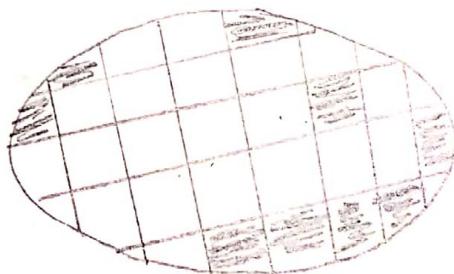


## Digital Hardware :-

Logic circuits are used to build computer hardware and many other types of products. All such products are broadly classified as Digital hardware.

## Integrated Circuit :-

- Integrated circuit placing a no. of transistors (an entire circuit) on a single chip.
- Integrated circuit chips are manufactured on a silicon wafer shown in figure. The wafer is cut to produce the individual chips.



- There exist a wide variety of chips that implement various functions that are useful in the design of digital hardware. The chips range from very simple chips with low functionality to extremely complex chips.

Ex:- A digital hardware product may require a micro processor to perform some arithmetic operations, memory chips to provide storage capability and interface chips that allow easy connection to input and output devices.

- For implementing logic circuits three main types of chips are used.

- 1, Standard
- 2, programmable logic devices (PLD)
- 3, Custom

## 1. Standard chips :-

- standard chips conform to an agreed upon standard in terms of functionality (fixed functionality) and physical configuration.
- Each standard chip contains less than 100 transistors and perform a single function.

## 2. Programmable Logic devices (PLD) :-

- In contrast to standard chip that have fixed functionality. it is possible to contract chip.
- These chips include a collection of programmable switches that allow the internal circuitry in the chip to be configured in many different ways.

### Advantage :-

- 1) Most types of PLD's can be programmed multiple times.
- 2) PLD's are available in wide range of sizes.  
ex:- Field programmable gate Array (FPGA) that contains more than 100 million transistors.

|     |      |      |      |
|-----|------|------|------|
| 111 | 111  | 1111 | 111  |
| 111 | 1111 | 1111 | 111  |
| 111 | 1111 | 1111 | 1111 |

### Disadvantage :-

The programmable switches consumes valuable chip area and limit the speed of operation.

### 3. Custom Designed chips :-

The logic circuiting is designed first and then an appropriate technology is chosen to implement the chip. Finally the chip is manufactured by a company. This approach is known as custom or semi-custom designed and such chips are called custom or semi-custom chips. And sometimes called application specific integrated chips (ASIC).

Advantage :-

→ Design can be optimised for a specific task.

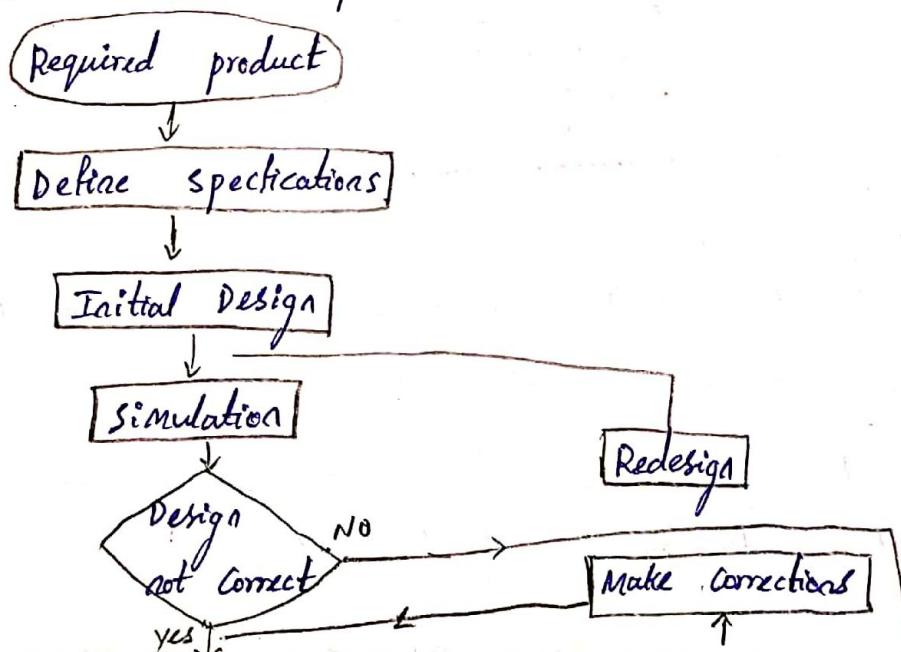
DisAdvantage :-

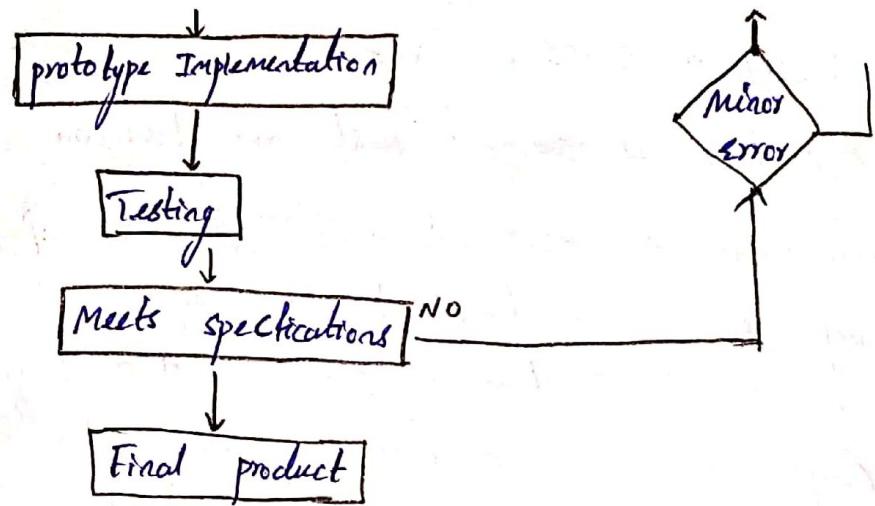
→ Takes considerable amount of time (months).

### 10.17 Design process :-

The flow chart depicts a typical development process. The process begins with definition of product specifications. The essential features of the product are identified and an acceptable method of evaluating the implemented features is done and the final product is established.

Fig Development process

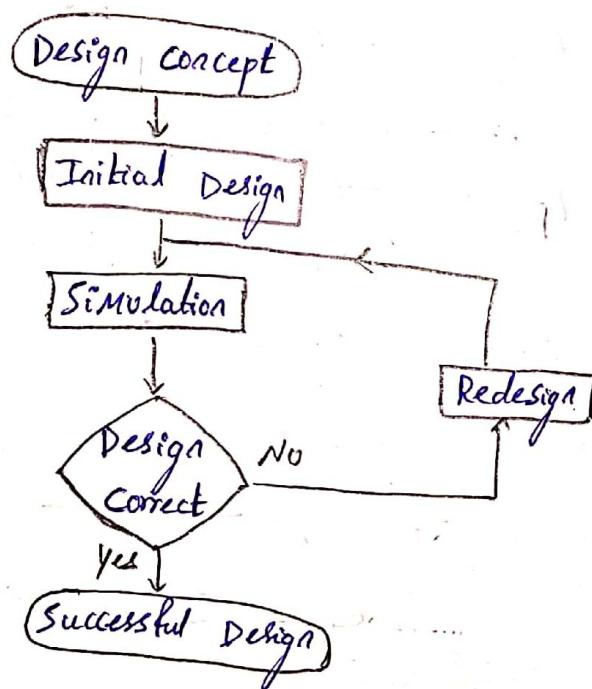




## Design of Digital Hardware :-

### 1. Basic Design Loop :-

Any design process comprises basic sequence of task that are performed in various situations . this sequence is shown in figure .

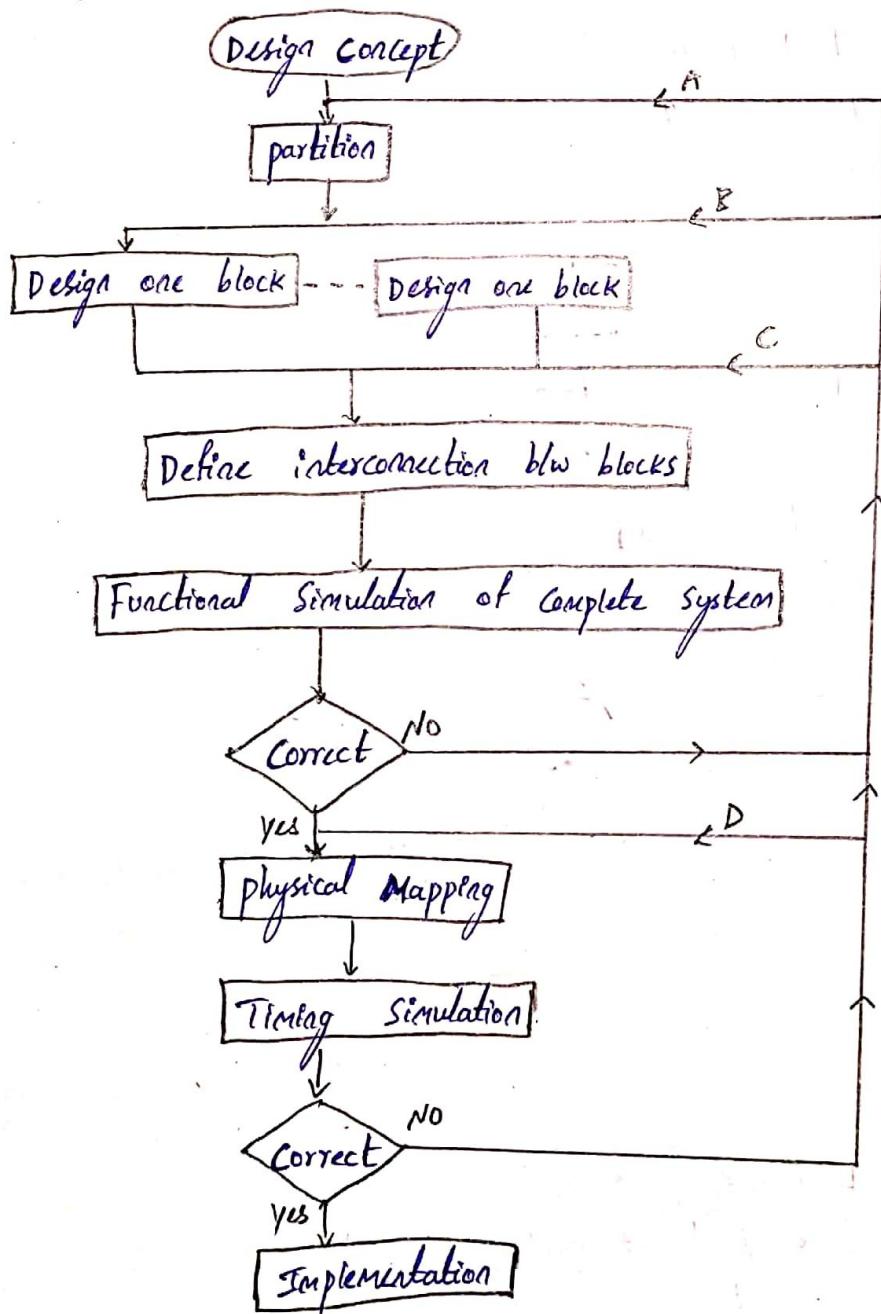


- The first step is to generate and initial design. this step requires a lot of manual effort.
- The next step is simulation of the design. There exist CAD tools to assist in this step.

- > If this simulation reveals some errors then the design must be change to overcome the problems.
- > The Redesigned version is again simulated to determine whether the errors have disappeared. This loop is repeated until the simulation indicates a successful design.

## 2. Design of digital Hardware unit :-

Fig:- Design flow for logic circuit



- partitioning the circuit into smaller block & then designing each block separately.
- Breaking down a large task into more manageable smaller parts is known as divide and conquer approach.

12/7/19

## Introduction to Logic Circuits :-

Variables & Function :-

Consider a Flash light

$L=0$  when light is off

$L=1$  when light is on

$x$  is the variable represent switch

switch open  $x=0$

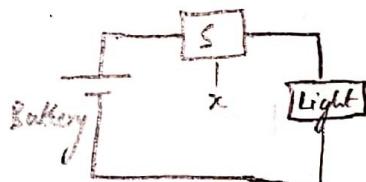
switch closed  $x=1$

$x$  is an i/p variables

When  $x=0$   $L=0$

"  $x=1$   $L=1$

$$L(x) = x$$

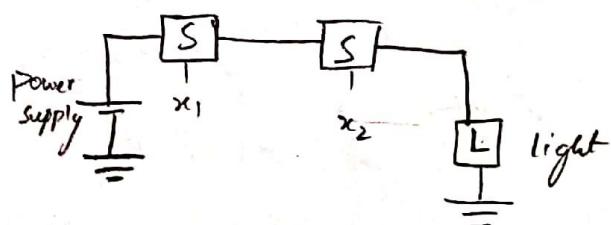


Consider 2 switches

$x_1, x_2$  - control i/p's

Switches can be connected in series & parallel

a) Logical AND function (Series connection)



$$L(x_1, x_2) = x_1 \cdot x_2$$

where  $L=1$  if  $x_1=1$  and  $x_2=1$

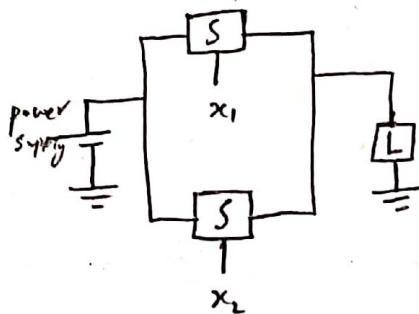
$L=0$  otherwise

$x_1, x_2$

| $x_1$ | $x_2$ | $L$ |
|-------|-------|-----|
| 0     | 0     | 0   |
| 0     | 1     | 0   |
| 1     | 0     | 0   |
| 1     | 1     | 1   |

' $\cdot$ ' symbol is AND operator

b) The logical OR function :-



$$x_1 + x_2$$

| $x_1$ | $x_2$ | $L$ |
|-------|-------|-----|
| 0     | 0     | 0   |
| 0     | 1     | 1   |
| 1     | 0     | 1   |
| 1     | 1     | 1   |

$$L(x_1, x_2) = x_1 + x_2$$

$L = 1$  if  $x_1 = 1$  or  $x_2 = 1$

or if  $x_1 = x_2 = 1$

$L = 0$  if  $x_1 = x_2 = 0$

'f' symbol is OR operator

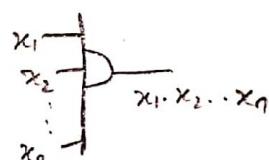
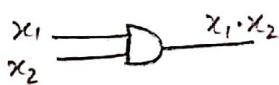
## Logic Gates & Networks

→ Each logic operation can be implemented electronically with transistors result in a circuit element called a logic gate.

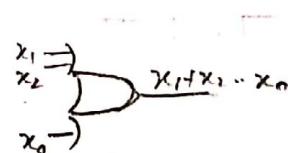
→ A logic gate has one or more inputs and one output that is a function of its input.

Graphical symbols for AND, OR, NOT gates

a) AND gates



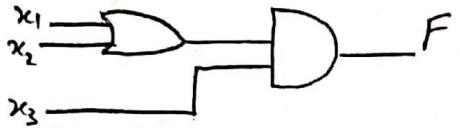
b) OR gate



c) NOT gate



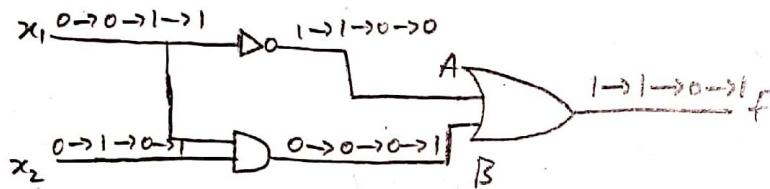
$$F = (x_1 + x_2) \cdot x_3$$



→ A larger circuit is implemented by a network of gates. Example:- the logic function  $F = (x_1 + x_2) \cdot x_3$  can be implemented by the network shown in above figure.

Analysis of a logic network :-

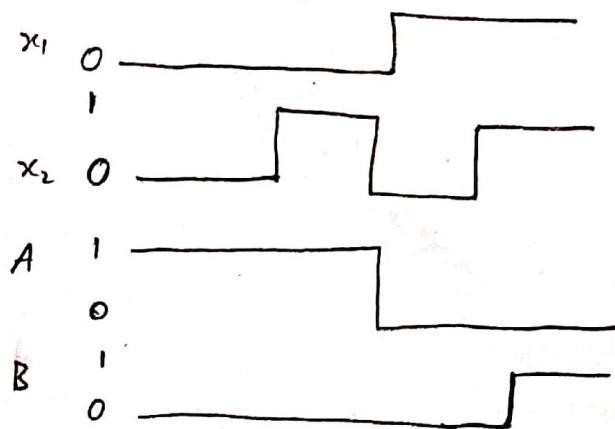
a) Network that implements  $f = \overline{x_1} + x_1 \cdot x_2$



b) Truth Tables for f

| $x_1$ | $x_2$ | $f(x_1, x_2)$ |
|-------|-------|---------------|
| 0     | 0     | 1             |
| 0     | 1     | 1             |
| 1     | 0     | 0             |
| 1     | 1     | 1             |

c) Timing diagram





15/7/19

## Boolean Algebra :-

- In 1854, George Boolean developed one algebraic system now called Boolean Algebra.
- Boolean Algebra is an algebraic structure defined by a set of elements 'B' together with 2 Binary operators (+ and ·).
- Boolean Algebra is based on set of rules that are derived from a small no. of basic assumption. These assumptions are called Axioms. (Accepted to be true).

Following axioms are true

$$0 \cdot 0 = 0$$

$$1 + 1 = 1$$

$$1 \cdot 1 = 1$$

$$0 + 0 = 0$$

$$0 \cdot 0 = 1 \cdot 0 = 0$$

$$1 + 0 = 0 + 1 = 1$$

If  $x=0$  then  $\bar{x}=1$ , If  $x=1$  then  $\bar{x}=0$

## Single Variable theorems

$$1 \cdot 0 = 0$$

$$x \cdot 0 = 0$$

$$x + 1 = 1$$

$$x + 0 = x$$

$$x \cdot x = x$$

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

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

$$\bar{\bar{x}} = x$$

## Two variable Theorem :-

### Duality principle :-

Duality principle is an important property of Boolean Algebra. It states that every algebraic expression deducible from the postulates of Boolean Algebra remains valid if the operators and identity elements are interchanged.

Step:-1 Interchange OR and AND operator

Step:-2 Replace all 1's by 0's, 0's by 1's

$$\text{Ex:- } x+1=1$$

$$x \cdot 0 = 0$$

## Two Variable Theorem :-

$$\begin{aligned} 1) \quad & x \cdot y = y \cdot x \\ & x+y = y+x \end{aligned} \quad \left. \begin{array}{l} \text{Commutative} \\ \text{Commutative} \end{array} \right\}$$

$$\begin{aligned} 2) \quad & x+(y+z) = (x+y)+z \\ & x \cdot (y \cdot z) = (x \cdot y) \cdot z \end{aligned} \quad \left. \begin{array}{l} \text{Associative} \\ \text{Associative} \end{array} \right\}$$

$$\begin{aligned} 3) \quad & x \cdot (y+z) = x \cdot y + x \cdot z \\ & x+(y \cdot z) = (x+y)(x+z) \end{aligned} \quad \left. \begin{array}{l} \text{Distributive} \\ \text{Distributive} \end{array} \right\}$$

$$\begin{aligned} 4) \quad & x+xy = x \\ & x(x+y) = x \end{aligned} \quad \left. \begin{array}{l} \text{Absorption} \\ \text{Absorption} \end{array} \right\}$$

$$\begin{aligned} 5) \quad & (x+y)' = x' \cdot y' \\ & (xy)' = x' + y' \end{aligned} \quad \left. \begin{array}{l} \text{De Morgan's} \\ \text{De Morgan's} \end{array} \right\}$$

$$\begin{aligned} 6) \quad & x \cdot y + x \cdot \bar{y} = x \\ & (x+y)(x+\bar{y}) = x \end{aligned} \quad \left. \begin{array}{l} \text{Combining} \\ \text{Combining} \end{array} \right\}$$

Simplify the Boolean expression to a minimum no. of literals

a)  $x \cdot y + x \cdot y'$

$$x(y + y')$$

$$x \cdot 1$$

$$= x$$

b)  $(x+y)(x+y')$

$$x + (y \cdot y')$$

$$x + 0$$

$$= x$$

c)  $xyz + x'y + xz'y$

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

$$xy + x'y$$

$$y(x+x')$$

$$\Rightarrow y \cdot 1 = y$$

d)  $(A+B)'(A'+B')'$

$$A' \cdot B' ((AB^*)')'$$

$$(A' \cdot B') \cdot (AB)$$

$$\cancel{(A \cdot B)} \cdot \cancel{(A \cdot B)} \quad (A' \cdot A) \cdot (B' \cdot B)$$

$$0 \cdot 0$$

$$x' \cdot x = 0$$

~~#8~~

$$= 0$$

e)  $xyz' + xy'z + xyz + x'y'z'$

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

$$xy \cdot 1 + x'y \cdot 1$$

$$y(x+x')$$

$$y \cdot 1$$

$$= y$$

## Canonical Form & standard Form :-

The standard form of minterms and maxterms demonstrate two important properties of Boolean Algebra.

- 1) Any Boolean function can be expressed as sum of minterms (by 'sum' means ORing of terms)
  - 2) Any Boolean function, can be expressed as sum of maxterms (by 'product' means ANDing of terms)
- ) Thus, expressing the Boolean function as a sum of minterms or product of maxterms is said to be in Canonical form.

1417119

A  $\Sigma^2$  Minterm :-

A product term in which all the variables appear exactly once either complimented or uncomplimented form is called minterm. There are  $2^n$  distinct minterm for n variables. its symbol is  $m_i$ .

→ for two variables 'x' & 'y' there are four minterms

$$X \cdot Y = 2^2 = 4 \text{ unit}$$

| X | Y |                         |
|---|---|-------------------------|
| 0 | 0 | $\bar{X} \bar{Y} - m_0$ |
| 0 | 1 | $\bar{X} Y - m_1$       |
| 1 | 0 | $X \bar{Y} - m_2$       |
| 1 | 1 | $X Y - m_3$             |

$\rightarrow$  for 3 variables

$3 \times 2 \times 2 \rightarrow 8$  minterms

| $x$ | $y$ | $z$ | product term            | Symbol | $m_0$ | $m_1$ | $m_2$ | $m_3$ | $m_4$ | $m_5$ | $m_6$ | $m_7$ |
|-----|-----|-----|-------------------------|--------|-------|-------|-------|-------|-------|-------|-------|-------|
| 0   | 0   | 0   | $\bar{x}\bar{y}\bar{z}$ | $M_0$  | 1     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |
| 0   | 0   | 1   | $\bar{x}\bar{y}z$       | $M_1$  | 0     | 1     | 0     | 0     | 0     | 0     | 0     | 0     |
| 0   | 1   | 0   | $\bar{x}yz$             | $M_2$  | 0     | 0     | 1     | 0     | 0     | 0     | 0     | 0     |
| 0   | 1   | 1   | $\bar{x}y\bar{z}$       | $M_3$  | 0     | 0     | 0     | 1     | 0     | 0     | 0     | 0     |
| 1   | 0   | 0   | $x\bar{y}\bar{z}$       | $M_4$  | 0     | 0     | 0     | 0     | 1     | 0     | 0     | 0     |
| 1   | 0   | 1   | $x\bar{y}z$             | $M_5$  | 0     | 0     | 0     | 0     | 0     | 1     | 0     | 0     |
| 1   | 1   | 0   | $xy\bar{z}$             | $M_6$  | 0     | 0     | 0     | 0     | 0     | 0     | 1     | 0     |
| 1   | 1   | 1   | $xyz$                   | $M_7$  | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 1     |

\* Maxterm :-

A **Sum term** in which all the variables appear exactly once either Complimented or uncomplicated form is called Maxterm. There are  $2^n$  distinct Maxterm for  $n$  variables its symbol is  $M_j$ .

$\rightarrow$  for two variables 'x' & 'y' there are four maxterms

$$x \ y \ z^2 = 4 \text{ units}$$

| $x$ | $y$ |                           |
|-----|-----|---------------------------|
| 0   | 0   | $-x + y - M_0$            |
| 0   | 1   | $-x + \bar{y} - M_1$      |
| 1   | 0   | $\bar{x} + y - M_2$       |
| 1   | 1   | $\bar{x} + \bar{y} - M_3$ |

→ For 3 variables

3 xyz variable → 8 Minterms

| X | Y | Z | sum term                          | Symbol         | M <sub>0</sub> | M <sub>1</sub> | M <sub>2</sub> | M <sub>3</sub> | M <sub>4</sub> | M <sub>5</sub> | M <sub>6</sub> | M <sub>7</sub> |
|---|---|---|-----------------------------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 0 | 0 | 0 | x+y+z                             | M <sub>0</sub> | 0              | 1              | 1              | 1              | 1              | 1              | 1              | 1              |
| 0 | 0 | 1 | x+y+ $\bar{z}$                    | M <sub>1</sub> | 1              | 0              | 1              | 1              | 1              | 1              | 1              | 1              |
| 0 | 1 | 0 | x+ $\bar{y}$ +z                   | M <sub>2</sub> | 1              | 1              | 0              | 1              | 1              | 1              | 1              | 1              |
| 0 | 1 | 1 | x+ $\bar{y}$ + $\bar{z}$          | M <sub>3</sub> | 1              | 1              | 1              | 0              | 1              | 1              | 1              | 1              |
| 1 | 0 | 0 | $\bar{x}$ +y+z                    | M <sub>4</sub> | 1              | 1              | 1              | 1              | 0              | 1              | 1              | 1              |
| 1 | 0 | 1 | $\bar{x}$ +y+ $\bar{z}$           | M <sub>5</sub> | 1              | 1              | 1              | 1              | 1              | 0              | 1              | 1              |
| 1 | 1 | 0 | $\bar{x}$ + $\bar{y}$ +z          | M <sub>6</sub> | 1              | 1              | 1              | 1              | 1              | 1              | 0              | 1              |
| 1 | 1 | 1 | $\bar{x}$ + $\bar{y}$ + $\bar{z}$ | M <sub>7</sub> | 1              | 1              | 1              | 1              | 1              | 1              | 1              | 0              |

$$M_j = \overline{m_j}$$

Verification :-

$$M_1 = \overline{x}\overline{y}z$$

$$M_1 = \overline{m_1} = \overline{\overline{x}\overline{y}z}$$

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

$$M_1 = \overline{m_1}$$

$$\text{ii) } m_5 = x\overline{y}z$$

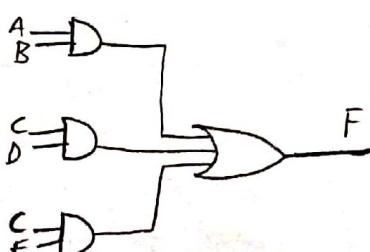
$$M_5 = \overline{m_5} = \overline{x\overline{y}z}$$

$$= \overline{x+y+\bar{z}} = M_5$$

$$\therefore M_5 = \overline{m_5}$$

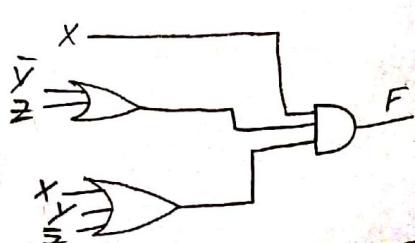
Sum of product (SOP)

$$\text{ex:- } F = AB + CD + CE$$



product of sum (POS)

$$\text{ex:- } F = x(\bar{y}+z), (x+y+\bar{z})$$



→ SOP are sum of minterms

→ POS are product of maxterms

Q) Express the Boolean function  $F = A + \bar{B}C$  as sum of minterms or sum of product terms

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

missing variables

$$\begin{aligned} F &= A(B+B')(C+C') + \bar{B}'C(A+A') \\ &= A(BC + BC' + \bar{B}C + \bar{B}C') + AB'C + A'\bar{B}'C \\ &= ABC + ABC' + AB'C + AB'C' + AB\bar{C} + A'B'C \end{aligned}$$

Removing the terms which appear more than once

$$\Rightarrow ABC + ABC' + AB'C + AB'C' + A'B'C$$

111 110 101 100 001

$$F = m_7 + m_6 + m_5 + m_4 + m_1$$

$$F(A, B, C) = \sum m(1, 4, 5, 6, 7)$$

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

| A | B | C | F                  |
|---|---|---|--------------------|
| 0 | 0 | 0 | 0 - m <sub>0</sub> |
| 0 | 0 | 1 | 1 - m <sub>1</sub> |
| 0 | 1 | 0 | 0 - m <sub>2</sub> |
| 0 | 1 | 1 | 0 - m <sub>3</sub> |
| 1 | 0 | 0 | 1 - m <sub>4</sub> |
| 1 | 0 | 1 | 1 - m <sub>5</sub> |
| 1 | 1 | 0 | 0 - m <sub>6</sub> |
| 1 | 1 | 1 | 1 - m <sub>7</sub> |

$$\sum m(1, 4, 5, 6, 7)$$

| $A + \bar{B}C$ |       |
|----------------|-------|
| 0+0.0          | 0+0.1 |
| 0+1.0          | 0+1.1 |
| 0+0=0          | 0+1=1 |
| 0+1.0          | 0+1.1 |
| 0+0.0          | 0+0.1 |
| 0+0=0          | 0+0=0 |
| 1+0.0          | 1+0.1 |
| 1+1.0          | 1+1.1 |
| 1+0=1          | 1+1=1 |
| 1+1.0          | 1+1.1 |
| 1+0.0          | 1+0.1 |
| 1+0=1          | 1+0=0 |

Hence verified ..

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

$$\text{Sol} \quad = x' + x \underset{x,y=0}{(xy + xz')} \underset{y,z=0}{(yy' + y'z')} \\ = x' + xy + xz' + xy'z'$$

Missing variables

$$\Rightarrow x'(y+z')(z+z') + xy(z+z') + xz'(y+z') \\ \Rightarrow (x'y+x'y') (z+z') + xyz + xyz' + xy'z' + xy'z \\ \Rightarrow x'yz + x'y'z' + x'y'z + xyz + xyz' + xy'z' + xy'z$$

$$\Leftarrow x'y'z + x'y'z'$$

Removing the terms which appears more than once

$$\Rightarrow x'yz + x'y'z' + x'y'z + x'y'z' + xyz + xyz' + xy'z' \\ 011 \quad 010 \quad 001 \quad 000 \quad 111 \quad 110 \quad 100$$

$$F = m_3 + m_2 + m_1 + m_0 + m_7 + m_6 + m_4$$

$$F(x, y, z) = \sum m(0, 1, 2, 3, 4, 6, 7)$$

2) Express the Boolean function  $F = xy + x'z$  as product of maxterms or product of sum

$$\text{i)} \quad F = xy + x'z$$

$$\text{Sol} \quad = (xy + x')(xy + z) \text{ from distributive law } (x+yz) = (x+y)(x+z) \\ \text{Again distributive law}$$

$$\Rightarrow (x+x')(y+x')(x+z)(y+z) \quad \because x+x' = 1$$

$$\Rightarrow 1 \cdot (y+x')(x+z)(y+z)$$

$$\Rightarrow (x'+y)(x+z)(y+z)$$

Missing variables

$$(x'+y+z)(x+z+yz)(y+z+xx')$$

$$\Rightarrow (x'y + z) (x'y + z') (x + z + y) (x + z + y') (y + z + x) (y + z + x')$$

$$F = (x'y + z) (x'y + z') (x + y + z) (x + y' + z) (x + y + z) (x + y + z)$$

Removing the terms which appears more than once

$$F = (x'y + z) (x'y + z') (x + y + z) (x + y' + z)$$

$$F = \underset{0}{(}x'y + z\underset{0}{\}) \underset{0}{(}x + y + z\underset{0}{\}) \underset{1}{(}x'y + z\underset{0}{\}) \underset{1}{(}x + y' + z\underset{1}{\})$$

$M_0$

$M_2$

$M_4$

$M_5$

$$F(x, y, z) = \pi M(0, 2, 4, 5)$$

$$F = xy + x'z$$

| $x$ | $y$ | $z$ | $F$ | $m_0$   | $m_1$   | $m_2$   | $m_3$   |
|-----|-----|-----|-----|---------|---------|---------|---------|
| 0   | 0   | 0   | 0   | - $M_0$ | 0·0+0·0 | 0·0+0·1 | 0·1+0·0 |
| 0   | 0   | 1   | 1   | - $M_1$ | 0·0+1·0 | 0·0+1·1 | 0·1+1·0 |
| 0   | 1   | 0   | 0   | - $M_2$ | 0·0+0·0 | 0+1·1   | 0+0·0   |
| 0   | 1   | 1   | 1   | - $M_3$ | 0·1+0·1 | 1·0+1·0 | 1·0+1·1 |
| 1   | 0   | 0   | 0   | - $M_4$ | 0+1·1   | 0+0·0   | 1·0+0·1 |
| 1   | 0   | 1   | 0   | - $M_5$ | 1·1+1·0 | 1·1+1·1 | 1·0+1·1 |
| 1   | 1   | 0   | 1   | - $M_6$ | 1+0·0   | 1+0·1   | 1+0·0   |
| 1   | 1   | 1   | 1   | - $M_7$ | 1+0·0   | 1+0·1   | 1+0·0   |

$\pi M(0, 2, 4, 5)$

Hence verified.

1917119 Optimized implementation of logic functions using K-map  
map method :-

→ It is used to simplify B.F. 4 Variables

→ It is also known as Karnaugh-map, K-map or map

→ It simplified expressions of SOP and POS.

Two variables map :-

$2^2 = 4$  minterms

|       |       |
|-------|-------|
| $m_0$ | $m_1$ |
| $m_2$ | $m_3$ |

|     |     |   |   |
|-----|-----|---|---|
| $x$ | $y$ | 0 | 1 |
| 0   | 0   | 0 | 1 |
| 1   | 0   | 2 | 3 |

- The map method provides a straight forward procedure for simplifying Boolean function up to 4 variables.
- The map is also known as karnaugh map or K-map.
- The map is a diagram made up of squares with each square representing one minterm of the function.
- K-map is a modified form of truth table in which arrangement of the combination is very much convenient.
- The simplified expressions reduce by the map are always in sum of product (SOP) or product of sum (POS).

Two variable map:-

There are four minterms for 2 binary variables ( $2^2 = 4$  minterms)

|       |       |
|-------|-------|
| $m_0$ | $m_1$ |
| $m_2$ | $m_3$ |

|       |       |   |
|-------|-------|---|
| $x=0$ | $y=0$ | 1 |
| $x=0$ | $y=1$ | 0 |
| $x=1$ | $y=0$ | 0 |

Ex:- 1) Simplify  $F(x,y) = \sum m(1,2,3)$

|       |       |   |
|-------|-------|---|
| $x=0$ | $y=0$ | 1 |
| $x=0$ | $y=1$ | 0 |
| $x=1$ | $y=0$ | 0 |

$$F = y + x$$

$$F = x + y$$

|     |     |
|-----|-----|
| $x$ | $y$ |
| 0   | 1   |
| 1   | 1   |

2)  $F(x,y) = \sum m(0,1,2)$

|       |       |   |
|-------|-------|---|
| $x=0$ | $y=0$ | 1 |
| $x=0$ | $y=1$ | 1 |
| $x=1$ | $y=0$ | 0 |

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

|     |     |
|-----|-----|
| $x$ | $y$ |
| 0   | 0   |
| 1   | 0   |

### Three - variable map :-

There are eight minterms for 3 binary variables ( $2^3 = 8$  minterms)

Three variable map consist of 8 squares

|     | $yz$ | 00    | 01    | 11    | 10    |
|-----|------|-------|-------|-------|-------|
| $x$ | 0    | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
|     | 1    | $m_4$ | $m_5$ | $m_7$ | $m_6$ |

Ex:- 1) Simplify  $F(x,y,z) = \sum m (1,2,3,5,7)$  using K-map

|     | $yz$ | 00 | 01 | 11 | 10 |
|-----|------|----|----|----|----|
| $x$ | 0    | 0  | 1  | 1  | 1  |
|     | 1    | 4  | 5  | 1  | 6  |

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

| $x$ | $y$ | $z$ |
|-----|-----|-----|
| 0   | 1   | 1   |
| 1   | 1   | 0   |

| $x$ | $y$ | $z$ |
|-----|-----|-----|
| 0   | 0   | 1   |
| 0   | 1   | 1   |

2)  $F(x,y,z) = \sum m (0,1,6,7)$

|     | $yz$ | 00 | 01 | 11 | 10 |
|-----|------|----|----|----|----|
| $x$ | 0    | 1  | 1  | 3  | 2  |
|     | 1    | 4  | 5  | 7  | 6  |

| $x$ | $y$ | $z$ |
|-----|-----|-----|
| 1   | 1   | 1   |
| 1   | 1   | 0   |

| $x$ | $y$ | $z$ |
|-----|-----|-----|
| 0   | 0   | 0   |
| 0   | 0   | 1   |

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

3)  $F(x,y,z) = \sum m (0,1,2,3,6,7)$

|     | $yz$ | 00 | 01 | 11 | 10 |
|-----|------|----|----|----|----|
| $x$ | 0    | 1  | 1  | 1  | 1  |
|     | 1    | 4  | 5  | 7  | 6  |

| $x$ | $x$ | $z$ |
|-----|-----|-----|
| 0   | 0   | 0   |
| 0   | 0   | 1   |

| $x$ | $y$ | $z$ |
|-----|-----|-----|
| 1   | 1   | 1   |
| 1   | 1   | 0   |

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

24/7

1) Represent  $F(x,y,z) = yz + x\bar{z}$  in K-map

| <del>x<sub>2</sub></del> | 00 | 01 | 11 | 10 |
|--------------------------|----|----|----|----|
| x                        | 0  |    | 1  |    |
| y                        | 0  |    |    |    |
| z                        | 0  | 1  | 1  | 1  |

| x | y | z |
|---|---|---|
| 0 | 1 | 1 |
| 1 | 1 | 1 |

2) Simplify  $\bar{x}\bar{y}z + yz + xz$

| <del>x<sub>2</sub></del> | 00 | 01 | 11 | 10 |
|--------------------------|----|----|----|----|
| x                        | 0  |    | 1  |    |
| y                        | 0  |    |    |    |
| z                        | 0  | 1  | 1  |    |

$$F = z$$

| <del>x<sub>2</sub></del> | <del>x<sub>1</sub></del> | y | z |
|--------------------------|--------------------------|---|---|
| 0                        | 0                        | 1 | 1 |
| 0                        | 0                        | 1 | 0 |

| x | y | z |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |

3) Simplify  $\sum m(0,2,4,5,6)$  using K-map

| <del>x<sub>2</sub></del> | 00             | 01             | 11             | 10             |
|--------------------------|----------------|----------------|----------------|----------------|
| x                        | m <sub>0</sub> | m <sub>1</sub> | m <sub>3</sub> | m <sub>7</sub> |
| 0                        | 1              |                |                |                |
| 1                        | m <sub>4</sub> | m <sub>5</sub> | m <sub>6</sub> | m <sub>6</sub> |

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

| x | y | z |
|---|---|---|
| 1 | 0 | 1 |
| 1 | 0 | 0 |

| x | y | z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |

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

4 - Variable map :-

$2^4 = 16$  minterms

| <del>w<sub>2</sub></del> | 00 | 01 | 11 | 10 |
|--------------------------|----|----|----|----|
| w <sub>1</sub>           | 0  | 1  | 3  | 2  |
| 00                       | 4  | 5  | 7  | 6  |
| 01                       | 12 | 13 | 15 | 14 |
| 11                       | 8  | 9  | 11 | 10 |

1) Simplify  $F(A, B, C, D) = \sum m(3, 7, 11, 12, 13, 14, 15)$

6)

| $wx$ | 00 | 01 | 11 | 10 |
|------|----|----|----|----|
| 00   | 0  | 1  | 1  | 2  |
| 01   | 4  | 5  | 7  | 6  |
| 11   | 12 | 13 | 15 | 14 |
| 10   | 8  | 9  | 11 | 10 |

| x | y | z | w | x | y | z | w |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |

$wz$

$$F = xy + wz$$

2) Simplify  $F(a, b, c, d) = \prod M(0, 1, 2, 3, 4, 7, 8, 11, 12, 14, 15)$

| $cd$ | 00 | 01 | 11 | 10 |
|------|----|----|----|----|
| ab   | 0  | 0  | 0  | 0  |
| 00   | 0  | 0  | 0  | 8  |
| 01   | 0  | 5  | 0  | 6  |
| 11   | 0  | 13 | 0  | 7  |
| 10   | 0  | 8  | 0  | 4  |

| a  | b | c | d | a | b | c | d |
|----|---|---|---|---|---|---|---|
| 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0  | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0  | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 0  | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| ab |   |   |   |   |   |   |   |
| cd |   |   |   |   |   |   |   |

$$F = ab' + cd + c'd' + abc$$

$$F = (a+b) \cdot (c+d') \cdot (c+d) \cdot (a'+b'+c')$$

| a   | b | c | d | a | b | c | d |
|-----|---|---|---|---|---|---|---|
| 0   | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0   | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1   | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1   | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| cd' |   |   |   |   |   |   |   |

3)  $f(w, x, y, z) = \sum m(0, 1, 2, 4, 6, 8, 9, 10, 12, 14)$

| $wz$ | 00 | 01 | 11 | 10 |
|------|----|----|----|----|
| 00   | 1  | 1  | 1  | 1  |
| 01   | 1  | 5  | 7  | 6  |
| 11   | 1  | 13 | 15 | 14 |
| 10   | 1  | 8  | 9  | 11 |

| w | x | y | z | w | x | y | z |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |

$\bar{xy}$

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

$$4) \text{ Karnaugh Map } f(w, x, y, z) = \sum m(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14)$$

|    | yz | 00 | 01 | 11 | 10 |
|----|----|----|----|----|----|
| wx | 0  | 1  | 1  | 3  | 2  |
| 00 | 1  | 1  |    | 7  | 6  |
| 01 | 1  | 1  |    | 1  |    |
| 11 | 1  | 1  | 13 | 15 | 14 |
| 10 | 1  | 1  | 9  | 11 | 10 |

| w | x | y | z |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 |

| w | x | y | z |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 |

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

5) Simplify  $F = \bar{A}\bar{B}\bar{C} + \bar{B}\bar{C}\bar{D} + A\bar{B}\bar{C} + A\bar{B}\bar{C}\bar{D}$

Sol

|    | CD | 00 | 01 | 11 | 10 |
|----|----|----|----|----|----|
| AB | 0  | 1  |    | 3  | 2  |
| 00 | 1  | 1  |    | 6  |    |
| 01 |    |    |    |    |    |
| 11 |    | 12 | 13 | 15 | 14 |
| 10 | 1  | 1  | 9  | 11 | 10 |

$$\begin{array}{l} ABCD \\ 0000 \\ 0001 \\ 1001 \\ 1000 \end{array}$$

$$\bar{BC}$$

$$ABCD$$

$$0000$$

$$0010$$

$$1000$$

$$1010$$

$$\bar{BD}$$

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

| A | B | C | D |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |

$$\begin{array}{l} A\bar{B}C\bar{D} \\ 1010 \end{array}$$

26/7/19

1) Represent the B.F in k-map  $\bar{b} + a\bar{c} + \bar{a}cd + b$

|      |  | $cd$ | 00 | 01 | 11 | 10 |
|------|--|------|----|----|----|----|
| $ab$ |  | 00   | 1  | 1  | 1  | 1  |
|      |  | 01   |    | 1  |    |    |
| $ab$ |  | 11   | 1  | 1  |    |    |
|      |  | 10   | 1  | 1  | 1  | 1  |

| $a\bar{c}$   | $a\bar{b}$ | $c$ | $d$ |
|--------------|------------|-----|-----|
| $ab\bar{c}d$ | 0          | 0   | 0   |
| 1000         | 0          | 0   | 1   |
| 1001         | 0          | 1   | 0   |
| 1100         | 0          | 0   | 1   |
| 1101         | 1          | 0   | 0   |
|              | 1          | 0   | 1   |
|              | 1          | 0   | 0   |
|              | 1          | 0   | 1   |
|              | 1          | 0   | 1   |

### Don't Care Conditions :-

Functions that have unspecified outputs for some input combinations are called incompletely specified functions.

The unspecified minterm of a function is called Don't care condition.

To distinguish don't care condition from 1's & 0's and 'X' is used.

ex:- 1 simplify  $F(a,b,c,d) = \sum m(13,15) + \sum d(5,7,8,10)$

Q1

|      |  | $cd$ | 00 | 01 | 11 | 10 |
|------|--|------|----|----|----|----|
| $ab$ |  | 00   | 0  | 1  | 3  | 2  |
|      |  | 01   | 4  | 5  | 7  | 6  |
| $ab$ |  | 11   | 12 | 13 | 15 | 14 |
|      |  | 10   | X  | 9  | 11 | X  |

| $a$ | $b$ | $c$ | $d$ |
|-----|-----|-----|-----|
| 0   | 1   | 0   | 1   |
| 0   | 1   | 1   | 1   |
| 1   | 1   | 0   | 1   |
| 1   | 1   | 1   | 1   |

$$F = bd$$

4) 2) Simplify  $F(w, x, y, z) = \sum(1, 5, 10) + d(8, 9, 11, 13)$

| $wx$ | 00 | 01 | 11 | 10 |
|------|----|----|----|----|
| $yz$ | 0  | 1  | 3  | 2  |
| $wx$ | 00 | 1  | 5  | 7  |
| $wx$ | 01 | 1  |    |    |
| $wx$ | 11 | 12 | 13 | 15 |
| $wx$ | 10 | X  | X  | X  |
| $wx$ | 10 | 8  | 9  | 11 |
| $wx$ | 10 | X  | X  | 11 |

| $wx$ | $yz$ |
|------|------|
| 0001 | 1002 |
| 0101 | 1000 |
| 1101 | 1001 |
| 1001 | 1011 |
| 1010 | 1010 |

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

3). Simplify  $F(w, x, y, z) = \sum(0, 2, 4, 6, 8) + \sum d(10, 11, 12, 13, 14, 15)$

| $wx$ | 00 | 01 | 11 | 10 |
|------|----|----|----|----|
| $yz$ | 0  | 1  | 3  | 2  |
| $wx$ | 00 | 1  |    |    |
| $wx$ | 01 | 1  | 5  | 7  |
| $wx$ | 11 | 12 | 13 | 15 |
| $wx$ | 10 | X  | X  | X  |
| $wx$ | 10 | 8  | 9  | 11 |
| $wx$ | 10 | 1  |    | X  |

| $wx$ | $yz$ |
|------|------|
| 0000 | 0000 |
| 0100 | 1100 |
| 1100 | 1000 |
| 1000 | 0010 |
| 0110 | 0110 |
| 1110 | 1110 |
| 1010 | 1010 |

$$F = \bar{z}$$

4)  $F(A, B, C, D) = \pi(1, 2, 3, 8, 9, 10, 11, 14) \cdot d(7, 15)$

| $AB$ | 00 | 01 | 11 | 10 |
|------|----|----|----|----|
| $CD$ | 0  | 1  | 3  | 2  |
| $AB$ | 00 | 0  | 0  | 0  |
| $AB$ | 01 | 0  | 5  | 6  |
| $AB$ | 11 | 12 | 13 | X  |
| $AB$ | 10 | 0  | 0  | 0  |
| $AB$ | 10 | 0  | 0  | 0  |

| $ABCD$      | $ABCD$ |
|-------------|--------|
| 0001        | 0011   |
| 0011        | 0010   |
| 1001        | 1011   |
| 1011        | 1010   |
| BD          | BC     |
| $ABCD$      | $ABCD$ |
| 1000        | 1111   |
| 1001        | 1110   |
| 1011        | 1011   |
| 1010        | 1010   |
| A $\bar{B}$ | AC     |

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

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

$$F(A, B, C, D) = \sum m(0, 2, 8, 9, 10, 15) + \sum d(1, 3, 6, 7)$$

Q1

|    | CD | 00 | 01 | 11 | 10 |
|----|----|----|----|----|----|
| AB | 00 | 0  | 1  | 3  | 2  |
|    | 00 | 1  | X  | X  | 1  |
|    | 01 | 4  | 5  | 7  | 6  |
|    | 11 | 12 | 13 | 15 | 14 |
|    | 10 | 8  | 9  | 11 | 10 |
|    |    | 1  | 1  | 1  | 1  |

| ABCD             | ABCD       |
|------------------|------------|
| 0000             | 0000       |
| 0010             | 0001       |
| 1000             | 1000       |
| 1010             | 1001       |
| $\bar{B}\bar{D}$ | $\bar{B}C$ |

| ABCD  |
|-------|
| 0111  |
| 1111  |
| $BCD$ |

$$F = \bar{B}\bar{C} + \bar{B}\bar{D} + BCD$$

$$6) F(A, B, C, D) = \sum m(0, 1, 4, 6, 8, 9, 10, 12) + \sum d(3, 7, 13, 14, 15)$$

|    | CD | 00 | 01 | 11 | 10 |
|----|----|----|----|----|----|
| AB | 00 | 1  | 1  | X  | 2  |
|    | 00 | 1  | 5  | 7  | 6  |
|    | 01 | 1  | X  | 1  | 1  |
|    | 11 | 1  | X  | X  | X  |
|    | 10 | 1  | 1  | 1  | 1  |
|    |    | 9  | 11 | 13 | 10 |

| ABCD             | ABCD |
|------------------|------|
| 0000             | 0000 |
| 0001             | 0001 |
| 1000             | 1000 |
| 1000             | 1000 |
| $\bar{B}\bar{E}$ |      |

| ABCD | ABCD |
|------|------|
| 0100 | 1110 |
| 1100 | 1100 |
| 0110 | 1000 |
| 1110 | 1010 |
| $BD$ | $AD$ |

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

31719. **prime Implicants :-**

→ It is the product term obtained by combining maximum possible no. of adjacent squares in the map.

**Essential prime implicant :-**

→ If a minterm in a square is covered by only one prime implicant then that prime implicant is called essential prime implicant.

$$\text{ex:- } F(A, B, C, D) = \sum m(1, 3, 4, 5, 6, 7, 12, 14)$$

|   |    |    |    |
|---|----|----|----|
| 0 | 1  | 1  | 2  |
| 1 | 4  | 5  | 6  |
| 1 | 12 | 11 | 13 |
| 1 | 14 | 15 | 16 |

Here  $A'D$ ,  $BD'$ ,  $A'B$  are p.I.  
But  $A'B$  is non essential p.I.  
 $A'D$  and  $BD'$  are essential p.I.  
 $F = A'D + BD'$

5 Variable Map :-

There are 32 minterms for 5 variables K-mapping.

Ex:-  $f(V, W, X, Y, Z) = \Sigma (0, 2, 4, 6, 9, 11, 13, 15, 17, 21, 25, 27, 29, 31)$

Formal :-

|    |    | V=0 |    |    |    |   |
|----|----|-----|----|----|----|---|
|    |    | 00  | 01 | 11 | 10 |   |
| Wx |    | 00  | 0  | 1  | 3  | 2 |
| 00 | 1  |     |    |    | 1  |   |
| 01 | 4  | 5   | 7  | 6  |    |   |
| 11 | 12 | 13  | 15 | 14 |    |   |
| 10 | 8  | 9   | 11 | 10 |    |   |

|    |    | V=1 |    |    |    |    |
|----|----|-----|----|----|----|----|
|    |    | 00  | 01 | 11 | 10 |    |
| Wx |    | 00  | 16 | 17 | 19 | 18 |
| 00 |    | 1   |    |    |    |    |
| 01 |    | 20  | 21 | 22 | 22 |    |
| 11 | 28 | 29  | 31 | 30 |    |    |
| 10 | 24 | 25  | 27 | 26 |    |    |

V1

$$\begin{matrix} V & W & X & Y & Z \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{matrix}$$

$$\bar{V}\bar{W}\bar{Z}$$

$$\begin{matrix} V & W & X & Y & Z \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{matrix}$$

$$W\bar{Z}$$

$$V\bar{Y}Z$$

$$F = WZ + \bar{V}\bar{W}\bar{Z} + V\bar{Y}Z$$

Note :- If same pair exist in both the boxes than V term is not included.

1)  $f(V, W, X, Y, Z) = \Sigma (1, 2, 6, 7, 9, 13, 14, 15, 17, 22, 23, 25, 29, 30, 31)$

Sol

|    |    | V=0 |    |    |    |   |
|----|----|-----|----|----|----|---|
|    |    | 00  | 01 | 11 | 10 |   |
| Wx |    | 00  | 0  | 1  | 3  | 2 |
| 00 |    | 1   |    |    |    |   |
| 01 | 4  | 5   | 7  | 6  |    |   |
| 11 | 12 | 13  | 15 | 14 |    |   |
| 10 | 8  | 9   | 11 | 10 |    |   |

|    |    | V=1 |    |    |    |    |
|----|----|-----|----|----|----|----|
|    |    | 00  | 01 | 11 | 10 |    |
| Wx |    | 00  | 16 | 17 | 19 | 18 |
| 00 |    | 1   |    |    |    |    |
| 01 |    | 20  | 21 | 22 | 22 |    |
| 11 | 28 | 29  | 31 | 30 |    |    |
| 10 | 24 | 25  | 27 | 26 |    |    |

$$Wx$$

$$y z$$

$$w y z$$

$$w\bar{z}$$

$$xy$$

$$Wx y z$$

$$Wx y \bar{z}$$

$$Wx \bar{y} z$$

$$Wx \bar{y} \bar{z}$$

$$W\bar{x} y z$$

$$Wx y z$$

$$Wx y \bar{z}$$

$$Wx \bar{y} z$$

$$Wx \bar{y} \bar{z}$$

$$W\bar{x} y z$$

$$Wx y z$$

$$Wx y \bar{z}$$

$$Wx \bar{y} z$$

$$Wx \bar{y} \bar{z}$$

$$W\bar{x} y z$$

$$Wx y z$$

$$Wx y \bar{z}$$

$$Wx \bar{y} z$$

$$Wx \bar{y} \bar{z}$$

$$W\bar{x} y z$$

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

## Karnaugh Map

### Quine McClusky Tabular method :-

Q1)  $F(A, B, C, D) = \sum(0, 1, 3, 7, 8, 9, 11, 15)$

Step 1 :- Represent all given minterms with their binary equivalents

minterms

|    | A | B | C | D |      |
|----|---|---|---|---|------|
| 0  | 0 | 0 | 0 | 0 | 0000 |
| 1  | 0 | 0 | 0 | 1 | 0001 |
| 3  | 0 | 0 | 1 | 0 | 0010 |
| 7  | 0 | 1 | 0 | 1 | 0011 |
| 8  | 1 | 0 | 0 | 0 | 0100 |
| 9  | 1 | 0 | 0 | 1 | 0101 |
| 11 | 1 | 1 | 0 | 1 | 0110 |
| 15 | 1 | 1 | 1 | 1 | 0111 |
|    |   |   |   |   | 1000 |
|    |   |   |   |   | 1001 |

Step 2 :- Arrange the minterms in the form of groups according to the no. of one's

minterms

|    | A | B | C | D |                |
|----|---|---|---|---|----------------|
| 0  | 0 | 0 | 0 | 0 | → 0 no. of 1's |
| 1  | 0 | 0 | 0 | 1 | → 1. no of 1's |
| 8  | 1 | 0 | 0 | 0 |                |
| 3  | 0 | 0 | 1 | 1 | → 2 no of 1's  |
| 9  | 1 | 0 | 0 | 1 |                |
| 7  | 0 | 1 | 1 | 1 | → 3 no of 1's  |
| 11 | 1 | 0 | 1 | 1 |                |
| 15 | 1 | 1 | 1 | 1 | → 4 no of 1's  |

Step 3 :- compare minterms in one group with every minterm in next highest index group place a 1 where the variable is unmatched.

Note :- only a single difference should come.

| Minterm | A | B | C | D |
|---------|---|---|---|---|
| 0,1     | 0 | 0 | 0 | = |
| 0,8     | - | 0 | 0 | 0 |
| 1,3     | 0 | 0 | - | 1 |
| 1,9     | - | 0 | 0 | 1 |
| 8,9     | 1 | 0 | 0 | - |
| 3,7     | 0 | - | 1 | 1 |
| 3,11    | - | 0 | 1 | 1 |
| 9,11    | 1 | 0 | - | 1 |
| 7,15    | - | 1 | 1 | 1 |
| 11,15   | 1 | - | 1 | 1 |

21B119 Step 4 :- Apply same process for resultant column and continue this until no further elimination of literals.

| Numbers    | A  | B | C | D |
|------------|----|---|---|---|
| 0,1,8,9    | 0  | 0 | 0 | - |
| 0,8,1,9    | -  | 0 | 0 | 0 |
| 1,8,9,11   | -  | 0 | 0 | 1 |
| 8,9,3,11   | +0 | 0 | 0 | 1 |
| 3,17,11,15 | -  | - | 1 | 1 |
| 3,11,7,15  | -  | - | 1 | 1 |

numbers

|              | A | B | C | D |
|--------------|---|---|---|---|
| 0, 1, 8, 9   | 0 | 0 | 0 | - |
| 1, 3, 9, 11  | - | 0 | - | 1 |
| 3, 7, 11, 15 | - | - | 1 | 1 |

Step 5 :- List all the prime implicants in the prime implicant chart

| prime implicant | literal representation | members $\Sigma m$ |       |       |       |       |       |          |
|-----------------|------------------------|--------------------|-------|-------|-------|-------|-------|----------|
|                 |                        | $m_0$              | $m_1$ | $m_3$ | $m_7$ | $m_8$ | $m_9$ | $m_{11}$ |
| 3, 7, 11, 15    | $CD$                   |                    |       |       | x     | (X)   |       | x (X)    |
| 1, 3, 9, 11     | $\bar{B}D$             |                    | x     | x     |       |       |       |          |
| 0, 1, 8, 9      | $\bar{B}\bar{C}$       | (X)                | x     |       |       | (X)   | x     |          |

$$F = CD + \bar{B}\bar{C}$$

K- mapping :-

| <del>AB</del> | 00 | 01 | 11 | 10 |
|---------------|----|----|----|----|
| 00            | 0  | 1  | 1  | 2  |
| 01            | 4  | 5  | 7  | 6  |
| 11            | 12 | 13 | 15 | 14 |
| 10            | 8  | 9  | 11 | 10 |
|               | 1  | 1  | 1  |    |

| A | B | C | D | $\bar{B}\bar{C}$ | $CD$ |
|---|---|---|---|------------------|------|
| 0 | 0 | 0 | 0 | 00               | 00   |
| 0 | 0 | 0 | 1 | 01               | 01   |
| 1 | 0 | 0 | 0 | 11               | 11   |
| 1 | 0 | 0 | 1 | 10               | 10   |

$$F = \bar{B}\bar{C} + CD$$

∴ Hence Verified,

1, Simplify using tabulation method

$$F(A, B, C, D) = \sum (0, 2, 4, 6, 7, 9) + \sum d(10, 11)$$

Sol Step 1 :- minterms A B C D

|    |   |   |   |
|----|---|---|---|
| 0  | 0 | 0 | 0 |
| 2  | 0 | 0 | 1 |
| 4  | 0 | 1 | 0 |
| 6  | 0 | 1 | 1 |
| 7  | 0 | 1 | 1 |
| 9  | 1 | 0 | 0 |
| 10 | 1 | 0 | 1 |
| 11 | 1 | 0 | 1 |

Step 2 :- minterms | A B C D

|    |   |   |   |
|----|---|---|---|
| 0  | 0 | 0 | 0 |
| 2  | 0 | 0 | 1 |
| 4  | 0 | 1 | 0 |
| 6  | 0 | 1 | 0 |
| 9  | 1 | 0 | 0 |
| 10 | 1 | 0 | 1 |
| 7  | 0 | 1 | 1 |
| 11 | 1 | 0 | 1 |

Step 3 :- minterms | A B C D

|      |   |   |   |   |
|------|---|---|---|---|
| 0,2  | 0 | 0 | - | 0 |
| 0,4  | 0 | - | 0 | 0 |
| 2,6  | 0 | - | 1 | 0 |
| 2,10 | - | 0 | 1 | 0 |
| 4,6  | 0 | 1 | - | 0 |

|              |       |   |   |   |   |
|--------------|-------|---|---|---|---|
| <del>R</del> | 6,7   | A | B | C | D |
| Q            | 9,11  | 0 | 1 | 1 | - |
| R            | 10,11 | 1 | 0 | - | 1 |
|              |       | 1 | 0 | 1 | + |

Step 4 :-

| Numbers | A | B | C | D |
|---------|---|---|---|---|
| 0,2,4,6 | 0 | - | - | 0 |
| 0,4,2,6 | 0 | - | - | 0 |

| Number    | A | B | C | D |
|-----------|---|---|---|---|
| P 0,2,4,6 | 0 | - | - | 0 |

Step 5 :-

| prime implicant | literal representation | minimum sum  |
|-----------------|------------------------|--|
| P 0,2,4,6       | $\bar{A}\bar{D}$       | $m_0 \quad m_2 \quad m_4 \quad m_6 \quad m_7 \quad m_9$<br>$(\times) \quad (\times) \quad (\times) \quad x \quad   \quad  $<br>$  \quad   \quad   \quad   \quad   \quad  $<br>$x \quad   \quad   \quad   \quad   \quad  $<br>$  \quad   \quad   \quad   \quad   \quad  $<br>$(\times) \quad   \quad   \quad   \quad   \quad  $ |
| Q 9,11          | $A\bar{B}D$            |  |
| R 6,7           | $\bar{A}BC$            |  |

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

Verification :-

| CD | 00 | 01 | 11 | 10 |
|----|----|----|----|----|
| AB | 00 | 01 | 11 | 10 |
| 00 | 1  | -  | 3  | 1  |
| 01 | 4  | 5  | 7  | 6  |
| 11 | 1  | -  | 1  | 1  |
| 10 | 12 | 13 | 15 | 14 |
|    | 8  | 9  | 11 | 10 |
|    | 1  | X  | X  |    |

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| A | B | C | D | A | B | C | D |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |

|   |   |   |   |
|---|---|---|---|
| A | B | C | D |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 |

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

Hence Verified

$$2) F(A, B, C, D) = \sum m(4, 8, 10, 11, 12, 15) + \sum d(9, 14)$$

Step 1 :-

Minterms A B C D

|    |   |   |   |   |
|----|---|---|---|---|
| 4  | 0 | 1 | 0 | 0 |
| 8  | 1 | 0 | 0 | 0 |
| 9  | 1 | 0 | 0 | 1 |
| 10 | 1 | 0 | 1 | 0 |
| 11 | 1 | 0 | 1 | 1 |
| 12 | 1 | 1 | 0 | 0 |
| 14 | 1 | 1 | 1 | 0 |
| 15 | 1 | 1 | 1 | 1 |

Step 2 :-

Minterms A B C D

|    |   |   |   |   |
|----|---|---|---|---|
| 4  | 0 | 1 | 0 | 0 |
| 8  | 1 | 0 | 0 | 0 |
| 9  | 1 | 0 | 0 | 1 |
| 10 | 1 | 0 | 1 | 0 |
| 12 | 1 | 1 | 0 | 0 |
| 11 | 1 | 0 | 1 | 1 |
| 14 | 1 | 1 | 1 | 0 |
| 15 | 1 | 1 | 1 | 1 |

Step 3 :-

Minterms A B C D

|        |   |   |   |   |
|--------|---|---|---|---|
| 4, 12  | - | 1 | 0 | 0 |
| 8, 9   | 1 | 0 | 0 | - |
| 8, 10  | 1 | 0 | - | 0 |
| 8, 12  | 1 | - | 0 | 0 |
| 9, 11  | 1 | 0 | - | 1 |
| 10, 11 | 1 | 0 | 1 | - |
| 10, 14 | 1 | - | 1 | 0 |
| 12, 14 | 1 | 1 | - | 0 |
| 11, 15 | 1 | - | 1 | 1 |
| 14, 15 | 1 | 1 | 1 | - |

Step 4 :-

Minterms A B C D

|                |   |   |   |   |
|----------------|---|---|---|---|
| 8, 9, 10, 11   | 1 | 0 | - | - |
| 8, 10, 9, 11   | 1 | 0 | - | - |
| 8, 10, 12, 14  | 1 | - | - | 0 |
| 8, 12, 10, 14  | 1 | - | - | 0 |
| 10, 11, 14, 15 | 1 | - | 1 | - |
| 10, 14, 11, 15 | 1 | - | 1 | - |

Minterms A B C D

|                |   |   |   |   |
|----------------|---|---|---|---|
| 8, 9, 10, 11   | 1 | 0 | - | - |
| 8, 10, 12, 14  | 1 | - | - | 0 |
| 10, 11, 14, 15 | 1 | - | 1 | - |
| 4, 12          | - | 1 | 0 | 0 |

Step 5 :-

| prime implicant | literal representation | Minterm SM |
|-----------------|------------------------|------------|
| 4, 12           | $A\bar{B}$             | x x x      |
| 8, 10, 12, 14   | $A\bar{D}$             | x x x      |
| 10, 11, 14, 15  | $AC$                   | x x ⊗      |
| 4, 12           | $B\bar{C}\bar{D}$      | x          |

Essential prime implicant -  $AC + B\bar{C}\bar{D}$

Non Essential P.I -  $A\bar{D} + A\bar{B}$

Since 8 is missing in PI

Answer =  $AC + B\bar{C}\bar{D} + A\bar{B} \text{ or } A\bar{D}$

Verification :-

| CD   | 00  | 01   | 11   | 10   |
|------|-----|------|------|------|
| AB   | 0   | 1    | 3    | 2    |
| BCD  | 000 | 1100 | 1100 | 1100 |
| 1111 | 4   | 5    | 7    | 6    |
| 1011 | 1   |      |      |      |
| 1110 |     |      |      |      |
| 1010 | 12  | 13   | 15   | 14   |
| AC   |     |      |      |      |
| 10   | 1   | X    | 1    | 10   |

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

## Number Representation :

### Addition & subtraction of signed & unsigned numbers :-

→ positive Integers (including zero) can be represented as unsigned numbers.

→ Both Signed and unsigned Binary numbers consist of string of bits when represented in a computer. the left most bit represents the sign and rest of bit represent the number.

ex:- 01001 can be represented as 9 (unsigned binary) or +9 (signed binary) because the leftmost bit is zero.

→ 11001 represents -9 when considered as a signed number because '1' in leftmost position designates a '-ve'.

→ Consider 9 represented in binary with 8 bits

+9 (00001001) although there is only one way to represent +9, there are 3 different ways to represent -9 with 8 bits.

3 different ways ..

1) Signed - magnitude representation

100001001

changing sign 0 to 1

2) Signed - 1's complement representation

11110110

3) Signed - 2's complement representation

2's Complement of 9 = 100001001

Take 1's Complement = 11110110

Add 1 = +1

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

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

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

| A | B | Sum | Carry |
|---|---|-----|-------|
| 0 | 0 | 0   | 0     |
| 0 | 1 | 1   | 0     |
| 1 | 0 | 1   | 0     |
| 1 | 1 | 0   | 1     |

## Signed Binary Numbers :-

| Decimal | Signed 2's complement | Signed 1's complement | Signed magnitude |
|---------|-----------------------|-----------------------|------------------|
| +7      | 0111                  | 0111                  | 0111             |
| +6      | 0110                  | 0110                  | 0110             |
| +5      | 0101                  | 0101                  | 0101             |
| -1      | 1111                  | 1110                  | 1001             |
| -2      | 1110                  | 1101                  | 1000             |
| -3      | 1101                  | 1100                  | 1011             |

3's comp = 0011  
 2's .. = 1100  
 +1  
 1101  
 3 = 0110  
 -3 = 1011

### Addition :-

$$\begin{array}{r}
 1. \quad +6 \rightarrow 00000110 \\
 +13 +, 00001101 \\
 \hline
 +19 \quad \underline{\underline{00010011}}
 \end{array}$$

Verify:

$$\begin{array}{r}
 2 \overline{)19} & - R \\
 2 \overline{)19} & \quad | \\
 2 \overline{)4} & \quad 0 \\
 2 \overline{)2} & \quad 0 \\
 \hline
 & \quad F
 \end{array}$$

$$\begin{array}{r}
 2. \quad +5 \rightarrow 00000101 \\
 +15 +, 00001111 \\
 \hline
 +20 \quad \underline{\underline{00010100}}
 \end{array}$$

$$\begin{array}{r}
 10011 \\
 2 \overline{)15} \quad 1 \\
 2 \overline{)17} \quad 1 \\
 2 \overline{)3} \quad 1 \\
 \hline
 1
 \end{array}
 \quad
 \begin{array}{r}
 2 \overline{)20} \quad 0 \\
 2 \overline{)10} \quad 0 \\
 2 \overline{)5} \quad 1 \\
 2 \overline{)2} \quad 0 \\
 \hline
 1
 \end{array}$$

$$\begin{array}{r}
 3. \quad -6 \rightarrow 11111010 \\
 +13 \rightarrow +00001101 \\
 \hline
 +7 \quad \underline{\underline{100000111}}
 \end{array}$$

2's Complement of 6

$$\begin{array}{r}
 0000110 \\
 1's Comp 11111001 \\
 +1 \\
 \hline
 1111010
 \end{array}$$

$$\begin{array}{r}
 4. \quad +6 \quad 00000110 \\
 -13 \quad 11110010 \\
 \hline
 -7 \quad \underline{\underline{11111001}}
 \end{array}$$

$$\begin{array}{r}
 2's Com of 13 \quad 00001101 \\
 000000101 \\
 11111000 \\
 +1 \\
 \hline
 1111001
 \end{array}
 \quad
 \begin{array}{r}
 11110010 \\
 +1 \\
 \hline
 11110011
 \end{array}$$

718119

## Subtraction :-

$$\text{Ex: } -1) (-6) - (-13)$$

$$-6 + 13$$

1

Take 2's complement of 6

$$\begin{array}{r}
 1111 \\
 1111010 \\
 + 00001101 \\
 \hline
 00000111
 \end{array}$$

discard  
carry

$$2) (-6) - (+13)$$

$$-6 - 13$$

1 1

Take 2's complement of both

$$\begin{array}{r}
 1111 \\
 1111010 \\
 11110011 \\
 \hline
 00001101
 \end{array}$$

discard  
carry

-1 means -ve

Combined circuit building block:

 $\oplus^3$  Operations :-

$$X \oplus Y = X\bar{Y} + \bar{X}Y \Rightarrow \text{---}$$

$$X \odot Y = XY + \bar{X}\bar{Y} \Rightarrow \text{---}$$

$$X \oplus Y \oplus Z = \bar{X}\bar{Y}Z + \bar{X}Y\bar{Z} + X\bar{Y}\bar{Z} + XYZ$$

## Half Adder :-

Addition 2 bits :-

I/p      o/p

x    y    c    s

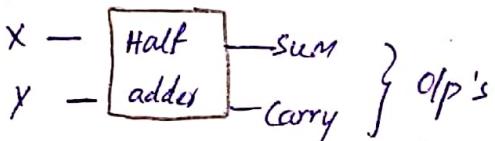
0    0    0    0

0    1    0    1 ←

1    0    0    1 ←

1    1    1    0

II/p's



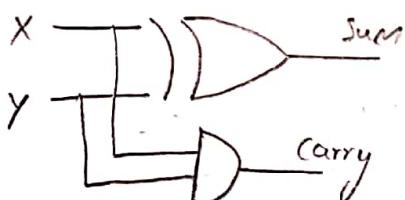
Boolean function :-

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

$$S = x \oplus y$$

$$C = x \cdot y$$

Logic diagram



## Full Adder :-

Addition of 3 bits

x    y    z    c    s

0    0    0    0    0

0    0    1    0    1

0    1    0    0    1

0    1    1    1    0

1    0    0    0    1

1    0    1    1    0

1    1    0    1    0

| $x$ | $y$ | $z$ | $c$ | $s$ |
|-----|-----|-----|-----|-----|
| 1   | 1   | 1   | 1   | 1   |

Boolean expression

$$S = \bar{x}\bar{y}z + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xyz$$

$$S = x \oplus y \oplus z$$

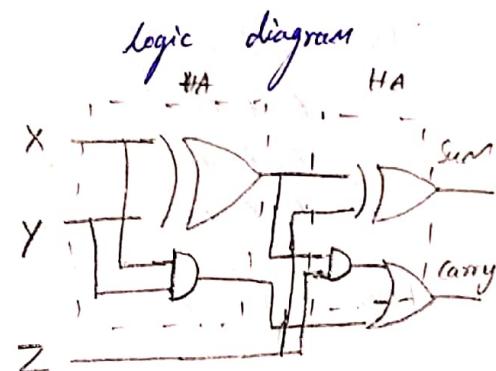
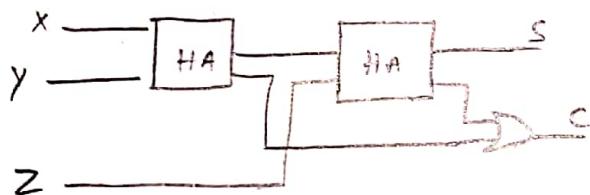
$$C = \bar{x}yz + x\bar{y}z + xy\bar{z} + xyz$$

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

$$\Rightarrow C = Z(x \oplus y) + xy$$

$$C = Z(x \oplus y) + xy$$

(or)



### Decoders ..

$n$  inputs -  $2^n$  outputs

- i) Draw the functional diagram for 3 to 8 line decoder

| $A_2$ | $A_1$ | $A_0$ | $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 1     |
| 0     | 0     | 1     | 0     | 0     | 0     | 0     | 0     | 0     | 1     | 0     |
| 0     | 1     | 0     | 0     | 0     | 0     | 0     | 0     | 1     | 0     | 0     |
| 0     | 1     | 1     | 0     | 0     | 0     | 0     | 1     | 0     | 0     | 0     |
| 1     | 0     | 0     | 0     | 0     | 0     | 1     | 0     | 0     | 0     | 0     |
| 1     | 0     | 1     | 0     | 0     | 1     | 0     | 0     | 0     | 0     | 0     |
| 1     | 1     | 0     | 0     | 1     | 0     | 0     | 0     | 0     | 0     | 0     |
| 1     | 1     | 1     | 1     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

918119 Boolean expression:

$$D_0 = \bar{A}_2 \bar{A}_1 \bar{A}_0$$

$$D_1 = \bar{A}_2 \bar{A}_1 A_0$$

$$D_2 = \bar{A}_2 A_1 \bar{A}_0$$

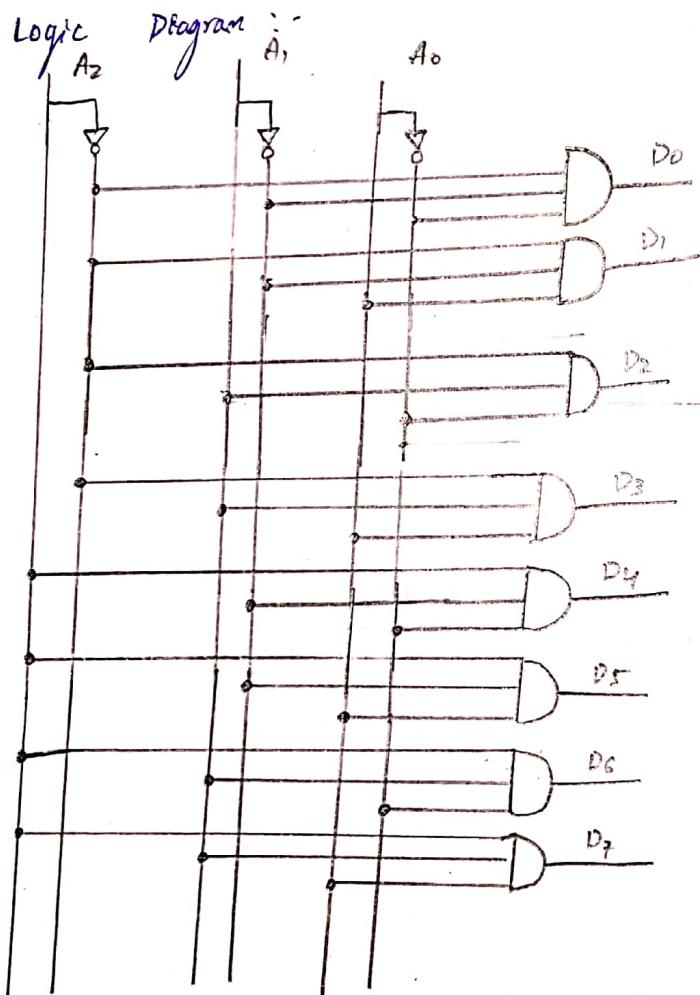
$$D_3 = \bar{A}_2 A_1 A_0$$

$$D_4 = A_2 \bar{A}_1 \bar{A}_0$$

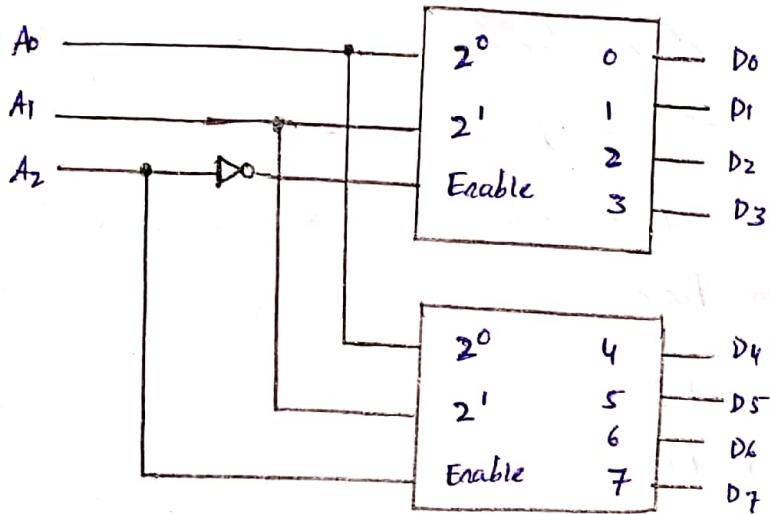
$$D_5 = A_2 \bar{A}_1 A_0$$

$$D_6 = A_2 A_1 \bar{A}_0$$

$$D_7 = A_2 A_1 A_0$$



1) Draw 3x8 decoder using 2x4 decoder only



When \$A\_2 = 0\$ the upper decoder enabled and the lower decoder is disabled. The outputs of the lower decoder becomes inactive and equal to zero. the upper decoder generates the minterms \$D\_0, D\_1, D\_2, D\_3\$ using the values \$A\_0 \& A\_1\$.

\$\rightarrow\$ When \$A\_2 = 1\$ the enabled conditions are reversed. minterms \$D\_4, D\_5, D\_6, D\_7\$ are generated.

### Encoder:-

The reverse operations of decoder

ex:- octal to Binary (8 ips) f (3 ips)

| \$D_7\$ | \$D_6\$ | \$D_5\$ | \$D_4\$ | \$D_3\$ | \$D_2\$ | \$D_1\$ | \$D_0\$ | \$A_2\$ | \$A_1\$ | \$A_0\$ |
|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| 0       | 0       | 0       | 0       | 0       | 0       | 0       | 1       | 0       | 0       | 0       |
| 0       | 0       | 0       | 0       | 0       | 0       | 1       | 0       | 0       | 0       | 1       |
| 0       | 0       | 0       | 0       | 0       | 1       | 0       | 0       | 0       | 1       | 0       |
| 0       | 0       | 0       | 0       | 1       | 0       | 0       | 0       | 0       | 1       | 1       |
| 0       | 0       | 0       | 1       | 0       | 0       | 0       | 0       | 1       | 0       | 0       |
| 0       | 0       | 1       | 0       | 0       | 0       | 0       | 0       | 1       | 0       | 1       |
| 0       | 1       | 0       | 0       | 0       | 0       | 0       | 0       | 1       | 1       | 0       |
| 1       | 0       | 0       | 0       | 0       | 0       | 0       | 0       | 1       | 1       | 1       |

| \$D_7\$ | \$D_6\$ | \$D_5\$ | \$D_4\$ | \$D_3\$ | \$D_2\$ | \$D_1\$ | \$D_0\$ | \$A_2\$ | \$A_1\$ | \$A_0\$ |
|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| 0       | 0       | 0       | 0       | 0       | 0       | 0       | 1       | 0       | 0       | 0       |
| 0       | 0       | 0       | 0       | 0       | 0       | 1       | 0       | 0       | 0       | 1       |
| 0       | 0       | 0       | 0       | 0       | 1       | 0       | 0       | 0       | 1       | 0       |
| 0       | 0       | 0       | 0       | 1       | 0       | 0       | 0       | 0       | 1       | 1       |
| 0       | 0       | 0       | 1       | 0       | 0       | 0       | 0       | 1       | 0       | 0       |
| 0       | 0       | 1       | 0       | 0       | 0       | 0       | 0       | 1       | 0       | 1       |
| 0       | 1       | 0       | 0       | 0       | 0       | 0       | 0       | 1       | 1       | 0       |
| 1       | 0       | 0       | 0       | 0       | 0       | 0       | 0       | 1       | 1       | 1       |

- An encoder is the digital function that performs the inverse operation of a decoder.
- Encoder has  $2^n$  i/p lines and  $n$  o/p lines.
- It is assumed that only one i/p has a value of 1 at any given time.

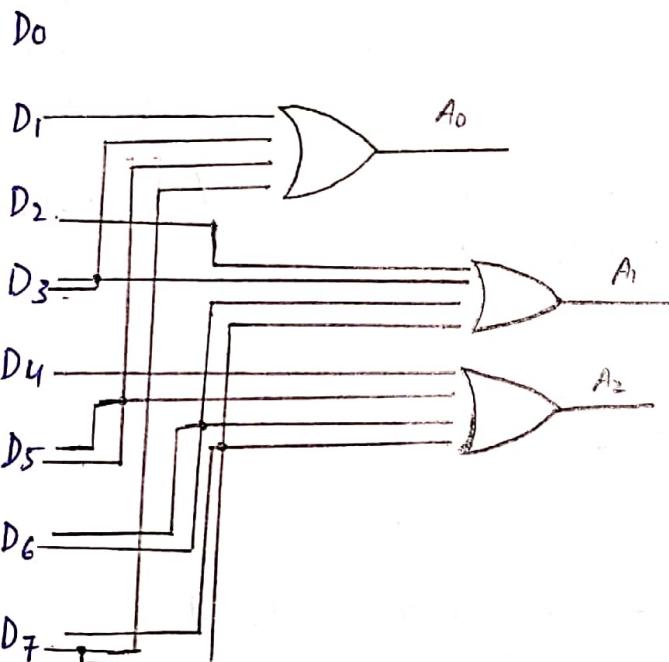
Boolean exp:

$$A_0 = D_1 + D_3 + D_5 + D_7$$

$$A_1 = D_2 + D_3 + D_6 + D_7$$

$$A_2 = D_4 + D_5 + D_6 + D_7$$

Logic diagram :-



Difference b/w Encoder and decoder:-

| parameters   | Encoder                     | decoder                          |
|--------------|-----------------------------|----------------------------------|
| input lines  | $2^n$                       | $n$                              |
| output lines | $n$                         | $2^n$                            |
| operation    | Simple                      | Complex                          |
| Applications | Email,<br>Video<br>encoders | Microprocessors,<br>Memory chips |

## Priority Encoder :-

A priority encoder is a combinational circuit that implements a priority. The operation of priority encoder is such that if 2 or more i/p's are equal to 1 at the same time, then the i/p having the highest priority take precedence.

4 i/p priority encoder

| Inputs         |                |                |                | outputs        |                |   |  |
|----------------|----------------|----------------|----------------|----------------|----------------|---|--|
| D <sub>3</sub> | D <sub>2</sub> | D <sub>1</sub> | D <sub>0</sub> | A <sub>1</sub> | A <sub>0</sub> | V |  |
| 0              | 0              | 0              | 0              | x              | x              | 0 | D <sub>3</sub> = highest priority<br>0/p of A <sub>1</sub> , A <sub>0</sub> = 11 |
| 0              | 0              | 0              | 1              | 0              | 0              | 1 |  |
| 0              | 0              | 1              | x              | 0              | 1              | 1 |  |
| 0              | 1              | x              | x              | 1              | 0              | 1 |  |
| 1              | x              | x              | x              | 1              | 1              | 1 |  |

1) i/p D<sub>3</sub> has highest priority so regardless of the values of other i/p's when D<sub>3</sub> i/p is 1 the o/p for A<sub>1</sub>, A<sub>0</sub> is 1,1 from this obtain the last row of the table.

2) D<sub>2</sub> has the next priority level. the o/p is 1,0 if D<sub>2</sub> is 1, provided that D<sub>3</sub>=0 regardless of the values of the lower priority i/p from this we obtained the 4<sup>th</sup> row of the table.

3) the valid o/p designated by V is said to be 1 only when 1 or more of the i/p's are equal to 1. if all the i/p's are zero then V=0 and the 2 o/p's A<sub>1</sub> & A<sub>0</sub> are not used (X) and are specified as don't care conditions in the output part of the table.

16/8/19

K-Map :-

|  |  | $D_1 D_0$ | 00 | 01 | 11 | 10 |
|--|--|-----------|----|----|----|----|
|  |  | $D_3 D_2$ | 00 | X  | -  | -  |
|  |  | 00        | 1  | 1  | 1  | 1  |
|  |  | 01        | 1  | 1  | 1  | 1  |
|  |  | 11        | 1  | 1  | 1  | 1  |
|  |  | 10        | 1  | 1  | 1  | 1  |

$$A_1 = D_2 + D_3$$

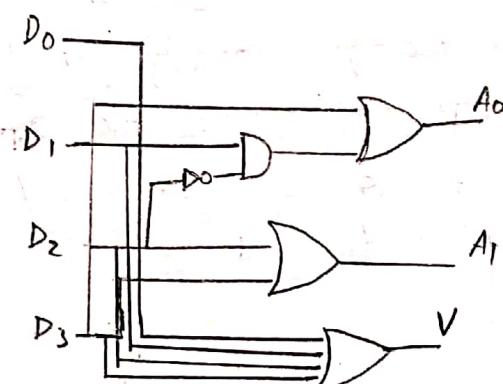
$$A_0 = A_0 = 3^{\text{th}} \text{ row } 2^1 = 2 \quad A_0 = 5^{\text{th}} \text{ row } 2^3 = 8$$

|  |  | $D_1 D_0$ | 00 | 01 | 11 | 10 |
|--|--|-----------|----|----|----|----|
|  |  | $D_3 D_2$ | 00 | X  | 1  | 1  |
|  |  | 00        | 1  |    |    |    |
|  |  | 01        |    | 1  |    |    |
|  |  | 11        | 1  | 1  | 1  | 1  |
|  |  | 10        | 1  | 1  | 1  | 1  |

$$A_0 = D_1 \bar{D}_2 + D_3$$

$$V = D_0 + D_1 + D_2 + D_3$$

Logic Diagram :-



| $A_1 = 4^{\text{th}} \text{ row } 2^2$ |       |       |       | $A_1 = 5^{\text{th}} \text{ row } 2^2$ |       |       |       |
|--|-------|-------|-------|--|-------|-------|-------|
| $D_3$                                  | $D_2$ | $D_1$ | $D_0$ | $D_3$                                  | $D_2$ | $D_1$ | $D_0$ |
| 0                                      | 1     | 0     | 0     | 1                                      | 0     | 0     | 0     |
| 0                                      | 1     | 0     | 1     | 1                                      | 0     | 0     | 1     |
| 0                                      | 1     | 1     | 0     | 1                                      | 0     | 1     | 0     |
| 0                                      | 1     | 1     | 1     | 1                                      | 0     | 1     | 1     |
| 1                                      | 0     | 0     | 0     | 1                                      | 1     | 0     | 0     |
| 1                                      | 0     | 0     | 1     | 1                                      | 0     | 1     | 1     |
| 1                                      | 1     | 0     | 0     | 1                                      | 0     | 1     | 1     |
| 1                                      | 1     | 0     | 1     | 1                                      | 0     | 0     | 1     |
| 1                                      | 1     | 1     | 0     | 1                                      | 0     | 0     | 0     |
| 1                                      | 1     | 1     | 1     | 1                                      | 0     | 0     | 1     |

| $A_0 = 3^{\text{th}} \text{ row } 2^1$ |       |       |       | $A_0 = 5^{\text{th}} \text{ row } 2^3$ |       |       |       |
|--|-------|-------|-------|--|-------|-------|-------|
| $D_3$                                  | $D_2$ | $D_1$ | $D_0$ | $D_3$                                  | $D_2$ | $D_1$ | $D_0$ |
| 0                                      | 0     | 1     | 0     | 0                                      | 0     | 1     | 1     |
| 0                                      | 0     | 1     | 1     | 0                                      | 0     | 1     | 0     |
| 1                                      | 1     | 0     | 0     | 0                                      | 0     | 1     | 0     |
| 1                                      | 1     | 0     | 1     | 1                                      | 0     | 1     | 1     |
| 1                                      | 1     | 1     | 0     | 1                                      | 0     | 0     | 0     |
| 1                                      | 1     | 1     | 1     | 1                                      | 0     | 0     | 1     |
| 1                                      | 0     | 0     | 0     | 1                                      | 0     | 0     | 1     |
| 1                                      | 0     | 0     | 1     | 1                                      | 0     | 1     | 0     |
| 1                                      | 0     | 1     | 0     | 1                                      | 0     | 1     | 0     |
| 1                                      | 0     | 1     | 1     | 1                                      | 0     | 0     | 1     |

 $\bar{D}_2 D_1$  $D_3$

## Multiplexer (MUX) :-

A multiplexer is the combinational circuit that selects binary information from 1 of many input line and directs the information to a single output line. The selection of a particular input line is controlled by a set of input variables called selection inputs.

There are  $2^n$  input lines and n selection inputs whose input combination determines which input is selected.

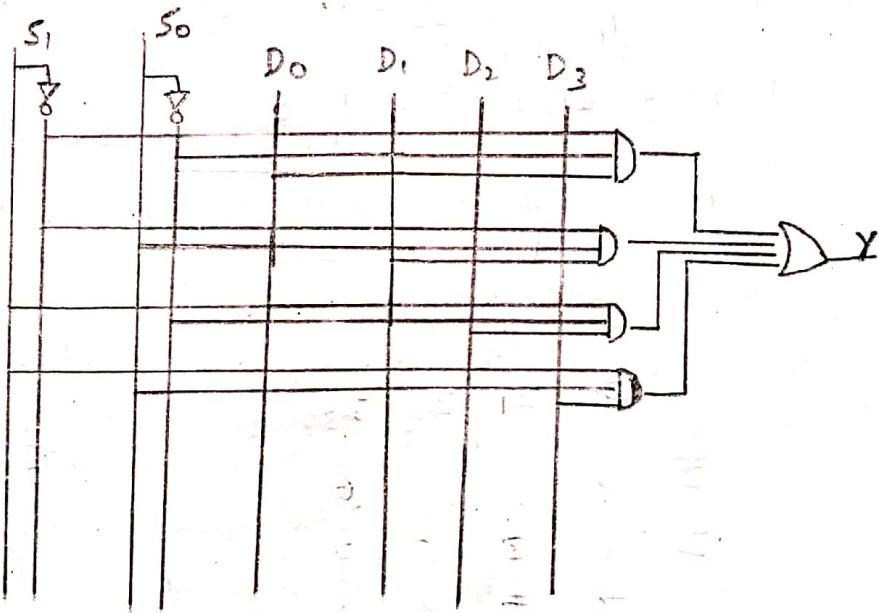
Ex:- 4 to 1 Line multiplexer (MUX)

The o/p's of AND gates are applied to a single OR gate to provide 1 line o/p.

Function Table :-

| $S_1$ | $S_0$ | $Y$   |
|-------|-------|-------|
| 0     | 0     | $D_0$ |
| 0     | 1     | $D_1$ |
| 1     | 0     | $D_2$ |
| 1     | 1     | $D_3$ |

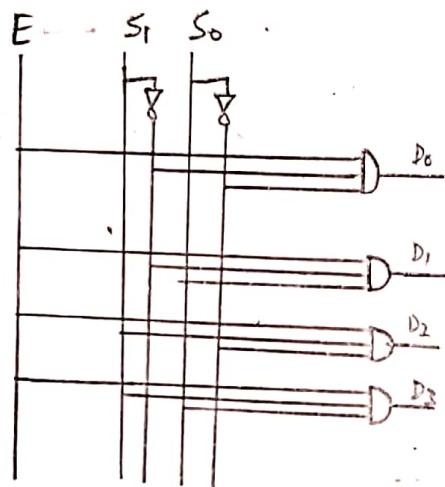
Selection i/p's :-



## De Multiplexer :-

A DeMultiplexer is a digital function that performs the inverse operation of Multiplexer i.e., A demultiplexer receives information from a single line and transmits it to one of  $2^n$  o/p lines.

Ex:- 1:4 line Demultiplexer



19/8/19

Q1 <sup>S2</sup> Implement the function  $F(x,y,z) = \sum m(1,2,6,7)$  using 4 to 1 line multiplexer

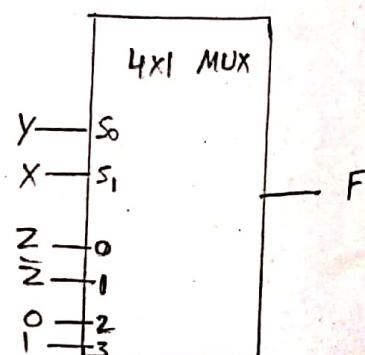
Sol 4 to 1 MUX

4 'ip'  $\Rightarrow$  2<sup>2</sup> select ip's (x,y)

Remaining inputs  $\Rightarrow 3-2=1$  (z)

no. of variables      select ip's  
given in ques

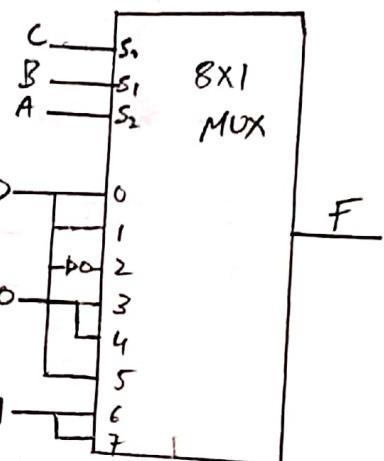
|           |           |       |       |
|-----------|-----------|-------|-------|
| $z_0$     | $z_1$     | $z_2$ | $z_3$ |
| $\bar{z}$ | 0         | 2     | 4     |
| $z$       | 1         | 3     | 5     |
| $z$       | $\bar{z}$ | 0     | 1     |



a) Implement the function  $F(A, B, C, D) = \sum m(1, 3, 4, 11, 12, 13, 14, 15)$  using 8 to 1 MUX.

Sol  $8 \rightarrow 2$  sel i/p's (A, B, C)

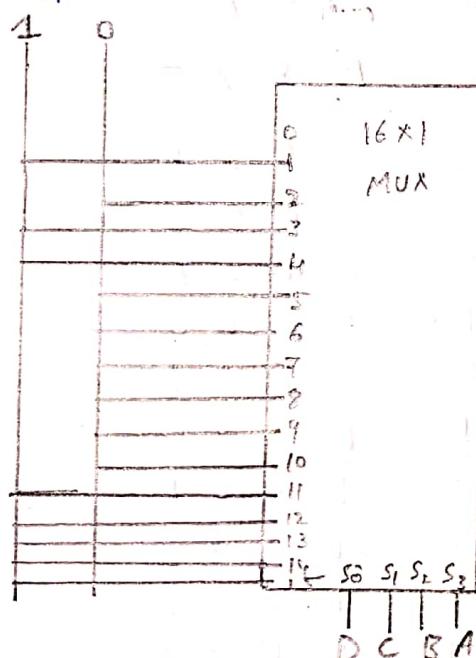
|   | D <sub>0</sub> | D <sub>1</sub> | D <sub>2</sub> | D <sub>3</sub> | D <sub>4</sub> | D <sub>5</sub> | D <sub>6</sub> | D <sub>7</sub> |
|---|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| D | 0              | 2              | ④              | 6              | 8              | 10             | ⑫              | ⑭              |
| D | ①              | ③              | 5              | 7              | 9              | ⑪              | ⑬              | ⑮              |
| D | D              | D              | D̄             | 0              | 0              | D              | 1              | 1              |



a) Implement the function  $F(A, B, C, D) = \sum m(1, 3, 4, 11, 12, 13, 14, 15)$  using 16 to 1 MUX.

Sol 16 to 1 MUX

16 i/p's  $\Rightarrow 2^4$  - select i/p's



b) Implement the function  $F(w, x, y, z) = \sum m(0, 1, 4, 7, 9, 13, 15)$

using 4:1 MUX

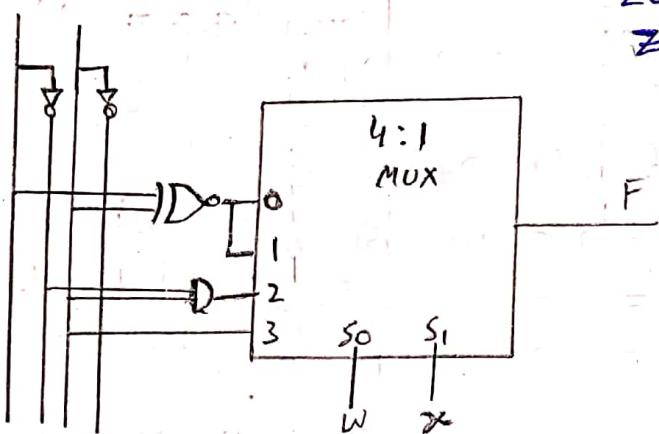
Sol 4 to 1 MUX

4 i/p's  $\Rightarrow 2^2$  - select i/p's (w, x)

Remaining i/p's  $\Rightarrow 4 - 2 = 2$  (y, z)

|            | Z <sub>0</sub> | Z <sub>1</sub> | Z <sub>2</sub> | Z <sub>3</sub> |
|------------|----------------|----------------|----------------|----------------|
| $\bar{y}z$ | ②              | ④              | ⑧              | ⑫              |
| $\bar{y}z$ | 1              | 5              | ⑨              | ⑬              |
| $y\bar{z}$ | 2              | 6              | 10             | ⑭              |
| $yz$       | ③              | ⑦              | 11             | ⑮              |

$$\begin{array}{llll}
 \bar{y}\bar{z} + yz & \bar{y}\bar{z} + yz & \bar{y}z & \bar{y}z + yz \\
 y\bar{z} & y\bar{z} & \bar{y}z & = (y + \bar{y}) \\
 & & & = z(C_1)
 \end{array}$$



~~AS3~~  
Code Converters :-

BCD to Excess - 3 (X<sub>S3</sub>)

Truth table :-

| Decimal digit | Input (BCD) |   |   |   | Output (Excess-3) |   |   |
|---------------|-------------|---|---|---|-------------------|---|---|
|               | A           | B | C | D | W                 | X | Y |
| 0             | 0           | 0 | 0 | 0 | 0                 | 0 | 1 |
| 1             | 0           | 0 | 0 | 1 | 0                 | 1 | 0 |
| 2             | 0           | 0 | 1 | 0 | 0                 | 1 | 0 |
| 3             | 0           | 0 | 1 | 1 | 0                 | 1 | 0 |
| 4             | 0           | 1 | 0 | 0 | 0                 | 1 | 1 |
| 5             | 0           | 1 | 0 | 1 | 1                 | 0 | 0 |
| 6             | 0           | 1 | 1 | 0 | 1                 | 0 | 1 |
| 7             | 0           | 1 | 1 | 1 | 1                 | 0 | 1 |
| 8             | 1           | 0 | 0 | 0 | 1                 | 0 | 1 |
| 9             | 1           | 0 | 0 | 1 | 1                 | 1 | 0 |

From the table 10, 11, 12, 13, 14, 15 are don't care

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

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

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

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

11/8/19

|    |    | CD | 00 | 01 | 11 | 10 |
|----|----|----|----|----|----|----|
|    |    | AB | 0  | 1  | 3  | 2  |
| 00 | 00 | 4  | 5  | 6  |    |    |
| 00 | 01 |    | 1  | 1  | 1  |    |
| 01 | 11 | X  | X  | X  | X  |    |
| 01 | 10 | 9  | 9  | 11 | 6  |    |
| 11 | 10 | 1  | 1  | X  | X  |    |

$$W = A + BC + BD$$

|    |    | CD | 00 | 01 | 11 | 10 |
|----|----|----|----|----|----|----|
|    |    | AB | 00 | 01 | 11 | 10 |
| 00 | 00 |    | 1  | 1  | 1  |    |
| 00 | 01 | 1  |    |    |    |    |
| 01 | 11 | X  |    | X  | X  | X  |
| 01 | 10 | 1  | X  | X  | X  | X  |

$$X = \bar{B}C + \bar{B}D + B\bar{C}\bar{D}$$

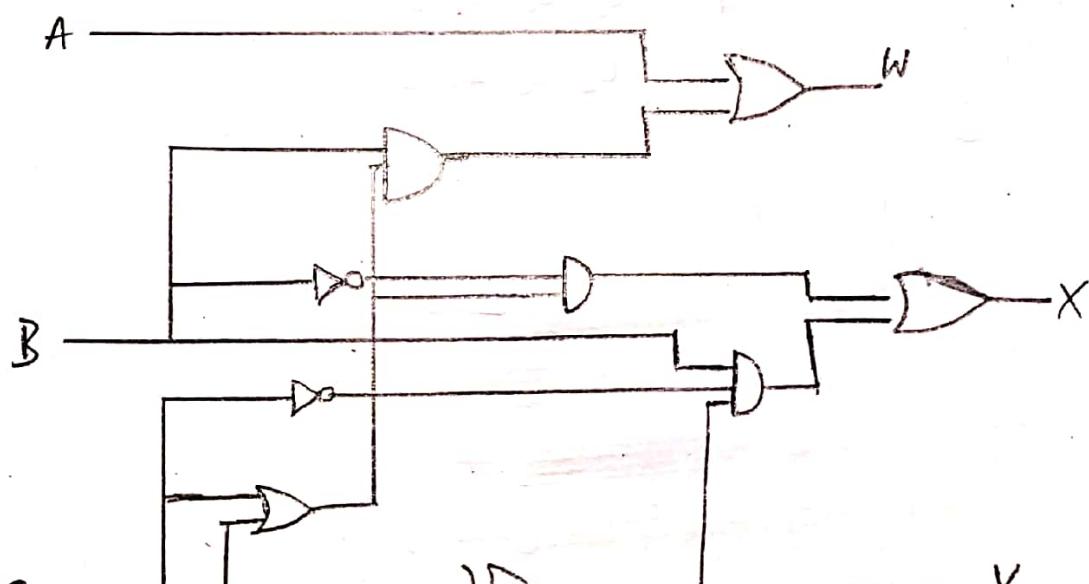
|   |   |   |   |  |  |
|---|---|---|---|--|--|
| 1 |   |   | 1 |  |  |
| 1 |   |   | 1 |  |  |
| X | X | X | X |  |  |
| 1 |   | X | X |  |  |

|   |  |   |   |   |  |
|---|--|---|---|---|--|
| 1 |  |   | 1 |   |  |
| 1 |  |   | 1 |   |  |
| X |  | X | X | X |  |
| 1 |  |   | X | X |  |

$$Y = \overline{CD} + CD$$

$$= \overline{C} \oplus D$$

$$Z = \overline{D}$$



A<sup>52</sup>  
Q) Design a BCD to decimal code converter

Sol:

| BCD Code       |                |                |                | Decimal        |                |                |                |                |                |                |                |                |                |
|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| B <sub>3</sub> | B <sub>2</sub> | B <sub>1</sub> | B <sub>0</sub> | D <sub>0</sub> | D <sub>1</sub> | D <sub>2</sub> | D <sub>3</sub> | D <sub>4</sub> | D <sub>5</sub> | D <sub>6</sub> | D <sub>7</sub> | D <sub>8</sub> | D <sub>9</sub> |
| 0              | 0              | 0              | 0              | 1              | 0              | 0              | 0              | 0              | 0              | 0              | 0              | 0              | 0              |
| 0              | 0              | 0              | 1              | 0              | 1              | 0              | 0              | 0              | 0              | 0              | 0              | 0              | 0              |
| 0              | 0              | 1              | 0              | 0              | 0              | 1              | 0              | 0              | 0              | 0              | 0              | 0              | 0              |
| 0              | 0              | 1              | 1              | 0              | 0              | 0              | 0              | 1              | 0              | 0              | 0              | 0              | 0              |
| 0              | 1              | 0              | 0              | 0              | 0              | 0              | 0              | 0              | 1              | 0              | 0              | 0              | 0              |
| 0              | 1              | 0              | 1              | 0              | 0              | 0              | 0              | 0              | 0              | 1              | 0              | 0              | 0              |
| 0              | 1              | 1              | 0              | 0              | 0              | 0              | 0              | 0              | 0              | 1              | 0              | 0              | 0              |
| 0              | 1              | 1              | 1              | 0              | 0              | 0              | 0              | 0              | 0              | 0              | 1              | 0              | 0              |
| 1              | 0              | 0              | 0              | 0              | 0              | 0              | 0              | 0              | 0              | 0              | 0              | 1              | 0              |
| 1              | 0              | 0              | 1              | 0              | 0              | 0              | 0              | 0              | 0              | 0              | 0              | 0              | 1              |

$$D_0 = \overline{B}_3 \cdot \overline{B}_2 \cdot \overline{B}_1 \cdot \overline{B}_0$$

$$D_1 = \overline{B}_3 \cdot \overline{B}_2 \cdot \overline{B}_1 \cdot B_0$$

$$D_2 = \overline{B}_3 \cdot \overline{B}_2 \cdot B_1 \cdot \overline{B}_0$$

$$D_3 = \overline{B}_3 \cdot \overline{B}_2 \cdot B_1 \cdot B_0$$

$$D_4 = \overline{B}_3 \cdot B_2 \cdot \overline{B}_1 \cdot \overline{B}_0$$

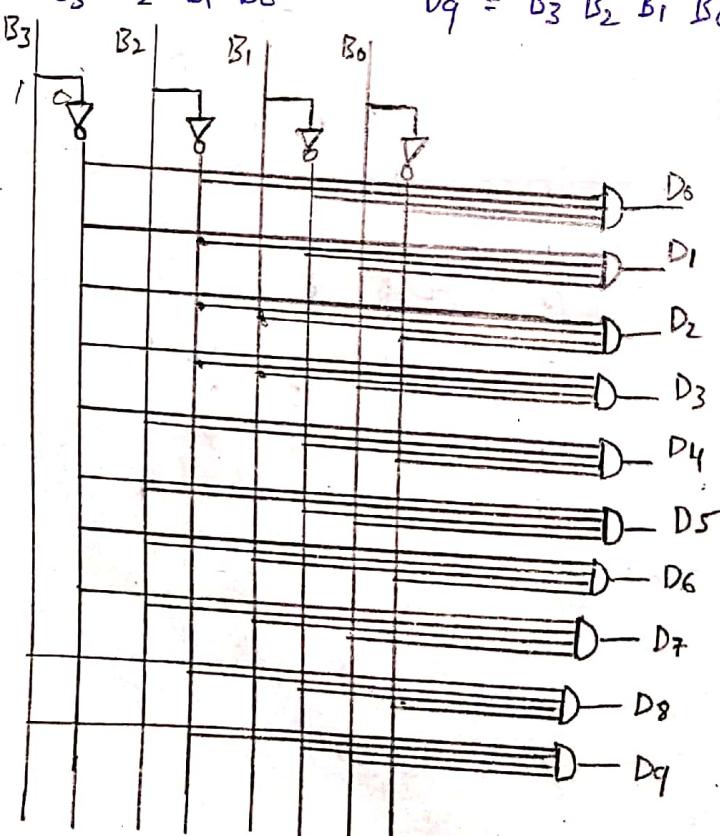
$$D_5 = \overline{B}_3 \cdot B_2 \cdot \overline{B}_1 \cdot B_0$$

$$D_6 = \overline{B}_3 \cdot B_2 \cdot B_1 \cdot \overline{B}_0$$

$$D_7 = \overline{B}_3 \cdot B_2 \cdot B_1 \cdot B_0$$

$$D_8 = B_3 \cdot \overline{B}_2 \cdot \overline{B}_1 \cdot \overline{B}_0$$

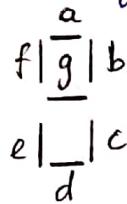
$$D_9 = B_3 \cdot \overline{B}_2 \cdot \overline{B}_1 \cdot B_0$$



Q) Design a BCD to Seven segment Decoder

Fig:- Seven Segment Display

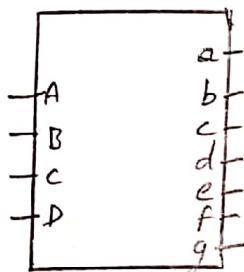
a) Segment designation b) Numeric designation for display



0 1 2 3 4

5 6 7 8 9

code converter



Truth Table :-

| BCD |   |   |   | Seven Segment Decoder |   |   |   |   |   |   |
|-----|---|---|---|-----------------------|---|---|---|---|---|---|
| A   | B | C | D | a                     | b | c | d | e | f | g |
| 0-0 | 0 | 0 | 0 | 1                     | 1 | 1 | 1 | 1 | 1 | 0 |
| 1-0 | 0 | 0 | 1 | 0                     | 1 | 1 | 0 | 0 | 0 | 0 |
| 2-0 | 0 | 1 | 0 | 1                     | 1 | 0 | 1 | 1 | 0 | 1 |
| 3-0 | 0 | 1 | 1 | 1                     | 1 | 1 | 1 | 0 | 0 | 1 |
| 4-0 | 1 | 0 | 0 | 0                     | 1 | 1 | 0 | 0 | 1 | 1 |
| 5-0 | 1 | 0 | 1 | 1                     | 0 | 1 | 0 | 1 | 1 | 1 |
| 6-0 | 1 | 1 | 0 | 1                     | 0 | 1 | 1 | 1 | 1 | 1 |
| 7-0 | 1 | 1 | 1 | 1                     | 1 | 1 | 0 | 1 | 1 | 1 |
| 8-1 | 0 | 0 | 0 | 1                     | 1 | 1 | 1 | 1 | 0 | 1 |
| 9-1 | 0 | 0 | 1 | 1                     | 1 | 1 | 1 | 1 | 1 | 1 |

All other

10, 11, 12, 13, 14, 15

i/p's

from above table  
+ 0, 11, 12, 13, 14, 15 are assigned as 0's (in d's)

$$a = \sum m(0, 2, 3, 5, 6, 7, 8, 9) + \sum d(1, 4, 1)$$

$$b = \sum m(0, 1, 2, 3, 4, 7, 8, 9) + \sum d(5, 6)$$

$$c = \sum m(0, 1, 3, 4, 5, 6, 7, 8, 9) + \sum d(2)$$

$$d = \sum m(0, 1, 2, 3, 6, 8, 9) + \sum d(1, 4, 5, 7)$$

$$e = \sum m(0, 2, \cancel{3}, 6, 8) + \sum d(1, 3, 4, \cancel{5}, 7, 9)$$

$$f = \sum m(0, 4, 5, 6, 9) + \sum d(1, 2, 3, 7)$$

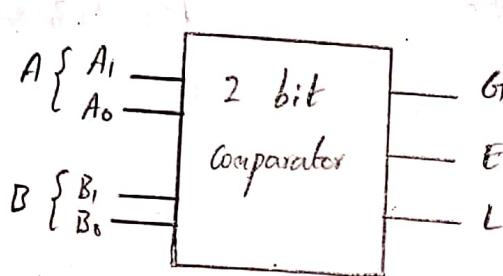
$$g = \sum m(2, 3, 4, 5, 6, 8, 9) + \sum d(0, 1, 7)$$

619119

## Arithmetic Comparison Circuits :-

## 2-bit Comparator :-

2-bit Comparator compares 2 binary numbers each of 2-bits ~~diff~~ and reduces their relation such as =, >, < the other. The figure below shows the block diagram of a 2-bit comparator which has 4 i/p's and 3 o/p's.



The first no. A is designated as  $A = A_1 A_0$

The 2nd no. B is designated as  $B = B_1 B_0$

This Comparator produces 3 o/p's as  
 $G_1$  ( $G_1 = 1$  if  $A > B$ )  
 $E$  ( $E = 1$  if  $A = B$ )  
 $L$  ( $L = 1$  if  $A < B$ )

Truth table :-

| I/P's |       |       |       | O/P's   |         |         |        |
|-------|-------|-------|-------|---------|---------|---------|--------|
| $A_1$ | $A_0$ | $B_1$ | $B_0$ | $A > B$ | $A = B$ | $A < B$ |        |
| 0     | 0     | 0     | 0     | 0       | 1       | 0       | 00 - 1 |
| 0     | 0     | 0     | 1     | 0       | 0       | 0       | 01 - 1 |
| 0     | 0     | 1     | 0     | 0       | 0       | 1       | 10 - 2 |
| 0     | 0     | 1     | 1     | 0       | 0       | 1       | 11 - 3 |
| 0     | 1     | 0     | 0     | 1       | 0       | 0       |        |
| 0     | 1     | 0     | 1     | 0       | 0       | 1       |        |
| 0     | 1     | 1     | 0     | 0       | 1       | 0       |        |
| 1     | 1     | 1     | 1     | 0       | 0       | 1       |        |
| 1     | 0     | 0     | 0     | 0       | 0       | 0       |        |
| 1     | 0     | 0     | 1     | 1       | 0       | 0       |        |
| 1     | 0     | 1     | 0     | 0       | 1       | 0       |        |
| 1     | 0     | 1     | 1     | 0       | 0       | 1       |        |
| 1     | 1     | 0     | 0     | 0       | 0       | 0       |        |
| 1     | 1     | 0     | 1     | 0       | 0       | 0       |        |
| 1     | 1     | 1     | 0     | 0       | 0       | 0       |        |
| 1     | 1     | 1     | 1     | 0       | 0       | 0       |        |

K-map :-

|                                 |  | A > B                           |    |    |    |    |
|---------------------------------|--|---------------------------------|----|----|----|----|
|                                 |  | B <sub>1</sub> , B <sub>0</sub> | 00 | 01 | 11 | 10 |
| A <sub>1</sub> , A <sub>0</sub> |  | 00                              | 00 | 00 | 00 | 00 |
| 01                              |  | 01                              | 1  | 1  | 1  | 1  |
| 11                              |  | 11                              | 1  | 1  | 1  | 1  |
| 10                              |  | 10                              | 1  | 1  |    |    |

$$\begin{array}{l}
 A_1 A_0 B_1 B_0 \\
 \hline
 11 & 00 \\
 11 & 01 \\
 11 & 10 \\
 10 & 00 \\
 10 & 01 \\
 \hline
 A_1 B_1 \\
 11\ 00 \\
 11\ 10 \\
 \hline
 A_1 A_0 B_0 \\
 01\ 00 \\
 11\ 00 \\
 \hline
 A_0 B_1 B_0
 \end{array}$$

$$G_1 = A_1 \bar{B}_1 + A_1 A_0 \bar{B}_0 + A_0 \bar{B}_1 \bar{B}_0$$

|                                 |  | A < B |    |    |    |    |
|---------------------------------|--|-------|----|----|----|----|
|                                 |  | 00    | 01 | 11 | 10 |    |
| A <sub>1</sub> , A <sub>0</sub> |  | 00    | 00 | 01 | 11 | 10 |
| 01                              |  | 01    | 1  | 1  | 1  | 1  |
| 11                              |  | 11    | 1  | 1  | 1  | 1  |
| 10                              |  | 10    |    |    |    |    |

$$\begin{array}{l}
 A_1 A_0 B_1 B_0 \\
 \hline
 00\ 01 \\
 00\ 11 \\
 \bar{A}_1 \bar{A}_0 B_1 B_0 \\
 00\ 10 \\
 01\ 11 \\
 01\ 10 \\
 \bar{A}_1 B_1 \\
 00\ 11 \\
 10\ 11 \\
 \bar{A}_0 B_1 B_0
 \end{array}$$

$$L = \bar{A}_1 B_1 + \bar{A}_1 \bar{A}_0 B_0 + \bar{A}_0 \bar{B}_1 B_0$$

A = B

|   |   |   |   |
|---|---|---|---|
| 1 |   |   |   |
|   | 1 |   |   |
|   |   | 1 |   |
|   |   |   | 1 |

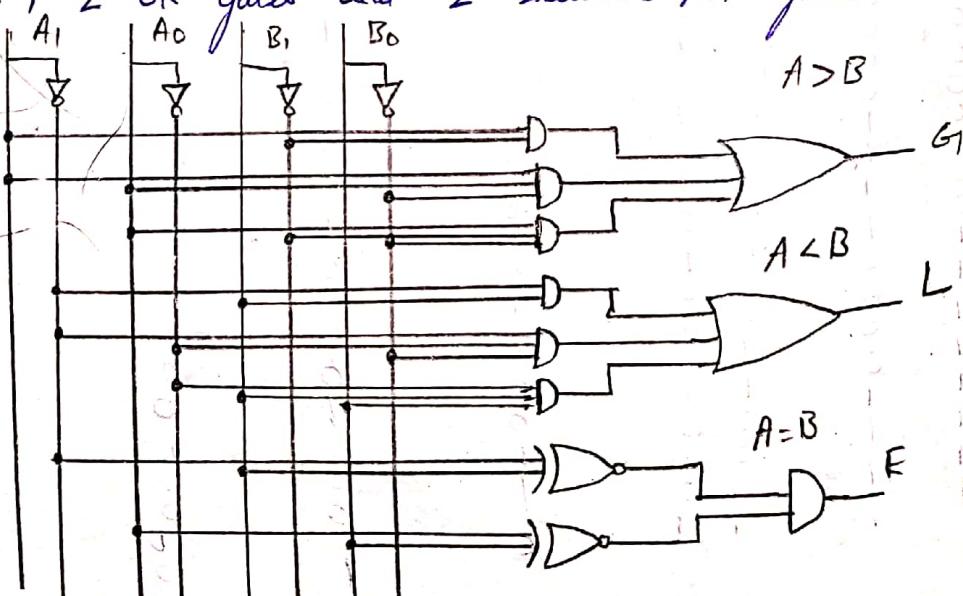
$$E = \bar{A}_1 \bar{A}_0 \bar{B}_1 \bar{B}_0 + \bar{A}_1 A_0 \bar{B}_1 B_0 + A_1 A_0 \bar{B}_1 B_0 + A_1 \bar{A}_0 B_1 \bar{B}_0$$

$$E = \bar{A}_1 \bar{B}_1 (\bar{A}_0 \bar{B}_0 + A_0 B_0) + A_1 B_1 (A_0 B_0 + \bar{A}_0 \bar{B}_0)$$

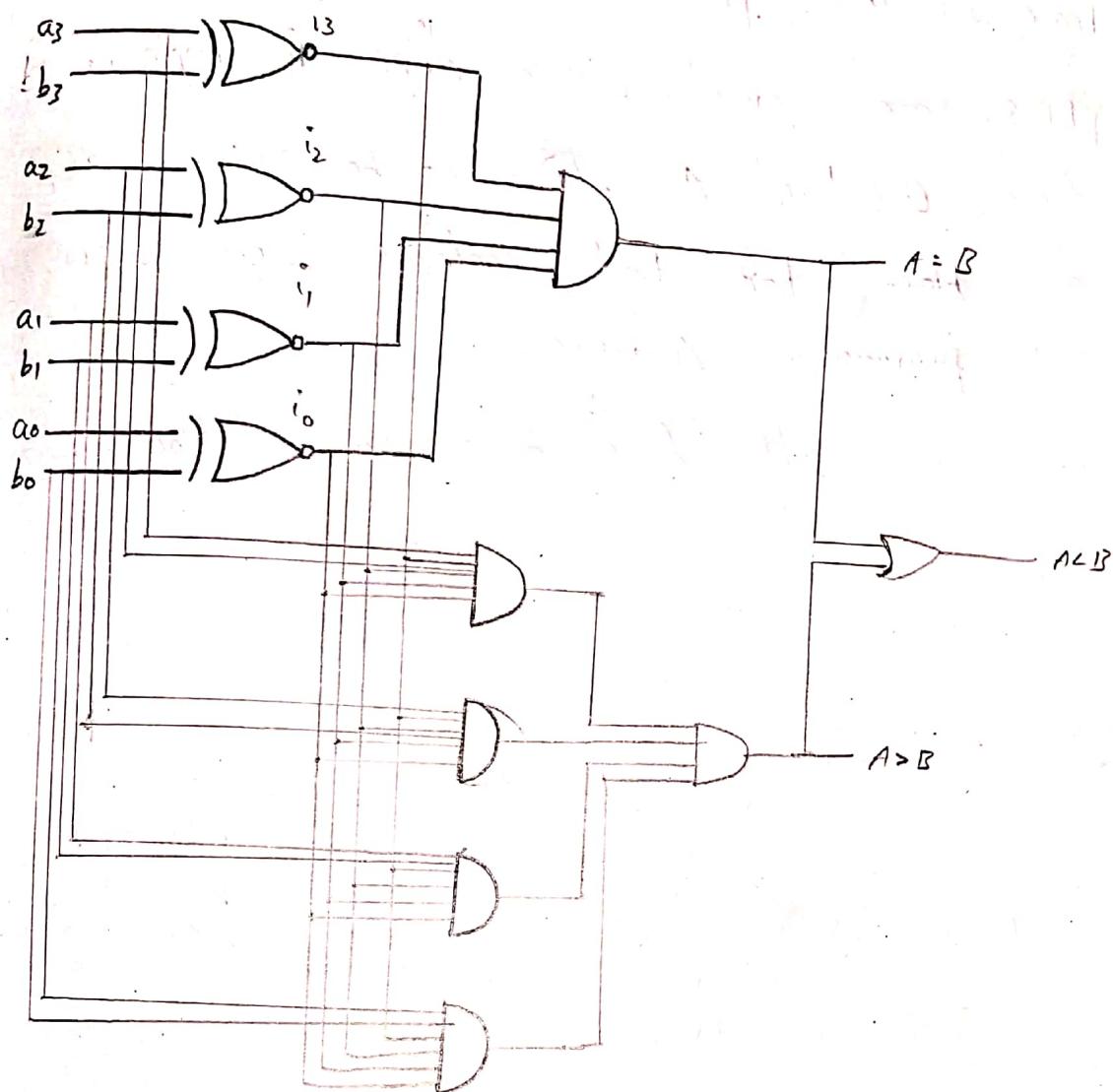
$$E = (\bar{A}_1 \bar{B}_1 + A_1 B_1) (\bar{A}_0 \bar{B}_0 + A_0 B_0)$$

$$E = (A_1 \odot B_1) (A_0 \odot B_0)$$

By using above Boolean expression for each o/p the logic diagram can be implemented by using 4 NOT gates, 7 AND gates, 2 OR gates and 2 exclusive NOR gates.



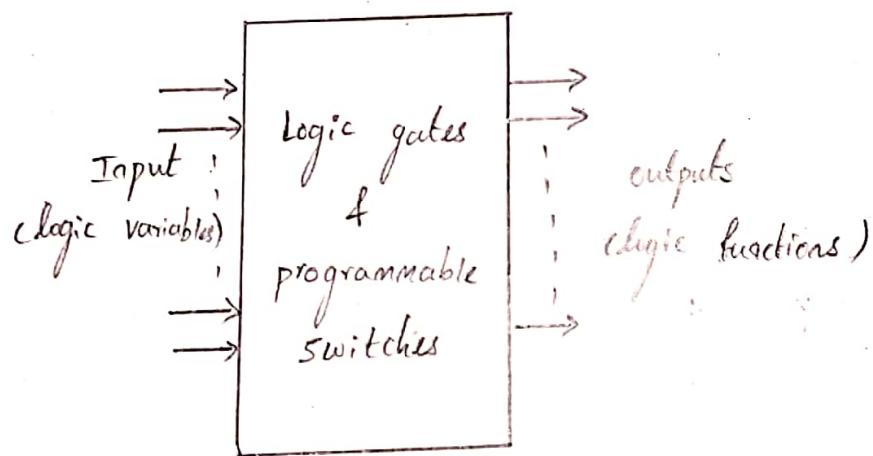
4-bit Comparator :-



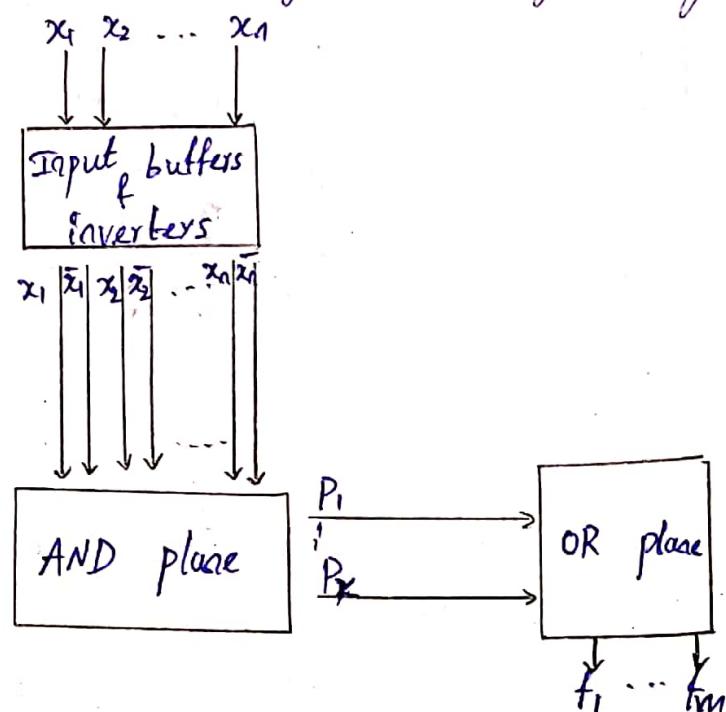
Design of combinational circuits using programmable logic devices (PLD's).

PLD's are general purpose chips for implementing logic circuitry. A PLD can be viewed as a black box that contains logic gates and programmable switches.

fig :- programmable logic device as a black box:-



General structure of programmable logic array (PLA):



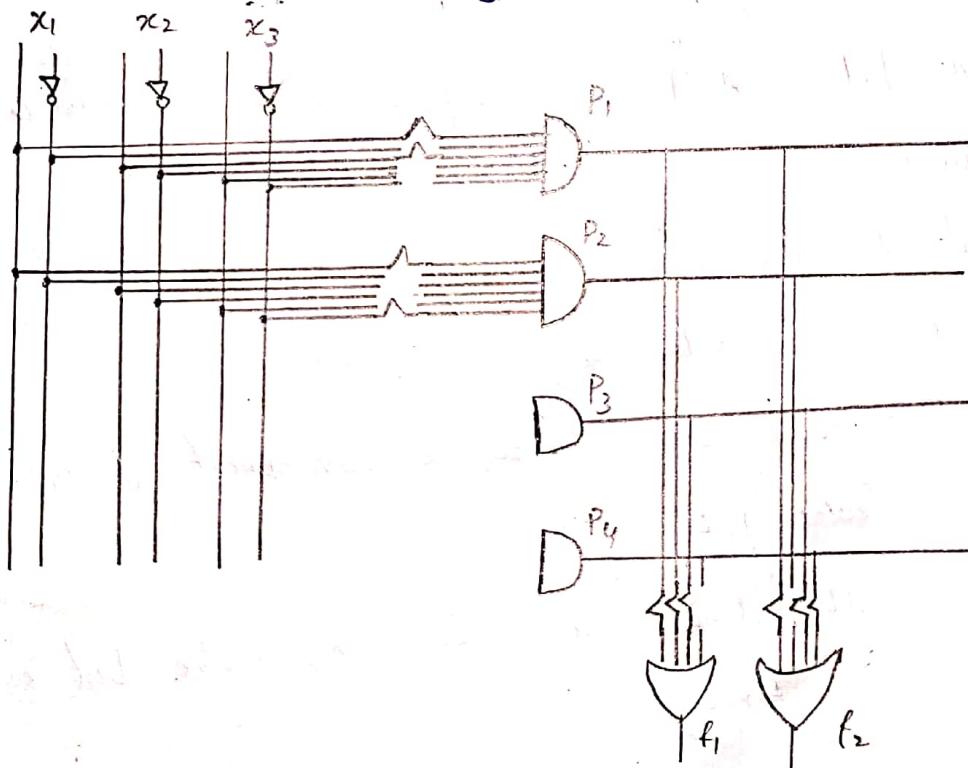
→ Several types of PLD's are available. The 1<sup>st</sup> developed was PLA. The general structure of PLA is shown in figure. The PLA i/p's  $x_1, x_2 \dots x_n$  pass through a set of buffers (which provide both true and complement of each i/p) into a circuit block called AND plane (or) AND Array.

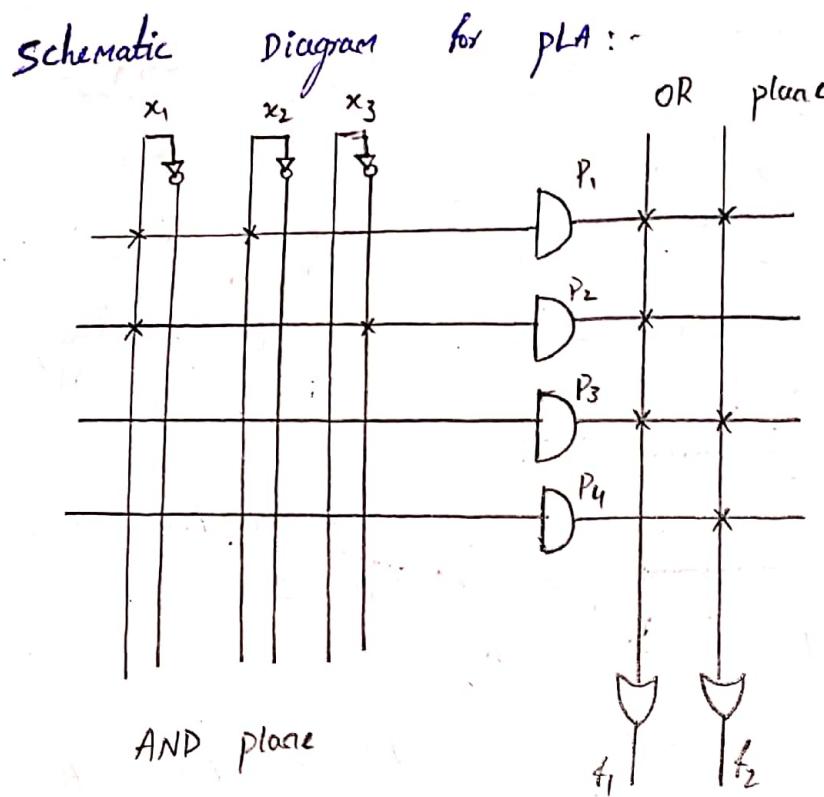
→ The AND array produces a set of product terms  $p_1, p_2, p_3, \dots$ . The product terms serve as i/p's to OR plane which produces the o/p  $f_1, f_2, \dots, f_m$ .

Q) Draw the gate level diagram of PLA for the function

$$f_1 = x_1 x_2 + x_1 \bar{x}_3 + \bar{x}_1 \bar{x}_2 x_3$$

$$f_2 = x_1 x_2 + \bar{x}_1 \bar{x}_2 x_3 + x_1 x_3$$





11/9/19

programmable Array logic (PAL) :-

In PLA both AND, OR places are programmable

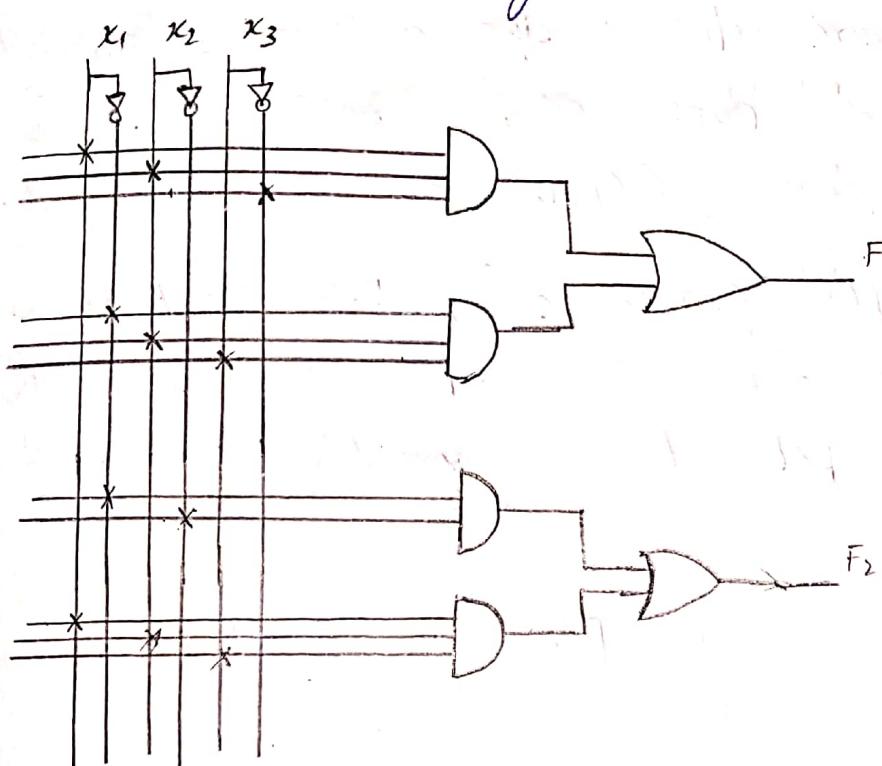
draw backs :-

- 1, Hard to fabricate correctly
  - 2, They reduce the speed performance
- These drawbacks led to the development of similarly device called 'PAL'.
- In PAL 'AND' place is programmable but 'OR' place is fixed.

Q1 Realize 2 logic functions

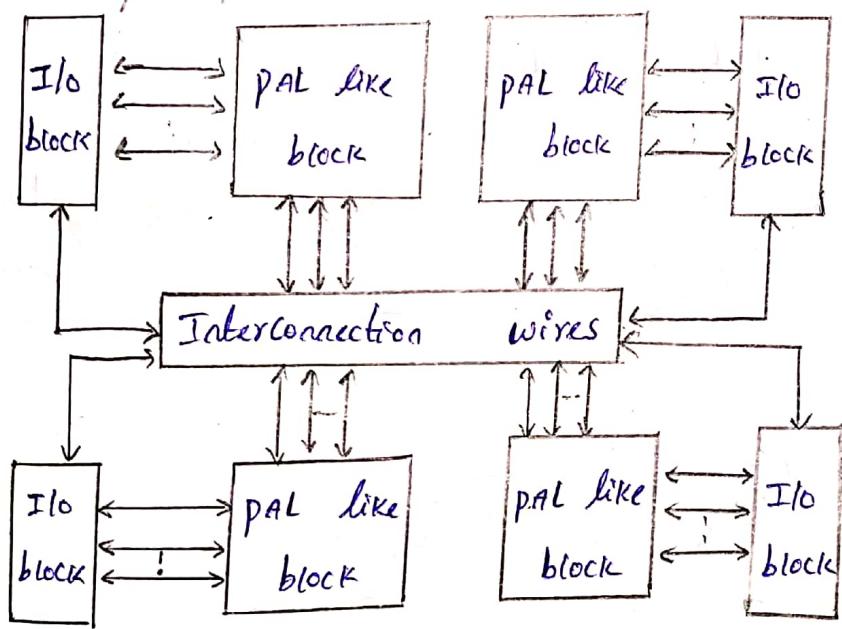
$$f_1 = x_1 x_2 \bar{x}_3 + \bar{x}_1 x_2 x_3$$

$$f_2 = \bar{x}_1 \bar{x}_2 + x_1 x_2 x_3 \text{ ... using PAL}$$



### Complex Programmable Logic Devices (CPLD's)

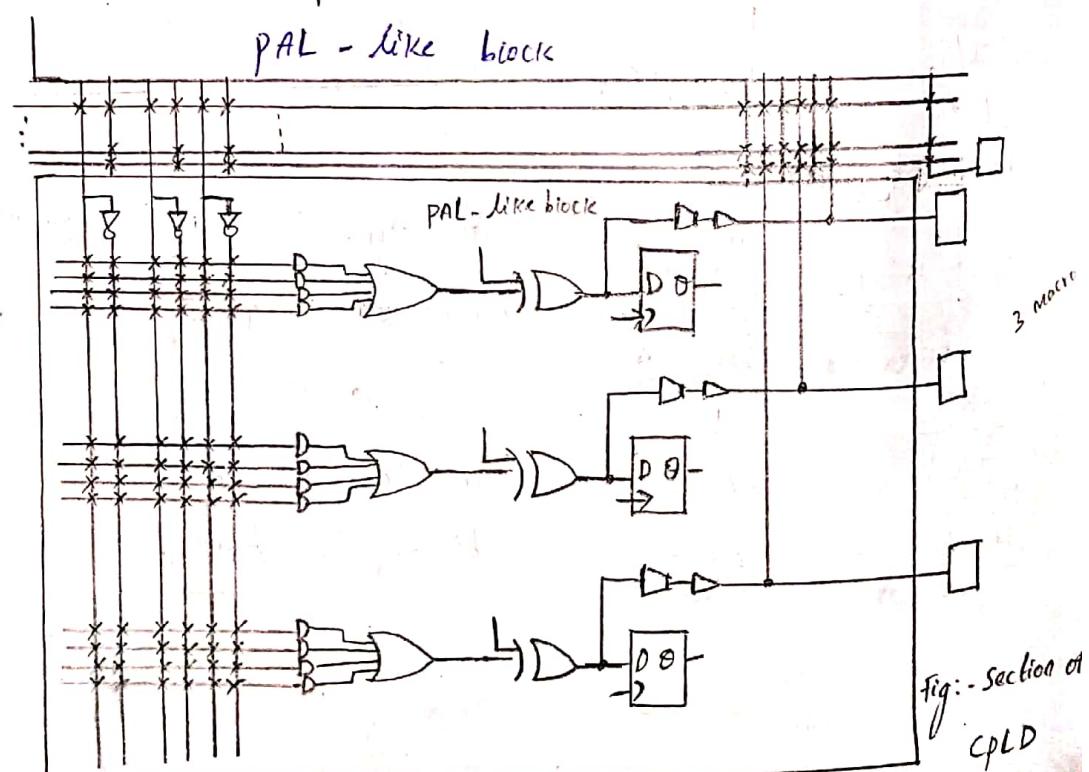
multiple circuit blocks  
in single chip



→ PLA's & PAL's chips are limited modest size supporting a combine no. of I/O's + O/P's of **not** more than 32. for implementation of circuits that require more I/O's & O/P's a more sophisticated types of chips called complex programmable logic devices (CPLD) can be used.

→ A **CPLD** Comprises of multiple circuit blocks on a single chip. Each circuit block is similar to PLA or PAL. the structure is shown in above figure.

→ It includes 4 PAL like blocks that are connected to a set of interconnection wire each PAL like block is also connected to a sub circuit called I/O block which is attached to a no. of chips I/O & O/P pins.



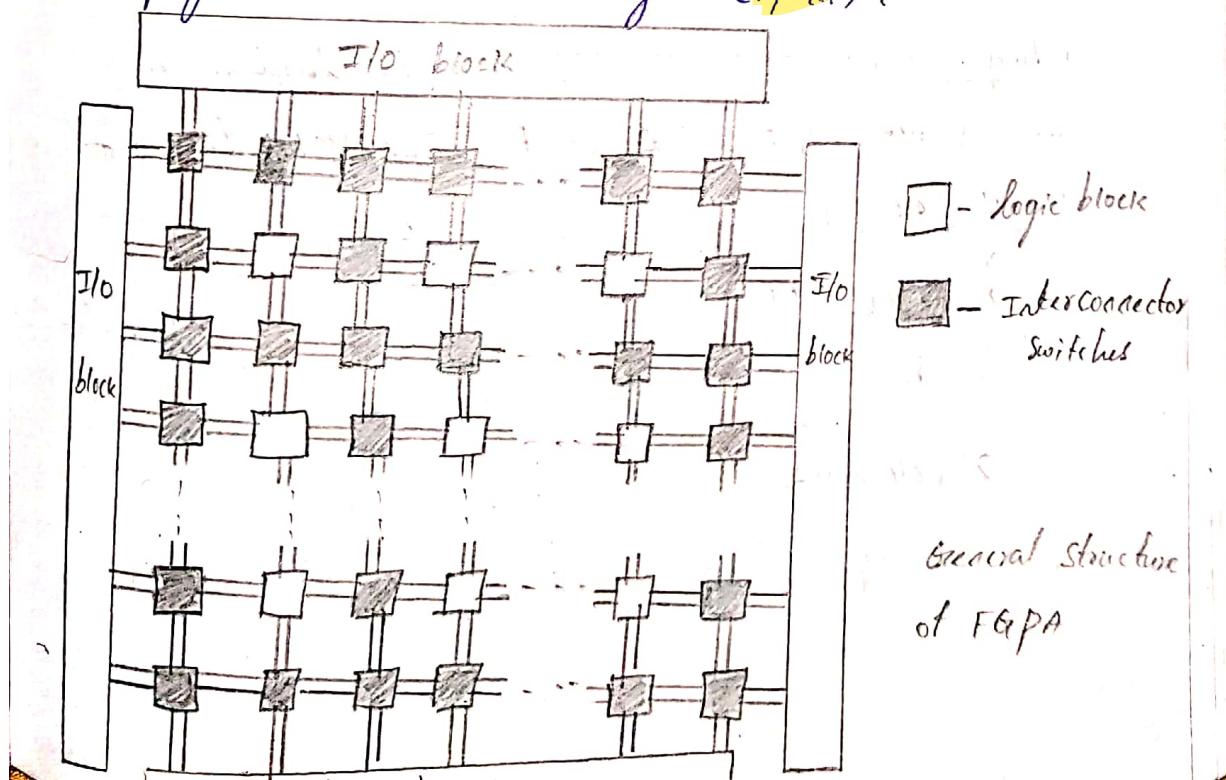
The PAL like block includes 3 macro cells each consisting of 4 ilp OR gate. The OR gate o/p is connected to exclusive OR. 1 ilp to the exclusive OR gate can be connected to 1 or 0. If connected to '1' than XOR gate complements OR gate o/p. if connected to '0' then XOR gate has no effect.

-> The macro cell also includes a flip-flop, a multiplexer and tri-state buffer. The flip-flop is used to store the o/p value produced by OR gate.

#### \* Applications of CPLD :-

- 1) Used for simple applications like address decoding.
- 2) Used in digital design to perform the functions of Boot loader.
- 3) Loading configuration data.
- 4) CPLD are ideal for high performance control applications.

#### Field programmable gate Arrays (FPGA) :-



A filled programmable gate array is a programmable logic device that supports implementation of relatively large logic circuit. FPGA do not contain AND, OR gate.

→ FPGA Contains 3 main types of resources

1, logic block

2, I/O block

3, Interconnection wires and switches

→ The logic blocks are arranged in a 2-D array  
I/O blocks are used for connecting to the pins of the package. The interconnection wires are organized as horizontal and vertical routing channel between rows and columns of logic blocks.

~~Ex:-~~ The most commonly used logic block is lookup table (LUT) which contains storage cells that are used to implement a small logic function. Each cell is capable of holding a single logic value either 0 or 1. The stored value is produced as the output of storage cell.

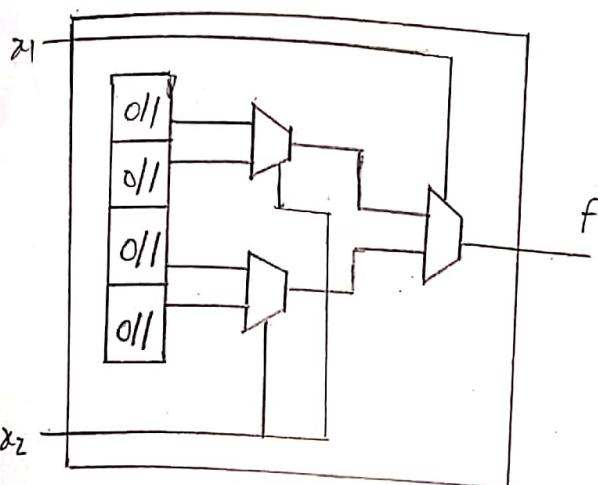
Ex:- 2 input LUT

$$f = \bar{x}_1 \bar{x}_2 + x_1 x_2$$

$$2^2 = 4 \text{ rows}$$

4 storage cells

a) circuit for a 2-ip LUT

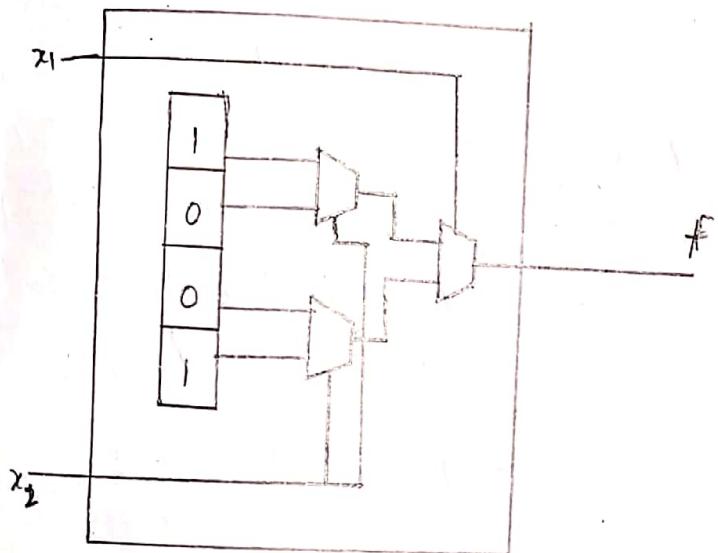


b)  $f_1 = \bar{x}_1 \bar{x}_2 + x_1 x_2$

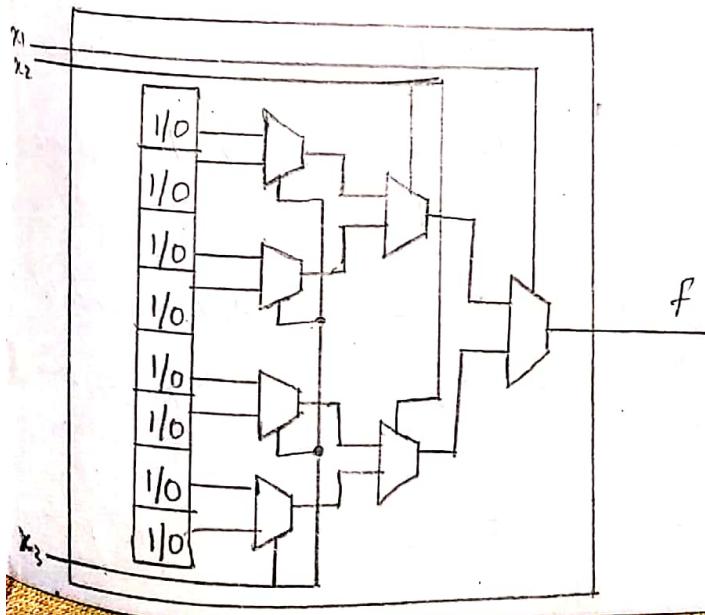
Truth table :-

| $x_1$ | $x_2$ | $f_1$ |
|-------|-------|-------|
| 0     | 0     | 1     |
| 0     | 1     | 0     |
| 1     | 0     | 0     |
| 1     | 1     | 1     |

c) Storage cell contents in LUT



Ex:- 3 input LUT



$$f = \bar{x}_1 \bar{x}_2 \bar{x}_3 + x_1 x_2 x_3$$

Truth table :-

| $x_1$ | $x_2$ | $x_3$ | $f$ |
|-------|-------|-------|-----|
| 0     | 0     | 0     | 1   |
| 0     | 0     | 1     | 0   |
| 0     | 1     | 0     | 0   |
| 0     | 1     | 1     | 0   |
| 1     | 0     | 0     | 0   |
| 1     | 0     | 1     | 0   |
| 1     | 1     | 0     | 0   |
| 1     | 1     | 1     | 1   |

16/9/19

### Differences b/w CPLD and FPGA

#### CPLD

- 1) An integrated circuit that helps to implement digital system
- 2) provides less logic resources
- 3) Cost effective
- 4) Made up of larger blocks
- 5) Uses EEPROM (not volatile)
- 6) Easier to predict delays.
- 7) Consumes low power

#### FPGA

- 1) An integrated circuit that is designed to be configured by a customer or designer after manufacturing
- 2) provides a massive amount of logic resources and storage elements to create complex systems.
- 3) Expensive than CPLD
- 4) made up of tiny logic blocks.
- 5) uses RAM (volatile)
- 6) Not easy to predict delays.
- 7) Consumes more power

- 8) More Secure
- 9) Suitable for small to medium scale applications.
- 10) It has gate array like architecture
- 11) It contains only a few blocks of logic (1000's)
- 8) less secure
- 9) suitable for complex applications. ~~If this part~~  
~~like architecture~~
- 10) It has gate array like architecture
- 11) It contains ~~lakh~~ of tiny logic blocks (100000)

### B.8. Verilog HDL :- (Hardware description language)

It is a case sensitive language all key words are in lower case

ports :-

provides a mean for a module to communicate through ip & op.

module :-

It is defined as an element or as a collection of lower level design blocks.

Verilog code for Basic gates :-

AND



Truth table :-

| a | b | c |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

code:-

```
module andl (a,b,c);  
// ports declaration  
input a,b;  
output c;  
and  a1(c,a,b);  
end module;
```

OR gate :-



TT :-

| a | b | c |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

code:-

```
module ori (a,b,c);  
// ports declaration  
input a,b;  
output c;  
or  o1(c,a,b);  
end module;
```

XOR :-



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

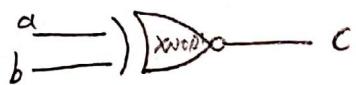
Truth Table :-

| a | b | c |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

code :-

```
module xor1(a,b,c);  
// ports declaration  
input a,b;  
output c;  
xor s,CC,a,b);  
end module;
```

XNOR :-



Truth table :-

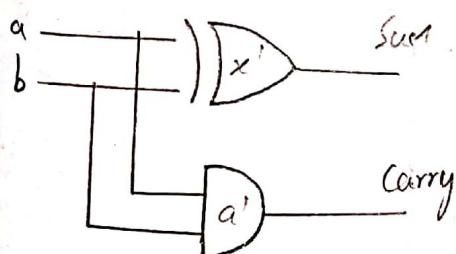
| a | b | c |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

code :-

```
module xnor(a,b,c);  
// ports declaration  
input a,b;  
output c;  
xnor s,CC,a,b);  
end module;
```

Verilog code for adders :-

Half Adder :-



$$\text{Sum} = a \oplus b$$

$$ab + \bar{a}b$$

$$\text{Carry} = ab$$

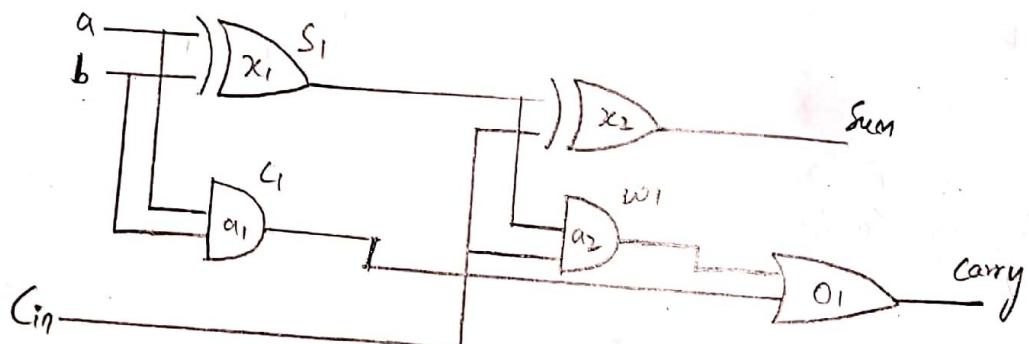
Truth Table :-

| a | b | Sum | Carry |
|---|---|-----|-------|
| 0 | 0 | 0   | 0     |
| 0 | 1 | 1   | 0     |
| 1 | 0 | 1   | 0     |
| 1 | 1 | 0   | 1     |

Code :-

```
module halfadder(a,b,sum,carry)
input (a,b);
output sum,carry;
xor x1 (sum,a,b);
and a1 (carry,a,b);
end module;
```

Full Adder :-



Truth Table :-

| a | b | Cin | Sum | Carry |
|---|---|-----|-----|-------|
| 0 | 0 | 0   | 0   | 0     |
| 0 | 0 | 1   | 1   | 0     |
| 0 | 1 | 0   | 1   | 0     |
| 0 | 1 | 1   | 0   | 0     |
| 1 | 0 | 0   | 1   | 1     |
| 1 | 0 | 1   | 0   | 0     |
| 1 | 1 | 0   | 0   | 1     |
| 1 | 1 | 1   | 1   | 1     |

$$\text{Sum} = x \oplus y \oplus \text{Cin}$$

$$\text{Carry} = \text{Cin}(x \oplus y) + xy$$

19/9/19

Code :-

```

module fulladder (a,b,cin,sum,carry);
    input (a,b,cin);
    output sum,carry;
    wire s1,c1,w1;
    xor x1 (s1,a,b);
    and a1 (c1,a,b);
    xor x2 (sum,s1,cin);
    and a2 (w1,s1,cin);
    or o1 (carry,w1,c1);
endmodule;

```

Verilog code for decoders :-

2:4 Decoder :-

Truth Table

| a | b | $y_3$ | $y_2$ | $y_1$ | $y_0$ |
|---|---|-------|-------|-------|-------|
| 0 | 0 | 0     | 0     | 0     | 1     |
| 0 | 1 | 0     | 0     | 1     | 0     |
| 1 | 0 | 0     | 1     | 0     | 0     |
| 1 | 1 | 1     | 0     | 0     | 0     |

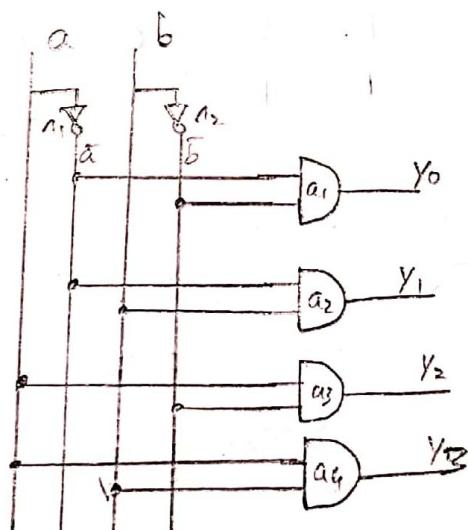
Code :-

```

module 2:4 decoder (a,b,y);
    input a,b;
    output [3:0]y;
    wire abar,bbar;
    not n1 (abar,a);
    not n2 (bbar,b);
    and o1 (y[0],abar,bbar);

```

Circuit :-



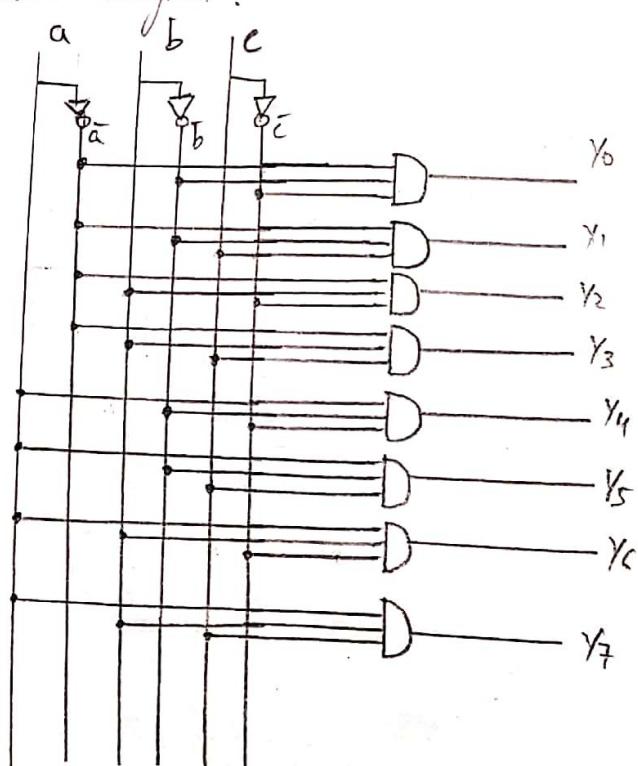
and  $a_2 [Y[1], abar, b];$   
 and  $a_3 [Y[2], a, bbar];$   
 and  $a_4 [Y[3], a, b];$   
 end module;

3:8 decoders :-

## Truth Table

| $a$ | $b$ | $c$ | $y_7$ | $y_6$ | $y_5$ | $y_4$ | $y_3$ | $y_2$ | $y_1$ | $y_0$ |
|-----|-----|-----|-------|-------|-------|-------|-------|-------|-------|-------|
| 0   | 0   | 0   | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 1     |
| 0   | 0   | 1   | 0     | 0     | 0     | 0     | 0     | 0     | 1     | 0     |
| 0   | 1   | 0   | 0     | 0     | 0     | 0     | 0     | 1     | 0     | 0     |
| 0   | 1   | 1   | 0     | 0     | 0     | 0     | 1     | 0     | 0     | 0     |
| 1   | 0   | 0   | 0     | 0     | 0     | 1     | 0     | 0     | 0     | 0     |
| 1   | 0   | 1   | 0     | 0     | 1     | 0     | 0     | 0     | 0     | 0     |
| 1   | 1   | 0   | 0     | 1     | 0     | 0     | 0     | 0     | 0     | 0     |
| 1   | 1   | 1   | 1     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

Circuit diagram :-



Code :-

```
module 3:8 decoder (a,b,c,y);
input a,b,c;
output [7:0]y;
wire abar,bbar,cbar;
not n1 (abar,a);
not n2 (bbar,b);
not n3 (cbar,c);
and a1 (y[0],abar,bbar,cbar);
and a2 (y[1],abar,bbar,c);
and a3 (y[2],abar,bbar,cbar);
and a4 (y[3],abar,b,c);
and a5 (y[4],a, bbar,cbar);
and a6 (y[5],a, bbar,c);
and a7 (y[6],a, b, cbar);
and a8 (y[7],a, b,c);
end module;
```

Universal Gates :-

i) NAND :-  $\overline{\overline{ab}}$

| AND |   | NAND |
|-----|---|------|
| a   | b | f    |
| 0   | 0 | 0    |
| 0   | 1 | 1    |
| 1   | 0 | 1    |
| 1   | 1 | 0    |

ii) NOR :-  $\overline{\overline{ab}}$

| OR |   | NOR |
|----|---|-----|
| a  | b | f   |
| 0  | 0 | 0   |
| 0  | 1 | 1   |
| 1  | 0 | 1   |
| 1  | 1 | 1   |

2019119 Using NAND GATES:-

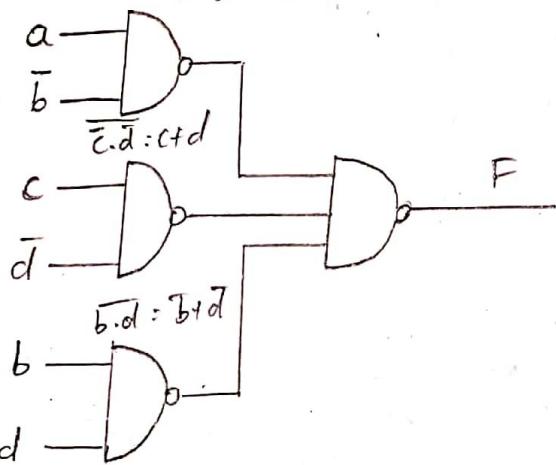
- i, In SOP, you will get one NAND gate at the last, so no need to complement the literals.
- ii, In POS, you will get two NAND gates at the last, so complement the literals.

Using NOR GATES:-

- i, In SOP, two NOR gates at the last, so complement the literals.
- ii, In POS, one OR gate at the last so no need to complement the literals

Ex:- 1,  $a\bar{b} + c\bar{d} + bd$  realize using NAND gates only

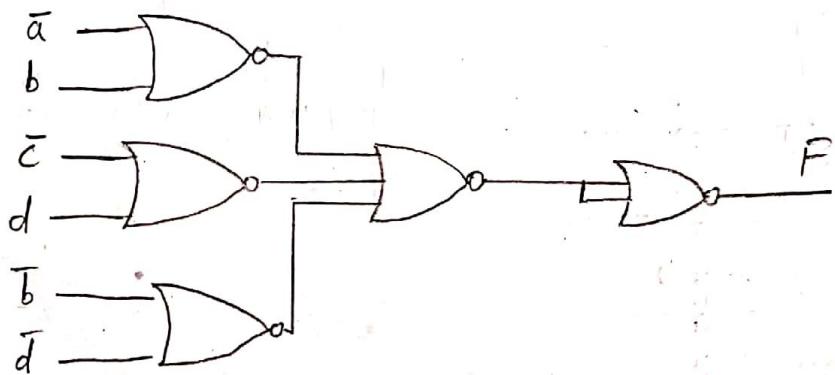
$$a \cdot \bar{b} = \bar{a} + b$$



$$(\bar{a} + b) \cdot (\bar{c} + d) \cdot (\bar{b} + d)$$

$$a \cdot \bar{b} + c \cdot \bar{d} + b \cdot \bar{d}$$

2, NOR gates only



1) find the minimal SOP & pos form for the function by map method.

$$F(a,b,c,d) = \sum m(0,2,8,9,10,15) + \sum d(1,3,6,7)$$

realize the logic circuit using NAND gates only.

Q1

|    |    | cd | 00 | 01 | 11 | 10 |
|----|----|----|----|----|----|----|
|    |    | ab | 00 | 01 | 11 | 10 |
| 00 | 00 |    | 1  | X  | X  | 1  |
|    |    |    | 4  | 5  | 7  | 6  |
| 01 | 01 |    |    | X  | X  |    |
|    |    |    | 12 | 13 | 15 | 14 |
| 11 | 10 |    | 8  | 9  | 11 | 10 |
|    |    |    | 1  | 1  | 1  | 1  |

| a | b | c | d | a | b | c | d |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |

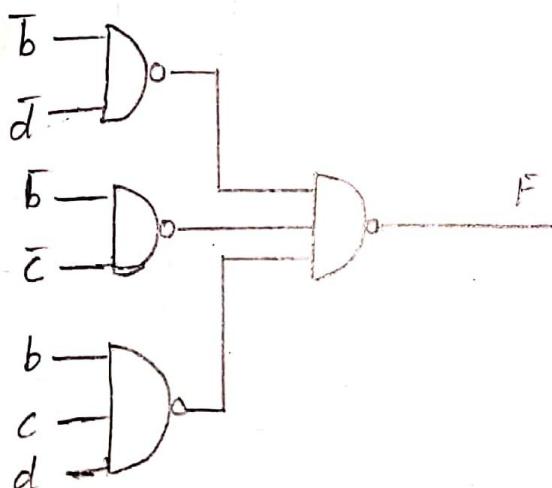
  

| a | b | c | d | a   | b | c | d |
|---|---|---|---|-----|---|---|---|
| 0 | 1 | 1 | 1 | 0   | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | -   | - | - | - |
|   |   |   |   | bcd |   |   |   |

$$SOP = F = \bar{b}\bar{d} + \bar{b}\bar{c} + bcd$$

$$POS = F = (\bar{b}+\bar{d}) \cdot (\bar{b}+\bar{c}) \cdot (\bar{b}+\bar{c}+\bar{d}).$$

NAND :- SOP



POS :- K-map

|  |    | cd | 00 | 01 | 11 | 10 |
|--|----|----|----|----|----|----|
|  |    | ab | 00 | 01 | 11 | 10 |
|  |    |    | X  | X  |    |    |
|  | 00 |    |    |    |    |    |
|  | 01 |    | 0  | 0  | X  | X  |
|  | 11 |    | 0  | 0  |    | 0  |
|  | 10 |    |    |    |    | 0  |

| a | b | c | d | a | b | c | d |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |

$$\bar{b} + \bar{c}$$

$$\bar{b} + \bar{d}$$

$$\bar{b} + c$$

$$\bar{b} + d$$

$$b + c$$

$$b + d$$

$$b + \bar{c}$$

$$b + \bar{d}$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

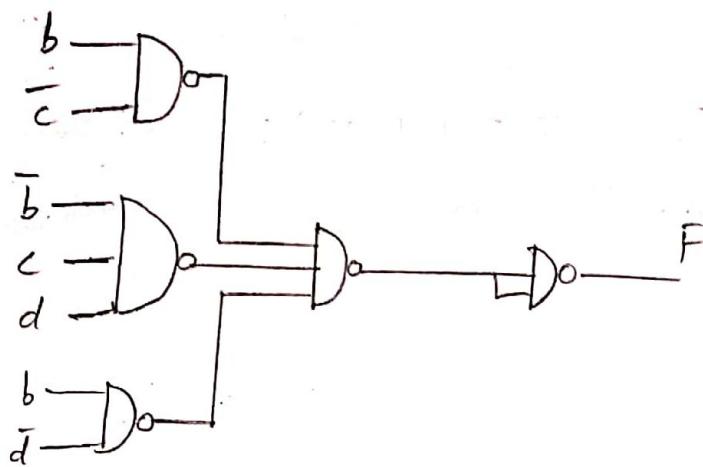
$$b + \bar{c} + \bar{d}$$

$$b + c + d$$

$$b + \bar{c} + d$$

$$b + c + \bar{d}$$

$$F = (b+c) (b+\bar{c}+d) (b+d)$$



2) Find the minimal SOP & POS for function by map method

$$F(a, b, c, d) = \Sigma m(2, 4, 9, 10, 15) + \Sigma d(0, 13, 14)$$

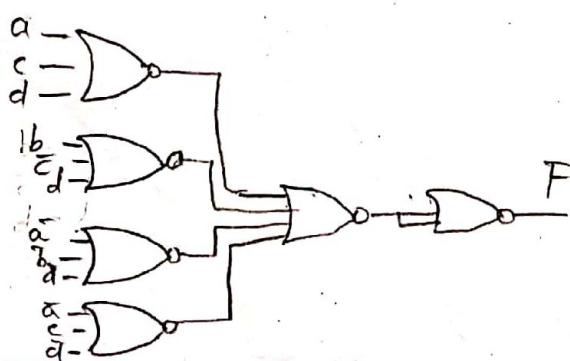
realize the logic circuit using NOR gates only.

SOP :-

|    | 00 | 01 | 11 | 10 |
|----|----|----|----|----|
| 00 | 0  | 1  | 3  | 1  |
| 01 | 1  | 4  | 5  | 6  |
| 11 | 12 | 13 | 15 | X  |
| 10 | 8  | 1  | 11 | 1  |

$$\begin{array}{ll}
 a \cdot b \cdot \bar{c} \cdot \bar{d} & ab \cdot \bar{c} \cdot \bar{d} \\
 00 \cdot 00 & 00 \cdot 0 \\
 01 \cdot 00 & 10 \cdot 10 \\
 \bar{a} \cdot \bar{c} \cdot \bar{d} & \bar{b} \cdot \bar{c} \cdot \bar{d} \\
 a \cdot b \cdot \bar{c} \cdot d & ab \cdot \bar{c} \cdot d \\
 11 \cdot 01 & 11 \cdot 01 \\
 11 \cdot 11 & 10 \cdot 01 \\
 ab \cdot d & a \cdot \bar{c} \cdot d
 \end{array}$$

$$F = \bar{a} \cdot \bar{c} \cdot \bar{d} + \bar{b} \cdot \bar{c} \cdot \bar{d} + ab \cdot d + a \cdot \bar{c} \cdot d$$



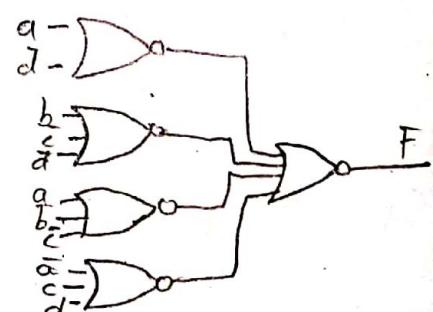
POS:-

|    | 00 | 01 | 11 | 10 |
|----|----|----|----|----|
| 00 | X  | 0  | 0  |    |
| 01 | 0  | 0  | 0  | 0  |
| 11 | 0  | X  |    | X  |
| 10 | 0  |    | 0  |    |

$$\begin{array}{lll}
 a \cdot b \cdot c \cdot d & a \cdot b \cdot \bar{c} \cdot \bar{d} & a \cdot \bar{b} \cdot \bar{c} \cdot \bar{d} \\
 00 \cdot 00 \cdot 0 & 00 \cdot 01 \cdot 0 & 00 \cdot 11 \cdot 0 \\
 00 \cdot 00 \cdot 1 & 00 \cdot 01 \cdot 1 & 00 \cdot 11 \cdot 1 \\
 01 \cdot 00 \cdot 1 & 01 \cdot 01 \cdot 0 & 01 \cdot 11 \cdot 0 \\
 01 \cdot 00 \cdot 0 & 01 \cdot 01 \cdot 1 & 01 \cdot 11 \cdot 1 \\
 a \cdot \bar{d} & a \cdot b \cdot \bar{c} \cdot \bar{d} & a \cdot b \cdot \bar{c} \cdot \bar{d} \\
 a \cdot \bar{d} & 11 \cdot 00 & 11 \cdot 00 \\
 a \cdot \bar{d} & 10 \cdot 00 & 10 \cdot 00 \\
 a \cdot \bar{d} & \bar{a} \cdot \bar{c} \cdot d & \bar{a} \cdot \bar{c} \cdot d
 \end{array}$$

$$F = (\bar{a} \cdot \bar{c} \cdot \bar{d}) \cdot (\bar{b} \cdot \bar{c} \cdot \bar{d}) \cdot (a \cdot \bar{b} \cdot \bar{c})$$

$$(\bar{a} \cdot \bar{c} \cdot d)$$



flip flops :-

difference b/w Latch and flip flops :-

1) A Latch is a electronic sequential logic circuit used to store information in an ~~asynchronous~~ arrangement.

2) A flip flop is a electronic sequential logic circuit used to store information in an synchronous arrangement.

3) one latch can stored 1 bit information.

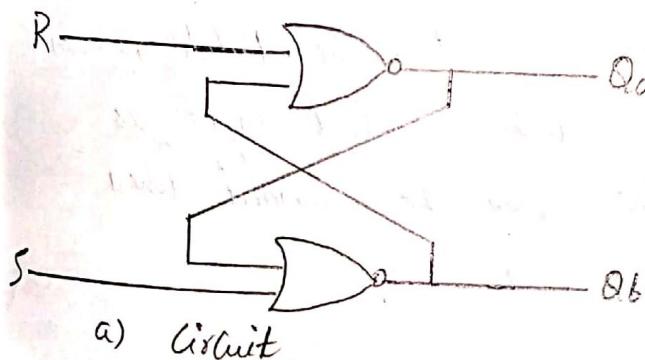
4) It has 2 stable states.

5) Latch has no clock input

6) flip flop has clock input

Basic Latch :-

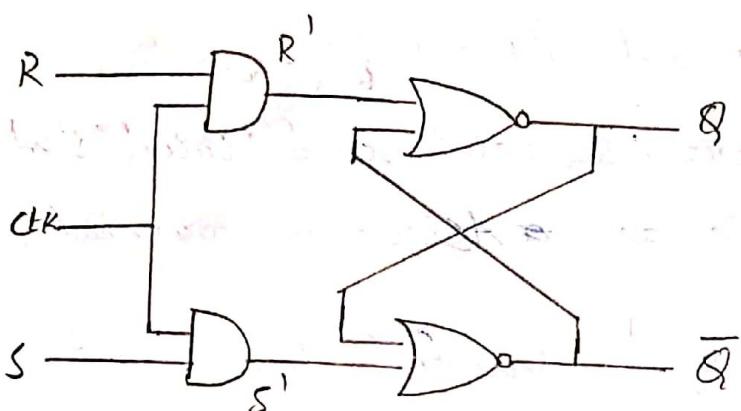
Fig :- A basic Latch (SR latch)



| S | R | Q <sub>a</sub> | Q <sub>b</sub>  |
|---|---|----------------|-----------------|
| 0 | 0 | 0/1            | 1/0 (no change) |
| 0 | 1 | 0              | 1 (change)      |
| 1 | 0 | 1              | 0               |
| 1 | 1 | 0              | 0               |

Basic Latch is a feedback connection of 2 NOR gates or 2 NAND gates which can store 1 bit information. it can set to one using S input and reset to zero using R input.

## Gated SR latch :-

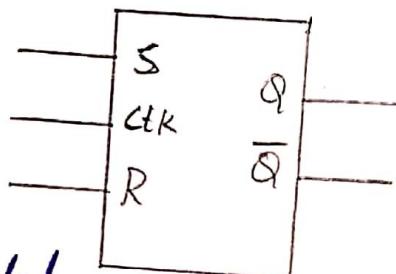


a) Circuit

| CTK | S | R | Q(t+1)    |
|-----|---|---|-----------|
| 0   | x | x | Q(t)      |
| 1   | 0 | 0 | Q(t)      |
| 1   | 0 | 1 | no change |
| 1   | 1 | 0 | 0         |
| 1   | 1 | 1 | 1         |
|     |   |   | x         |

b) Truth Table

c) Graphical symbol



Gated Latch :-

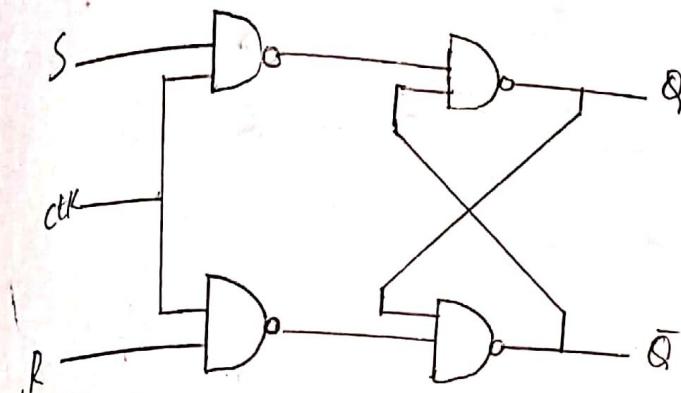
→ Gated Latch is a basic latch that includes inputs & controlled input signal. The Latch remain in its existing state when control input is equal to zero. Its state may be changed when the control signal is equal to 1.

→ There are 2 types of gated latches

1) Gated SR Latch :- uses the S & R input to set the latch to one or reset it to 0 respectively.

2) Gated D Latch :- uses the D input to force the latch into a state that has the same logic value as D input.

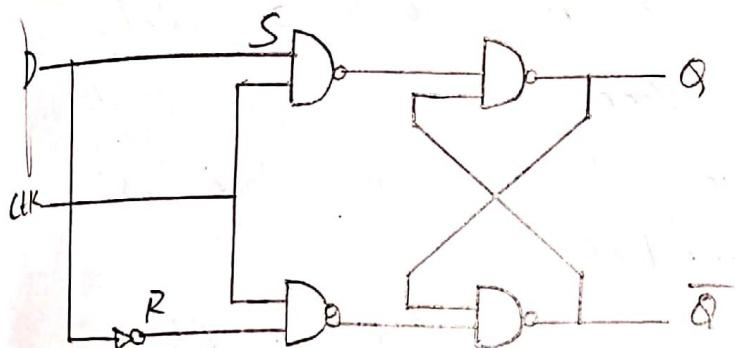
Gated SR Latch with NAND gates :-



a) Circuit

b) Truth Table  
Same of gated SR Latch

Gated D Latch :-

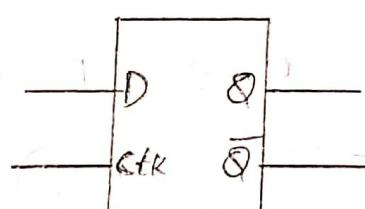


a) Circuit

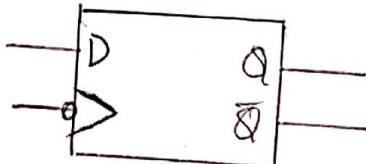
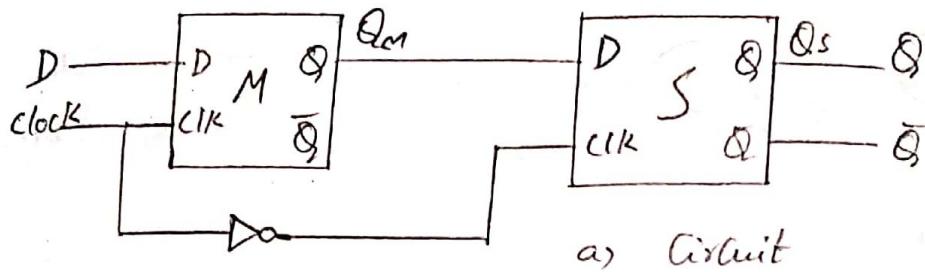
b) Truth Table

| CLK | D | $Q(t+1)$ |
|-----|---|----------|
| 0   | X | $Q(t)$   |
| 1   | 0 | 0        |
| 1   | 1 | 1        |

c) Graphical Symbol



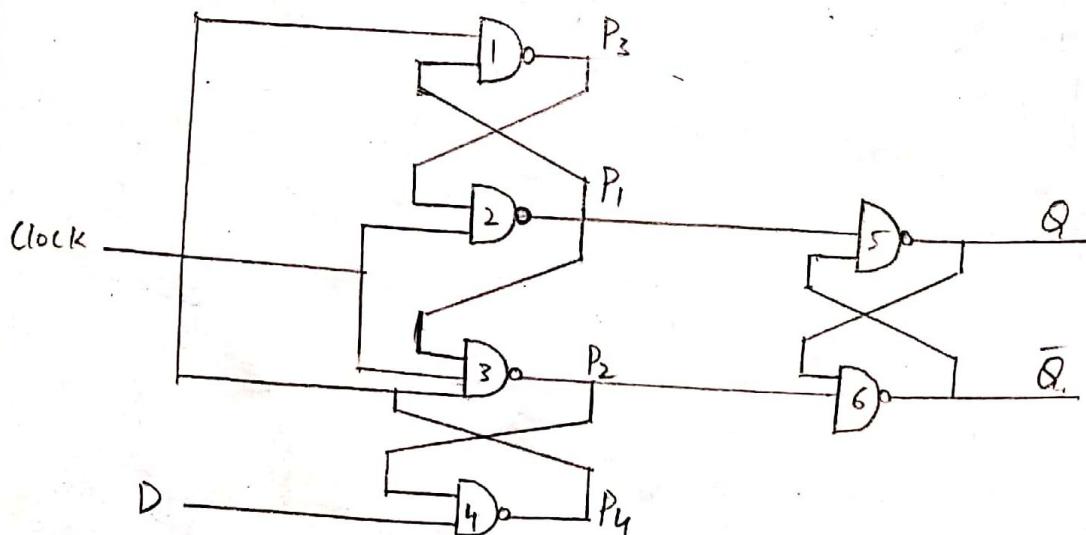
# Master Slave & Edge triggered D ff



b) Graphical symbol

- Master slave ff is build with 2 gated latches. The Master stage is active during half of the clock cycle and the slave stage is active during the other half.
- If the master stage is gated D latch then it behaves as Edge triggered flip flop
- If the master stage is gated SR latch then the flip flop is level sensitive.

## Edge triggered D ff



If has 6 NAND gates  
when,  $\text{clock} = 0$

$$P_1 = P_2 = 1$$

$$P_3 = D$$

$$P_4 = \bar{D}$$

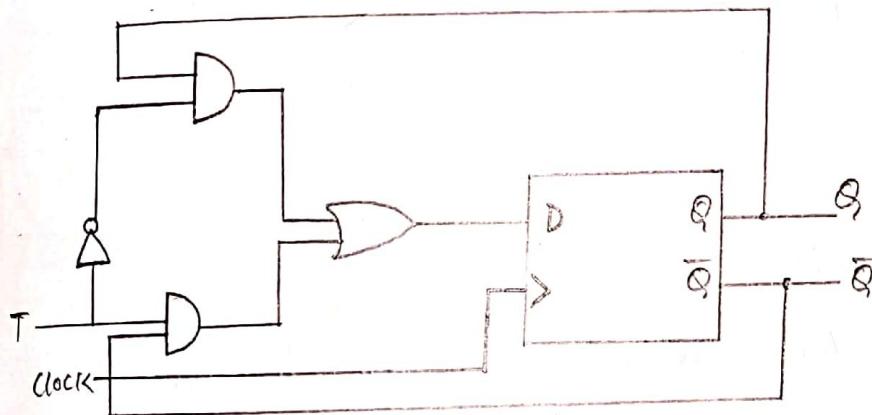
when  $\text{clock} = 1$

$$P_1 = \bar{D}$$

$$P_2 = D$$

which sets  $Q = D$   
 $\bar{Q} = \bar{D}$

27/9/19  
T-ff :-

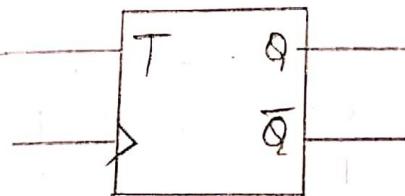


a) Circuit

b) Truth Table

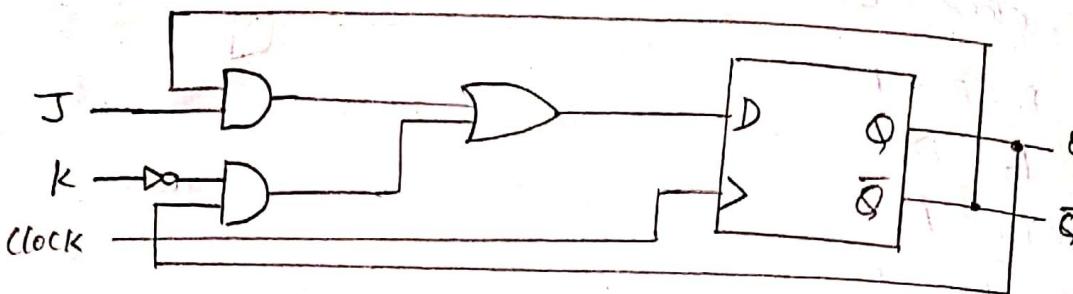
| T | $Q(t+1)$     |
|---|--------------|
| 0 | $Q(t)$       |
| 1 | $\bar{Q}(t)$ |

c) Graphical symbol



- The name T-ff derives from the behaviour of the circuit which Toggles its state when  $T=1$
- The Toggle feature makes the T-ff a useful element for building counter circuit.

JK ff :-



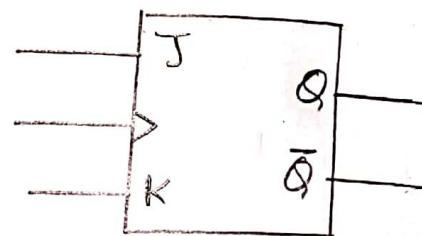
a) Circuit

$$\therefore D = J\bar{Q} + \bar{K}Q$$

b) Truth Table

| J | K | $Q(t+1)$     |
|---|---|--------------|
| 0 | 0 | $Q(t)$       |
| 0 | 1 | 0            |
| 1 | 0 | 1            |
| 1 | 1 | $\bar{Q}(t)$ |

c) Graphical Symbol



Excitation Tables :-

It lists the required ip's for a given change of state.

1) JK - ff

| $Q(t)$ | $Q(t+1)$ | J | K |
|--------|----------|---|---|
| 0      | 0        | 0 | x |
| 0      | 1        | 1 | x |
| 1      | 0        | x | 1 |
| 1      | 1        | x | 0 |

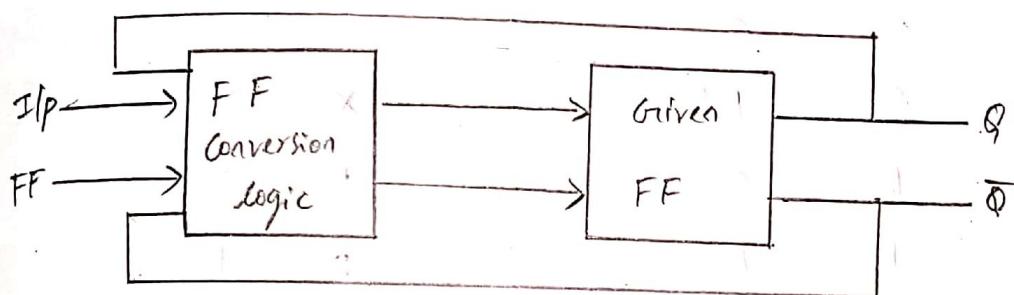
2, SR - ff

| $Q(t)$ | $Q(t+1)$ | S | R |
|--------|----------|---|---|
| 0      | 0        | 0 | x |
| 0      | 1        | 1 | 0 |
| 1      | 0        | 0 | 1 |
| 1      | 1        | x | 0 |

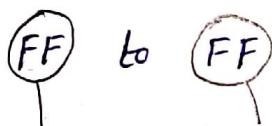
| 3) D-ff |          | D |
|---------|----------|---|
| $Q(t)$  | $Q(t+1)$ |   |
| 0       | 0        | 0 |
| 0       | 1        | 1 |
| 1       | 0        | 0 |
| 1       | 1        | 1 |

| 4) T-ff |          | T |
|---------|----------|---|
| $Q(t)$  | $Q(t+1)$ |   |
| 0       | 0        | 0 |
| 0       | 1        | 1 |
| 1       | 0        | 1 |
| 1       | 1        | 0 |

$\times^2$  FF Conversion :-



1) SR to JK ff



Exc table      char table (or) Truth Table

Excitation Table of SR

| $Q(n)$ $Q(n+1)$ |   | S | R |
|-----------------|---|---|---|
| 0               | 0 | 0 | x |
| 0               | 1 | 1 | 0 |
| 1               | 0 | 0 | 1 |
| 1               | 1 | x | 0 |

char table of JK

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

Combine 2 tables

| I/p's | present state | Next state | FF i/p's |   |   |
|-------|---------------|------------|----------|---|---|
| J     | K             | $Q(n)$     | $Q(n+1)$ | S | R |
| 0     | 0             | 0          | 0        | 0 | X |
| 0     | 0             | 1          | 1        | X | 0 |
| 0     | 1             | 0          | 0        | 0 | X |
| 0     | 1             | 1          | 0        | 0 | 1 |
| 1     | 0             | 0          | 1        | 1 | 0 |
| 1     | 0             | 1          | 1        | X | 0 |
| 1     | 1             | 0          | 1        | 1 | 0 |
| 1     | 1             | 1          | 0        | 0 | 1 |

K-map :-

S :-

| J |   | K $Q(n)$ |   | 00 |   | 01 |   | 11 |   | 10 |   |
|---|---|----------|---|----|---|----|---|----|---|----|---|
|   |   | 0        | 1 | 0  | X | 1  | 1 | 0  | X | 1  | 1 |
| 0 | 0 | X        |   |    |   |    |   |    |   |    |   |
| 1 | 1 | 1        | X |    |   |    |   |    |   |    |   |

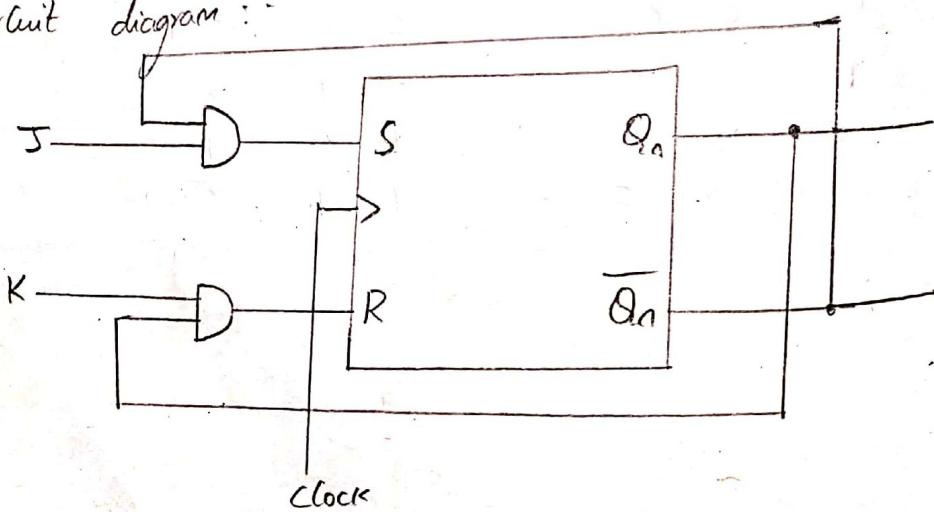
R :-

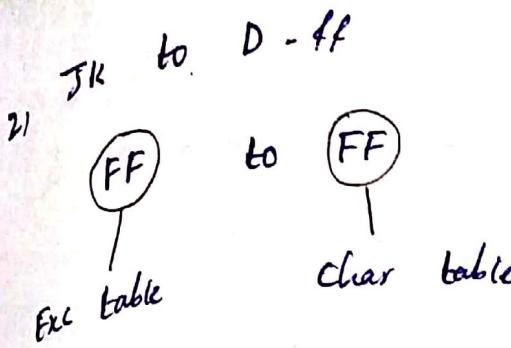
| J |   | K $Q(n)$ |   | 00 |   | 01 |   | 11 |   | 10 |   |
|---|---|----------|---|----|---|----|---|----|---|----|---|
|   |   | 0        | 1 | 0  | X | 1  | 1 | 0  | X | 1  | 1 |
| 0 | 0 | X        |   |    |   |    |   |    |   |    |   |
| 1 | 1 | 1        | X |    |   |    |   |    |   |    |   |

$$S = J \bar{Q}_n$$

$$R = K Q_n$$

Circuit diagram :-





Exc table of JK

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

char table of D

| clk | D | $Q(n+1)$ |
|-----|---|----------|
| 0   | X | $Q(n)$   |
| 1   | 0 | 0        |
| 1   | 1 | 1        |

combine 2 tables :-

| I/p's | present state | Next state | FF i/p's |
|-------|---------------|------------|----------|
| D     | $Q(n)$        | $Q(n+1)$   | J    K   |
| 0     | 0             | 0          | 0    X   |
| 0     | 1             | 0          | X    X   |
| 1     | 0             | 1          | X    X   |
| 1     | 1             | 1          | X    0   |

K-map :-  $J = \overline{D}$

$$D \begin{array}{|c|c|} \hline 0 & 1 \\ \hline 0 & 0 \quad X \\ \hline 1 & 1 \quad X \\ \hline \end{array}$$

$J = \overline{D}$

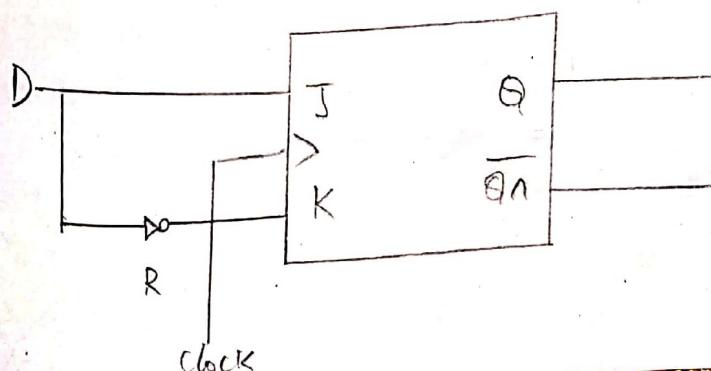
K-map :-  $K = \overline{D}$

$$D \begin{array}{|c|c|} \hline 0 & 1 \\ \hline 0 & 0 \quad 1 \\ \hline 1 & X \quad 0 \\ \hline \end{array}$$

$K = \overline{D}$

$$D \begin{array}{|c|c|} \hline 0 & 1 \\ \hline 0 & 0 \\ \hline 1 & 1 \\ \hline \end{array}$$

circuit diagram :-



30/9/19

Characteristic Table :-

Characteristic table define next state as a function  
of i/p & present state

1) JK ff

| J | K | $Q(t+1)$          | operation  |
|---|---|-------------------|------------|
| 0 | 0 | $Q(t)$            | No change  |
| 0 | 1 | 0                 | Reset      |
| 1 | 0 | 1                 | Set        |
| 1 | 1 | $\overline{Q(t)}$ | Complement |

2) SR ff

| S | R | $Q(t+1)$ | operation |
|---|---|----------|-----------|
| 0 | 0 | $Q(t)$   | No change |
| 0 | 1 | 0        | Reset     |
| 1 | 0 | 1        | Set       |
| 1 | 1 | ?        | undefined |

3) D ff

| D | $Q(t+1)$ | operation |
|---|----------|-----------|
| 0 | 0        | Reset     |
| 1 | 1        | Set       |

4) T ff

| T | $Q(t+1)$          | operation  |
|---|-------------------|------------|
| 0 | $Q(t)$            | No change  |
| 1 | $\overline{Q(t)}$ | Complement |

3)  $\overline{S} \cdot R$  to  $T$   
 Excitation character

Exc table of SR

| $Q(a)$ | $Q(a+1)$ | S | R |
|--------|----------|---|---|
| 0      | 0        | 0 | x |
| 0      | 1        | 1 | 0 |
| 1      | 0        | 0 | 1 |
| 1      | 1        | x | 0 |

char table of T

| T | $Q(t+1)$          |
|---|-------------------|
| 0 | $Q(t)$            |
| 1 | $\overline{Q(t)}$ |

Combine 2 tables :-

| I/p's | present<br>$Q(a)$ | Next<br>$Q(a+1)$ | FF<br>I/p's |
|-------|-------------------|------------------|-------------|
| T     |                   |                  | S R         |
| 0     | 0                 | 0                | 0 x         |
| 0     | 1                 | 1                | x 0         |
| 1     | 0                 | 1                | 1 0         |
| 1     | 1                 | 0                | 0 1         |

K-Map S =

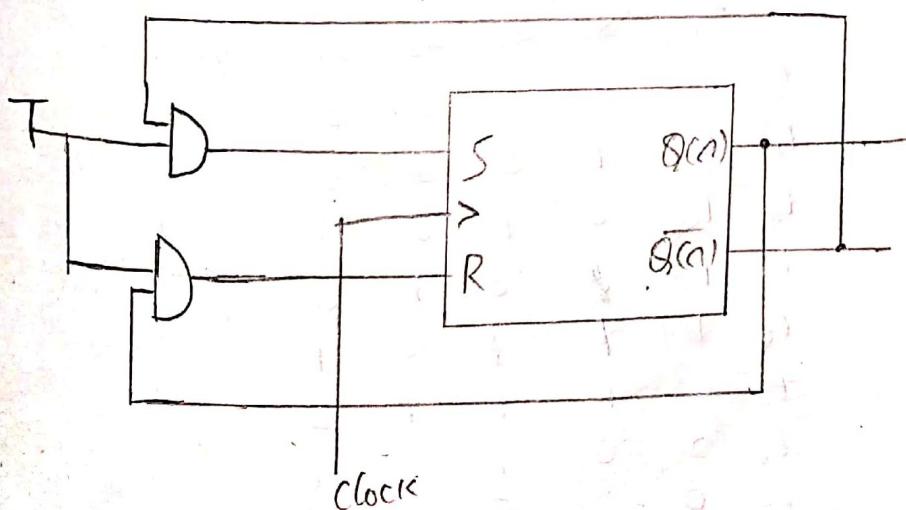
|   |        |   |
|---|--------|---|
| T | $Q(a)$ | S |
| 0 | 0      | 1 |
| 1 | 0      | 0 |

$$S = T \overline{Q(a)}$$

|   |        |   |
|---|--------|---|
| T | $Q(a)$ | R |
| 0 | 0      | 1 |
| 1 | 0      | 0 |

$$R = T Q(a)$$

Circuit diagram :-



4/10/19

## Registers :-

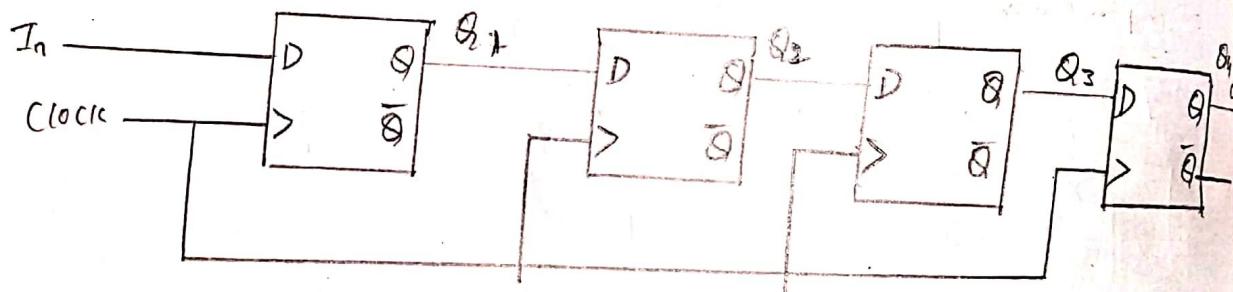
When a set of  $n$ -ff is used to store  $n$ -bits of information than these flip flops are called as Registers.

- A common clock is used for each flip flop in a register.

## Shift Register:-

A register that provides the ability to shift its contents is called a shift register.

Ex:- 4 bit shift register



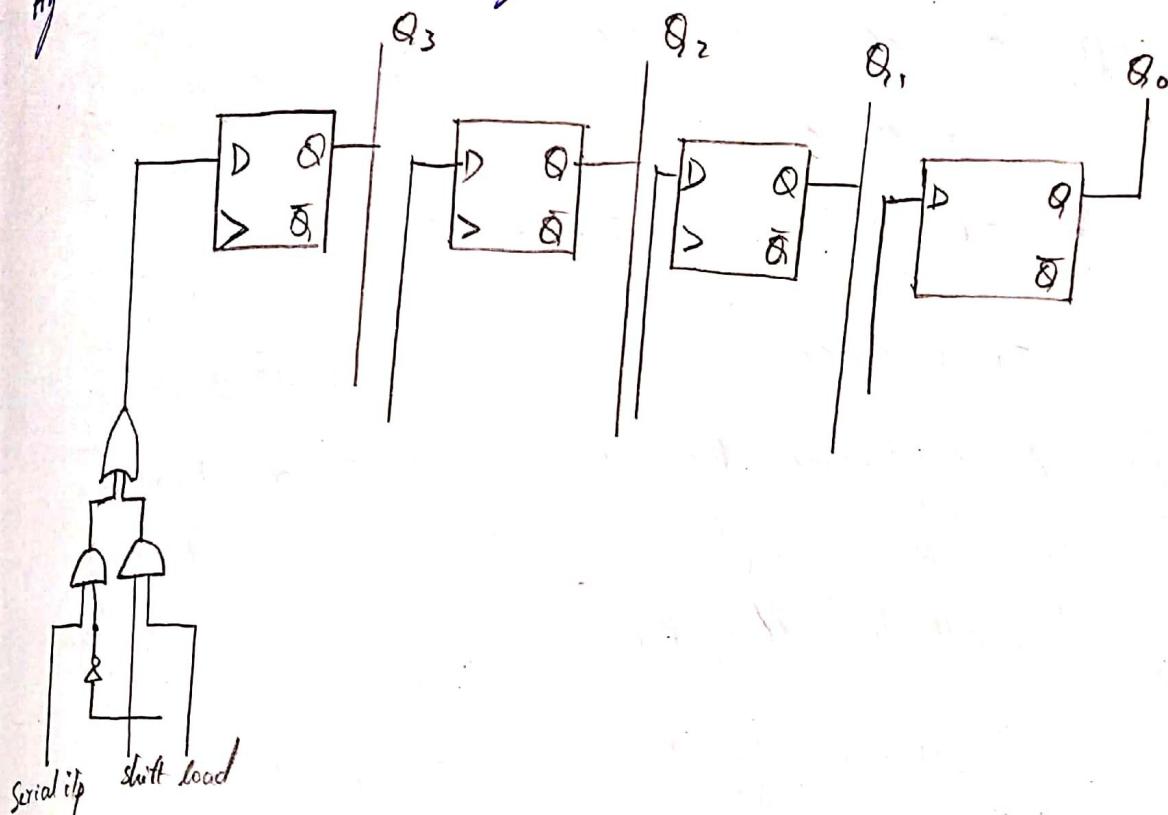
a) Circuit

b) Sample sequence

8 consecutive clock cycles 1, 0, 1, 1, 1, 0, 0, and 0.

|       | In | $Q_1$ | $Q_2$ | $Q_3$ | $Q_4$ | Out |
|-------|----|-------|-------|-------|-------|-----|
| to    | 1  | 0     | 0     | 0     | 0     | 0   |
| $t_1$ | 0  | 1     | 0     | 0     | 0     | 0   |
| $t_2$ | 1  | 0     | 1     | 0     | 0     | 0   |
| $t_3$ | 1  | 1     | 0     | 1     | 0     | 0   |
| $t_4$ | 1  | 1     | 1     | 0     | 1     | 1   |
| $t_5$ | 0  | 1     | 1     | 1     | 0     | 1   |
| $t_6$ | 0  | 0     | 1     | 1     | 1     | 1   |
| $t_7$ | 0  | 0     | 0     | 1     | 1     | 1   |

parallel - access shift Registers :-  
 all bits at once - II (parallel)  
 1 bit at a time - serial  
 Fig : 4-bit shift register with parallel access

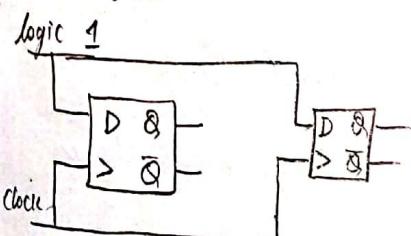


Counters :- (next page diff) (rough)

Synchronize

clock pulse

signal



As synchronize

clock pulse p<sup>t</sup> t

logic 1



Counters :-

diff :-

A Counter is a register that goes through a pre determined sequence of states upon the application of clock pulse

-1 2 types of counter

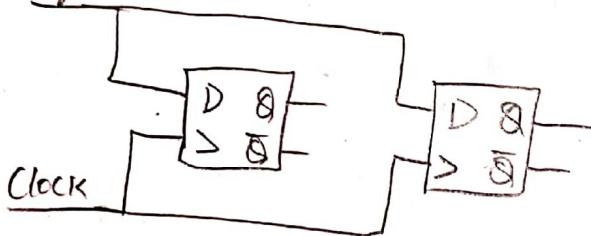
1) Synchronous

2) Asynchronous

Synchronous

1) In Synchronous the clock pulse are applied simultaneously to all the flip flops.

2) Circuit diagram of 2 bit synchronous counter logic 1



3, Speed low

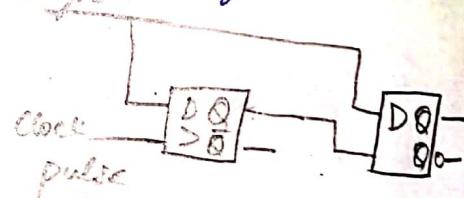
4, Simple hardware

5, higher & lower frequency

Asynchronous

1, In Asynchronous counters the clock pulse is applied to the 1<sup>st</sup> flip flop.

2, Circuit diagram of 2 bit Asynchronous counter logic 1

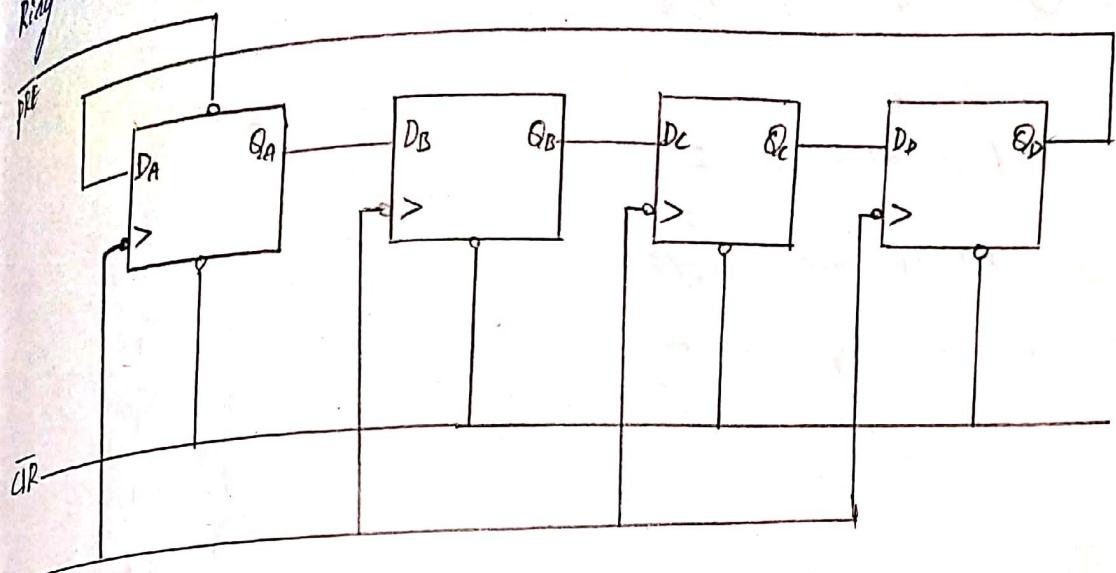


3, Speed high

4, Complex hardware

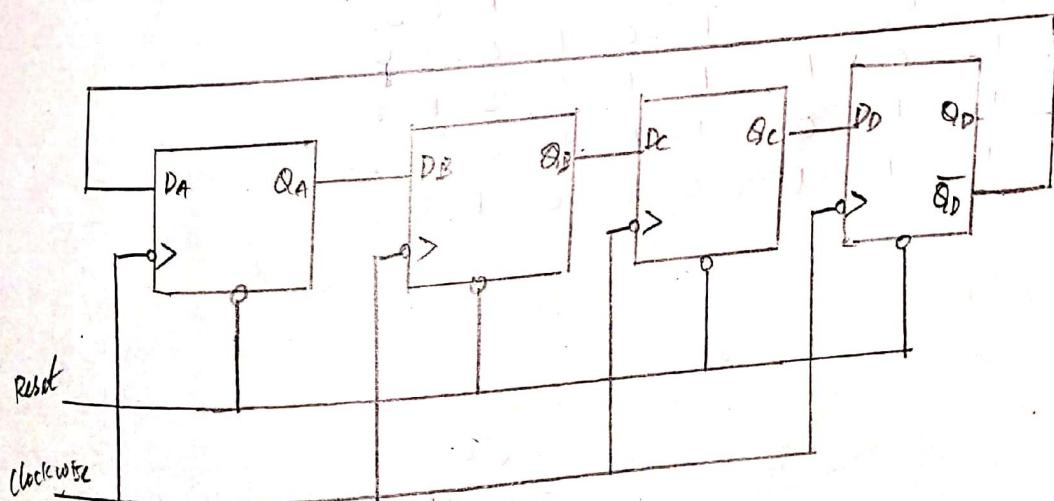
5, lower frequency

Counters :-



|   | clock pulse | Q <sub>A</sub> | Q <sub>B</sub> | Q <sub>C</sub> | Q <sub>D</sub> |
|---|-------------|----------------|----------------|----------------|----------------|
| 0 |             | 1              | 0              | 0              | 0              |
| 1 |             | 0              | 1              | 0              | 0              |
| 2 |             | 0              | 0              | 1              | 0              |
| 3 |             | 0              | 0              | 0              | 1              |
| 4 |             | 1              | 0              | 0              | 0              |

Johnson or Twisting Ring or Switch Tail Counter:-



|               | Clock pulse | $Q_A$ | $Q_B$ | $Q_C$ | $Q_D$ |
|---------------|-------------|-------|-------|-------|-------|
| shift '1'     | 0           | 0     | 0     | 0     | 0     |
| to right      | 1           | 1     | 0     | 0     | 0     |
| the shift 0's | 2           | 1     | 1     | 0     | 0     |
|               | 3           | 1     | 1     | 1     | 0     |
|               | 4           | 1     | 1     | 1     | 1     |
|               | 5           | 0     | 1     | 1     | 1     |
|               | 6           | 0     | 0     | 1     | 1     |
|               | 7           | 0     | 0     | 0     | 1     |

~~21110119~~  $\times^S^3$   
3-bit synchronous up counter using Tff

State table:

| present state |       |       | N.S    |        |        | Ex. table<br>of Tff |             |             | state diagram |     |     |
|---------------|-------|-------|--------|--------|--------|---------------------|-------------|-------------|---------------|-----|-----|
| $Q_C$         | $Q_B$ | $Q_A$ | $Q'_C$ | $Q'_B$ | $Q'_A$ | $T_C$               | $\bar{T}_B$ | $\bar{T}_A$ | 111           | 110 | 000 |
| 0             | 0     | 0     | 0      | 0      | 1      | 0                   | 0           | 1           | 111           | 110 | 000 |
| 0             | 0     | 1     | 0      | 1      | 0      | 0                   | 1           | 1           | 101           | 100 | 001 |
| 0             | 1     | 0     | 0      | 1      | 0      | 0                   | 1           | 1           | 101           | 100 | 001 |
| 0             | 1     | 1     | 0      | 1      | 1      | 0                   | 0           | 1           | 100           | 011 | 010 |
| 1             | 0     | 0     | 1      | 0      | 0      | 1                   | 1           | 0           | 110           | 111 | 000 |
| 1             | 0     | 1     | 1      | 0      | 1      | 0                   | 0           | 0           | 011           | 010 | 001 |
| 1             | 1     | 0     | 1      | 1      | 0      | 0                   | 1           | 1           | 011           | 010 | 001 |
| 1             | 1     | 1     | 0      | 0      | 0      | 1                   | 1           | 1           | 000           | 001 | 000 |

Truth Table:

$$\bar{T}_A = 1$$

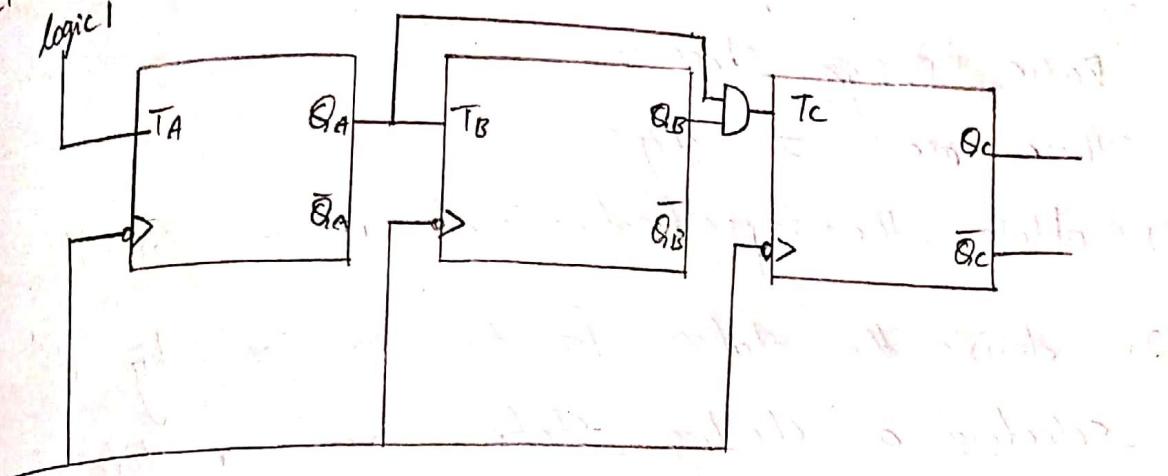
$$\bar{T}_B$$

| $Q_C$ | 00 | 01 | 11 | 10 |
|-------|----|----|----|----|
| 0     | 1  | 1  | 1  |    |
| 1     | 1  | 1  | 1  |    |

| $Q_C$ | 00 | 01 | 11 | 10 |
|-------|----|----|----|----|
| 0     |    |    | 1  |    |
| 1     |    |    | 1  |    |

circuit diagram :-

logic 1



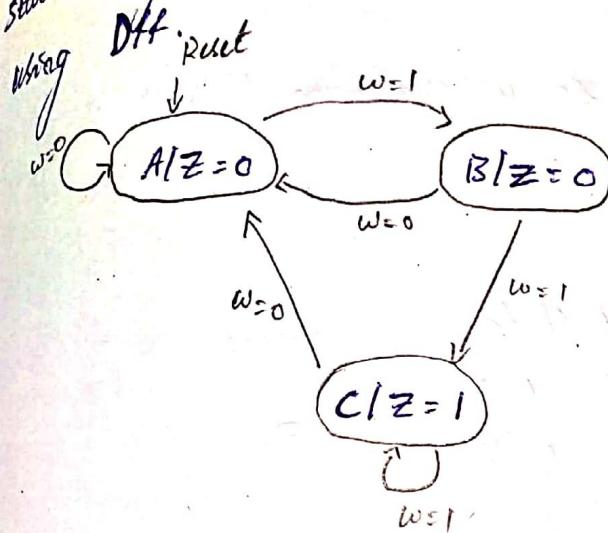
## Synchronous Sequential Circuits

Basic Design Steps :-

There are 7 steps :-

- 1) obtain the specification of desired circuit.
- 2, derive the states for the machine by <sup>1st</sup> selecting a starting state. Create a state diagram.
- 3, Create a state table from the state diagram.
- 4, performed state minimization that is a procedure that minimizes the no. of states (if many states are given).
- 5, Decide on the no. of state variables needed to represent to all states and perform the State assignment. (There can be many state assignments).
- 6, choose the type of flip flops to be used in the circuit.
- 7, Implement the circuit as indicated by logic expression.

Q) A state diagram Derive a circuit that realizes the FSM



Q) step 1:- State table

| present state | Next state |       | output<br>$z$ |
|---------------|------------|-------|---------------|
|               | $w=0$      | $w=1$ |               |
| A             | A          | B     | 0             |
| B             | A          | C     | 0             |
| C             | A          | C     | 1             |

Each state variable may be implemented in the form of flip flop (DFF). Since, there are 3 states to be realize it is sufficient to use 2 state variables  $y_1, y_2$ .

$$2^2 = 4$$

$$3(A, B, C)$$

$$3 < 4$$

1 is unused (don't care)

State assignment :-

$$y_2 y_1 = \begin{array}{l} 00 (A) \\ 01 (B) \\ 10 (C) \end{array}$$

As D is not there so 4<sup>th</sup> valuation  $y_2 y_1 = 11$  is not needed

Fig. state assignment :-

|   | present state |       | Next State |       |       |       | output<br>$z$ |
|---|---------------|-------|------------|-------|-------|-------|---------------|
|   | $y_2$         | $y_1$ | $y_2$      | $y_1$ | $w=0$ | $w=1$ |               |
| A | 0             | 0     | 0          | 0     | 0     | 1     | 0             |
| B | 0             | 1     | 0          | 0     | 1     | 0     | 0             |
| C | 1             | 0     | 0          | 0     | 1     | 0     | 1             |
|   | 1             | 1     | d          | d     | dd    | dd    | d             |

K-map :-

|   | $y_2 y_1$ | 00 | 01 | 11 | 0 |
|---|-----------|----|----|----|---|
| w |           |    |    |    |   |
| 0 |           |    |    | d  |   |
| 1 | 1         | d  |    |    |   |

$$y_1 = w \bar{y}_1 \bar{y}_2$$

|   | $y_2 y_1$ | 00  | 01 | 11 | 10 |
|---|-----------|-----|----|----|----|
| w |           |     |    |    |    |
| 0 |           |     |    | d  |    |
| 1 | 1         | (d) | d  | 1  |    |

$$\begin{aligned} y_2 &= w y_1 + w y_2 \\ y_2 &= w(y_1 + y_2) \end{aligned}$$

|       | $y_2$ | 0 | 1 |
|-------|-------|---|---|
| $y_1$ |       |   |   |
| 0     |       |   |   |
| 1     | 1     | d |   |

$$z = y_2$$

$y_1$  is 1 at 1<sup>st</sup> line

where  $w=1, y_2 y_1 = 00$

23/10/19

a) explain state assignment problem with ex.

$$A = \begin{matrix} y_2 & y_1 \\ 0 & 0 \end{matrix}$$

$$B = \begin{matrix} 0 & 1 \end{matrix}$$

$$C = \begin{matrix} 1 & 1 \end{matrix}$$

∴ is not there so 4<sup>th</sup> valuation  $y_2 y_1 = 10$  is not needed

Fig. Improved state Assignment:

| present state | Next state |       |       |       | output |   |
|---------------|------------|-------|-------|-------|--------|---|
|               | w=0        |       | w=1   |       |        |   |
|               | $y_2$      | $y_1$ | $y_2$ | $y_1$ |        |   |
| A             | 0          | 0     | 0     | 0     | 1      | 0 |
| B             | 0          | 1     | 0     | 0     | 1      | 0 |
| C             | 1          | 1     | 0     | 0     | 1      | 1 |
|               | 1          | 0     | d     | d     | d      | d |

K-map:-

$$y_1 =$$

| w | $y_2$ | $y_1$ | 00 | 01 | 11 | 10 |
|---|-------|-------|----|----|----|----|
| 0 |       |       |    |    |    | d  |
| 1 |       |       | 1  | 1  | 1  | d  |

$$y_1 = w y_2 y_1$$

|   |   |   |
|---|---|---|
| 1 | 0 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |
| 1 | 1 | 0 |

$$y_1 = w$$

| w | $y_2$ | $y_1$ | 00 | 01 | 11 | 10 |
|---|-------|-------|----|----|----|----|
| 0 |       |       |    |    |    | d  |
| 1 |       |       | 1  | 1  | 1  | d  |

| $y_2$ | $y_1$ | 0 | 1 |
|-------|-------|---|---|
| 0     | 1     | 1 | 1 |
| 1     | d     | 1 | 1 |

$$y_2 = w y_2 y_1$$

$$\begin{matrix} 1 \\ 1 \\ 1 \\ 1 \end{matrix}$$

$$y_2 = w y_1$$

$$\begin{matrix} 1 \\ 1 \\ 1 \\ 0 \end{matrix}$$

$$Z = \begin{matrix} y_2 & y_1 \\ 1 & 0 \\ 1 & 1 \end{matrix}$$

$$Z = y_2$$

## State Minimization :-

The purpose of state minimization is to reduce the no. of states in a sequential circuit. So that the circuit requires fewer Ts.

partitioning minimization procedure :-

- Q) Minimize the given state table using the partitioning procedure

| present state | next state |     | output<br>z |
|---------------|------------|-----|-------------|
|               | w=0        | w=1 |             |
| A             | B          | C   | 1           |
| B             | D          | F   | 1           |
| C             | F          | E   | 0           |
| D             | B          | G   | 1           |
| E             | F          | C   | 0           |
| F             | E          | D   | 0           |
| G             | F          | G   | 0           |

A)  $P_1 = (ABCDEF)G$

olp is 0

olp is 1

$P_3 = ( ) ( ) ( ) ( )$

$P_2 = (\underline{ABD}) \underline{(C(EFG))}$

Now consider all 0 successors & 1. successors

$ABD$  olp when  $w=0 \Rightarrow \underline{BDB}$  - in same block

$w=1 \Rightarrow \underline{CFG}$  " " "

$C(EFG)$  olp when  $w=0 \Rightarrow \underline{FFEF}$  " " "

$w=1 \Rightarrow \underline{ECDG}$  - in different block

D present state 'F'

$$P_3 = \underline{(ABD)} C(EG) F$$

Now consider all 0 successors & 1. successors

ABD o/p when  $w=0 \Rightarrow BDB$  - in same block

$A(\textcircled{B})D$   $w=1 \Rightarrow CFG$  " diff block

F is from diff block, it present state is B  
can't be equivalent A & C

CEG<sub>1</sub> o/p when  $w=0 \Rightarrow FFF$  - in same block

$w=1 \Rightarrow ECG_1$  " " "

$$\therefore P_4 = (AD)(B) C(EG_1) CF$$

Now consider all 0 successors & 1. successors

(1) A<sub>1</sub> o/p when  $w=0 \Rightarrow BBB$  - in same block  
 $w=1 \Rightarrow CG_1$  " " "

CEG<sub>1</sub> o/p when  $w=0 \Rightarrow FFF$  - in same block  
 $w=1 \Rightarrow ECG_1$  " " "

$$\therefore P_5 = (AD)(B) C(EG) CF$$

Since  $P_5 = P_4$  and no new blocks are generated, it follows that states in each block are equivalent.

Take the 1<sup>st</sup> letter of each block

A represent both the states A and D

C " states C, E and G<sub>1</sub>

Fig. Minimized state table

| present state | next state |       | output<br>Z | $D=A$<br>$E=C$<br>$G_1=C$ |
|---------------|------------|-------|-------------|---------------------------|
|               | $w=0$      | $w=1$ |             |                           |
| A             | B          | C     | 1           |                           |
| B             | A          | F     | 1           |                           |
| C             | F          | C     | 0           |                           |
| F             | C          | A     | 0           |                           |

## UNIT-V

①

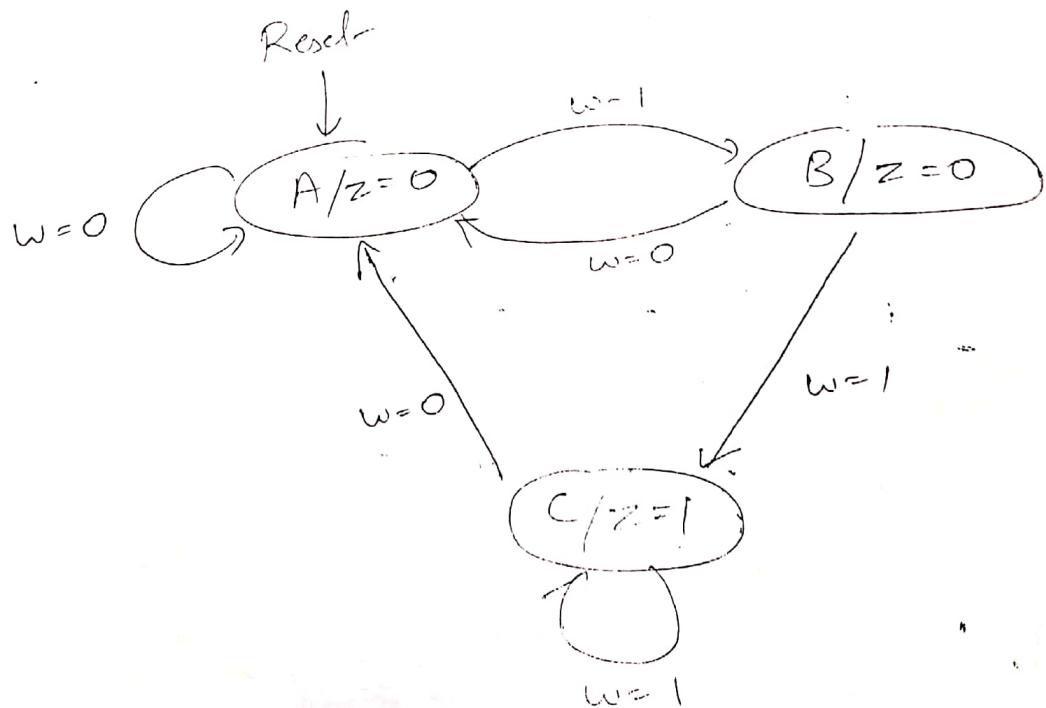
### Synchronous Sequential Circuits:

#### \* Basic Design Steps:

1. Obtain the specification of the desired circuit.
2. Derive the states for the machine by first selecting a starting state. Create a state diagram.
3. Create a state table from the state diagram.
4. Perform state minimization a procedure that minimizes the number of states. (if required i.e., if many states)
5. Decide on the number of state variables needed to represent all states & perform the state assignment. (whereas <sup>no. of</sup> <sub>state</sub> <sup>no. of</sup> <sub>variables</sub>)
6. Choose the type of flip-flops to be used in the circuit.
7. Implement the circuit as indicated by the logic expression.

Example:

Q. An FSM is defined by following state diag.  
Derive a circuit that realizes the FSM using D<sub>ff</sub>.



State table:

table indicates all transitions from each **present state** to the **next state** for different values of the input signal. & o/p  $g$  is specified w.r.t. the present state.

Fig 1. State table for the seq. ckt

| Present State | Next State |       | Output $g$ |
|---------------|------------|-------|------------|
|               | $w=0$      | $w=1$ |            |
| A             | A          | B     | 0          |
| B             | A          | C     | 0          |
| C             | A          | C     | 1          |

$g$  is 0 in  
'0', B  
1 in C

Each state may be implemented in the form of ff.

Since there are 3 states to be realized, it is sufficient to use 2 state variables  $y_1$  and  $y_2$ .

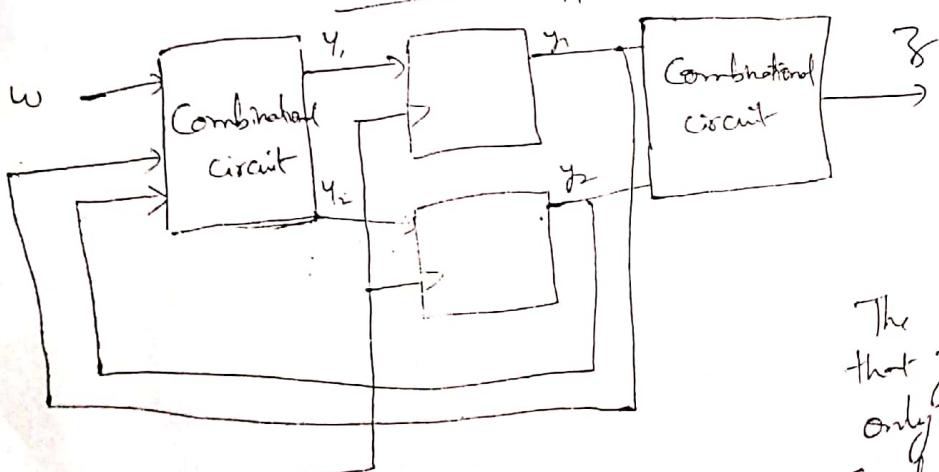
$2^2 = 4$  A,B,C  
 $3 \leq 4$   
1 is unused

Choice

State Assignment:

from the above fig, the output  $g$  is determined only by the present state of the circuit. Thus, the block diag. shows that  $g$  is a function of only  $y_1$  and  $y_2$  (Moore type)

Fig. A general sequential circuit with ipw, o/p  $g$  and 2 state ff's.



The block diag. shows that  $g$  is a func. of only  $y_1$  and  $y_2$ . So design is Moore type.

K-m  
4

(2)

The possible assignments is given in fig. where the states A, B and C are represented by  $y_2y_1 = 00, 01$  and 10 respect.

$$A = \begin{smallmatrix} y_2 \\ y_1 \end{smallmatrix} \begin{smallmatrix} 0 \\ 0 \end{smallmatrix}$$

$$B = \begin{smallmatrix} y_2 \\ y_1 \end{smallmatrix} \begin{smallmatrix} 0 \\ 1 \end{smallmatrix}$$

$$C = \begin{smallmatrix} y_2 \\ y_1 \end{smallmatrix} \begin{smallmatrix} 1 \\ 0 \end{smallmatrix}$$

As D is not there so 4<sup>th</sup> valuation  $y_2y_1 = 11$  is not needed

Fig. State assigned table for the sequential circuit in fig(2)

|   | Present State<br>$y_2 y_1$ | Next State |     | Output<br>$\gamma$ | From Fig.<br>A = 00 |
|---|----------------------------|------------|-----|--------------------|---------------------|
|   |                            | w=0        | w=1 |                    |                     |
| A | 0 0                        | 0 0        | 0 1 | 0                  |                     |
| B | 0 1                        | 0 0        | 1 0 | 0                  |                     |
| C | 1 0                        | 0 0        | 1 0 | 1                  |                     |
|   | 1 1                        | d d        | d d | d                  |                     |

Choice of ffs and derivation of next state & output expression

K-map

|   | $y_1$ | $y_2$ | $y_3$ | $y_4$ | $y_5$ | $y_6$ | $y_7$ | $y_8$ |
|---|-------|-------|-------|-------|-------|-------|-------|-------|
| w | 0     | 0     | 0     | 0     | 0     | 0     | 1     | 1     |
| 0 | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

$$Y_1 = w \bar{y}_1 \bar{y}_2$$

$y_1$  is '1' at 1<sup>st</sup> line  
where  $w=1, y_2y_1=00$   
 $y_1$  is 'd' at 4<sup>th</sup> line  
where  $w=0, y_2y_1=11$   
and  $w=1, y_2y_1=11$

$y_2$

|   | $y_1$ | $y_2$ | $y_3$ | $y_4$ | $y_5$ | $y_6$ | $y_7$ | $y_8$ |
|---|-------|-------|-------|-------|-------|-------|-------|-------|
| w | 0     | 0     | 0     | 0     | 0     | 0     | 1     | 1     |
| 0 | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

$$\begin{aligned} Y_2 &= w y_1 + w y_2 \\ &= w(y_1 + y_2) \end{aligned}$$

?

2 var. K-map ( $y_2$  &  $y_1$ )

|   | $y_1$ | 0 | 1 |
|---|-------|---|---|
| 0 | 0     | 1 | 1 |
| 1 | 1     | 1 | d |

$$\gamma = Y_2$$

?

|   | $y_1$ | $y_2$ | $y_3$ | $y_4$ | $y_5$ | $y_6$ | $y_7$ | $y_8$ |
|---|-------|-------|-------|-------|-------|-------|-------|-------|
| w | 0     | 0     | 0     | 0     | 0     | 0     | 1     | 1     |
| 0 | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

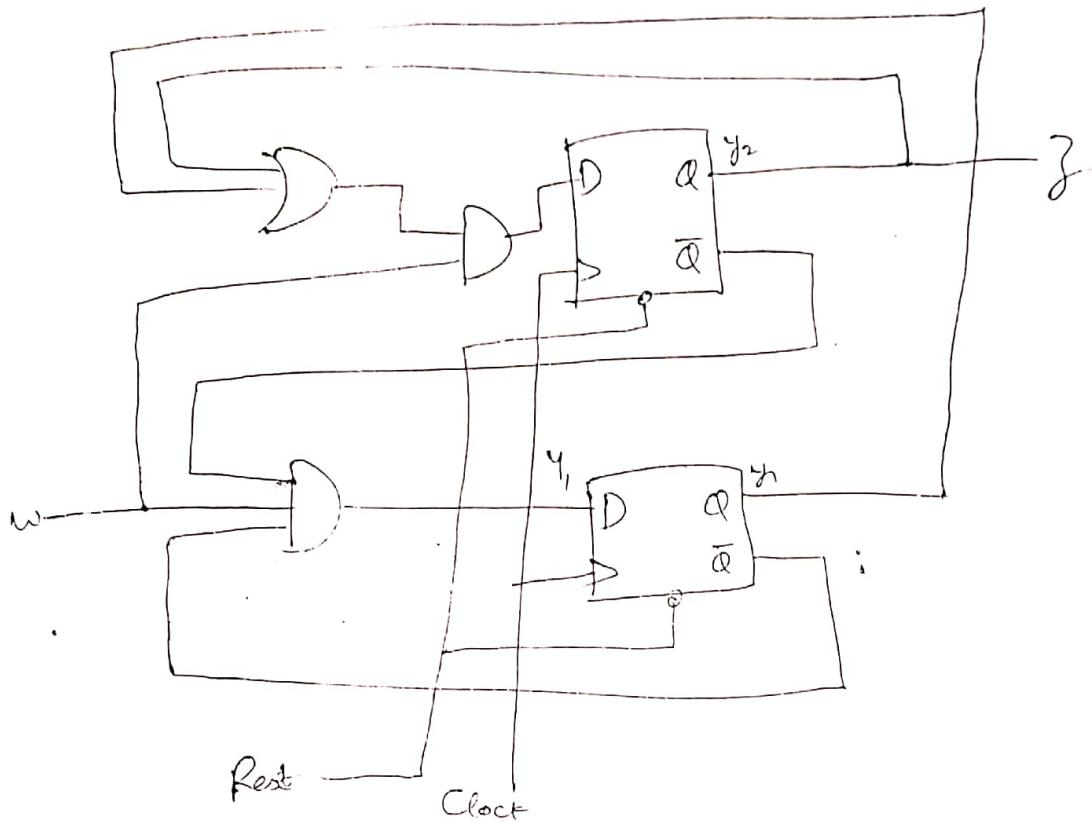
rows  
f  
type

g Diff

$$D_1 = Y_1 \text{ and } D_2 = Y_2$$

$$\begin{aligned}Y_1 &= w\bar{y}_1\bar{y}_2 \\Y_2 &= w(y_1 + y_2) \\z &= y_2\end{aligned}$$

Fig. Final implementation of the sequential circuit



Explain state assignment problem with an ex.

Refer previous question fill state table

Fig. gives one more possible alternative. In this case the states A, B and C are

$$\begin{array}{c|cc}A & y_2 & y_1 \\ \hline A & 0 & 0 \\ B & 0 & 1 \\ C & 1 & 1\end{array}$$

D is not there so 4<sup>th</sup> valuation  $y_2y_1=10$  is not needed

| Present State | Next state |     | Output z |
|---------------|------------|-----|----------|
|               | w=0        | w=1 |          |
| A 0 0         | 0 0        | 0 1 | 0        |
| B 0 1         | 0 0        | 1 1 | 0        |
| C 1 1         | 0 0        | 1 1 | 1        |
| 1 0           | dd         | dd  | cl       |

Fig. Improved state assignment for seq. ckt in fig①

(3)

K-map

| $y_1$ | 00 | 01 | 11 | 10 |
|-------|----|----|----|----|
| 0     | 1  | 1  | 1  | d  |
| 1     | 1  | 1  | 1  | d  |

$$y_1 = D_1$$

$$y_1 = w$$

| $y_2$ | 00 | 01 | 11 | 10 |
|-------|----|----|----|----|
| 0     | 1  | 1  | 1  | d  |
| 1     | 1  | 1  | 1  | d  |

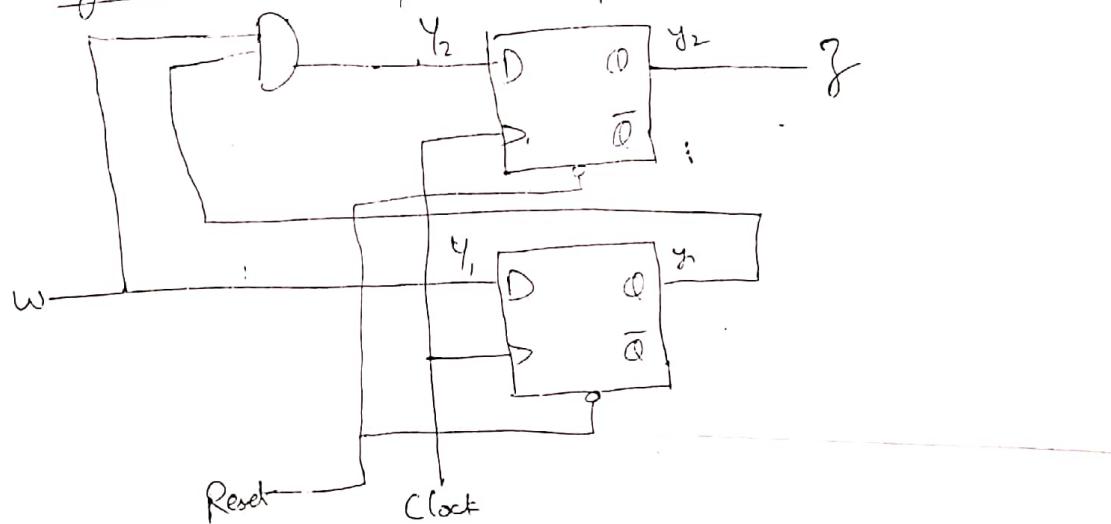
$$y_2 = D_2$$

$$y_b = w y_f$$

| $y_b$ | 00 | 01 | 11 | 10 |
|-------|----|----|----|----|
| 0     | 1  | d  | 1  | 1  |
| 1     | d  | 1  | 1  | 1  |

$$z = y_2$$

Fig. Final circuit for the improved state assignment



Q. What is state minimization? How do you identify the equivalent states?

### State Minimization:

The purpose of state minimization is to reduce the number of states in a sequential circuit so that the circuit requires fewer flip flops.

State minimization is based on the concept of the behavioral equivalence of FSMs.

### State equivalence:

2 states  $S_i$  and  $S_j$  are said to be equivalent if and only if for every possible input sequence, the same output sequence will be produced regardless of whether  $S_i$  or  $S_j$  is the initial state.

### Partitioning Minimization Procedure:

Minimize the given state table using the partitioning procedure.

| Present State | Next State |       | O/P<br>$Z$ |
|---------------|------------|-------|------------|
|               | $w=0$      | $w=1$ |            |
| A             | B          | C     | 1          |
| B             | D          | F     | 1          |
| C             | F          | E     | 0          |
| D             | B          | G     | 1          |
| E             | F          | C     | 0          |
| F             | E          | D     | 0          |
| G             | F          | G     | 0          |

The initial partition contains all states in a single block

$$P_1 = (ABCDEF)G$$

The next partition separates the states that have different outputs

ABD o/p is 1

CEFG ... 0

$$\therefore P_2 = (ABD)(CEFG)$$

(4)

Now consider all 0-successors and 1-successors of states in each block.

ABD op's when  $w=0 \Rightarrow$  BDB in same block

" "  $w=1 \Rightarrow$  CFG " " "

CEFG op's when  $w=0 \Rightarrow$  FFEF in same block

" "  $w=1 \Rightarrow$  ECDG in diff. blocks

D is from diff. block

for CEFG, when  $w=1$ , D present state is 'F'

$$\therefore P_3 = (ABD)(CEG)(F)$$

Repeating the process yields the following

ABD op's when  $w=0 \Rightarrow$  BDB in same block

" "  $w=1 \Rightarrow$  CFG in diff. blocks (see  $P_3$ )

F is from diff. block, its present state is 'B'

can be equivalent  
A & C

CEG op's when  $w=0 \Rightarrow$  FFF in same block

" "  $w=1 \Rightarrow$  ECG " "

$$\therefore P_4 = (AD)(B)(CEG)(F)$$

Repeating the process

AD op's when  $w=0$  B,B in same block

" "  $w=1$  CG " "

CEG " "  $w=0$  FFF " "

$w=1$  ECG " "

lock

$$\therefore P_5 = (AD)(B)(CEG)(F)$$

Since,  $P_5 = P_4$  and no new blocks are generated,  
it follows that states in each block are equivalent.

different

Take the 1st letters of each block

A represents both the states A and D

C " states C, E and G

Fig. Minimized state table

| Present State | Next State |     | Output Z |
|---------------|------------|-----|----------|
|               | w=0        | w=1 |          |
| A             | B          | C   | 1        |
| B             | A          | F   | 1        |
| C             | F          | C   | 0        |
| F             | C          | A   | 0        |

$$\begin{aligned}D &= A \\E &= C \\C &= C\end{aligned}$$

Finite State Machine (FSM) representation using Moore and Mealy state models:

Moore Machine: The sequential circuits in which the output depends only on states are referred as moore model circuits.

Mealy Machine: The sequential circuits in which output depends on input as well as states are referred as mealy model circuits.

#### Moore Model

The sequential circuits in which the output depends only on states are referred as moore model circuits.

Input changes does not effect the output.

Requires more number of states for implementing same functions

$$\text{Ex: } Z = A$$

#### Mealy Model

1. The sequential circuits in which output depends on input as well as on states are referred as mealy model circuits.

2. Input changes may effect the o/p of the circuit.

3. Requires less no. of states for implementing same functions

$$\text{Ex: } Y = (A+B)\bar{X}$$

Consider the sequence of values of the  $w$  and  $z$  signals during 11 clock cycles as shown in fig.

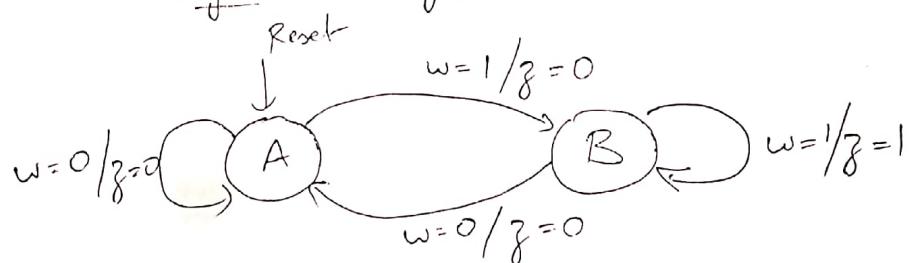
|              |       |       |       |       |       |       |       |       |       |       |          |
|--------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|----------|
| Clock cycle: | $t_0$ | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ | $t_8$ | $t_9$ | $t_{10}$ |
| $w:$         | 0     | 1     | 0     | 1     | 1     | 0     | 1     | 1     | 0     | 1     |          |
| $z:$         | 0     | 0     | 0     | 0     | 1     | 0     | 0     | 1     | 1     | 0     | 0        |

O/p  $z$  should be equal to 1 in the same clock cycle when the second occurrence of  $w=1$  is detected.

Fig. State table

As long as  $w=0$ , the machine should remain in state A, producing an output  $z=0$ . When  $w=1$ , the machine has to move to a new state, B, to record the fact that an i/p 1 has occurred.

Fig. State diag.



The state table is shown in fig. The table shows that the output  $z$  depends on the present value of input  $w$  and also on present state.

Fig. State Table

| Present State | Next State |       | Output $z$ |       |
|---------------|------------|-------|------------|-------|
|               | $w=0$      | $w=1$ | $w=0$      | $w=1$ |
| A             | A          | B     | 0          | 0     |
| B             | A          | B     | 0          | 1     |

Fig. shows the state assigned table.  
there are only 2 states, it is sufficient to use a single rate variable  $y$ .

Fig. State-assigned table for the FSM

| Present State | Next State |       | Output |       |
|---------------|------------|-------|--------|-------|
|               | $w=0$      | $w=1$ | $w=0$  | $w=1$ |
| $y$           | $y$        | $y$   | $z$    | $z$   |
| A             | 0          | 0     | 0      | 0     |
| B             | 1          | 0     | 0      | 1     |



$$y = w$$

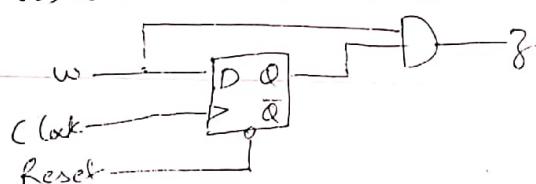
$$w = \text{i/p}$$

$y$  = present state

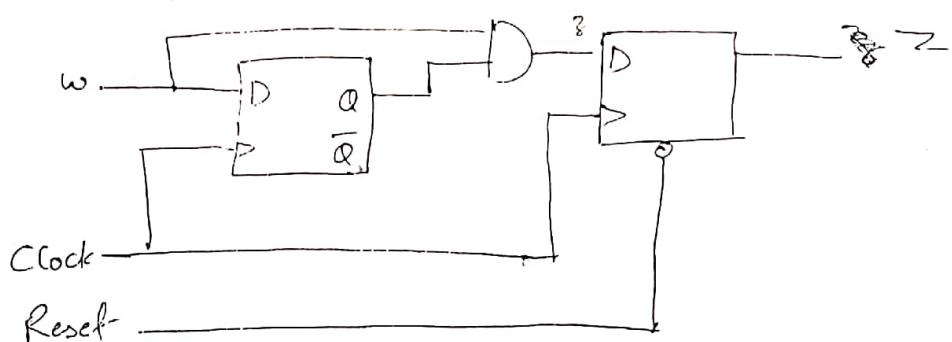


$$z = wy$$

The resultant circuit is shown in fig.



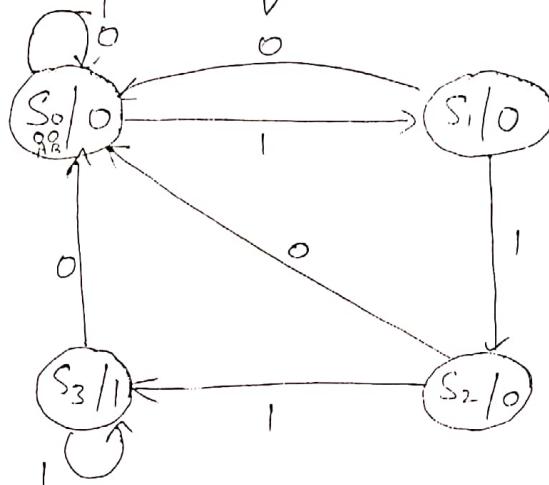
This circuit can be changed from Mealy-type to Moore type (output depends on present state)



Design of FSM circuit detecting '111'. It has one i/p and 1 o/p. The o/p should become '1' whenever the above sequence is detected.

## Design of FSM for sequence generation & detection:

State diagram for sequence detector



From the fig., it is derived by starting with state  $S_0$ , if the i/p is 0, the circuit stays in the same state, but if the i/p is 1, it goes to state  $S_1$  to indicate that a 1 was detected.

If the next input is 1, the change is to state  $S_2$  to indicate the arrival of 2 consecutive 1, but if the i/p is 0 we go back to state  $S_0$ .

The 3rd consecutive 1 sends the circuit to state  $S_3$ . If more 1's are detected, the circuit stays at  $S_3$ . Any '0' i/p sends the circuit back to  $S_0$ .

In this way, the circuit stays at  $S_3$  as long as there are 3 or more consecutive 1's received. This is a type Moore Model seq. circuit since the o/p is 1 when circuit is in state  $S_3$  and 0 otherwise.

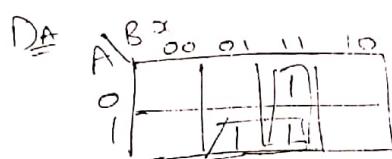
Synthesis using D ff:

In the state table, A and B are present state values of ffs A and B.  $x$  is the i/p,  $D_A$  and  $D_B$  are the i/p equations. Y is the o/p.

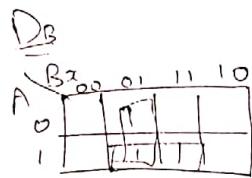
State table for sequence detector

| Present State |   | Input<br>$x$ | Next State |       | Output -<br>$y$ |
|---------------|---|--------------|------------|-------|-----------------|
| A             | B |              | $D_A$      | $D_B$ |                 |
| 0             | 0 | 0            | 0          | 0     | 0               |
| 0             | 0 | 1            | 0          | 1     | 0               |
| 0             | 1 | 0            | 0          | 0     | 0               |
| 0             | 1 | 1            | 1          | 0     | 0               |
| 1             | 0 | 0            | 0          | 0     | 0               |
| 1             | 0 | 1            | 1          | 1     | 0               |
| 1             | 1 | 0            | 0          | 0     | 1               |
| 1             | 1 | 1            | 1          | 1     | 1               |

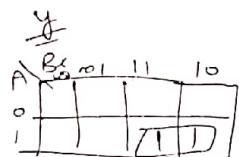
Map



$$D_A = A \cdot x + B \cdot x'$$

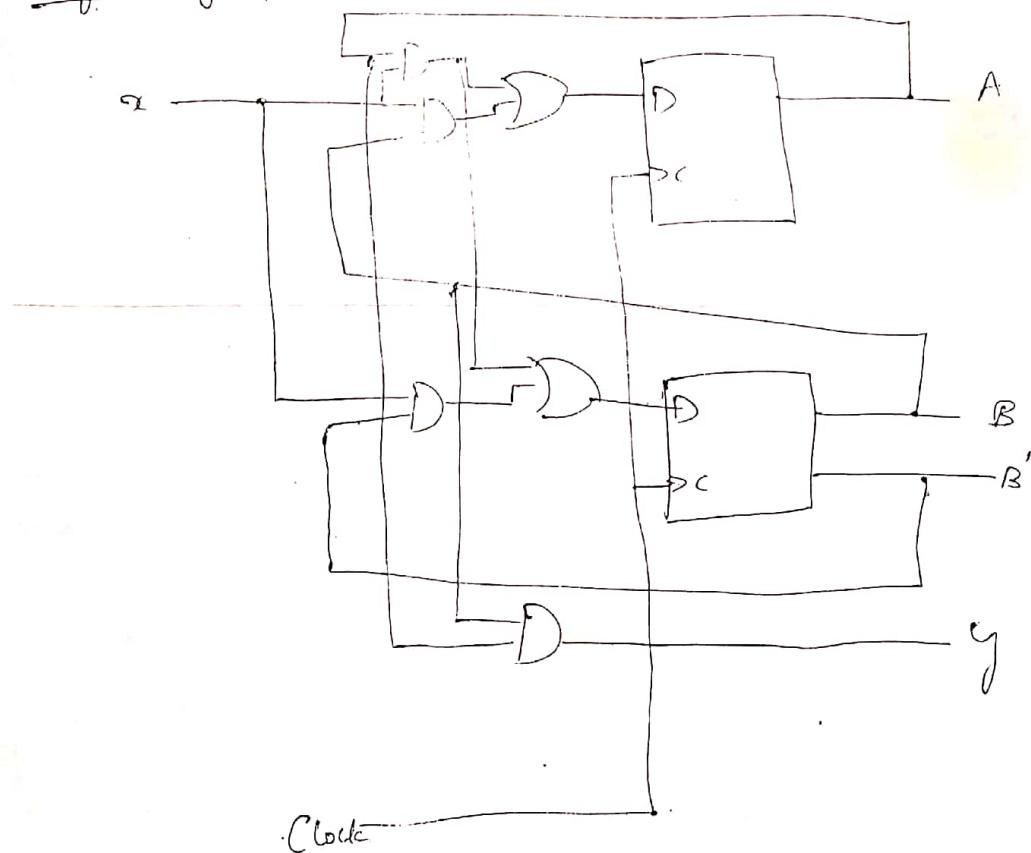


$$D_B = A \cdot x + B \cdot x'$$



$$y = AB$$

Logic diag for sequence detector



## Algorithmic State Machine (ASM) Charts:

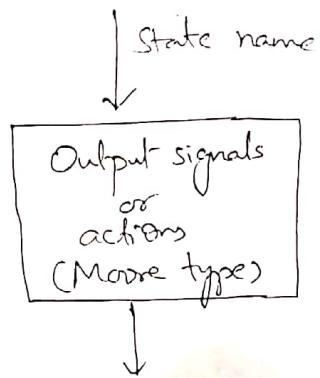
An ASM chart is a type of **flowchart** that can be used to represent the **state transitions** & generated **outputs** for an **FSM**.

The 3 types of elements used in ASM charts are:

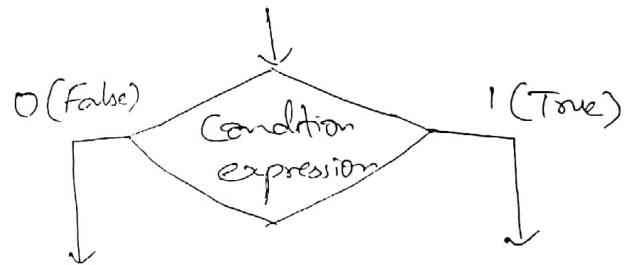
- i) State Box
- ii) Decision Box
- iii) Conditional output box

fig. Elements used in ASM charts

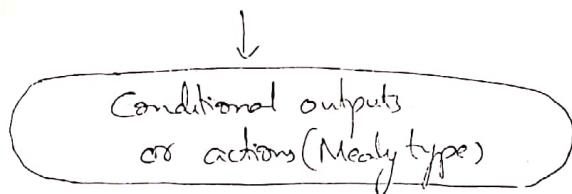
a) State box



b) Decision box



c) Conditional output box

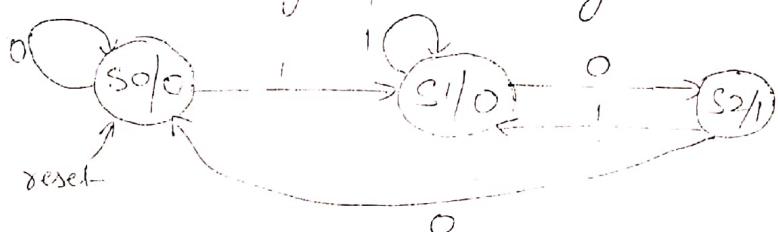


- i) **State box:** A rectangle represents a state of FSM. It is equivalent to a **node** in the state diagram or row in a state table. The name of the state is indicated outside the box in the top-left corner.

Decision box: A diamond indicates that the stated condition expression is to be tested & the exit path is to be chosen accordingly. The condition expression consists of one or more inputs to the FSM.

Conditional output box: An oval denotes the output signals that are of Mealy type. These outputs depend on the values of state variables and the inputs of FSM.

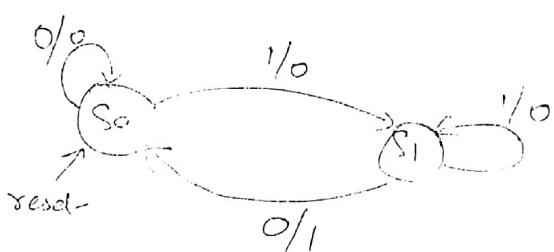
Draw the state diag. for detecting the sequence '01'



Meaning of states : S0  $\Rightarrow$  No elements of the seq. observed

S1  $\Rightarrow$  '1' observed

S2  $\Rightarrow$  "10" observed



Meaning of states : S0  $\Rightarrow$  No elements of the seq. observed  
S1  $\Rightarrow$  '1' observed