4	1	0	0	1	1	0
5	1	0	1	1	x	0
6	1	1	0	1	1	0
7	1	1	1	0	0	1

$S = \sum m(4,6) + d(1,5)$.

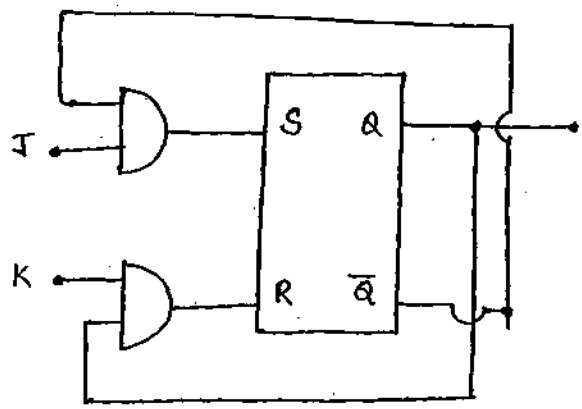
$R = \sum m(3,7) + d(0,2)$.

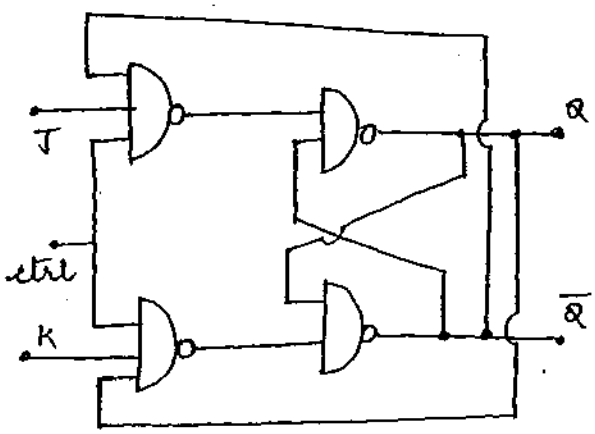


$S = J\bar{Q}$.



$R = KQ$.



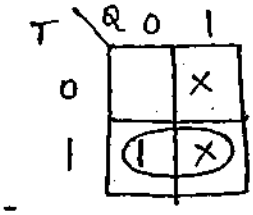


J-K to T-flipflop :-

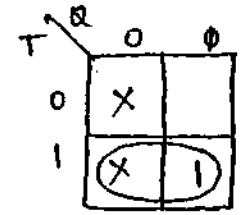
	T	Q(n)	Q(n+1)	J	K
0	0	0	0	0	X
1	0	1	1	X	0
2	1	0	1	1	X
3	1	1	0	X	1

$J = \sum m(2) + d(1,3)$

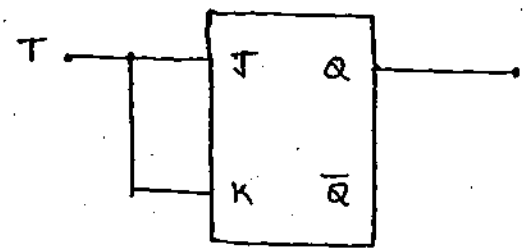
$K = \sum m(3) + d(0,2)$.

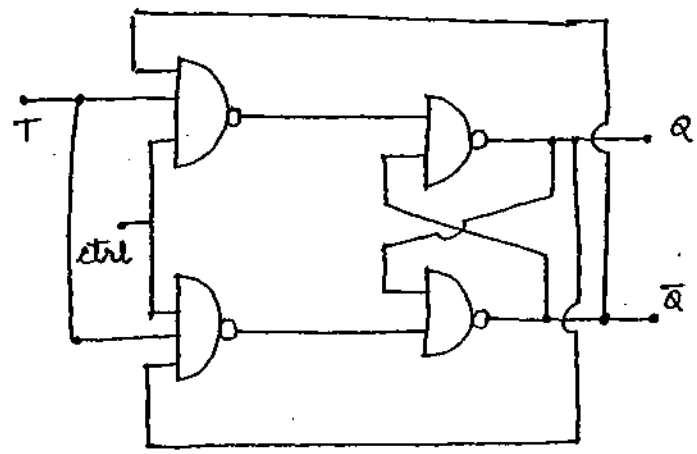


$J = T$



$K = T$.

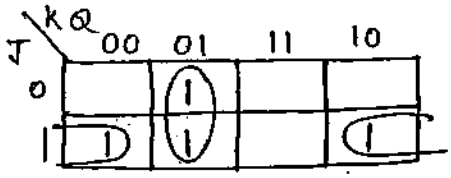




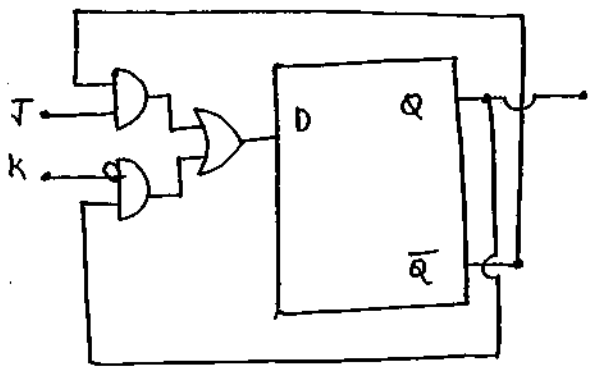
Delay 'D' to J-K flipflop :-

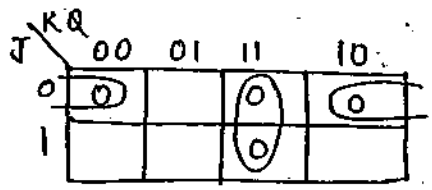
	J	K	Q(n)	Q(n+1)	D
0	0	0	0	0	0
1	0	0	1	1	1
2	0	1	0	0	0
3	0	1	1	0	0
4	1	0	0	1	1
5	1	0	1	1	1
6	1	1	0	1	1

$D = \sum m\,(1,4,5,6)$

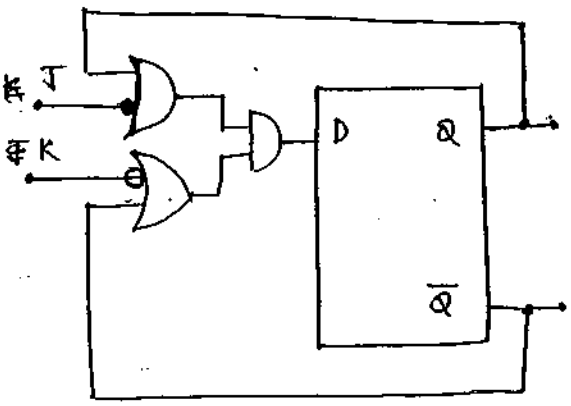


$D = J\bar{Q} + \bar{K}Q$ ⟶ SOP form



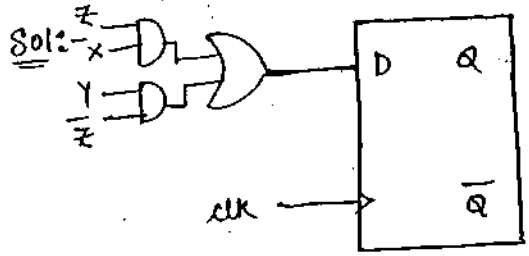


$D = (J+Q)(\bar{K}+\bar{Q})$ ⟶ POS form.



* Q.6 Pgno.78

Sol:-

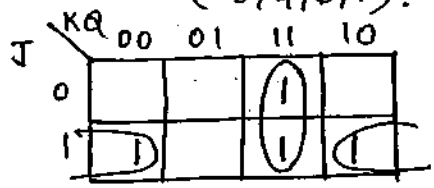


It is JK-flipflop with $Y = J$ and $X = K$, by using verification method.

'T' to J-K flipflop :-

	J	K	Q(n)	Q(n+1)	T
0	0	0	0	0	0
1	0	0	1	1	0
2	0	1	0	0	0
3	0	1	1	0	1
4	1	0	0	1	1
5	1	0	1	1	0
6	1	1	0	1	1
7	1	1	1	0	1

$T = \sum m\,(3,4,6,7)$.

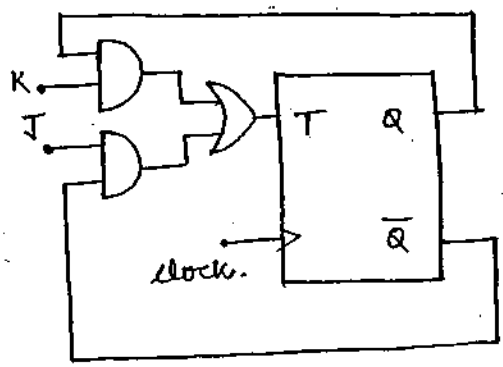


$T = J\bar{Q} + KQ$ ⟶ SOP

In POS

$T = (K+\bar{Q}$

$(J+$



* Q.10 Pgno.84

$T = (J+Q)(K+\bar{Q})$ ⟶ POS fo

$= JK + J\bar{Q} + KQ + 0$

$= J\bar{Q} + KQ + JK$

$= QK + \bar{Q}J + JK$

$= QK + \bar{Q}J$

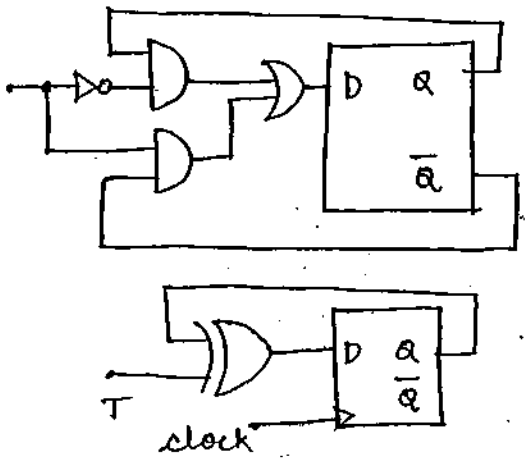
$= J\bar{Q} + KQ.$

Delay to 'T' flipflop :-

	T	Q(n)	Q(n+1)	D
0	0	0	0	0
1	0	1	1	1
2	1	0	1	1
3	1	1	0	0

$D = \sum m(1,2)$

$D = \overline{T}Q + \overline{Q}T.$



* Q.5 pgno. 78

Eg :- $X = T$ → Toggle flipflop.

Eg :- AB flipflop function is defined as shown in table, then construct D and T flipflops from AB.

A	B	Q(n+1)
0	0	0
0	1	$\overline{Q(n)}$
1	0	$\overline{Q(n)}$
1	1	1

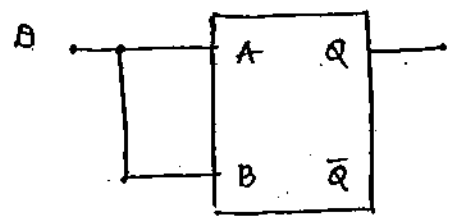
D-flipflop construction,

D	Q(n)	Q(n+1)	A	B
0	0	0		
0	1	0		
1	0	1		
1	1	1		

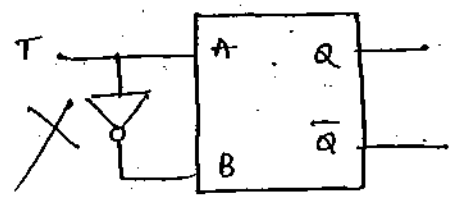
The excitation table cannot be written for AB flipflop.

On observation,

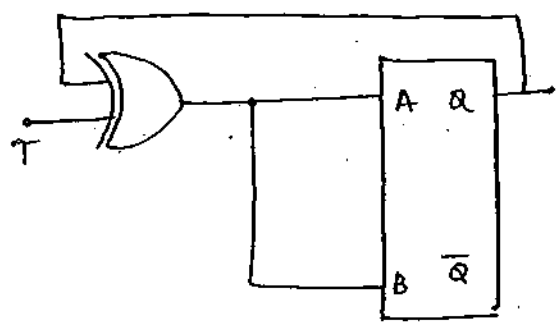
$D = A = B.$



T- flipflop construction,



from D- flipflop, T-flipflop should be constructed.

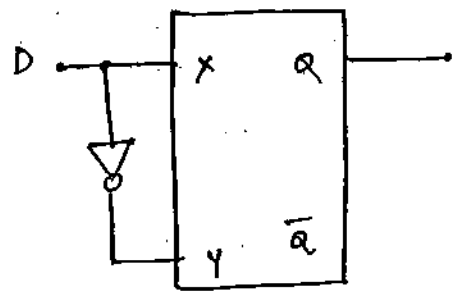


Eg:- The XY flipflop fn is defined as shown in the table. Construct JK from XY.

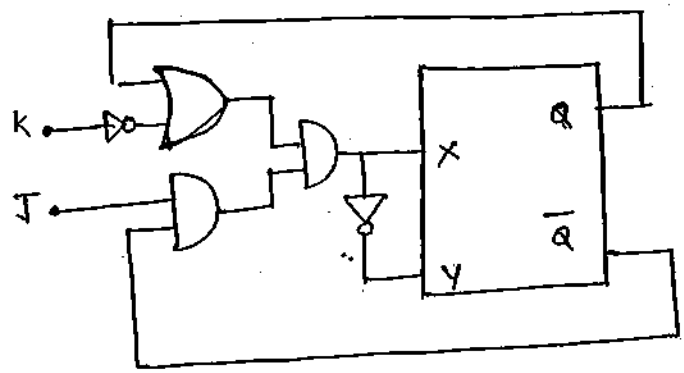
X	Y	Q(n+1)
0	0	$\overline{Q(n)}$
0	1	0
1	0	1
1	1	$\overline{Q(n)}$.

Sol:- Excitation table for XY flipflop cannot be obtained properly.

XY to JK



From 'D' flipflop we can construct J-K flipflop.

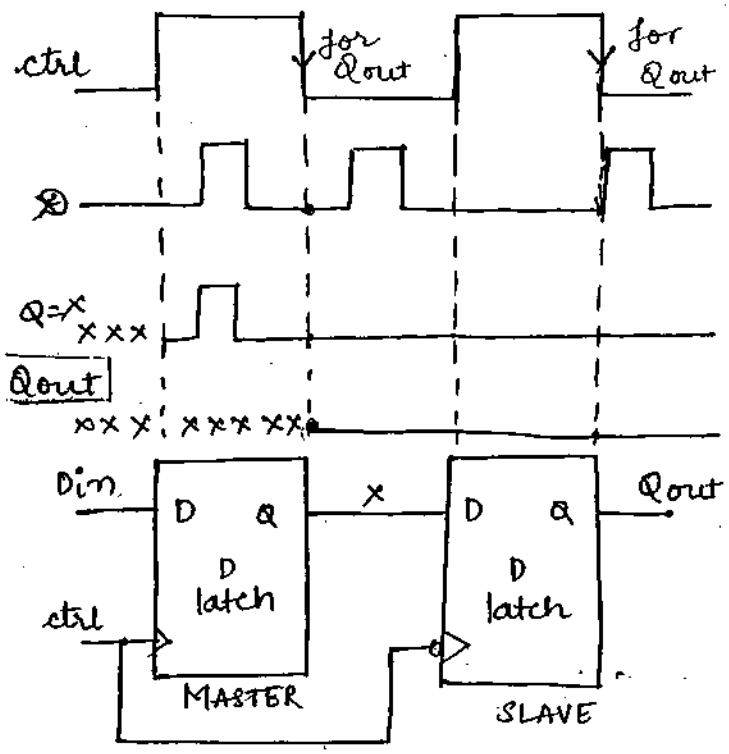


J	K	Q_n	Q_{n+1}	X	Y
0	0	0	0		
0	0	1	1		
0	1	0	0		
0	1	1	0		
1	0	0	1		
1	0	1	1		
1	1	0	1		
1	1	1	0		

D-latch :-

for +ve enable latch

ctrl	D	Q(n+1)
0	x	Q(n)
1	0	0
1	1	1.



logic '1' (+ve) enable latch

logic '0' (-ve) enable latch.

for -ve enable latch

ctrl	D	Q(n+1)
1	x	Q(n)
0	0	0
0	1	1

When master latch is disabled, the output of master latch depends on the i/p at which the master is disabled.

The slave latch is enabled when master is disabled. At this point Qout is changed i.e., entire circuit is responding. So the entire o/p depending on the control at which it is changing from high to low.

Logic '1' enable latch followed by logic '0' enable latch will give -ve edge triggered flipflops.

Logic '0' enable latch followed by logic '1' enable latch will give +ve edge triggered flipflop.

Latch is responding to i/p's depending on level of control. So it is called level sensitive (or) level triggered. Flipflop is responding to i/p's only once in a clock cycle at clock edge. So it is called edge sensitive (or) edge triggered.

After the change of clk edge, the flipflop takes some time to change its state. It is called clock to Q delay (or) flipflop delay. (t_{clk-Q} (or) t_{FF}).

Inputs gets settled before clock edge occurences for some time. The minimum time is "set-up time" (t_{su}).

Clock to Q delay (or) flipflop delay :-

After clock edge occurence flipflop taking some time to change the state is called flipflop delay (or) clock to Q delay.

Setup time :-

The inputs must get settled atleast setup time before clock edge occurence to captured by flipflop.

Eg:- Design

SYNCHRONOUS COUNTER :-

2 Bit Binary counter using D-flipflop :-

(Q_1Q_0)PS (present state)		NS (Next state) (Q_1Q_0)
00		01
01	← for →	10
10	binary	11
11	counter	00

PS is the state before the occuren of clock edge. NS is the state aft the occurence of clock edge.

For the 'D' flipflop,

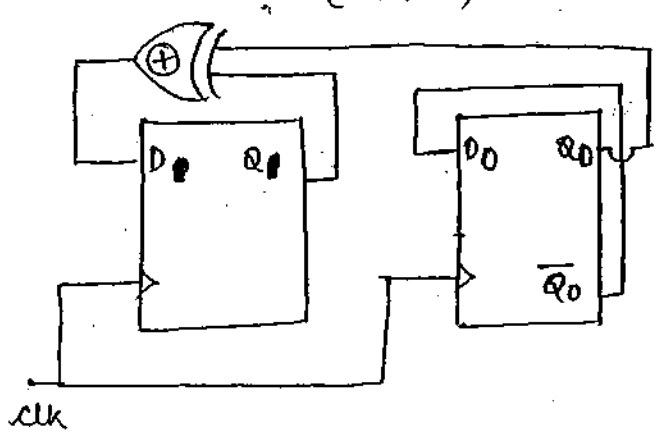
	PS $(Q_1\ Q_0)$	NS $(Q_1\ Q_0)$	FF i/p's $(D_1\ D_0)$
0	0 0	0 1	0 1
1	0 1	1 0	1 0
2	1 0	1 1	1 1
3	1 1	0 0	0 0

present state + i/p's → next stat

$\therefore D_1 = \overline{Q_1}Q_0 + Q_1\overline{Q_0} = Q_0 \oplus Q$

$D_0 = \overline{Q_1}\,\overline{Q_0} + Q_1\overline{Q_0} \neq Q_0 \odot Q$

$(\overline{Q_1} + \overline{Q_1})\overline{Q_0} = \overline{Q_0}$.

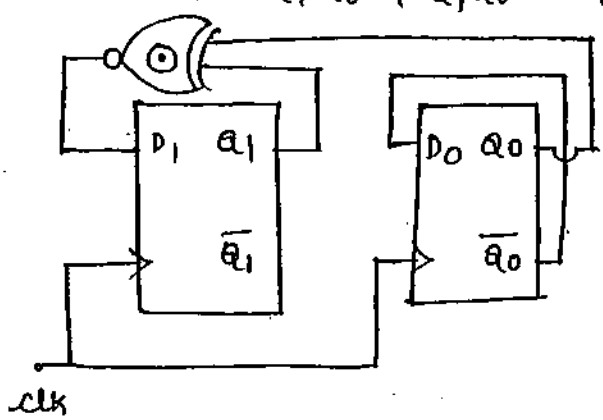


Design 2-bit binary down-counter :- (using D-flipflops).

	PS		NS		FF i/p's	
	Q_1	Q_0	Q_1	Q_0	D_1	D_0
3	1	1	1	0	1	0
2	1	0	0	1	0	1
1	0	1	0	0	0	0
0	0	0	1	1	1	1

$D_1 = Q_1\bar{Q}_0 + \bar{Q}_1\bar{Q}_0 = Q_1 \odot Q_0$

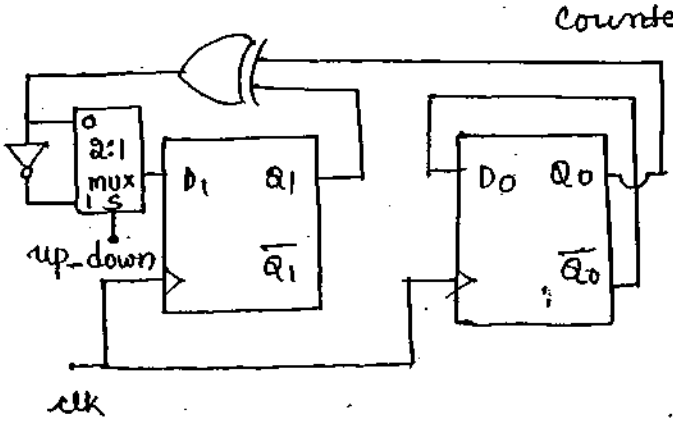
$D_0 = Q_1\bar{Q}_0 + \bar{Q}_1\bar{Q}_0 = \bar{Q}_0$



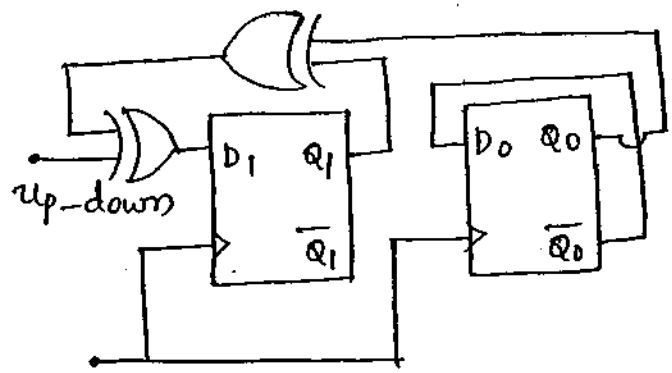
Design 2-bit up/down counter using D-flipflops :-

up_down = 0 ⟶ Up counter

up_down = 1 ⟶ down counter.



Instead of MUX, another X-OR gate also can be used.

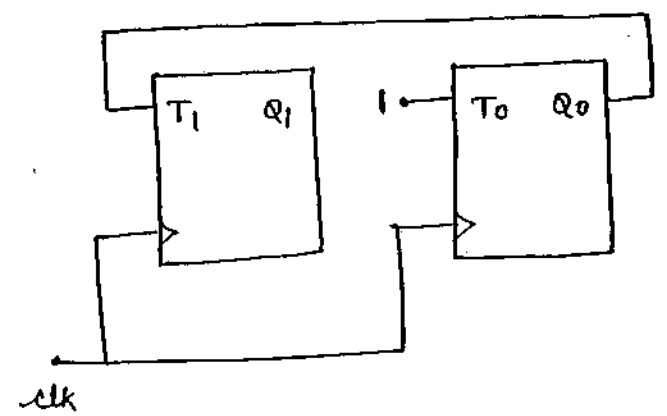


Design 2-bit binary counter using T-flipflop :-

	PS		NS		FF i/p's	
	Q_1	Q_0	Q_1	Q_0	T_1	T_0
0	0	0	0	1	0	1
1	0	1	1	0	1	1
2	1	0	1	1	0	1
3	1	1	0	0	1	1

$T_1 = \bar{Q}_1 Q_0 + Q_1 Q_0 = Q_0.$

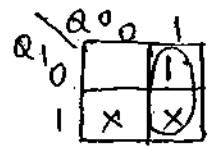
$T_0 = 1.$



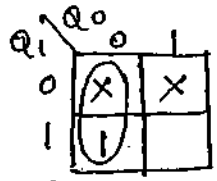
Design 2-bit gray counter using J-K flipflop :-

PS		NS		FF i/p's			
Q_1	Q_0	Q_1	Q_0	J_1	K_1	J_0	K_0
0	0	0	1	0	x	1	x
0	1	1	1	1	x	x	0
1	1	1	0	x	0	x	1
1	0	0	0	x	1	0	x
0	0						

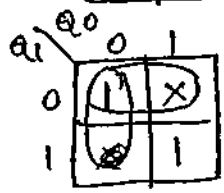
$T_1 = Q_0$.



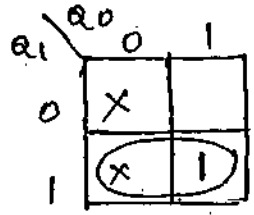
$K_1 = \overline{Q_0}$

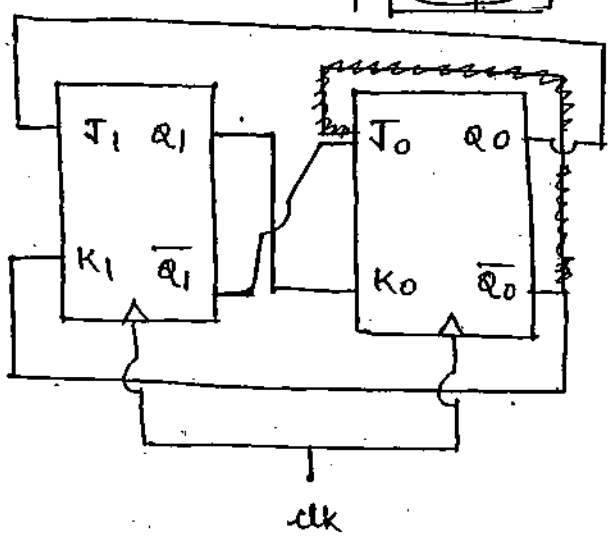


$T_0 = \overline{Q_1}$



$K_0 = Q_1$





Design a counter which is going through the states

$00 \longrightarrow 10 \longrightarrow 11 \longrightarrow 01$

using T-flipflops :-

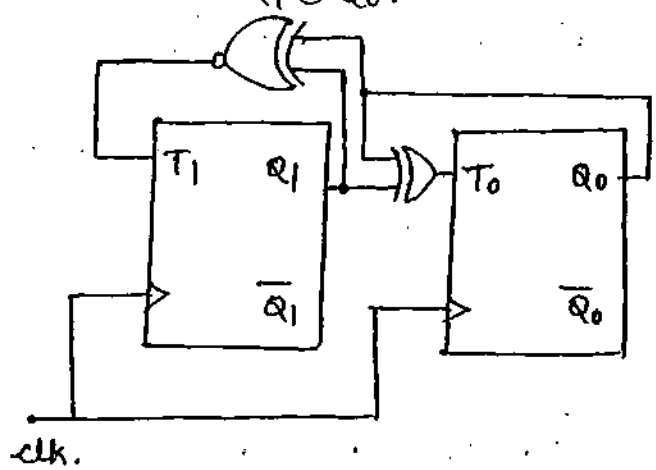
P.S		N.S		FF i/ps	
Q_1	Q_0	Q_1	Q_0	T_1	T_0
0	0	1	0	1	0
1	0	1	1	0	1
1	1	0	1	1	0
0	1	0	0	0	1

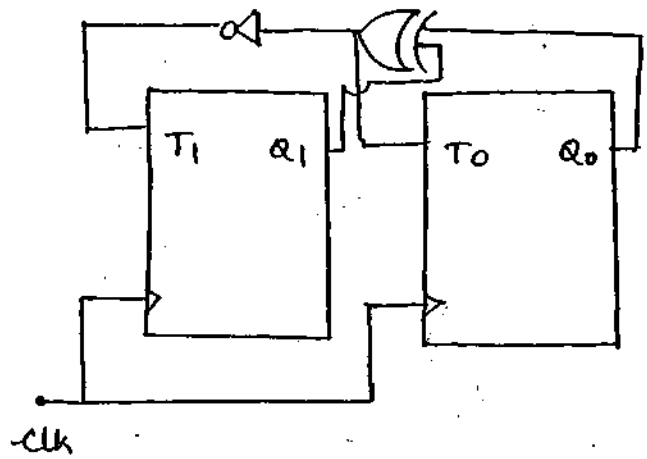
$T_1 = \overline{Q_1}\,\overline{Q_0} + Q_1 Q_0$

$= Q_1 \odot Q_0$.

$T_0 = Q_1\overline{Q_0} + \overline{Q_1} Q_0$

$= Q_1 \oplus Q_0$.





Design a modulo-6 counter using T-flipflops.

PS			NS			FF i/ps		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0	T_2	T_1	T_0
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	0	0	0	1	0	1

$T_2 = \overline{Q_2} Q_1 Q_0 + Q_2 \overline{Q_1} Q_0$. $+d(6,$

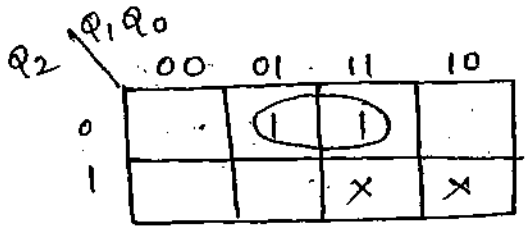
$T_1 = \overline{Q_2}\,\overline{Q_1} Q_0 + \overline{Q_2} Q_1 Q_0$. $+d(6,$

$= \overline{Q_2} Q_0$

$T_n = \Sigma m.(0,1,2,3,4,5) + d(6,7$

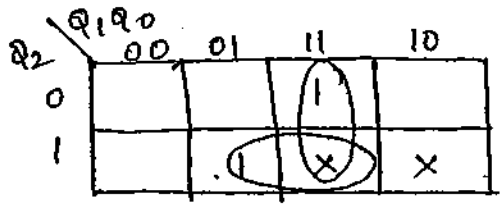
$T_0 = 1$.

for T_1 ;



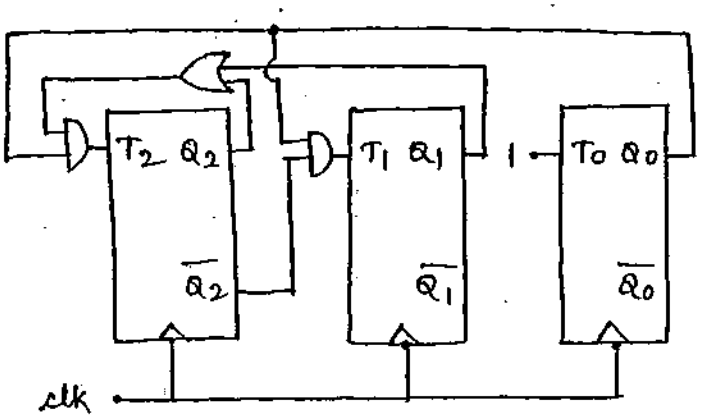
$T_1 = \overline{Q_2} Q_0$

for T_2 ;

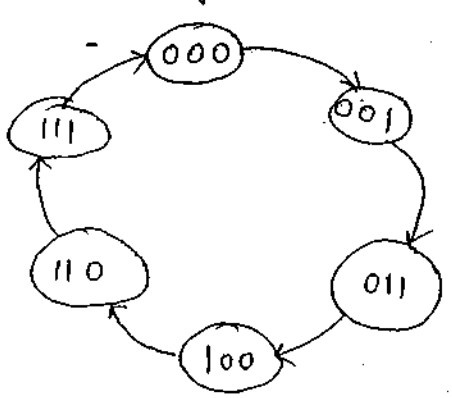


$T_2 = Q_2 Q_0 + Q_1 Q_0$

$T_2 = Q_0 (Q_1 + Q_2)$.

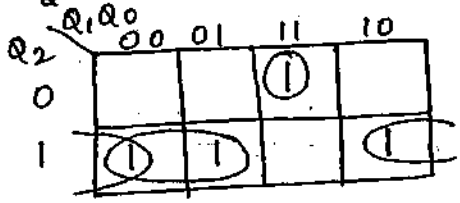


Design a counter which is going through the states specified in the diagram, assuming unused states are initialised to next immediate used state using D-flipflops.



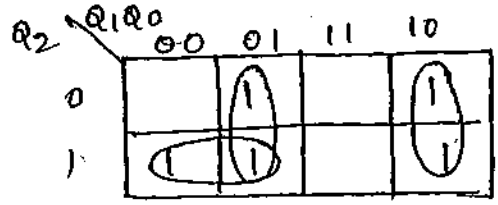
	PS $Q_2\ Q_1\ Q_0$			NS $Q_2\ Q_1\ Q_0$			FF i/p's $D_2\ D_1\ D_0$		
0	0	0	0	0	0	1	0	0	1
1	0	0	1	0	1	1	0	1	1
2	0	1	0	0	1	1	0	1	1
3	0	1	1	1	0	0	1	0	0
4	1	0	0	1	1	0	1	1	0
5	1	0	1	1	1	0	1	1	0
6	1	1	0	1	1	1	1	1	1
7	1	1	1	0	0	0	0	0	0

$D_2 =$



$D_2 = Q_2 \overline{Q_1} + Q_2 \overline{Q_0} + \overline{Q_2} Q_1 Q_0$

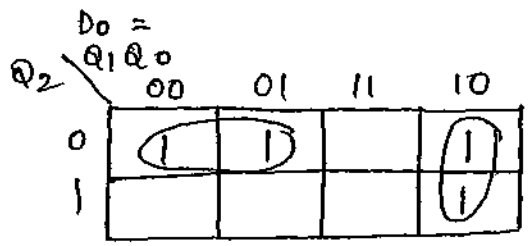
$D_1 =$



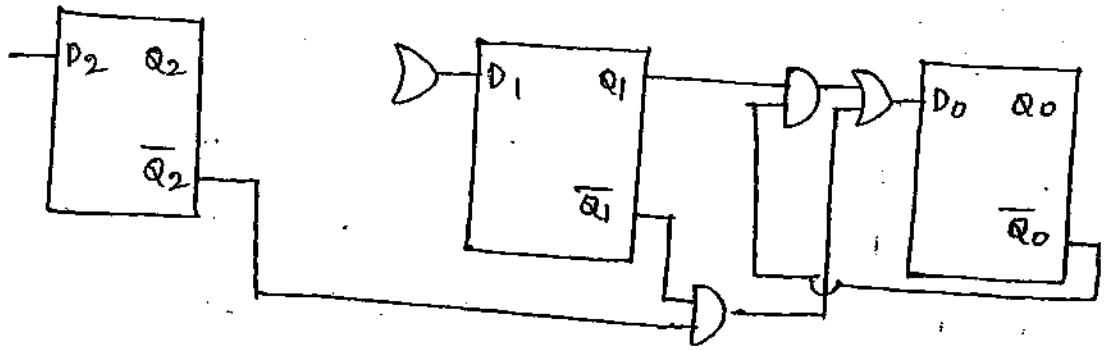
$D_1 = Q_2 \overline{Q_1} + \overline{Q_1} Q_0 + Q_1 \overline{Q_0}$.

$= Q_2 \overline{Q_1} + (Q_1 \oplus Q_0)$.

$D_0 =$

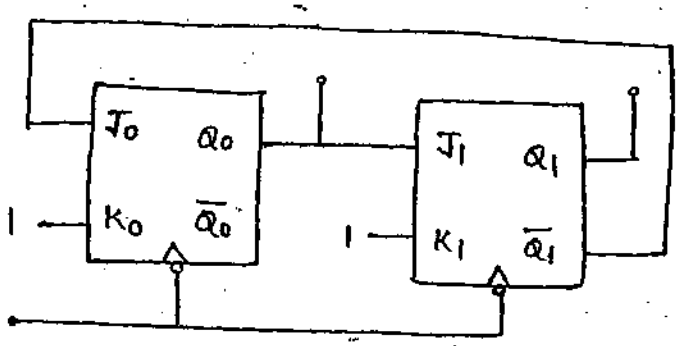


$D_0 = \overline{Q_2} \overline{Q_1} + Q_1 \overline{Q_0}$.



* Q.8 pgno.78

Sol:- mod-k counter.



PS	PI				NS	
Q_1	Q_0	J_1	K_1	J_0	K_0	$Q_1 Q_0$.
		Q_0	1	$\overline{Q_1}$	1	
0	0	0	1	1	1	0 1
0	1	1	1	1	1	1 0
1	0	0	1	0	1	0 0
0	0	0	1	1	1	0 1

} as we are getting 3 states. It is mod-3 counter.

* Q.4 pgno.82

Sol:-

PS			PI			NS		
q_2	q_1	q_0	D_2	D_1	D_0	q_2	q_1	q_0
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	0
0	1	0	1	0	0	1	0	0
1	0	0	0	1	1	0	1	1
0	1	1	1	1	1	1	1	1
1	1	1	1	0	0	1	0	0
1	0	0	0	1	0	0	1	0
0	1	0	1	0	0	1	0	0
1	0	0	0	1	0	0	1	0

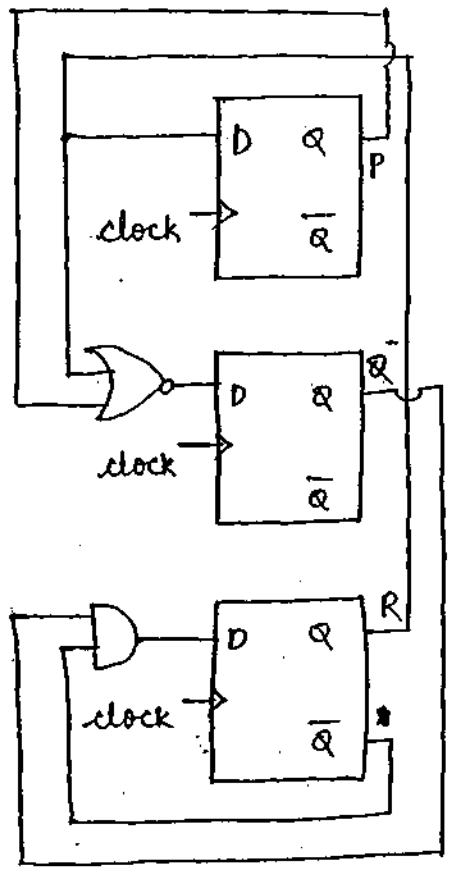
$q_1 (q_2 \oplus q_0)$

Data

* Q.5 and 6 pgno.82

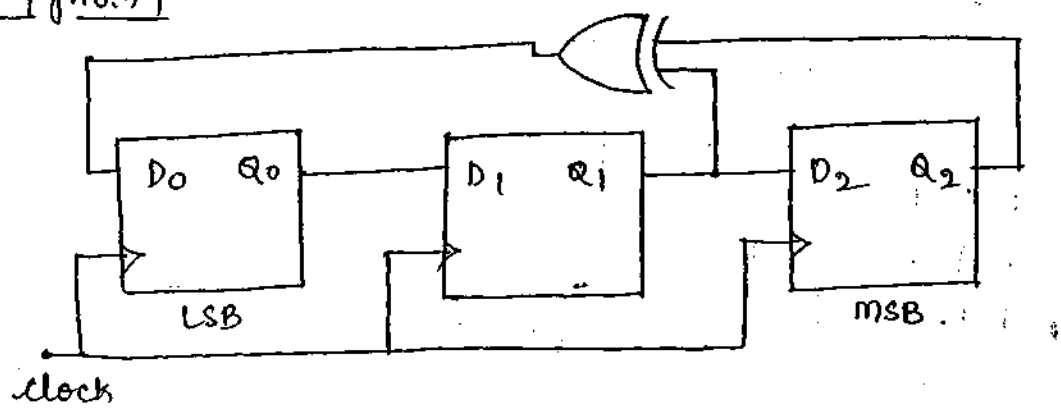
P.S			P.I			N.S		
P	Q	R	D_P	D_Q	D_R	P	Q	R
Q	Q	Q	R	$\overline{P+R}$	$Q\overline{R}$			
0	0	0	0	1	0	0	1	0
0	1	0	0	1	1	0	1	1
0	1	1	1	0	0	1	0	0
1	0	0	0	0	0	0	0	0

4 states



for PS PQR = 010
NS PQR = 011.

* Q.13 pgno.99



P.S			P.I			N.S			State = $4Q_2+2Q_1+Q_0$.
Q_2	Q_1	Q_0	D_2	D_1	D_0	Q_2	Q_1	Q_0	
0	0	1	Q_1	Q_0	$Q_1 \oplus Q_2$				= 0+0+1 = 1.
0	0	1	0	1	0	0	1	0	= 0+2+0 = 2
0	1	0	1	0	1	1	0	1	= 4+1 = 5.
1	0	1	0	1	1	0	1	1	= 2+1 = 3.
0	1	1	1	1	1	1	1	1	= 4+2+1 = 7.
1	1	1	1	1	0	1	1	0	= 4+2 = 6
1	1	0	1	0	0	1	0	0	= 4.

* Q.14 pgno.79.

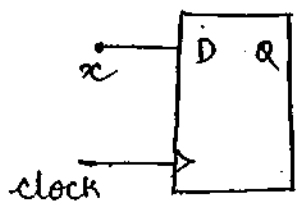
Sol:-

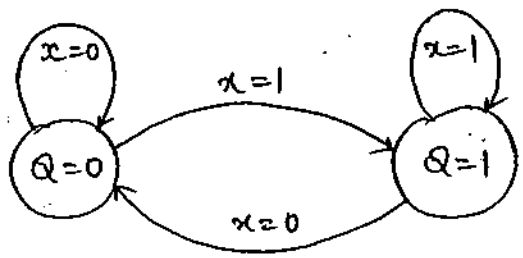
PS			PI			NS		
Q_A	Q_B	Q_C	D_A	Q_B	D_C	Q_A	Q_B	Q_C
(0	0	0)	$\overline{Q_B \oplus Q_C}$	Q_A	Q_B.			
0	0	0	$\overline{0}=1$	0	0	$\overline{0}=1$	0	0
1	0	0	1	1	0	1	1	0
1	1	0	0	1	1	0	1	1
0	1	1	1	0	1	1	0	1
1	0	1	0	1	0	0	1	0
0	1	0	0	0	1	0	0	1
0	0	1	0	0	0	0	0	0

count sequence at Q_A is 01101000

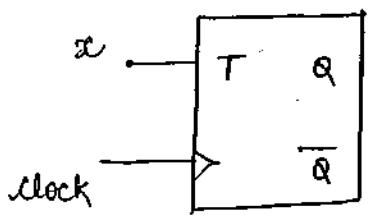
STATE DIAGRAM :-

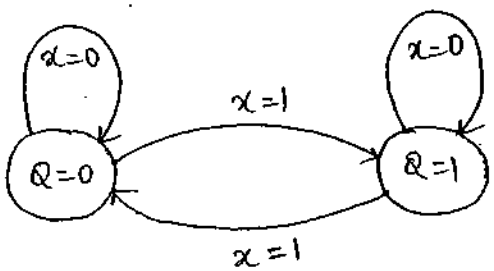
State diagram of D-flipflop :-



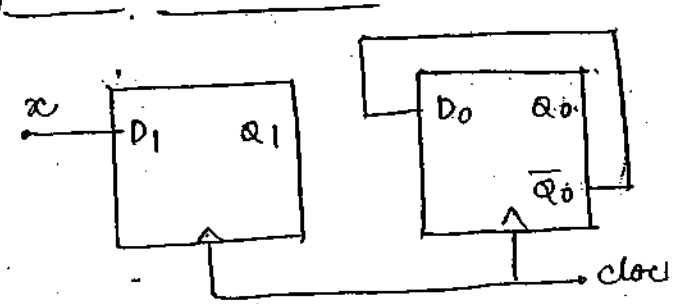


State diagram of T-flipflop :-

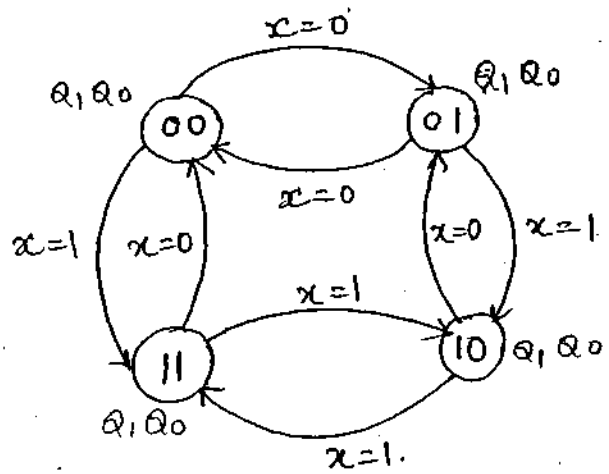




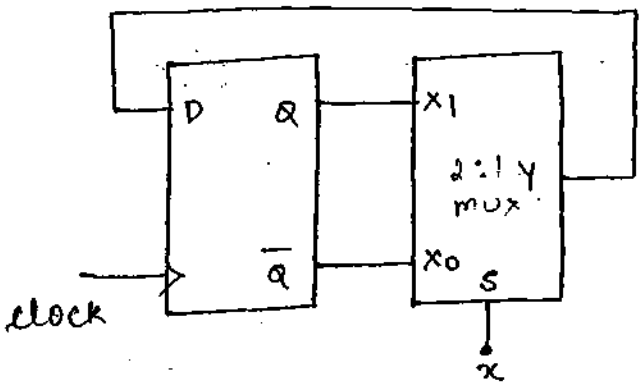
Draw the state diagram w.r.t x for the circuit :-



Next state $\longrightarrow x\overline{Q_0}$.

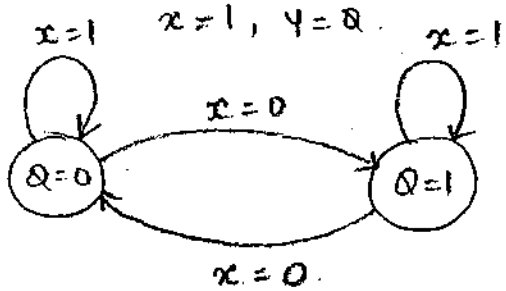


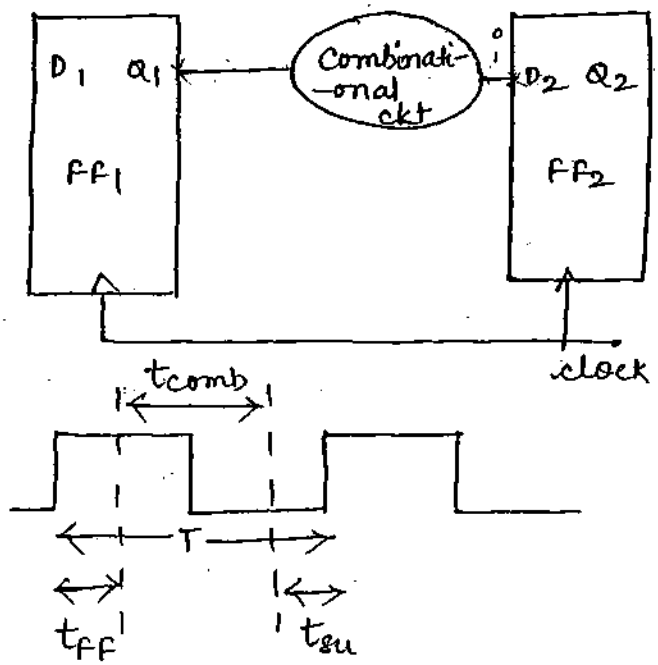
Eg:- Draw the state diagram w.r.t x.



$x = 0,\ Y = \bar{Q}$

$x = 1,\ Y = Q$





$T_{min} = t_{FF} + t_{comb} + t_{su}$

(t_{FF}: (FF1), t_{su}: (FF2))

$$f_{max} = \frac{1}{T_{min}}.$$

t_{FF} → time delay for the response

t_{su} → set-up time for inputs.

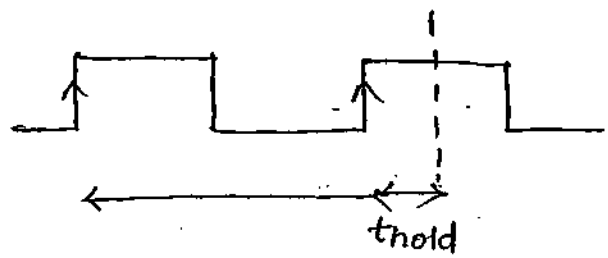
t_{comb} → time delay due to combinational circuit.

t_{hold} → minimum time required to change the inputs.

$$t_{ck-Q} + t_{comb} \geq t_{hold}$$

(FF1) (FF2)

t_{hold} is to allow the flipflop to respond to the previous input.



Eg:- $t_{ck-Q} = 1ns$
$t_{su}/t_h = 1ns/1ns$ } for FF1
$t_{comb} = 2ns$
$t_{ck-Q} = 2ns$
$t_{su}/t_h = 2ns/2ns$ } for FF2

Find f_{max}.

Sol:- $T_{min} = 1ns + 2ns + 2ns$

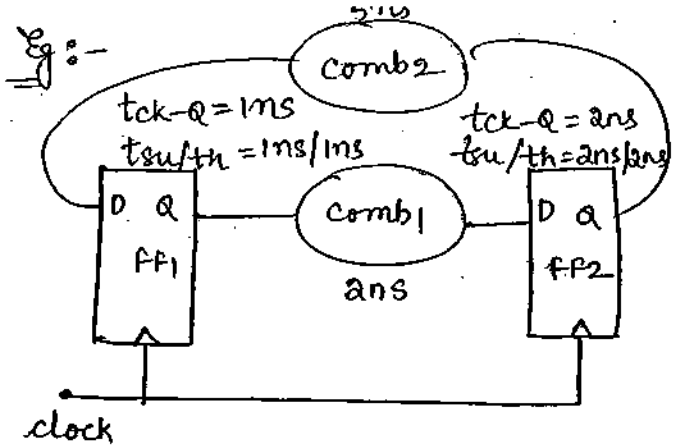
$= 5ns.$

$$f_m = \frac{1}{T_{min}}$$

$f_{max} = 0.2 \times 10^{+9}$

$= 200\ MHz$

Eg:-



$(t_{ck+Q} + t_{comb} \geq t_{hold})$

find max. freq. at which ckt works profierly.

Sol:- FF1 ⟶ FF2

$T_{min_1} = 1ns + 2ns + 2ns$

$= 5ns$

FF2 ⟶ FF1.

$T_{min_2} = 2ns + 5ns + 1ns$

$= 8ns.$

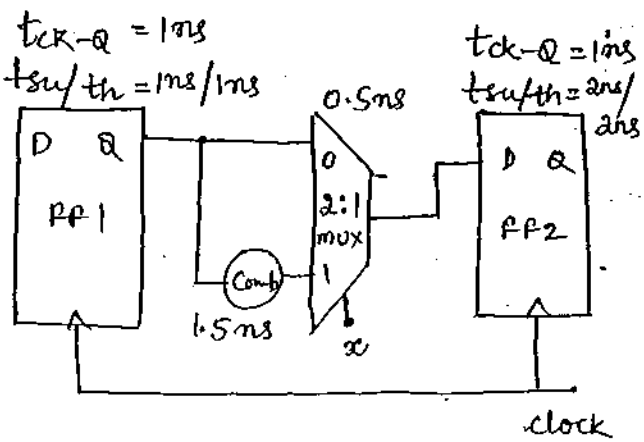
$\therefore T_{min} = 8ns.$ i.e., $T_{min} = \max\{T_{min1}, T_{min2}\}$.

$f_{max} = \frac{1}{T_{min}}$

$= \frac{1}{8ns}$

$= 125\ MHz.$

Eg:-



find max clock frequency.

for $x = 0$,

$t_{min} = 1ns + 0.5ns + 2ns$

$= 3.5\ ns.$

for $x = 1$,

$t_{min} = 1ns + 1.5ns + 0.5ns$

2ns

$= 5ns.$

$\therefore t_{min} = \max\{3.5ns, 5ns\}$

$= 5ns$

$f_{max} = \frac{1}{5ns}$

$= 0.2 \times 10^9$

$= 200\ MHz.$

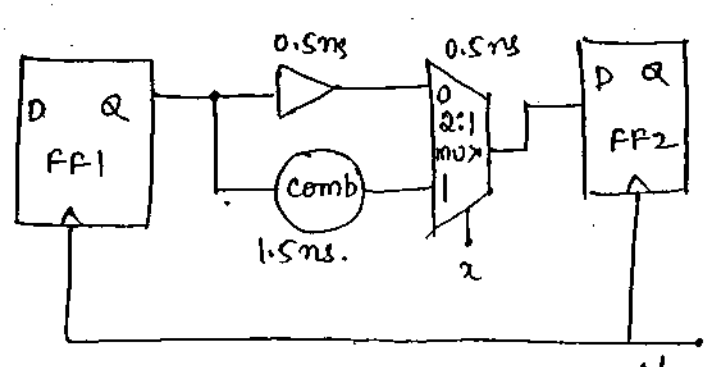
for $x = 0$,

$t_{ck+Q} + t_{comb} \geq$ thold conditi is not satisfied.

$1ns + 0.5ns \not\geq 2ns$

In this case we have to add buffer in the path which shoul have minimum of 0.5ns delay

for $x = 1$, the hold condition is satisfied. $1ns + 1.5 + 0.5 > 2ns.$



for $x = 0$, $t_{min} = 1ns + 0.5ns +$ $0.5ns + 2ns$

for x=1,

$t_{min} = 5ns$.

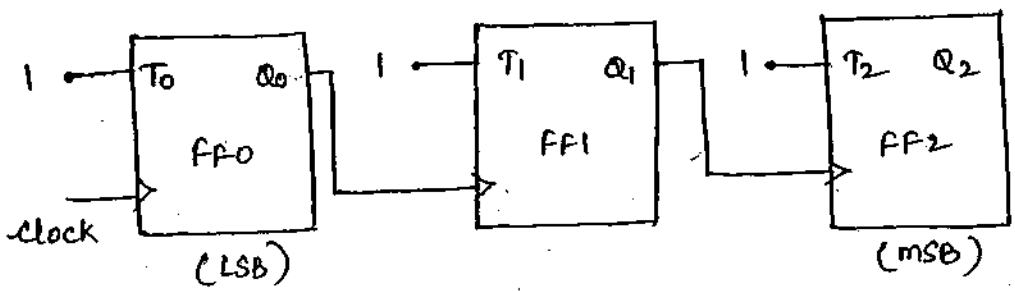
$\therefore f_{max} = \frac{1}{t_{min}}$

$= \frac{1}{5ns}$

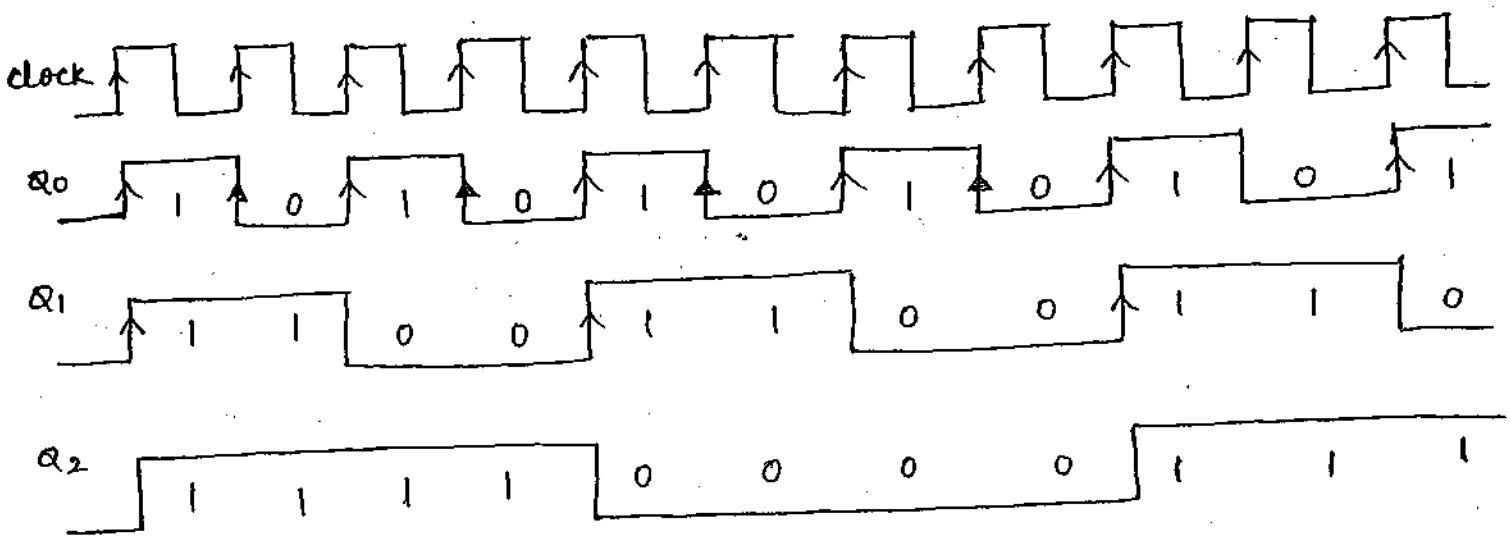
$= 200\ MHz$.

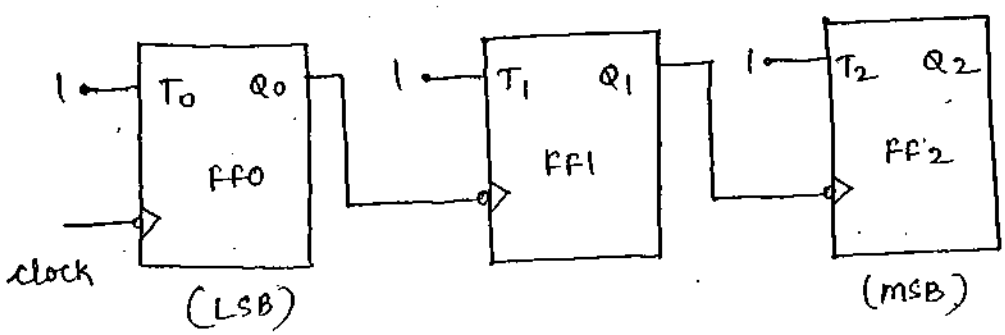
ASYNCHRONOUS SEQUENTIAL CIRCUIT :-

(RIPPLE COUNTERS).

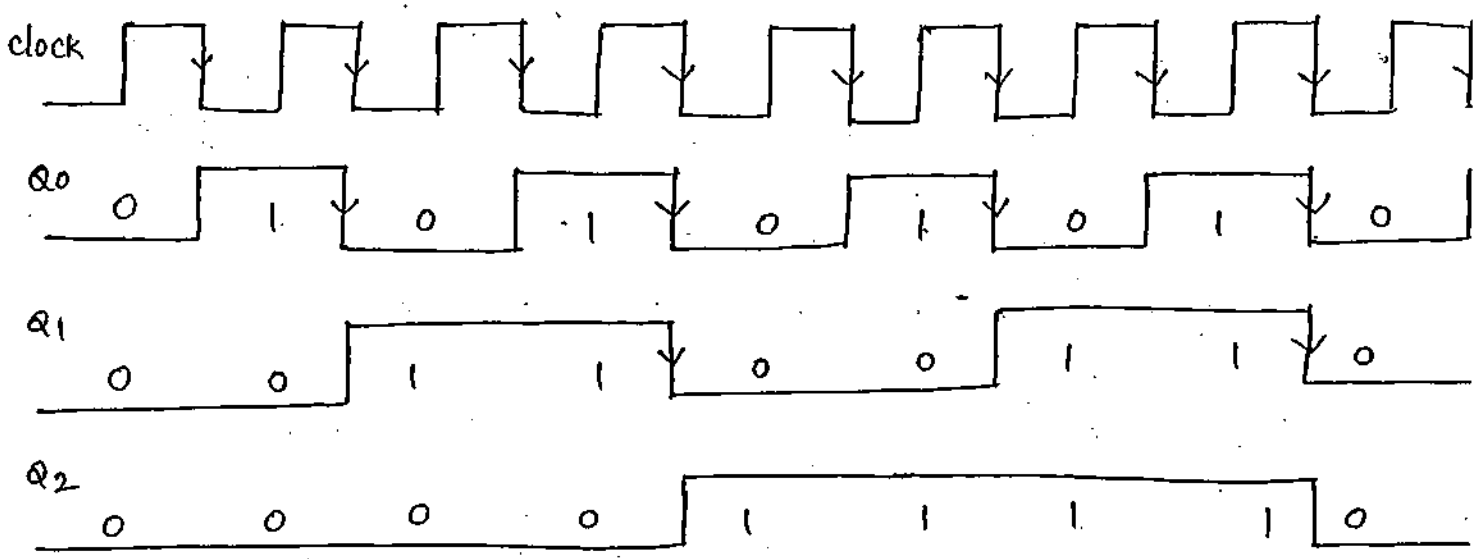


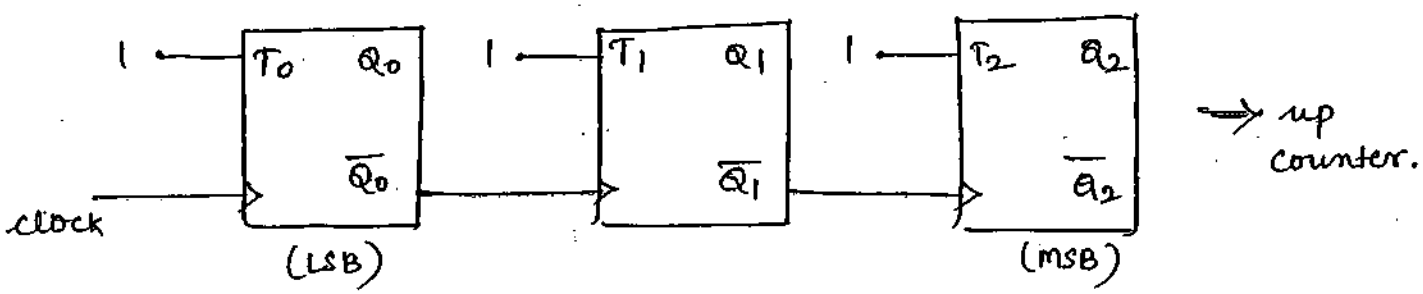
⇒ Down counter



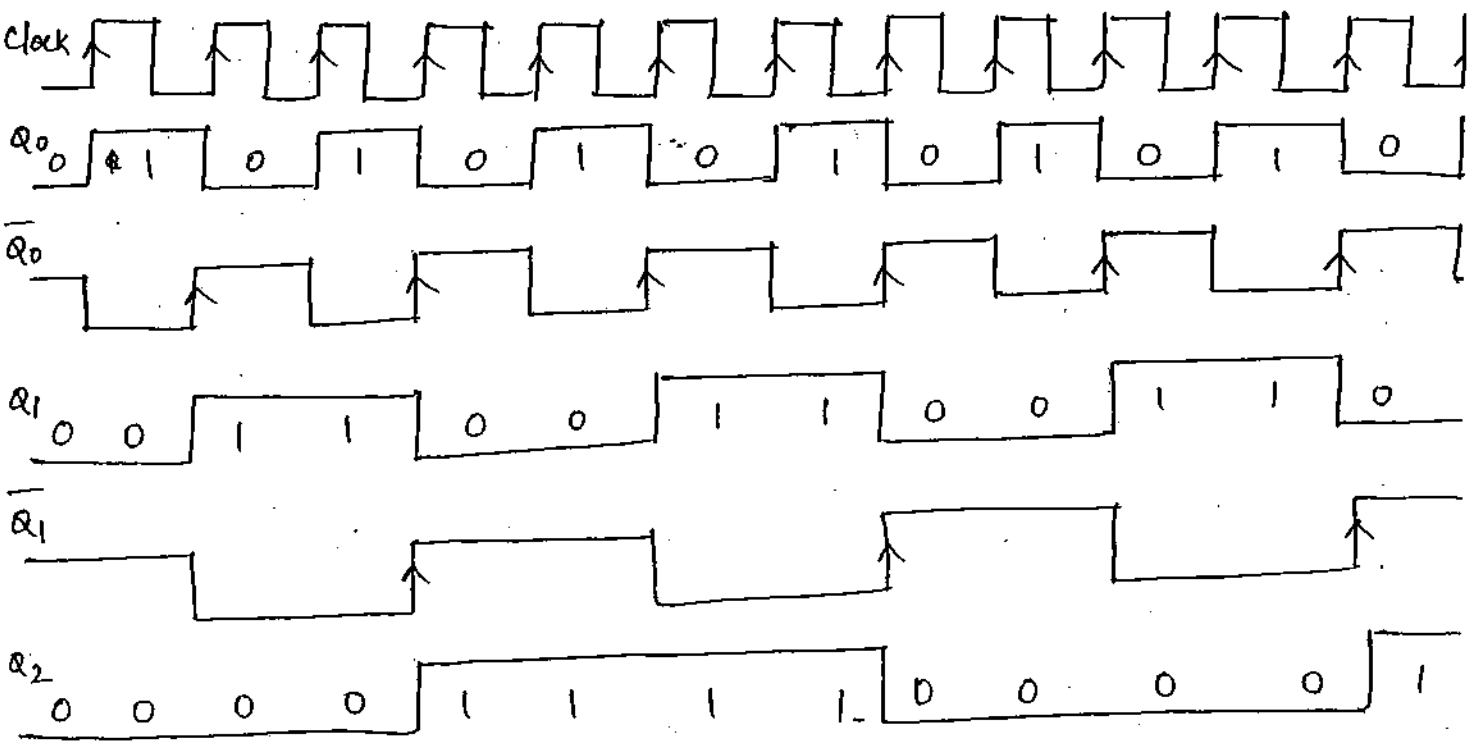


⇒ up counter.

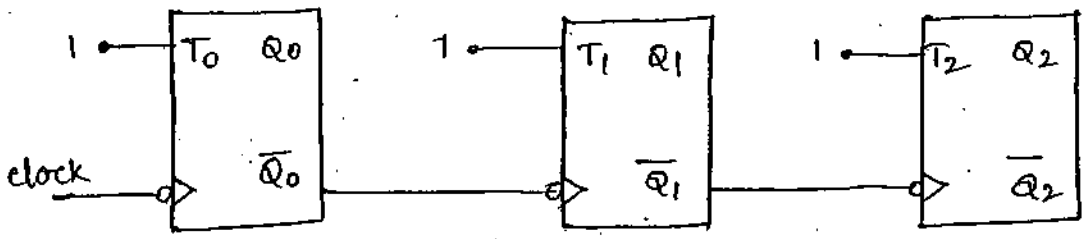




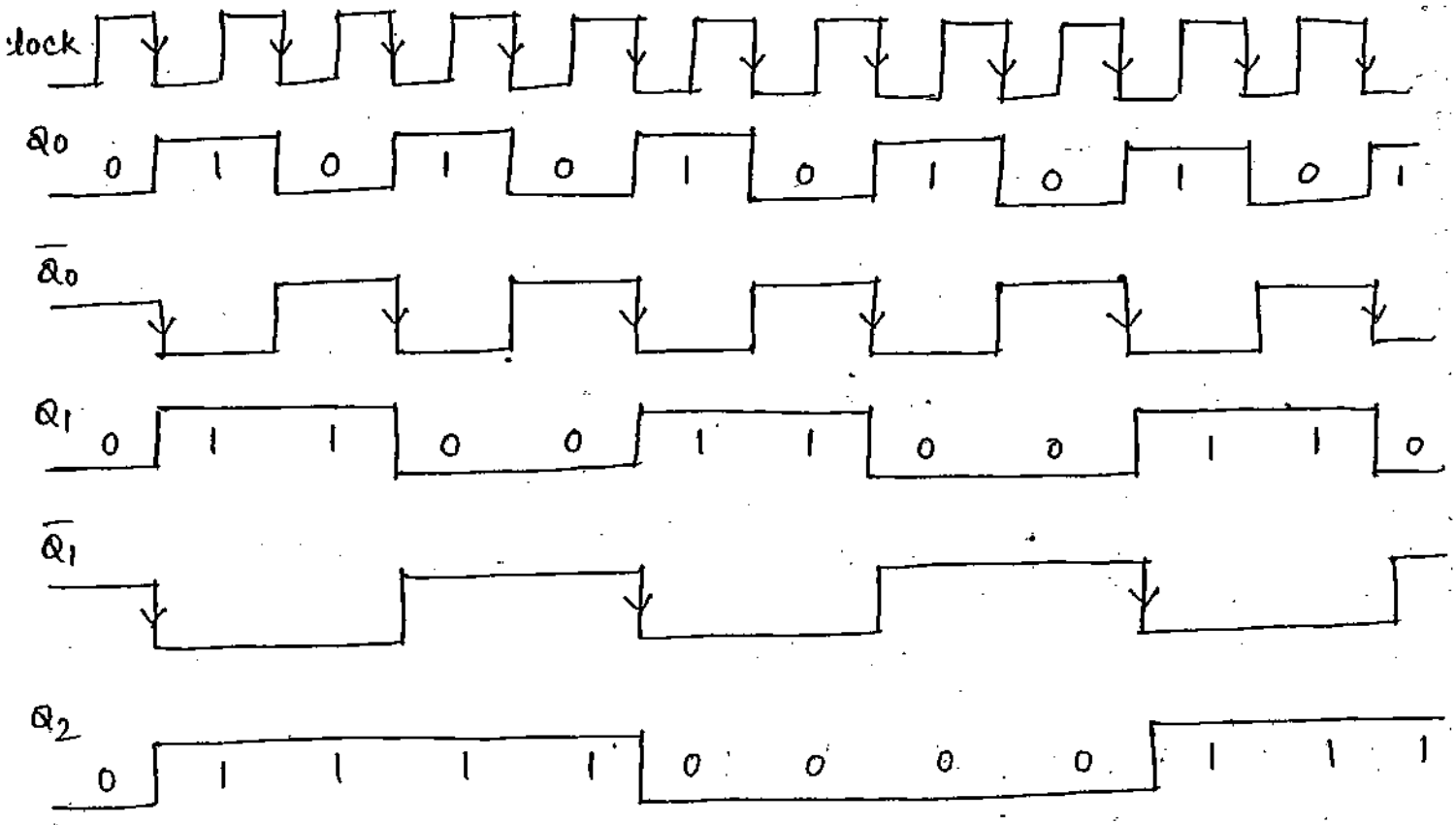
flipflops are +ve edge triggered flipflops.



Instead of +ve edge of $\bar{Q}_0$, -ve edge of Q_0 also can be considered



⇒ Down counter.



NOTE :-

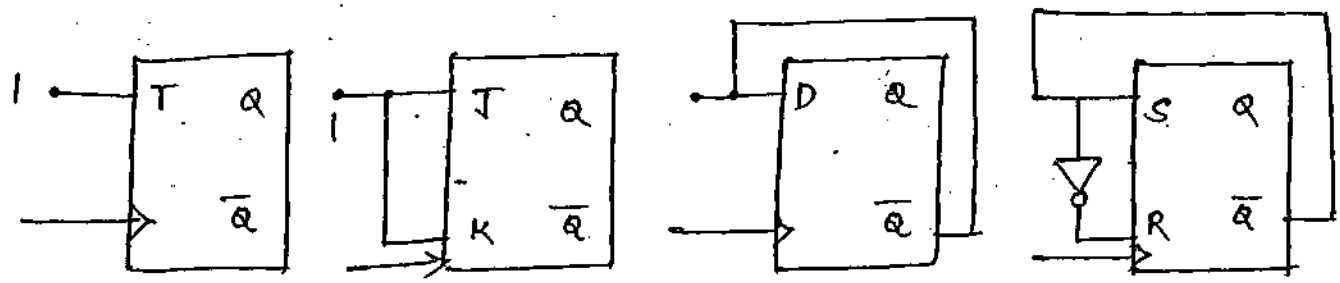
ⓐ positive edge triggered flipflops passing 'Q' as clock to the next stages results down counter.

ⓑ -ve edge triggered flipflops passing 'Q' as a clock to the next stages results up counter.

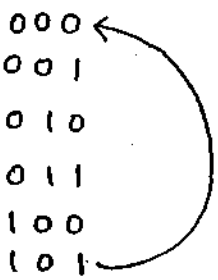
ⓒ +ve edge triggered flipflops passing $\overline{Q}$ as clock to the next stages results up counter.

ⓓ -ve edge triggered flipflops passing $\overline{Q}$ as a clock to the next stages results down counter.

Instead of T-flipflop, J-K flipflops also can be used in asynchro counters for J=K=1. D-flipflop also can be used by $D=\overline{Q}$.



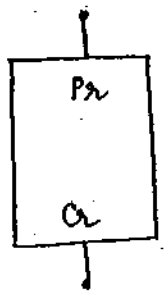
Eg:- Design mod-6 asynchronous up counter. using J-K flipflops.



Preset pin $\Rightarrow Q=1$.
clear pin $\Rightarrow Q=0$.
} Both should not be applied enabled at same time, it remains in indeterminate state.

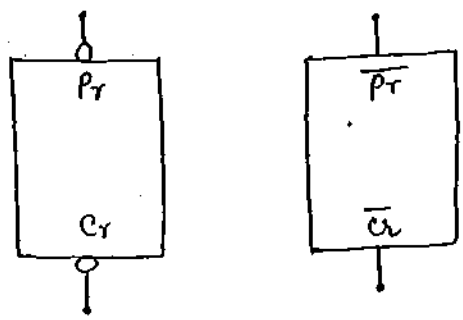
enable

Logic-'1' preset and clear :-



Pr	Cr	function
0	0	normal ff.
0	1	Q=0.
1	Q	Q=1.
1	1	Indeterminate state.

Logic-0 enable preset and clear :-

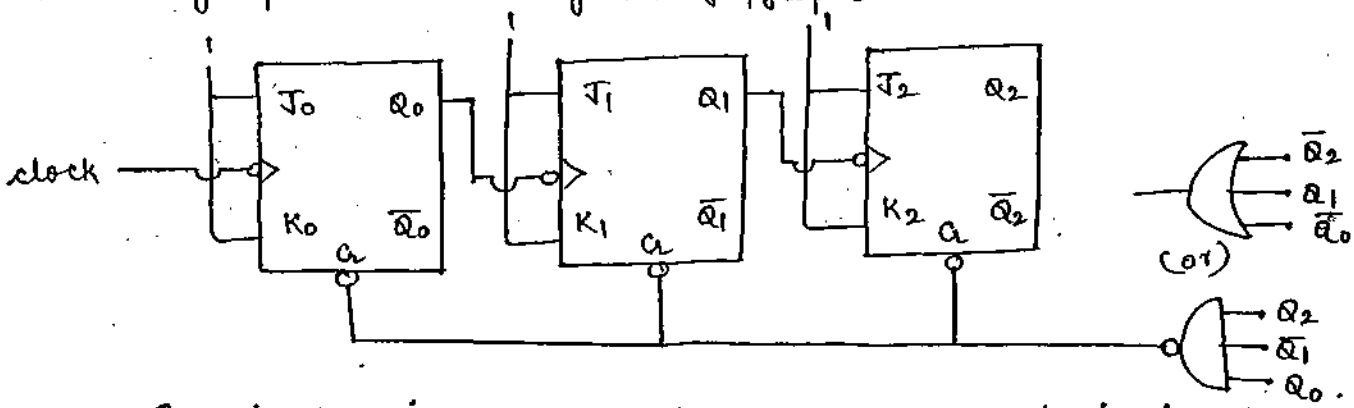


Pr	Cr	function
0	0	Indeterminate state.
0	1	Q=1
1	0	Q=0
1	1	normal FF function

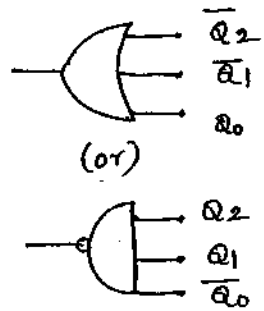
mod-6 counter :

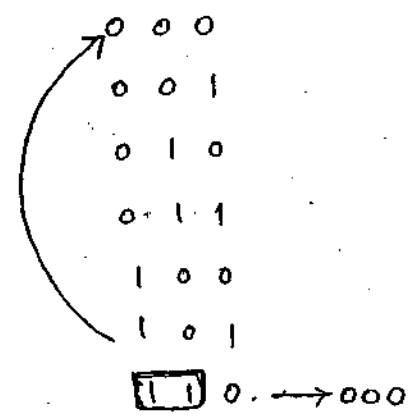
0 0 0
0 0 1
0 1 0
0 1 1
1 0 0
1 0 1

→ Binary up-counter using JK-flipflop :-

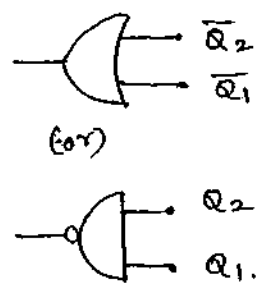


In this 101 is not counted, as soon as 101 obtained, asynchronously the o/p is cleared. So it acts as mod-5 counter. Inorder to count 101 also, next state 110 is used to clear the o/p, so that it acts as mod-6 counter.

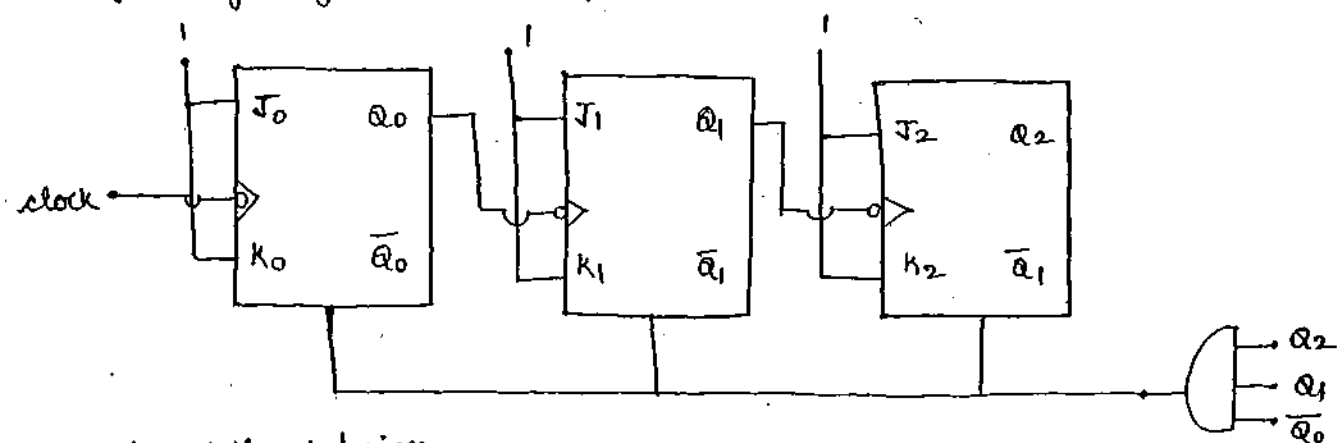




Optimally the gates can be used as,



→ By using logic '1' enable pins,

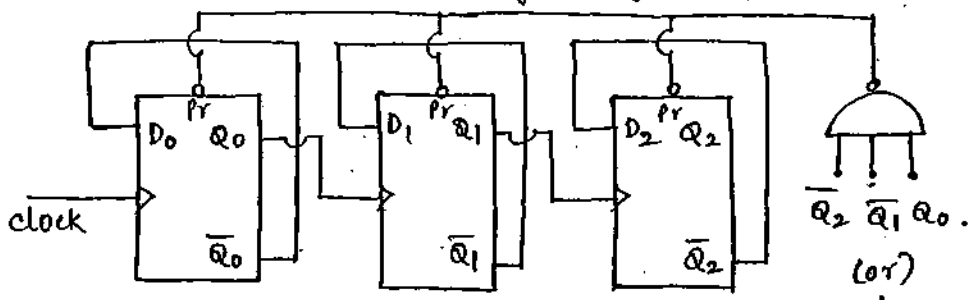


for optimal design,

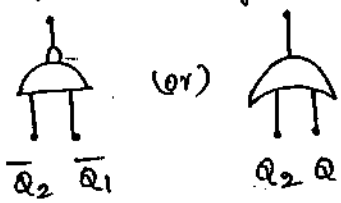
Eg: Design mod-6 asynchronous down counter using delay flipflops.

Sol:-

1 1 1
1 1 0
1 0 1
1 0 0
0 1 1
0 1 0
0 0 1 → 111.



For the optimal design,

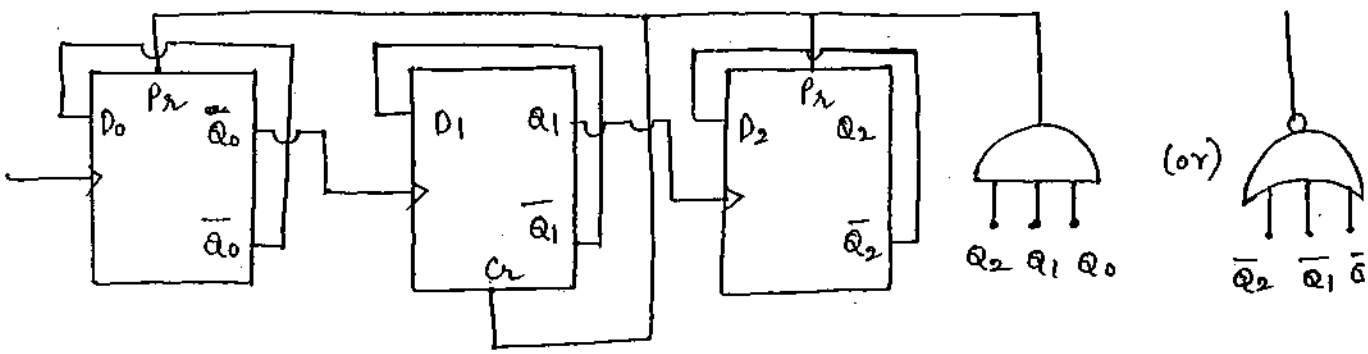


111

→ By using logic-1 enable preset pins,

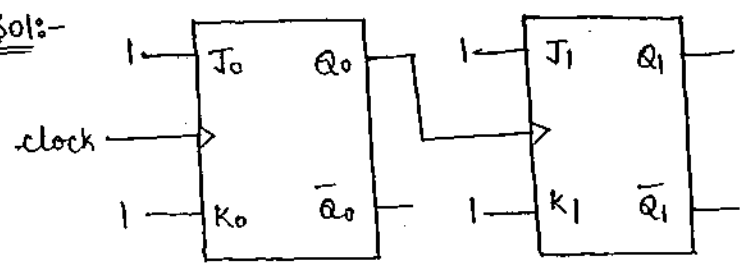
→ To count

111 → 101.
101
100
011
010
001
000

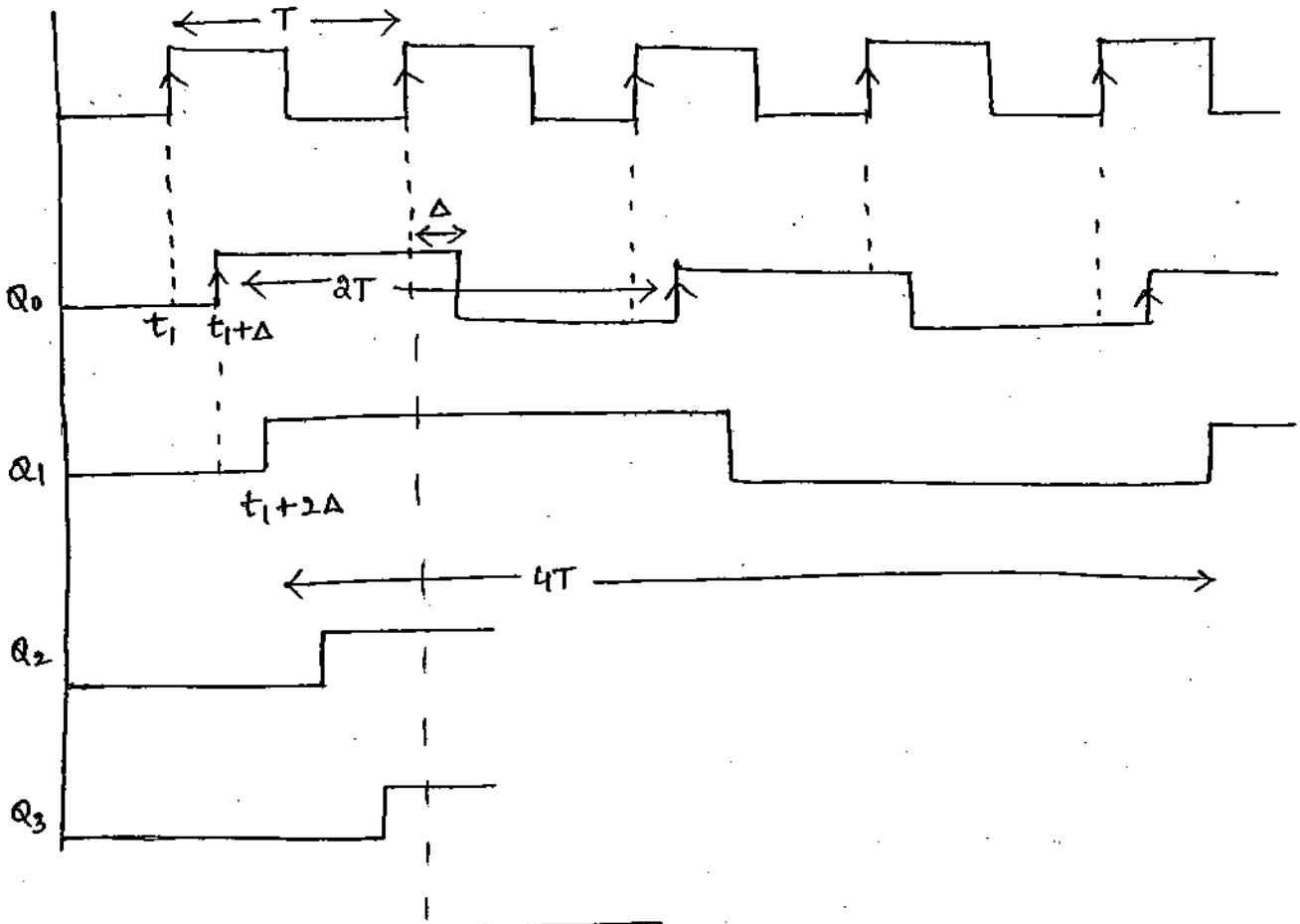


Q.7

Sol:-



propagation delay is ΔT,



$n\Delta < T$ → By using this how many flipflops should be used can be known in asynchronous counter.

2^n → Maximum modulus of the counter.

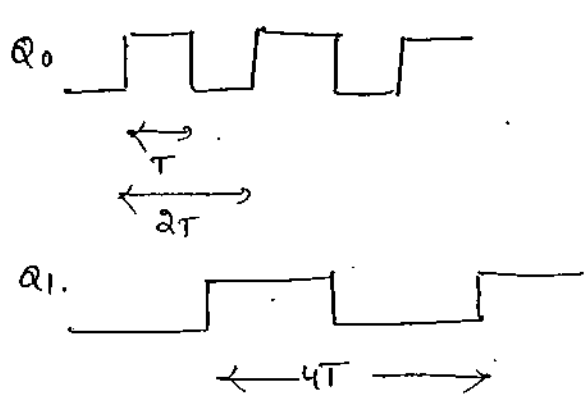
$T_{Q_0} = 2T.$

$f_{Q_0} = \frac{1}{2T} = \frac{f}{2}.$

$T_{Q_1} = 4T.$

$f_{Q_1} = \frac{1}{4T} = \frac{f}{4}.$

Q_1	Q_0
0	0
0	1
1	0
1	1



Q.11

Sol:- $Q_3\ Q_2\ \bar{Q}_1$ ← NAND gate i/p's.

1 1 0

$Q_3\ Q_2\ Q_1$.

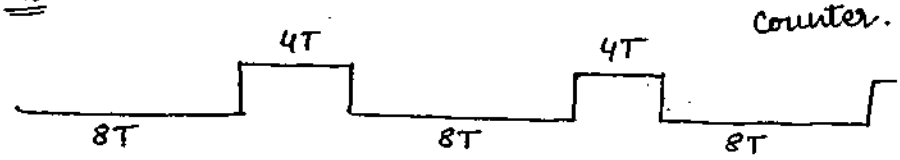
$f = 10\ KHz$.

up counter [$Q_3\ Q_2\ Q_1\ Q_0$

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 → 000

mod-12 counter.

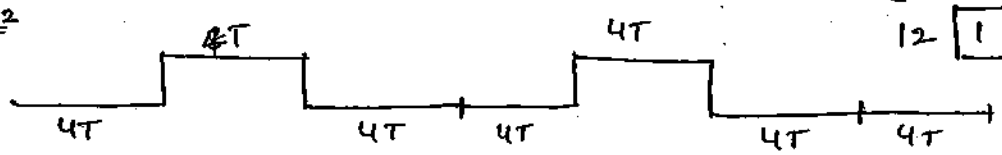
Q_3



$T_{Q_3} = 4T + 8T = 12T$.

$f_{Q_3} = f/12 = \frac{10}{12} = 0.833\ KHz$.

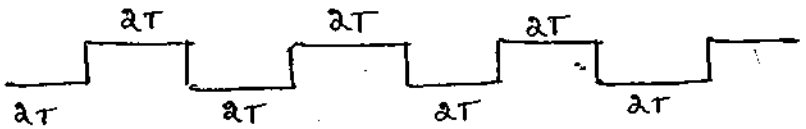
Q_2



$T_{Q_2} = 8T + 4T = 12T$.

$f_{Q_2} = f/12 = 10/12 = 0.833 KHz$.

Q_1



$T_{Q_1} = 2T + 2T = 4T$.

$f_{Q_1} = f/4 = \frac{10\ KHz}{4} = 2.5\ KHz$.

Q_0

$T_{Q_0} = 2T$

$f_{Q_0} = f/2 = \frac{10\ KHz}{2} = 5\ KHz$.

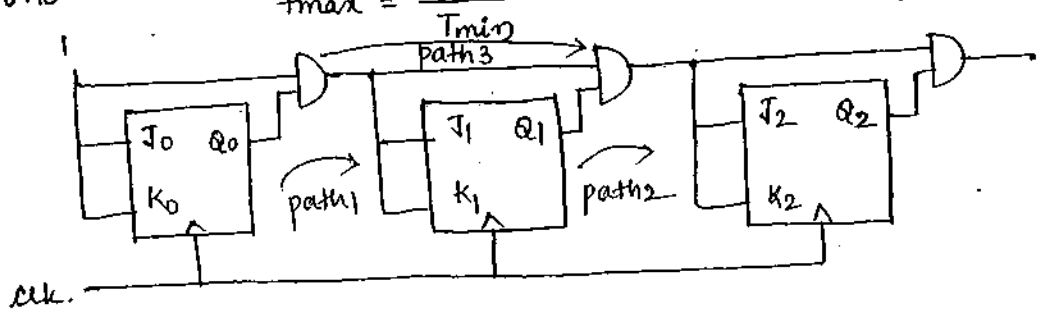
Q1

Sol:- $T_{ff} = 10ns$.

$T_{and} = 10ns$.

$T_{min} = max(T_1, T_2)$.

$f_{max} = \frac{1}{T_{min}}$



$T_{min1} = 10ns + 10ns + 0$

$= 20ns. = T_{min2}$

$T_{min3} = 10ns + 10ns + 10ns = 30ns.$

$\therefore T_{min} = \max(T_{min1}, T_{min3}).$

$= 30ns.$

$f_{max} = \frac{1}{T_{min}}$

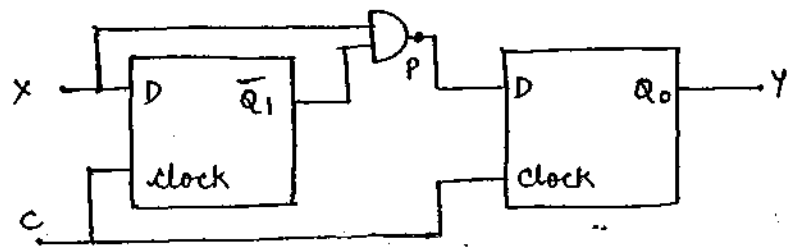
$= \frac{1}{30ns}$

$= 33.3\ MHz$

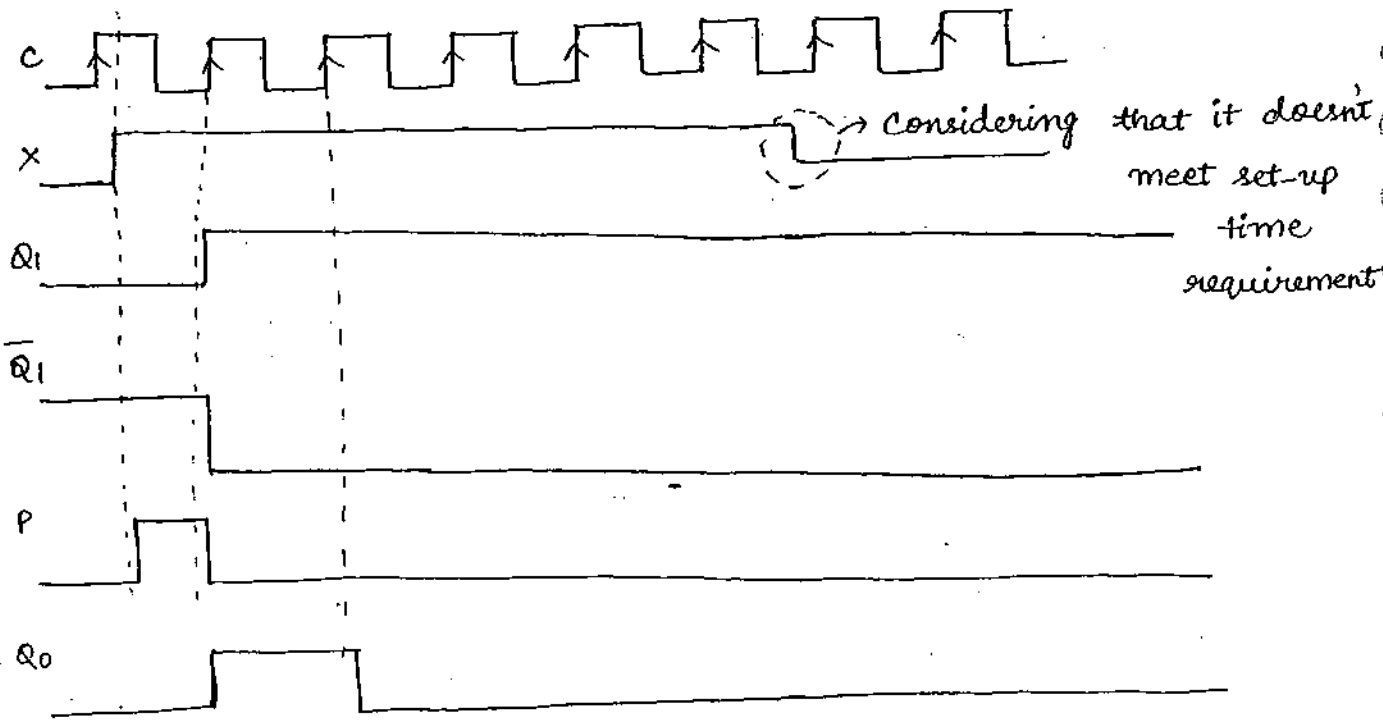
from options, $f_{max} = 25KHz$.

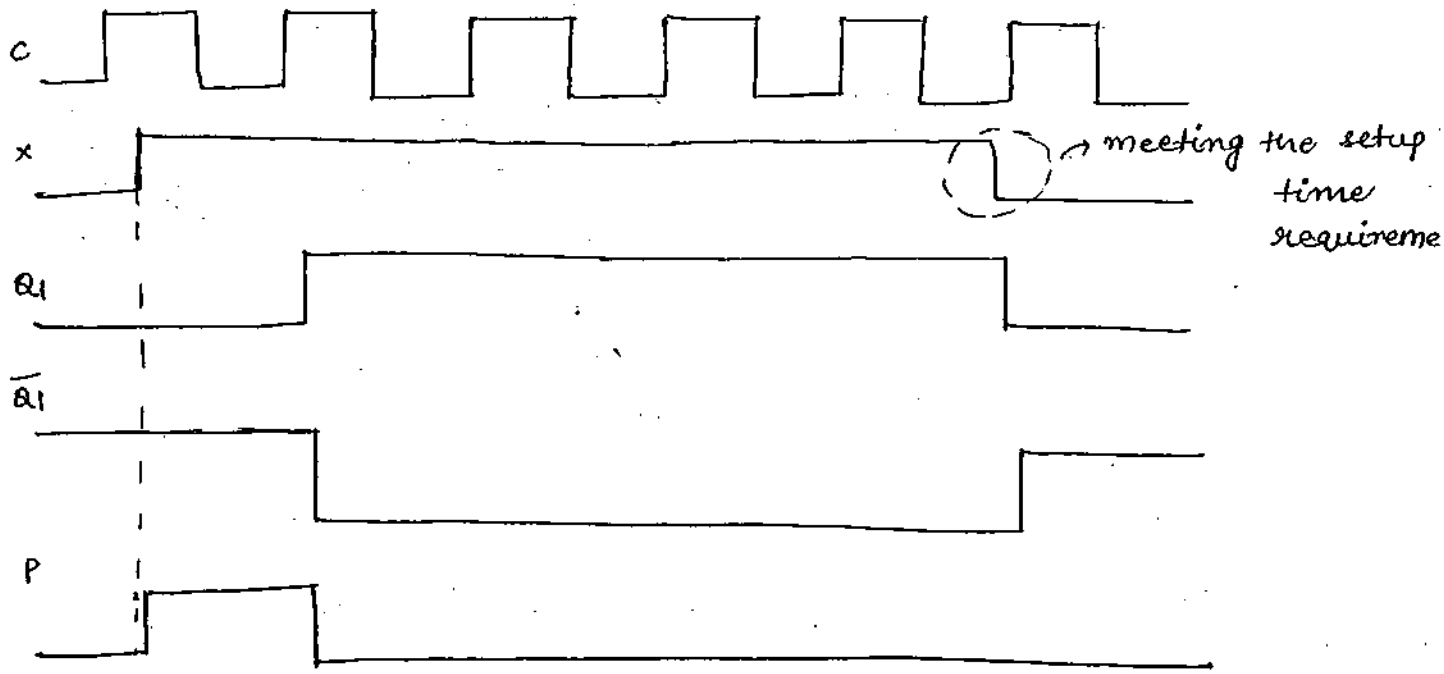
Q.2 Sol:- $Q_0 = Q_1 = 0$

$T_{su} = 20ns.$



$C > 40\ ns.$

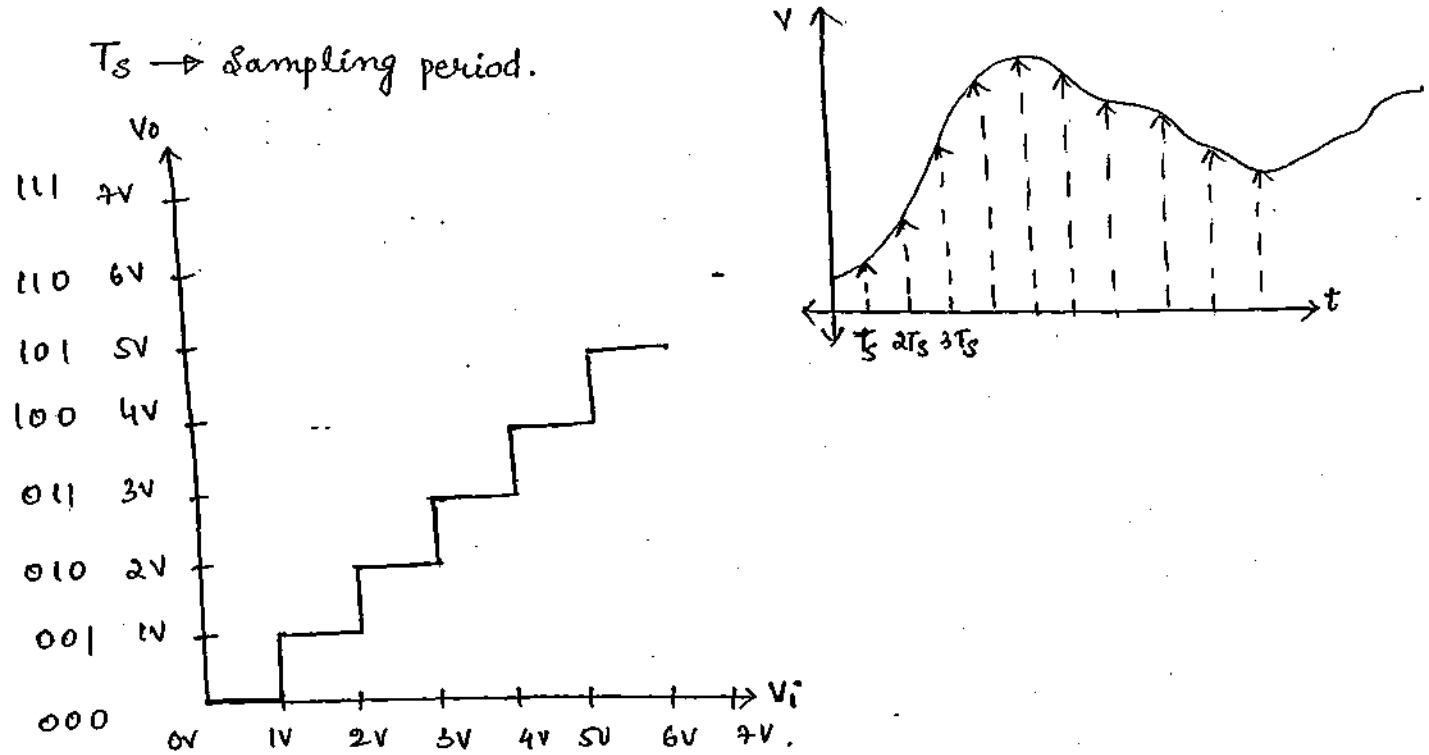




$Y = Q_0$

A/D and D/A CONVERTERS

$T_s \rightarrow$ Sampling period.



max possible error = step size.

Analog output, $V_o = K(1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0) \longrightarrow 100$ i/p.

$\Downarrow$

stepsize.

$$V_o = K(4+0+0)$$

$$V_o = 4K$$

$$= 4 \times 1V$$

$$= 4V.$$

DIGITAL TO ANALOG CONVERTERS :-

(i) Weighted resistor D/A.

(ii) R-2R resistor n/w D/A.

Weighted resistor D/A :-

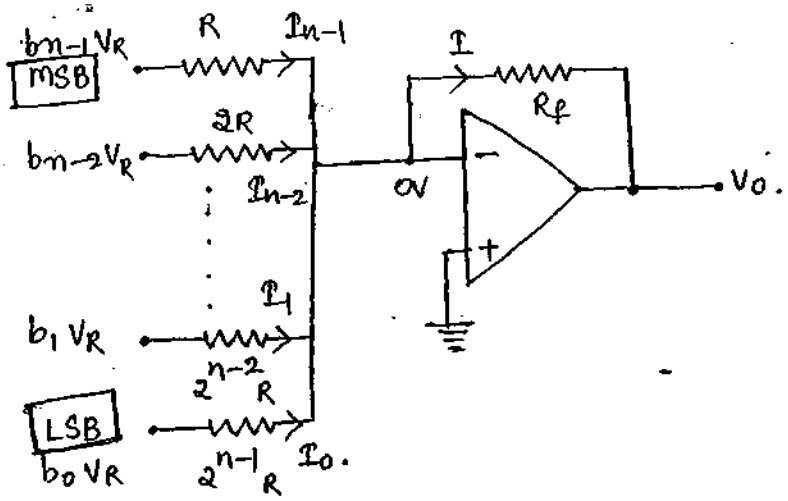
$$I_{n-1} + I_{n-2} + \ldots I_1 + I_0 = I.$$

$$\frac{b_{n-1}V_R}{R} + \frac{b_{n-2}V_R}{2R} + \ldots + \frac{b_1 V_R}{2^{n-2}R} + \frac{b_0 V_R}{2^{n-1}V_R} = \frac{-V_o}{R_f}.$$

$$\frac{V_R}{R \cdot 2^{n-1}} \left\{ b_{n-1} 2^{n-1} + b_{n-2} 2^{n-2} + \ldots b_1 2 + b_0 \right\} = \frac{-V_o}{R_f}.$$

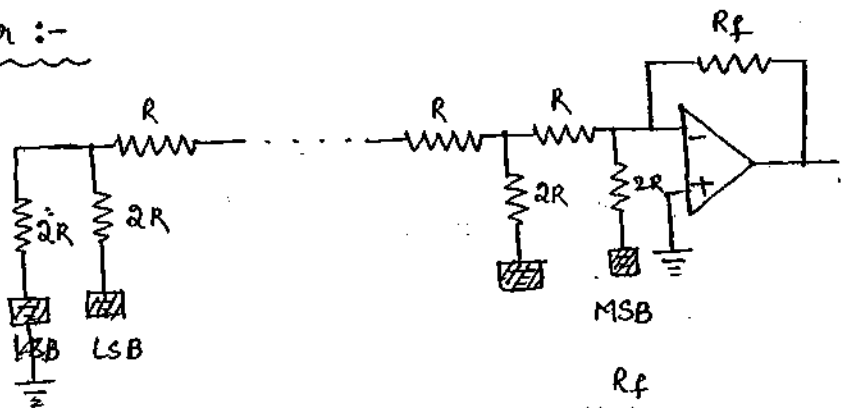
$$V_{out} = \frac{-R_f V_R}{R\, 2^{n-1}} \left\{ b_{n-1} 2^{n-1} + b_{n-2} 2^{n-2} + \ldots + b_1 2^1 + b_0 \right\}.$$

$= K$ { Decimal equivalent of digital i/p }.

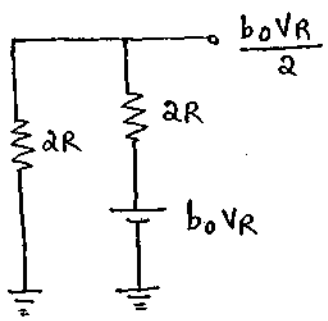


* In weighted resistor D/A converter, for n-bit conversion it requires 'n' resistances.
* It requires very large resistance $2^{n-1}R$ for large number of bit conversions.
* As it is using many resistances in the circuit, the linearity is missing.

R-2R resistor ntw D/A converter :-

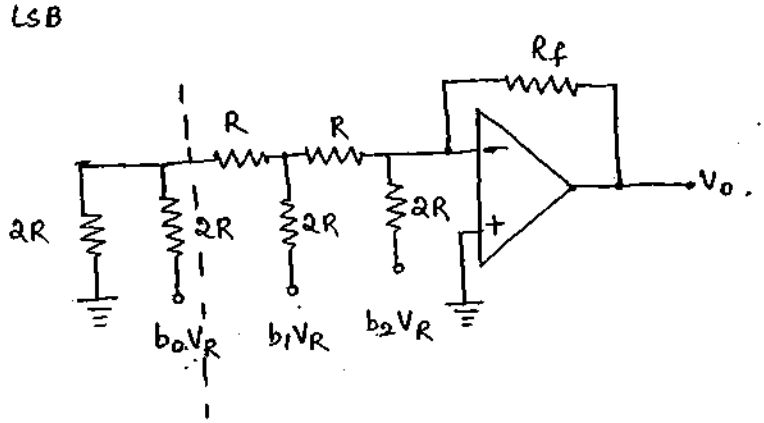


3-bit A/D converter :-



$R_{th} = 2R \| 2R$

$= R$.

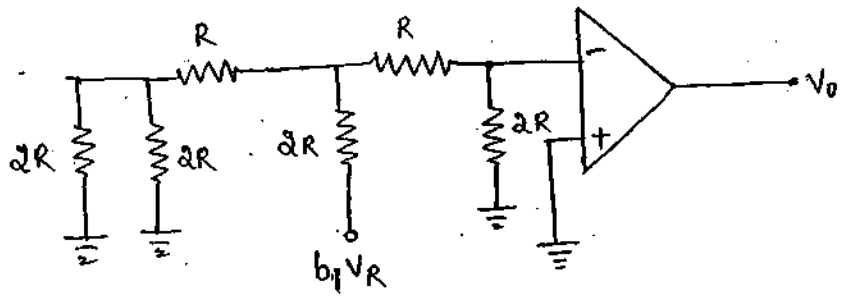


$\therefore R_{th\,eff} = R$.

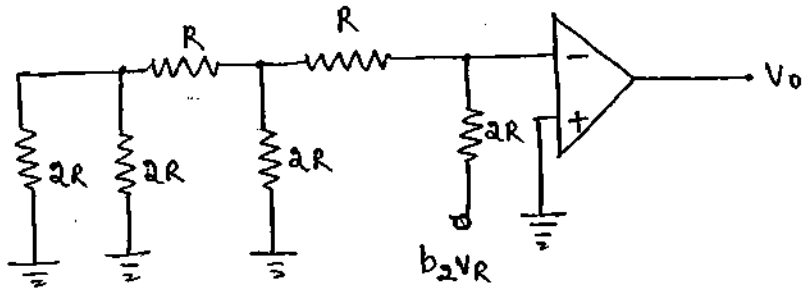
$$V_{o_{th\,eff}} = \frac{b_0 V_R}{8}.$$

$$V_{o,0} = -\frac{R_f}{R} \times \frac{b_0 V_R}{8}.$$

$V_{0,1} = \frac{-R_f}{R} \frac{b_1 V_R}{4}$.



$V_{0,2} = \frac{-R_f}{R} \cdot \frac{b_2 V_R}{2}$



$V_0 = V_{0,0} + V_{0,1} + V_{0,2}$.

$= \frac{-R_f}{R} \times \frac{V_R}{8} \left(b_0 + b_1 \cdot 2 + b_2 \cdot 4 \right)$.

$= \frac{-R_f}{R} \times \frac{V_R}{8} \left(b_2 \cdot 2^2 + b_1 \cdot 2^1 + b_0 2^0 \right)$.

= K (decimal equivalent of decimal Digital)

$$\text{Resolution} = \frac{V_R}{2^n}.$$

As only two resistances R, 2R used, the linearity is improved.

If we consider full scale voltage,

$$\text{Resolution} = \frac{V_{FS}}{2^n - 1}$$

ANALOG TO DIGITAL CONVERTERS :-

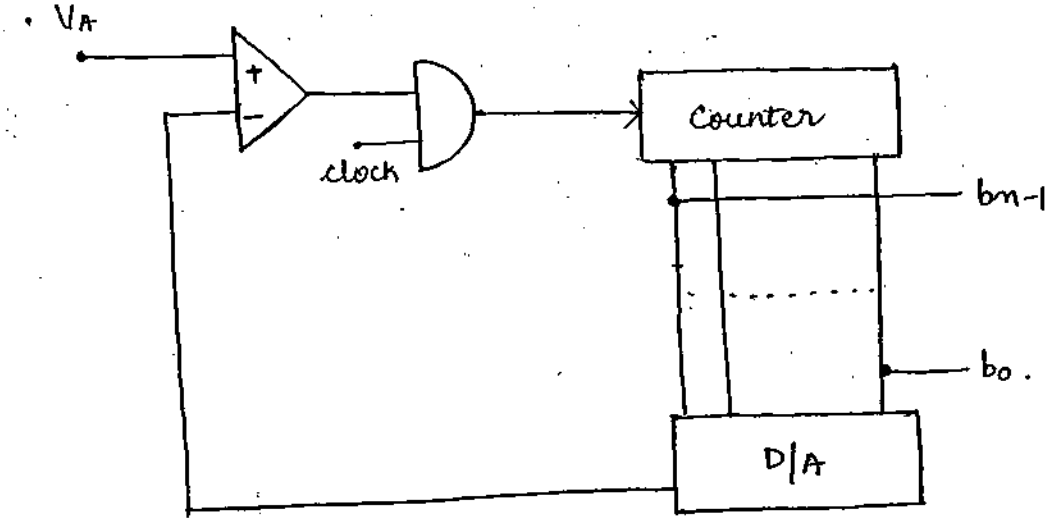
(1) Counter type A/D.

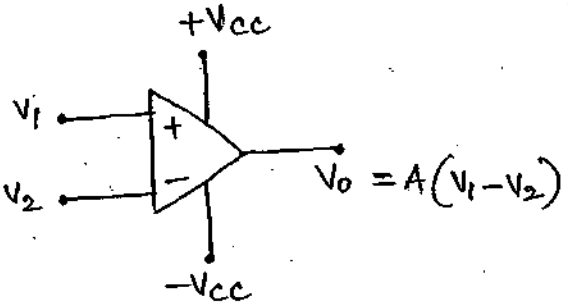
(2) Flash type A/D.

(3) Successive approximation type A/D.

(4) Dual slope type A/D.

Counter type A/D :-





$V_1 > V_2 \Rightarrow V_o = +Vcc \longrightarrow$ logic '1'.

$V_1 < V_2 \Rightarrow V_o = -Vcc \longrightarrow$ logic '0'.

$V_1 = V_2 \Rightarrow V_o = 0$.

considering no offset voltages.

conversion time is approximately $\underline{2^n}$ clock cycles.

If the analog o/p of D/A converter is less than analog i/p which w want to convert to digital, the comparator o/p is logic '1', AND gate o/p is clock. As the counter is getting the clock cycles, counter is incrementing at eve clock edge.

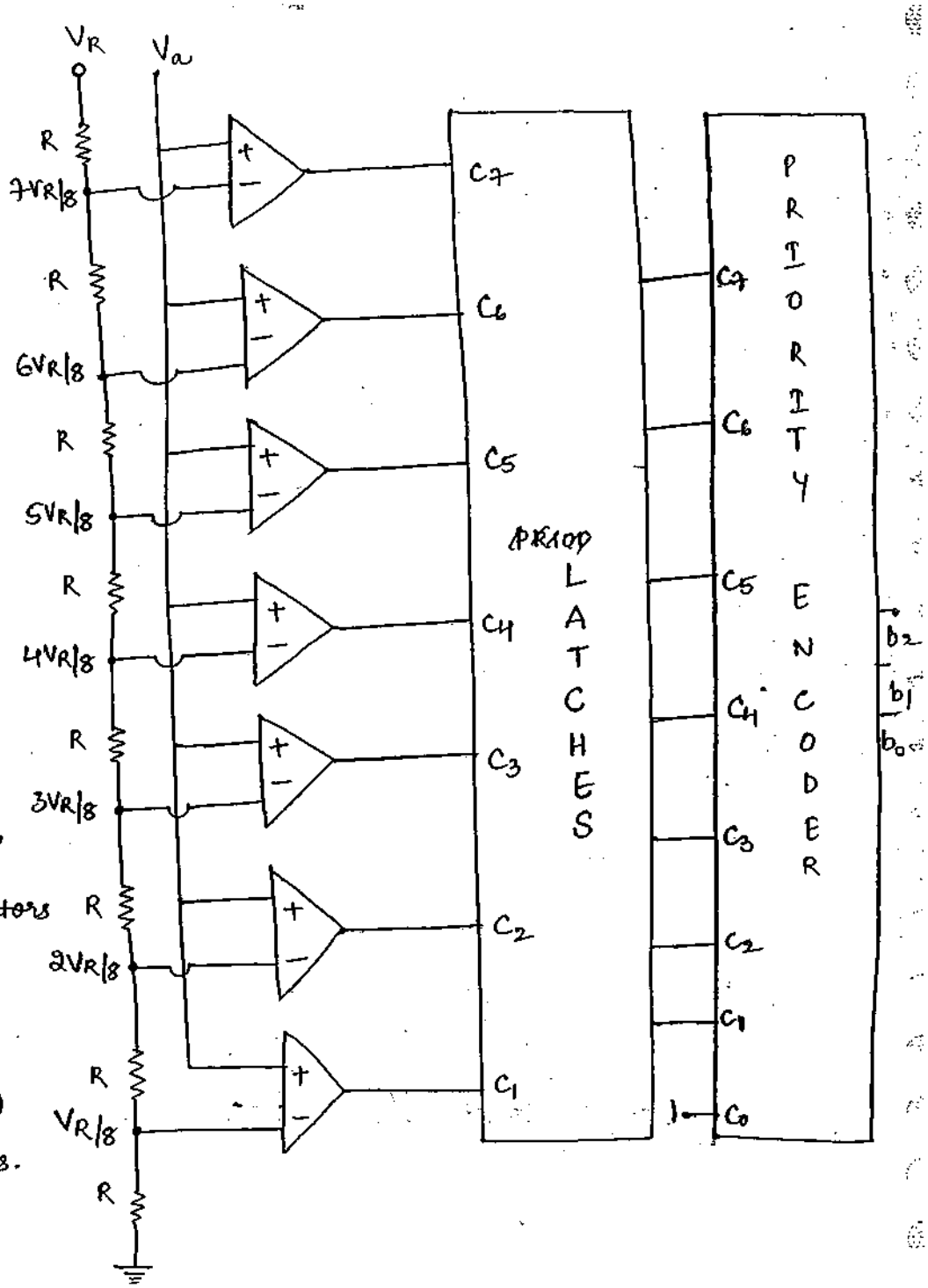
If the analog o/p of D/A converter is exceeding the analog i/p, comparator o/p is logic '0', AND gate o/p is zero. As counter is not getting the clock cycles, counter stops counting.

The count sequence of the counter is considered as approx. digital o/ for analog i/p. The max conversion time is approx. 2^n clock cycles.

Flash type A/D :-

* The fastest A/D.
* $2^n - 1$ comparators are required for n-bit conversion.

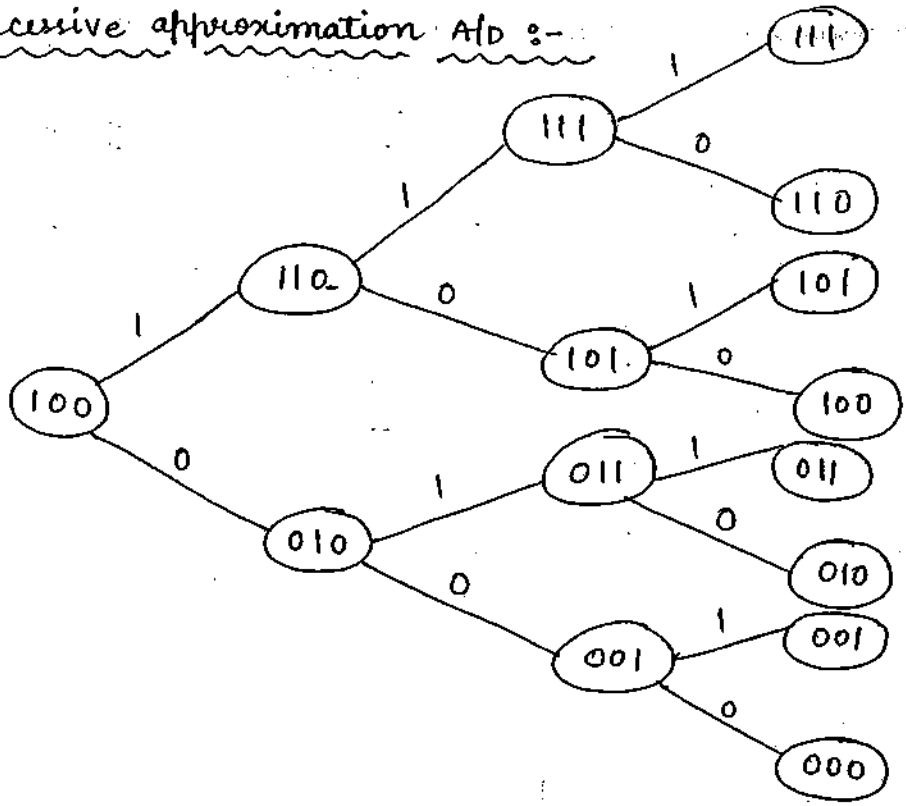
* Only ideal op-amps are considered such that current entering the op-amp will be zero as the i/p impedance of ideal op-amp is ∞ for an ideal op-amp.
* flash type A/D takes very less conversion time.
* The circuit is very complex as it is using $2^n - 1$ op-amp comparators for n-bit conversion.
* The approximation time (conversion time) is in ~~n~~ nanoseconds.

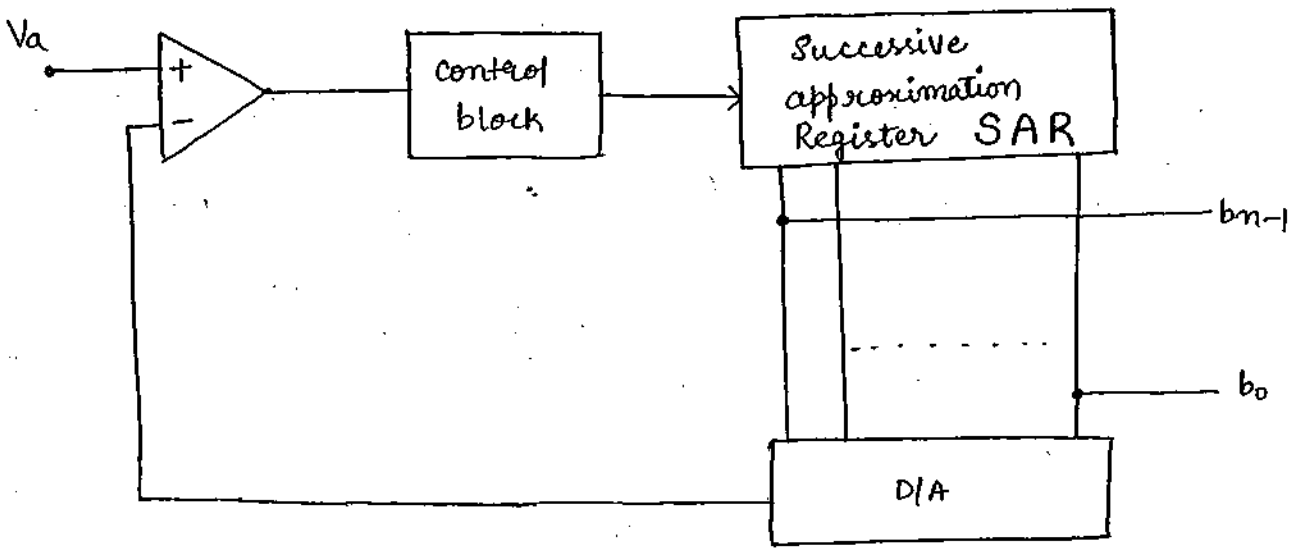


	C_1	C_2	C_3	C_4	C_5	C_6	C_7	b_2	b_1	b_0
$V_a < \frac{V_R}{8}$	0	0	0	0	0	0	0	0	0	0
$\frac{V_R}{8} < V_a < \frac{2V_R}{8}$	1	0	0	0	0	0	0	0	0	1
$\frac{2V_R}{8} < V_a < \frac{3V_R}{8}$	1	1	0	0	0	0	0	0	1	0
$\frac{3V_R}{8} < V_a < \frac{4V_R}{8}$	1	1	1	0	0	0	0	0	1	1
$\frac{4V_R}{8} < V_a < \frac{5V_R}{8}$	1	1	1	1	0	0	0	1	0	0
$\frac{5V_R}{8} < V_a < \frac{6V_R}{8}$	1	1	1	1	1	0	0	1	0	1
$\frac{6V_R}{8} < V_a < \frac{7V_R}{8}$	1	1	1	1	1	1	0	1	1	0
$V_a > \frac{7V_R}{8}$	1	1	1	1	1	1	1	1	1	1

* It doesnot require any clock and counter.

Successive approximation A/D :-





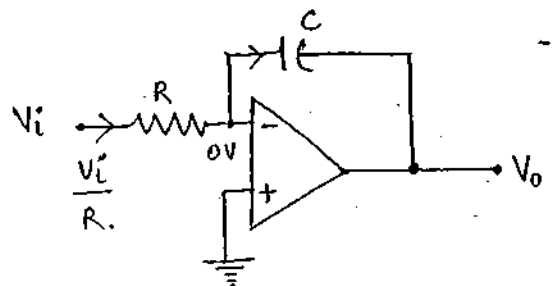
* It is faster than all the A/D converters except flash type. The approximation time is n-clock cycles for n-bit conversion.

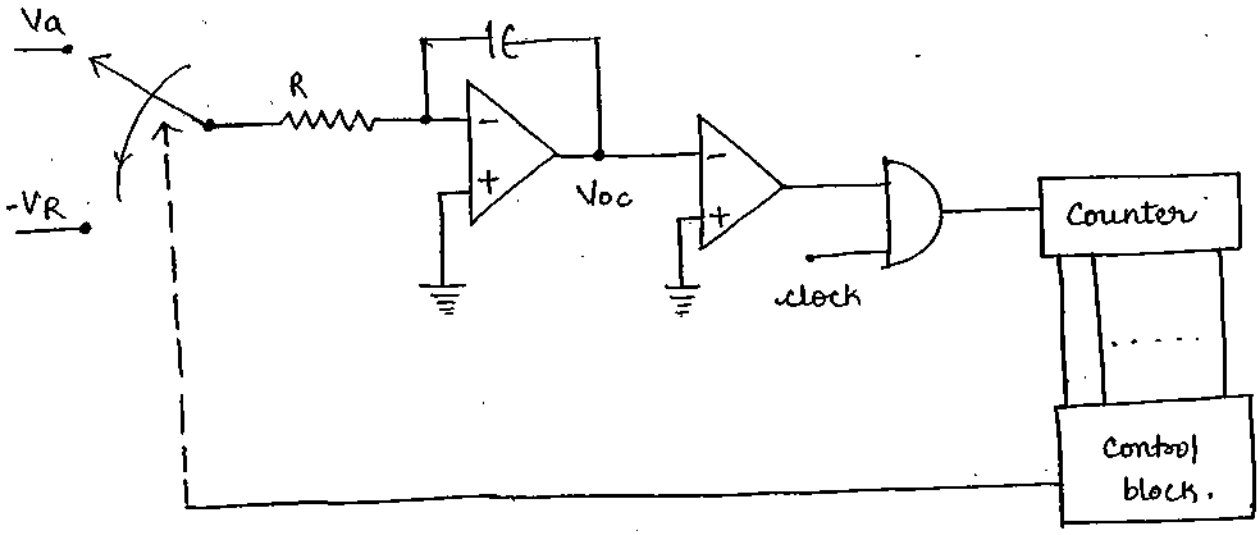
Dual slope A/D :-

$$V_o = -V_c.$$

$$V_o = \frac{-1}{C}\int i_c\, dt. \cdots$$

$$= \frac{-1}{C}\int \frac{V_i}{R}\, dt.$$





$$V_{oc} = -\frac{1}{\tau}\int v_i\,dt = -\frac{1}{\tau}\int V_1\,dt = -\frac{1}{\tau} V_i t \Big]_0^{T_1} = -\frac{1}{\tau} V_i T_1 .$$

$$V_o = -\frac{V_a}{\tau} T_1 + \frac{V_R}{\tau}(t - T_1).$$

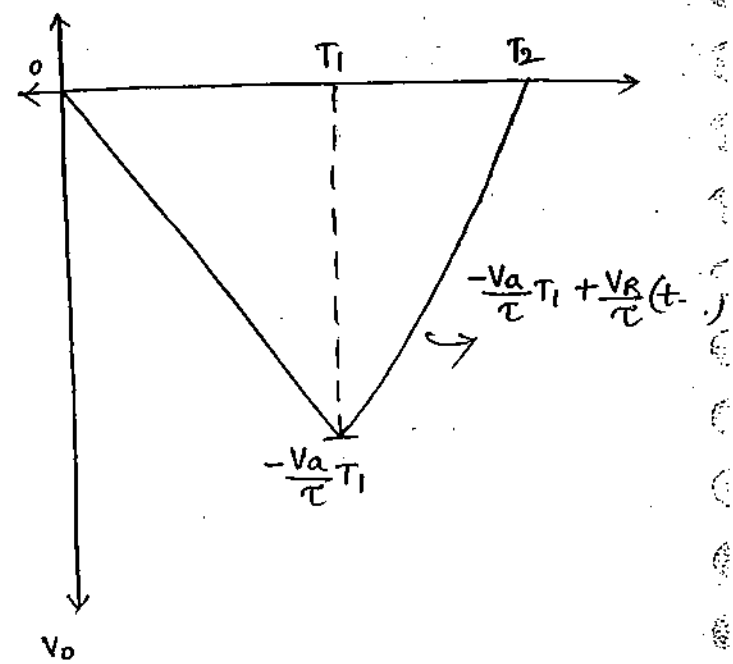
at $t = T_2$, $V_o = 0$.

$$0 = -\frac{V_a}{\tau} T_1 + \frac{V_R}{\tau}(T_2 - T_1).$$

$$\frac{V_a}{\tau} T_1 = \frac{V_R}{\tau}(T_2 - T_1).$$

$$V_a T_1 = V_R (T_2 - T_1)$$

$$V_a (2^n T_{clk}) = V_R (N \cdot T_{clk}).$$



$$\boxed{V_a = \frac{V_R}{2^n} \cdot N}$$

$\boxed{V_a \propto N}$ Deintegration time depends on applied analog i/p.

$$V_a t_1 = V_R t_2 .$$

* It is the slowest A/D ucon 2^{n+1} [i.e., $2^n + 2^n$] clock cycles are required approximately.

Q.1

Sol:- $V_o = \left(1+\frac{7}{1}\right) \times \frac{1}{2^4}\left(0+1\times2^1+0\times2^2+1\times2^3\right)$

$= 8 \times \frac{1}{16}(2+8) = \frac{8\times10}{16} = 5V.$

[OR]

$V_{o,1} = \left(1+\frac{R_f}{R}\right)V_i = \left(1+\frac{7k}{1k}\right)\times\frac{1}{8}V = 1V.$

$V_{o,2} = \left(1+\frac{R_f}{R}\right)\frac{1}{2}V = \left(1+\frac{7}{1}\right)\times\frac{1}{2}V = \frac{8}{2} = 4V.$

$V_o = V_{o,1} + V_{o,2} = 1+4 = 5V.$

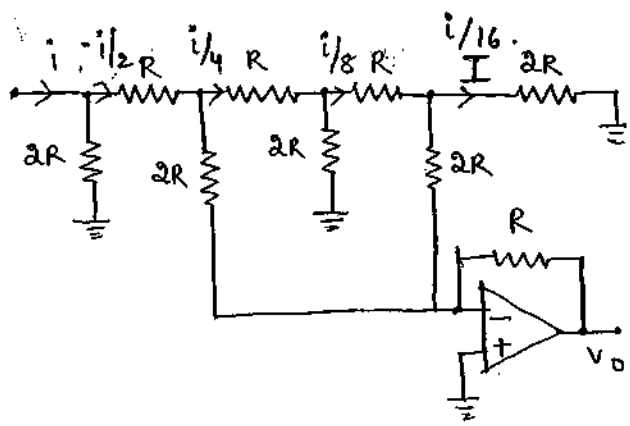
Q.2

Sol:-

Q_2	Q_1	Q_0	
D_3	D_2	D_1	D_0	
0	0	0	0	→ 0
0	0	0	1	→ 1
0	0	1	0	→ 2
0	0	1	1	→ 3
1	0	0	0	→ 8
1	0	0	1	→ 9
1	0	1	0	→ 10
1	0	1	1	→ 11.

Q.3 & 4

Sol:- D/A converter.



$V_R = 10V$ and $R = 10K\Omega$,

$i = \frac{V_R}{R} = \frac{10}{10K} = 1mA.$

$I = i/16 = \frac{1}{16}mA = 62.5\,\mu A.$

$V_o = -I_f R.$

$I_f = \frac{i}{4} + \frac{i}{16} = \left(\frac{1}{4}+\frac{1}{16}\right)mA = 0.3125\,mA = 312.5\,\mu A.$

$V_o = -0.3215mA \times 10K\Omega = -3.215\ V.$

Q 5 & 6

Sol:- $V_{DAC} = \sum_{r=0}^{3} 2^{n-1} b_n$ volts.

$V_{DAC} = 2^{-1} b_0 + 2^0 b_1 + 2^1 b_2 + 2^2 b_3$.

$V_{DAC} = 2^{-1}(1) + 0 + 0 + 0 = 0.5\,V$.

$V_{in} = 6.2\,V$.

no. of clock cycles $= \frac{6.2}{0.5} = 12.4$ cycles.

$\simeq$ 13 cycles for stable reading.

for 13 cycles, $V_{in} = 13 \times 0.5$

$= 6.5\,V$.

error $= 6.5 - 6.2 = 0.3\,V$.

Q9 Sol:- 8 bit D/A converter,

$V_{FS} = 20\,V$.

Resolution $= \frac{20V}{2^8 - 1} = \frac{20}{255} = 0.0784$.

11 011011.

$\Rightarrow 2^7 + 2^6 0 + 2^4 + 2^3 + 0 + 2^1 + 2^0$

$\Rightarrow$ 219. $V_0 = \frac{20}{255} \times 219 = 17.176\,V$.

Q11 Sol:- 10 bit A/D converter

$\frac{5-0}{2^{10}-1} = 4.89\,mV \simeq 5\,mV$.

Q12 Sol:- $3\frac{1}{2}$ digit DVM,

$V_R = 100\,mV$.

$T_1 = 300\,msec$.

$T_2 = 370.2\,ms$.

$V_a \times T_1 = V_R \times T_2$.

$V_a = \frac{100 \times 370.2}{300}$

$V_a = 123.4\,mV$.

Q13 Sol:- $V_{in} = 1.275$, o/p is 11111111.

quantization error = Resolution $= \frac{V_{FS}}{2^n - 1} = \frac{1.275}{2^8 - 1}$

$= 5\,mV$.

Q.14

Sol:- 8 bit D/A converter

$V_{ref} = 8V$.

to $\begin{cases} 00000000 \longrightarrow 0V \\ 11110000 \longrightarrow 240V. \end{cases}$

$V_p = \frac{8}{2^8} \times 240 = 7.5V.$

$V_{p-p} = 7.5 - 0 = 7.5V.$

Q.15

Sol:- 10 bit A/D converter.

10 bit counter.

$f_{clk} = 1MHz.$

$T_s \geq 2^{n+1} T_{clk}.$

$f_s \leq \frac{1}{2^{10+1}} \times 1M$

$f_s \leq \frac{10^6}{2^{11}}$

$f_s \leq 488.281 Hz.$

$f_s \cong 500 Hz.$

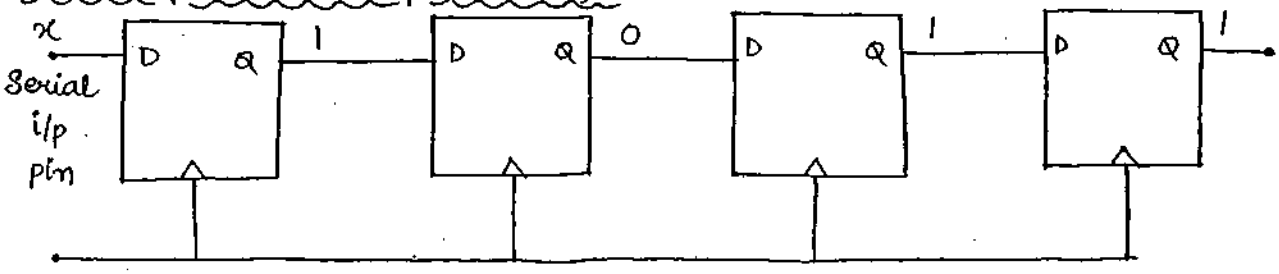
$T_s \geq$ approximation time of A/D.

$f_s \leq \frac{1}{\text{approx. time of A/D}}.$

REGISTER :-

To store n-bit information collectively, we use 'n' number of flipflops collectively is called n-bit register.

Serial i/p & serial o/p (SISO) :-



Ex:- 1001

1st clock : 1 x x x

2nd clock : 0 1 x x

3rd clock : 0 0 1 x

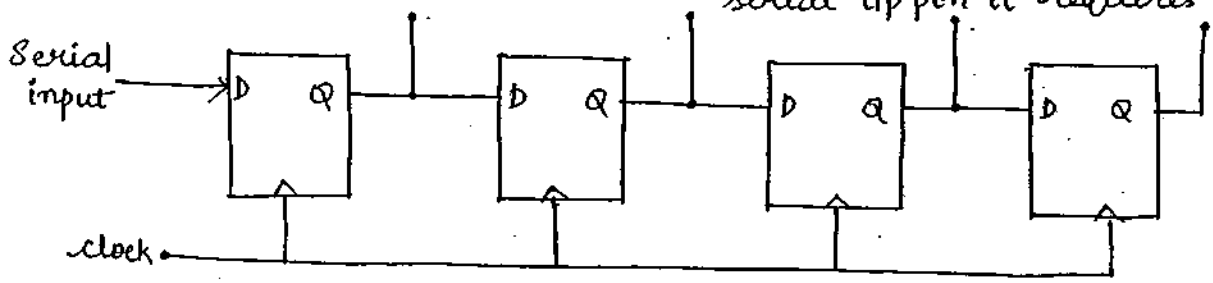
4th clock : 1 0 0 1.

In SISO operation using n-bit shift register, it requires 2n-1 clock cycles to get n-bit informati at serial o/p pin which is feeded serially at serial i/p pin.

Ex: To get n-bit serial o/p for n-bit serial i/p, it requires (2n-1) clock cycles.

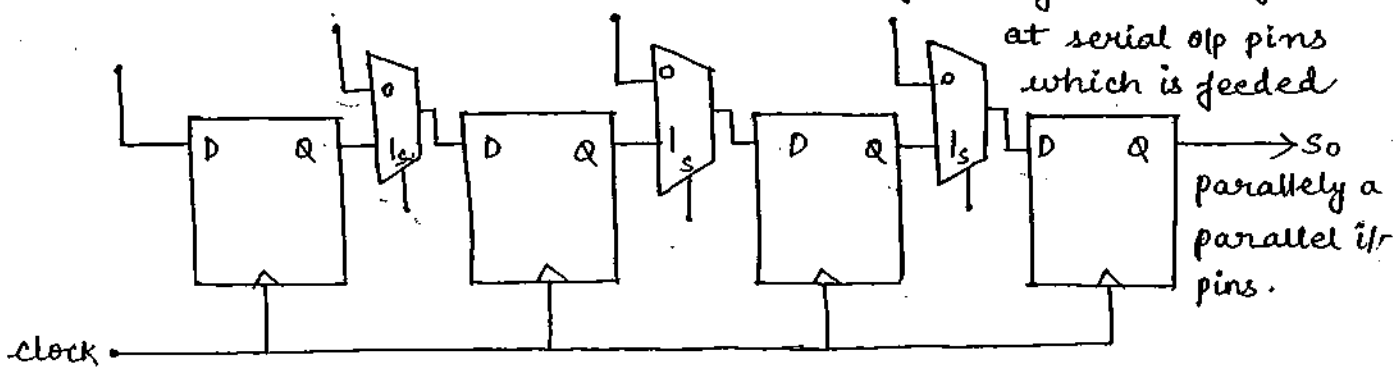
$n+(n-1)$

Serial input parallel output :- (SIPO) To get n-bit information on parallel o/p pins which is feeded serially at serial i/p pin it requires 'n' clock cycles.



* It requires 'n' clock cycles to get o/p (parallel o/p).

Parallel input & serial output :- (PISO) In this operation, it requires n-clock cycles to get n-bit information at serial o/p pins which is feeded parallely a parallel i/p pins.

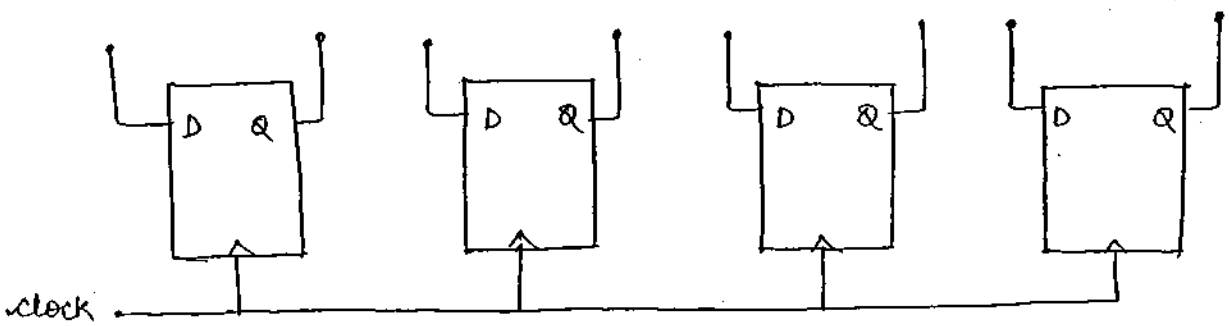


* It requires n clock cycles for required serial n-bit data at o/p.

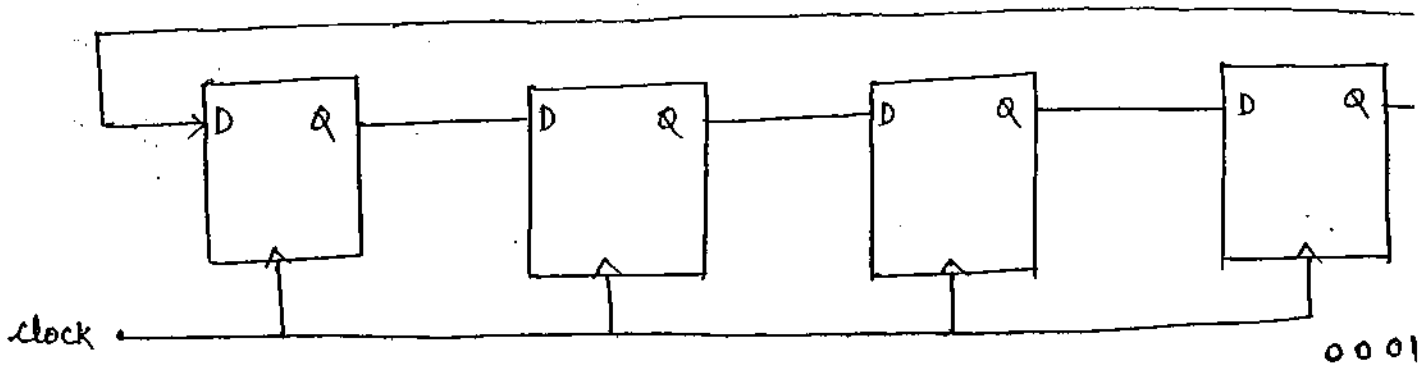
for S=0, parallel i/p.

S=1, for serial o/p.

Parallel input and parallel output :- (PIPO) It requires 1 clock cycle.



RING COUNTER :-



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

o/p of last flipflop feeded as i/p to the first flipflop.

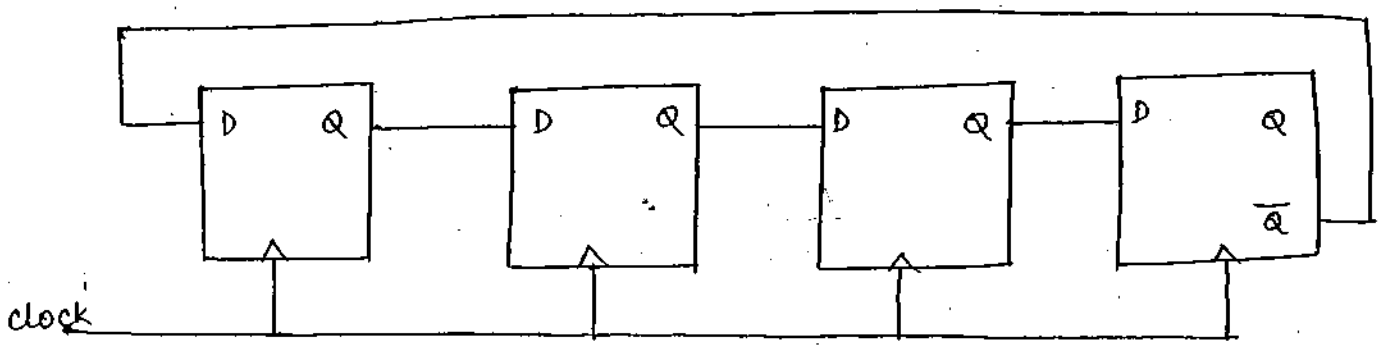
In ring counter using n-bit shift register we can generate maximum n-different vectors Ex:- 0000

→ 0000

⋮

→ 0000.

TWISTED RING COUNTER (JOHNSON COUNTER) :-



In twisted ring counter using n-bit shift register wecan generat maximum '2n' different vectors.

Pgno.83

Q.8 Sol:- $K = 2^n$

$4 = 2^2$

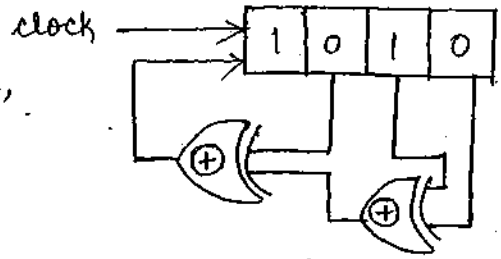
$n = 2, \quad K = 4$

Q_1	Q_0	Y_0	Y_1	Y_2	Y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

K-bit ring Counter

Q.9 Sol:- 1010.

as $(A \oplus B) \oplus C = A \oplus B \oplus C$,



1010

1st clk → 1101

2nd clk → 0110

3rd clk → 0011

4th clk → 0001

5th clk → 1000

6th clk → 0100

7th clk → 1010

Eg:- In the shift register shown in the figure, the contents of the register after occurence of the first clock cycle is 1100 what must be the state to be initialised.

1001 ⟶ Ans

1100

Q.14

Sol:- 8 bit D/A converter to $\begin{cases} 00000000 \longrightarrow 0V \\ 11110000 \longrightarrow 240V. \end{cases}$

$V_{ref} = 8V.$

$V_o = \frac{8}{2^8} \times 240 = 7.5V.$

$V_{p-p} = 7.5 - 0 = 7.5V.$

Q.15

Sol:- 10 bit A/D converter.

10 bit counter.

$f_{clk} = 1MHz.$

$T_s \geq$ approximation time of A/D.

$f_s \leq \frac{1}{\text{approx. time of A/D}}.$

$T_s \geq 2^{n+1} T_{clk}.$

$f_s \leq \frac{1}{2^{10+1}} \times 1M$

$f_s \leq \frac{10^6}{2^{11}}$

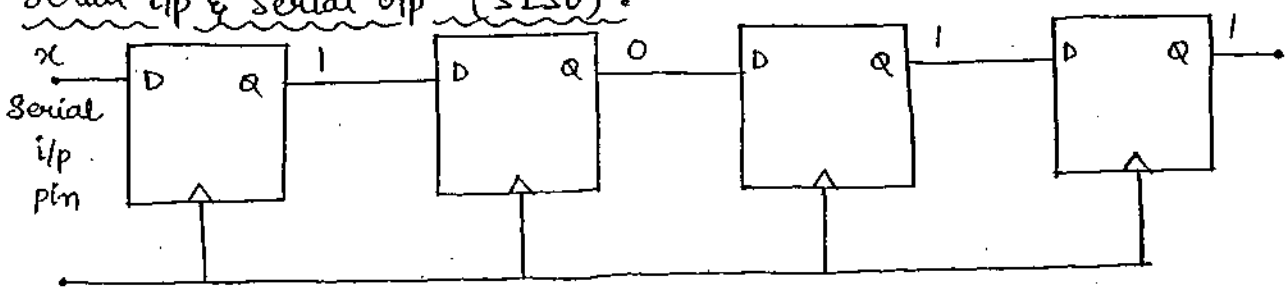
$f_s \leq 488.281 Hz.$

$f_s \cong 500 Hz.$

REGISTER :-

To store n-bit information collectively, we use 'n' number of flipflops collectively is called n-bit register.

Serial i/p & serial o/p (SISO) :-



Ex:- 1001

1st clock : 1 x x x

2nd clock : 0 1 x x

3rd clock : 0 0 1 x

4th clock : 1 0 0 1.

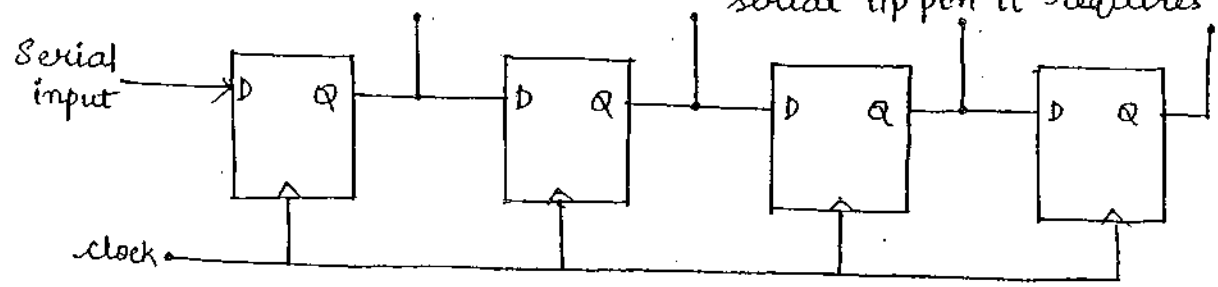
In SISO operation using n-bit shift register, it requires 2n-1 clock cycles to get n-bit informati at serial o/p pin which is feeded serially at serial i/p pin.

Ser: To get n-bit serial o/p for n-bit serial i/p, it requires (2n-1) clock cycles.

⇓

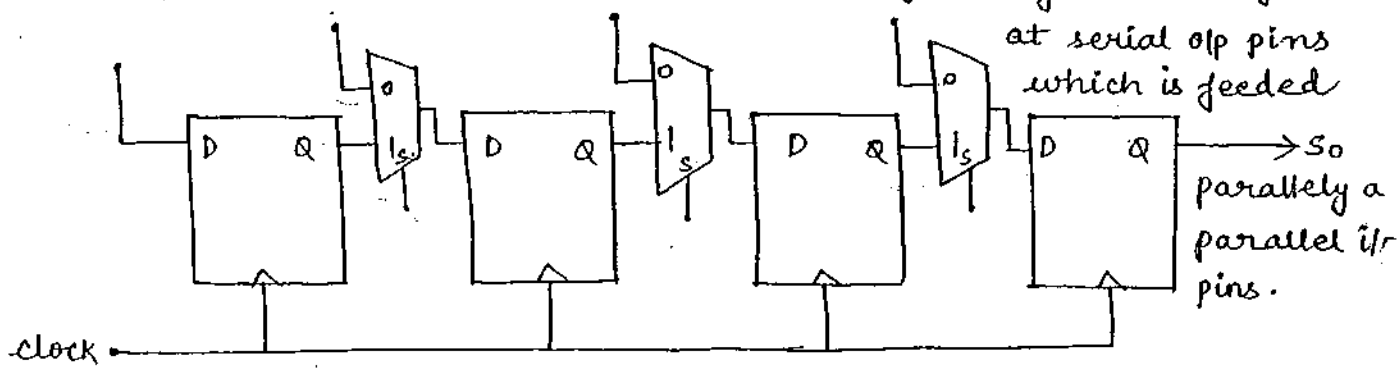
n+(n-1)

Serial input parallel output :- (SIPO) To get n-bit information on parallel o/p pins which is feeded serially at serial i/p pin it requires 'n' clock cycles.



* It requires 'n' clock cycles to get o/p (parallel o/p).

Parallel input & serial output :- (PISO) In this operation, it requires n-clock cycles to get n-bit information at serial o/p pins which is feeded parallely a parallel i/p pins.



* It requires n clock cycles for required serial n-bit data at o/p.

for S=0, parallel i/p.

S=1, for serial o/p.

Parallel input and parallel output :- (PIPO) It requires 1 clock cycle.