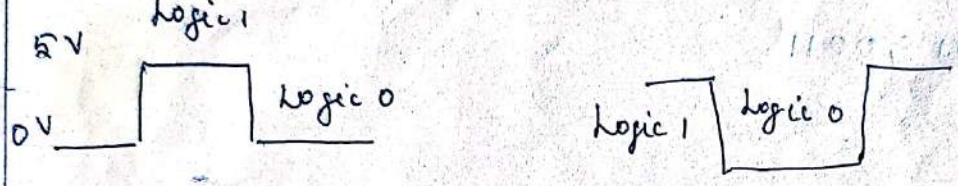
DIGITAL SYSTEM DESIGN - U23ECT31
DIGITAL PRINCIPLES AND SYSTEM DESIGN - U23ECT23
UNIT-I (NUMBER SYSTEMS AND CODE CONVERSIONS)

DIGITAL ELECTRONICS :

* Digital electronics acts as switch, which is represented as logic HIGH and logic LOW (as '0' or '1').

Positive logic	negative logic
Low = 0 High = 1	Low = 1 High = 0



(a) +ve logic (b) -ve logic

→ Advantages of Digital Systems :-

* Accuracy level of digital system is higher
* Noise effect is less
* Power consumption for digital IC is less than Analog IC.

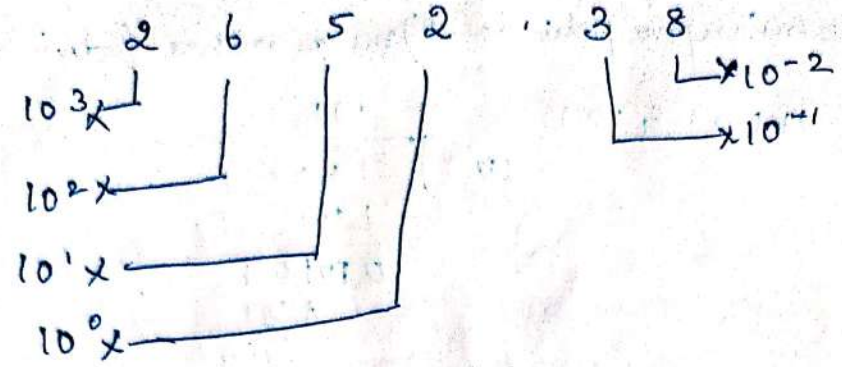
REVIEW OF NUMBER SYSTEMS :- / CONVERSIONS :-

* Mathematical object used to count, label and measure the task performed by the logic function.
* 4 ways of numbering system :
  i) Decimal
  ii) Binary
  iii) Octal
  iv) Hexa-decimal.

i) Decimal :-

* Ten distinct digits 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9 is known as decimal number system.

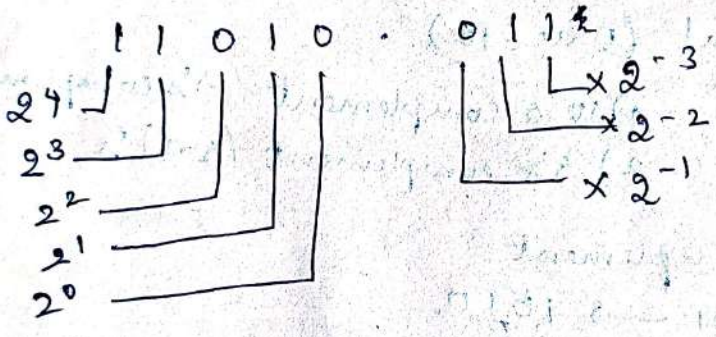
Eg :- $(2652.38)_{10}$



(ii) Binary :-

* Number system uses only digits 0 and 1 is known as binary number.
* Symbol 0 and 1 known as bits.
* The weight or Place value in terms of power 2 and as $2^0, 2^1, 2^2 \ldots$ etc.

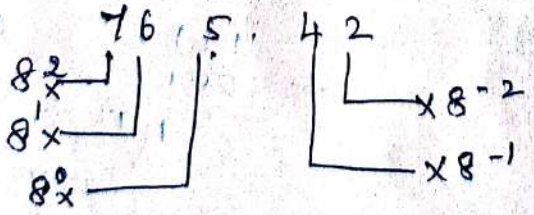
Eg :- $(11010.011)_2$



(iii) Octal :-

* Eight digits 0, 1, 2, 3, 4, 5, 6 and 7 is called octal number.

eg :- $(765.42)_8$



(iv) Hexadecimal :-

* 16 distinct digit symbols. 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 Plus the A, B, C, D, E and F as 16 digit.

Eg:- $(A8C.28)_{16}$

A 8 C . 2 B

16^2 16^1 16^0 $\times 16^{-2}$ $\times 16^{-1}$

Problems:-

* Decimal to Binary

$(455)_{10}$

2|455 – Reminder
2|227 – 1
2|113 – 1
2|56 – 0
2|28 – 0
2|14 – 0
2|7 – 0
2|3 – 1
1 – 1

$\therefore (455)_{10} = (111000111)_2$

$0.22 \times 2 = 0.44$
$0.44 \times 2 = 0.88$
$0.88 \times 2 = 1.76$
$1.76 \times 2 = 1.52$

$(256.22)_{10}$

2|256
2|128 – 0
2|64 – 0
2|32 – 0
2|16 – 0
2|8 – 0
2|4 – 0
2|2 – 0
1 – 0

* Decimal to octal

$(455)_{10}$

8|455 – 7
8|56 – 7
7 – 0

$\therefore (455)_{10} = (707)_8$

* Decimal to Hexadecimal

$(455)_{10}$

16|455
16|28 – 7
1 – 12 (C)

$\therefore (455)_{10} = (1C7)_{16}$

* Binary to decimal

$(11101)_2$

1 1 1 0 1

1×2^0 1×2^1 1×2^2 1×2^3 1×2^4

1 0 4 8 16

29

$(11101)_2 = (29)_{10}$

Binary to octal

$(11101)_2$ (3 bit)

LSB MSB

011 101

MSB LSB

1st 3 bit: 101 → $1\times2^0=1$, $0\times2^1=0$, $1\times2^2=4$ → 5

2nd 3 bit: 011 → $1\times2^0=1$, $1\times2^1=2$, $0\times2^2=0$ → 3

$(101)_5 \quad (011)_3$

$= (35)_8$

Binary to Hexadecimal

$(11101)_2$ (4 bit)

1st: 0001 → 1×2^0, 0×2^1, 0×2^2, 0×2^3 → 1

$(0001)_2 = (1)_{16}$

2nd: 1101 → $1\times2^0=1$, $0\times2^1=0$, $1\times2^2=4$, $1\times2^3=8$ → 13

$(1101)_2 = (13)_{10} = D$

$\therefore (11101)_2 = (1D)_{16}$

* Octal to Decimal

$(35)_8$

35

$5 \times 8^0 = 5$

$3 \times 8^1 = 24$

29

$\therefore (35)_8 = (29)_{10}$

to Binary

$(35)_8$

3 5

2 | 3 — 1

2 | 1 — 0

1

2 | 5 — 1

2 | 2 — 0

1 — 0

011 101

$\therefore (35)_8 = (011101)_2$

To Hexa

$(35)_8$

0001 1101

1 13 (D)

$\therefore (35)_8 = (1D)_{16}$

NUMBER REPRESENTATION :-

i) Fixed Point Representation.

ii) Floating Point Representation.

i) Fixed Point Representation :-

Integer	Fractional

← Fixed (Left side) → ← Fixed (Right Side) →

Eg : $(01101100)_2$

2^{-3}

$(00011.011)_2$

ii) Floating Point Representation :-

$= (0.m) \times B^{exp}$

Signed	Mantissa	Exponent

= Sign bit × Mantissa × Base Exponent.

CONVERSIONS :-

* Decimal
* Octal
* Hexadecimal
* Binary.

BOOLEAN ALGEBRA :- (REVIEW) :-

* Set of elements, set of operations
* Two binary operator '+' and '.'.
* Element '0' respect to '+' $A + 0 = 0 + A = A$

  Element '1' respect to '.' $A \cdot 1 = 1 \cdot A = A$

* Structure

i) Commutative with respect to '+'

$$A+B = B+A$$

ii) commutative with Respect to '.'

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

iii) operator '.' distributive over '+'

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

iv) operator '+' distributive over '.'

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

Associative laws:-

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

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

$\therefore A+\bar{A} = 1 \quad , A \cdot \bar{A} = 0$

THEOREMS:-

i) Demorgan's Theorems:-

Theorem 1:- $\overline{AB} = \bar{A} + \bar{B}$ (Complement of AND = OR operation)

A	B	AB	$\overline{AB}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

$\Rightarrow$

A	B	$\bar{A}$	$\bar{B}$	$\bar{A}+\bar{B}$
0	0	1	1	1
0	1	1	0	1
1	0	0	1	1
1	1	0	0	0

PRODUCT → SUM

Theorem 2:- $\overline{A+B} = \bar{A} \cdot \bar{B}$ (Complement OR = AND operation)

A	B	A+B	$\overline{A+B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

$\Rightarrow$

A	B	$\bar{A}$	$\bar{B}$	$\bar{A} \cdot \bar{B}$
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	0

SUM → PRODUCT

ii) Duality Theorem:-

Theorem 1:- $A+\bar{A} = 1$

Theorem 2:- $A \cdot \bar{A} = 0$

iii) Basic Theorem :-

Theorem 1a :- $A+A = A$ (Idempolency)

Theorem (b) :- $A \cdot A = A$ (")

Theorem 2a :- $A+1 = 1$

(b) :- $A \cdot 0 = 0$

Theorem 3a :- $\bar{\bar{A}} = A$ (Involution)

Theorem 4(a) :- $A + AB = A$ (Absorption)

(b) :- $A(A+B) = A$ (Absorption)

Theorem 5(a) :- $A+\bar{A}B = A+B$

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

[$A \cdot 1 = A$
$A+\bar{A} = 1$
$A \cdot A = A$
$A\bar{A} = 0$]

Postulates]
[$1+A = A$
$A \cdot 1 = A$
$1+A = 1$]

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

iv) Consensus Theorem :-

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

$= AB + \bar{A}C + (A+\bar{A})BC$ → redundant.

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

$= AB + \bar{A}C$

BOOLEAN EXPRESSIONS :-

i) Minterms :-

* AND operations called minterms . denoted 'm'

ii) Maxterms :-

* A and B are combined with OR operations

$A+B$ called Maxterm . Denoted 'M'

SUM OF PRODUCT (SOP) :-

* Also called as Sum of Minterms.
* Minterm have a more variables combined with AND logic

Eg :- $f(A,B,C) = \underbrace{AB + B\bar{C} + A\bar{C}}_{\text{Sum}}$

Product

PRODUCT OF SUM SIMPLIFICATION :-

* Product of sum is called Product of Maxterms.

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

Product

Sum terms

CONONICAL FORMS MIN TERM AND MAX TERM :-

* Boolean Function that is expressed as a Product of Maxterms (or) sum of minterms is called canonical form.

* 1 Maxterm → F = 1
  0 Maxterm → F = 0

* Minterms :-

$$0 \leq I < 2^n$$

3 Variables

A	B	C	Representation	Minterms
0	0	0	m_0	$\bar{A}\bar{B}\bar{C}$
0	0	1	m_1	$\bar{A}\bar{B}C$
0	1	0	m_2	$\bar{A}B\bar{C}$
0	1	1	m_3	$\bar{A}BC$
1	0	0	m_4	$A\bar{B}\bar{C}$
1	0	1	m_5	$A\bar{B}C$
1	1	0	m_6	$AB\bar{C}$
1	1	1	m_7	ABC

Eg :- SOP Form $Y = \Sigma m(111, 101, 010)$ [Canonical Form]

$Y = m_7 + m_5 + m_2$

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

ii) Maxterms :-

* Range $0 \le i \le 2^n$.

A	B	C	Representation	Maxterms
0	0	0	M_0	$A+B+C$
0	0	1	M_1	$A+B+\bar{C}$
0	1	0	M_2	$A+\bar{B}+C$
0	1	1	M_3	$A+\bar{B}+\bar{C}$
1	0	0	M_4	$\bar{A}+B+C$
1	0	1	M_5	$\bar{A}+\bar{B}+\bar{C}$
1	1	0	M_6	$\bar{A}+\bar{B}+C$
1	1	1	M_7	$\bar{A}+\bar{B}+\bar{C}$

Eg :- Pos form $Y = \Pi M(000, 010, 101)$

$Y = \Pi M(0, 2, 5)$

$Y = M_0, M_2, M_5$

Problems:- Express Boolean fn $F(A, B, C) = A + \bar{B}C$ in Sum of Minterms. (X)

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

Expand Reorder.

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

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

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

Omit Repeated product terms

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

SIMPLIFICATION OF BOOLEAN EXPRESSIONS:-

* Minimization of Boolean Expressions.

Problems :- Examples :-

1) $F = A(A+C)$

$= A \cdot A + A \cdot C \quad [\because A \cdot A = A]$

$= A + A \cdot C \quad [\because 1 + C = 1]$

$= A(1+C)$

$\boxed{F = A}$

A (A+C)
A(A·A
8421
000
001
010
100
A · $\bar{B}$

2) Simplify the given Boolean Expressions (2.)

$F = x' + xy + xz' + xy'z'$

$F = \bar{x} + xy + x\bar{z} + x\bar{y}\bar{z} \quad [\because A + \bar{A}B = A + B]$

$F = \bar{x} + y + x\bar{z} + x\bar{y}\bar{z} \quad [\because A + AB = A]$

$F = \bar{x} + y + x\bar{z} \quad [\because A + \bar{A}B = A + B]$

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

KARNAUGH MAP :-

* Method provides a simple forward procedure for minimizing Boolean functions.
* The Map Method is called Karnaugh map or K-Map.

Minimization of SOP form :-

i) one Variable :-

* 0 and 1

A	$\bar{A}$
0	
1	A

ii) Two Variable :-

* $2^n = 2^2 = 4$

A \ B	0	1
0	00	01
1	10	11

=

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

iii) Three Variables :-

A \ BC	00	01	11	10
0	000	001	011	010
1	100	101	111	110

⇒

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

iv) Four Variables :-

AB \ CD	00	01	11	10
00	0000	0001	0011	0010
01	0100	0101	0111	0110
11	1100	1101	1111	1110
10	1000	1001	1011	1010

=

AB \ CD	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	$\bar{A}\bar{B}\bar{C}\bar{D}$	$\bar{A}\bar{B}\bar{C}D$	$\bar{A}\bar{B}CD$	$\bar{A}\bar{B}C\bar{D}$
$\bar{A}B$	$\bar{A}B\bar{C}\bar{D}$	$\bar{A}B\bar{C}D$	$\bar{A}BCD$	$\bar{A}BC\bar{D}$
AB	$AB\bar{C}\bar{D}$	$AB\bar{C}D$	$ABCD$	$ABC\bar{D}$
$A\bar{B}$	$A\bar{B}\bar{C}\bar{D}$	$A\bar{B}\bar{C}D$	$A\bar{B}CD$	$A\bar{B}C\bar{D}$

Grouping two adjacent pairs :-

eg :-

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

$F = \bar{B}$

2 variables :-

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

$F = A + B$

3 Variable :-

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

$F = \bar{A}C$

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

$F = A\bar{C}$

4 Variables :-

AB \ CD	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	0	1	1	0
AB	0	0	0	0
$A\bar{B}$	0	0	0	0

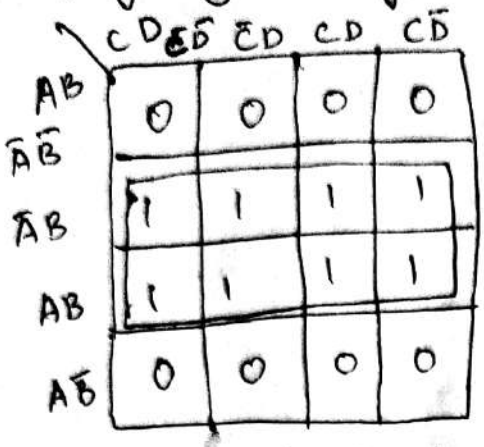
Grouping 4 adjacent Ones (Quad)

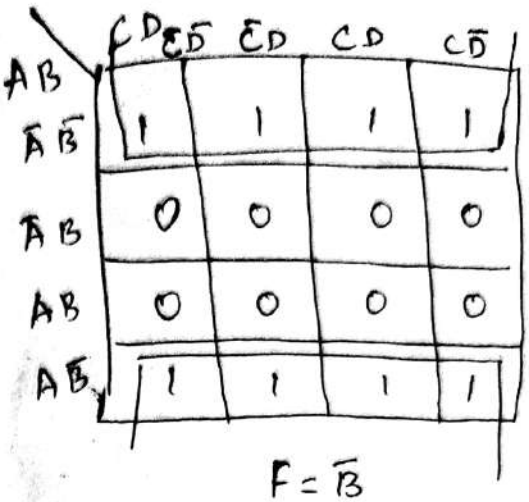
$F = \bar{A}BD$

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

$F = C$

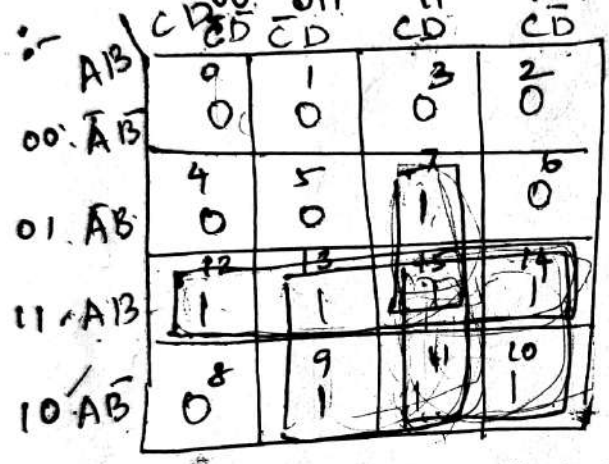
Grouping Eight Adjacent ones (octet):-





Problems:- 1) Simplify the following expression using k-map method $Y = \Sigma m(7,9,10,11,12,13,14,15)$

Soln:-



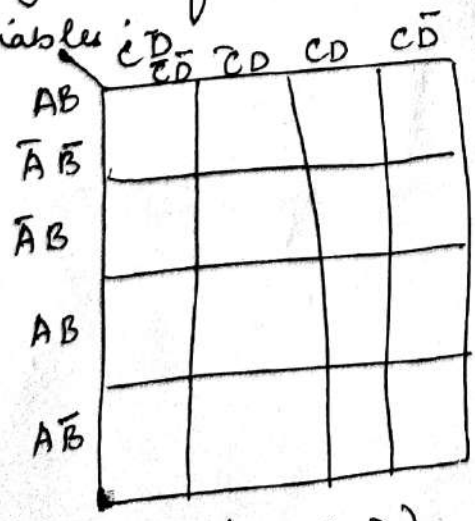
	8 4 2 1
7	0 1 1 1
9	1 0 0 1
10	1 0 1 0
11	1 0 1 1
12	1 1 0 0
13	1 1 0 1
14	1 1 1 0
15	1 1 1 1

$Y(A,B,C,D) = AB + AD + AC + BCD + ACD$

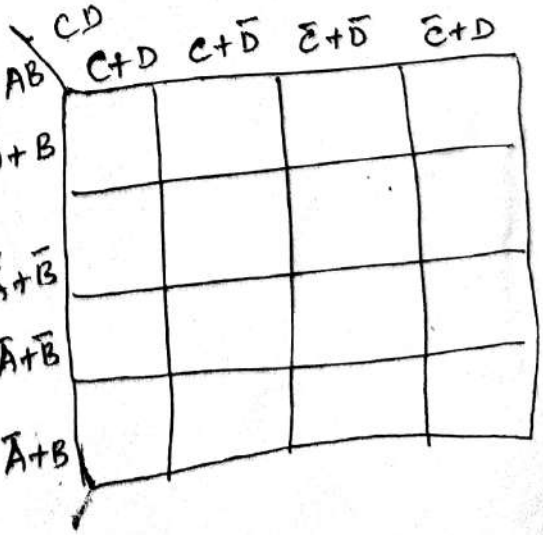
$Y(A,B,C,D) = AB + AD + AC + BCD$

(1') Minimization of POS forms:-

4 Variables



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



1) Simplify Boolean Expression using K-map.

$Y(A,B,C,D) = \pi M(5,7,13,15)$

AB \ CD	C+D	C+$\bar{D}$	$\bar{C}$+$\bar{D}$	$\bar{C}$+D
A+B	0	1	3	2
A+$\bar{B}$	4	5	7	6
$\bar{A}$+$\bar{B}$	12	13	15	14
$\bar{A}$+B	8	9	11	10

⇒

AB \ CD				
	1	1	1	1
	1	0	0	1
	1	0	0	1
	1	1	1	1

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

Don't care condition :-

* The o/p levels are indicated by 'X' (cross) and called as Don't care condition.

'x' and 'd' - denotes the don't care terms.

Problem:- Simplify the following expression using K-map

$Y(A,B,C) = \sum m(0,1,3,5,6) + d(2,4)$

A \ BC	$\bar{B}\bar{C}$	$\bar{B}$C	BC	B$\bar{C}$
$\bar{A}$	0	1	3	2
A	4	5	7	6

A \ BC	$\bar{B}\bar{C}$	$\bar{B}$C	BC	B$\bar{C}$
$\bar{A}$	1	1	1	X
A	X	1	0	1

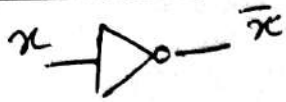
$Y(A,B,C) = \bar{A} + \bar{B} + \bar{C}$

IMPLEMENTATION OF BOOLEAN EXPRESSIONS USING UNIVERSAL GATES:-

Digital logic gates :-

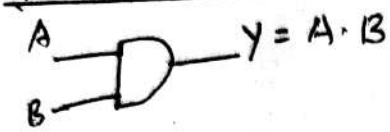
* NOT
* AND
* OR
* NAND
* NOR
* Exclusive-OR (EX-OR)
* Exclusive-NOR (EX-NOR)
* Buffer.

i) NOT Gates :-



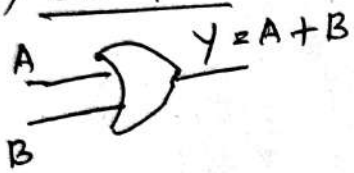
I/p	O/p
x	$\bar{x}$
0	1
1	0

ii) AND Gates :-



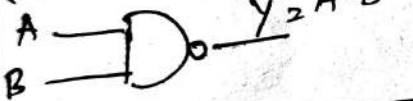
I/p		O/p
A	B	Y = A · B
0	0	0
0	1	0
1	0	0
1	1	1

iii) OR Gates :-

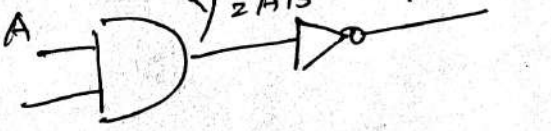


I/p		O/p
A	B	Y = A + B
0	0	0
0	1	1
1	0	1
1	1	1

iv) NAND Gates :-

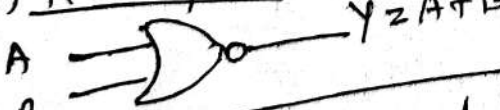


(or)

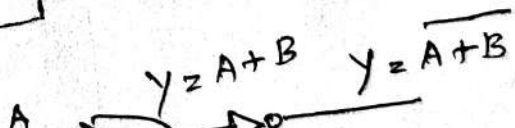


I/p		O/p
A	B	Y = $\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

v) NOR Gates :-

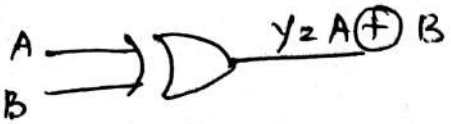


(or)



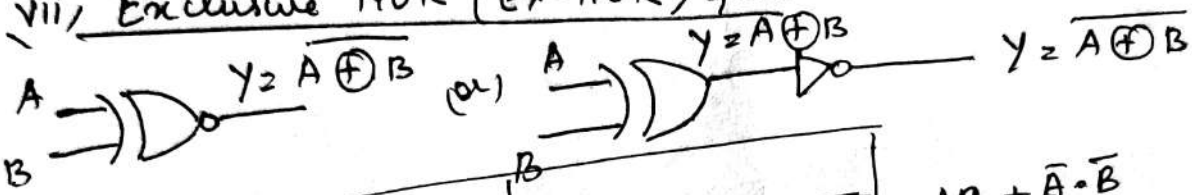
I/p		O/p
A	B	Y = $\overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0

vi) Exclusive (or) (EX-OR) Gate :- $A\bar{B} + \bar{A}B$



I/p		o/p
A	B	$y = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

vii) Exclusive NOR (EX-NOR) Gate :-



I/p		o/p
A	B	$y = \overline{A \oplus B}$
0	0	1
0	1	0
1	0	0
1	1	1

$\rightarrow AB + \bar{A} \cdot \bar{B}$

viii) Buffer :-

I/p	o/p
A	y
0	0
1	1

UNIVERSAL GATES :- (TABULATION METHODS) :-

* NAND and NOR Gates are universal gates.
* AND, OR, NOT → basic gates

NAND GATE :-

Implement AND operation using NAND gate :-

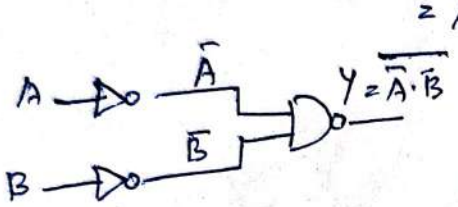
$$Y = A \cdot B$$
$$Y = \overline{\overline{A \cdot B}} \quad (\because \overline{\overline{A}} = A)$$

A	B	$\overline{AB}$	$\overline{A}\overline{B}$
0	0	1	0
0	1	1	0
1	0	1	0
1	1	0	1

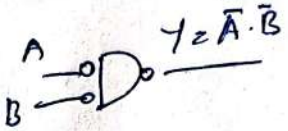
Implement OR operation using NAND Gate:-

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

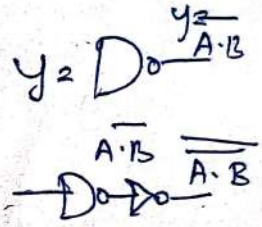
$$= \overline{\overline{A}\cdot\overline{B}}$$



(or)



A	B	$\overline{A}$	$\overline{B}$	$\overline{A}\cdot\overline{B}$	$\overline{\overline{A}\cdot\overline{B}}$
0	0	1	1	1	0
0	1	1	0	0	1
1	0	0	1	0	1
1	1	0	0	0	1



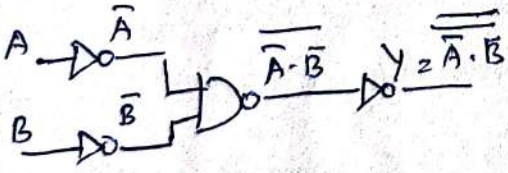
$= A\cdot B$.

Implement NOT operation using NAND Gate:-

$$Y = A\overline{B} = X\overline{X} = \overline{X}+\overline{X} = \overline{X}$$

Implement NOR operation using NAND Gate:-

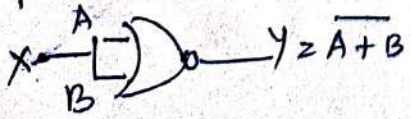
$$Y = \overline{\overline{A}\cdot\overline{B}}$$



A	B	A + B	$\overline{A+B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

NOR Gate:-

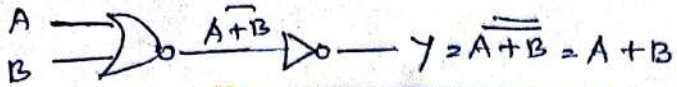
Implement NOT function using NOR Gate:-



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

Implement OR function using NOR Gate :-

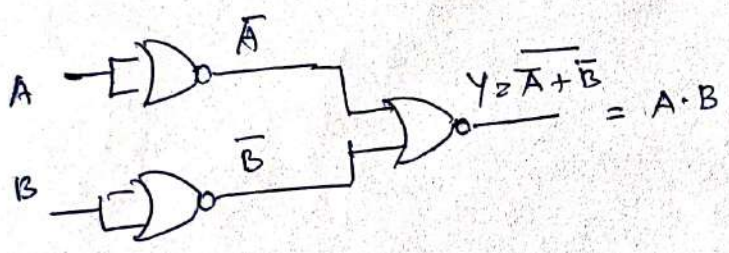
$\overline{\overline{A+B}} = A+B$.



A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

Implement AND function using NOR Gate :-

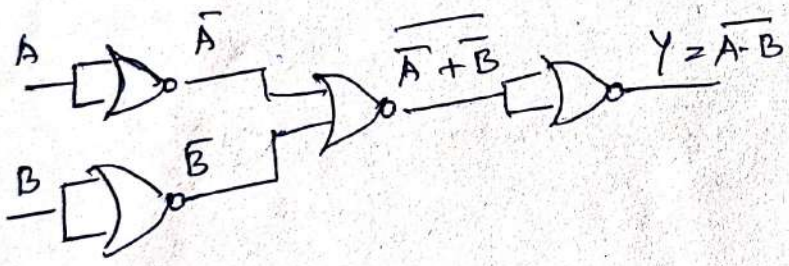
$Y = A \cdot B \Rightarrow \overline{\overline{A}+\overline{B}}$



A	B	A · B
0	0	0
0	1	0
1	0	0
1	1	1

Implement NAND function using NOR Gate :-

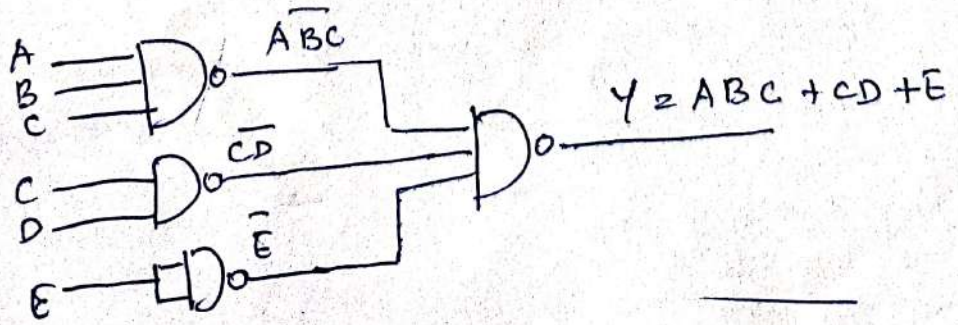
$Y = A \cdot B = \overline{\overline{A}+\overline{B}}$



A	B	$\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

NAND - NAND Implementation :-

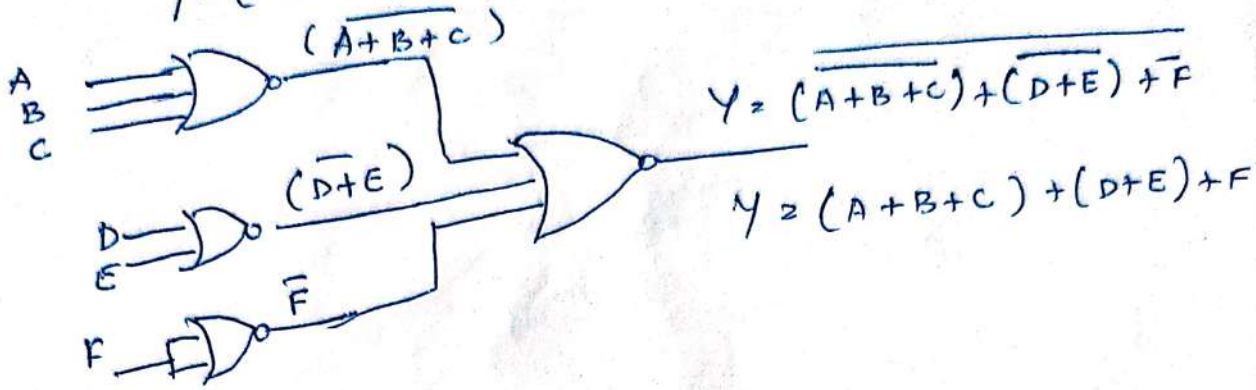
* AND - OR logic $Y = ABC + CD + E$ (SOP)



NOR-NOR Implementation:-

* POS Form

$$Y = (A+B+C)(D+E)F$$



TABULATION METHOD:- (253).

* Quine-McCluskey

Procedure:- S_1: Minterms in binary form

S_2: Minterms to no. of 1's and split term

S_3:- Compare binary number with every term next higher category.

S_4:- Repeat S_3

S_5:- Prime implicants.

S_6: Prime Implicant chart.

Eg:- $f(a,b,c,d) = \sum m(0,1,2,3,8,9)$

Tabulation Method (or) Quine McCluskey Method.

minterms	Binary Representation
m_0	0000
m_1	0001
m_2	0010
m_3	0011
m_8	1000
m_9	1001

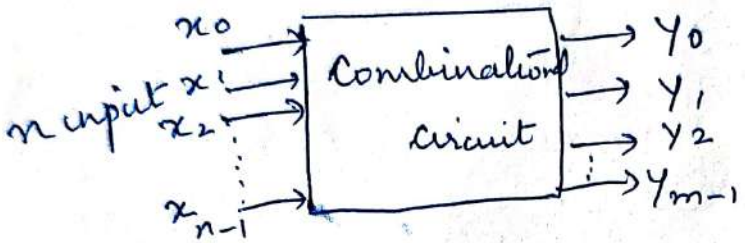
minterms	Binary Representation	Group
m_0	0000	Group 0 & 1's zero
m_1	0001	→ Group 1
m_2	0010	
m_8	1000	

PROBLEM FORMULATION AND DESIGN OF COMBINATIONAL CIRCUITS - CODE - CONVERTERS :-

A Digital system have combinational (or) sequential.

Combinational circuit :-

* O/p at any time are determined from the Present combination of inputs.



Design Procedure :-

S1 :- Determine i) no. of i/p
ii) no. of o/p's.

S2 :- Truth table determine b/w i/p and o/p's.

S3 :- Boolean function of i/p variables using K-map (or) Quine Mc clusky method.

S4 :- Draw Logic diagram.

Pratical Design i) Minimum no. of gates
ii) min no. of i/p gates
iii) min. propagation time of signal through the circuit
iv) min. no. of interconnections
v) limitation of driving capabilit

CODE CONVERTERS :-

1) Binary to BCD converters.
2) BCD to Binary converters
3) BCD to Excess 3 code converters
4) Excess 3 to BCD code
5) Binary to Gray code Converter
6) Gray to Binary code.

Binary code				BCD code				
D	C	B	A	B_4	B_3	B_2	B_1	B_0
0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	1
0	0	1	0	0	0	0	1	0
0	0	1	1	0	0	0	1	1
0	1	0	0	0	0	1	0	0
0	1	0	1	0	0	1	0	1
0	1	1	0	0	0	1	1	0
0	1	1	1	0	0	1	1	1
1	0	0	0	0	1	0	0	0
1	0	0	1	0	1	0	0	1
1	0	1	0	1	0	0	0	0
1	0	1	1	1	0	0	0	1
1	1	0	0	1	0	0	1	0
1	1	0	1	1	0	0	1	1
1	1	1	0	1	0	1	0	0
1	1	1	1	1	0	1	0	1

2) BCD to Binary converters :-

BCD i/p					Binary o/p's.				
B_4	B_3	B_2	B_1	B_0	A	B	C	D	E
0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	0	0	1	0
0	0	0	1	1	0	0	0	1	1
0	0	1	0	0	0	0	1	0	0
0	0	1	0	1	0	0	1	0	1
0	0	1	1	0	0	0	1	1	0
0	0	1	1	1	0	0	1	1	1
0	1	0	0	0	0	1	0	0	0
0	1	0	0	1	0	1	0	0	1

3) BCD to Excess -3 code Converter :-

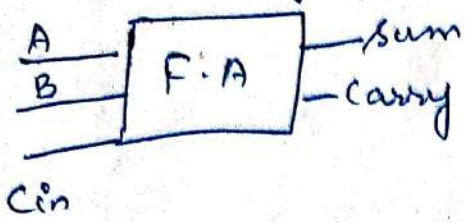
BCD code				Excess -3 code.			
B_3	B_2	B_1	B_0	E_3	E_2	E_1	E_0
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	0	1	0	0	1	1	1
0	1	0	0	1	0	0	0
0	1	0	1	1	0	0	1
0	1	1	0	1	0	1	0
0	1	1	1	1	0	1	1
1	0	0	0	1	1	0	0
1	0	0	1				

4) Excess -3 TO BCD converter :-

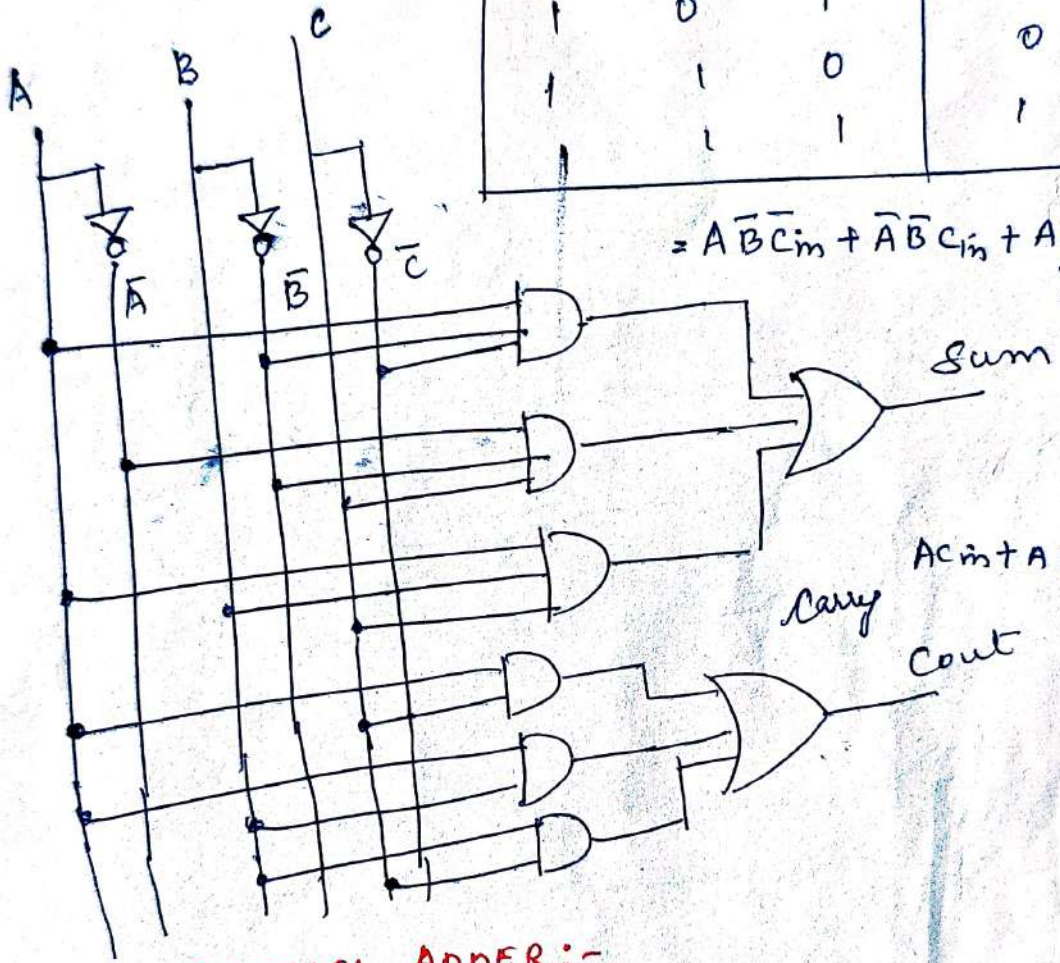
Excess -3				BCD			
E_3	E_2	E_1	E_0	B_3	B_2	B_1	B_0
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	1	0	0	0	1	1
0	1	1	1	0	1	0	0
1	0	0	0	0	1	0	1
1	0	0	1	0	1	1	0
1	0	1	0	0	1	1	1
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	1

ii) Full adder :-

* 3 binary i/p produce o/p sum and carry bit.



A	B	C_{in}	S	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

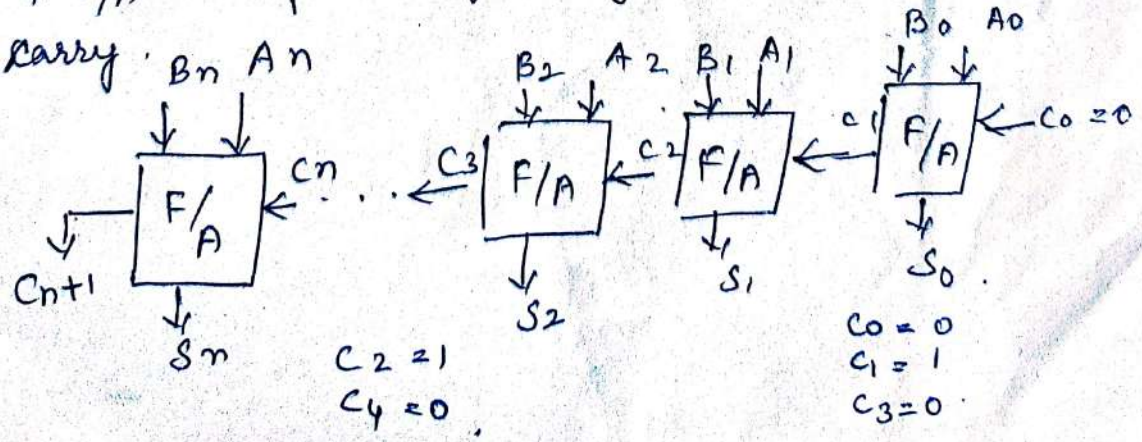


$= A\bar{B}\bar{C}_{in} + \bar{A}\bar{B}C_{in} + ABC_{in} + \bar{A}BC_{in}$

$AC_{in} + AB + BC_{in}$

BINARY PARALLEL ADDER :-

* F/A is capable of adding two 1 bit number and i/p carry.



CARRY LOOK AHEAD ADDER :-

* The signal must propagate through the gates has required some time delay is known as Propagation delay.

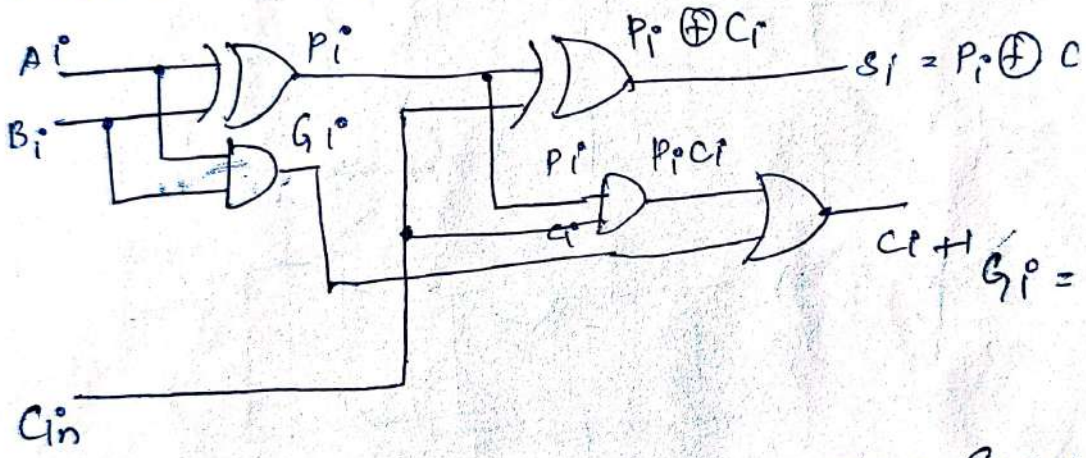
* Augend (A) and Addend (B).

Augend A = 0101 Addend B = 0011

Cin	1	1	1	0
Augend — A	0	1	0	1
Added :- B	0	0	1	1
S	1	0	0	0
Cout	0	1	1	1

Propogation delay time in an adder is the time it takes the carry to propagate through the F.A.

* Two Functions :-



* Two binary Variable Carry propagate & Carry generate

$P_i = A_i \oplus B_i$ → carry propagate

$G_i = A_i B_i$ → Carry generate.

O/p sum & carry.

$S_i = P_i \oplus C_i$

$C_{i+1} = G_i + P_i C_i$

* Boolean fn Carry O/p's.

C_0 = I/p carry

$C_1 = G_0 + P_0 C_0$

$C_2 = G_1 + P_1 G_0 + P_1 P_0 G_0$

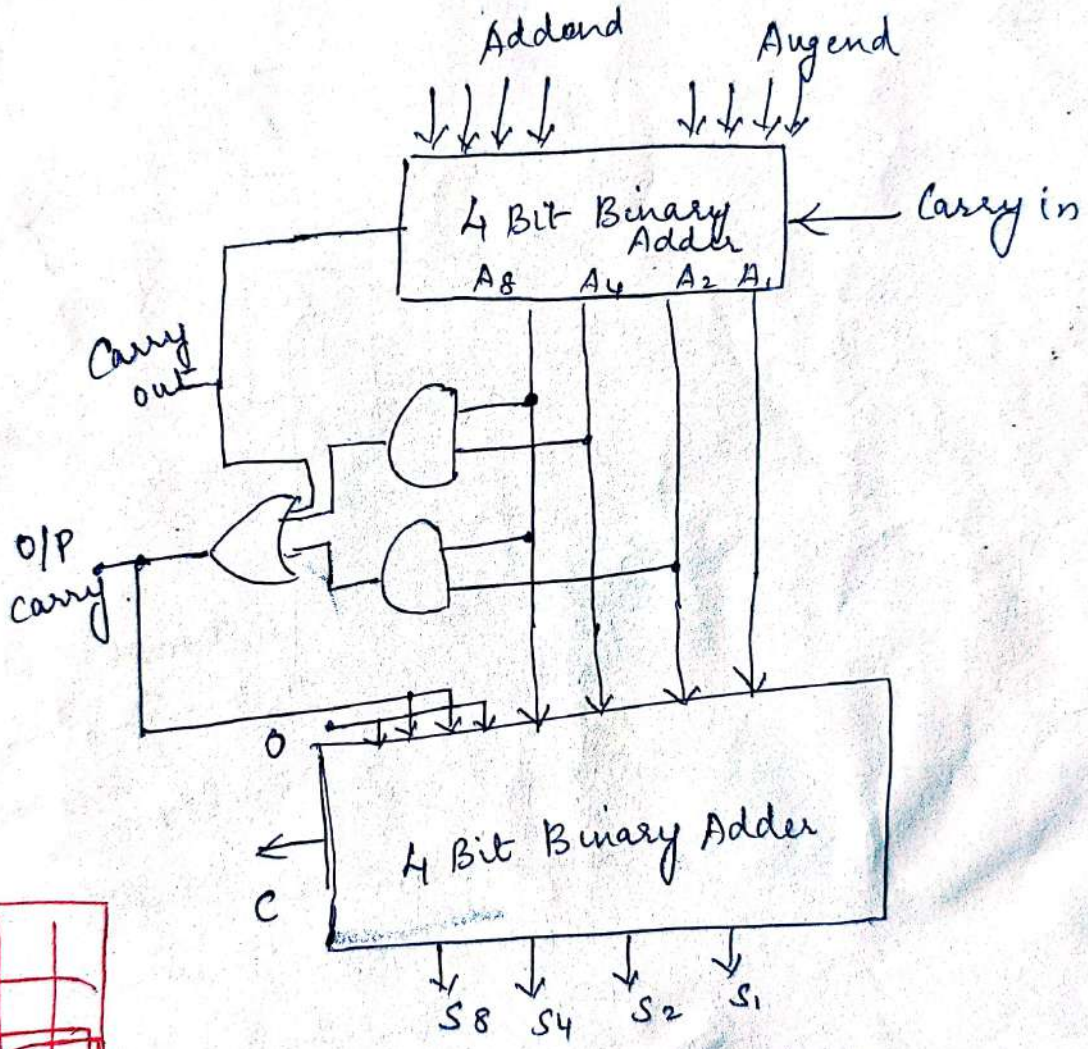
$C_3 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$

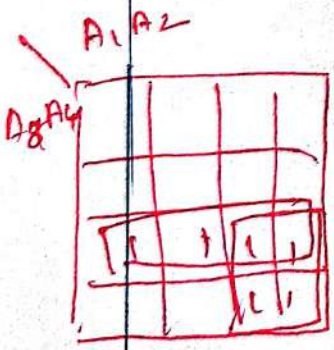
BCD ADDER

* BCD adder is a circuit that adds two BCD digits in parallel and produce a Sum in BCD.
* 0000 to 1001 (10 digit) each digit coded 4-bit binary number.

Requirement to implement BCD Adder.

* Two 4-bit binary adder.
* Logic circuit to detect Sum greater than 9.





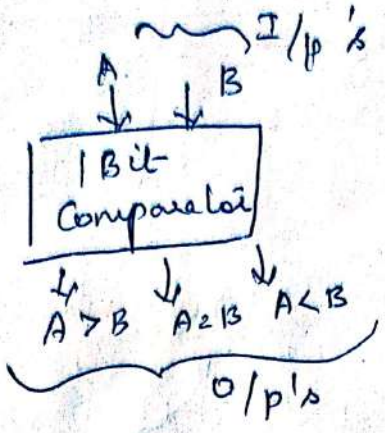
$$C = K + A_8A_4 + A_8A_2$$

MAGNITUDE COMPARATOR :-

* Comparison of two numbers is an operation that determines one number greater than, less

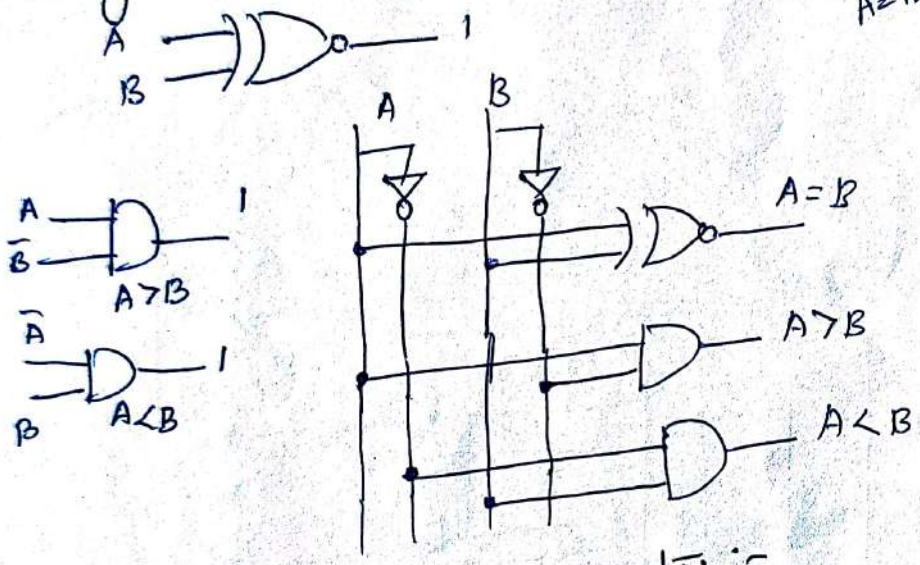
a equal to the other number.

* 3 binary Variables $A > B$, $A = B$ (OR) $A < B$.



Single Bit magnitude Comparator :-

$A=B \rightarrow$ EX-NOR.



A	B	O/p
0	0	A = B
0	1	A < B
1	0	A > B
1	1	A = B

2-Bit Magnitude comparator :-

* A_1, A_0 and B_1, B_0 and O/p $A > B$, $A = B$, $A < B$.

K-map :- $(A = B) = (A_0 \odot B_0)(A_1 \odot B_1)$.

4-Bit Magnitude comparator :-

* Gives one of the O/p's, $A = B$, $A < B$, $A > B$.

Let $A = A_3 A_2 A_1 A_0$

$B = B_3 B_2 B_1 B_0$

DECODER :-

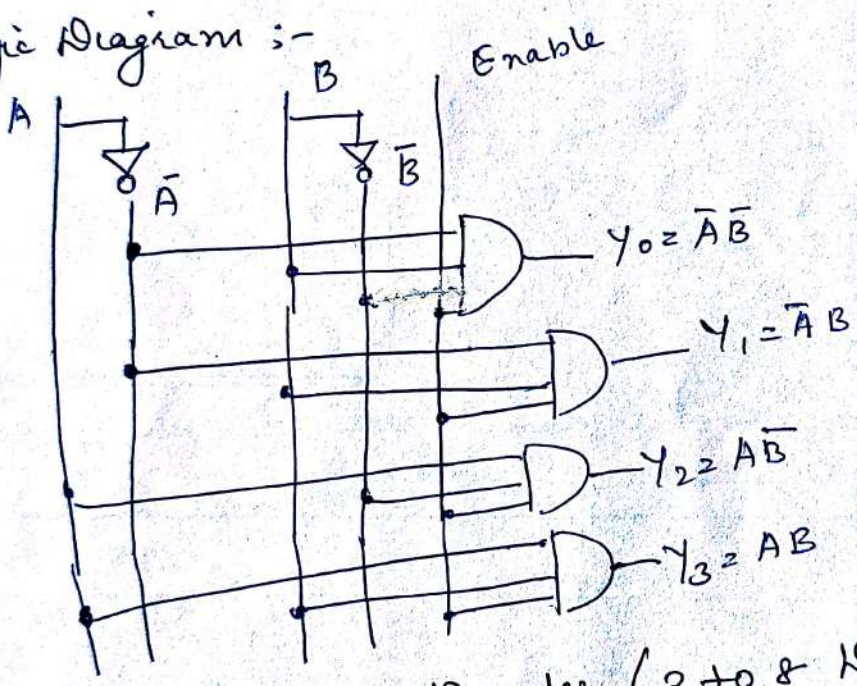
* A decoder is a combinational circuit that converts binary information from n input lines to max. 2^n O/p lines.

* Two to Four line Decoder :-

I/p's			O/p's			
E	A	B	Y_0	Y_1	Y_2	Y_3
0	X	X	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

$Y_0 = \bar{A}\bar{B}$, $Y_1 = \bar{A}B$, $Y_2 = A\bar{B}$, $Y_3 = AB$.

Logic Diagram :-



ii) Binary to Octal Decoder (3 to 8 Decoder) :-

$Y_0 = \bar{A}\bar{B}\bar{C}$, $Y_1 = \bar{A}\bar{B}C$, $Y_2 = \bar{A}B\bar{C}$, $Y_3 = \bar{A}BC$, $Y_4 = A\bar{B}\bar{C}$

$Y_5 = A\bar{B}C$, $Y_6 = AB\bar{C}$, $Y_7 = ABC$.

iii) BCD To Decimal Decoder :-

* BCD has Value 0 to 9 with four bit binary
* A i/p A, B, C, D and 10 o/p $Y_0 - Y_9$.

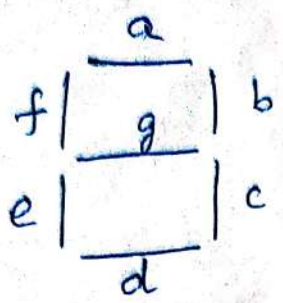
$Y_0 = \bar{A}\bar{B}\bar{C}\bar{D}$, $Y_1 = \bar{A}\bar{B}\bar{C}D$, $Y_2 = \bar{A}B\bar{C}\bar{D}$, $Y_3 = \bar{A}\bar{B}CD$,

$Y_4 = \bar{A}B\bar{C}\bar{D}$, $Y_5 = \bar{A}B\bar{C}D$, $Y_6 = \bar{A}BC\bar{D}$, $Y_7 = \bar{A}BCD$,

$Y_8 = A\bar{B}\bar{C}\bar{D}$, $Y_9 = A\bar{B}\bar{C}D$.

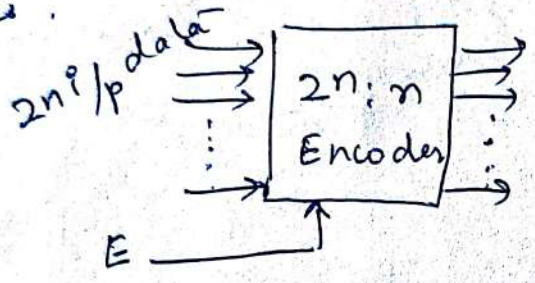
iii) BCD to Seven Segment Decoder :-

* Decimal Digit ⇒ 0 to 9



ENCODER :-

* Encoder is a digital circuit, 2^n i/p lines and n o/p lines.



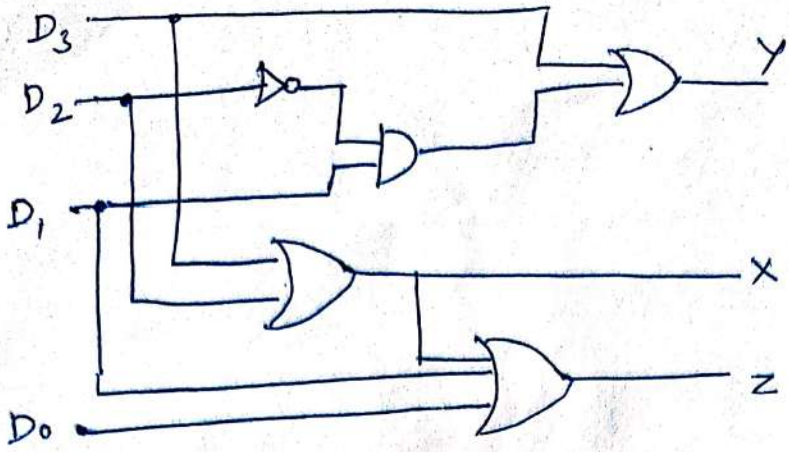
PRIORITY ENCODER :-

* Priority encoder such that two or more i/p's are equal to 1 at same time.

I/p's				O/p's		
D_0	D_1	D_2	D_3	x	y	z
0	0	0	0	x	x	0
1	0	0	0	0	0	1
x	1	0	0	0	1	1
x	x	1	0	1	0	1
x	x	x	1	1	1	1

$X = D_2 + D_3$, $Y = D_3 + D_1\bar{D}_2$,
$Z = D_0 + D_1 + \bar{D}_2 + D_3$

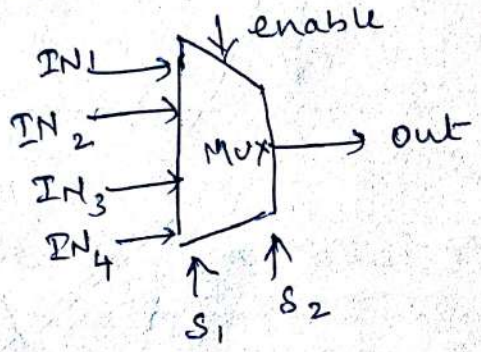
Logic diagram four-input priority encoder :-



MUX/DEMUX :-

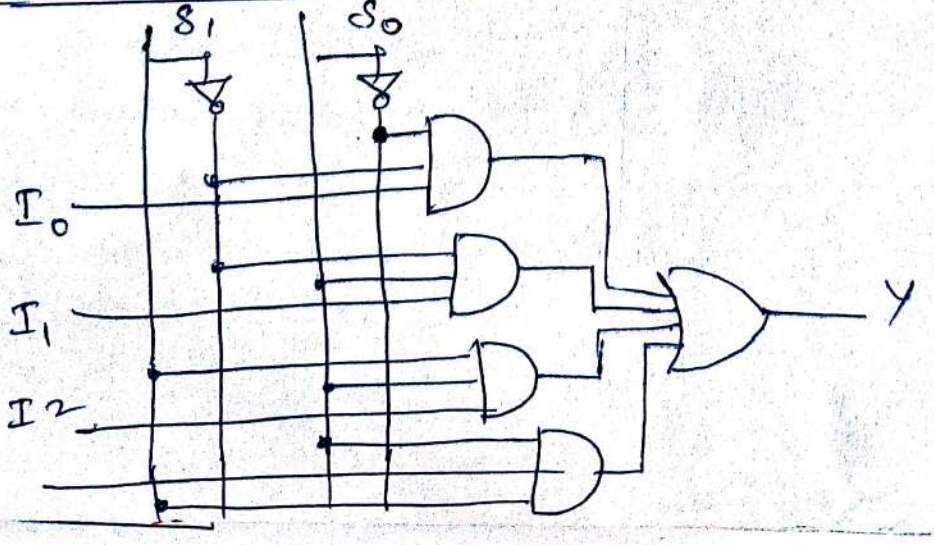
Multiplexers :-

* Many to one
* 2^n I/p lines and n selection lines.



1) 4 to 1 Multiplexer :-

Data I/p	Selection lines S_1	S_0	O/p Y
I_0	0	0	I_0
I_1	0	1	I_1
I_2	1	0	I_2
I_3	1	1	I_3



ii) 8 to 1 MUX :-

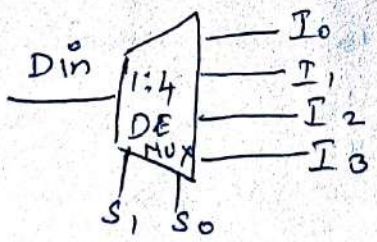
* 8 input lines $2^3 = 8$

Data I/p	Selection lines			O/p Y
	S_2	S_1	S_0	
I_0	0	0	0	I_0
I_1	0	0	1	I_1
I_2	0	1	0	I_2
I_3	0	1	1	I_3
I_4	1	0	0	I_4
I_5	1	0	1	I_5
I_6	1	1	0	I_6
I_7	1	1	1	I_7

DEMUX :-

* Demultiplexer is a circuit that receives information on a single line & Transmit this information on n O/p lines..

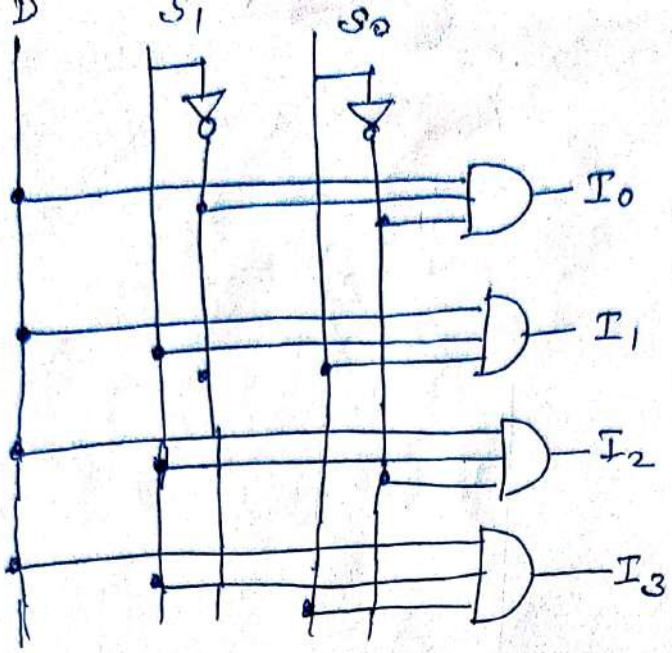
* Din has path to all four O/p information. It is also called 'Data Distributor'



i) 1 to 4 Demultiplexer :-

* Single I/p (D) and 4 O/p (I_0 to I_3), Two selection lines (S_1 and S_0)

I/p's			O/p s			
S_1	S_0	Data I/p	I_0	I_1	I_2	I_3
0	0	D	D	0	0	0
0	1	D	0	D	0	0
1	0	D	0	0	D	0
1	1	D	0	0	0	D



1 to 8 Demultiplexer :-

* Single I/p and 8 O/p (I_0 to I_7)

$I_0 = D \cdot \bar{S_2} \cdot \bar{S_1} \cdot \bar{S_0}$ $\quad I_3 = D \cdot \bar{S_2} \cdot S_1 S_0$ $\quad I_6 = D \cdot S_2 \cdot S_1 \cdot \bar{S_0}$

$I_1 = D \cdot \bar{S_2} \cdot \bar{S_1} \cdot S_0$ $\quad I_4 = D \cdot S_2 \cdot \bar{S_2} \bar{S_0}$ $\quad I_7 = D \cdot S_2 \cdot S_1 S_0$

$I_2 = D \cdot \bar{S_2} \cdot S_1 \cdot \bar{S_0}$ $\quad I_5 = D \cdot S_2 \cdot \bar{S_1} \cdot S_0$

CASE STUDY:

PARITY GENERATOR / CHECKER:-

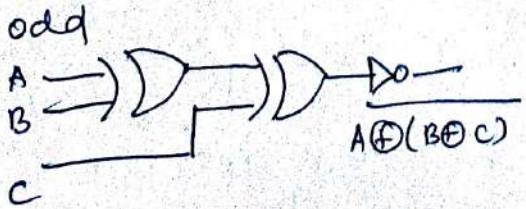
* Purpose of detecting errors during Transmission of binary information
* Parity bit in Transmitter called Parity generator
* Parity bit in Receiver is parity checker.

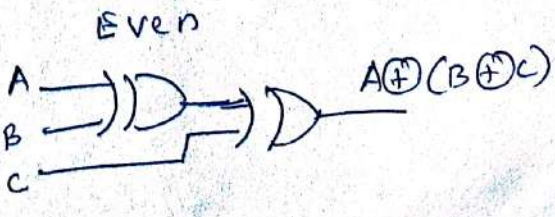
Even Parity :- → make the number of 1's an even no.

Odd Parity → make the number of 1's as odd no.

$$\text{odd} = \overline{A \oplus (B \oplus C)}$$

$$\text{Even} = A \oplus (B \oplus C)$$

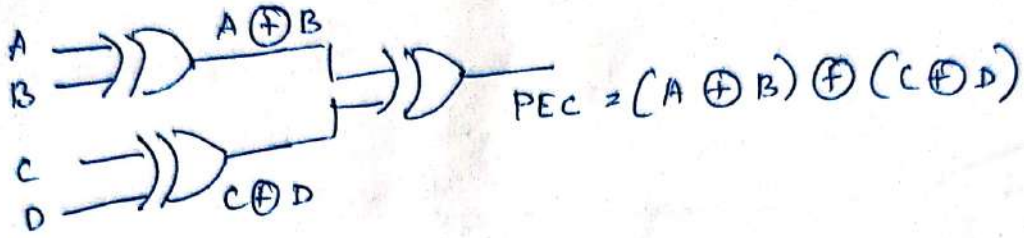




Parity checker:-

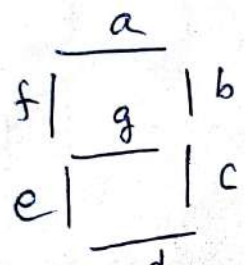
* check the error.
* Parity error checker will be 1.
  will be 0 if no error.

$$PEC = (A \oplus B) \oplus (C \oplus D)$$



SEVEN SEGMENT DISPLAY DECODER:-

* 7 o/p (a, b, c, d, e, f, g)



I/p				o/p							Display Digit
A	B	C	D	a	b	c	d	e	f	g	
0	0	0	0	1	1	1	1	1	1	0	0
0	0	0	1	0	1	1	0	0	0	0	1
0	0	1	0	1	1	0	1	1	0	1	2
0	0	1	1	1	1	1	1	0	0	1	3
0	1	0	0	0	1	1	0	0	1	1	4
0	1	0	1	1	0	1	1	0	1	1	5
0	1	1	0	1	0	1	1	1	1	1	6
0	1	1	1	1	1	1	0	0	0	0	7
1	0	0	0	1	1	1	1	1	1	1	8
1	0	0	1	1	1	1	1	0	1	1	9

UNIT-III (SYNCHRONOUS SEQUENTIAL CIRCUITS):-

LATCHES:-

* o/p of latch changes when i/p changes. (store element till the power is delivered)
  1) Gated SR latch.
  2) Gated D latch.

Gated SR Latch:-

* 2 i/p Set(S) and Reset(R), two universal gates NAND or NOR.

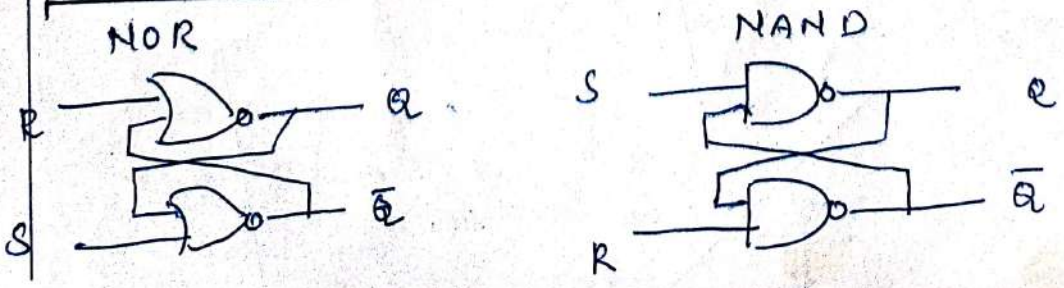
NOR Based SR Latch

I/p's S	I/p's R	o/p's Q_{n+1}	o/p's $\bar{Q}_{n+1}$	Remarks
0	0	Q_n	$\bar{Q}_n$	no change
0	1	0	1	Reset
1	0	1	0	Set
1	1	x	x	Indeterminate

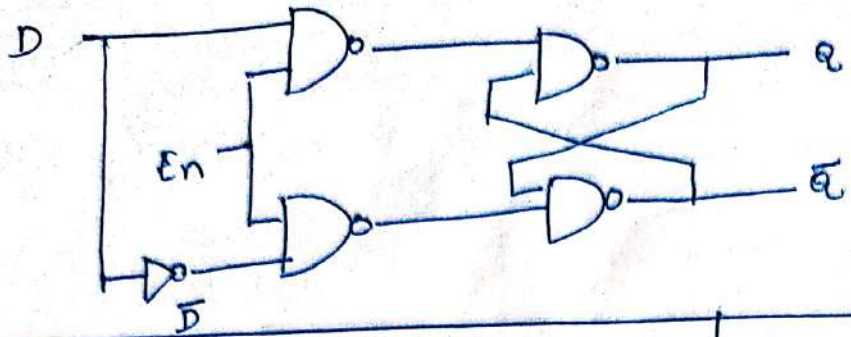
NAND Based SR Latch

I/p S	I/p R	o/p Q_{n+1}	o/p $\bar{Q}_{n+1}$	Remark
0	0	x	x	Indeterminate
0	1	1	0	Set
1	0	0	1	Reset
1	1	Q_n	$\bar{Q}_n$	no change

Q_n = Present state
Q_{n+1} = Next state



Gate D Latch:-



I/p		o/p		Remark
En	D	Q_n	Q_{n+1}	
0	x	x	Q_n	no change
1	0	x	0	Reset
1	1	x	1	Set

Q_n - Present state

Q_{n+1} - next state

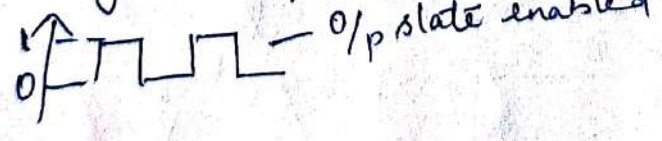
FLIP FLOPS:-

* Storage elements that operates with signal levels (rather than signal Transitions) are referred as Latches,
* Latches are controlled by clock called Flip flops.
  i) Level Triggering
  ii) Edge Triggering

Level Triggering:-

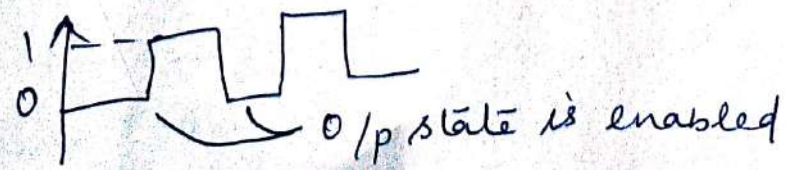
a) +ve level Triggering:-

* I/p changes only when its enable i/p (1) (HIGH)



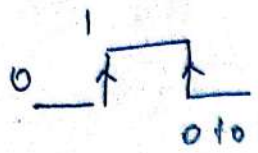
b) -ve level :-

* i/p changes when enable i/p (0) (LOW)

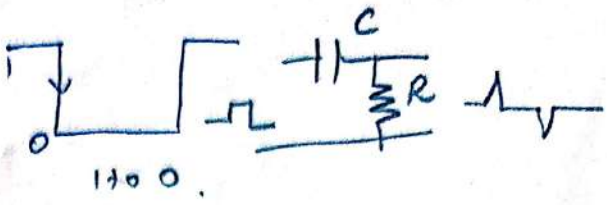


ii) Edge Triggering :-

+ve Pulse

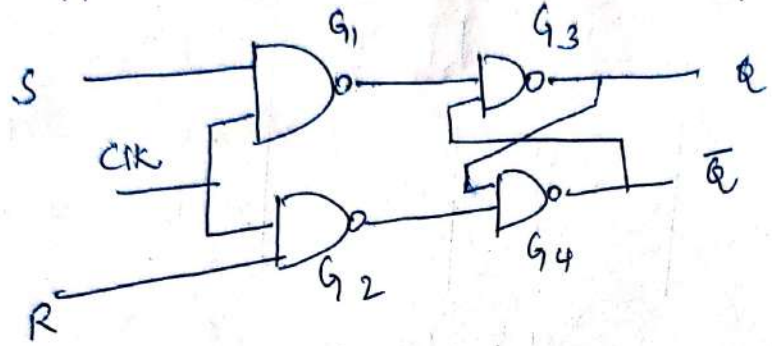


-Ve Pulse



SR Flip Flops :-

* 3 i/p → Clock, Set and Reset, 2 o/p Q and $\bar{Q}$



$Q_n = 0$
$\bar{Q}_n = 1$

Characteristic Table :-

Present	Data i/p's		Next State	Remarks
Q_n	S	R	Q_{n+1}	
0	0	0	0	no change
0	0	1	0	Reset
0	1	0	1	Set
0	1	1	X	Indeterminate
1	0	0	1	No change
1	0	1	0	Reset
1	1	0	1	Set
1	1	1	X	Indeterminate

Characteristic equation :-

Q_n \ SR	$S\bar{R}$	$\bar{S}\bar{R}$	$\bar{S}R$	SR
Q_n	0	0	X	1
$\bar{Q}_n$	1	0	X	1

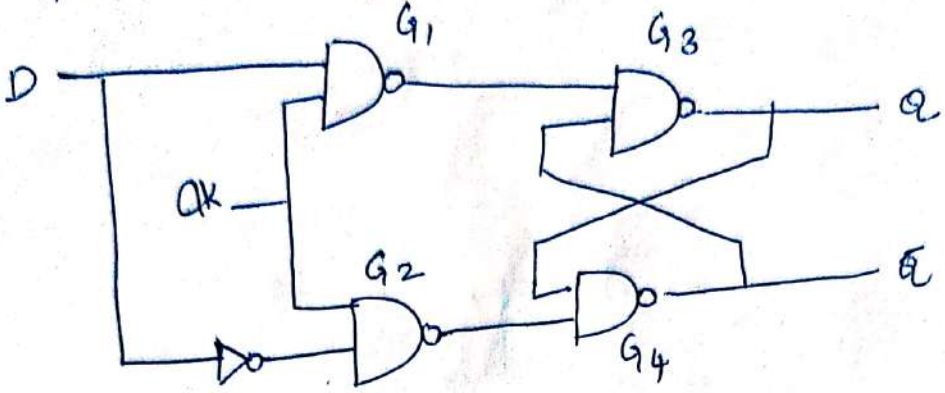
$$Q_{n+1} = S + Q_n\bar{R}$$

Excitation Table

Present state	Next state	I/P's	
Q_n	Q_{n+1}	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

D FLIP FLOP :-

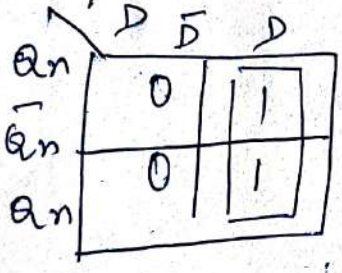
* 2 I/p and 2 O/p.
* I/p's are Delay (D) and clk (clock), Outputs Q and $\bar{Q}$



Characteristic Table :-

Present state	Data I/p's	Next state
Q_n	D	Q_{n+1}
0	0	0
0	1	0
1	0	0
1	1	1

Characteristic equations :-



$Q_{n+1} = D$

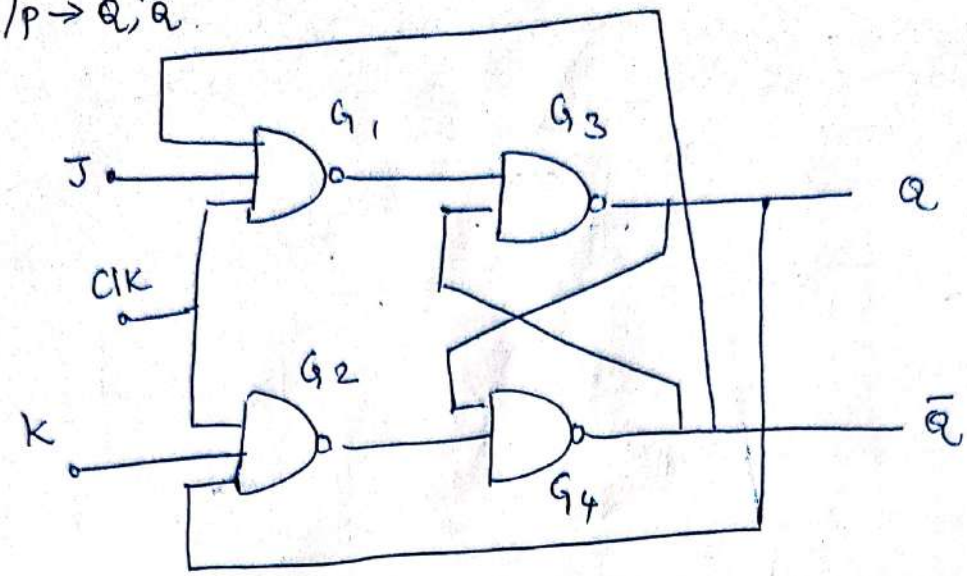
Excitation Table :-

Present state	Next state	Input
Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

JK FLIP FLOP :-

* 3 I/p and 2 O/p's

* I/p → CLK, J, K.
* O/p → Q, $\bar{Q}$.



Characteristic Table :-

Present State	Data i/p's		Next state	Remarks
Q_n	J	K	Q_{n+1}	
0	0	0	0	no change
0	0	1	0	Reset
0	1	0	1	Set
0	1	1	1	Toggle
1	0	0	1	no change
1	0	1	0	Reset
1	1	0	1	Set
1	1	1	0	Toggle

Characteristic equation :-

	$\bar{J}\bar{K}$	$\bar{J}K$	JK	$J\bar{K}$
Q_n	0	0	1	1
$\bar{Q}_n$				
Q_n	1	0	0	1

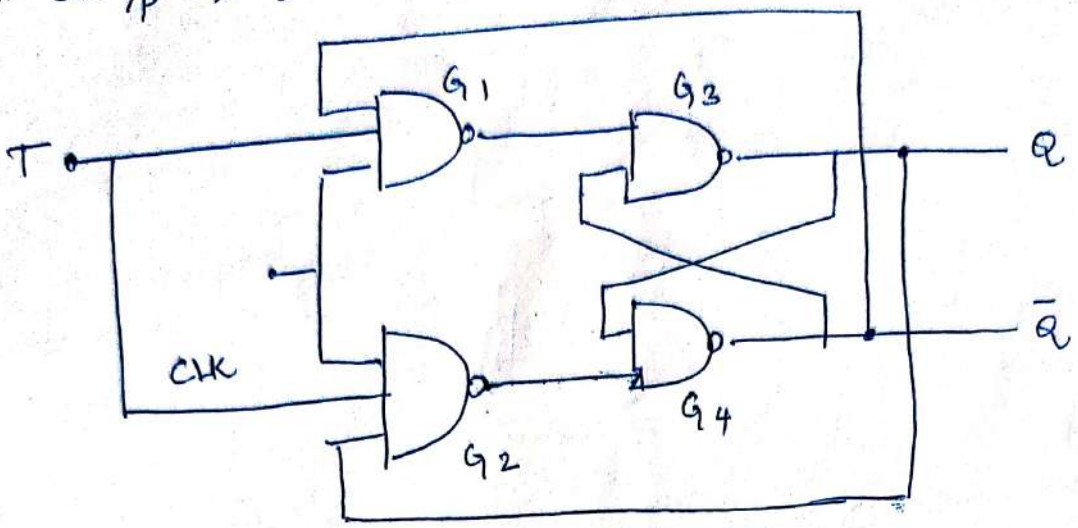
$$Q_{n+1} = \bar{Q}_n J + Q_n \bar{K}$$

Excitation Table :-

P.S Q_n	N.S Q_{n+1}	I/P J	I/P k
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

T-FLIP FLOP :-

* Two i/p T(Toggle) and CLK (Clock)
* 2 o/p → Q and $\bar{Q}_n$



Characteristic Table :-

Present state	Data I/p	Next state
Q_n	T	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

Characteristic equations :-

	T	$\bar{T}$	T
Q_n	0	1	
$\bar{Q}_n$			
Q_n	1	0	

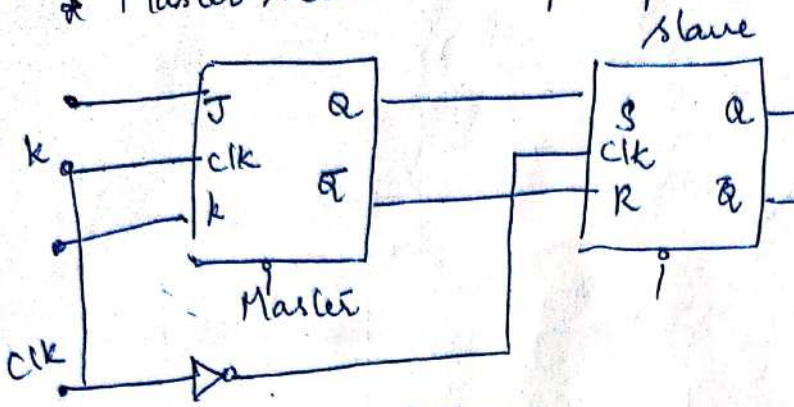
$$Q_{n+1} = \bar{Q}_n T + Q_n \bar{T}$$

Excitation Table :-

Present state	Next state	Input
Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

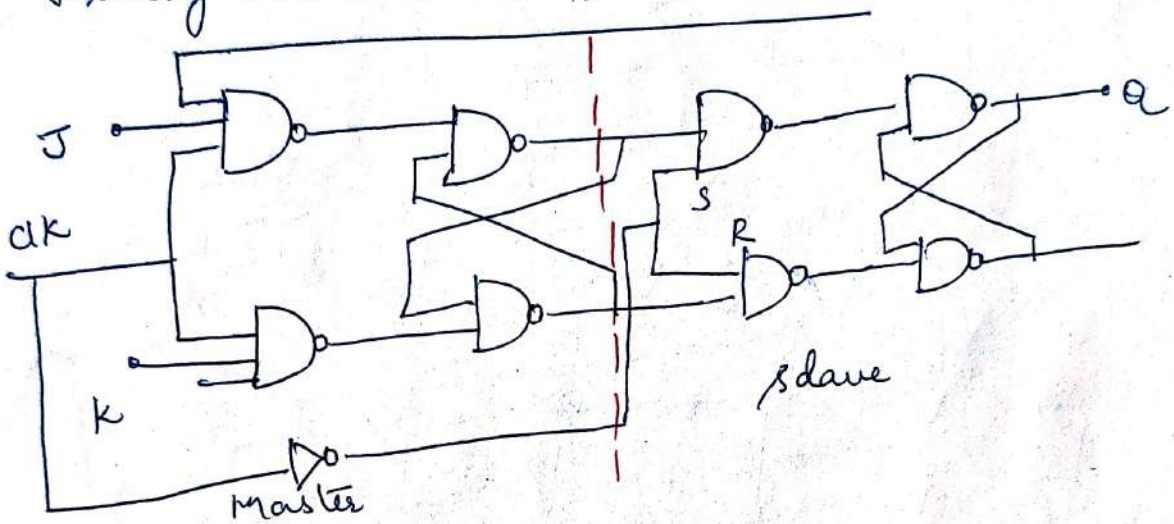
MASTER-SLAVE FLIP FLOP:-

* Master Slave JK Flip Flop.



Case 1:-
J=1, K=0, clk=H
→ Ma ↑, S↑, R, ↓, Q→↑, Q̄ ↓
C2: J=0, K=0, Q ↓, Q̄ ↑
C3: J=1, K=1
FF → Set
C4: J & K → ↓
no change.

* using NAND Gate:-



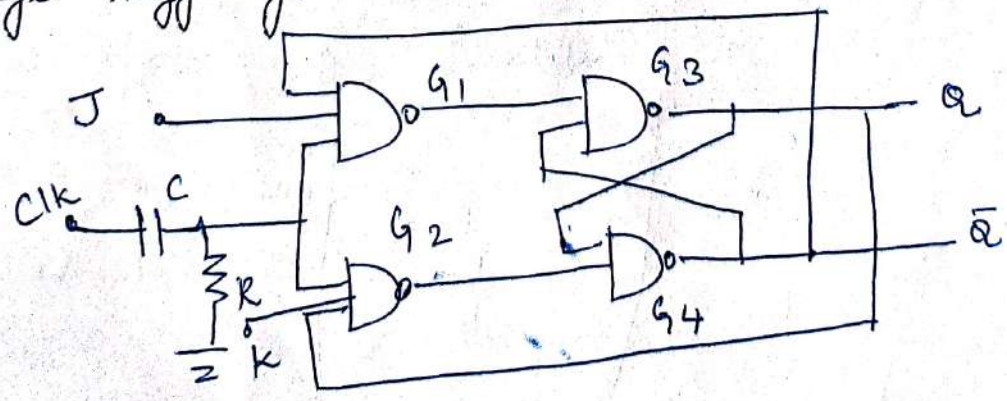
EDGE TRIGGERED JK FLIP FLOP:-

* FF is switched by momentary change in i/p signal. Momentary change called Trigger.

1. Level Triggering 2. Edge Triggering.

Level Triggering:-
HIGH or LOW.

Edge Triggering:-

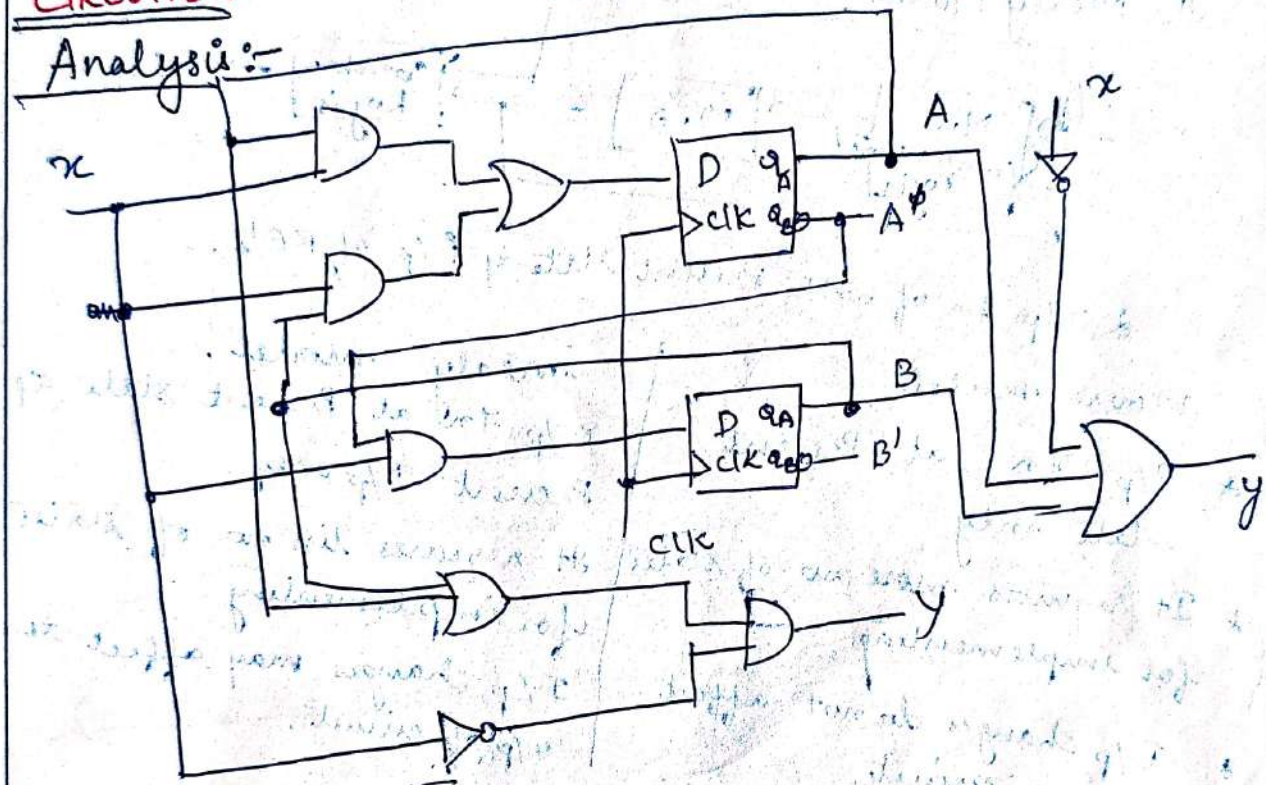


Truth Table :-

Inputs			o/p's		
Clk	J	K	Q_{n+1}	$\bar{Q}_{n+1}$	Remarks
↑	0	0	Q_n	$\bar{Q}_n$	no change
↑	0	1	0	1	Reset
↑	1	0	1	0	set
↑	1	1	$\bar{Q}_n$	Q_n	Toggle

ANALYSIS AND DESIGN OF CLOCKED SEQUENTIAL CIRCUITS :-

Analysis :-



i) state equation
ii) state Table
iii) state diagram
iv) Flip flop eqn's.

$A(n+1) = Ax + Bx$

$B(n+1) = \bar{A}x.$

i) State eqn :- $y(n) = (A+B)\bar{x}$

ii) state Table :-

Time sequence of i/p's, o/p's and FF states is called state table.

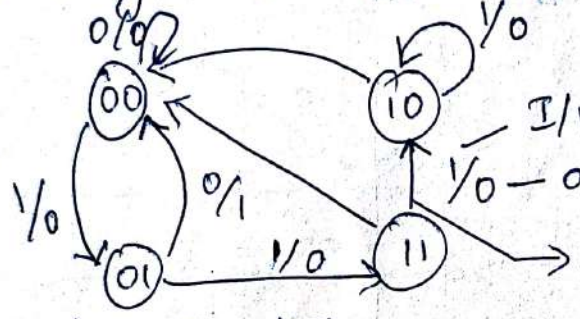
i) I/p ii) P.S iii) N.S iv) o/p

Present state		I/p (x)	Next state		O/p y
A	B		A	B	
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

Formula: $\bar{Q}_A Q_B$ | $y = \bar{x}Q_B + xQ_A$

(40)

iii) State Diagram :-



iv) FF input equation :-

$$D_B = \bar{A}x.$$

Design of clocked Sequential circuits :-

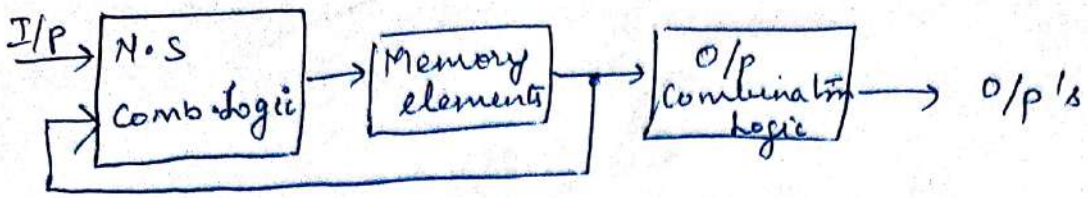
Flow design :-

- State diagram
- ↓ State Table
- ↓ State Reduction
- ↓ Excitation Table
- ↓ k-map simplification
- ↓ Logic diagram

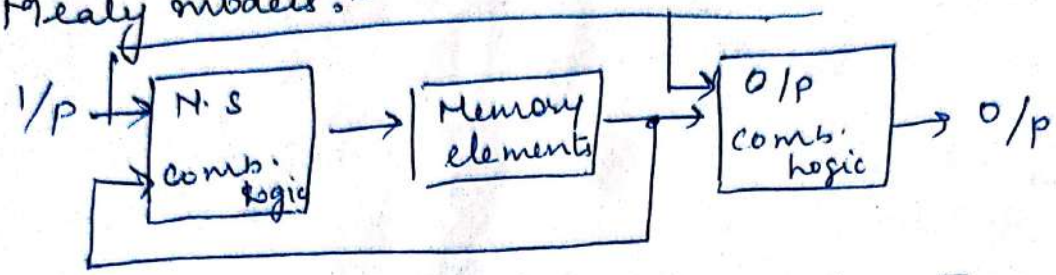
MOORE / MEALY MODELS :-

i) Moore models :-

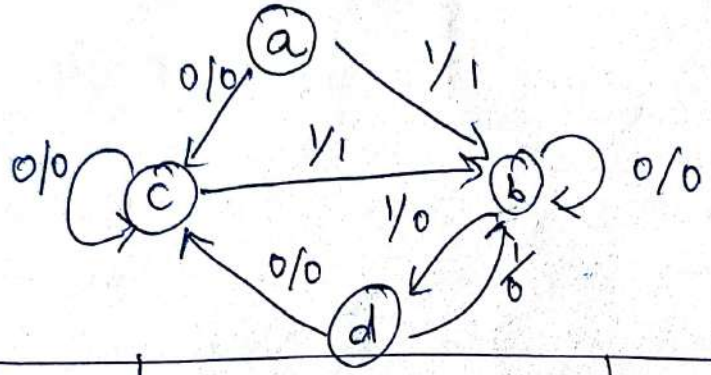
* O/p fn of the present state of flip flops



II) Mealy models :-



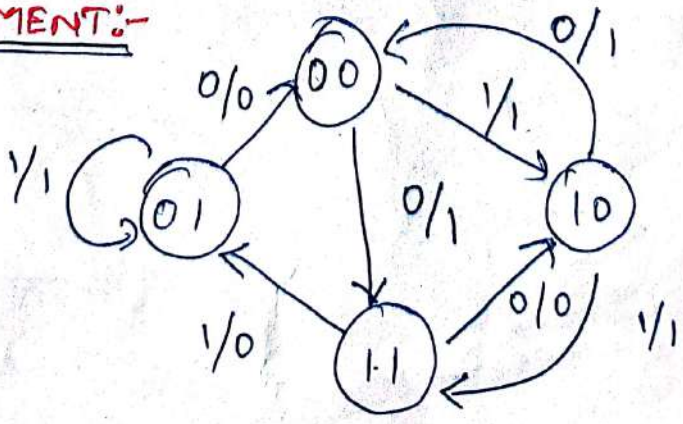
STATE MINIMIZATION :- (State Reduction Techniques)



Present state	N.S		O/p	
	x_2 0	x_2 1	x_2 0	x_2 1
a	c	b	0	1
b	b	d	0	0
c	c	b	0	1
d	c	b	0	0

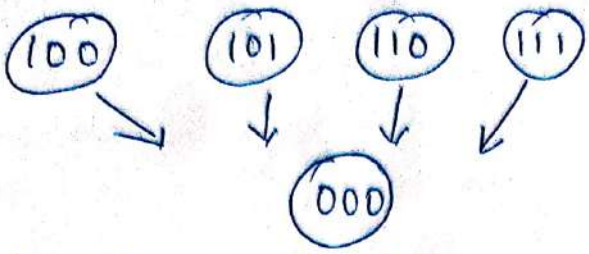
* 'a' and 'c' are same → Redundant state
* Replace 'c' by 'a'.

STATE ASSIGNMENT :-

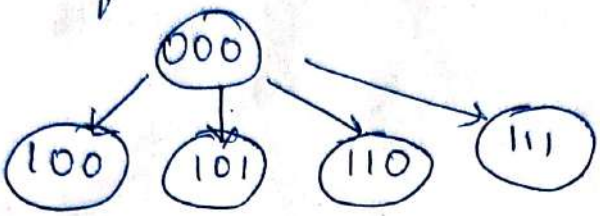


* Two basic Rules:-

R1 :- 4 states whose next states are same



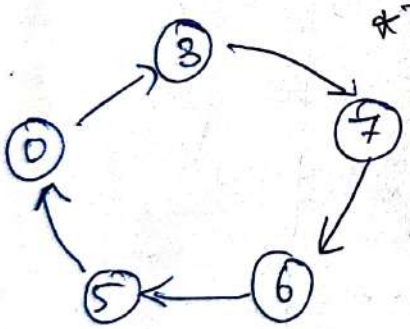
R_2 : 000 has four next state :-



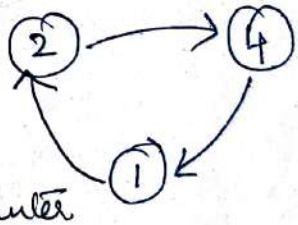
LOCK-OUT CONDITION CIRCUIT IMPLEMENTATION:-

* The next state of some unused state and never arrived at a used state then the counter is said to be lockout conditions.
* Lockout condition called self starting counter

Desired sequence



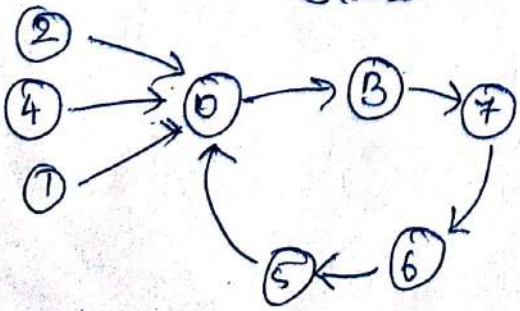
Unused state forming lockout

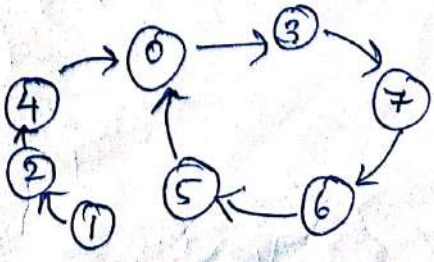


* The counter which never goes in lockout condition is called self starting counter

* The circuit that goes in lockout condition is called bushless circuit.

State diagram For removing lockout

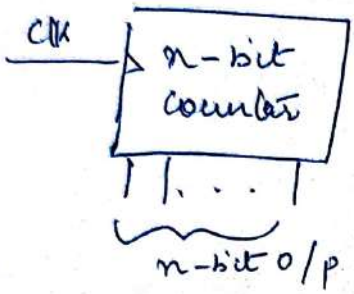




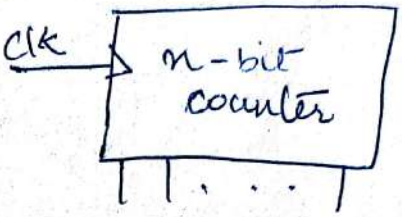
COUNTERS :-

* A group of flip flops connected together to form a register. Registers are capable of counting the no. of clock pulses arriving at its clock input is called counter.

a) +ve edge Triggered n-bit Counter



b) -ve edge Triggered n-bit counter

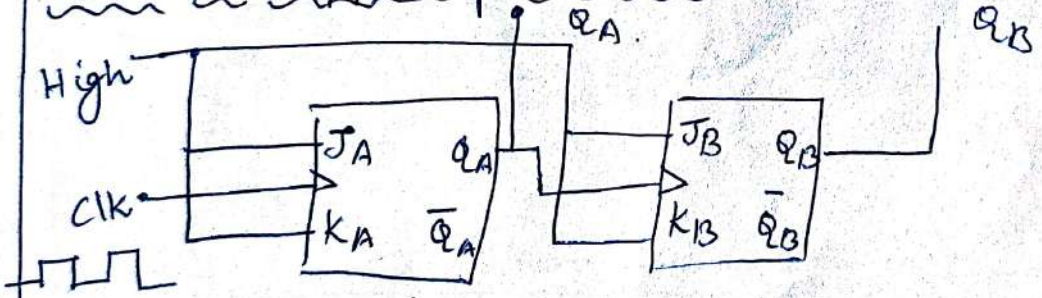


RIPPLE COUNTERS :- (or Asynchronous) :-

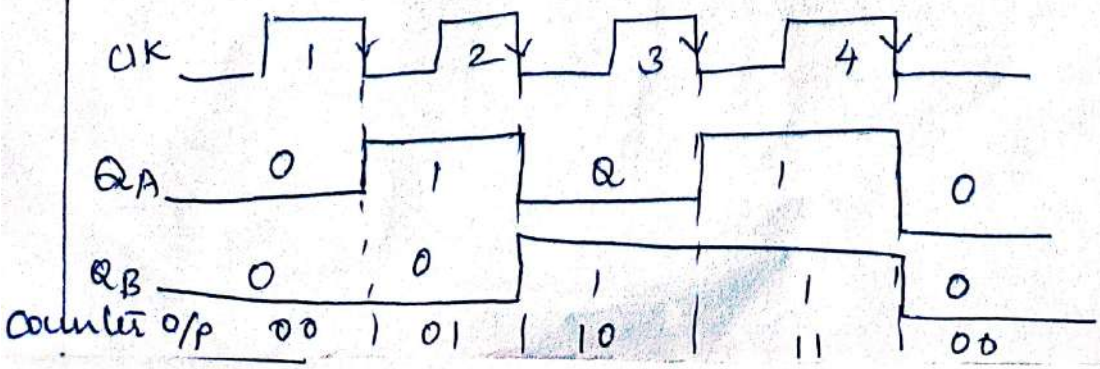
* Binary ripple counter consist of a series connection of complementing FF's with o/p of each FF connected to the clock i/p of next higher order FF.

1) Ripple up counter :-

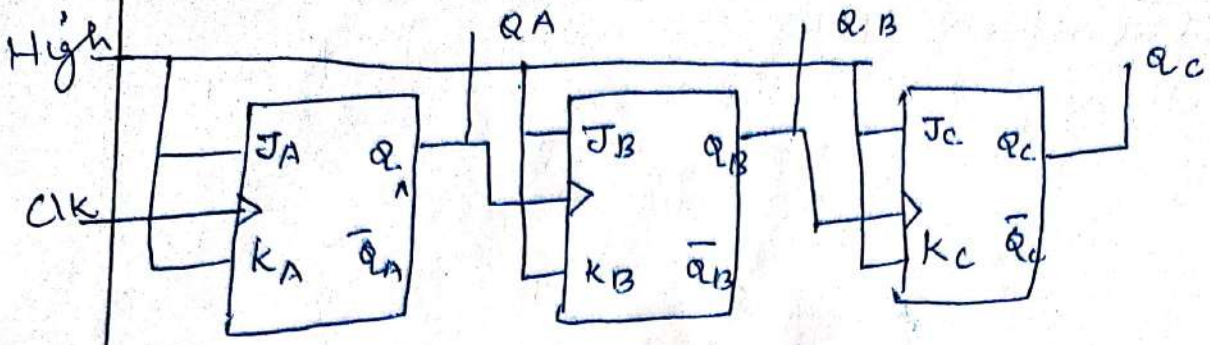
Two Bit Ripple up counter :-



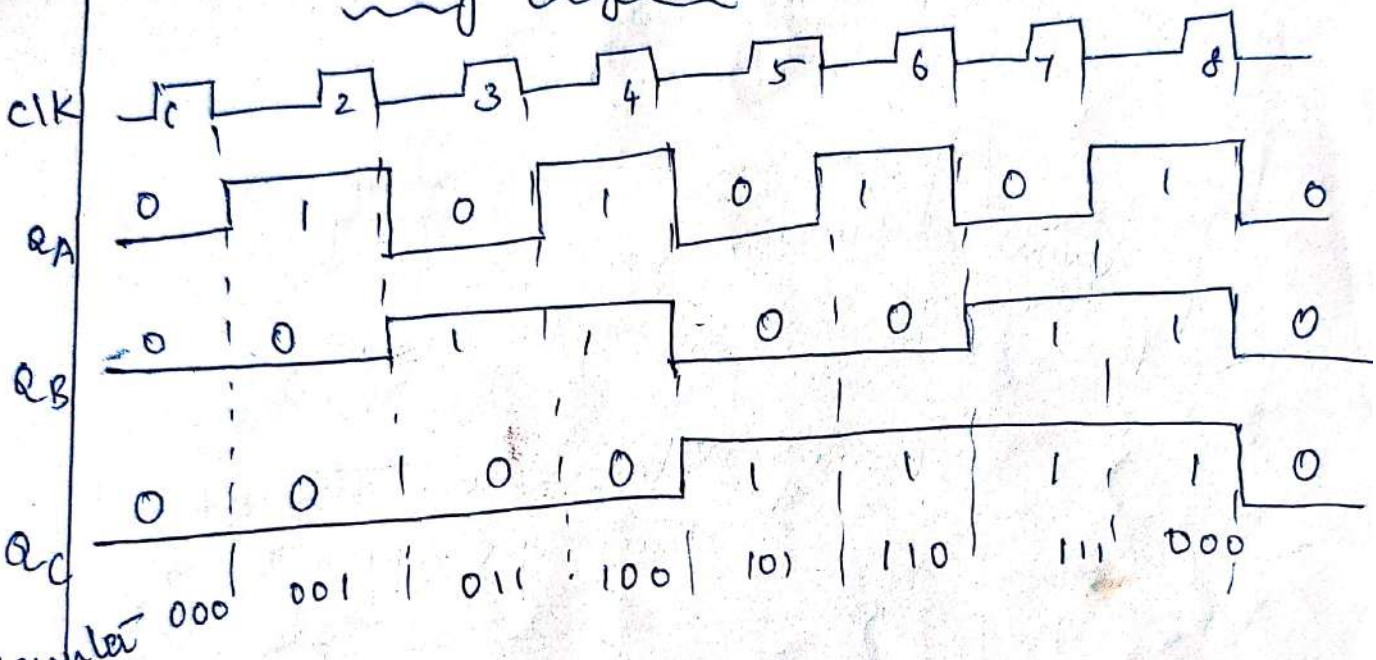
Timing diagram :-



→ Three Bit Ripple counter :-

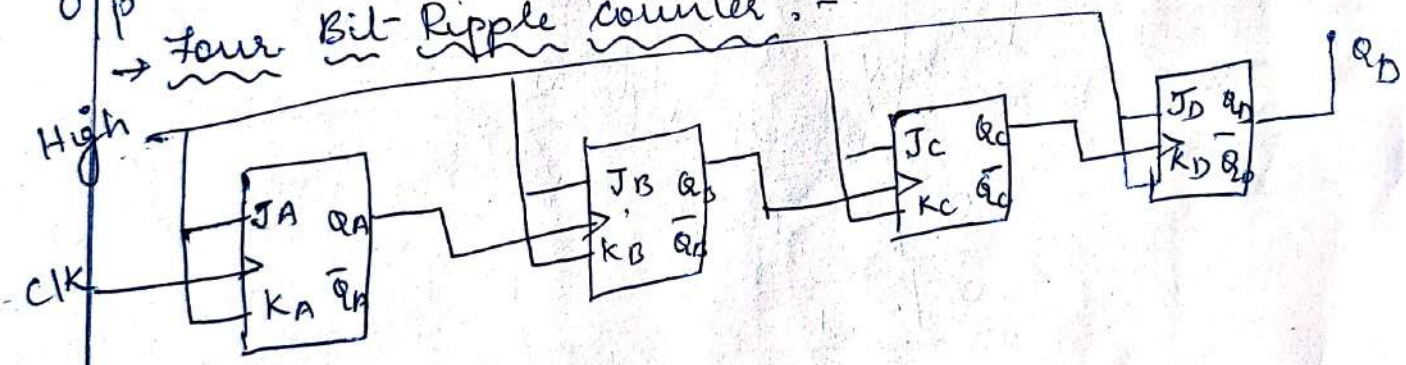


Timing diagram :-



Counter O/P

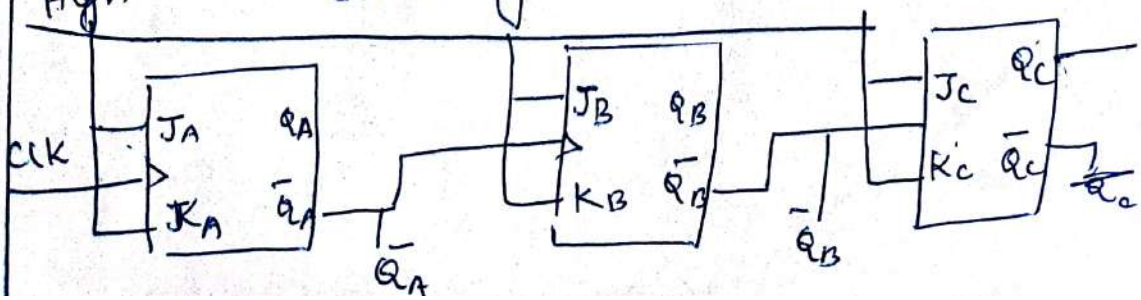
→ Four Bit Ripple counter :-

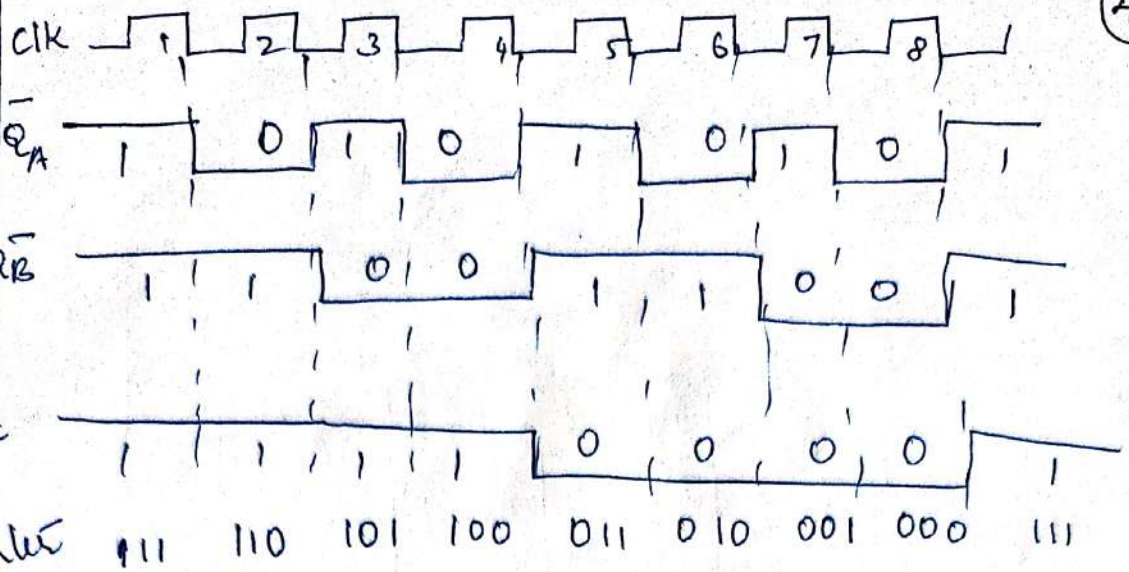


(ii) Asynchronous Down counter :-

* up counter which incremented by one from Zero to maximum count.

3bit synchronous counter.





iii) Asynchronous up/down counter :-

* up/down counter has counting from zero to Maximum count as well as max. count to zero.

* up/$\overline{\text{down}}$ = 0 → count down
  up/down = 1 → count up.

Truth Table :-

	I/p		O/p
up/$\overline{\text{down}}$	Q	$\bar{Q}$	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

Down counter up/$\overline{\text{down}}$ = 0

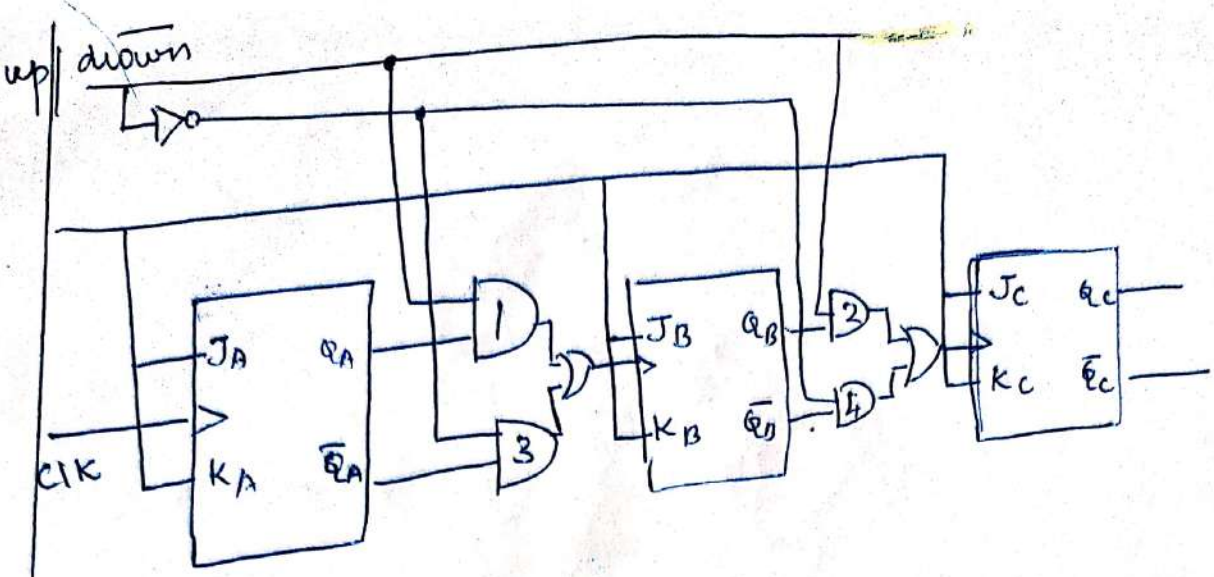
up counter up/$\overline{\text{down}}$ = 1

$Y = \bar{Q}$ for down counter

$Y = Q$ for up counter

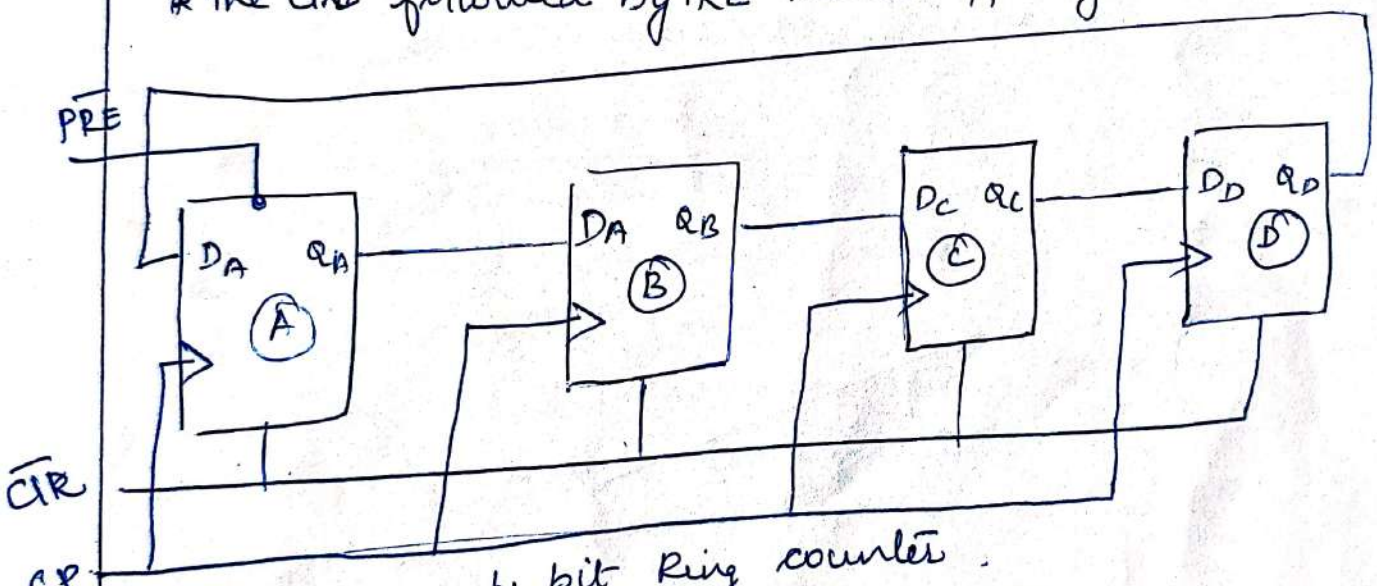
up/$\overline{\text{down}}$ \ $Q\bar{Q}$	00	01	11	10
0	0	1	1	0
1	0	0	1	1

$Y = \overline{\text{up}/\overline{\text{down}}}\,\bar{Q} + \text{up}/\overline{\text{down}}\,Q$

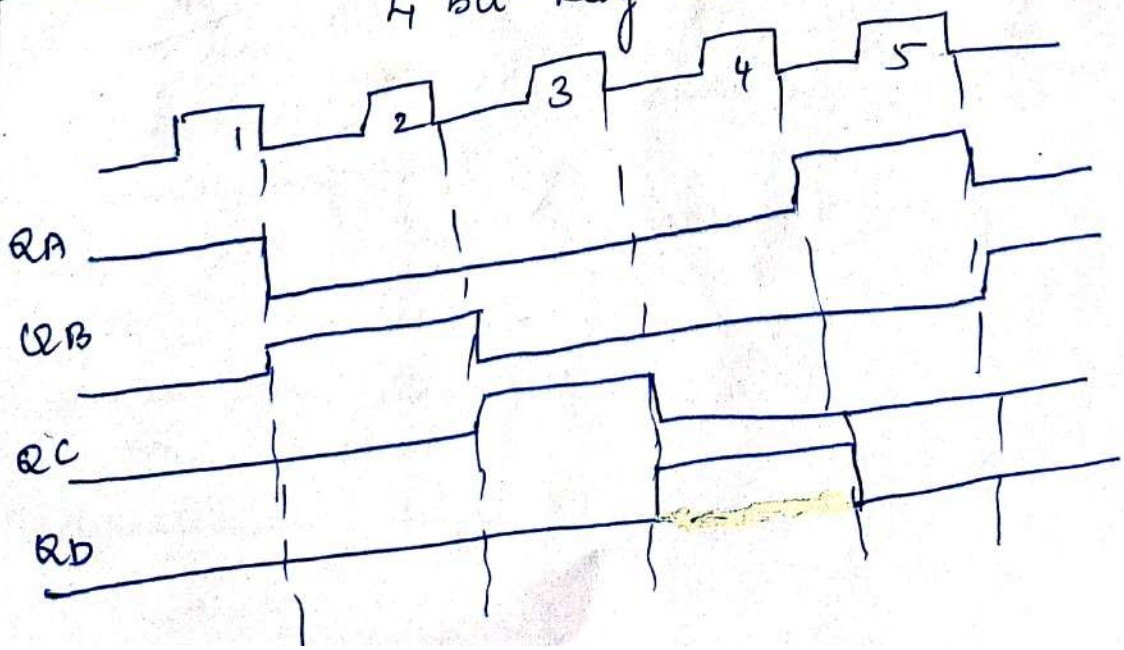


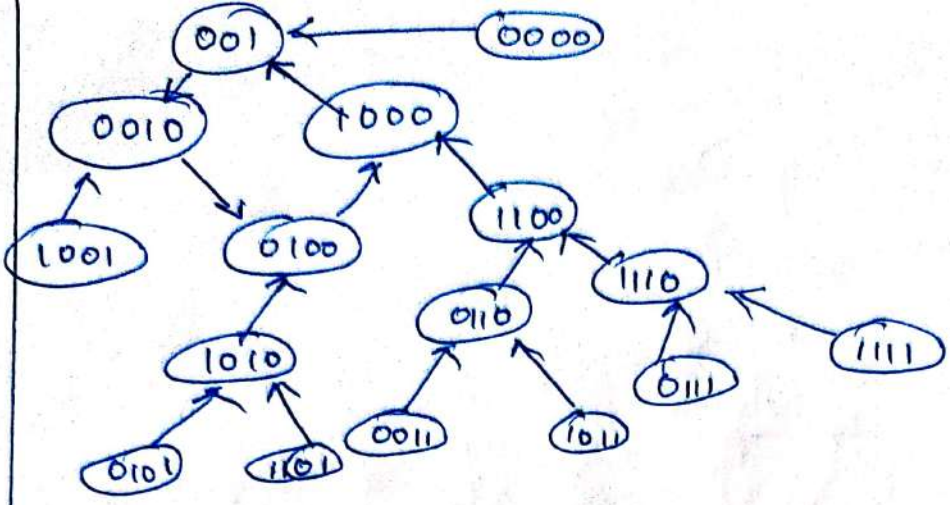
RING COUNTERS :-

* 4 bit Ring counter → Ring counter is circular shift register with only one FF
* Q o/p of each stage connected to D I/p of next stage.
* The $\overline{CLR}$ followed by $\overline{PRE}$ makes o/p stage 1.



4 bit Ring counter.





SHIFT REGISTERS :-

* A Register capable of shifting the binary information held in each cell to its neighbouring in a selection direction is called shift Register.
* Consist of chain of FF's in cascade.
* O/p of one FF connected to next FF.
* FF. receives common clock pulses.

UNIVERSAL SHIFT REGISTERS :-

* A register that can shift data to right and left and has parallel load capabilities is called universal shift Register.

* → clk control to clear the register to 0.
  - → clk i/p synchronize the operations
  - → shift right control → Serial i/p & o/p lines in shift Register.
  - → shift left control → Serial i/p & o/p lines associated with shift left.
  - → ||el load control to enable a ||el Transfer & n i/p lines
  - → n ||el o/p lines.

unidirectional shift Register :-
One direction.

Bidirectional shift Register :-
both direction.

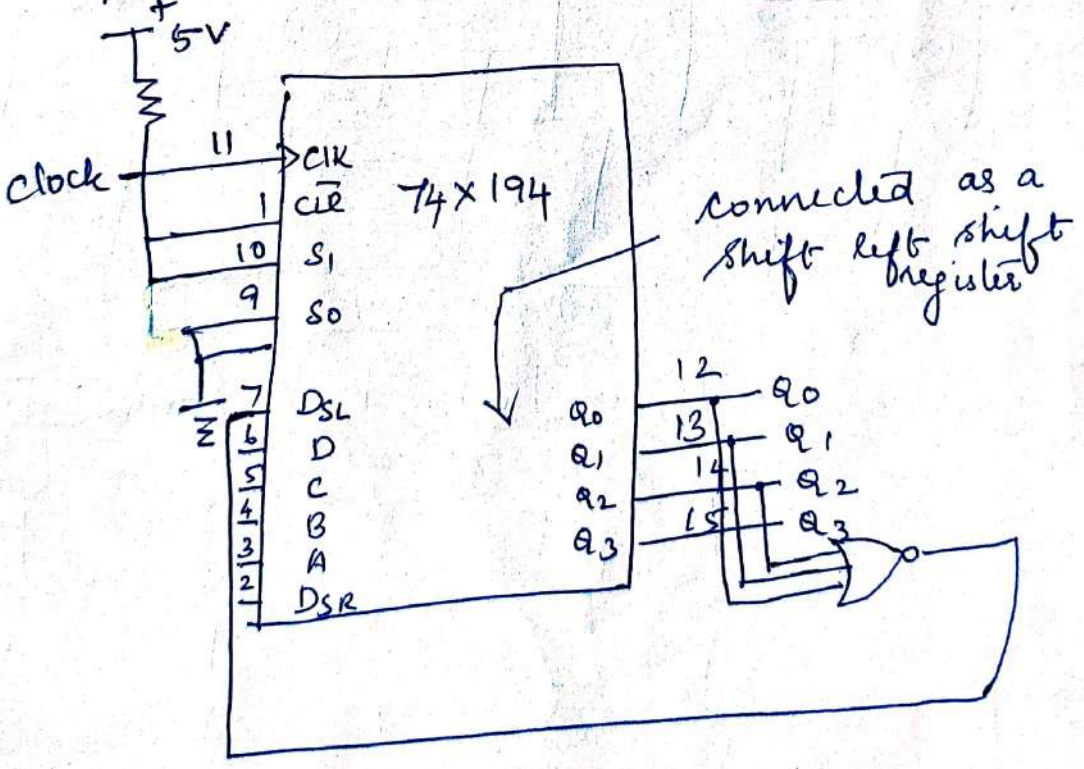
universal shift Register :-

Clock Pulse	Q_A	Q_B	Q_C	Q_D
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1
4	1	0	0	0

* Ring counter can be used for counting the no. of pulses.
* The o/p of N clock pulses this circuit is also referred to as divide by N-counter (or) an N:1 scalar.

Self correcting counters:-

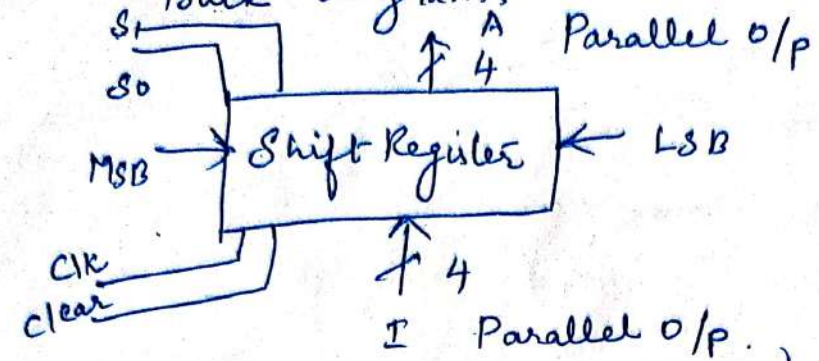
* Ring counters suffer from one major problem - if its single 1 o/p is lost due to temporary hardware problem.



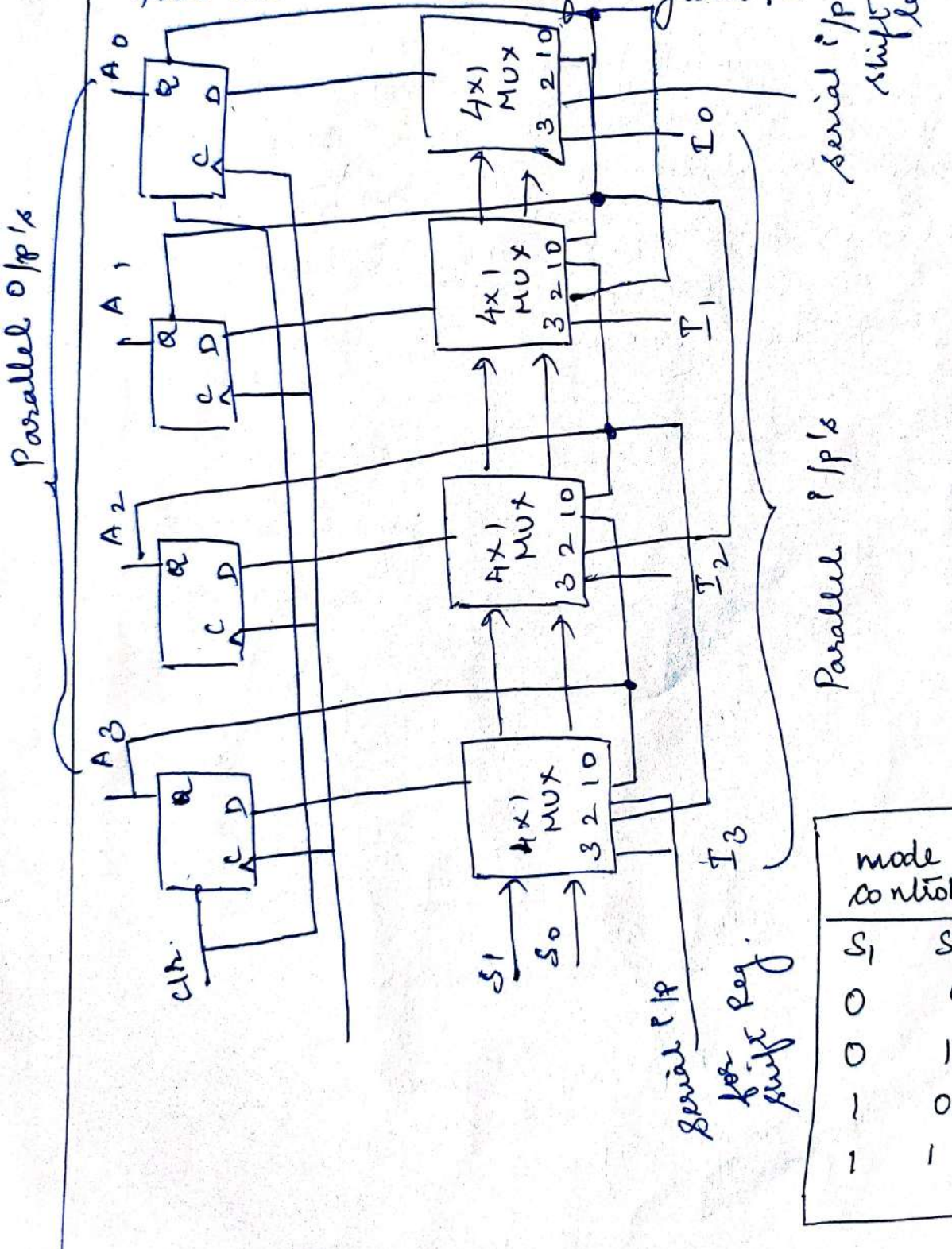
State diagram:-

* Both shifts and IIRL load capabilities

Block diagram:-



Four Bit universal shift Register :-



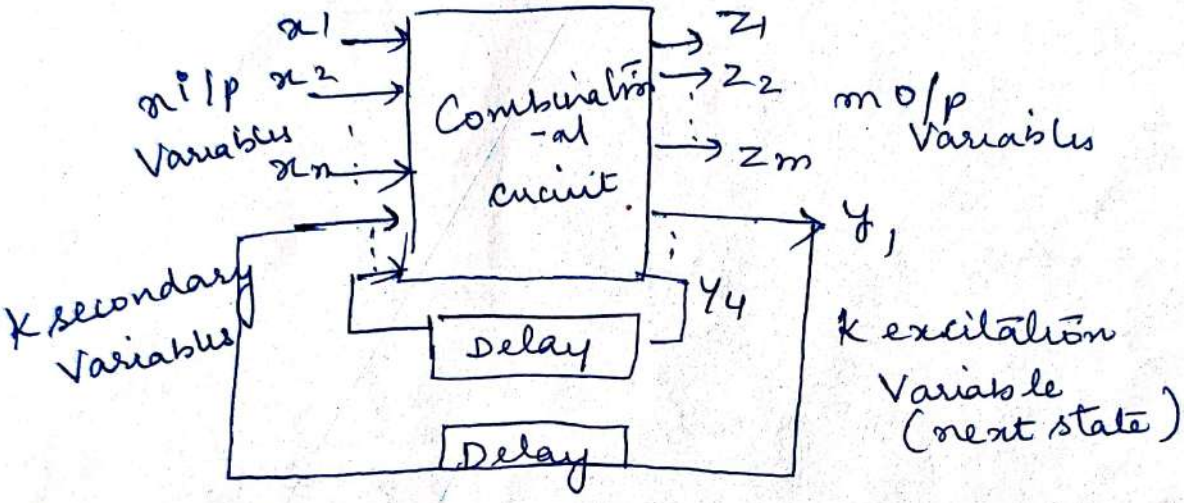
mode control		Reg. operation
S_1	S_0	
0	0	no change
0	1	Shift Reg
1	0	Shift left
1	1	IIRL load

UNIT-IV ASYNCHRONOUS SEQUENTIAL LOGIC CIRCUITS :-

STABLE AND UNSTABLE STATES :-

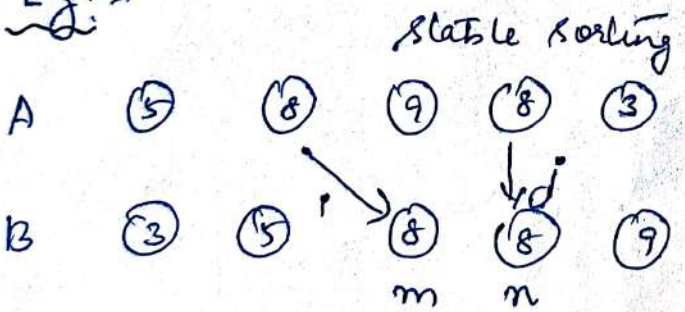
* A sequential circuit is specified by a time sequence of i/p, o/p's and internal states.

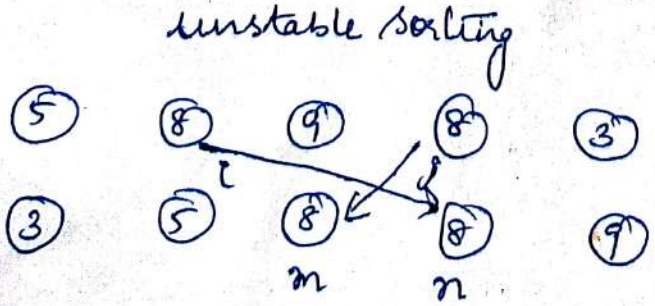
* Asynchronous sequential circuits do not use clock pulses



* System is said to be stable, if its o/p is under control. Otherwise, it is said to be unstable.

Eg:-





CYCLES AND RACES :-

* A cycle occurs when an asynchronous circuit makes a transition through a series of unstable states.

Eg :-

y_1y_2 \ x	0	1
00	(00)	01
01		11
11		10
10		(10)

$00 \rightarrow 01 \rightarrow 11 \rightarrow 10$

Races :-

* Two or more binary state variable change their value.

Eg :-

y_1y_2 \ x	0	1
00	(00)	11
01		11
11		(11)
10		11

Possible Transitions

$00 \rightarrow 11$

$00 \rightarrow 01$ (y_2 faster) $\rightarrow 11$

$00 \rightarrow 10$ (y_1 faster) $\rightarrow 11$

Eg for non-critical Races.

y_1y_2 \ x	0	1
00	(00)	11
01		(01)
11		(11)
10		(10)

Possible transition

$00 \rightarrow 11$

$00 \rightarrow 01$

$00 \rightarrow 10$

Free state assignment, two commonly used techniques :-

i) Shared row state assignment.

ii) One hot state assignment.

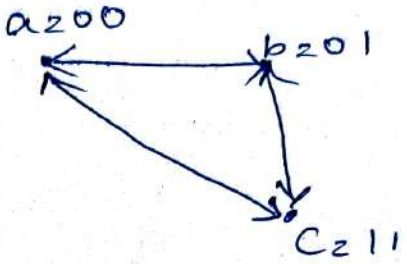
i) Shared Row state Assignment.

* Two binary Values are said to be adjacent if they differ in only one Variable.
* 00 and 01 are adjacent to each other.

3 row Flow Table.

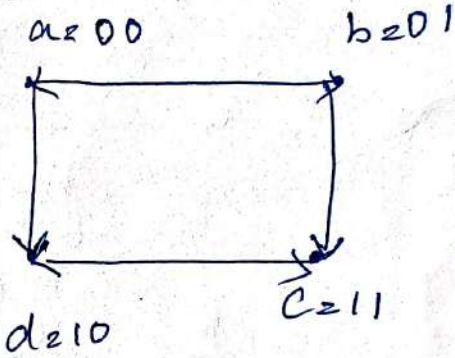
	00	01	11	10
a	(a)	b	c	(c)
b	a	(b)	(b)	(a)
c	a	(c)	(c)	(c)

Transition diagram



modified table.

	00	01	11	10
a	a	b	d	a
b	a	b	b	c
c	d	c	c	c
d	a	–	c	



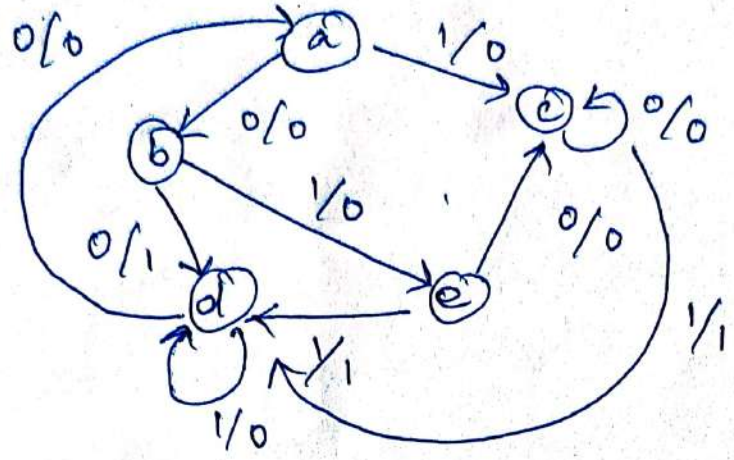
ii) One hot state Assignment :-

* An another method for finding a race free state assignment
* only one Variable is active (or) 'hot' for each row in original flow table.

S.V				State	I/p $x_1 x_2$			
S_4	S_3	S_2	S_1		00	01	11	10
0	0	0	1	A	(A)	B	C	C
0	0	1	0	B	A	(B)	C	D
0	1	0	0	C	A	B	(C)	(C)
1	0	0	0	D	(D)	B	C	(C)

STATE REDUCTION:-

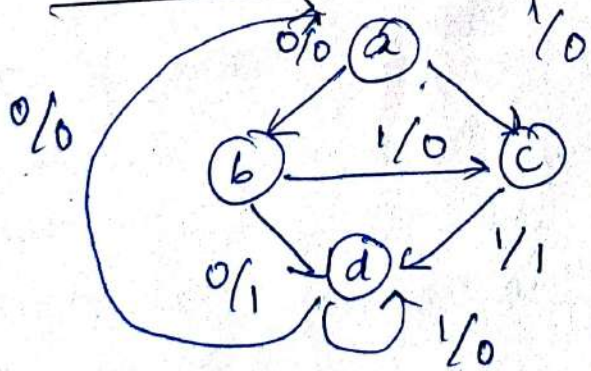
* This Technique basically avoids the introduction of redundant states.
* no. of required F-F's and logic gates.



State Table:-

Present state	Next state		O/p	
	$x=0$	$x=1$	$x=0$	$x=1$
a	b	c	0	0
b	d	e	1	0
c	c	d	0	1
d	a	d	0	0
e	c	d	0	1

Reduced state diagram:-



Reduced State

P.S	N.S		O/p	
	$x=0$	$x=1$	$x=0$	$x=1$
a	b	c	0	0
b	d	c	1	0
c	c	d	0	1
d	a	d	0	0

RACE FREE ASSIGNMENTS :-

* Race free state assignment, two commonly used Technique.
  i) Shared row state assignment
  ii) One hot state assignment.

i) Shared row state assignment :-

* Races can be avoided by making a proper binary assignment to the state Variable.

ii) One hot state assignment :-

* A Race free state assignment another method
* Only one Variable is active or 'hot' for each row in original flow table.

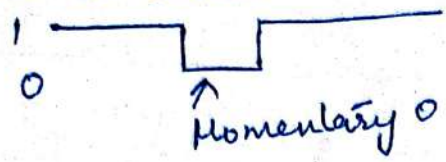
HAZARDS :-

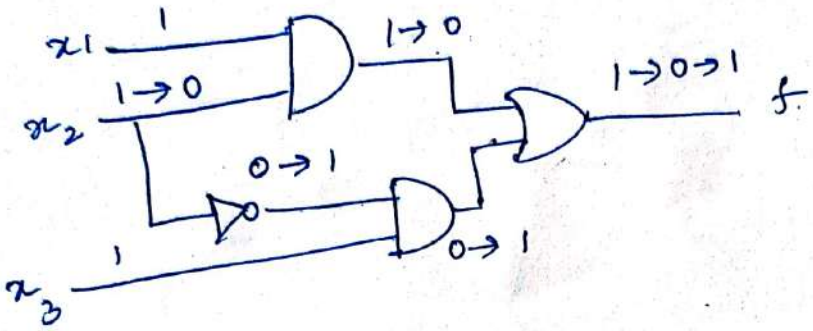
* A Hazard is the potential or actual malfunction of a logic network during the transition between two i/p states as result of single variable change.
* Two Types of Hazards.
  i) Static Hazards
  ii) Dynamic Hazards.

i) Static Hazards :-

* In logic networks is a transient change of an o/p value that is supposed to remain fixed during the transition b/w 2 i/p states differing in Value of one Variables.
  i) Static 1 hazards
  ii) Static 0 hazards

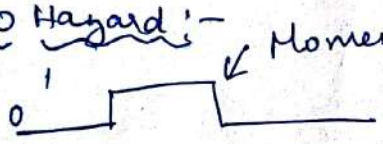
i) Static -1 Hazard :-

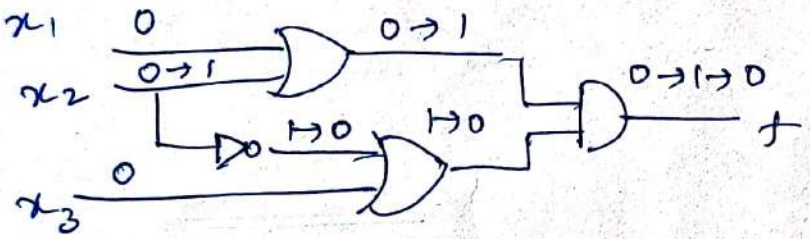




* Transition b/w i/p states $x_1x_2x_3 = 111$, $x_1x_2x_3 = 101$

ii) Static -0 Hazard :-





* i/p states $x_1x_2x_3 = 000$, $x_1x_2x_3 = 010$.

Elimination Static Hazards :-

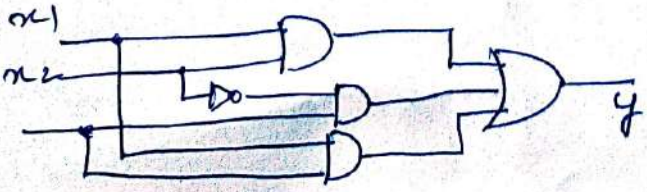
Static -1 Hazard :-

* Removed by covering adjacent cells with redundant grouping.

$$y = x_1x_2 + \bar{x}_2x_3$$

k-map :-

x_3 \ x_1x_2	00	01	11	10
0	0	0	1	0
1	1	0	1	1



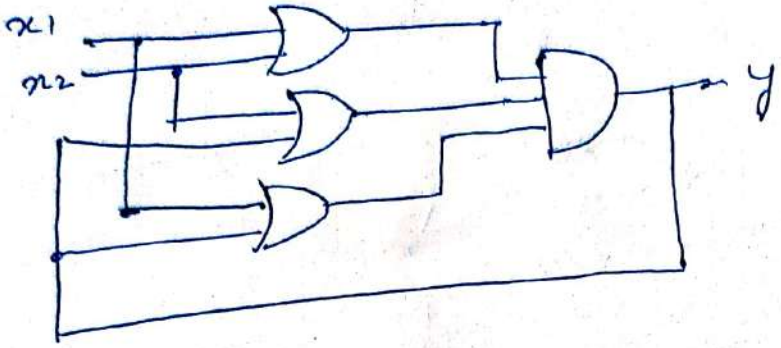
Static 1 hazard.

Elimination of static -0 Hazard :-

$$y = x_1\bar{x}_2 + x_1 y + \bar{x}_2 y$$

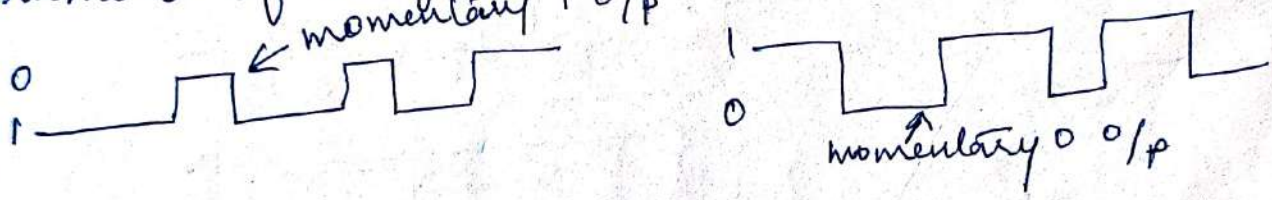
K-map :-

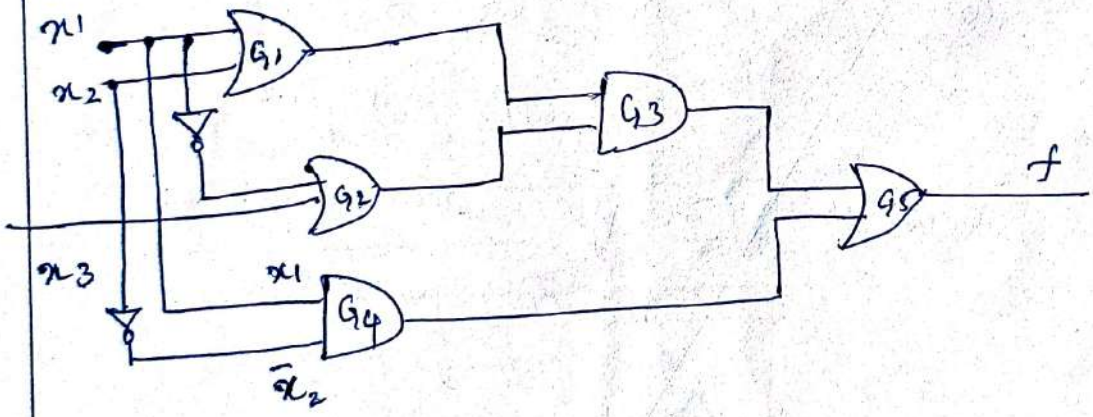
$$y = \bar{\bar{y}} = (y + \bar{x}_2)(x_1 + x_2)(y + x_1)$$



ii) Dynamic Hazards :-

* Dynamic hazard is defined as a transient change occuring three or more times at an O/p terminal of a logic network





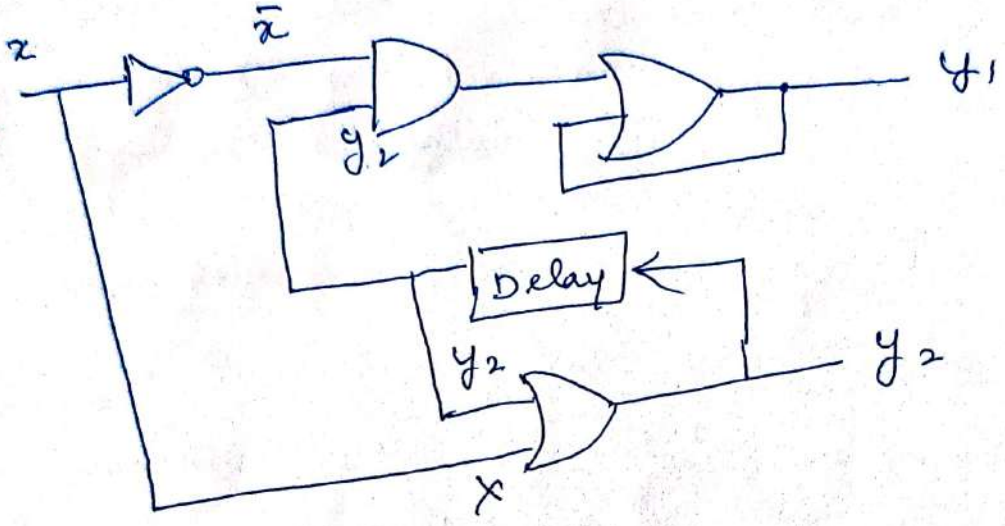
* Change of x_1 from 0 to 1, N/W O/p will appear as $0 \to 1 \to 0 \to 1$

ESSENTIAL HAZARDS :-

* After avoiding critical race and eliminating static and dynamic hazards, a fundamental mode

asynchronous sequential N/w may still malfunction. These N/w are subject to still another type of hazard called essential hazard.

* Always be eliminated in a realization by the insertion of sufficient delay.



Flow table :-

P.S	N.S	
	I/p x	
	0	1
A	(A)	B
B	C	(B)
C	(C)	(C)

Transition Table :-

P.S. y_1y_2	N.S I/p x	
	0	1
A → 00	(00)	01
B → 01	11	(01)
C → 11	(11)	(11)

FUNDAMENTAL AND PULSE MODE SEQUENTIAL CIRCUITS :-

Design of Pulse mode circuit :-

* In P.M.C, no clock is present, I/p's occur on only one line at a time and only uncompleted

* Steps to design of pulse mode asynchronous

1. Define states & draw a state diagram and (or state table.
2. Minimize the S.T.
3. Do state assignment.
4. Choose the type of latch or FF to be used and determine excitation eqn's.
5. Construct excitation.
6. Determine O/p eqn.
7. Draw logic diagram

Analysis of Fundamental mode sequential circuits :-

1. Determine the next-sec. state and O/p eqns.
2. Construct the state table
3. Construct Transition Table
4. Construct O/p map.

Design of Fundamental mode sequential circuits :-

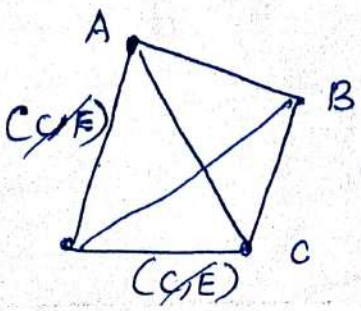
1. Derivation of Primitive flow table :-

* Asynchronous sequential circuit is same as that of state table in synchronous sequential circuits.

* Logic propagation delay and cause the state 'flow' from one to another.

2. Reduction of Primitive flow table :-

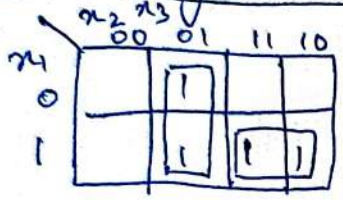
E.



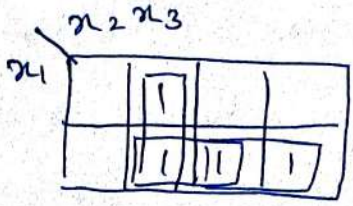
DESIGN OF HAZARD FREE CIRCUITS:-

* The unwanted switching transients (glitches) that may appear at the o/p of a circuit are called Hazards.
* Two Types :- i) Static Hazards
  ii) Dynamic Hazards.

Eliminating a Hazard:-

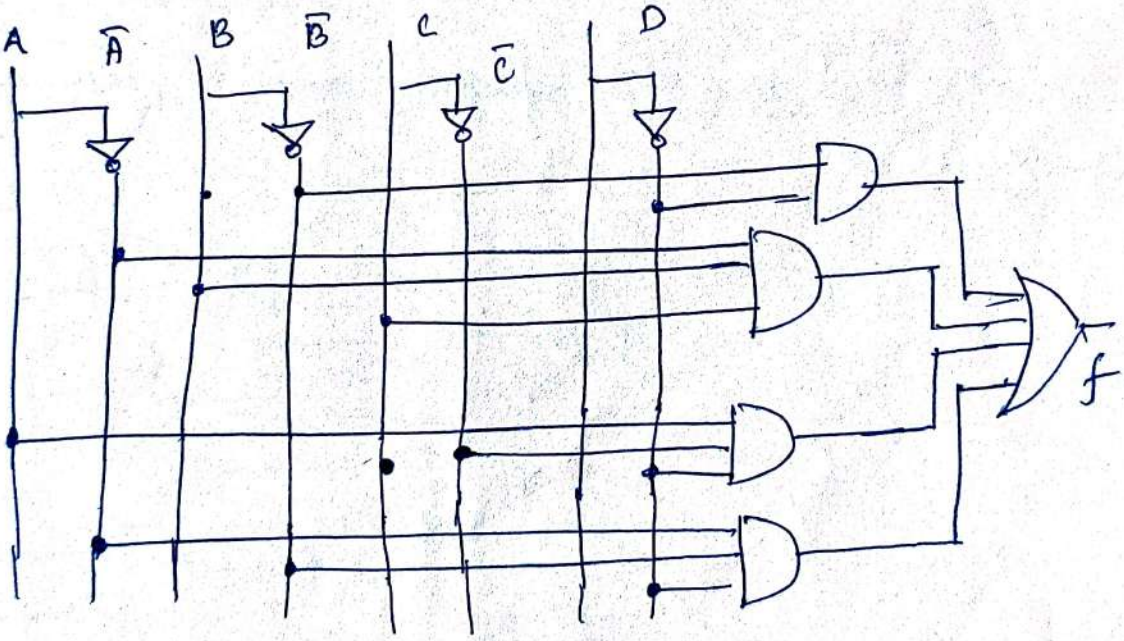


$y = x_1x_2 + \bar{x}_2x_3$

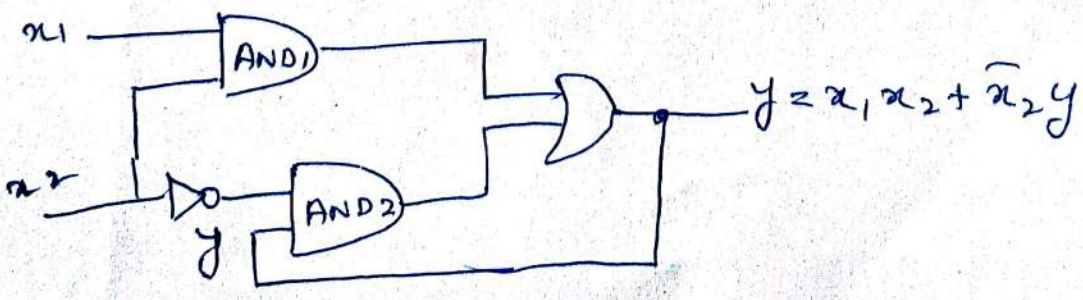


$y = x_1x_2 + \bar{x}_2x_3 + x_1x_3$

* Hazard exists because of change in i/p results in Product term.



Hazards in Sequential circuits:-



Transition Table:-

y_0	x_0x_2 00	01	11	10
0	(0)	(0)	1	0
1	(1)	0	(1)	(1)

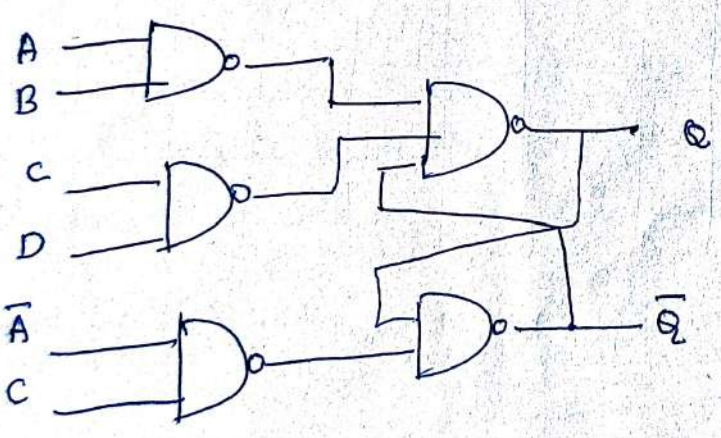
* In synchronous sequential circuits, the hazards due to combinational circuits.

Essential Hazards:-

Another type of hazard in asynchronous sequential circuits called essential hazards.

Eliminating Essential Hazards:-

* 0 signal applied to S or R i/p's of NOR latch have no effect on state.
* Momentary signal 1 applied to S and R of NAND latch.

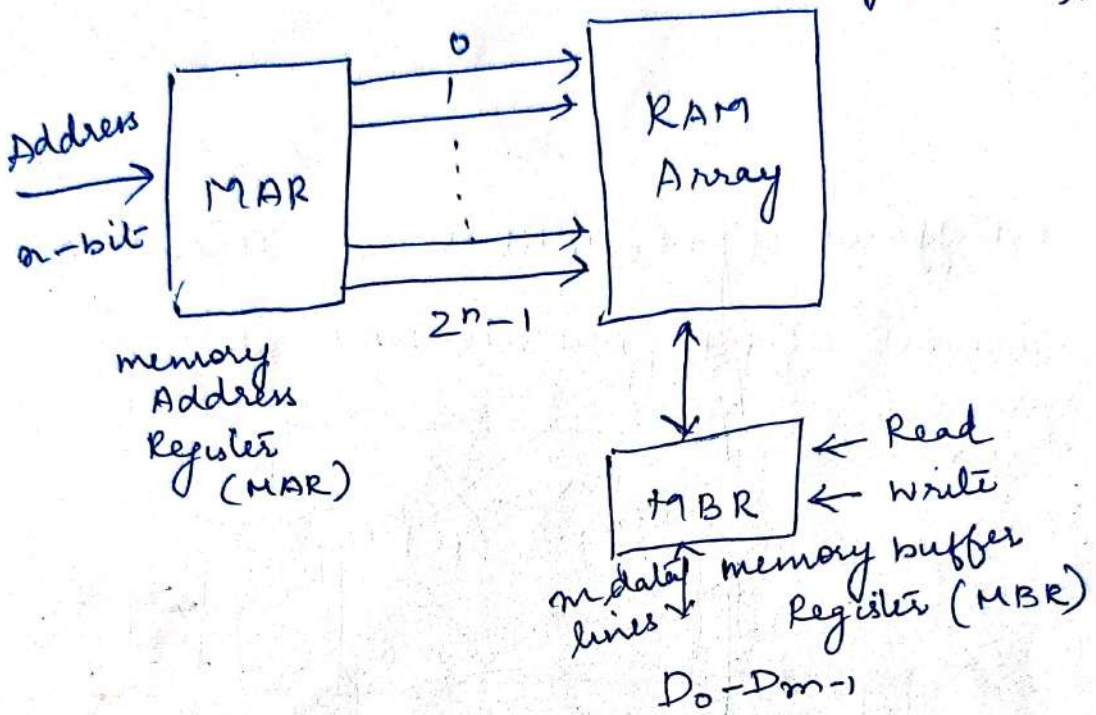


UNIT-V MEMORY AND PROGRAMMABLE LOGIC.

RAM AND ROM :-

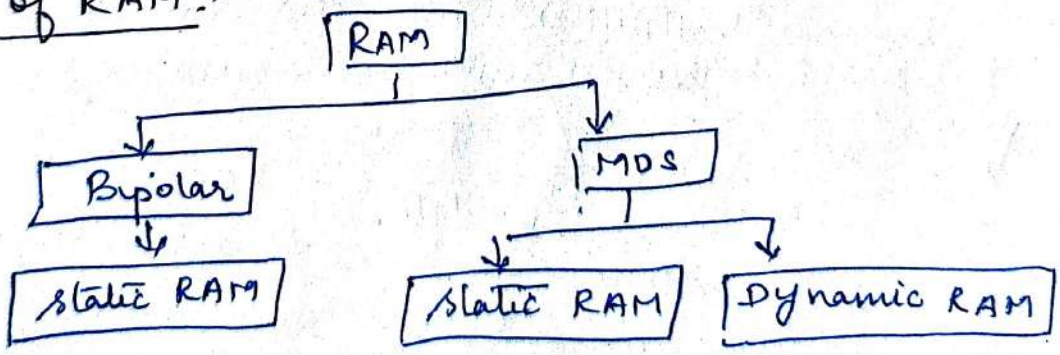
RAM organization :-

* RAM is a Volatile memory which both read and write operations.
* It is also called Read Write memory (RWM).



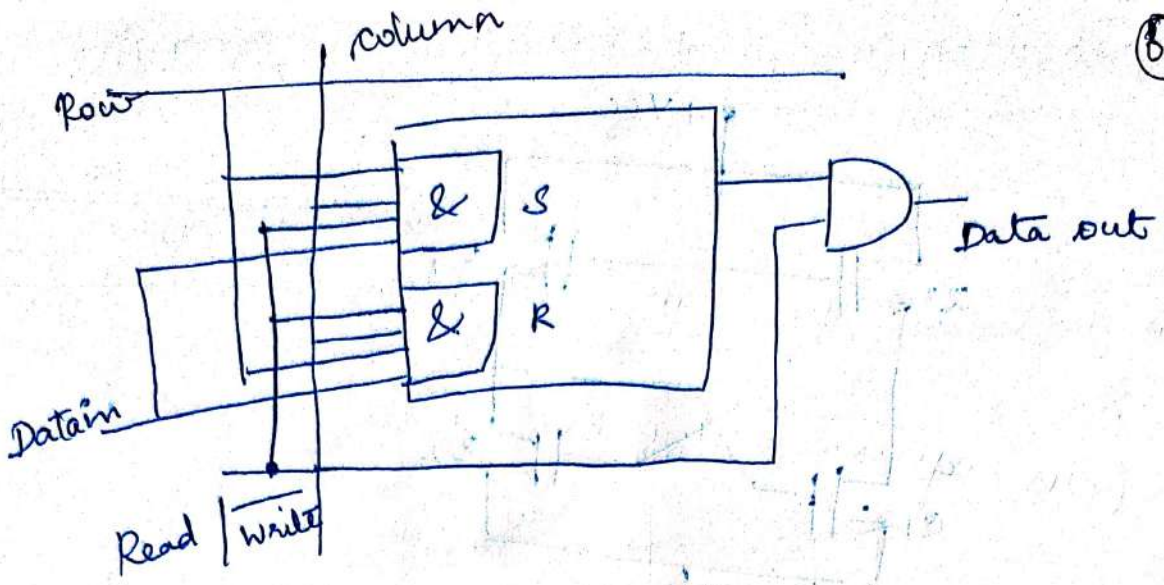
* When a Read is asserted, the data from the memory location, selected by the n-bit address and it is placed in Memory Buffer Register (MBR).

Types of RAM :-



static RAM cell :-

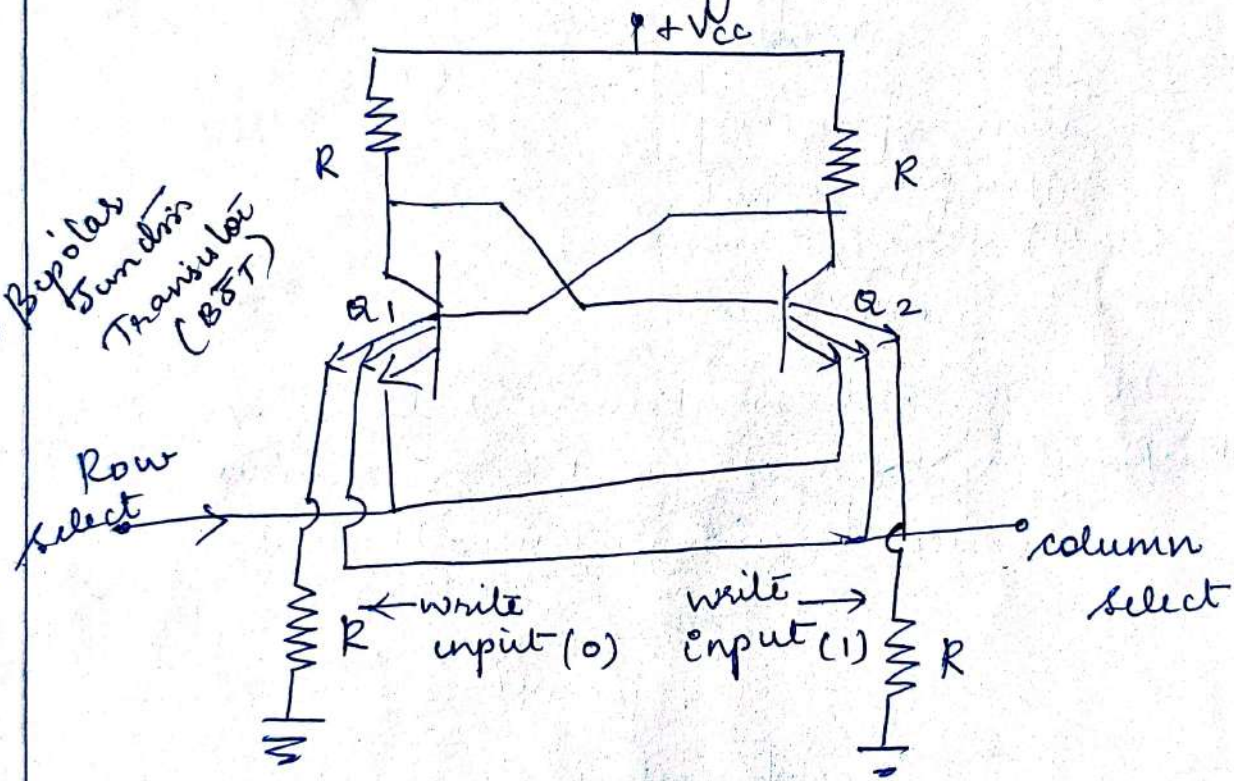
* Row and column lines, the flip flop for a 1 and



Resetting the FF for 0, when READ / $\overline{\text{WRITE}}$ line is low

* READ / $\overline{\text{WRITE}}$ line is HIGH, FF is unaffected.

Static RAM cell using Bipolar Transistor.



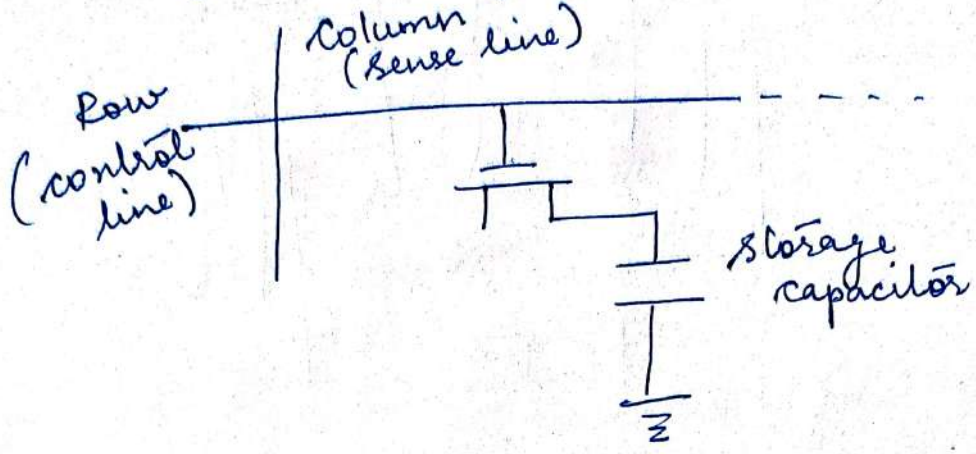
* BJT Q_1 and Q_2.
* Row and column select lines is HIGH → To select.
  Row and column select lines is LOW → Disabled Memory.

MOS Static RAM cell :-

* Q_1 and Q_2 act as switch, Q_3 and Q_4 as active load resistors.

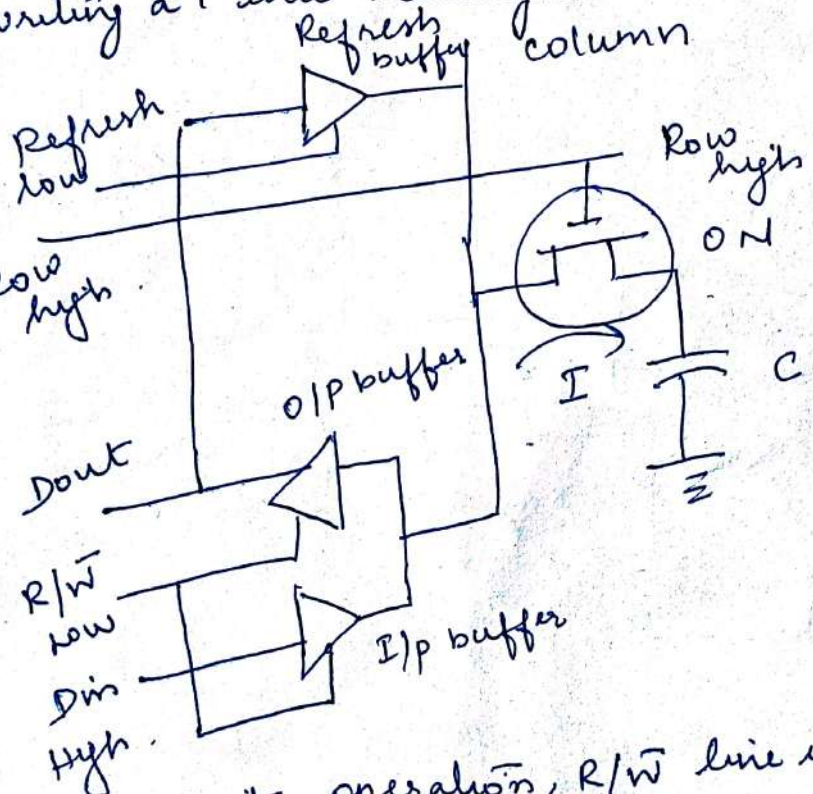
2. Dynamic RAM (DRAM):-

* DRAM is lowest cost, high density random access memory available.
* Memory sizes ranging from 16 to 256 mega bytes.



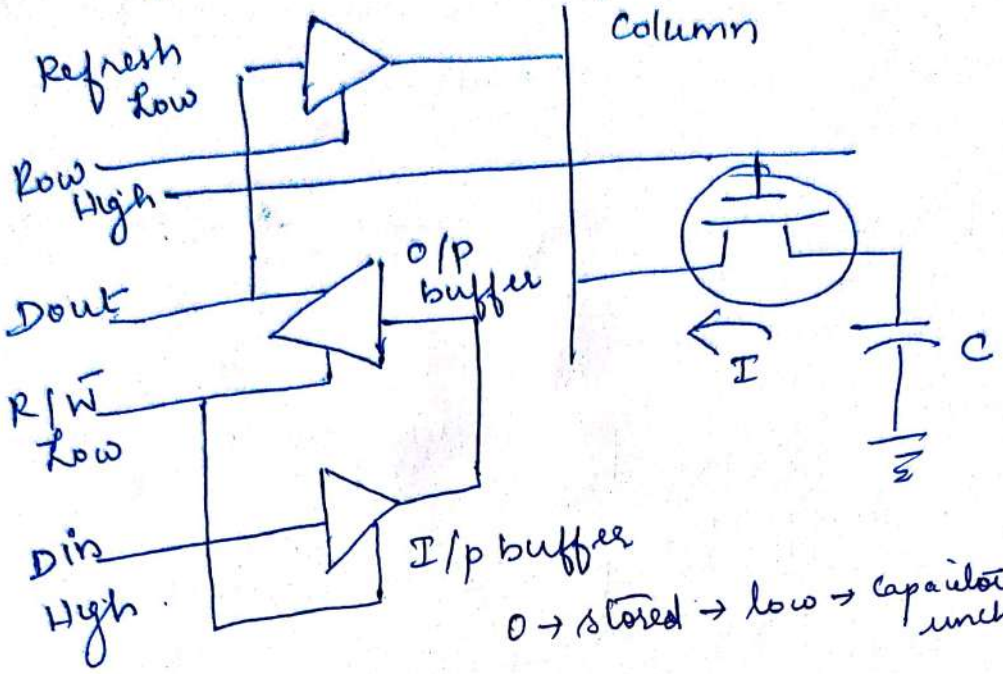
write operation:-

a) writing a 1 into memory cell:



* To enable write operation, R/W line is made low enables input buffer and disables O/P buffers.
* To write 1 into cell, Din lines is HIGH, the transistor is turned ON by a HIGH on Row line.
* When 0 is stored, LOW is applied to the Din line.
* Storing the charge (either 1 or 0) on capacitor.

b) writing a 0 into memory cell.

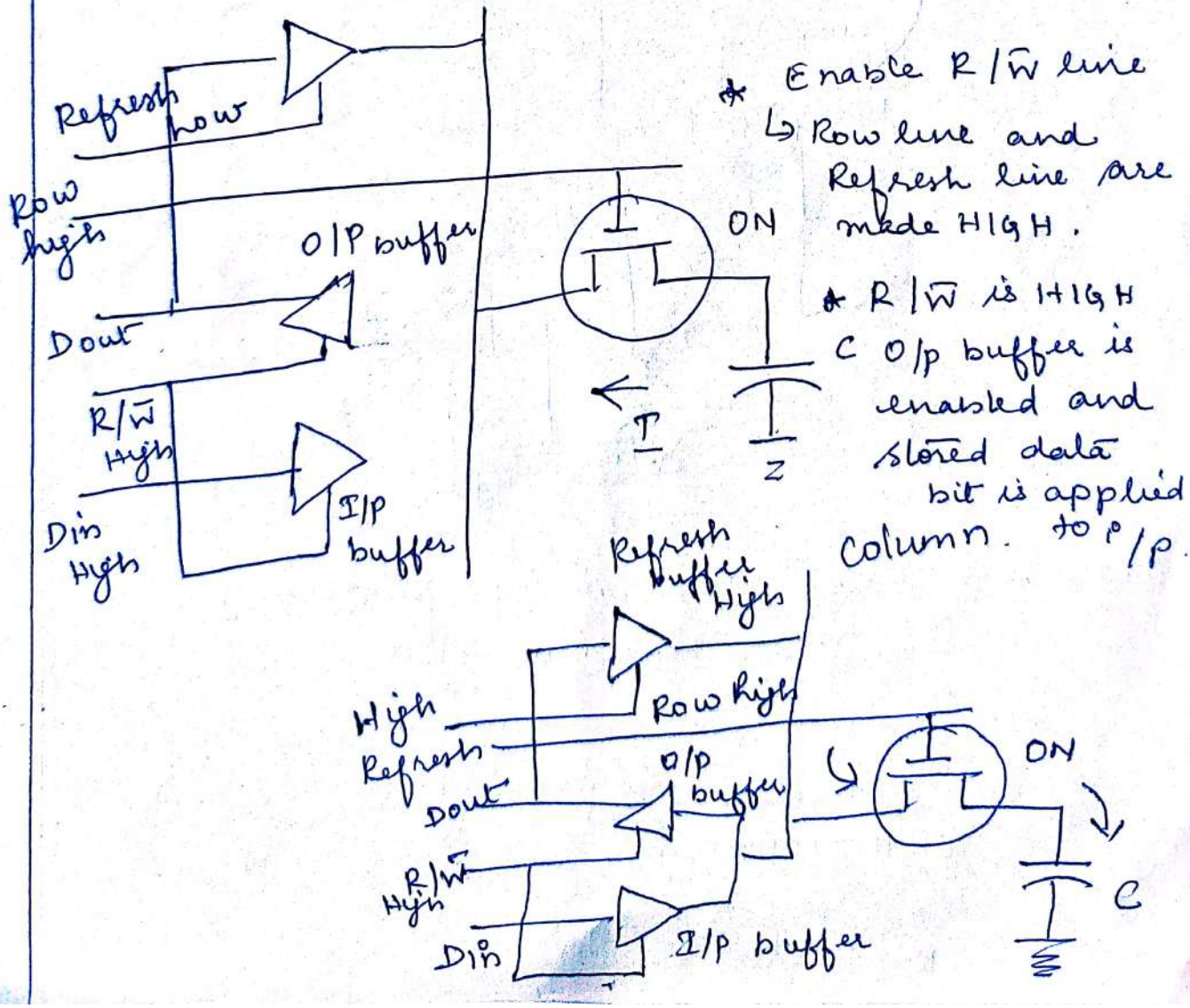


0 → stored → low → capacitor uncharged

Read Operation :-

Reading a 1 from memory

Refreshing a stored 1



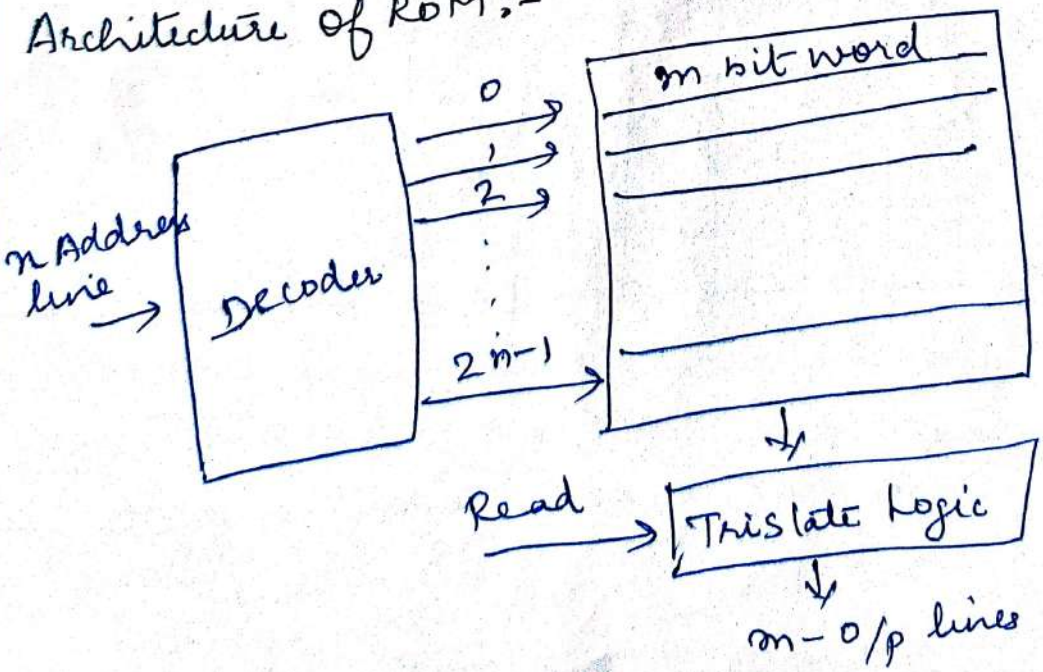
Advantages of RAM :-

* Fast operating speed → Time can be low as 15ns
* non - destructive Read out → RAM does not affect content stored.
* Low power dissipation :- typically less than 1mW for static and 0.5mW as Dynamic RAM.
* Economy :- MOS Memories are more economical than magnetic core
* Compatibility : self compatible..

ROM :-

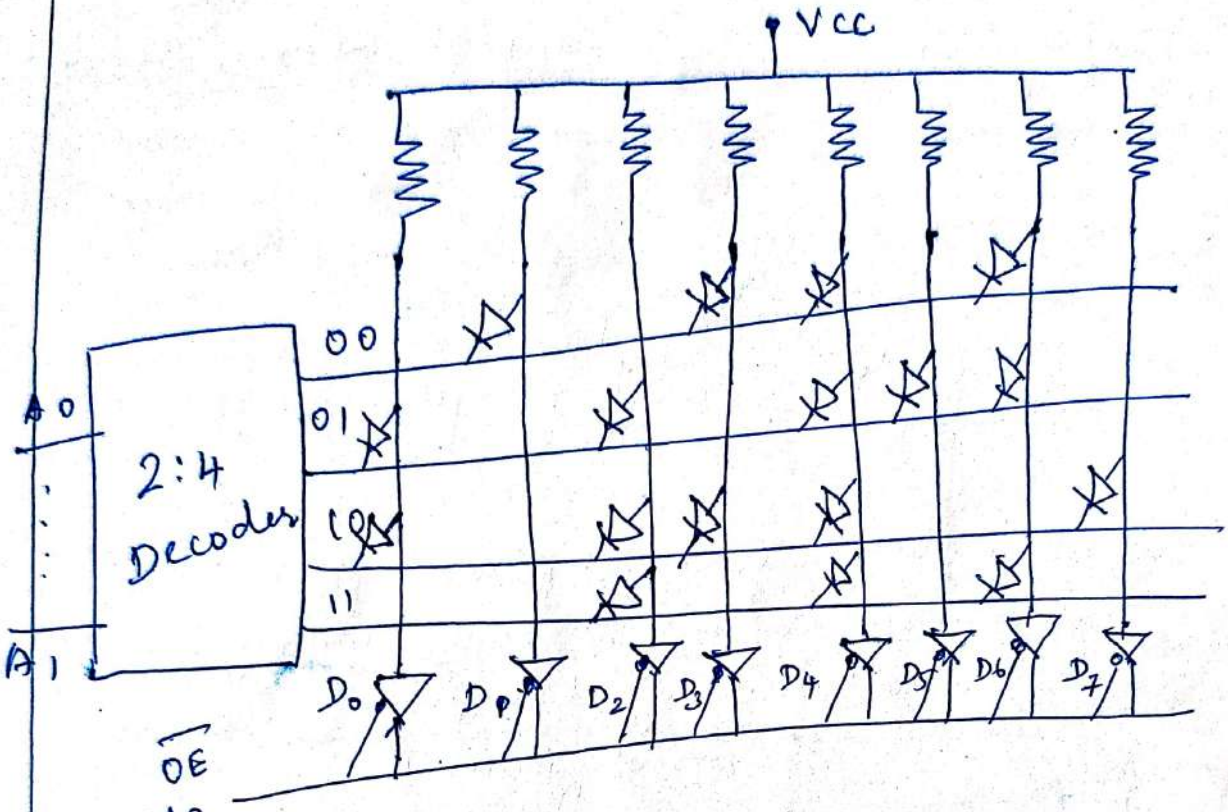
* Read only Memory (ROM) is a semiconductor memory device used to store the information.
* ROM is Programmed for a particular purpose during manufacturing process.
* ROM is a combinational logic circuit and includes both decoder and OR gate with single IC Package.

Architecture of ROM :-



* n address lines and m o/p lines.
* no. of words 2^n and no. of bits m.

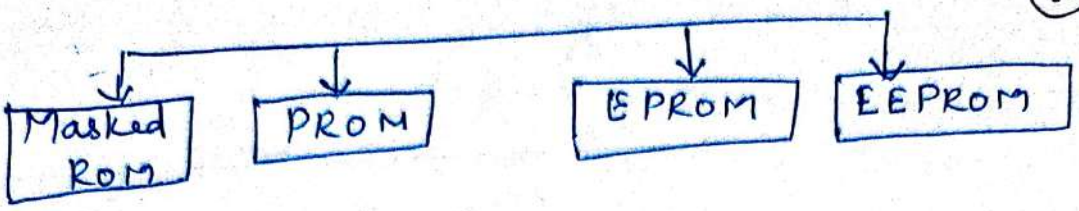
Simple four Byte diode ROM:-



Address in binary	Binary Data								Data in hex
	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	
0 0	1	0	1	0	0	1	0	1	A5
0 1	0	1	0	1	0	0	0	1	51
1 0	0	1	0	0	0	1	1	0	46
1 1	1	1	0	1	0	1	0	1	D5

* ROM may be programmed into two different ways. The 1st is called MASK programming is done by the manufacturer during last fabrication Process. The second type is Programmable Read only memory (PROM).

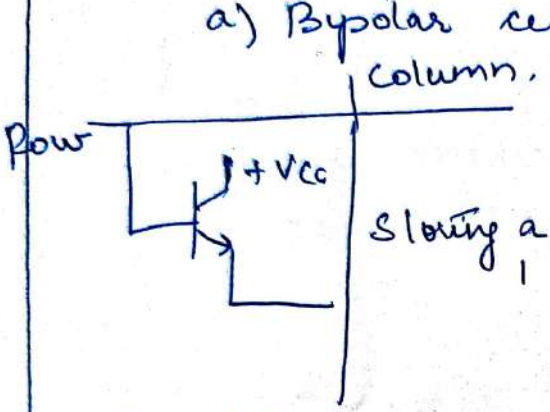
ROM

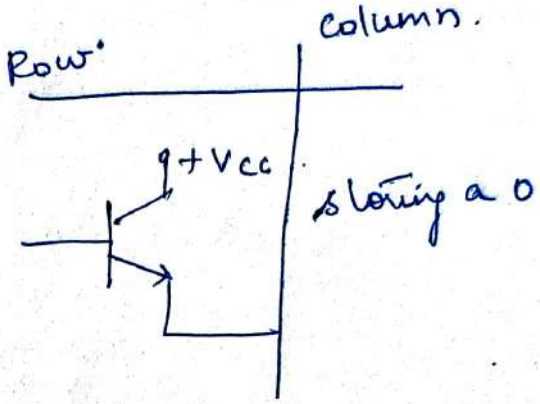


Masked ROM :-

* Permanently Programmed during the manufacturing Process.

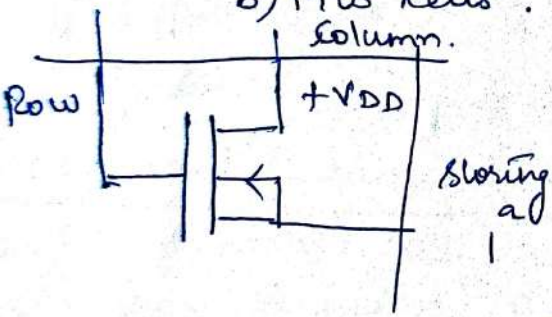
a) Bypolar cell.

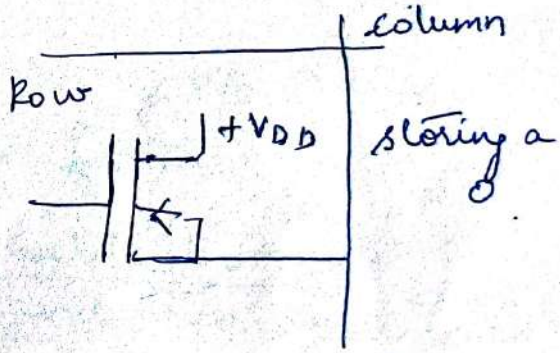




* Row line turn ON → HIGH
Column line is low (0) when ROW is addressed

b) MOS cells.





* Corresponding mask for paths to produce 1's and 0's according to truth table. This process is called MASK Programming.

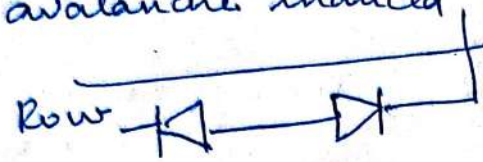
PROM - Programmable Read only memory :-

* PROM can be programmed electrically by the user but cannot be reprogrammed.

→ Fuse Technology used in PROM :-

* 3 basic fuse. 1. metal links.
2. silicon links.
3. p-n Junctions.

* Metal link made of nichrome material. It is either "blown" open or left intact during programming.
* Silicon links are formed by narrow notched strips of polycrystalline silicon.
* p-n Jn's technology also referred as shorted Jn or avalanche induced migration.



* 2 MOS Technologies used for fabrication of Programmable memories.
  1. FAMOS - Floating Gate Avalanche Injection
  2. MAOS.

FAMOS :→ Storage device used in silicon gate MOSFET with no electrical connection to gate.

MAOS PROM :-
* gate dielectric is used for charge storage and provide a reprogramming feature.

EPROM - Erasable Programmable ROM :-
* PROM device is erased and reprogrammed is called Erasable PROM.
* EPROM store 1's and 0's as a packet of charged in buried layer of IC chip.

EPROM Programming :-
* Erased cell in EPROM contains 1.
* 0's and 1's are presented in data.

Disadv :-
* not possible to erase selective information
* Entire memory can be erased before reprogramming.

EEPROM - Electrically Erasable Programmable ROM:-

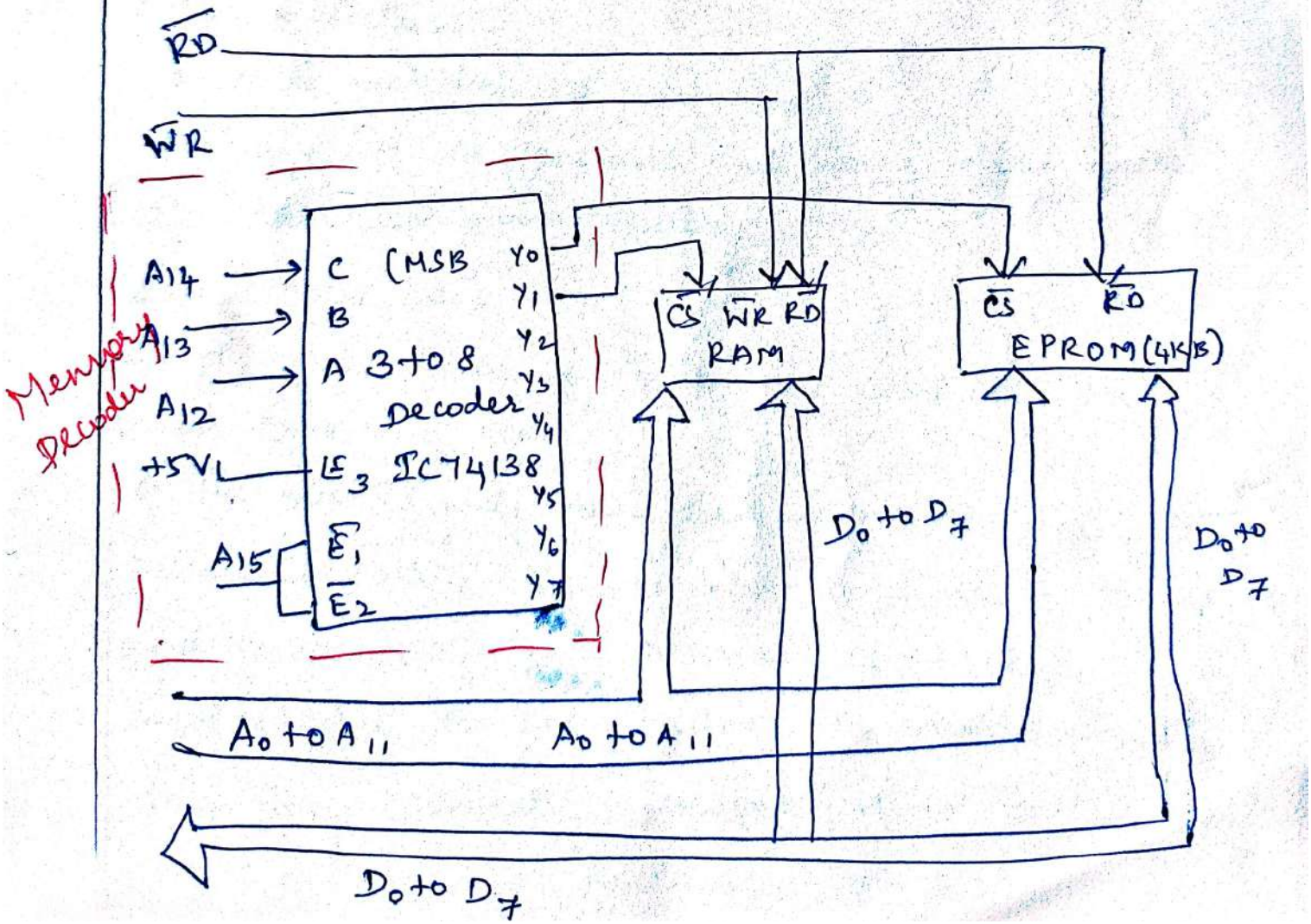
* ROM Device is EEPROM which is electrically alterable Programmable ROM (EAPROM).
* EEPROM is non-Volatile like EPROM but does not require ultraviolet light.

MEMORY DECODING:-

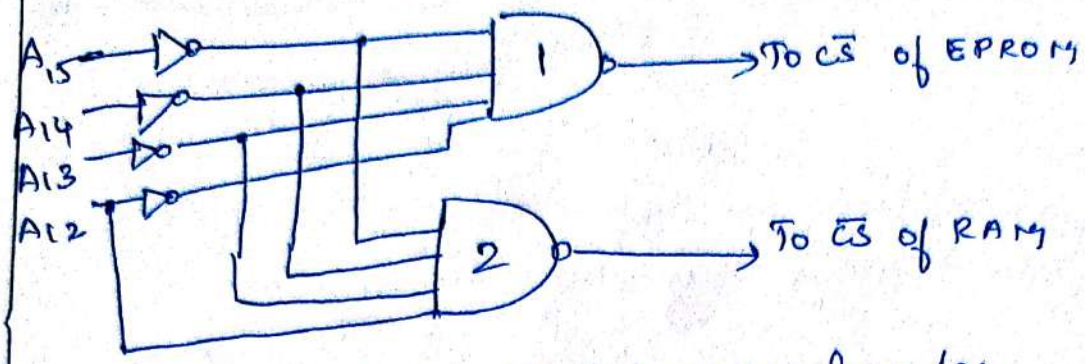
* The memory IC used in a digital system is selected or enabled only for the range of addresses assigned to it and this process called Memory Decoding.

Design of memory Decoder:-

* Address Signal EPROM & RAM,
  EPROM (4Kbyte): 000H to 0FFFH
  RAM (4Kbyte): 1000H to 1FFFH.



Memory Decoder using logic gates



* NAND gate 1 output becomes low for

$A_{15} A_{14} A_{13} A_{12} = 0000$

$A_{15} A_{14} A_{13} A_{12} = 0001$.

ERROR DETECTION AND CORRECTION :-

* The electrical signals affecting the data path of a memory unit may cause occasional errors in storing and retrieving the binary information.

Error Detection :-

* Error Detection is parity bit.
* Parity bit is generated and stored along with the data word in memory.

Error correction :-

* The error retrieval system is corrected using error correction code.
* The word will read back from memory, the associated Parity bits are also read from memory and compared with a new set of checks.
* check bit do not match the stored parity that they generate a unique pattern called a syndrome.

* A syndrome is a check bit that can be used to identify the bit that is error or correct.

i) Hamming code :-

* In Hamming code k-parity bits are added to an n-bit data word, forming a new word n+k bits. The bit positions are numbered in sequence 1 to n+k.

ie: $2^0 = 1$
$2^1 = 2$
$2^2 = 4$
$2^3 = 8$
$2^4 = 16$

To illustrate the Hamming code :-

* 8bit data 11000100

Bit position	1	2	3	4	5	6	7	8	9	10	11	12
Parity position	P_1	P_2		P_4				P_8				
8bit data word			1		1	0	0				0	0

ii) Single error correction, Double error Detection :-

* The Hamming code can detect and correct only a single error. By adding another parity bit to the coded word, The Hamming code can be used to correct a single error and detect double errors.

* 12 bit coded word becomes $00111001010 0 P_{13}$

* 13 bit word, $P = 0 \rightarrow$ Parity is correct (even parity)
$P = 1 \rightarrow$ 13 bits is incorrect.

Case i) $C = 0, P = 0$, no error

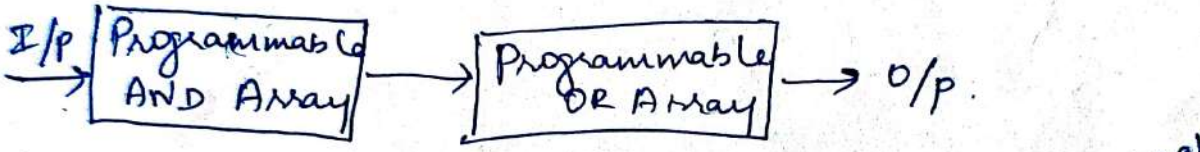
" ii) $C \neq 0, P = 1$, Single error occured

Case iii) $C \neq 0, P = 0$, Double error occured

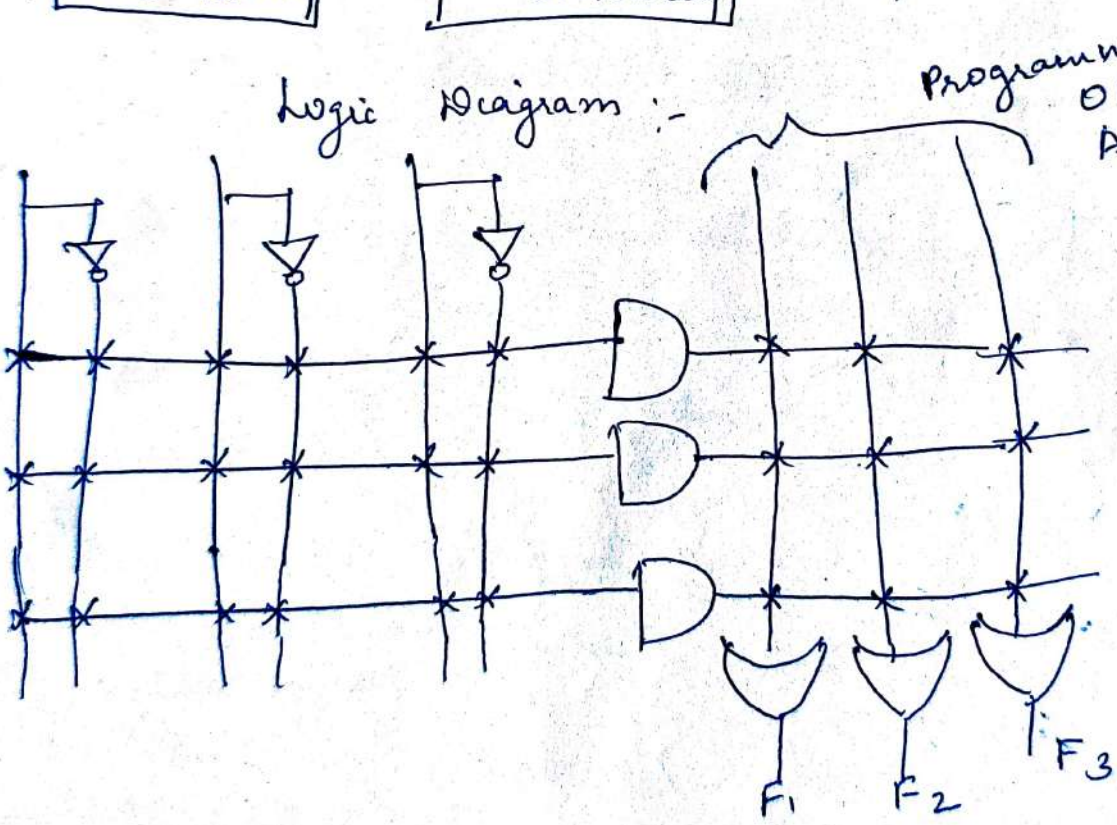
" iv) $C = 0, P = 1$, error occured in P_{13} bit.

PROGRAMMABLE LOGIC ARRAY:-

* PLA is a type of fixed architecture logic devices with programmable AND gates followed by Programmable OR Gates.
* AND & OR Gates inside the PLA are initially fabricated with fuses.
* Boolean fn's are implemented in Sum of Product (SOP) form.



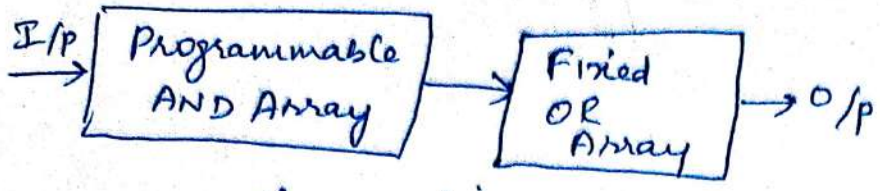
Logic Diagram :-



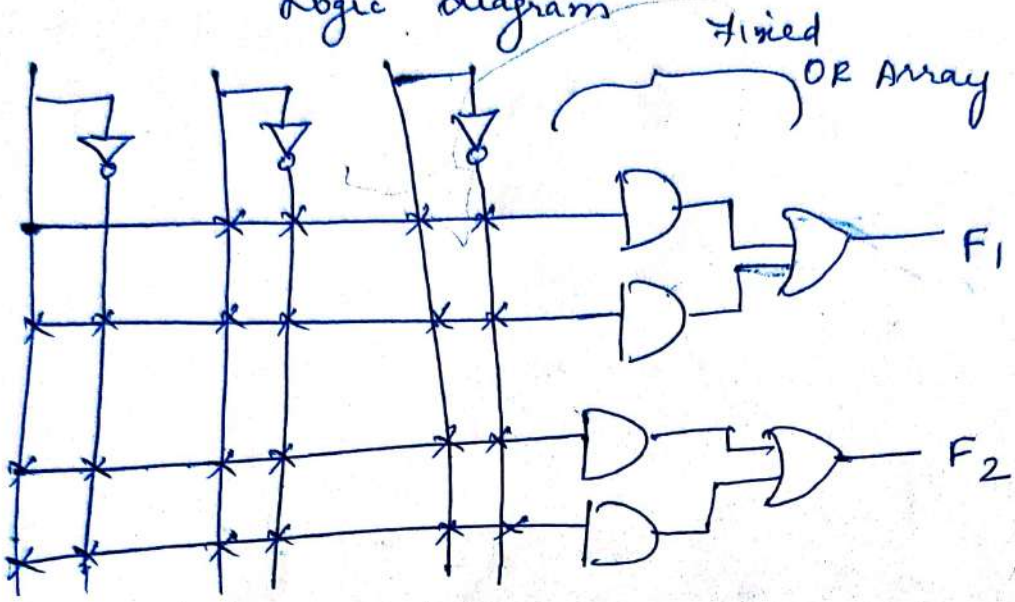
PROGRAMMABLE ARRAY LOGIC (PAL):-

* Programmable Array logic (PAL) is a type of fixed architecture logic device with Programmable AND Arrays

followed by fixed OR Arrays.



Logic diagram



SEQUENTIAL PROGRAMMABLE DEVICES :-

* Sequential Programmable Devices (SPD's) are electron circuits that are used for controlling mechanical devices or machines.
* SPD's can trigger actions such as counting, sequencing and timing for each other devices.
* They provide flexibility, reliability and efficiency to a wide Variety.
* Application of Sequential programmable Devices.
  - → Machine control and automation
  - → cybersecurity
  - → Medical instrumentation.
  - → utility operation monitoring and control.

APPLICATION OF SPECIFIC INTEGRATED CIRCUIT :- (ASIC):-

* An integrated circuit is a small silicon semiconductor crystall called a chip.
* components such as diodes, Transistors, resistors and capacitors.

Types of ASIC :-

i) Full custom ASIC :-

* An engineer designe some or all logic cells, circuits (or) layout specifically for one ASIC

ii) Standard cell Based ASIC :-

* A cell based ASIC uses predesigned logic cells known as standard cells.
* Larger predesigned cells like microcontrollers or microprocessors cknown as Mega cells.

iii) Gate Array Based ASIC :-

* The Transistor Predefined on silicon wafer.