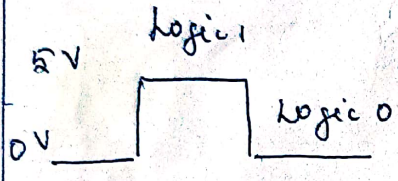


UNIT - I (NUMBER SYSTEMS AND CODE CONVERSIONS)

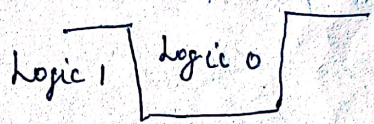
DIGITAL ELECTRONICS :

* 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.

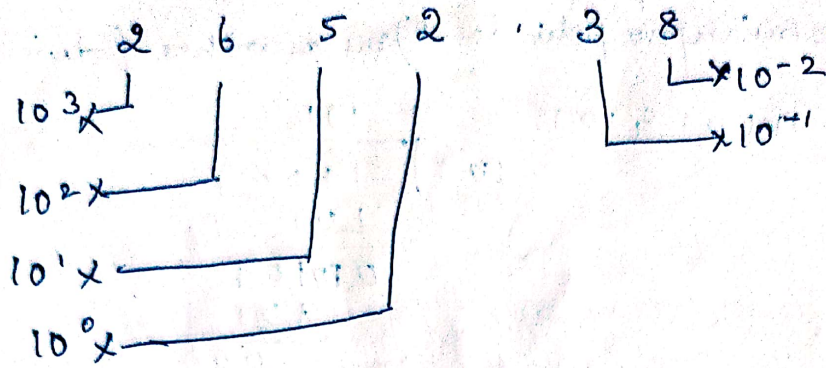
* 4 ways of numbering system:

- 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.

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



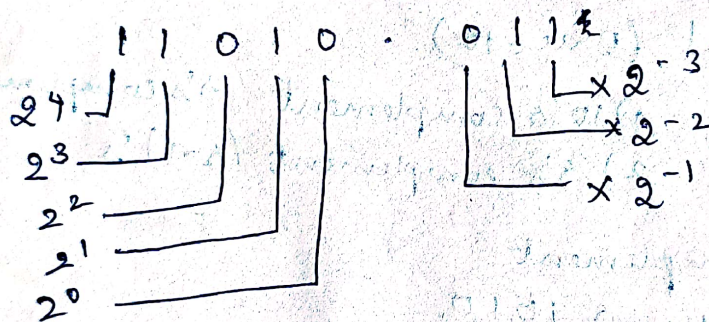
(ii) Binary :-

* 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 $2^0, 2^1, 2^2, \dots$ etc.

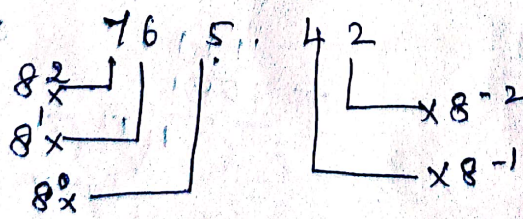
Eq:- $(11010.011)_2$



(iii) Octal :-

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

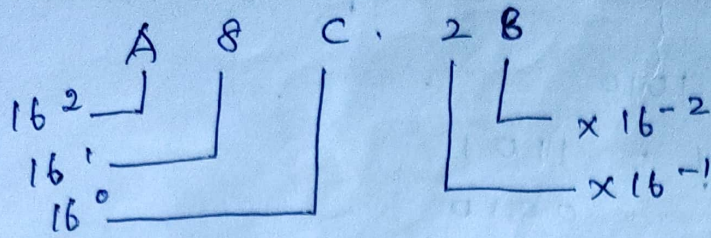
Eq:- $(765.42)_8$



(iv) Hexadecimal :-

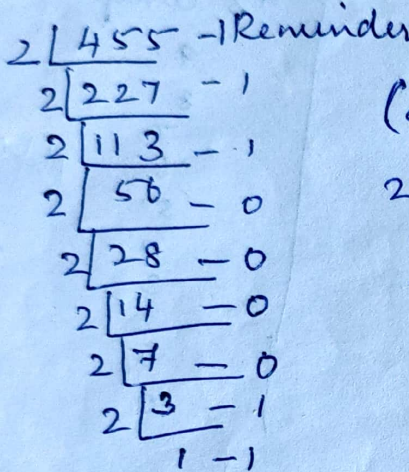
* 16 distinct digit symbols. 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 Plus the A, B, C, D, E and F as 16 digit.

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

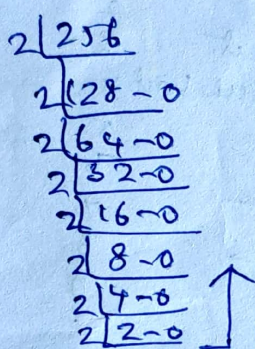


Problems:-

* Decimal to Binary
(455)₁₀

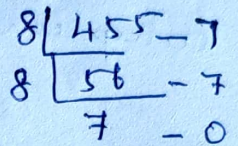


0.22 x 2 = 0.44
0.44 x 2 = 0.88
0.88 x 2 = 1.76
1.76 x 2 = 3.52
(256.22)₁₀



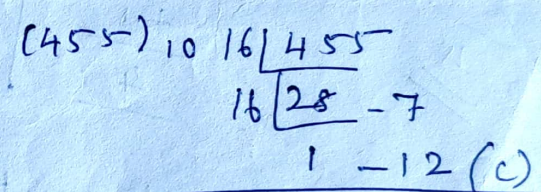
$\therefore (455)_{10} = (111000111)_2$

* Decimal to octal
(455)₁₀



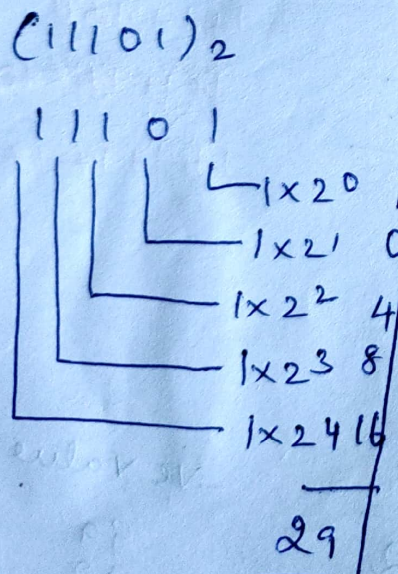
$\therefore (455)_{10} = (707)_8$

* Decimal to Hexadecimal
(455)₁₀



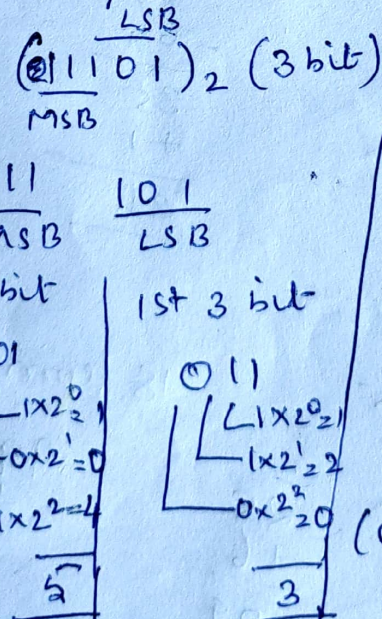
$\therefore (455)_{10} = (7C7)_{16}$

Binary to decimal



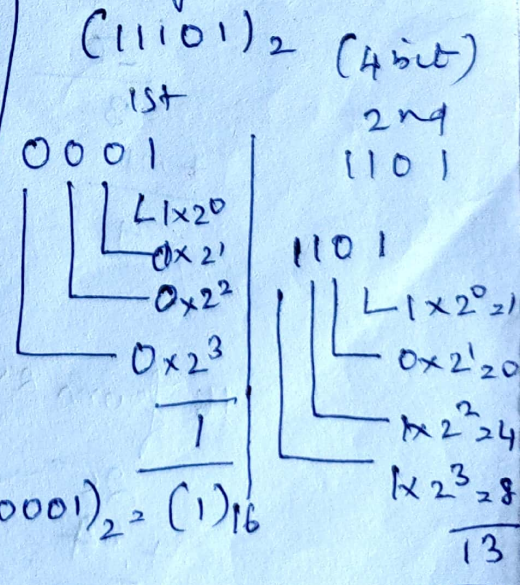
$(11101)_2 = (29)_{10}$

Binary to octal



$(101)_5 = (011)_3 = (35)_8$

Binary to Hexadecimal



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

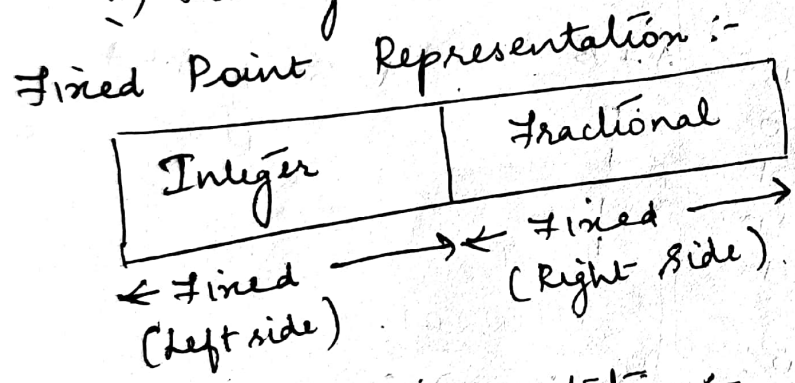
* Octal to Decimal
 $(35)_8$
 35
 $5 \times 8^0 = 5$
 $3 \times 8^1 = 24$
 $\underline{\quad}$
 29
 $\therefore (35)_8 = (29)_{10}$

to Binary
 $(35)_8$
 $3 \quad 5$
 $2 \overline{) 3} \text{---} 1$
 $2 \overline{) 2} \text{---} 0$
 $\underline{\quad}$
 1
 011
 $2 \overline{) 5} \text{---} 1$
 $2 \overline{) 2} \text{---} 0$
 $\underline{\quad}$
 101
 $\therefore (35)_8 = (011101)_2$

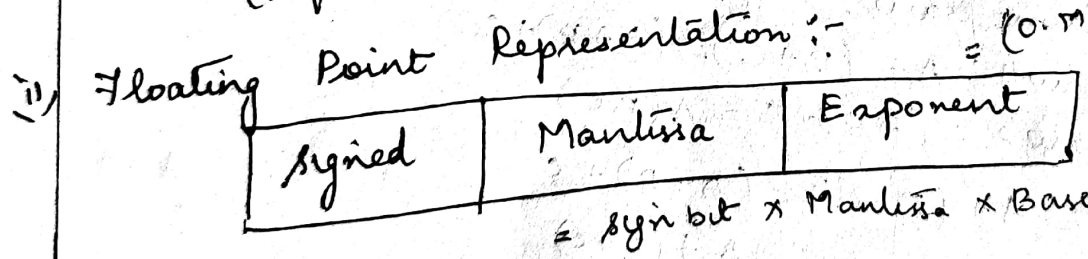
To Hexa
 $(35)_8$
 $0001 \quad 1101$
 $\underline{\quad} \quad \underline{\quad}$
 $1 \quad 13(D)$
 $\therefore (35)_8 = (1D)_{16}$

NUMBER REPRESENTATION:-

- i) Fixed Point Representation.
- ii) Floating Point Representation.



Eg: $(01101100)_2$
 2^2
 2^{-3}
 $(00011.011)_2$



CONVERSIONS:-

- * Decimal
- * Octal
- * Hexadecimal
- * Binary.

BOOLEAN ALGEBRA:- (REVIEW):-

- * Set of elements, set of operations
- * Two binary operator '+' and '.'
- * Element '0' respect to '+'
- * Element '1' respect to '.'

$A + 0 = 0 + A = A$
 $A \cdot 1 = 1 \cdot A = A$

* 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:-

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

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

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

THEOREMS:-

i) DeMorgan's Theorems:-

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

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

⇒

PRODUCT (Complement of AND = OR operation)

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

Theorem 2:- $\overline{A + B} = \bar{A} \cdot \bar{B}$ (SUM PRODUCT (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

⇒

SUM PRODUCT (Complement OR = AND operation)

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

ii) Duality Theorem:-

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

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

iii) Basic Theorem :-

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

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

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

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

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

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

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

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

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

⑥

$$\begin{bmatrix} A \cdot 1 = A \\ A + \overline{A} = 1 \\ A \cdot A = A \\ A\overline{A} = 0 \end{bmatrix}$$

Postulates

$$\begin{bmatrix} 1 + A = A \\ A \cdot 1 = A \\ 1 + A = 1 \end{bmatrix}$$

$$\begin{bmatrix} A + \overline{A} = 1 \end{bmatrix}$$

iv) Consensus Theorem :-

$$\begin{aligned} AB + \overline{A}C + BC &= AB + \overline{A}C \\ &= AB + \overline{A}C + (A + \overline{A})BC \rightarrow \text{redundant} \\ &= AB + \overline{A}C + AB + \overline{A}C \\ &= AB + \overline{A}C \end{aligned}$$

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 two or more variables combined with AND logic

Eg :- $f(A, B, C) = \overbrace{AB + B\overline{C} + A\overline{C}}^{\text{Sum}}$

Product

PRODUCT OF SUM SIMPLIFICATION:-

(7)

* Product of sum is called Product of Maxterms.

$$f(A, B, C) = (A+B) \cdot \overbrace{(B+C)}^{\text{Product}} \cdot (A+C)$$

↓
Sum terms

CANONICAL 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 $\rightarrow F=1$
0 Maxterm $\rightarrow F=0$

(i) Minterms:-

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

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 $Y = \sum m(111, 101, 010)$ [Canonical Form]

$$Y = m_7 + m_5 + m_2$$

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



ii) Maxterms :-

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

(8)

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

Eg :- Pos Form $Y = \prod M(000, 010, 101)$

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

$$Y = M_0, M_2, M_5$$

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.

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

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

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

Omit Repeated product terms

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

SIMPLIFICATION OF BOOLEAN EXPRESSIONS :-

* Minimization of Boolean Expressions.

Problems :- Examples :-

1) $F = A(A+C)$
 $= A \cdot A + A \cdot C$ [$\because A \cdot A = A$]
 $= A + A \cdot C$ [$\because 1 + C = 1$]
 $= A(1+C)$
 $F = A$

$A(A+C)$
 $A(A \cdot A)$
 8421
 000
 001
 010
 100
 $\overline{A} \cdot \overline{B}$

2) Simplify the given Boolean Expressions (x.)
 $F = X' + XY + XZ' + XY'Z'$

$F = \overline{X} + XY + X\overline{Z} + X\overline{Y}\overline{Z}$ [$\because A + \overline{A}B = A + B$]
 $F = \overline{X} + Y + X\overline{Z} + X\overline{Y}\overline{Z}$ [$\because A + AB = A$]
 $F = \overline{X} + Y + X\overline{Z}$
 $F = \overline{X} + X\overline{Z} + Y = \overline{X} + \overline{Z} + Y$ [$\because A + \overline{A}B = A + B$]

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	0	\overline{A}
	1	A

ii) Two Variable :-
 * $2^n = 2^2 = 4$

	B	\overline{B}	B
A	00	01	
\overline{A}	10	11	
	$\overline{A}\overline{B}$	$\overline{A}B$	
	$A\overline{B}$	AB	

iii) Three Variables :-

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

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

iv) Four Variables :-

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

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

Grouping Two adjacent pairs :-

eg:-

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

$F = \bar{B}$

2 variable:-

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

$F = A + B$

3 Variable

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

$F = \bar{A}C$

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

$F = A\bar{C}$

4 Variables :-

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

$F = \bar{A}BD$

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

$F = C$

Grouping Eight-Adjacent Ones (octet):-

(11)

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

$F = B$

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

$F = \bar{B}$

Problems:- 1) Simplify the following expression using

k-map method.

$Y = \sum m(7, 9, 10, 11, 12, 13, 14, 15)$

Soln:-

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

7	8421
9	0111
10	1001
11	1010
12	1011
13	1100
14	1101
15	1110
	1111

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

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

(ii) Minimization of POS Forms:-

4 Variables

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

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

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

1) Simplify Boolean Expression using K-map.

(12)

$$Y(A, B, C, D) = \sum m(5, 7, 13, 15)$$

	CD		
AB	0	1	1
A+B	1	1	1
A+B	0	0	1
A+B	1	0	1
A+B	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 expressions using K-map

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

	BC	BC	BC
A	0	1	3
A	4	5	7
A	2	6	

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

	BC	BC	BC
A	1	1	1
A	X	1	0
A	X	1	1

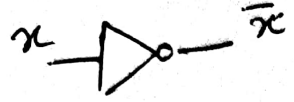
IMPLEMENTATION OF BOOLEAN EXPRESSIONS USING

UNIVERSAL GATES:-

Digital Logic Gates:-

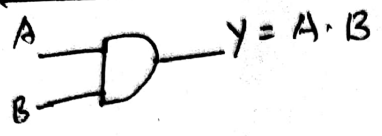
- * NOT
- * AND
- * OR
- * NAND
- * NOR
- * Exclusive-OR (EX-OR)
- * Exclusive-NOR (EX-NOR)
- * Buffer.

i) NOT Gates :-



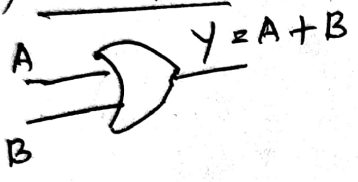
I/p		O/p
x	\bar{x}	
0	1	
1	0	

ii) AND Gates :-



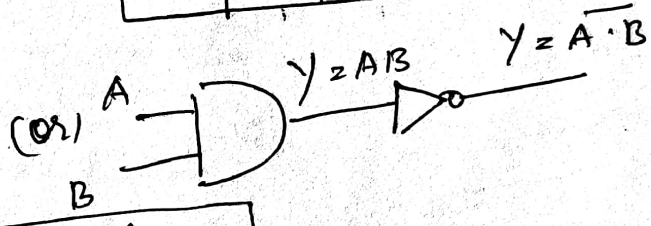
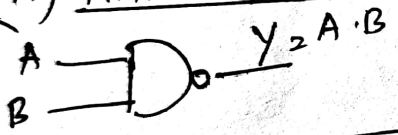
I/p		O/p
A	B	$y = A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

iii) OR Gates :-



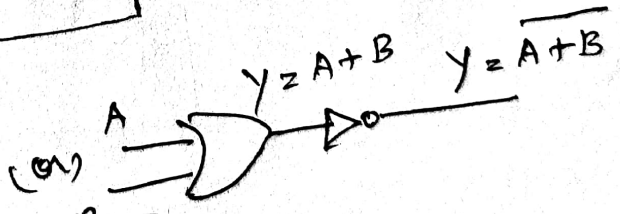
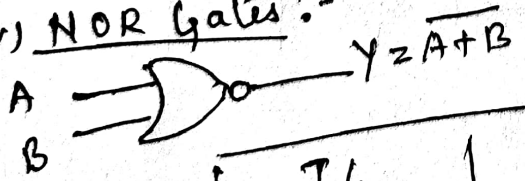
I/p		O/p
A	B	$y = A + B$
0	0	0
0	1	1
1	0	1
1	1	1

iv) NAND Gates :-



I/p		O/p
A	B	$y = A \cdot \bar{B}$
0	0	1
0	1	1
1	0	1
1	1	0

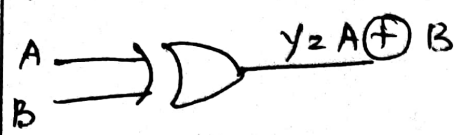
v) NOR Gates :-



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

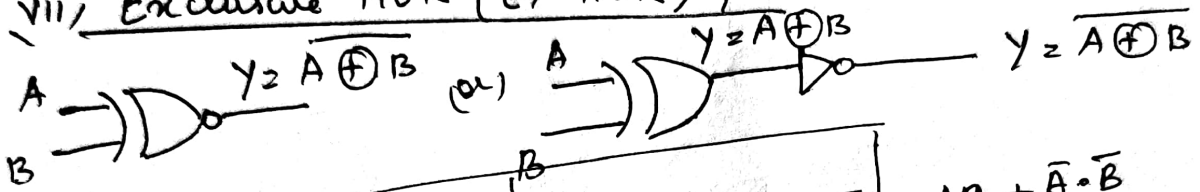
vi, Exclusive (OR) (EX-OR) Gate :-

$A\bar{B} + \bar{A}B$ (14)



I/p		O/p
A	B	$Y = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

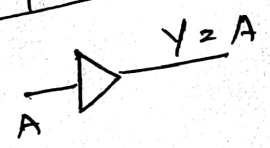
vii, Exclusive NOR (EX-NOR) Gate :-



I/p		O/p
A	B	$Y = A \oplus B$
0	0	1
0	1	0
1	0	0
1	1	1

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

viii, Buffer :-



I/p	O/p
A	Y
0	0
1	1

UNIVERSAL GATES :- (TABULATION METHODS) :-

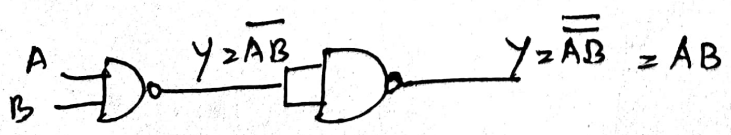
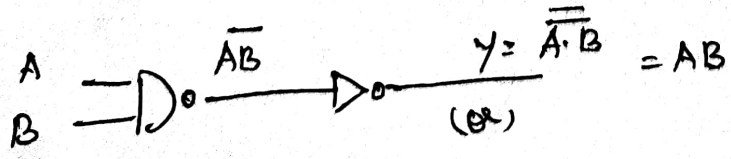
* NAND and NOR gates are universal gates.

* AND, OR, NOT → basic gates

NAND GATE :-

Implement AND operation using NAND gate :-

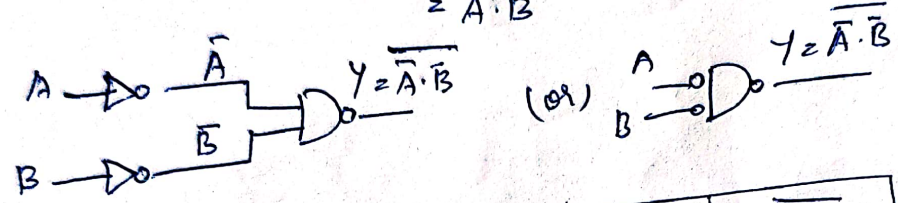
$Y = A \cdot B$
 $Y = \overline{\overline{A \cdot B}}$ ($\because \overline{\overline{A}} = A$)



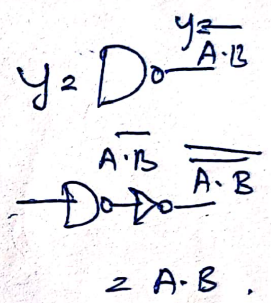
A	B	$\bar{A}\bar{B}$	$\overline{A\bar{B}}$
0	0	1	0
0	1	1	0
1	0	1	0
1	1	0	1

Implement OR operation using NAND Gate:-

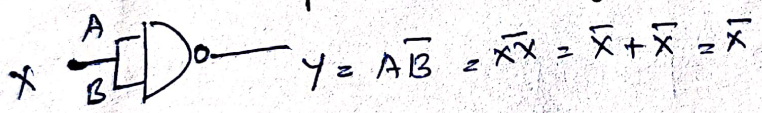
$$Y = A + B = \overline{\bar{A} \cdot \bar{B}}$$



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

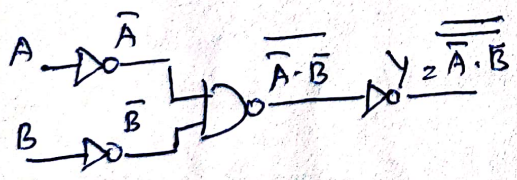


Implement NOT operation using NAND Gate:-



Implement NOR operation using NAND Gate:-

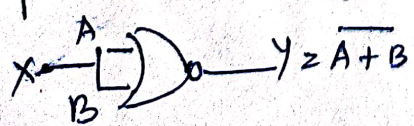
$$Y = \overline{A + B} = \overline{\overline{\bar{A} \cdot \bar{B}}}$$



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:-



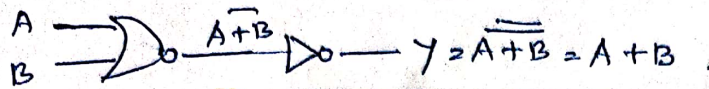
A	B	$A + B$
0	0	1
0	1	0
1	0	0
1	1	0

$x=0 \left\{ \begin{array}{l} y=1 \\ y=0 \end{array} \right.$

$x=1 \left\{ \begin{array}{l} y=0 \\ y=0 \end{array} \right.$

Implement OR Function using NOR Gate :-

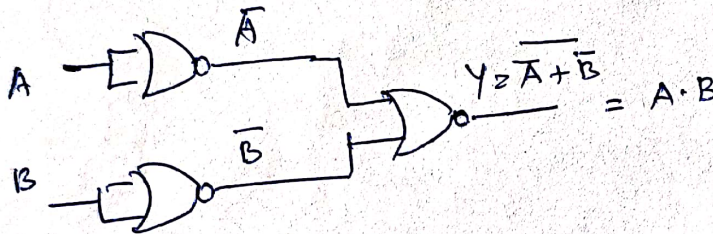
$$\overline{\overline{A+B}} = A+B$$



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

Implement AND Function using NOR Gate :-

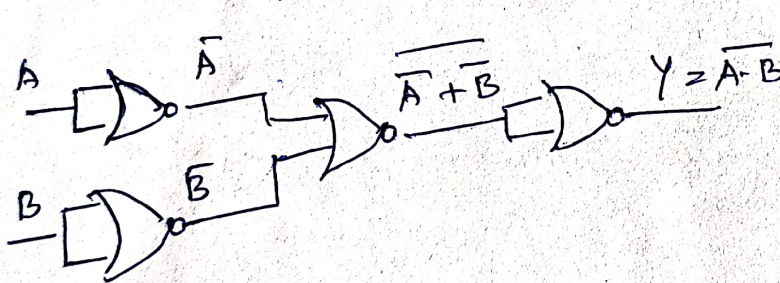
$$Y = A \cdot B \Rightarrow \overline{\overline{A+B}}$$



A	B	A · B
0	0	0
0	1	0
1	0	0
1	1	1

Implement NAND function using NOR Gate :-

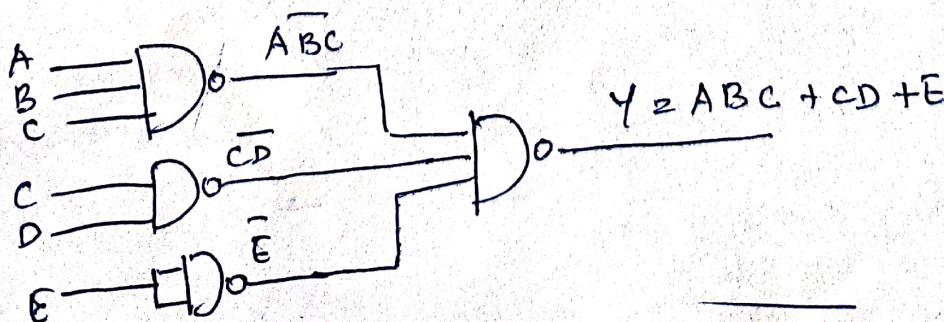
$$Y = \overline{A \cdot B} = \overline{\overline{A+B}}$$



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

NAND-NAND Implementation :-

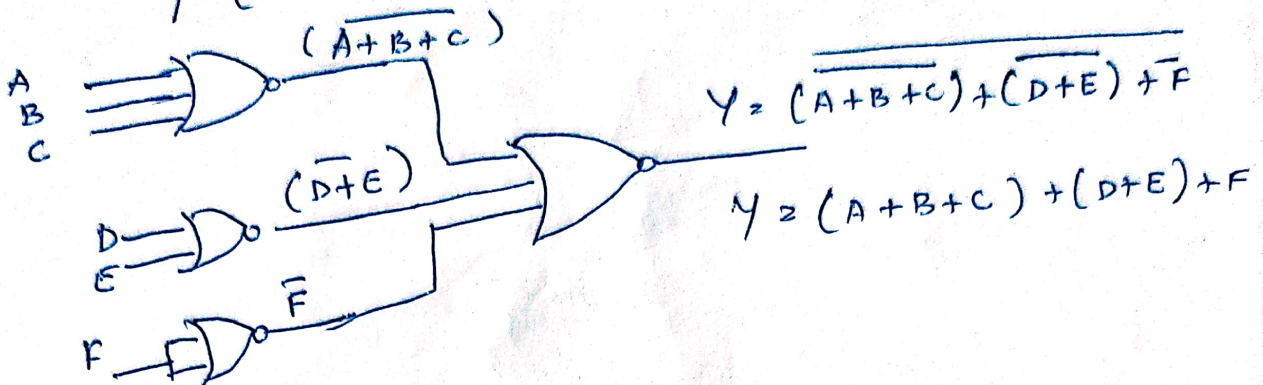
* AND-OR logic $Y = ABC + CD + E$ (SOP)



NOR-NOR Implementation:-

* POS Form

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



TABULATION METHOD:- (2-53)

of Quine-McCluskey

Procedure:- S1: Min terms in binary form

S2: Min terms to no. of 1's and split term

S3: - Compare binary number with every term next higher category.

S4: - Repeat S3

S5: - Prime implicants.

S6: Prime Implicant chart.

Eg:- minis of (a, b, c, d) $\leq m$ (0, 1, 2, 3, 8, 9)

min terms	Binary Representation
m ₀	0000
m ₁	0001
m ₂	0010
m ₃	0011
m ₈	1000
m ₉	1001

min terms	Binary Representation	Group
m ₀	0000	Group 0 1's zero
m ₁	0001	Group 1
m ₂	0010	
m ₈	1000	

UNIT - II (COMBINATIONAL LOGIC CIRCUIT AND DESIGN)

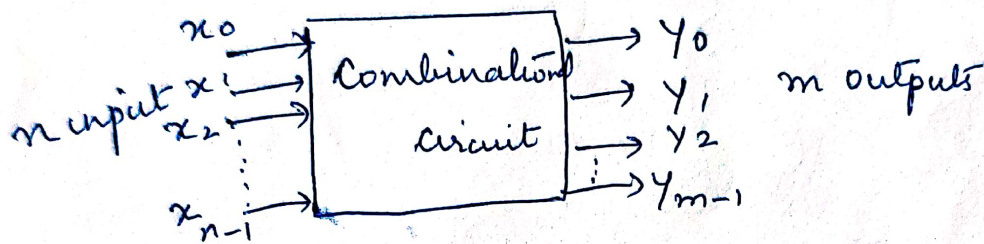
18

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 McClusky method.
- S4 :- Draw logic diagram.
Practical 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 capability

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 converter :-

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	0
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 :-

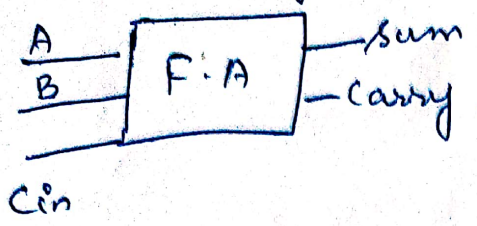
BCD code				Excess-3 code			
B ₃	B ₂	B ₁	B ₀	E ₃	E ₂	E ₁	E ₀
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

4) Excess-3 TO BCD Converter :-

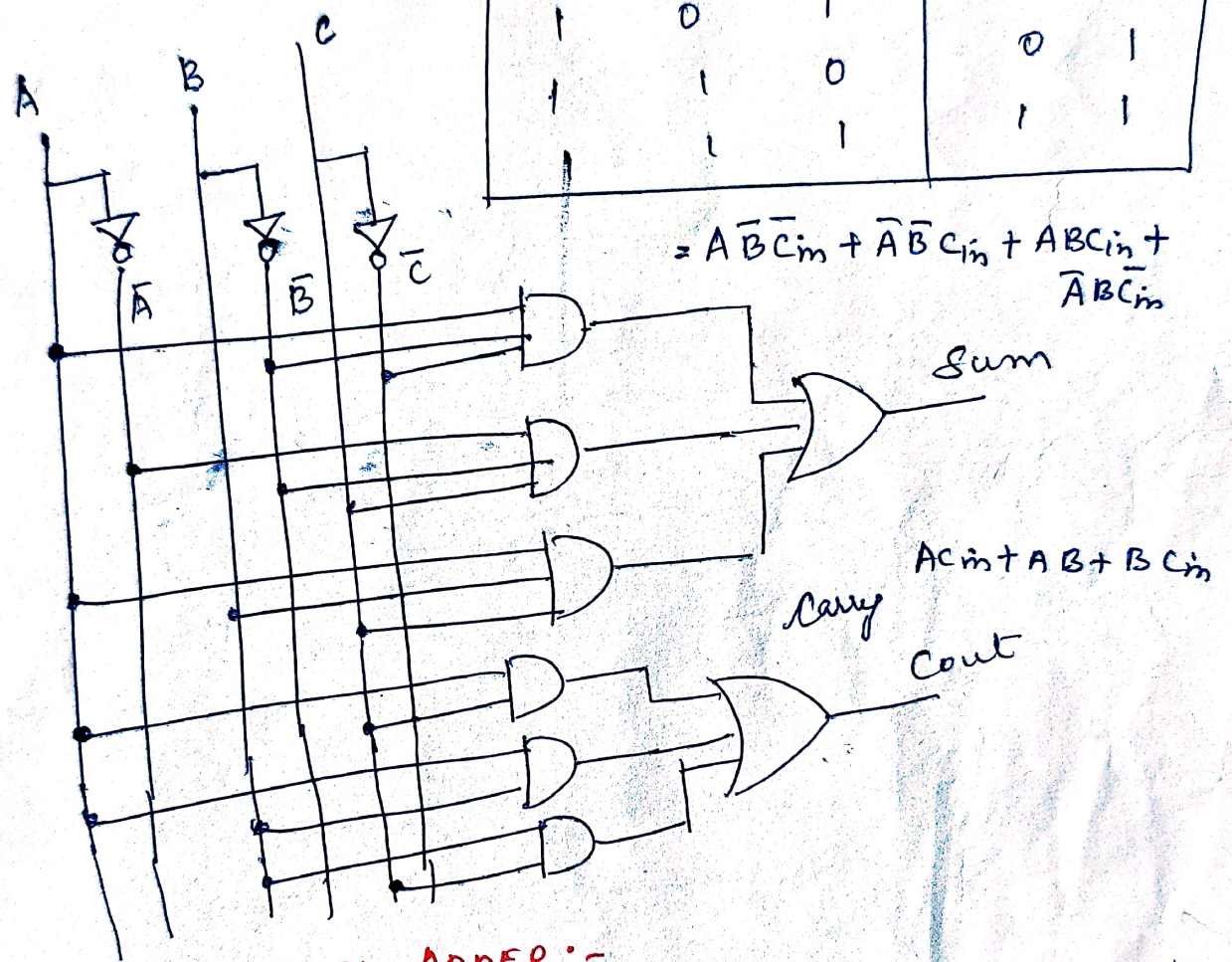
Excess-3				BCD			
E ₃	E ₂	E ₁	E ₀	B ₃	B ₂	B ₁	B ₀
0	0	1	1	0	0	0	0
0	0	1	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	0	0	0	0	1	1
0	1	1	0	0	0	1	1
0	1	1	1	0	1	0	0
1	0	0	0	0	1	0	1
1	0	0	1	0	1	1	0
1	0	1	0	0	1	1	1
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	1

ii) Full adder :-

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

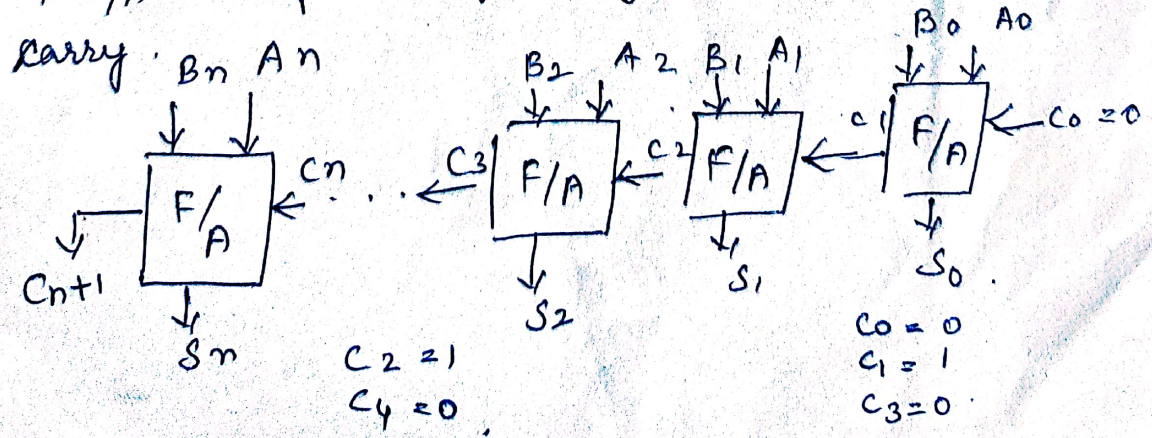


A	B	C _{in}	S	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



BINARY PARALLEL ADDER :-

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



CARRY LOOK AHEAD ADDER :-

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

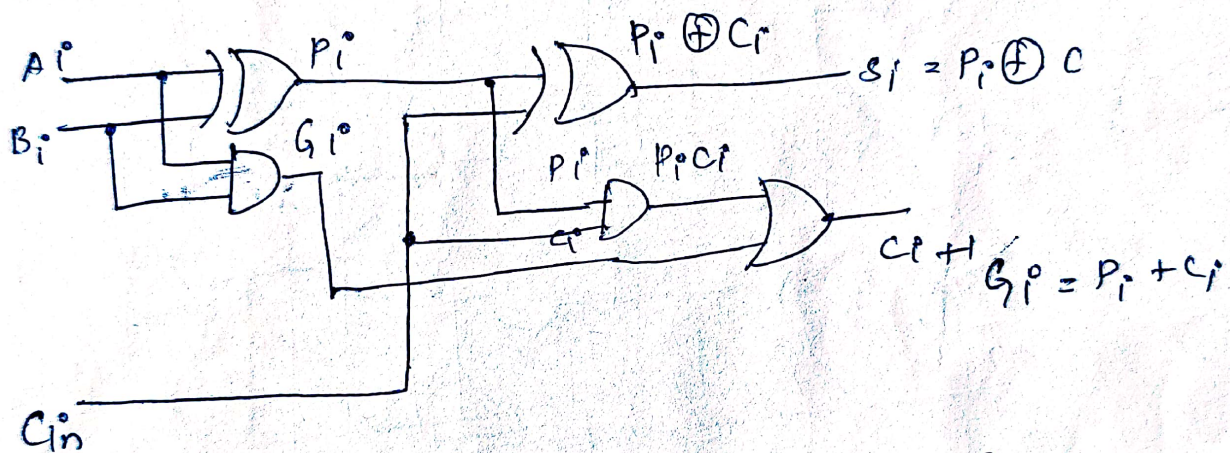
* Augend (A) and Addend (B).

Augend A = 0101 Addend B = 0011

	Cin	1	1	0	0
Augend - A		0	1	0	1
Addend - B		0	0	1	1
S		1	0	0	0
Cout		0	1	1	1

Propagation 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

$P_i = A_i \oplus B_i \rightarrow$ carry propagate

$G_i = A_i B_i \rightarrow$ 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

$C_1 = G_0 + P_0 C_0$

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

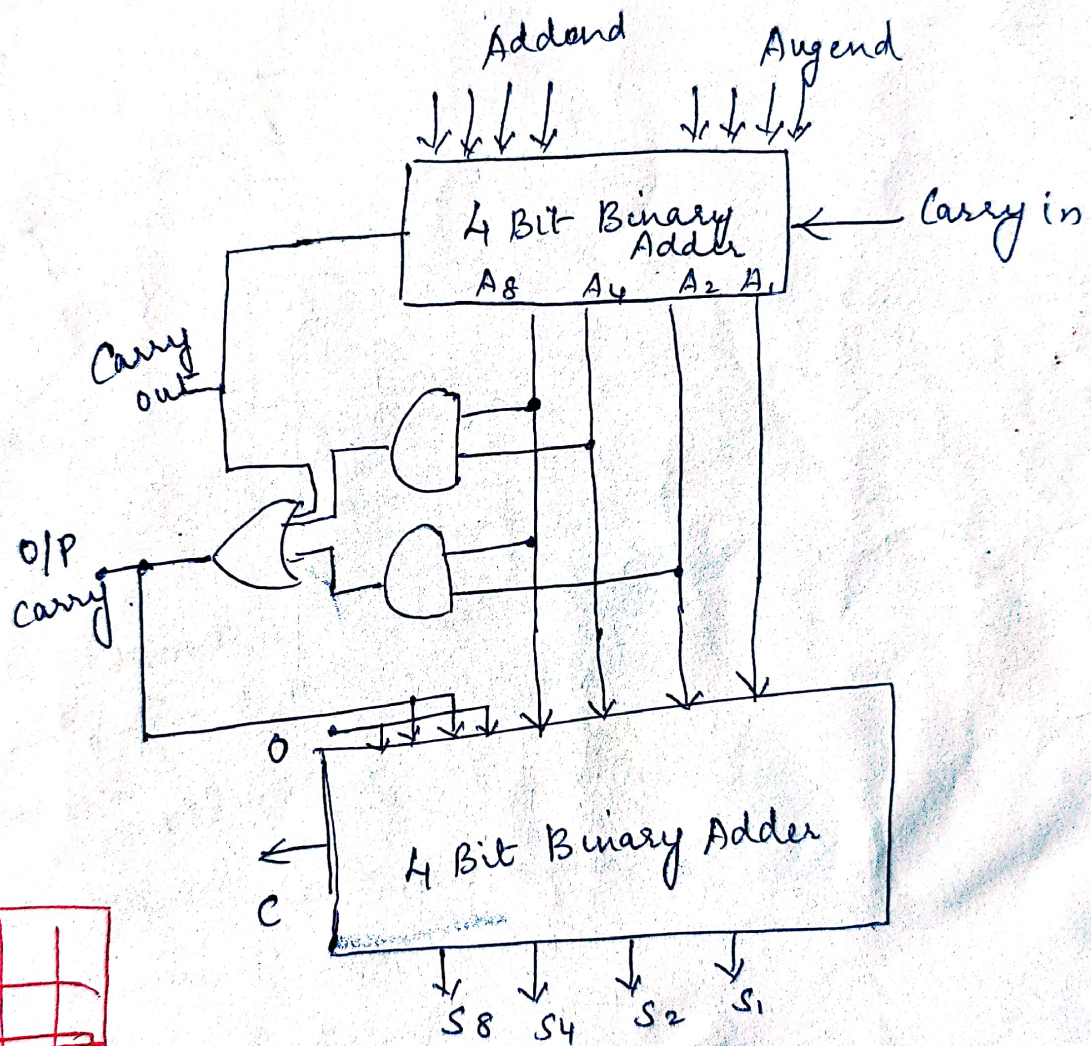
$C_3 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$

BCD ADDER

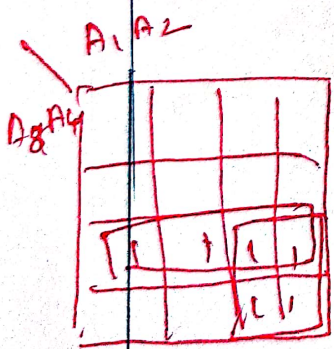
- * BCD adder is a circuit that adds two BCD digits in parallel, and produce a sum in BCD.
- * 0000 to 1001 each digit-coded 4-bit binary number.

Requirement to implement BCD Adder.

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



$$C = K + A_8 A_4 + A_8 A_2$$

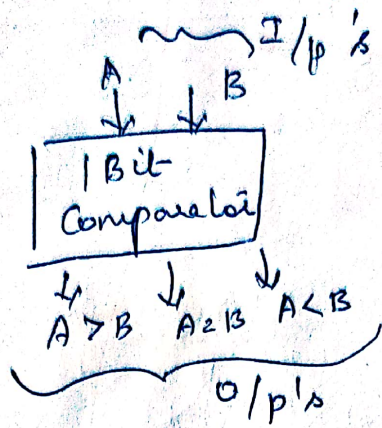


MAGNITUDE COMPARATOR :-

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

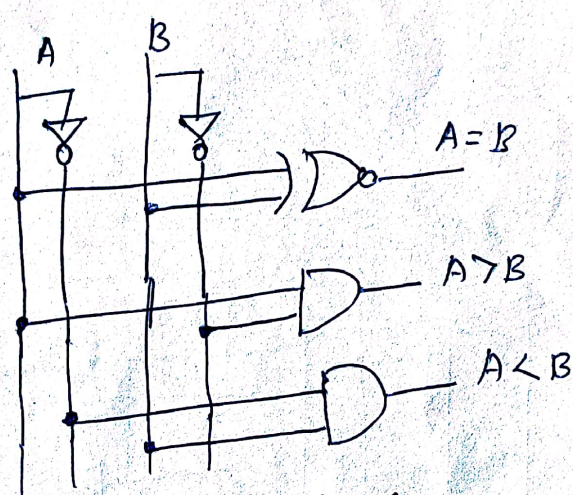
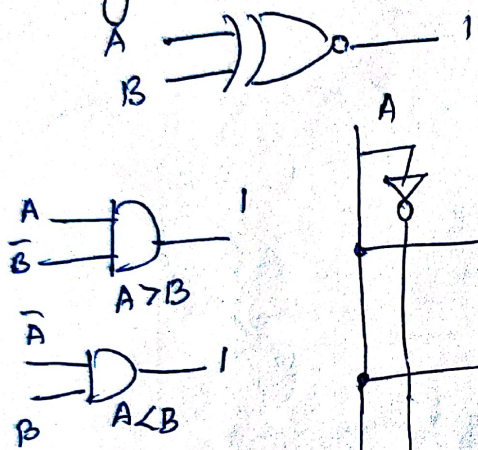
a equal to the other number.

* 3 binary variables $A > B$, $A = B$ (or) $A < B$.



Single Bit magnitude comparator :-

$A = B \rightarrow EF-NOR$



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 :-

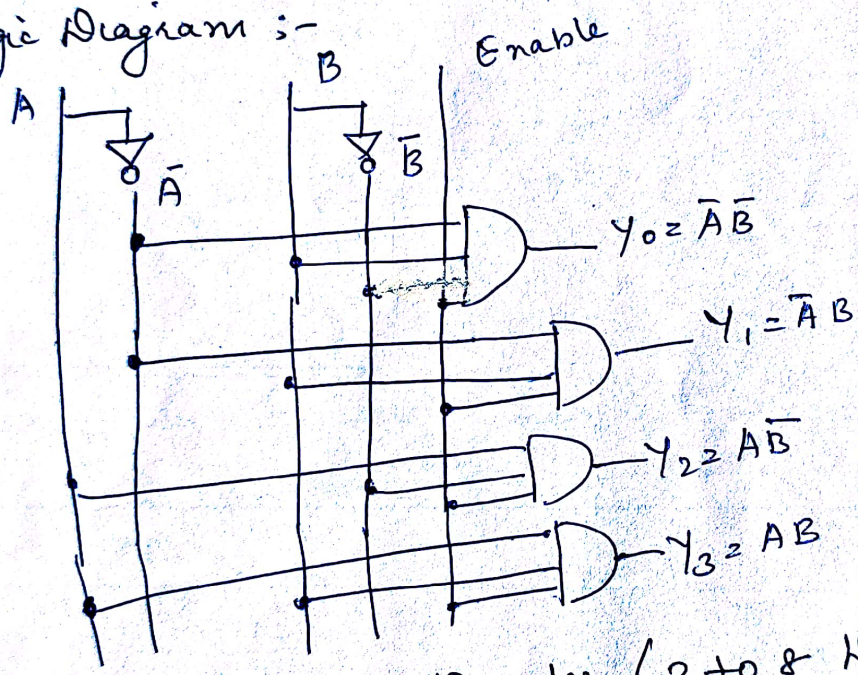
* A decoder is a combinational circuit that converts binary information from n input lines to max. 2^n o/p lines.

* 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) :-

$Y_0 = \bar{A}\bar{B}\bar{C}$, $Y_1 = \bar{A}\bar{B}C$, $Y_2 = \bar{A}B\bar{C}$, $Y_3 = \bar{A}BC$, $Y_4 = A\bar{B}\bar{C}$, $Y_5 = A\bar{B}C$, $Y_6 = AB\bar{C}$, $Y_7 = ABC$.

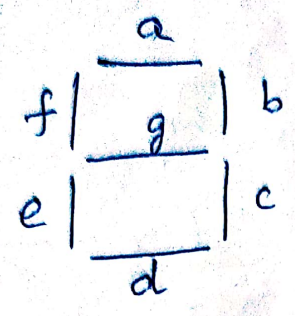
iii) BCD To Decimal Decoder :-

* BCD has value 0 to 9 with four bit binary
 * I/p A, B, C, D and O/p $Y_0 - Y_9$

$Y_0 = \bar{A}\bar{B}\bar{C}\bar{D}$, $Y_1 = \bar{A}\bar{B}\bar{C}D$, $Y_2 = \bar{A}\bar{B}C\bar{D}$, $Y_3 = \bar{A}\bar{B}CD$,
 $Y_4 = \bar{A}B\bar{C}\bar{D}$, $Y_5 = \bar{A}B\bar{C}D$, $Y_6 = \bar{A}BC\bar{D}$, $Y_7 = \bar{A}BCD$,
 $Y_8 = A\bar{B}\bar{C}\bar{D}$, $Y_9 = A\bar{B}\bar{C}D$.

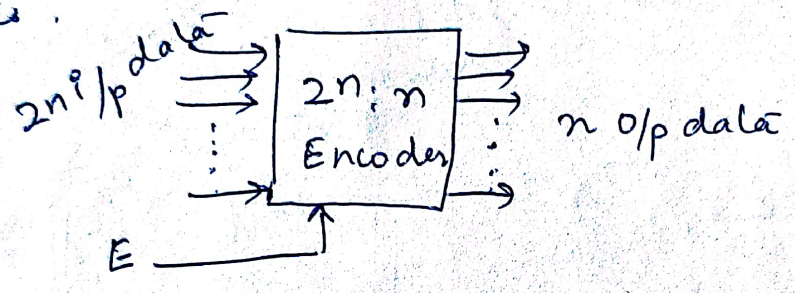
iii) BCD to Seven Segment Decoder:-

* Decimal Digit \rightarrow 0 to 9



ENCODER:-

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



PRIORITY ENCODER:-

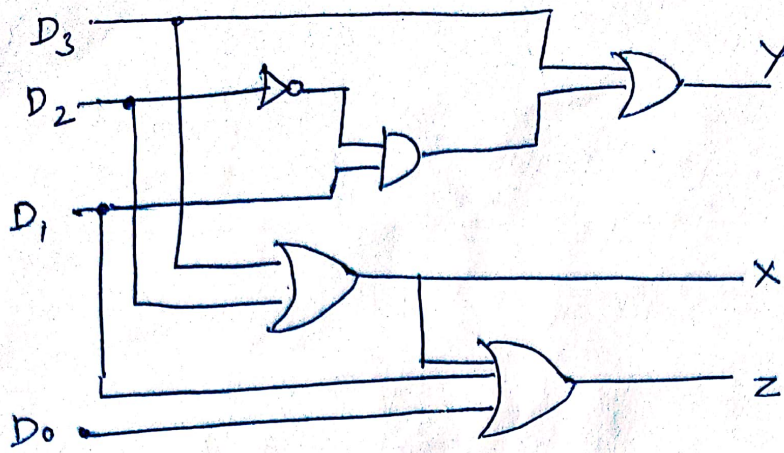
* Priority encoder such that two or more i/p's are equal to 1 at same time.

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

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

$$Z = D_0 + D_1 + \bar{D}_2 + D_3$$

Logic diagram four-input priority encoder:-

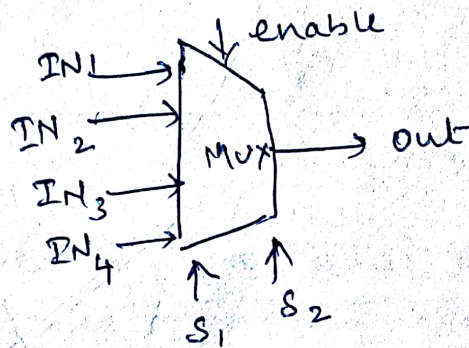


MUX/DEMUX :-

Multiplexers:-

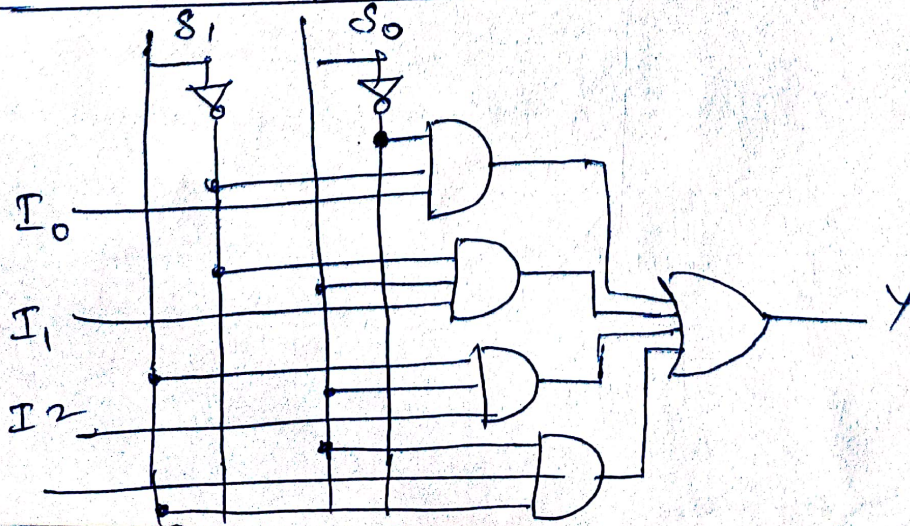
* Many to one

* 2^n I/p lines and n selection lines.



1) 4 to 1 Multiplexer :-

Data I/p	Selection lines		O/p Y
	S_1	S_0	
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 :-

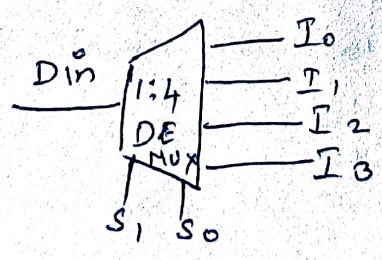
* 8 input lines $2^3 = 8$

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 o/p lines.

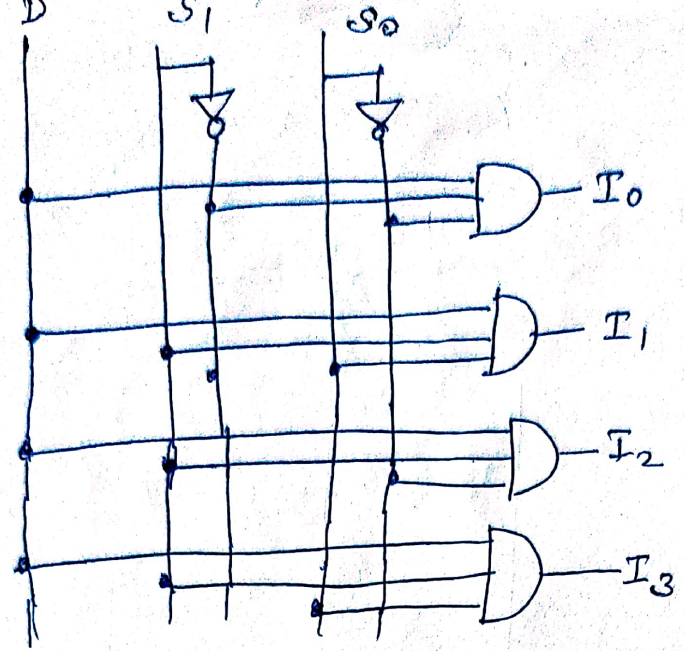
* Din has path to all four o/p information. It is also called 'Data Distributor'



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		Data I/p	o/p's			
S_1	S_0		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



ii) 1 to 8 Demultiplexer :-

* Single I/p and 8 O/p (I₀ to I₇)

$$\begin{aligned}
 I_0 &= D \cdot \bar{S}_2 \cdot \bar{S}_1 \cdot \bar{S}_0 & I_3 &= D \cdot \bar{S}_2 \cdot S_1 \cdot S_0 & I_6 &= D \cdot S_2 \cdot S_1 \cdot \bar{S}_0 \\
 I_1 &= D \cdot \bar{S}_2 \cdot \bar{S}_1 \cdot S_0 & I_4 &= D \cdot S_2 \cdot \bar{S}_2 \cdot \bar{S}_0 & I_7 &= D \cdot S_2 \cdot S_1 \cdot S_0 \\
 I_2 &= D \cdot \bar{S}_2 \cdot S_1 \cdot \bar{S}_0 & I_5 &= D \cdot S_2 \cdot \bar{S}_1 \cdot S_0 & & &
 \end{aligned}$$

CASE STUDY:

PARITY GENERATOR / CHECKER:-

* Purpose of detecting errors during Transmission of binary information

* Parity bit in Transmitter called Parity generator

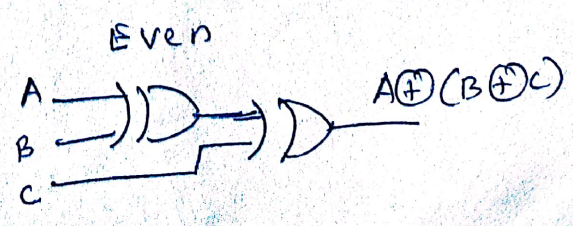
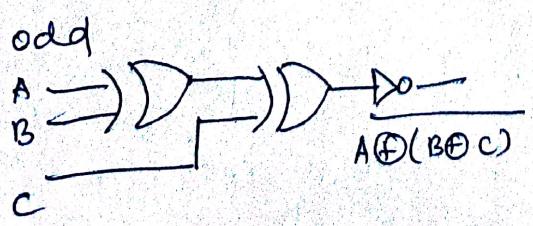
* Parity bit in Receiver is Parity checker.

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

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

odd = $A \oplus (B \oplus C)$

Even = $A \oplus (B \oplus C)$

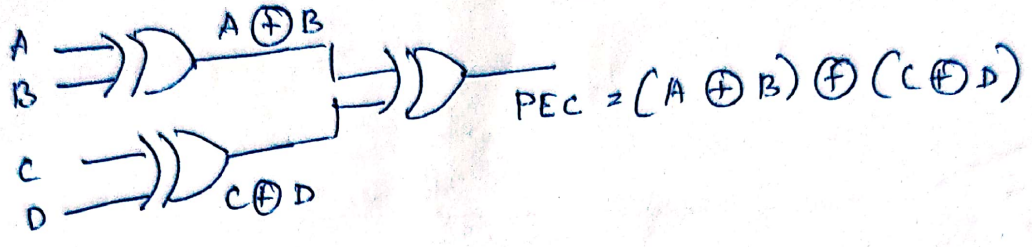


Parity checker:-

* check the error.

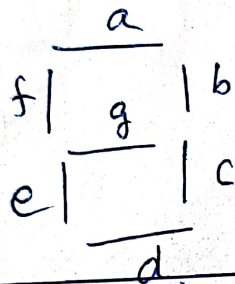
* Parity error checker will be 1.
will be 0 if no error.

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



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) :- 32

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 :-

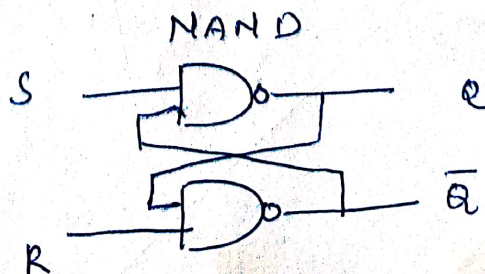
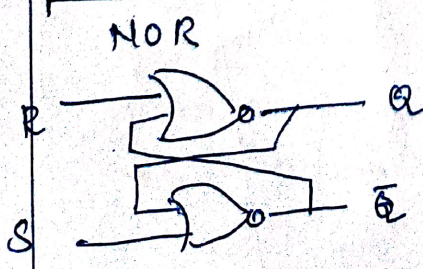
* 2 i/p Set (S) and Reset (R), two universal gates NAND or NOR.

NOR Based SR Latch

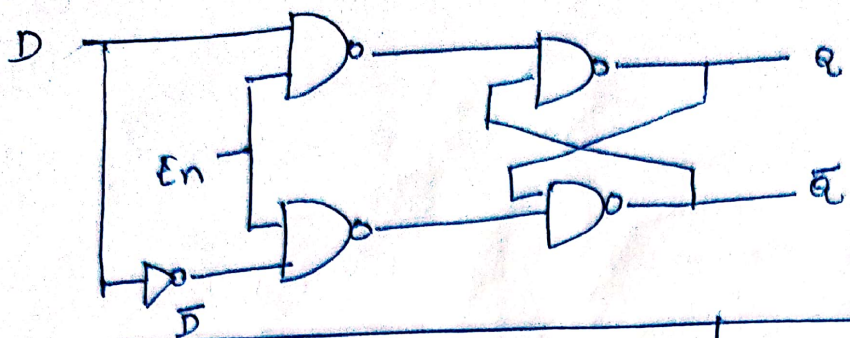
I/p's		O/p's		Remarks
S	R	Q_{n+1}	\bar{Q}_{n+1}	
0	0	Q_n	\bar{Q}_n	no change
0	1	0	1	Reset
1	0	1	0	Set
1	1	X	X	Indeterminate

NAND Based SR Latch

I/p		o/p		Remark
S	R	Q_{n+1}	\bar{Q}_{n+1}	
0	0	X	X	Indeterminate
0	1	1	0	Set
1	0	0	1	Reset
1	1	Q_n	\bar{Q}_n	no change
		$Q_n = \text{Present state}$ $Q_{n+1} = \text{Next state}$		



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.

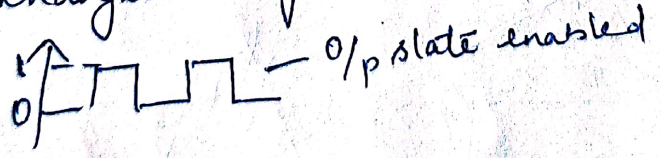
* Latches are controlled by clock called Flip Flops.

- i) Level Triggering
- ii) Edge Triggering

Level Triggering :-

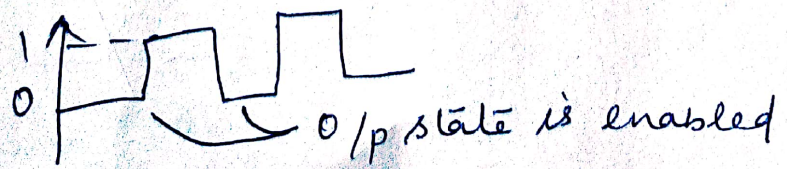
a) +ve level Triggering :-

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



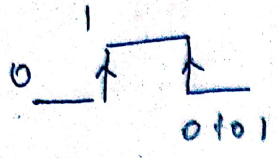
b) -ve level :-

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

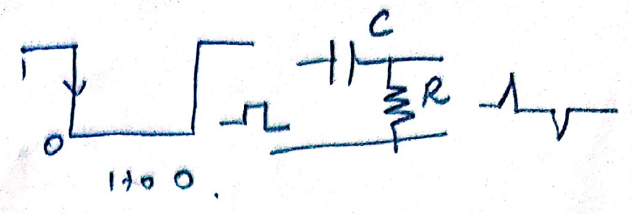


ii) Edge Triggering:-

+ve Pulse

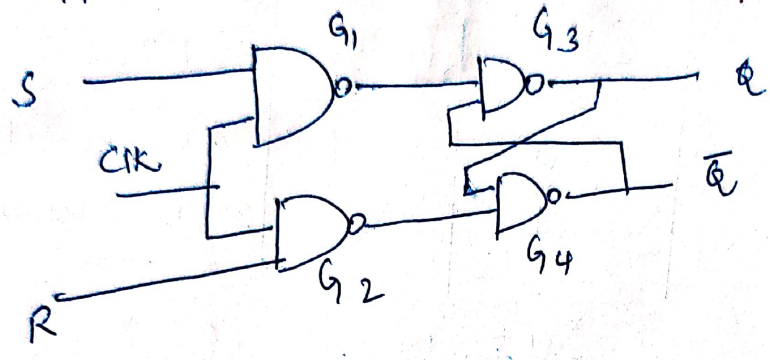


-ve Pulse



SR Flip Flops:-

* 3 i/p → clock, set and Reset, 2 o/p Q and \bar{Q}



$Q_n = 0$
 $Q_n = 1$

Characteristic Table:-

Present Q_n	Data i/p's		next state Q_{n+1}	Remarks
	S	R		
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

Excitation Table

Characteristic equation:-

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

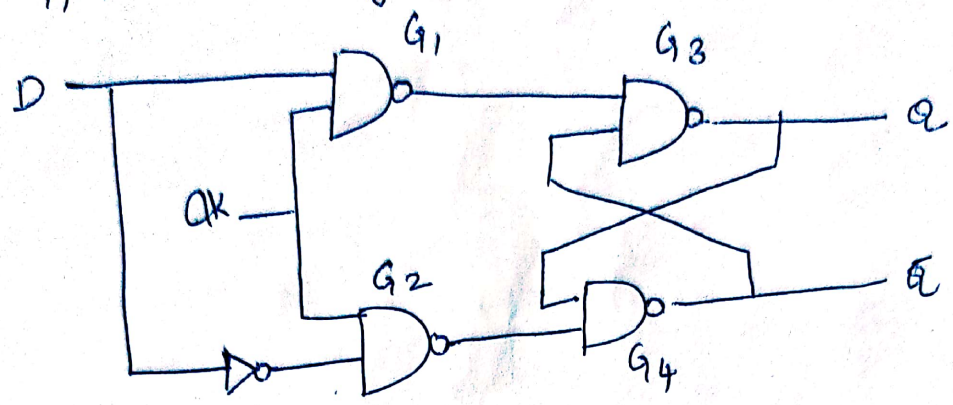
$Q_{n+1} = S + Q_n \bar{R}$

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

Characteristic Equations :-

	D	\bar{D}	D
Q_n	0	1	1
\bar{Q}_n	0	1	1
Q_n	0	1	1

$Q_{n+1} = D$

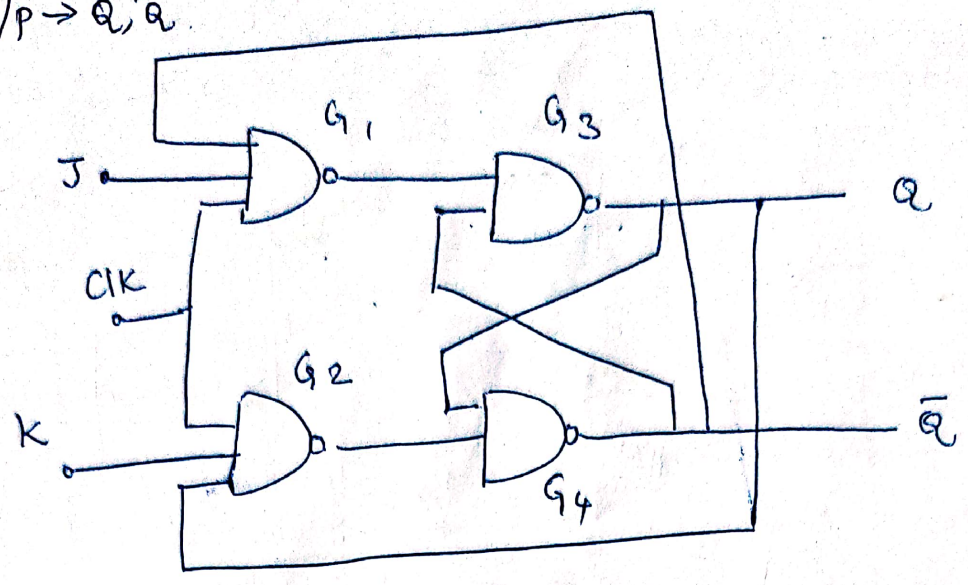
Excitation Table :-

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 :-

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

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

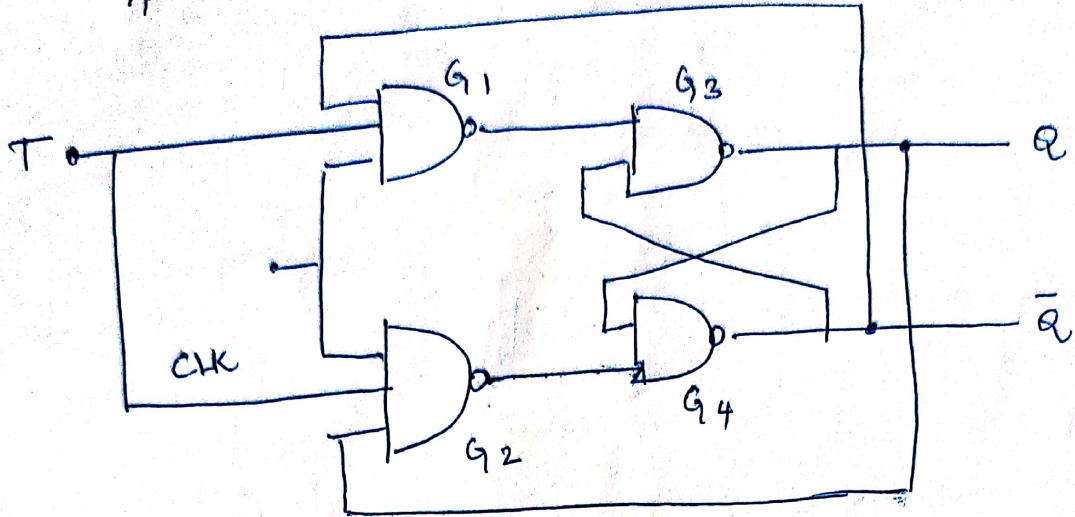
Excitation Table :-

P.S Q_n	N.S Q_{n+1}	I/p	
		J	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 $\rightarrow Q$ and \bar{Q}



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	1	0	

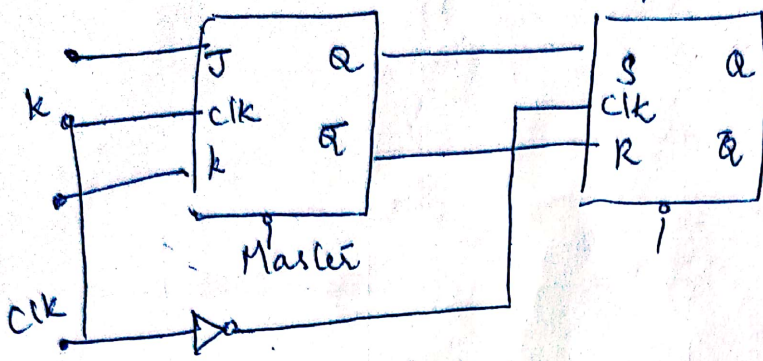
$$Q_{n+1} = \bar{Q}_n T + Q_n \bar{T}$$

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

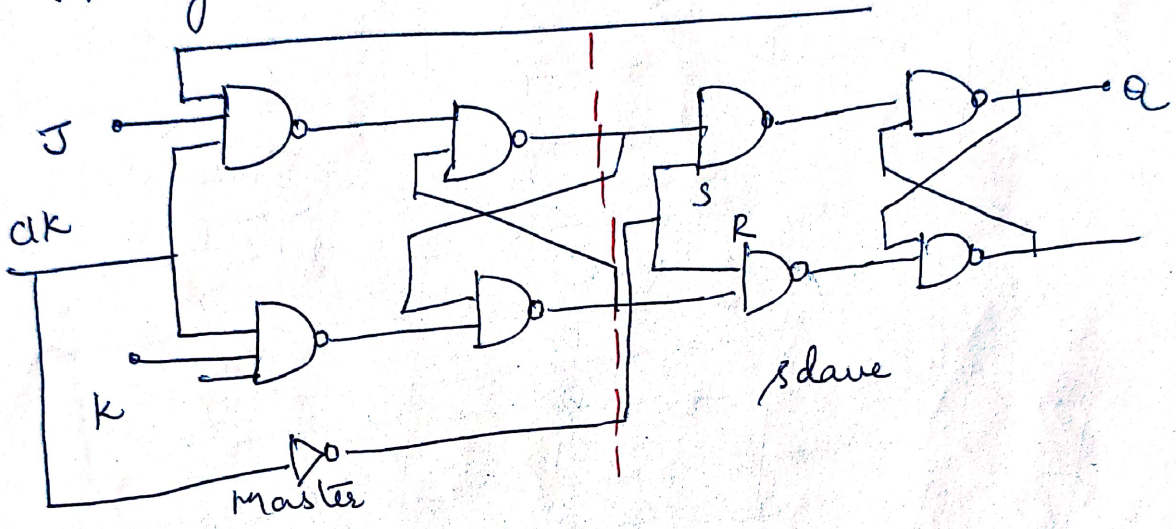
MASTER-SLAVE JK FLIP FLOP:-

* Master slave JK Flip Flop



- Case 1:-
 $J=1, K=0, CLK=H$
 $\rightarrow Q \uparrow, S \uparrow, R, \downarrow$
 $Q=1, \bar{Q} \downarrow$
 Case 2: $J=0, K=0,$
 $Q \downarrow, \bar{Q} \uparrow$
 Case 3: $J=1, K=1$
 FF \rightarrow Set
 Case 4: $J \neq K \rightarrow \downarrow$
 no change

* using NAND Gate:-



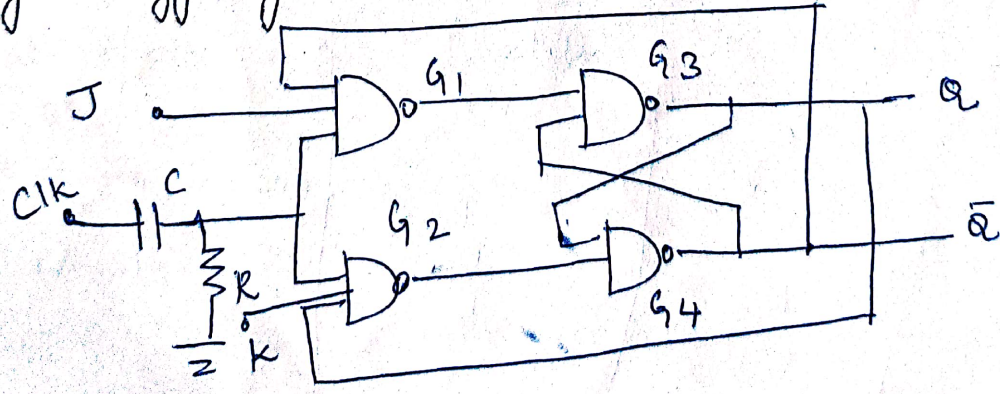
EDGE TRIGGERED JK FLIP FLOP:-

* 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:-



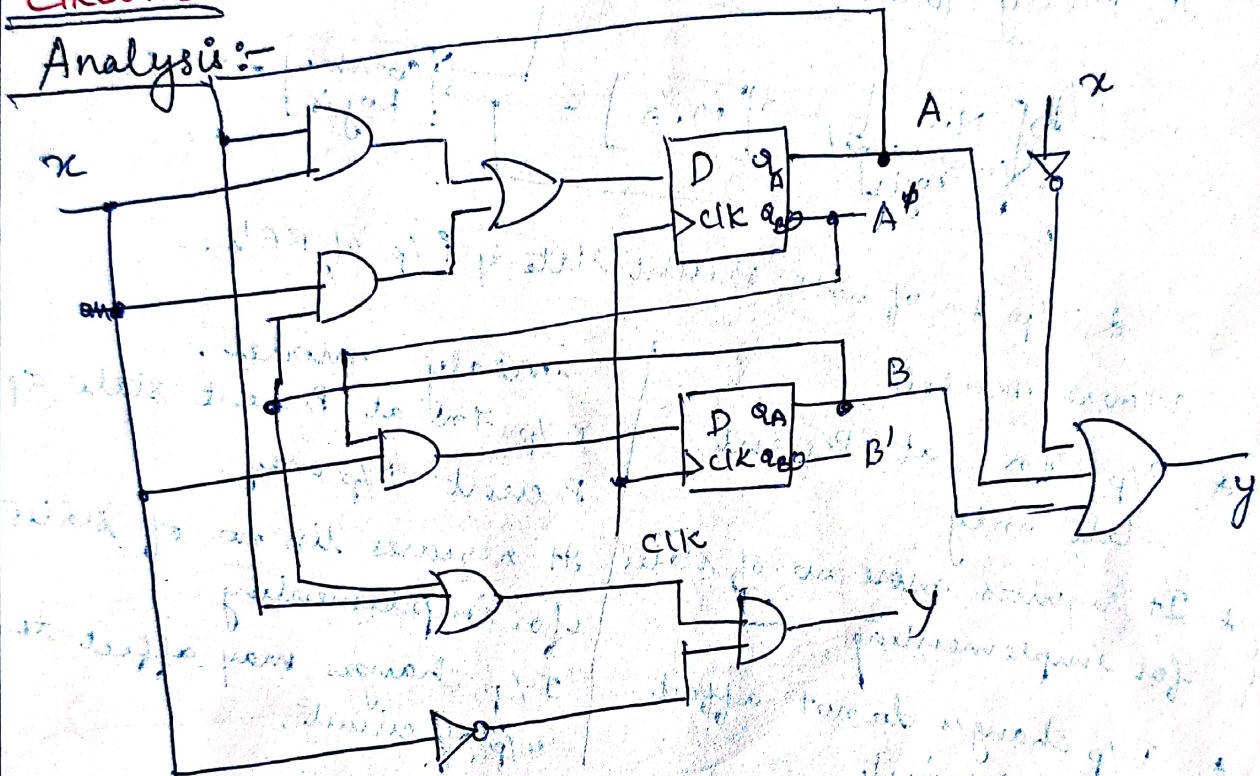
Truth Table :-

Inputs			o/p's		Remarks
clk	J	K	Q_{n+1}	\bar{Q}_{n+1}	
↑	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 :- $y(n) = (A+B)x$

ii) state Table :-

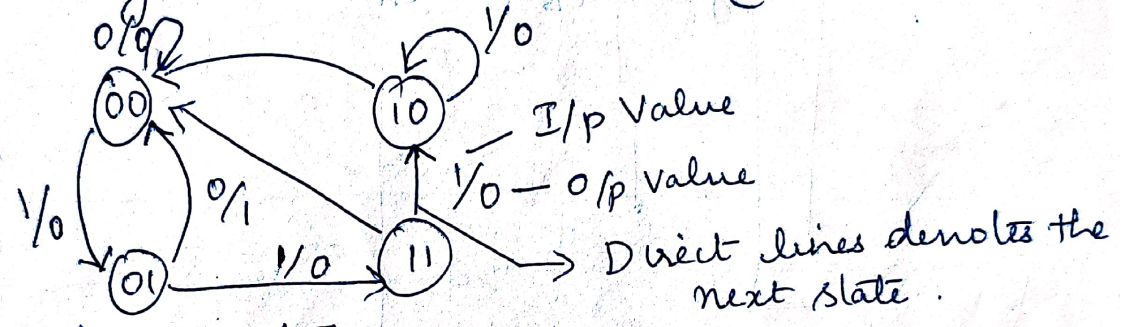
Time sequence of i/p's, o/p's and FF states is called state table.

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

Present state		I/p (x)	Next state		Formula	O/p y
A	B		A	B	$\bar{A}AQB$	$y = (\bar{A} + B)$
0	0	0	0	0	$\bar{0} \cdot 0 \cdot 0 = 0$	0
0	0	1	0	1	$\bar{0} \cdot 0 \cdot 1 = 0$	0
0	1	0	0	0	$\bar{0} \cdot 1 \cdot 0 = 0$	1
0	1	1	1	1	$\bar{0} \cdot 1 \cdot 1 = 0$	0
1	0	0	0	0	$\bar{1} \cdot 0 \cdot 0 = 0$	1
1	0	1	1	0	$\bar{1} \cdot 0 \cdot 1 = 0$	0
1	1	0	0	0	$\bar{1} \cdot 1 \cdot 0 = 0$	1
1	1	1	1	0	$\bar{1} \cdot 1 \cdot 1 = 0$	0

(40)
 $\bar{x}QB + xQA$

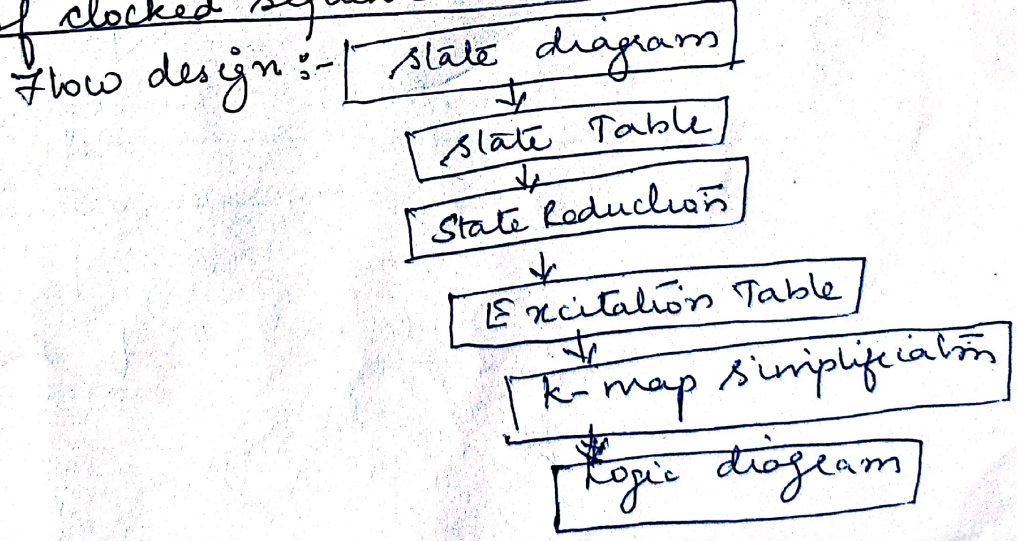
iii) state Diagram :-



iv) FF input equation :-

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

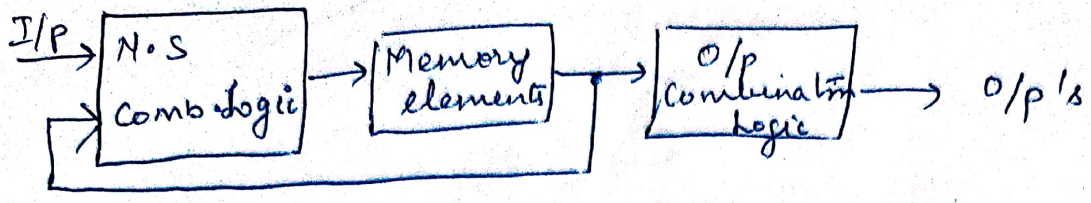
Design of clocked sequential circuits :-



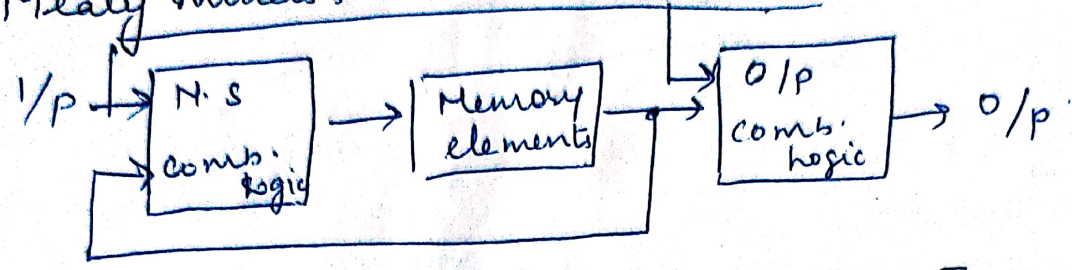
MOORE / MEALY MODELS :-

1) Moore models :-

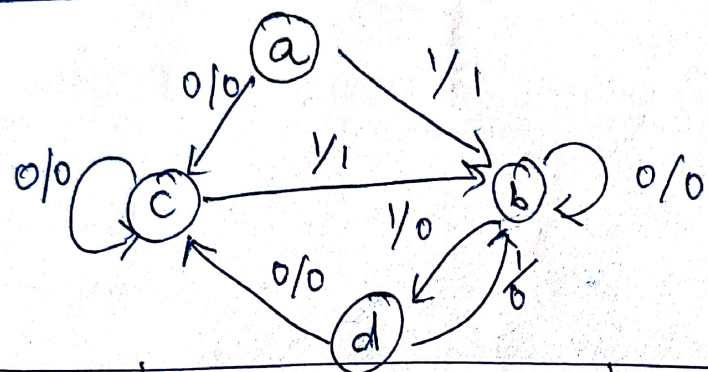
* o/p fn of the present state of flip flops



ii) Mealy models:-



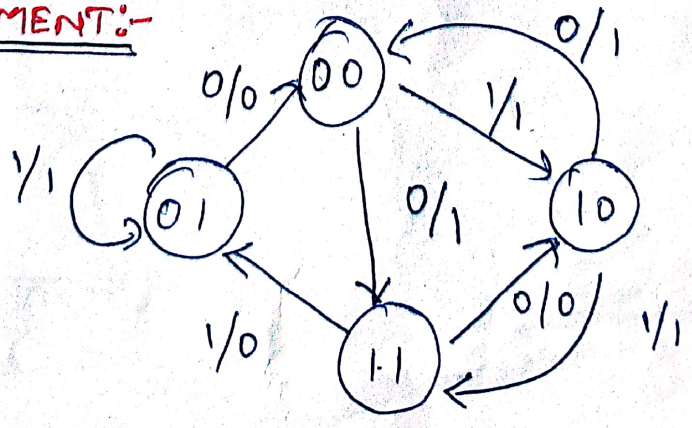
STATE MINIMIZATION:- (State Reduction Techniques)



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

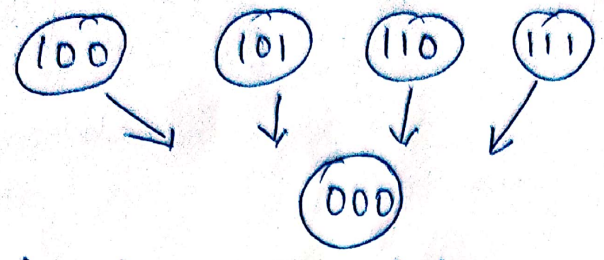
* 'a' and 'c' are same → redundant state
 * Replace 'c' by 'a'.

STATE ASSIGNMENT:-

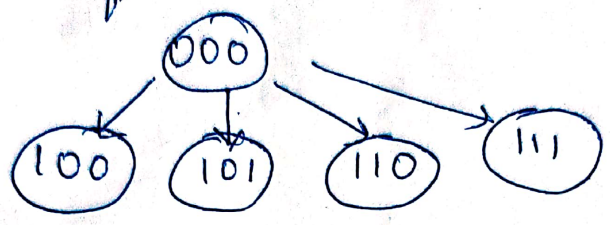


* Two basic Rules:-

R₁ :- 4 states whose next states are same



R₂ :- 000 has four next state :-



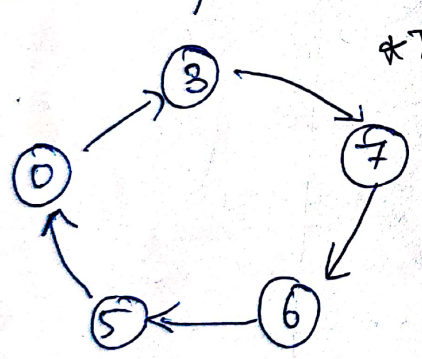
LOCK-OUT CONDITION CIRCUIT IMPLEMENTATION:-

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

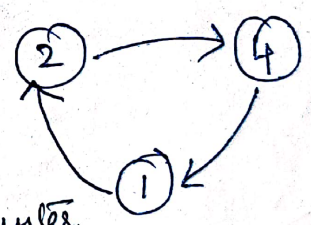
* Lockout condition called self starting counter

Desired sequence

Unused state forming lockout

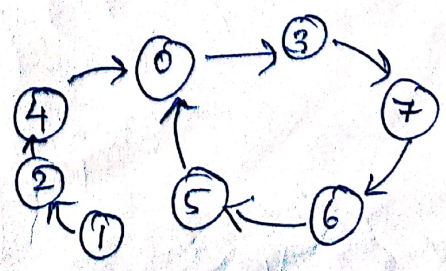
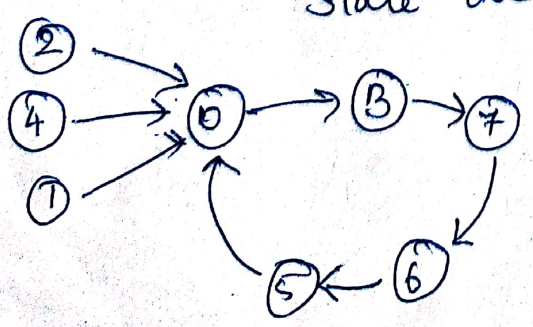


* The counter which never goes in lockout condition is called self starting counter



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

State diagram for removing lockout

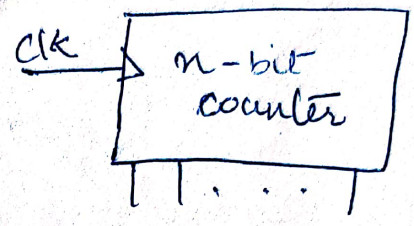
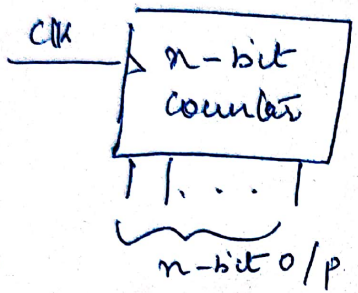


COUNTERS:-

* 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

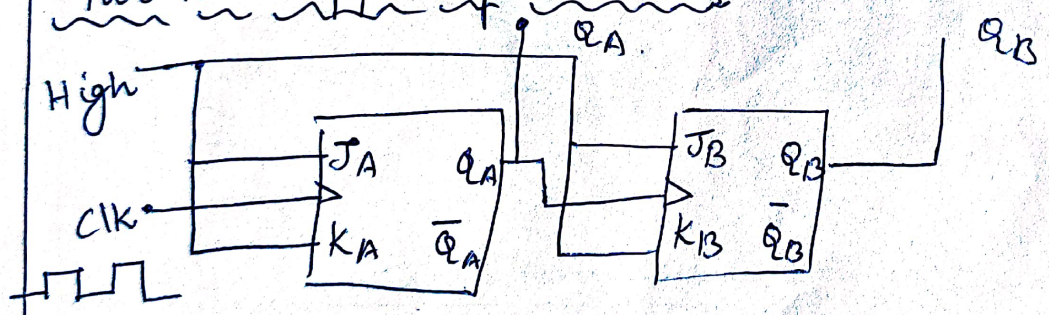


RIPPLE COUNTERS:- (or Asynchronous) :-

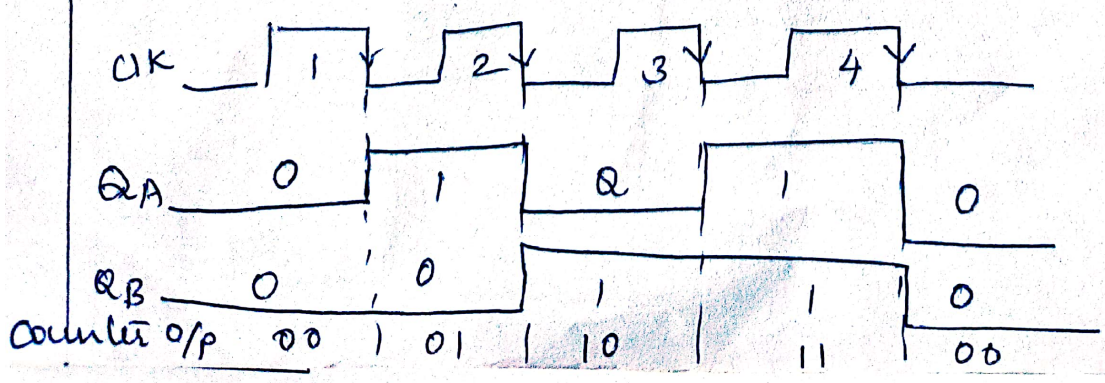
* Binary ripple counter consist of a series connection of complementing FF's with o/p of each FF connected to the clock i/p of next higher order FF.

1) Ripple up counter:-

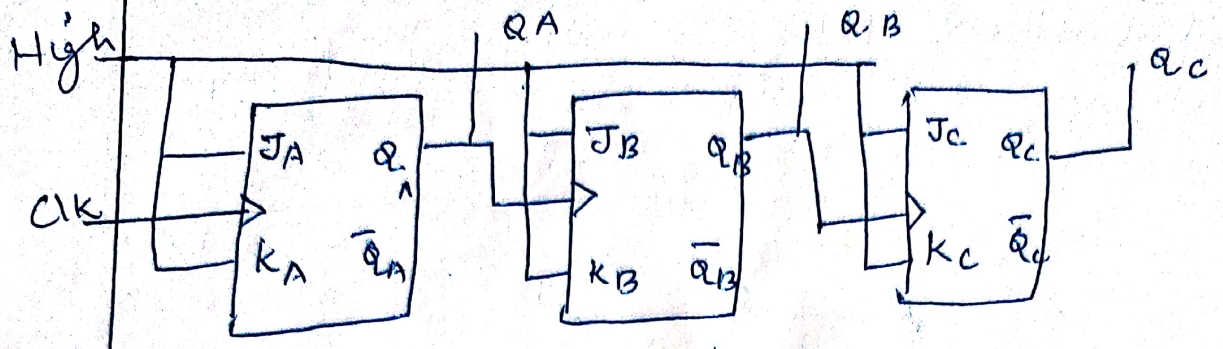
Two Bit ripple up counter:-



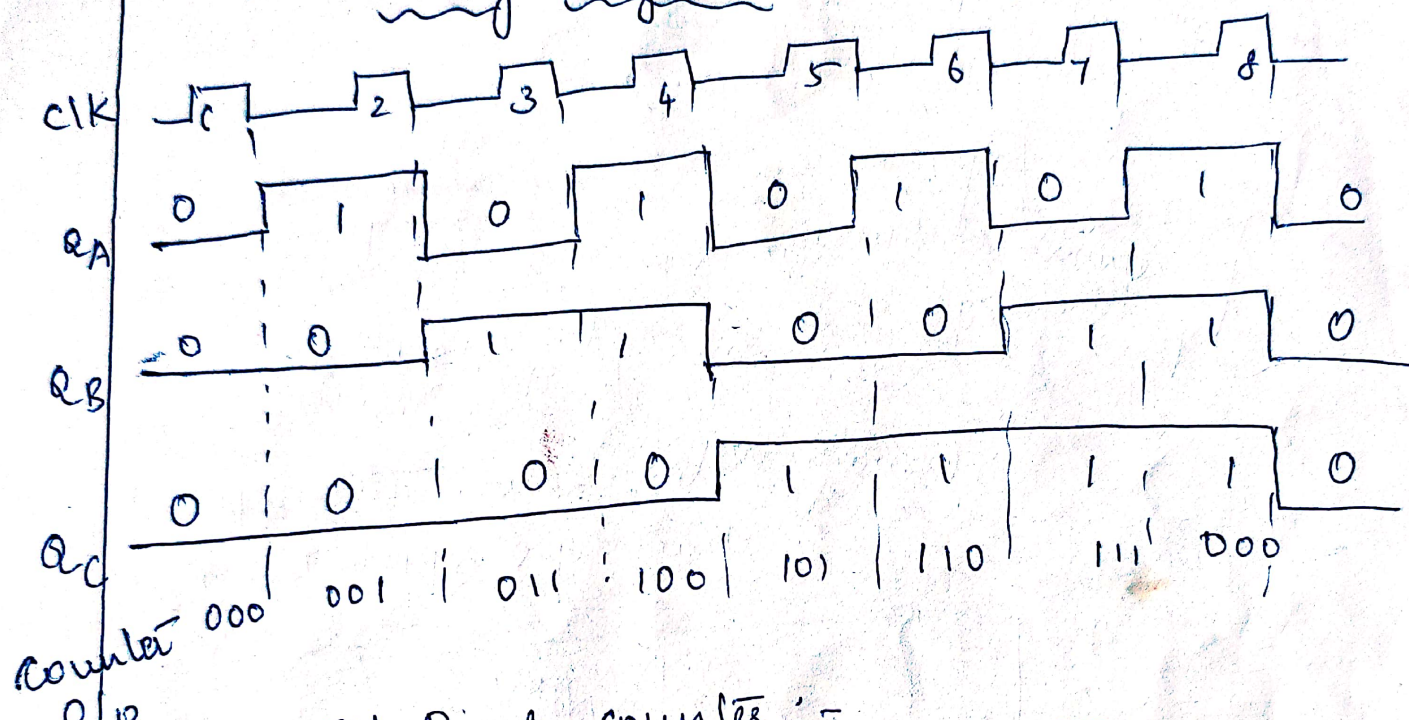
Timing diagram:-



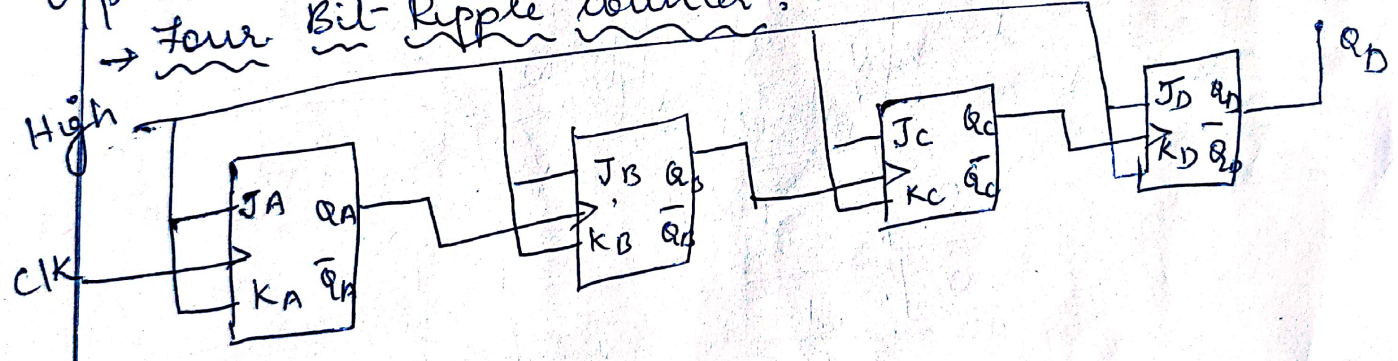
→ Three Bit Ripple counter :-



Timing diagram :-



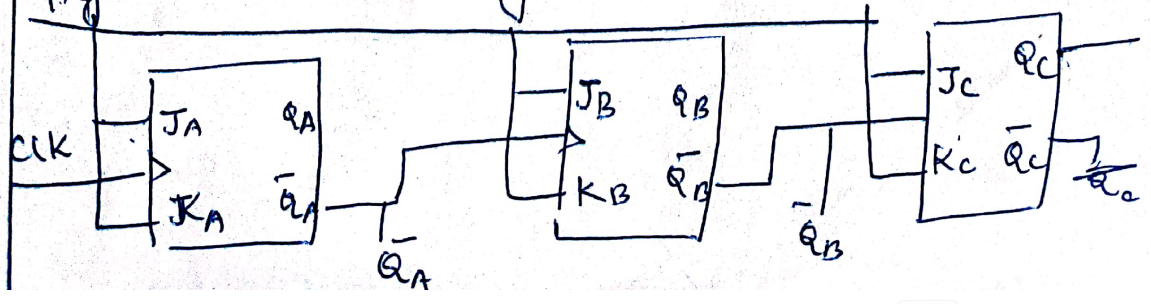
→ Four Bit-Ripple counter :-

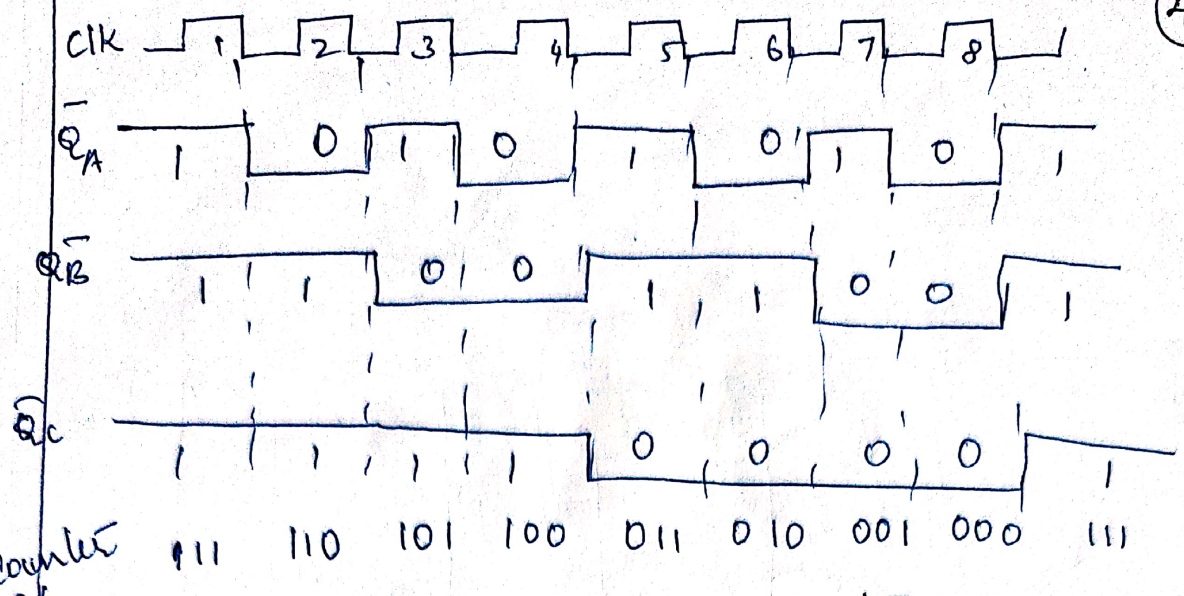


(ii) Asynchronous Down counter :-

* 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.

- * $up/down = 0 \rightarrow$ count down
- * $up/down = 1 \rightarrow$ count up.

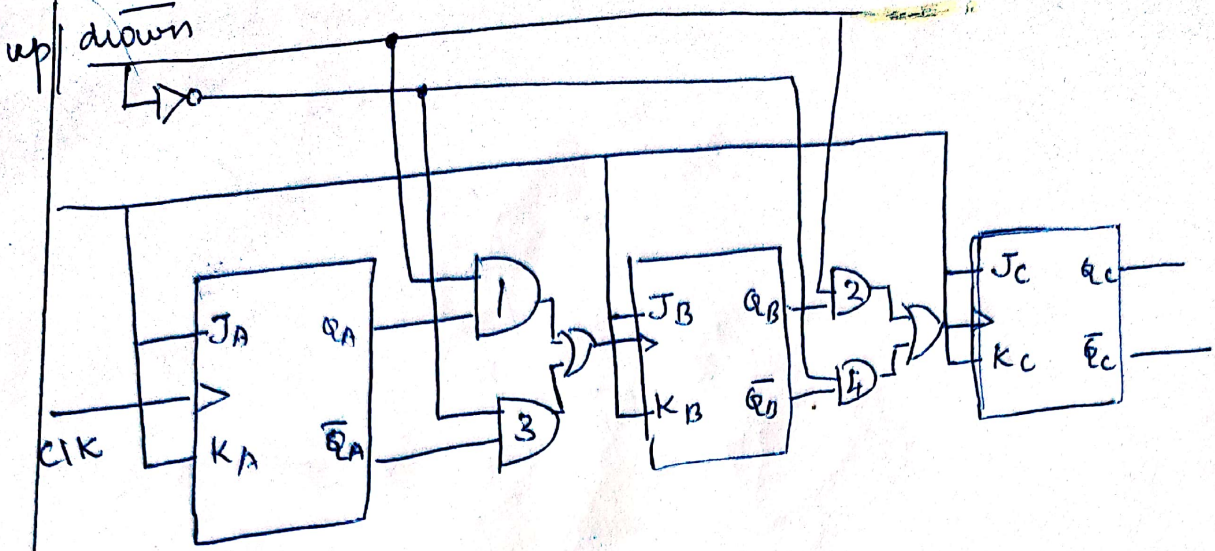
Truth Table :-

	I/P		O/P	
	up/down	Q	\bar{Q}	Y
Down counter up/down = 0	0	0	0	0
	0	0	1	1
	0	1	0	0
	0	1	1	1
up counter up/down = 1	1	0	0	0
	1	0	1	0
	1	1	0	1
	1	1	1	1

Annotations: $Y = \bar{Q}$ for down counter (rows 1-4), $Y = Q$ for up counter (rows 5-8).

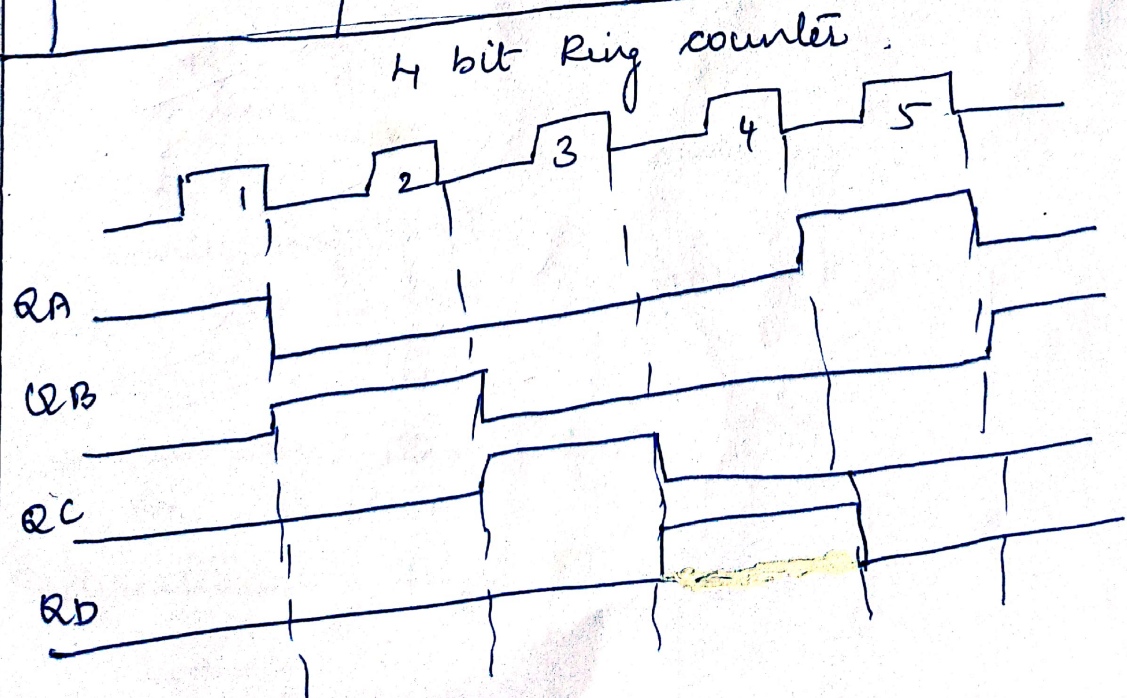
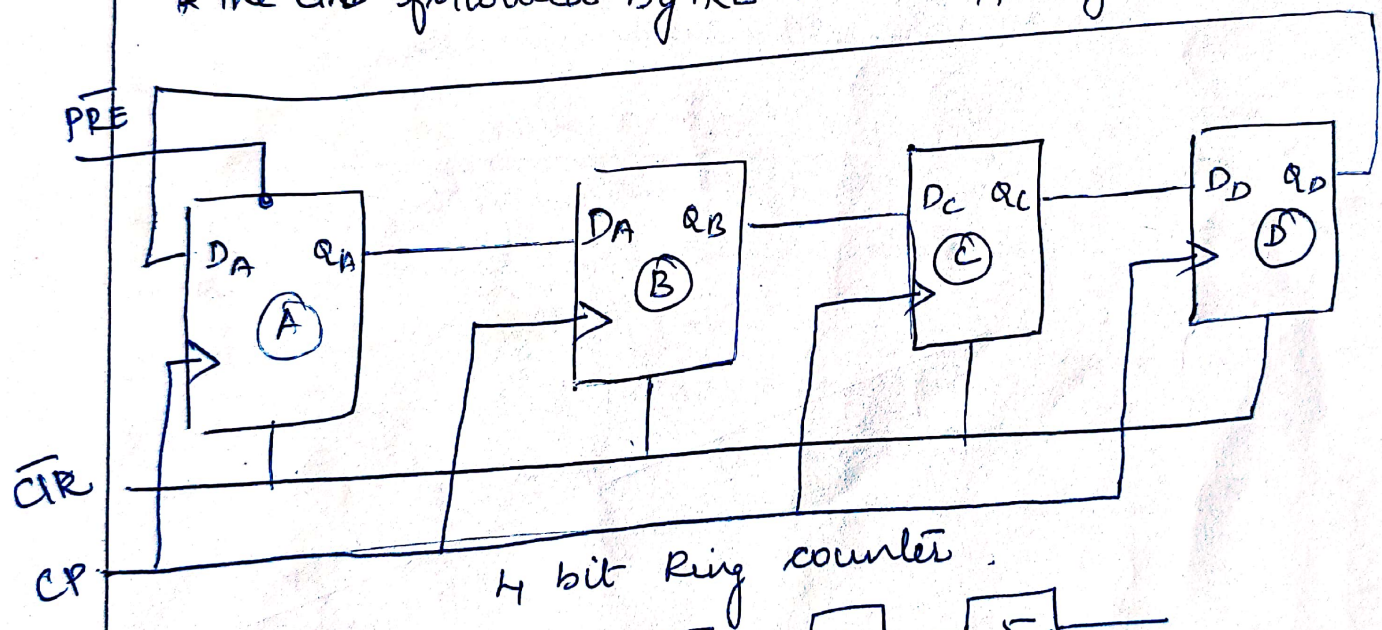
up/down	00	01	11	10
0	0	1	1	0
1	0	0	1	1

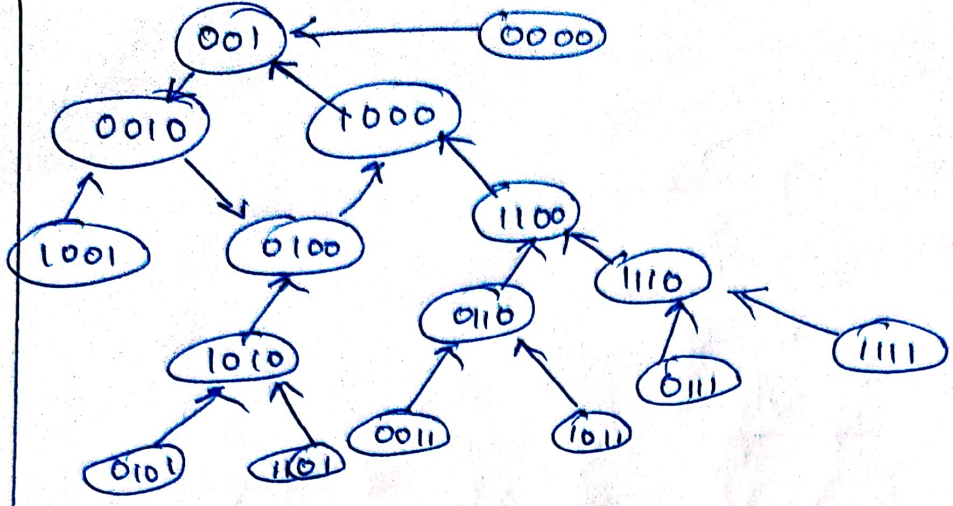
$$Y = up/down \bar{Q} + up/down Q$$



RING COUNTERS :-

- * 4 bit Ring counter \rightarrow 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.





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 ~~to~~ 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
- n // 2 load control to enable all the transfers & n i/p lines
- n // 2 o/p lines.

unidirectional shift Register :-
One direction.

Bidirectional shift Register :-
both direction.

universal shift Register :-

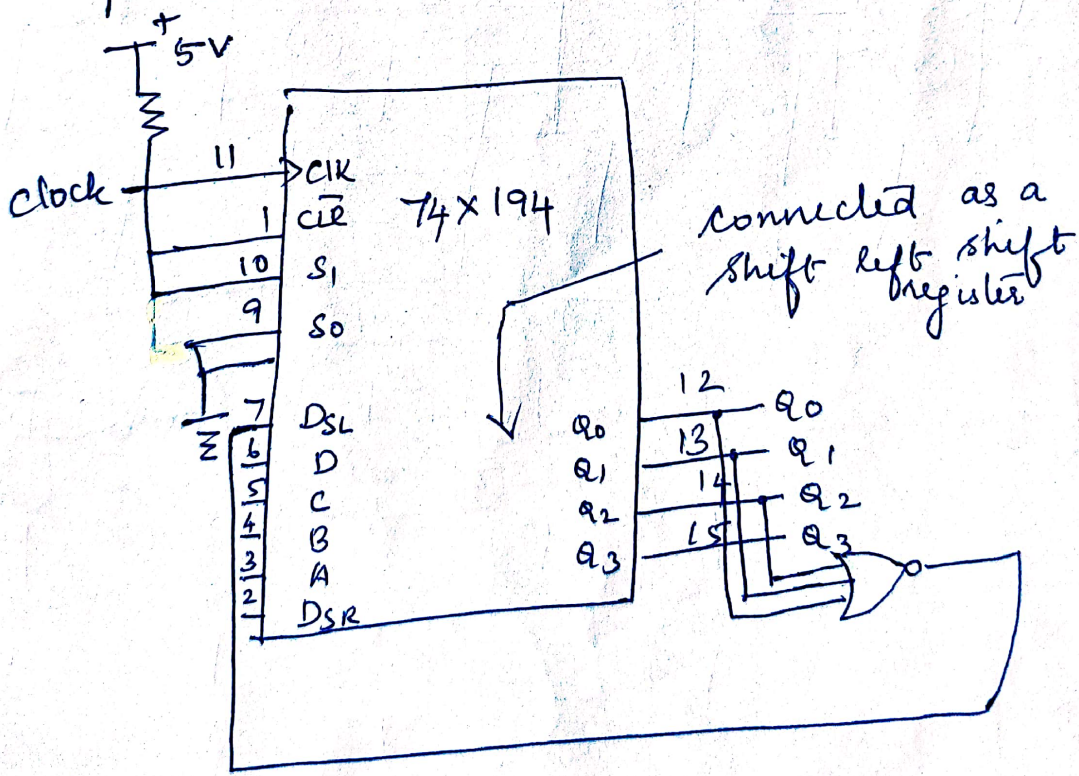
Clock Pulse	Q _A	Q _B	Q _C	Q _D
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1
4	1	0	0	0

* 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.

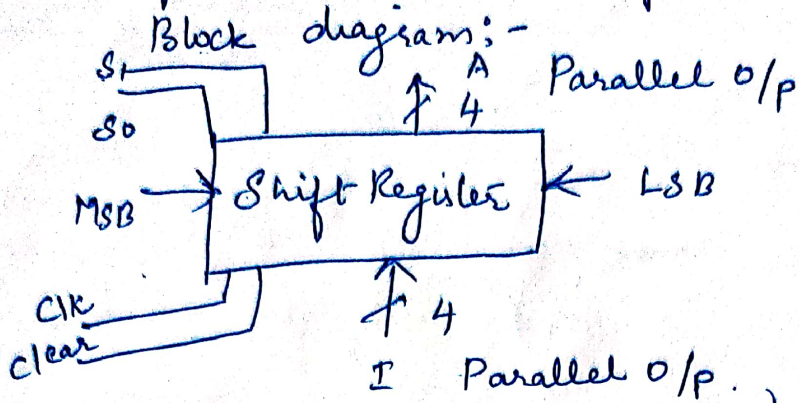
Self correcting counters:-

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

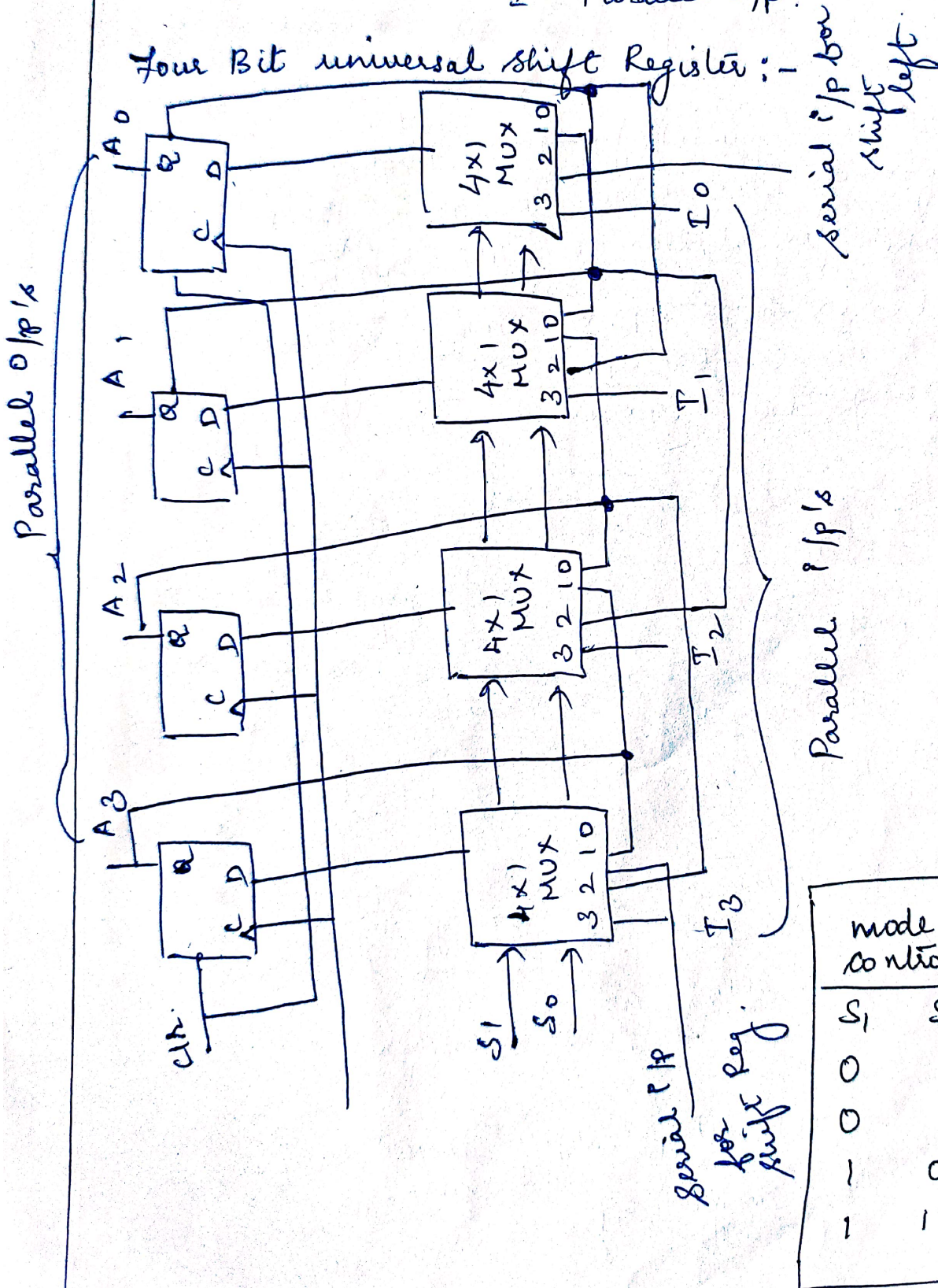


state diagram:-

* Both shifts and 112 load capabilities



Four Bit universal shift Register:-



mode control		Reg. operation
S ₁	S ₀	no change
0	0	Shift Reg
1	0	Shift left
1	1	112 load

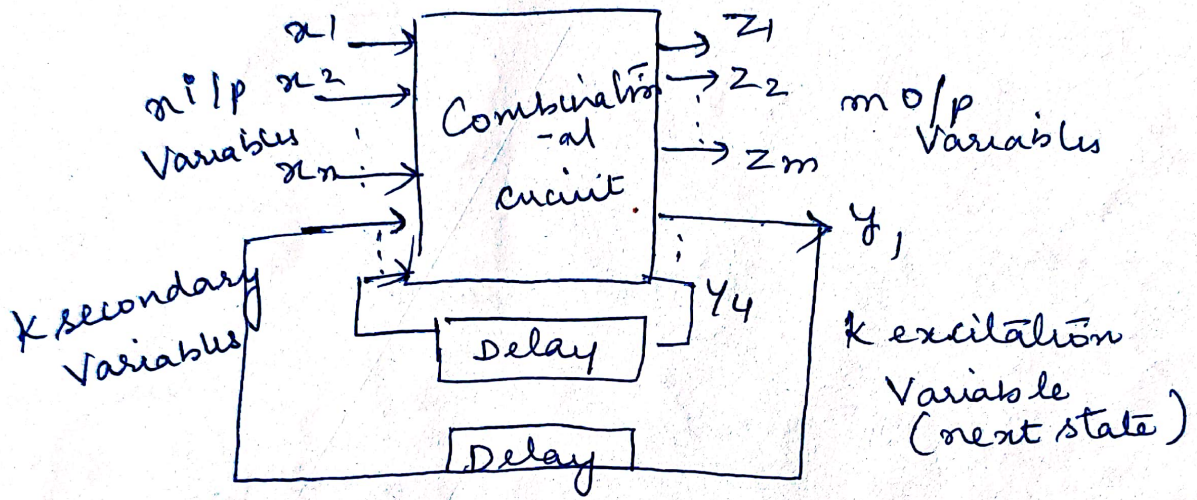
(49)

UNIT-IV ASYNCHRONOUS SEQUENTIAL LOGIC
CIRCUITS :-

STABLE AND UNSTABLE STATES :-

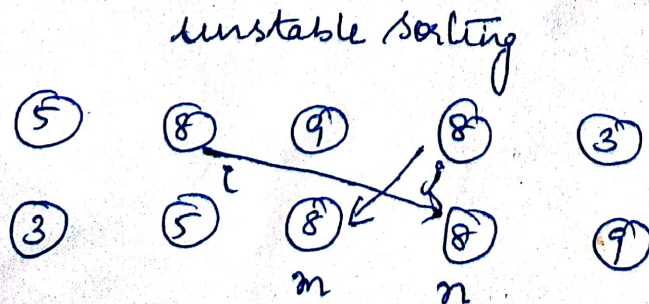
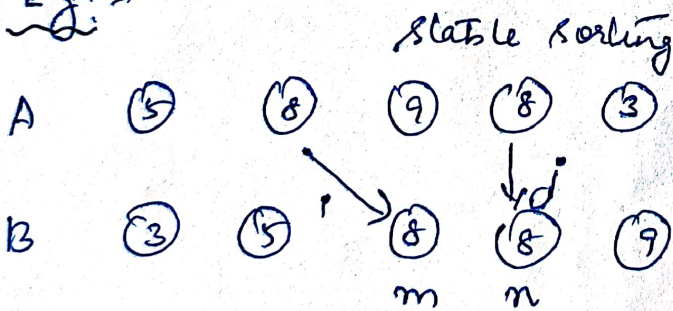
* A sequential circuit is specified by a time sequence of i/p, o/p's and internal states.

* Asynchronous sequential circuits do not use clock pulses



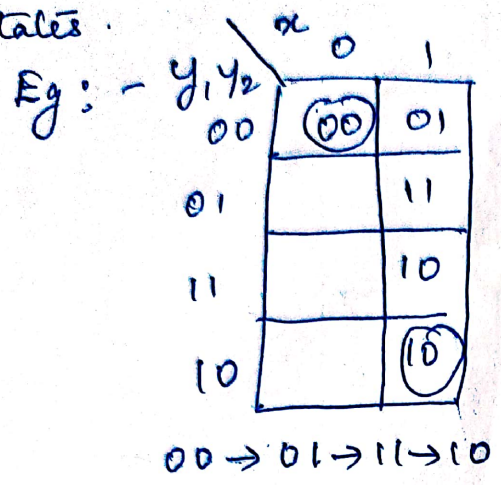
* System is said to be stable, if its o/p is under control. Otherwise, it is said to be unstable.

Ex:-



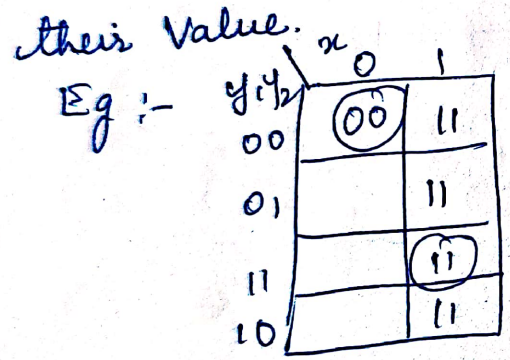
CYCLES AND RACES:-

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



Races:-

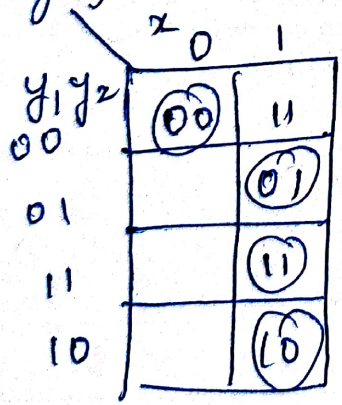
* Two or more binary state variable change their value.



Possible Transitions

- 00 → 11
- 00 → 01 (y_2 faster) → 11
- 00 → 10 (y_1 faster) → 11

Eg for non-critical Races.



Possible transition

- 00 → 11
- 00 → 01
- 00 → 10

Free state assignment, two commonly

used techniques:-

- (i) shared row state assignment.
- (ii) one hot state assignment.

i) Shared Row state Assignment.

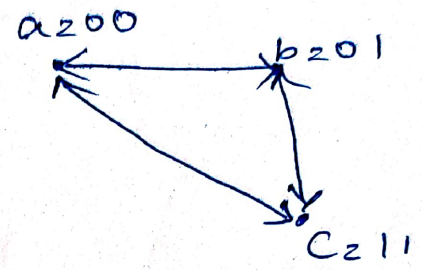
* Two binary values are said to be adjacent if they differ in only one variable.

* 00 and 01 are adjacent to each other.

3 row Flow Table.

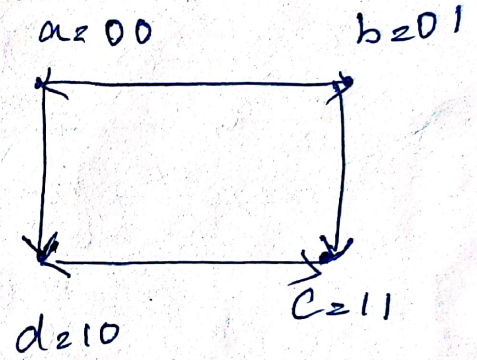
	00	01	11	10
a	(a)	b	c	(a)
b	a	(b)	(b)	(a)
c	a	(c)	(c)	(c)

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

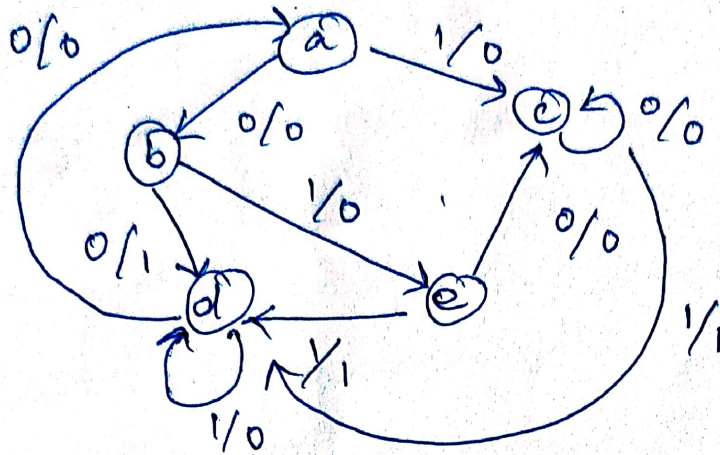
* only one variable is active (or) 'hot' for each row in original flow table.

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.

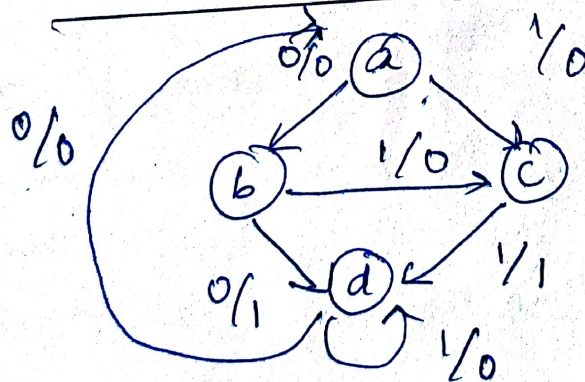
* no. of required F-F's and logic gates.



State Table:-

Present state	Next state		O/p	
	x ₂₀	x ₂₁	x ₂₀	x ₂₁
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 ₂₀	x ₂₁	x ₂₀	x ₂₁
a	b	c	0	0
b	d	c	1	0
c	c	d	0	1
d	a	d	0	0

RACE FREE ASSIGNMENTS:-

53

* 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.

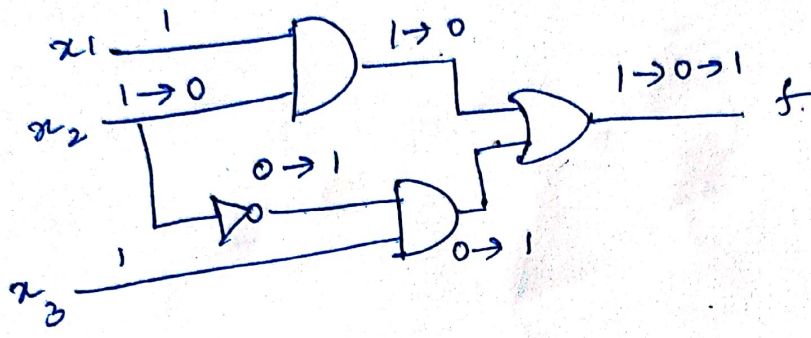
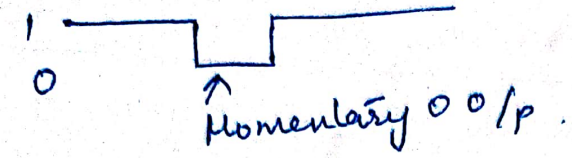
- i) static Hazards
- ii) Dynamic Hazards.

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 variable.

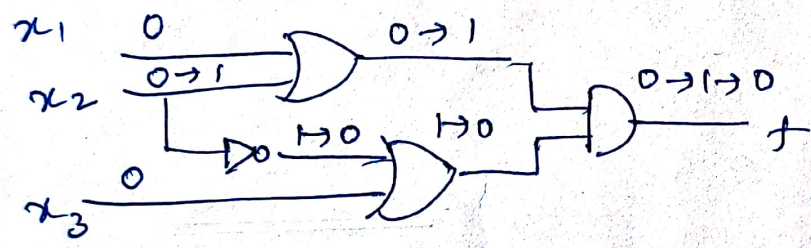
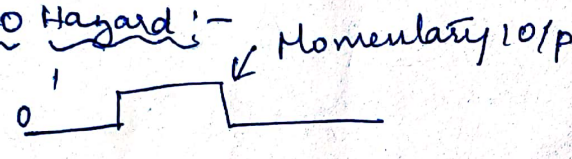
- i) static 1 hazards
- ii) static 0 hazards

1) static -1 Hazard:-



* Transition b/w i/p states $x_1, x_2, x_3 = 111, x_1, x_2, x_3 = 101$

ii) static -0 Hazard:-



* i/p states $x_1, x_2, x_3 = 000, x_1, x_2, x_3 = 010$

Elimination static Hazards:-

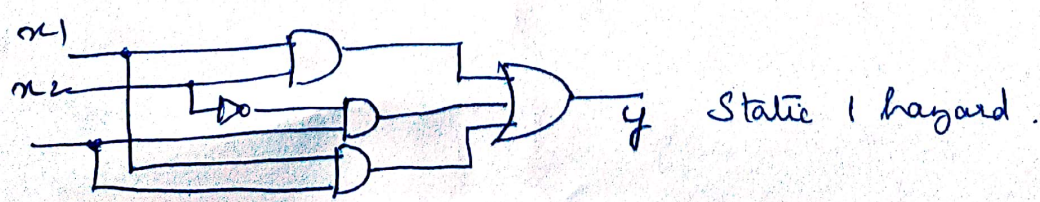
static -1 Hazard:-

* Removed by covering adjacent cells with redundant grouping.

$$y = x_1 x_2 + \bar{x}_2 x_3$$

k-map:-

		$x_1 x_2$			
		00	01	11	10
x_3	0	0	0	1	0
	1	1	0	1	1

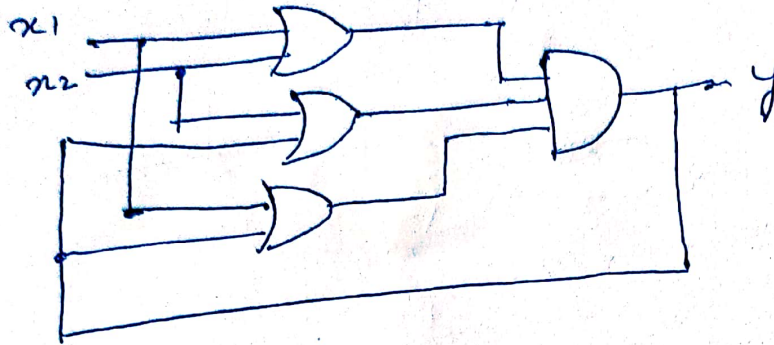


Elimination of static 0 hazard :-

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

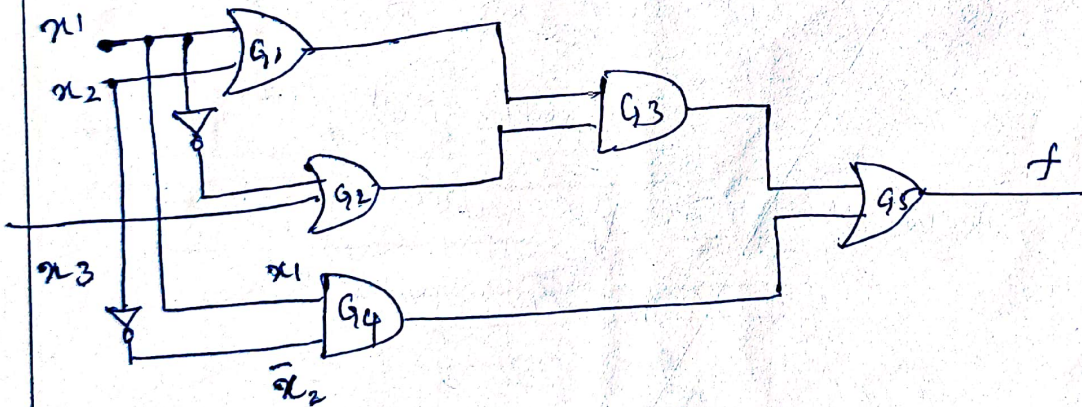
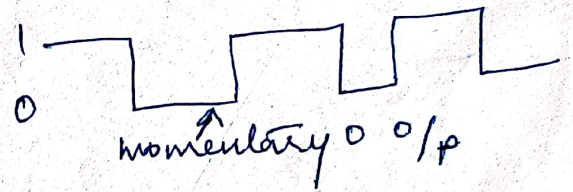
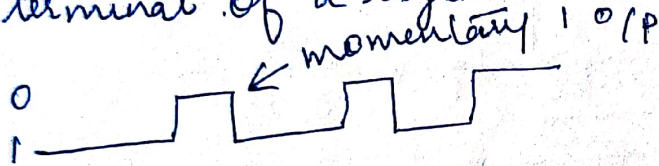
K-map :-

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



ii) Dynamic Hazards :-

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



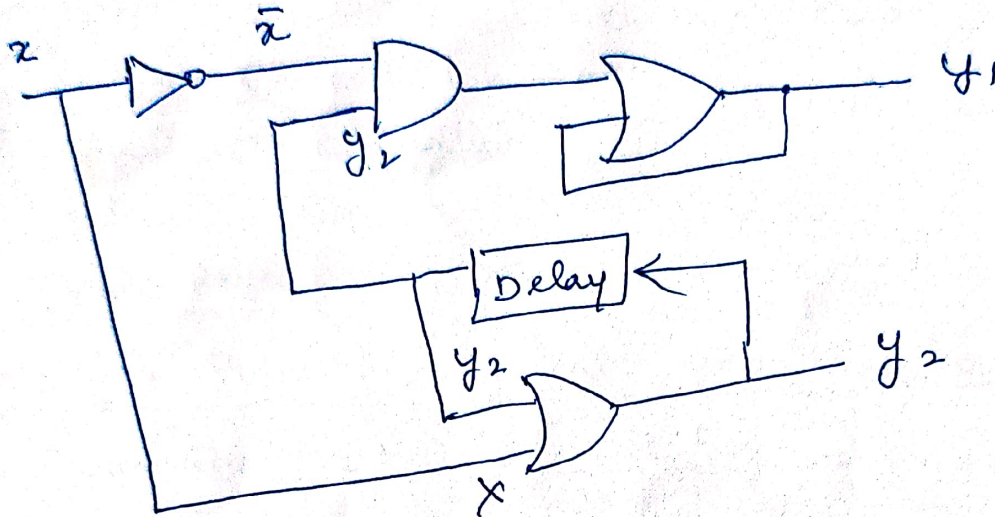
* change of x_1 from 0 to 1, N/w o/p will appear as $0 \rightarrow 1 \rightarrow 0 \rightarrow 1$

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
A	0	1
B	(A)	(B)
C	(C)	(C)

Transition Table :-

P.S. y1 y2	N.S	
	I/p	x
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 (57)

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.

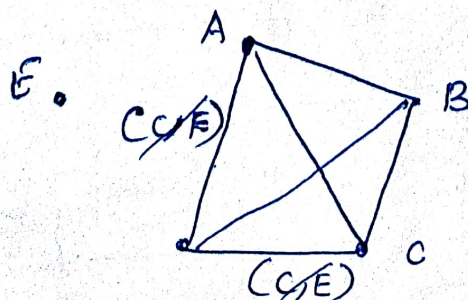
Design of Fundamental mode sequential circuits :-

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 :-



DESIGN OF HAZARD FREE CIRCUITS:-

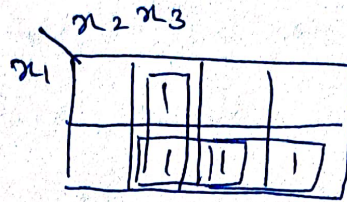
* The unwanted switching transients (glitches) that may appear at the o/p of a circuit are called Hazards.

- * Two Types :- 1) static Hazards
- 2) Dynamic Hazards.

Eliminating a Hazard:-

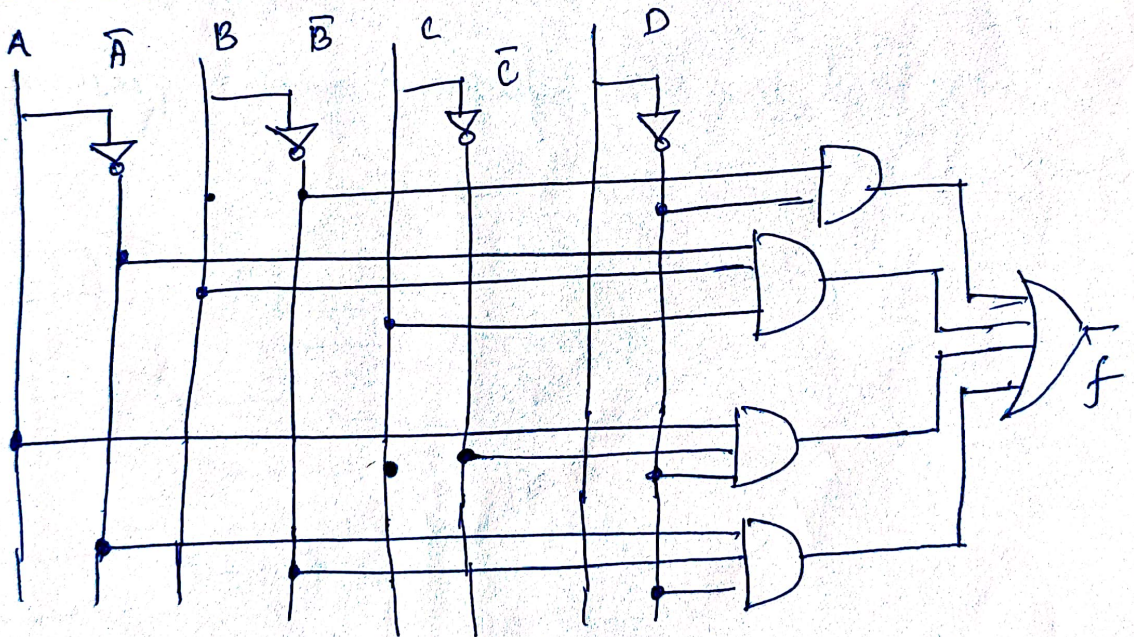


$$y = x_1 x_2 + \bar{x}_2 x_3$$

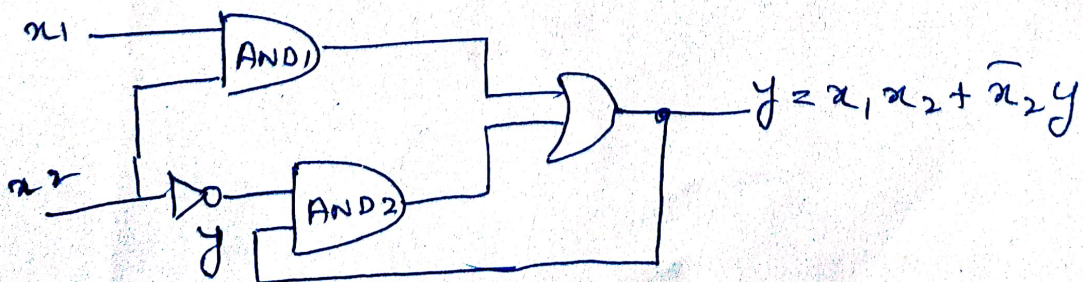


$$y = x_1 x_2 + \bar{x}_2 x_3 + x_1 x_3$$

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



Hazards in sequential circuits:-



Transition Table:-

59

	$x_1 x_2$	11	10
0	0	1	0
1	1	1	1

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

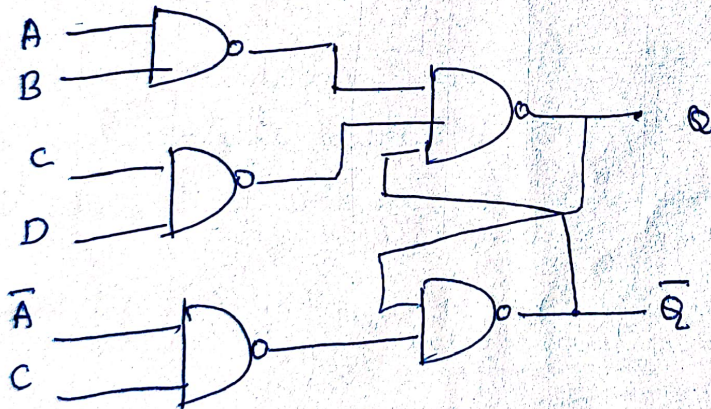
Essential Hazards:-

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

Eliminating Essential Hazards:-

* 0 signal applied to S or R P/p's of NOR latch have no effect on state.

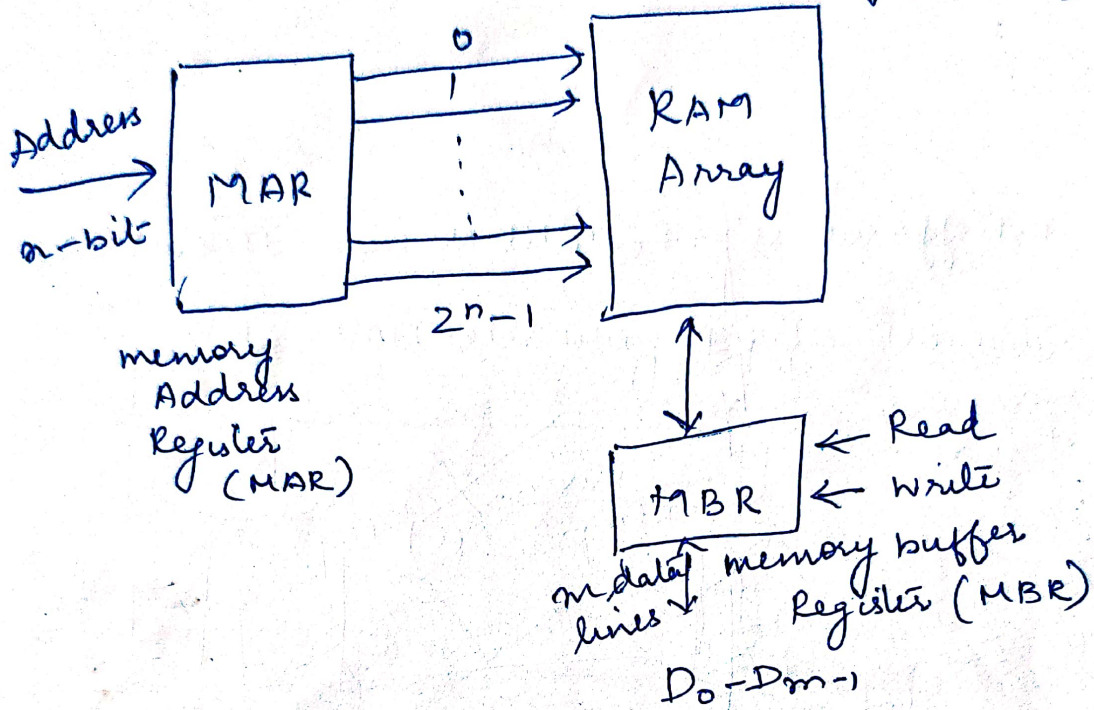
* Momentary signal 1 applied to S and R of NAND latch.



RAM AND ROM:-

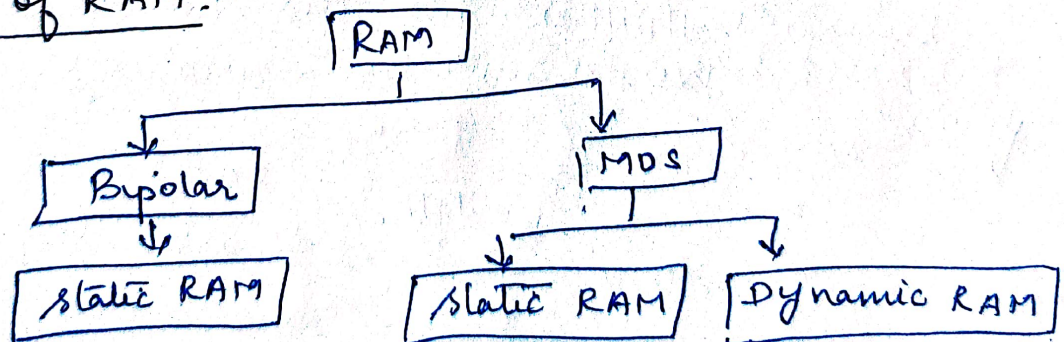
RAM Organization:-

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



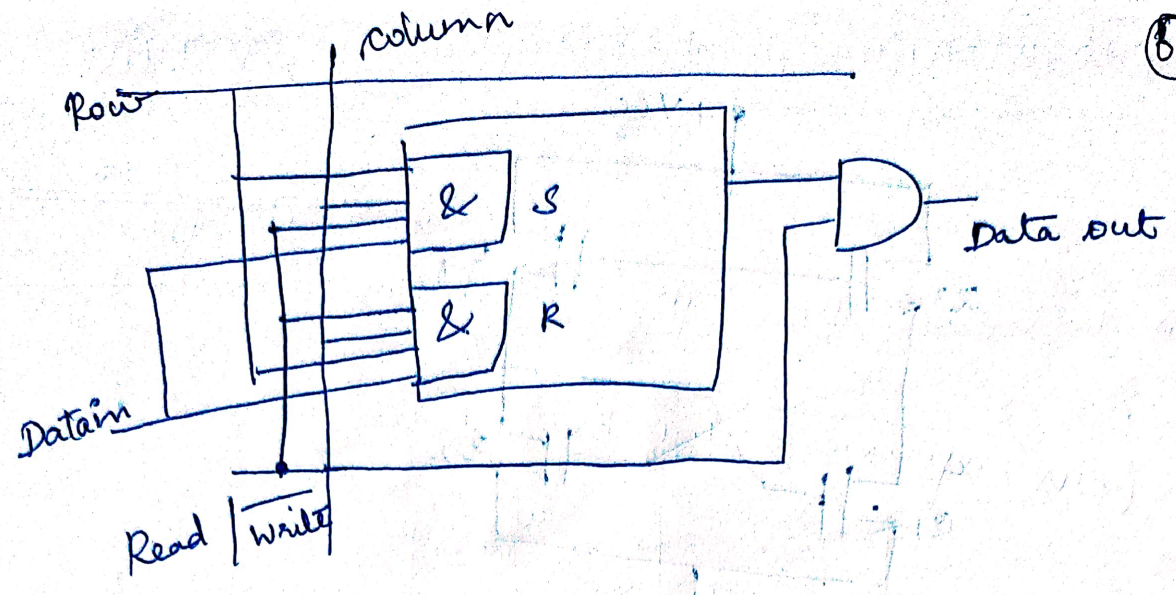
* When a Read is asserted, the data from the memory location, selected by the n-bit address and it is placed in Memory Buffer Register (MBR).

Types of RAM:-



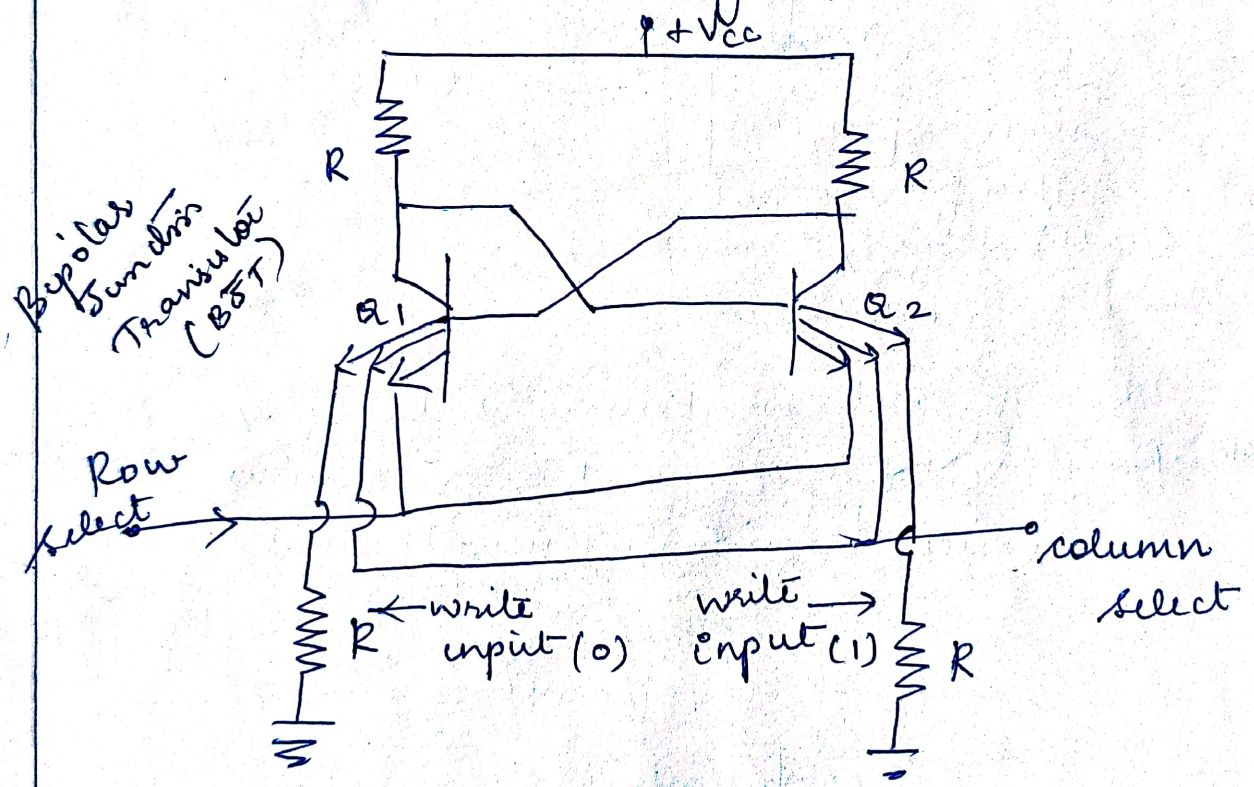
Static RAM cell :-

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



Resetting the FF for 0, when READ/WRITE line is low
 * READ/WRITE line is HIGH, FF is unaffected.

Static RAM cell using Bipolar Transistor.



Bipolar Transistor (BJT)

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

MOS Static RAM cell:-

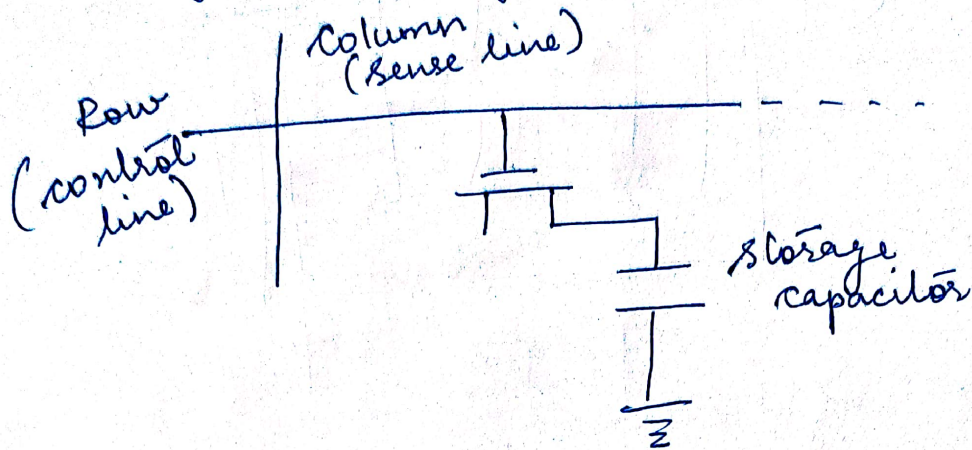
* Q_1 and Q_2 act as switch, Q_3 and Q_4 as active load resistors.

2. Dynamic RAM (DRAM):-

(62)³

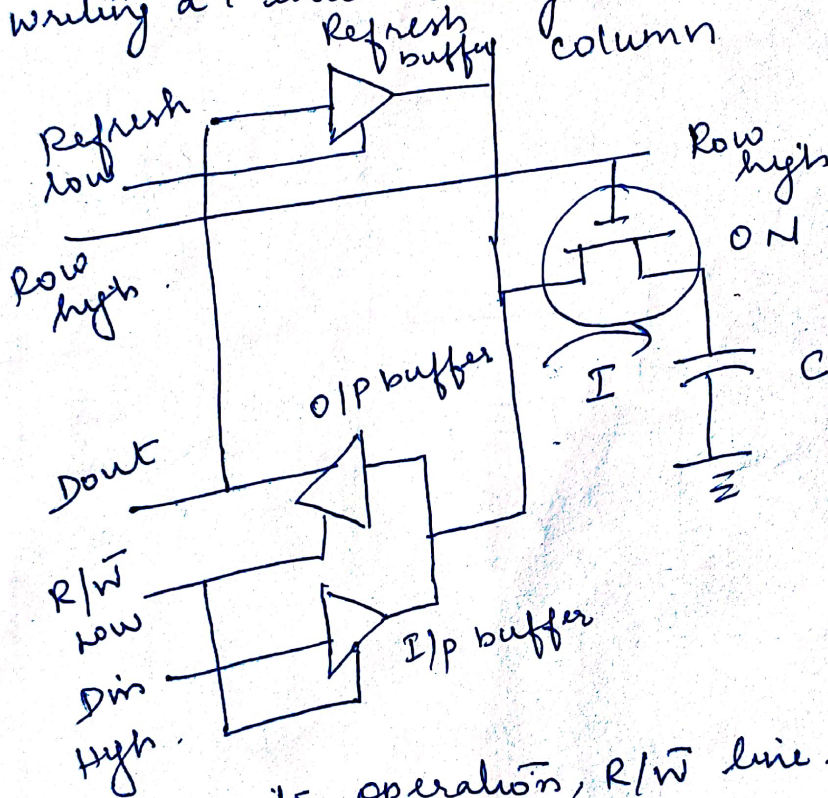
* DRAM is lowest cost, high density random access memory available.

* Memory size ranging from 16 to 256 mega bytes.



write operation:-

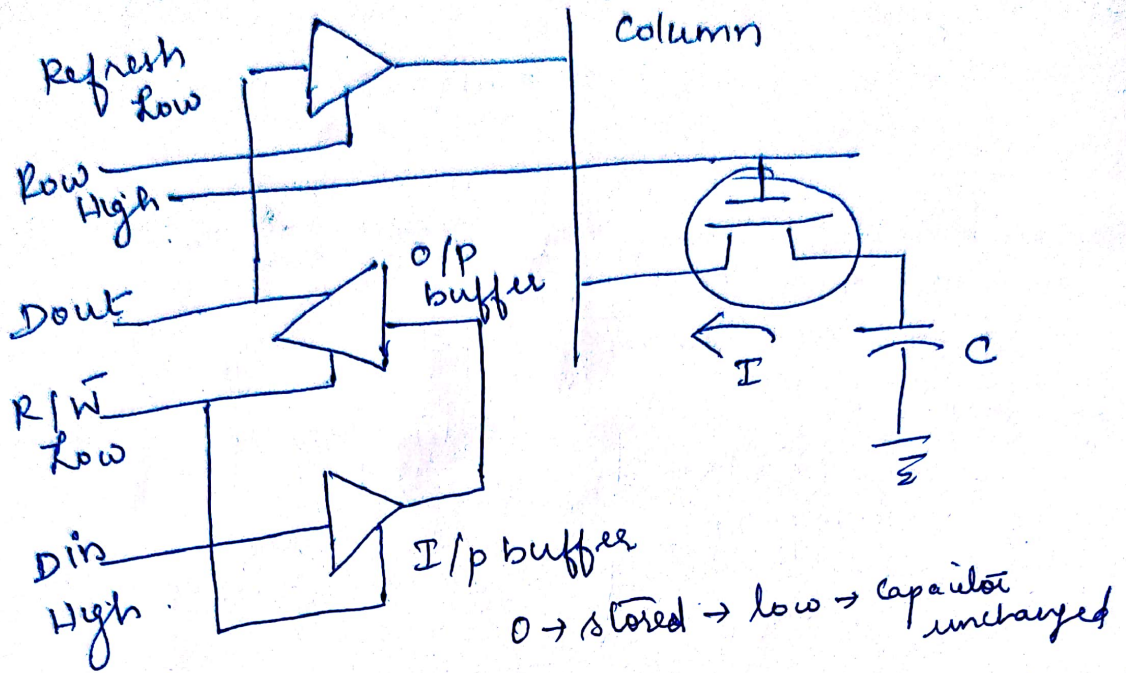
a) writing a 1 into memory cell:



- * To enable write operation, R/w line is made low enables input buffer and disables O/p buffers.
- * To write 1 into cell, Din line 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.

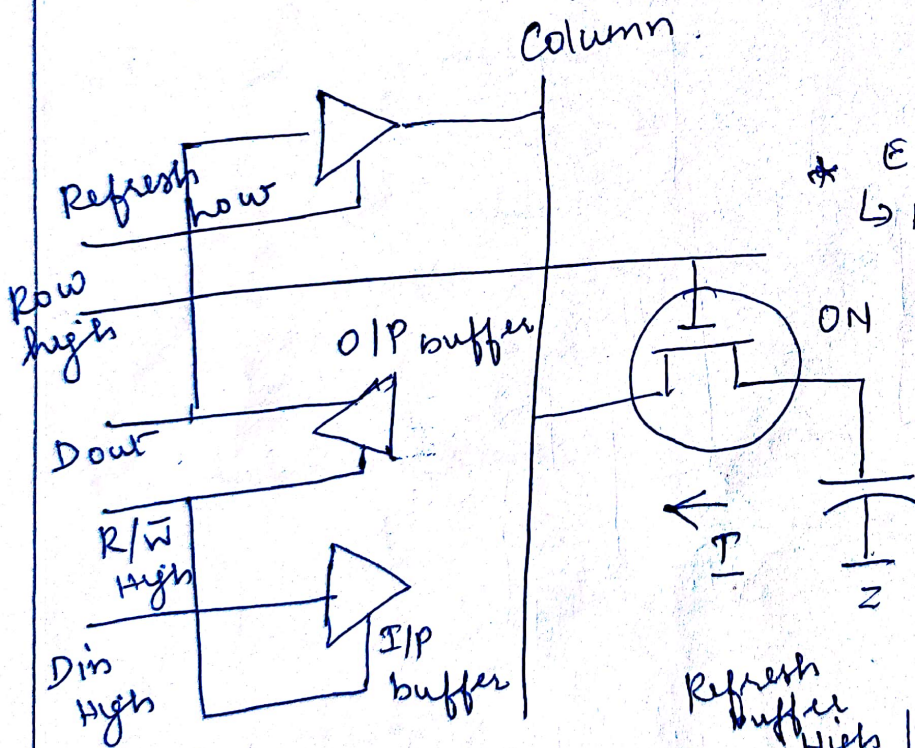
(63) 4



Read operation :-

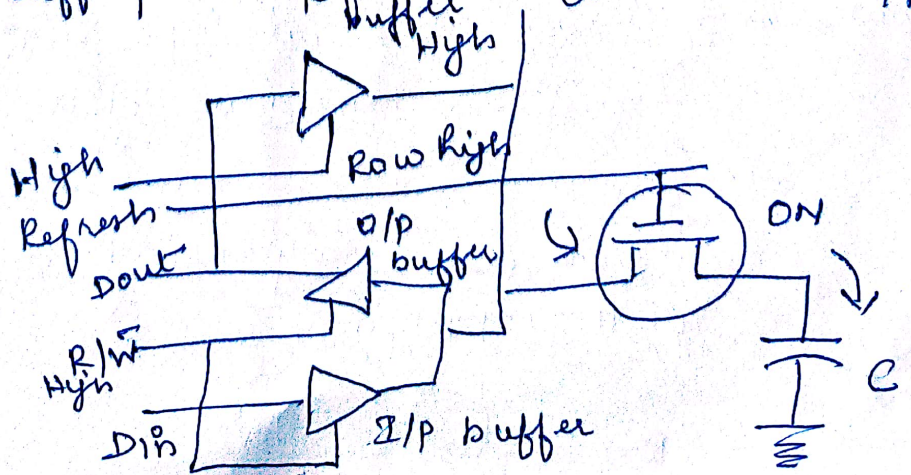
Reading a 1 from memory

Refreshing a store



* Enable R/W line
 \hookrightarrow Row line and Refresh line are made HIGH.

* R/W is HIGH
C O/p buffer is enabled and stored data bit is applied Column. to I/p.



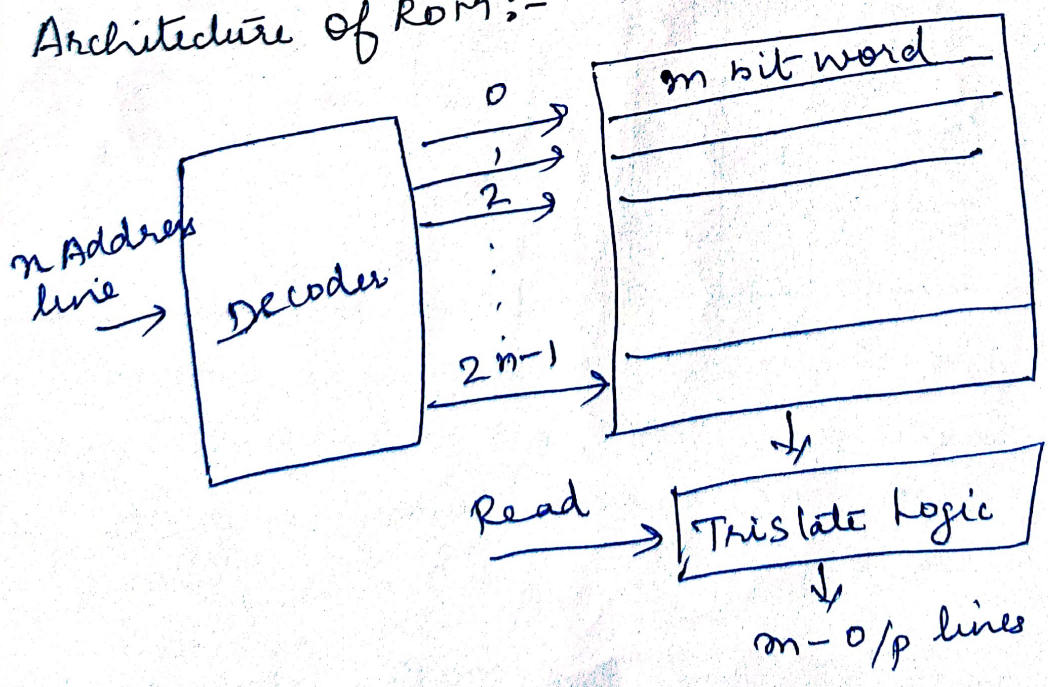
Advantages of RAM :-

- * Fast operating speed \rightarrow Time can be low as 15ns
- * non-destructive Readout \rightarrow 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..

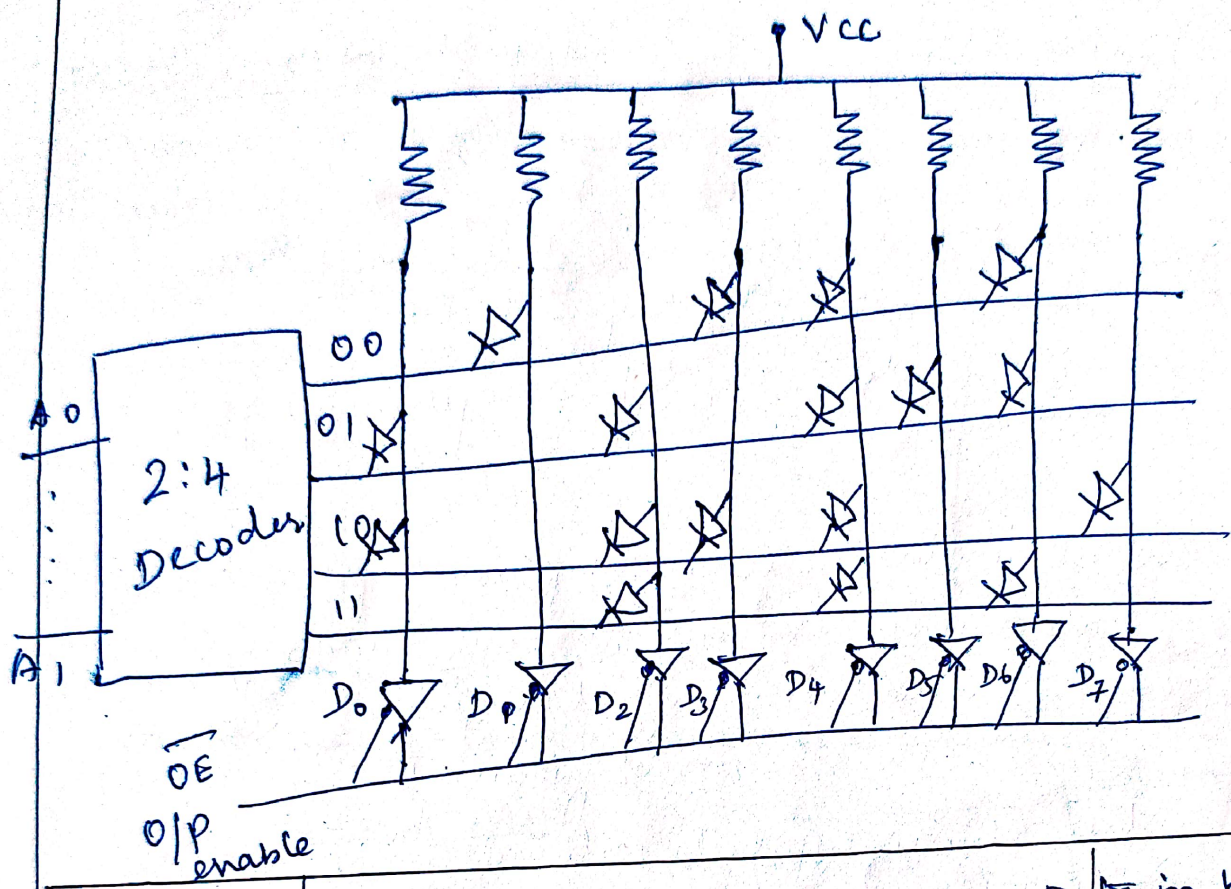
ROM :-

- * 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 :-

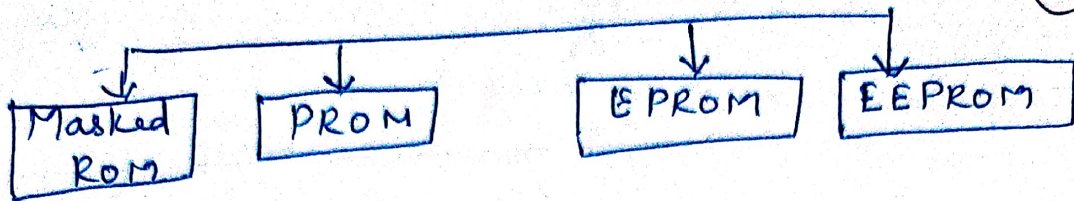


* n address lines and m o/p lines,
 * no. of words 2^n and no. of bits m.
Simple four Byte diode ROM:-



Address in binary	Binary Data								Data in hex
	D ₀	D ₁	D ₂	D ₃	D ₄	D ₅	D ₆	D ₇	
00	1	0	1	0	0	1	0	1	A5
01	0	1	0	1	0	0	0	1	51
10	0	1	0	0	0	1	1	0	46
11	1	1	0	1	0	1	0	1	D5

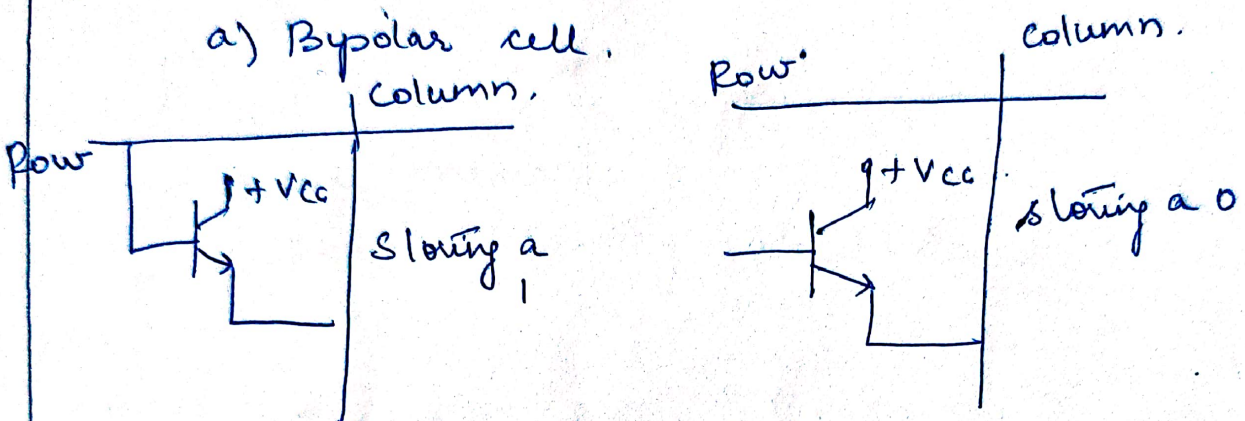
* ROM may be programmed into two different ways. The 1st is called MASK programming is done by the manufacturer during last fabrication process. The second type is Programmable Read only memory (PROM).



Masked ROM :-

* Permanently Programmed during the manufacturing Process.

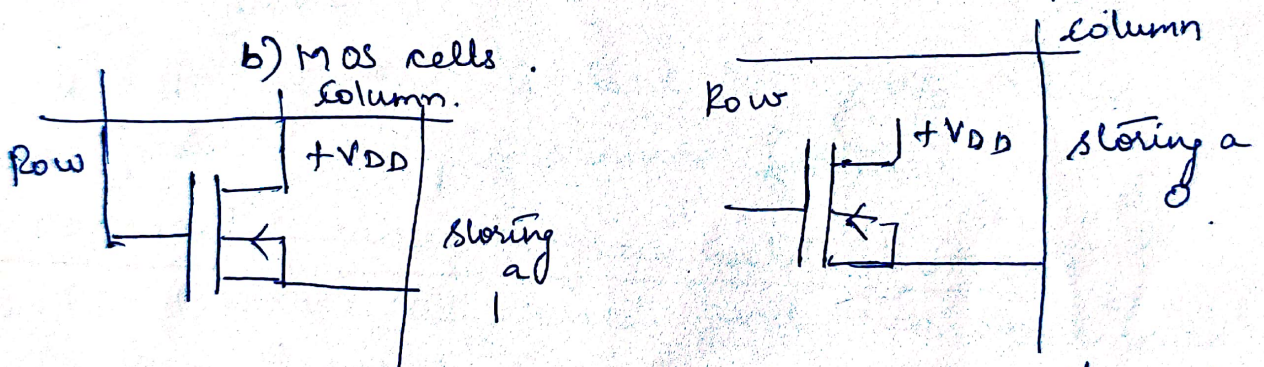
a) Bipolar cell.



* Row line turn ON \rightarrow 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.

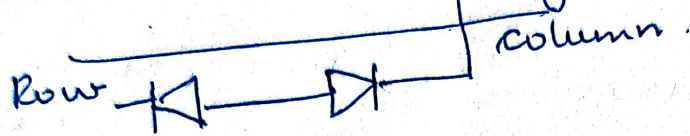
PROM - Programmable Read only memory :-

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

\rightarrow Fuse Technology used in PROM :-

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

- * Metal links 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 junction technology also referred as shotted 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.

A EPROM store 1's and 0's as 4 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

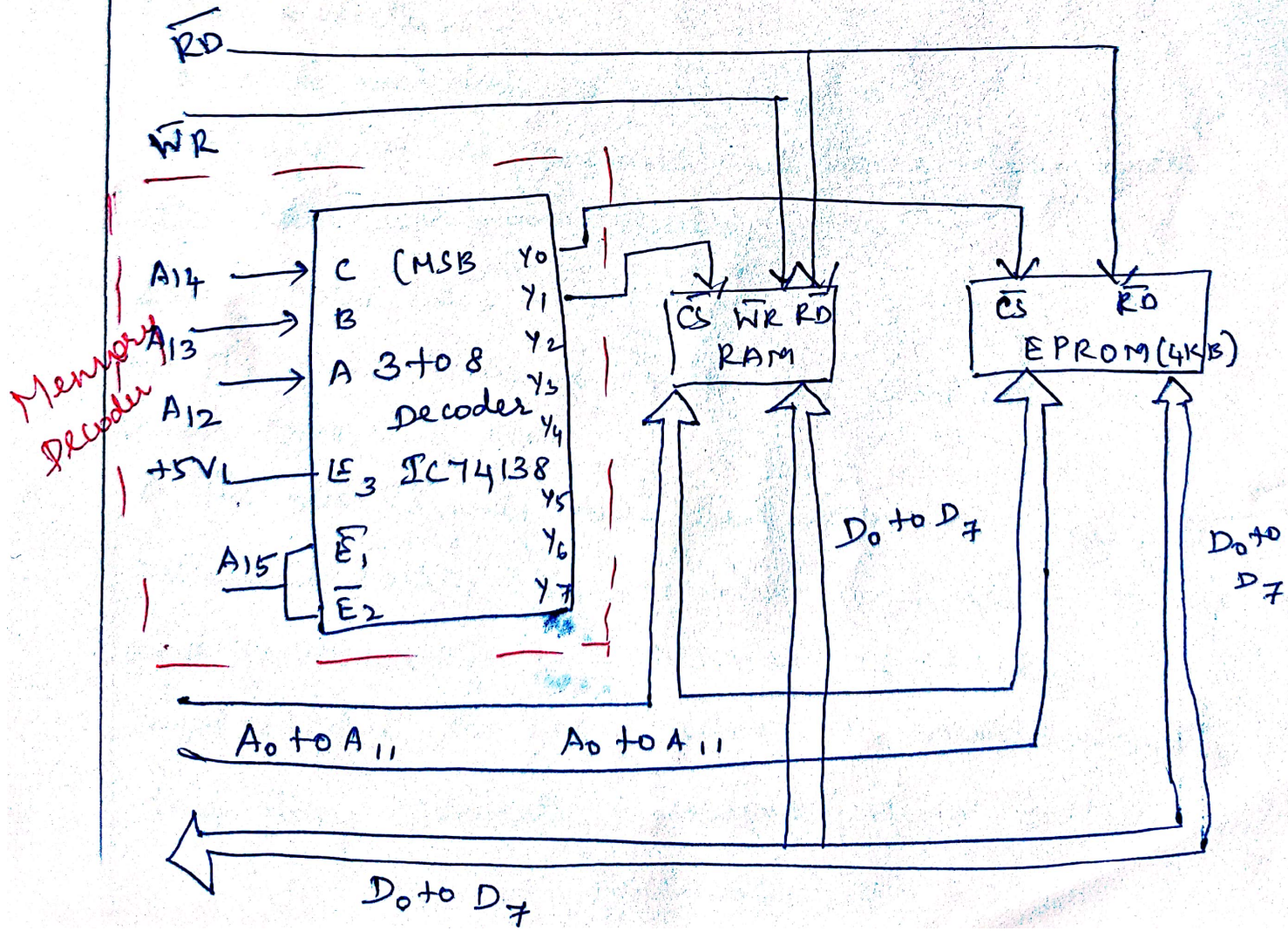
- * ROM Device is EEPROM which is electrically alterable Programmable ROM (EAPROM).
- * EEPROM is non-volatile like EPROM but does not require ultraviolet light.

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.

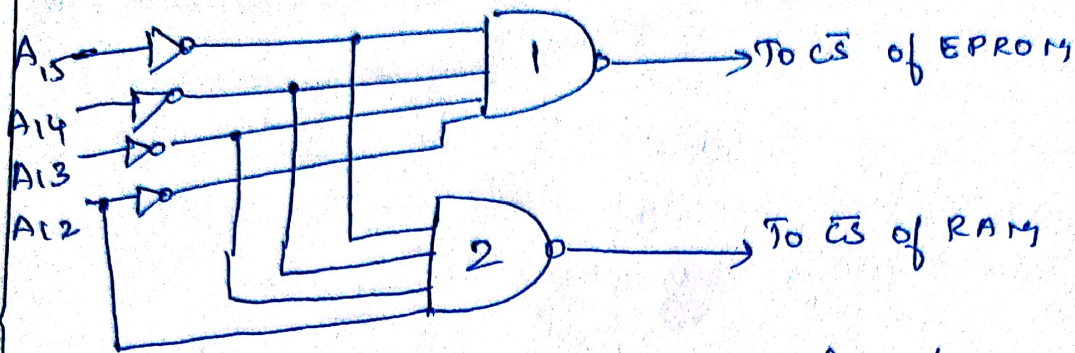
Design of memory Decoder :-

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



Memory Decoder using logic gates

(69)¹⁶



* 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. (70)

→ 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 :-

* 8 bit 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				
8 bit 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 001110010100 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 occurred

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

(71) 12

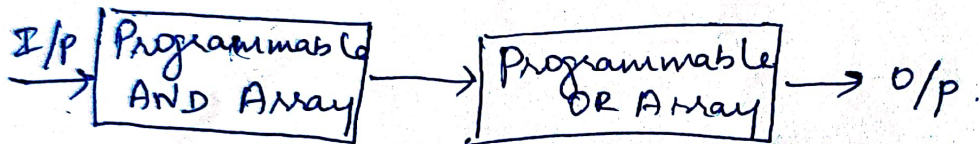
" iv) $C = 0, P = 1$, error occurred in P_{13} bit.

PROGRAMMABLE LOGIC ARRAY:-

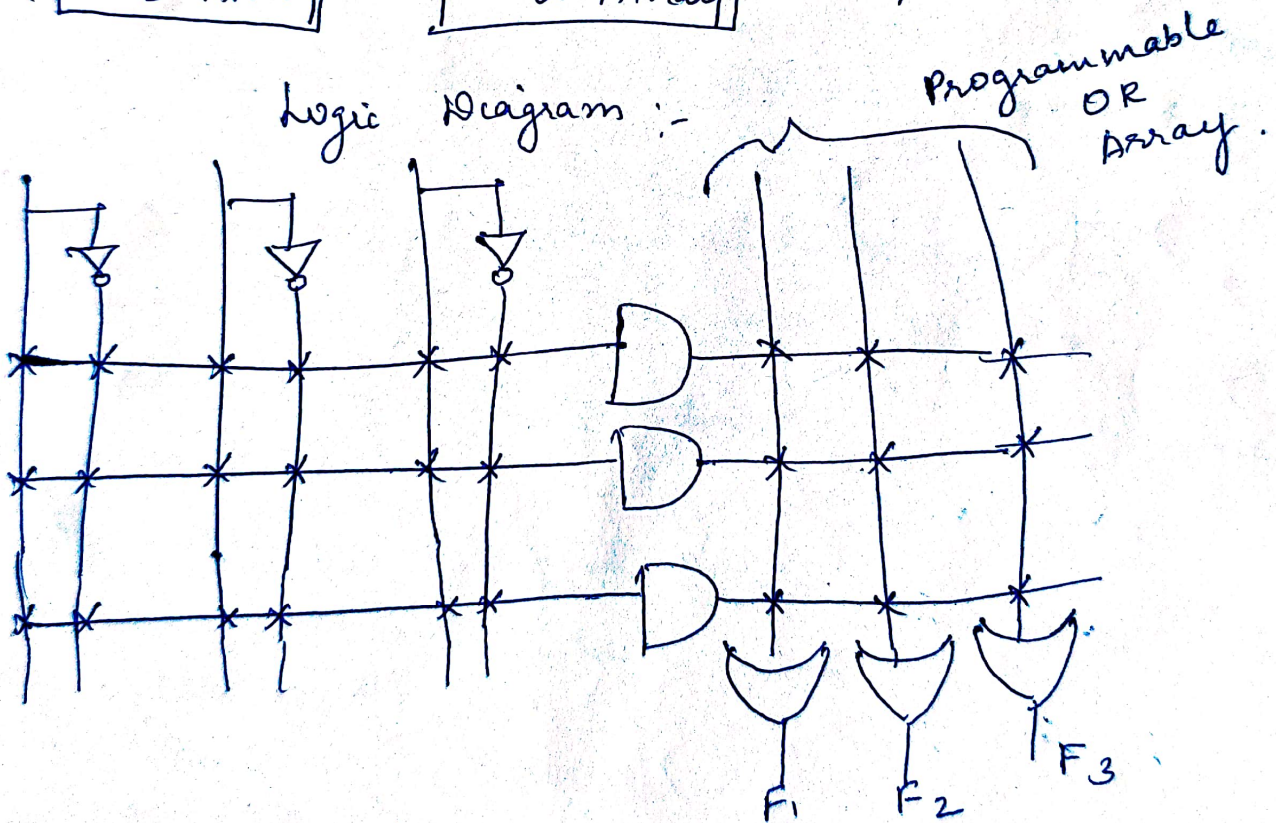
* PLA is a type of fixed architecture logic devices with programmable AND gates followed by Programmable OR Gates.

* AND & OR Gates inside the PLA are initially fabricated with fuses.

* Boolean fn's are implemented in Sum of Product (SOP) form.



Logic Diagram :-

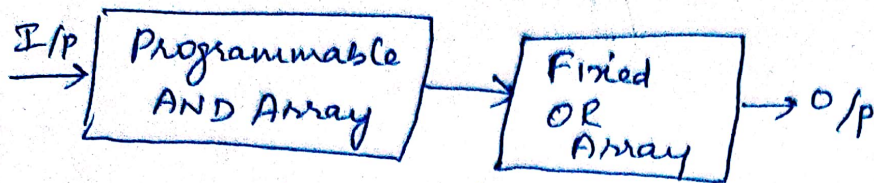


PROGRAMMABLE ARRAY LOGIC (PAL):-

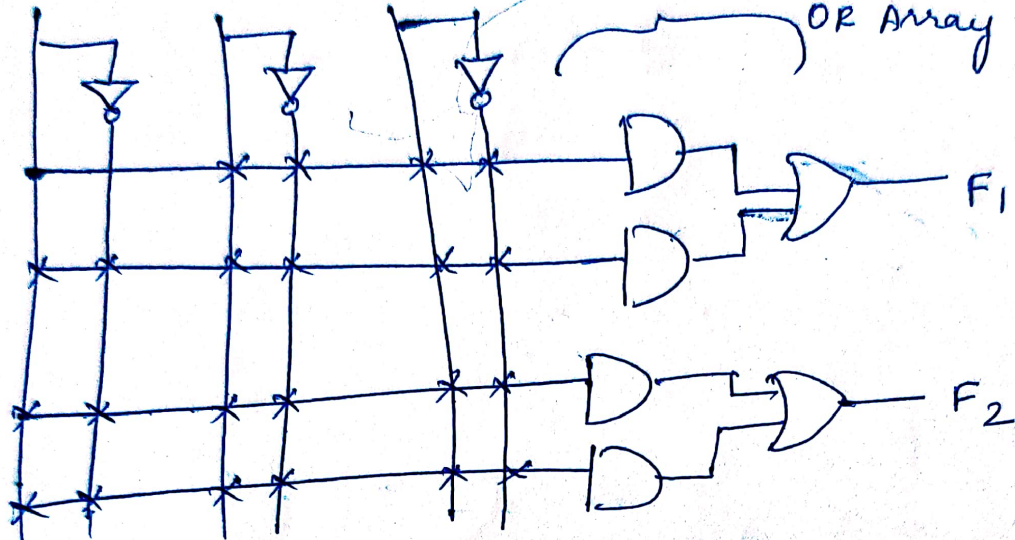
* Programmable Array logic (PAL) is a type of fixed architecture logic device with Programmable AND Arrays

followed by fixed OR Arrays.

(72)¹⁵



Logic diagram



SEQUENTIAL PROGRAMMABLE DEVICES :-

* Sequential Programmable Devices (SPD's) are electronic 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.

* Applications 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 ¹⁴ (73) semiconductor crystal called a chip.

* components such as diodes, Transistors, resistors and capacitors.

Types of ASIC :-

i) Full custom ASIC :-

* An engineer designs 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 known as Mega cells.

iii) Gate Array Based ASIC :-

* The Transistor Predefined on silicon wafer.