

o Digital Signal :-

A digital signal has two discrete levels or values. The two discrete signal level High and Low can also be represented by the binary digits 1 and 0 respectively.

- A binary digit (0 or 1) is referred to as bit.

- A digital signal can have only one of the two possible levels 1 or 0, the binary number system can be used for the analysis & design of digital systems.

- The two levels (or states) can also be designated as on & off or True & false.

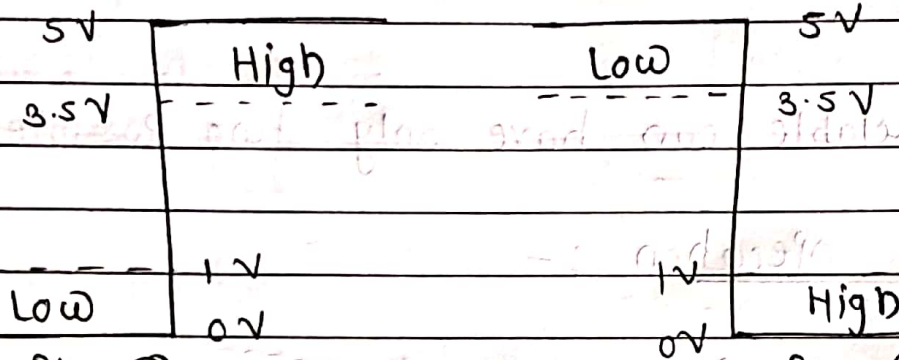


fig :- Digital signal representation

fig (a) positive logic

fig (b) Negative logic

o Logic Gate :-

- Logic gates are digital circuits.
- The building blocks of any digital system.
- It is an electronic circuit having one (or) more inputs and only one output.

OR

- The digital devices which perform boolean function by taking two inputs and giving one output.
- Logic gates are based on "Boolean Algebra".
- Logic gates are made up of diode and transistor.

Types of Logic Gates

* Basic Gates :-

- AND Gate
- OR Gate
- NOT Gate

* Universal Gates :-

- NAND Gate
- NOR Gate

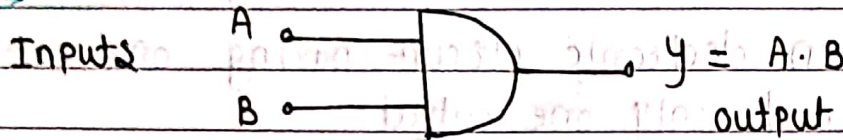
* Special Gates :-

- EX - OR Gate
- EX - NOR Gate

AND Gate :-

- An AND gate has two or more inputs and only one output which is equal to product of all inputs. It performs "Multiplication" operation.

o Symbol :-



o Boolean Expression :-

$$y = A \cdot B$$

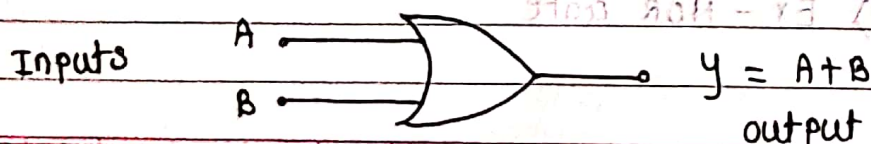
o Truth table :-

Input		Output
A	B	$y = A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

OR Gate :-

- It has two or more inputs and only one output.
- The output is equal to the sum of all inputs. It performs "Addition operation".

o Symbol :-



o Boolean Expression :-

$$Y = A + B$$

o Truth Table :-

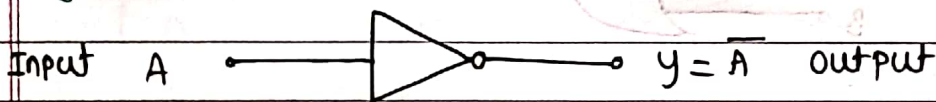
Input		Output
A	B	$Y = A + B$
0	0	0
0	1	1
1	0	1
1	1	1

IC 7432

NOT Gate or "Inverter" Gate :-

- Single input and single output.
- Output is the complement of the input logic.

o Symbol :-



o Boolean Expression :-

$$Y = \bar{A}$$

o Truth Table :-

Input	Output
A	$\bar{A}$
0	1
1	0

IC 7404

Universal Gates :-

[NOR & NAND Gate]

- It is a gate which can be used to implement any other gates without using other types of gates.

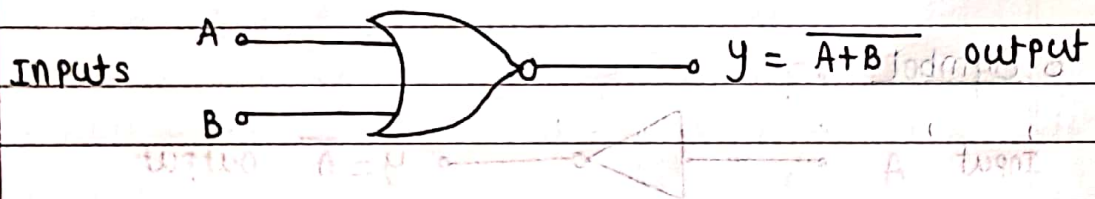
Ex: NOR gate

NAND gate

NOR Gate :-

- It has two or more inputs but only one output.
- The output is the complement of the sum of the inputs.

Symbol :-



Boolean Expression :-

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

Truth Table :-

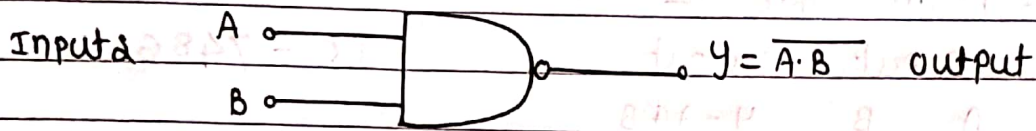
Inputs		output
A	B	$y = \overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

IC - 7402

NAND Gate :-

- It has two or more inputs but only one output.
- The output is the complement of the product of inputs.

o Symbol :-



o Boolean Expression :-

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

o Truth Table :-

Inputs		Output
A	B	$y = \overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

IC - 7400

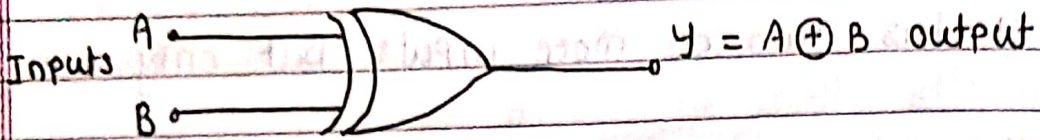
Special gates :-

[XOR & XNOR Gate]

XOR Gate :-

- Exclusive - OR Gate [EX-OR]
- It has two or more inputs & only one output.

o Symbol :-



o Boolean Expression :-

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

o Truth Table :-

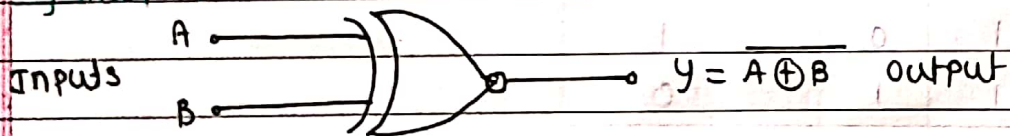
Input		output
A	B	$Y = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

IC - 7486

XNOR Gate (Exclusive - NOR gate) :-

- It has two or more inputs & only one output.
- The output is the complement of the XOR gate.

o Symbol :-



o Boolean Expression :-

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

o Truth Table

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

IC - 7426

o Boolean Algebra :-

- It is used to analyze and simplify the digital (logic) circuits. there should be no change in functionality while using the set of rules.
- It is also called as Binary or Logical algebra.
- It is invented by George Boole in 1854.

Key Point

- ① Symbols in boolean algebra doesn't represent numerical values.
- ② Arithmetic operation are not performed in boolean algebra.
- ③ A variable can have only two possible values (0, 1)

Logical operation :-

- ① Inversion, $A, y = \bar{A}$] NOT gate
- ② Logical multiplication, $Y = AB$] AND gate
- ③ Logical Addition, $Y = A + B$] OR gate
- ④ NOT AND, $Y = \overline{A \cdot B}$] NAND gate
- ⑤ NOT OR, $Y = \overline{A + B}$] NOR gate

⑥ Addition / subtraction, $Y = A \oplus B$] EX-OR Gate

⑦ NOT EX-OR, $Y = \overline{A \oplus B}$] EX-NOR Gate.

EX - OR

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

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

Rules in Boolean Algebra :-

i) Variables can have only two values

1 $\rightarrow$ High

0 $\rightarrow$ Low

ii) Complement of variable (A)

$\hookrightarrow \bar{A}$] If $A = 0, \bar{A} = 1$

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

iii) ORing (+), $A \text{ OR } B$ $A + 0 = A, A + 1 = 1$

$\hookrightarrow A + B$ $A + A = A, A + \bar{A} = 1$

iv) ANDing (.), $A \text{ AND } B$ $A \cdot 0 = 0, A \cdot 1 = A$

$\hookrightarrow A \cdot B$ $A \cdot A = A, A \cdot \bar{A} = 0$

$A + 0 = A, (A \cdot 1) = A$

o Boolean Algebra :-

- A Lattice is said to be boolean Algebra both distributive and complemented.

$$i) A + 1 = 1$$

$$A + 0 = A$$

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

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

$$ii) A \cdot 1 = A$$

$$A \cdot 0 = 0$$

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

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

$$iii) \bar{\bar{A}} = A$$

Law's of Boolean Algebra :-

1) Commutative Law :-

$$A + B = B + A$$

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

2) Distributive Law :-

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

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

3) Associative Law :-

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

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

4) Absorption Law :-

$$A + AB = A$$

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

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

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

5) Identity Laws :-

$$A + 0 = A$$

$$A \cdot 1 = A$$

6) Complement Laws :-

$$A + A' = 1$$

$$0' = 1$$

$$A \cdot A' = 0$$

$$1' = 0$$

7) Idempotent Laws :-

$$A + A = A$$

$$A \cdot A = A$$

8) Uniqueness of Complement :-

$$A + x = 1 \text{ and } A \cdot x = 0 \text{ then } x = A'$$

9) Involution Law :-

$$(A')' = A$$

10) De-morgan's Laws :-

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

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

11) Boundedness Laws :-

$$A + 1 = 1$$

$$A \cdot 0 = 0$$

Boolean Laws :-

① Commutative Law :-

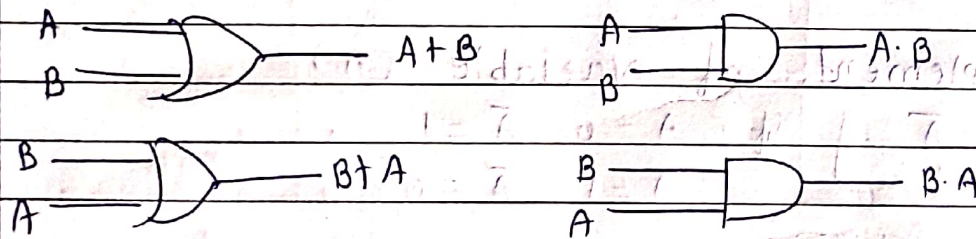
- Commutative law defines that the interchanging of variables in the AND and OR operation doesn't change the output.

① $A + B = B + A$

② $A \cdot B = B \cdot A$

Input variables		AND		OR	
A	B	$A \cdot B$	$B \cdot A$	$A + B$	$B + A$
0	0	0	0	0	0
0	1	0	0	1	1
1	0	0	0	1	1
1	1	1	1	1	1

Logic Symbol



② Associative Law :-

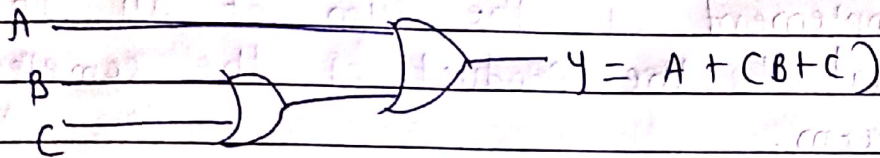
- Associative law defines that the different grouping of variables in the multivariable AND or OR operation doesn't change the output.

① $A + (B + C) = (A + B) + C$

② $A \cdot (B \cdot C) = (A \cdot B) \cdot C$

Input

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



③ Distributive law

- Distributive law defines that the distribution of variables with AND operation over OR operation is equal to the distribution of OR operation over AND operation.

① $A + B \cdot C = (A + B) \cdot (A + C)$

② $A \cdot (B + C) = AB + AC$

Input

A	B	C	B · C	A + B	A + C	A + B · C	(A + B) · (A + C)
0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0
0	1	0	0	1	0	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

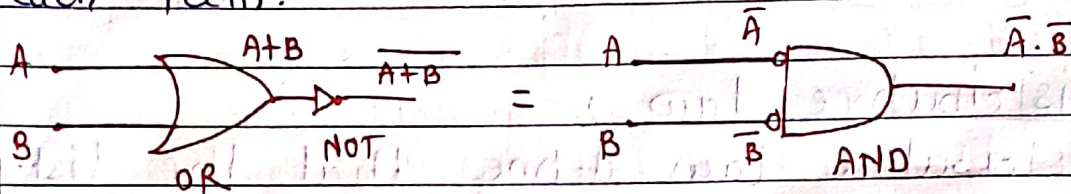
o Demorgan's Theorem :-

- A famous mathematician De-morgan derived the two most important theorems of Boolean Algebra.
- power tools used in digital system design.

i) Demorgan's first Theorem :-

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

- The complement of the sum of all the terms is equal to the product of the complement of each term.



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

Proof :- Boolean Algebra

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

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

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

i) $(A+B) + (\overline{A+B}) = 1$

ii) $(A+B) \cdot (\overline{A+B}) = 0$

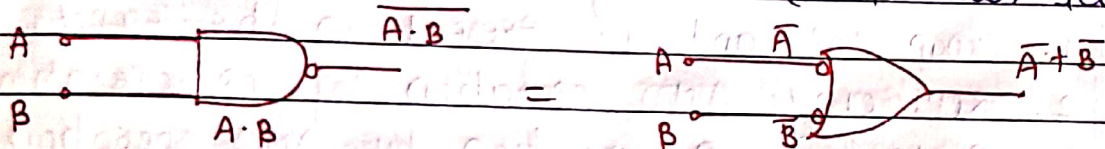
$$\begin{aligned} \rightarrow i) (A+B) + (\overline{A+B}) &= (A+B) + (\overline{A} \cdot \overline{B}) \\ &= (A+B+\overline{A}) \cdot (A+B+\overline{B}) \\ &= (1+B) \cdot (1+A) \\ &= 1 \end{aligned}$$

$$\begin{aligned} ii) (A+B) \cdot (\overline{A+B}) &= (A+B) \cdot (\overline{A} \cdot \overline{B}) \\ &= A\overline{A} + B\overline{B} \\ &= \{A \cdot (\overline{A} \cdot \overline{B}) + B \cdot (\overline{A} \cdot \overline{B})\} \\ &= 0 \end{aligned}$$

ii) Demorgan's second Theorem :-

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

- The complement of the product of all terms is equal to the sum of the complements of each term.



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

Proof :- Boolean Algebra :-

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

$$a + \overline{a} = 1$$

$$a \cdot \overline{a} = 0$$

$$i) (\overline{A \cdot B}) + A \cdot B = 1$$

$$ii) (\overline{A \cdot B}) (A \cdot B) = 0$$

$$\begin{aligned}
 \text{i) } A \cdot B + (\overline{A} \cdot \overline{B}) &= A \cdot B + (\overline{A} + \overline{B}) \\
 &= (\overline{A} + \overline{B} + A) \cdot (\overline{A} + \overline{B} + B) \\
 &= (1 + \overline{B}) \cdot (1 + \overline{A}) \\
 &= 1
 \end{aligned}$$

$$\begin{aligned}
 \text{ii) } A \cdot B \cdot (\overline{A} \overline{B}) &= AB(\overline{A} + \overline{B}) \\
 &= AB\overline{A} + AB\overline{B} \\
 &= 0
 \end{aligned}$$

• Duality Theorem :-

- Duality theorem says that in the logic function applying the following changes in the AND, OR & NOT operations doesn't affect the output. The principles of duality are,

1. Swap '0' and '1' present in the expression.
2. Replacing AND operation by OR operation.
3. Replacing OR operation by AND operation.

$$\text{Ex. i) } A + 0 = A \cdot 1 = A$$

In the above expression, OR operation is replaced by AND operation & the complement is taken for '0' present in the expression. For both logic expressions the output is same.

$$\text{ii) } A(C+B+C) = AB + AC$$

After applying duality theorem in above expression

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

A	B	C	BC	A+B	A+C	A+BC	(A+B) · (A+C)
0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0
0	1	0	0	1	0	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

o Number system :-

In general, in any number system there is an ordered set of symbols known as digits with rules defined for performing arithmetic operations like, addition, multiplication.

- The number system in which an ordered set of ten symbols = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 is known as digits are used to specify any number.

Number system : Base or Radix of Number system.

- 1) Binary
- 2) Octal
- 3) Decimal
- 4) Hexadecimal

Number system	Base	Digits used
1) Binary	2	0, 1
2) Octal	8	0, 1, 2, 3, 4, 5, 6, 7
3) Decimal	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
4) Hexadecimal	16	0-9, A, B, C, D, E, F

[Where A=10, B=11, C=12, D=13, E=14, F=15]

o Binary Number System :-

- In the binary number system, the total number of symbols are two (0, 1); base is 2, and the radix point is known as the binary point.
- These symbols are known as bits.

$$\text{Ex} - (1111.111)_2 = 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3}$$

o Octal Number system :-

- The number system with base 8 is known as the octal number system.
- Eight symbols are used to represent numbers in the this system & these are 0, 1, 2, 3, 4, 5, 6, 7 & 7

$$\text{Ex} :- (576.217)_8 = 5 \times 8^2 + 7 \times 8^1 + 6 \times 8^0 + 2 \times 8^{-1} + 1 \times 8^{-2} + 7 \times 8^{-3}$$

o Decimal Number system

- In the decimal number system, the total number of symbols are ten (0, 1, 2, 3, 4, 5, 6, 7, 8, 9); base is 10 & the radix point is known as the decimal point.
- The leftmost digit has the max^m value, known as most significant bit digit (MSD).

- The rightmost digit has the min^m value & hence it is known as least significant digit (LSD).

$$\text{Ex:- } (555.55)_{10} = 5 \times 10^2 + 5 \times 10^1 + 5 \times 10^0 + 5 \times 10^{-1} + 5 \times 10^{-2}$$

Hexadecimal Number system:-

- The number system with base 16 is known as the hexadecimal number system.

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

$$\text{Ex:- } (215A.268)_{16} = 2 \times 16^3 + 1 \times 16^2 + 5 \times 16^1 + 10 \times 16^0 + 2 \times 16^{-1} + 6 \times 16^{-2} + 8 \times 16^{-3}$$

* Binary to Decimal Number conversion:-

$$\begin{aligned} \text{Ex 1)} \quad (11011)_2 &= 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 \\ &= 16 + 8 + 2 + 1 \\ &= (27)_{10} \end{aligned}$$

$$\begin{aligned} 2) \quad (0.1101)_2 &= 1 \times 2^{-1} + 1 \times 2^{-2} + 0 \times 2^{-3} + 1 \times 2^{-4} \\ &= 0.5 + 0.25 + 0.0625 \\ &= (0.8125)_{10} \end{aligned}$$

$$\begin{aligned} 3) \quad (1101.1101)_2 &= 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} + 0 \times 2^{-3} + 1 \times 2^{-4} \\ &= 8 + 4 + 0 + 1 + 0.5 + 0.25 + 0 + 0.0625 \\ &= (13.8125)_{10} \end{aligned}$$

Combinational Logic DesignDate _____
Page _____o Standard representation for logic functions :-

- Logic functions are expressed in terms of logical variables. The values assumed by the logic function as well as the logic variables are in the binary form.
- Any logic expression can be expressed in the following two standard forms.

1. Sum of Product form (SOP)

2. Product of Sum form (POS)

These two forms are suitable for reducing the given logic to its simplest form.

1) Sum of Product form (SOP) :-

- Sum of product term is consisting of sum (OR operation) of many terms.
- The sum of the set of product terms is called as "SOP" function.

$$Y = AB + AC + BC$$

↑ ↑ ↑
Product terms
↑ ↑
Sum

This logic expression is in the form of sum of three terms AB, AC and BC with each individual term is product of two variables. Therefore such expressions are known as expression in SOP form.

2) Product of sum form (POS) :-

- Product of sum terms is consisting of product (AND operation) of many terms.
- The product of the set of sum terms is called pos function.

$$Y = \underbrace{(A+B)}_{\text{sum terms}} \cdot \underbrace{(B+C)}_{\text{sum terms}} \cdot \underbrace{(A+C)}_{\text{sum terms}}$$

products

- It is in the form of product of three terms $(A+B)$, $(B+C)$ and $(A+C)$ with each term is in the form of sum of two variables. Such expression are said to be in the product of sum (POS) form.

Standard or canonical SOP and POS forms :-

- Each term used in a switching equation must contain all the available input variables.
- The two formats of a switching eqⁿ in the standard form are :
 - ① Sum of product (SOP) format
 - ② Product of sum (POS) format
- Many times switching equations written in SOP or POS form are not standard that means each term may not contain all the input variables.

o Standard SOP form :-

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

Each individual term is called "minterm"

o Standard POS form :-

$$Y = \underbrace{(A+B)}_{\uparrow} \cdot \underbrace{(A+\bar{B})}_{\uparrow}$$

Each individual term is called "maxterm"

o K-map representation of logic functions (SOP & POS forms) :-

- K-map is nothing but "Karnaugh map".

- This is another simplification technique to reduce the Boolean equation.

- The information contained in a truth table or available in the SOP or POS form is represented on a K-map.

K-map structure :-

* 2 variable K-map :

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

fig :- 2 variable K-map

- The structure of 2 input (variable) K-map is shown above fig. A 2 variable K-map consists of $2^2 = 4$ rectangular boxes. Inside these boxes we have to enter the values of output, Y for different combinations of inputs A & B.

* 3 variable K-map :

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

fig :- 3 variable K-map

- The structure of 3 input (variable) K-map is shown in above fig. A 3 variable K-map consists of $2^3 = 8$ rectangular boxes. Inside these boxes we have to enter the values of output Y for different combinations of input A, B & C.

* 4 variable K-map :

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

fig :- 4 variable K-map

- The structure of 4 input (variables) K-map is shown in above fig. A 4 variable K-map consists of $2^4 = 16$ rectangular boxes. Inside these boxes we have to enter the values of output Y for different combinations of inputs A, B, C & D.

* - The logical expression in standard SOP form can be represented on a K-map by simply entering 1's in the cells (boxes) of the K-map corresponding to each minterm present in the equation. The remaining cells (boxes) are filled with zero.

Ex. Represent the equation given below on K-map.

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

Ans :- The given expression is in the standard SOP form. Each term represents a minterm. We have to enter 1's in the boxes corresponding to each minterm.

These are 3 variables, so $2^3 = 8$ cells (boxes)

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

Logical expression in the standard POS form can be represented on K-map by entering 0's in the cells of K-map corresponding to each maxterm present in the given eqⁿ. Remaining cells are filled with 1's.

Ex. Represent the following standard pos expression on K-map.

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

Ans :- Each term in the given logical equation is a maxterm. Enter 0 corresponding to each maxterm. The given expression has three maxterm.

$$(A+B+C) = M_0, (A+\bar{B}+C) = M_2, (\bar{A}+\bar{B}+C) = M_6$$

Hence we have to write the structure of 3-variable K-map as usual & enter 0's at M_0, M_2 & M_6 .

$M_0 = (A+B+C)$

	BC	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$	
A	$\bar{A}$	0	1	1	0	$M_2 = (A+\bar{B}+C)$
A	A	1	1	1	0	$M_6 = (\bar{A}+\bar{B}+C)$

0's

$M_0 = (A+B+C)$

	BC	00	01	11	10	
A	0	0	1	1	0	$M_2 = (A+\bar{B}+C)$
A	1	1	1	1	0	$M_6 = (\bar{A}+\bar{B}+C)$

fig :- Representation of standard pos on K-map.

- Minimization of logical function for minterms and maxterms (Upto 4 variables) :

Minterm :- Each individual term in the standard sop form is called as "minterm".

Maxterm :- Each individual term in the standard pos form is called as "Maxterm".

Standard sop : $Y = \underbrace{ABC}_{\uparrow} + \underbrace{A\bar{B}\bar{C}}_{\uparrow} + \underbrace{\bar{A}BC}_{\uparrow}$

Each individual term is called "minterm"

Standard POS : $Y = \underbrace{(A+B)}_{\uparrow} \cdot \underbrace{(A+\bar{B})}_{\uparrow}$

Each individual term is called "Maxterm"

Variables			Minterms	Maxterms
A	B	C	m_i	M_i
0	0	0	$\bar{A}\bar{B}\bar{C} = m_0$	$A+B+C = M_0$
0	0	1	$\bar{A}\bar{B}C = m_1$	$A+B+\bar{C} = M_1$
0	1	0	$\bar{A}B\bar{C} = m_2$	$A+\bar{B}+C = M_2$
0	1	1	$\bar{A}BC = m_3$	$A+\bar{B}+\bar{C} = M_3$
1	0	0	$A\bar{B}\bar{C} = m_4$	$\bar{A}+B+C = M_4$
1	0	1	$A\bar{B}C = m_5$	$\bar{A}+B+\bar{C} = M_5$
1	1	0	$AB\bar{C} = m_6$	$\bar{A}+\bar{B}+C = M_6$
1	1	1	$ABC = m_7$	$\bar{A}+\bar{B}+\bar{C} = M_7$

* Representation of Logical Expressions using Minterm & Maxterms.

1. $Y = \underbrace{ABC}_{m_7} + \underbrace{\bar{A}BC}_{m_3} + \underbrace{A\bar{B}\bar{C}}_{m_4}$ ----- logical expression
----- corresponding minterm.

$\therefore Y = m_7 + m_3 + m_4$

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

where Σ denotes sum of products

$$2. Y = \underbrace{(A + \bar{B} + C)}_{M_2} \underbrace{(A + B + C)}_{M_0} \underbrace{(\bar{A} + \bar{B} + C)}_{M_6} \dots \text{logical expression}$$

corresponding maxterm

$$Y = M_2 M_0 M_6$$

$$Y = \Pi M(0, 2, 6)$$

where Π denotes product of sum.

Example 2 :-

Q. Simplify the following three variable Boolean expression
 $Y = \Sigma m(2, 4, 6)$

Ans :- The given expression can be written in sop form as follows

$$Y = m_2 + m_4 + m_6$$

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

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

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

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

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

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

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

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

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

This is simplified expression.

Q. Simplify the following three variable logic expressⁿ.
 $Y = \pi M(1, 3, 5)$

Ans :- The given expression can be expressed in terms of maxterms as,

$$Y = M_1 \cdot M_3 \cdot M_5$$
$$= (A+B+\bar{C})(A+\bar{B}+\bar{C})(\bar{A}+B+\bar{C})$$

Bring the expression into SOP form

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

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

but $A\bar{A} = 0$, $B\bar{B} = 0$ and $\bar{C}\bar{C} = \bar{C}$

$$\therefore Y = (A+B+\bar{C})(0 + AB + A\bar{C} + \bar{A}\bar{B} + 0 + \bar{B}\bar{C} + \bar{A}\bar{C} + B\bar{C} + \bar{C})$$

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

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

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

but $(A+\bar{A}) = 1$, $(\bar{B}+B) = 1$

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

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

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

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

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

but,

$$AAB = AB, A\bar{A}\bar{B} = 0, ABB = AB, \bar{A}B\bar{B} = 0 \text{ and } \bar{C}\bar{C} = \bar{C}$$

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

but, $AB + AB = AB$

$$\begin{aligned}
 Y &= AB + A\bar{C} + B\bar{C} + AB\bar{C} + \bar{A}\bar{B}\bar{C} + \bar{C} \\
 &= AB + AB\bar{C} + A\bar{C} + B\bar{C} + \bar{A}\bar{B}\bar{C} + \bar{C} \\
 &= AB(C + \bar{C}) + A\bar{C} + B\bar{C} + \bar{A}\bar{B}\bar{C} + \bar{C} \\
 &= AB + A\bar{C} + B\bar{C} + \bar{A}\bar{B}\bar{C} + \bar{C} \dots\dots 1 + \bar{C} = 1 \\
 &= AB + \bar{C}(A + B + \bar{A}\bar{B} + 1)
 \end{aligned}$$

$$Y = AB + \bar{C} \dots\dots\dots (A + B + \bar{A}\bar{B} + 1) = 1$$

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

This is the simplified expression.

Q. Convert the expression $Y = M_1, M_3, M_5$ into minterm & simplify.

Ans :- The given boolean expression is

$$Y = \prod M(1, 3, 5)$$

using the complementary relationship betⁿ sop & pos forms we can write

$$\begin{aligned}
 Y &= \sum m(0, 2, 4, 6, 7) \\
 &= m_0 + m_2 + m_4 + m_6 + m_7 \\
 &= \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + A\bar{B}\bar{C} + AB\bar{C} + ABC \\
 &= \bar{C}(\bar{A}\bar{B} + \bar{A}B + A\bar{B} + AB) + ABC \\
 &= \bar{C}[\bar{A}(\bar{B} + B) + A(\bar{B} + B)] + ABC
 \end{aligned}$$

$$\text{But, } (\bar{B} + B) = 1$$

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

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

o Minimization of SOP Expressions & K-map Simplification

Minimization Procedure :

- step 1 Prepare the k map and place 1's according to the given truth table or logical expression.
fill the remaining cells by 0's.
- step 2 Locate the isolated 1's i.e. the 1's which can not be combined with any other 1. Encircle such 1's.
- step 3 Identify the 1's which can be combined to form a pair in only one way and encircle them.
- step 4 Identify the 1's which can form a quad in only one way and encircle them.
- step 5 Identify the 1's which can form an octet in only one way and encircle them.
- step 6 After identifying the pairs, quads and octets, check if any 1 is yet to be encircled. If yes then encircle them with each other or with the already encircled 1's (by means of overlapping).

Note that the no. of groups should be minimum also any 1 can be included any no. of times without affecting the expression.

Ex. 1) A logical expression in the standard SOP form is as follows

$$Y = \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}Bc + A\bar{B}c$$

minimize it using the K-map technique.

Ans :- Given :- $Y = \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}Bc + A\bar{B}c$

$$\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{array}$$

$$Y = \sum m(0, 2, 3, 5)$$

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

Pair 1 : $\bar{A}B$

ABC

Pair 2

$\bar{A}\bar{C}$

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

Ex. 2) For the logical expression given below draw K map and obtain the simplified logical expression.

$$Y = \sum m(1, 5, 7, 9, 11, 13, 15)$$

Realize the minimized expression using the basic gates.

Ans :- Given :- $Y = \sum m(1, 5, 7, 9, 11, 13, 15)$

$$= m_1 + m_5 + m_7 + m_9 + m_{11} + m_{13} + m_{15}$$

It can be expressed on K-map as given below

AB \ CD	00	01	11	10
00	0	1	0	0
01	0	1	1	0
11	0	1	1	0
10	0	1	1	0

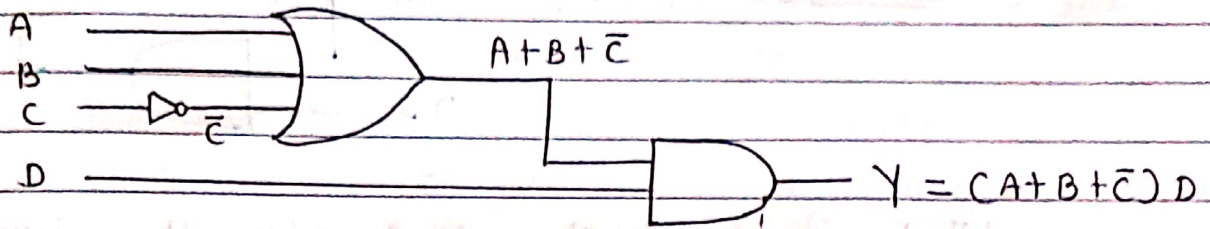
$$\text{Quad 1} = \bar{C}D$$

$$\text{Quad 2} = BD$$

$$\text{Quad 3} = AD$$

$$\therefore Y = \underbrace{\bar{C}D}_{\text{Quad}_1} + \underbrace{BD}_{\text{Quad}_2} + \underbrace{AD}_{\text{Quad}_3} \dots \text{simplified logical expression}$$

Realization



Ex 8 Minimize the following Boolean expression using k-map and realize it using the basic gates.

$$Y = \sum m(1, 3, 5, 9, 11, 13)$$

Ans:- The given equation can be expressed in terms of the minterms as

$$Y = m_1 + m_3 + m_5 + m_9 + m_{11} + m_{13}$$

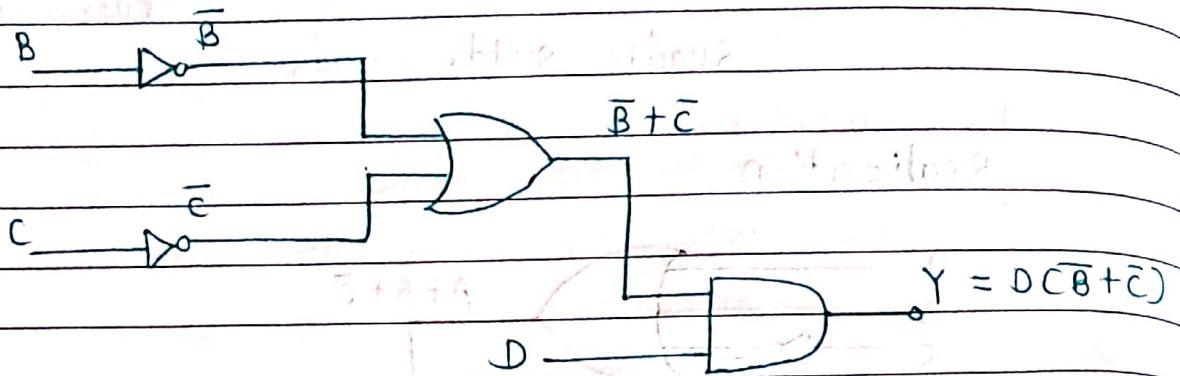
The corresponding k-map is given by

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

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

This is the simplified expression.

Realization using Basic gates :-



Ex. 4) What is Karnaugh map? Give the structure of 4 variable k-map. Simplify the following boolean equation using k-map.

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

Ans :- Given :- K map for the given equation is as follows

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

AB \ CD	CD				
	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$	
$\bar{A}\bar{B}$ 00	0	1	1	1	Pair 1 = $\bar{A}\bar{C}\bar{D}$
$\bar{A}B$ 01	0	0	0	1	
AB 11	1	1	0	1	Pair 2 = $BC\bar{D}$
$A\bar{B}$ 10	1	1	1	0	

Quad 1 = $A\bar{C}$

Quad 2 = $\bar{B}D$

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

◦ Don't Care Condition :-

- For SOP form, we enter 1's corresponding to the combinations of input variables which produce a high output and we enter 0's in the remaining cells of the K-map.
- For the POS form we enter 0's corresponding to the combinations of inputs which produce a low output and enter 1's in the remaining cells of the K-map.
- But is not always true that the cells not containing 1's (in SOP) will contain 0's because some combinations of input variable do not occur.
- Also for the some functions the outputs corresponding to certain combinations of the input variables do not matter.
- In such situation we have a freedom to assume 0 or 1 as output for each of these combinations. These conditions are known as the "Don't care" conditions & in the K-map it is represented as X (cross) mark in the corresponding cell.
- The don't care condition (X) may be assumed to be 0 or 1 as per the need for simplification.

Ex. 1) Simplify the expression given below using K-map. The don't care conditions are indicated by d().

$$Y = \sum m(1, 3, 7, 11, 15) + d(0, 2, 5)$$

Ans: - The given eqⁿ is

$$Y = \sum m(1, 3, 7, 11, 15) + d(0, 2, 5)$$

$$Y = m_1 + m_3 + m_7 + m_{11} + m_{15} + d(0, 2, 5)$$

Regular minterms so enter 1's
Don't care condition so enter x marks

∴ Required K-map is

AB \ CD	CD			
	00	01	11	10
$\bar{A}\bar{B}$ 00	x	1	1	x
$\bar{A}B$ 01	0	x	1	0
AB 11	0	0	1	0
$A\bar{B}$ 10	0	0	1	0

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

Note :- Every don't care mark need not be considered while grouping.

Ex. 2) Minimize the following expression using K-map

$$Y = \sum m(0, 1, 5, 9, 13, 14, 15) + d(3, 4, 7, 10, 11)$$

The given equation is

$$Y = \sum m(0, 1, 5, 9, 13, 14, 15) + d(3, 4, 7, 10, 11)$$

$$Y = m_0 + m_1 + m_5 + m_9 + m_{13} + m_{14} + m_{15} + [d(3, 4, 7, 10, 11)]$$

$\therefore$ K-map is given by

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

$$Y = D + \bar{A}\bar{C} + AC$$

$$= D + (A \odot C)$$

Ex - NOR

Ex-3) Simplify the following Boolean function using K-map

$$f(A, B, C, D) = \sum m(4, 5, 6, 7, 8, 12) + d(1, 2, 3, 9, 11, 14)$$

The given function is

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

K-map for the given function is

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

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

o Design Examples :

i) Arithmetic Circuits :-

* Binary adder -

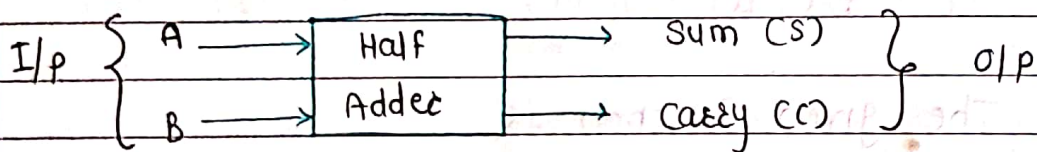
The most basic arithmetic operation is addition. The circuit which performs the addition of two binary numbers is known as Binary adder.

The Binary adders are of two types

1. Half adder
2. Full adder

1. Half adder :-

Half adder is a combinational logic circuit with two inputs and two outputs. The half adder circuit is designed to add two single bit binary number A and B. It is the basic building block for addition of two single bit numbers. The circuit has two outputs carry & sum.



Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

K-map for Sum & carry outputs.

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

fig. K-map for sum

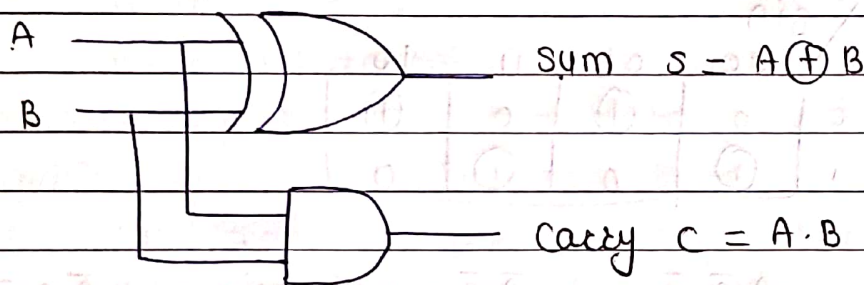
$\therefore S = A\bar{B} + \bar{A}B$
 $= A \oplus B$

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

fig. K-map for carry

$\therefore C = A \cdot B$

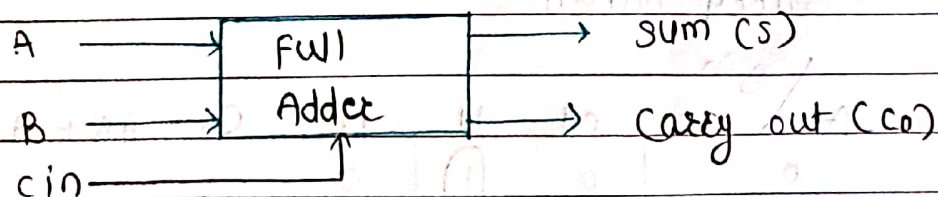
$\therefore$ Half adder circuit is shown below



2. Full adder :-

The addition of three bits is not possible to perform by using a half adder. Hence we can not use a half adder in practice.

Full adder is a combinational logic circuit with three inputs and two output. It can add two one bit numbers A & B and carry c_{in} .



Inputs			output	
A	B	cin	S	Co
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

K-map :

for sum (S) output :-

A \ B cin				
	00	01	11	10
0	0	1	0	1
1	1	0	1	0

$$S = A \bar{B} \bar{c}_{in} + \bar{A} \bar{B} c_{in} + A B c_{in} + \bar{A} B \bar{c}_{in}$$

$$= \bar{c}_{in} (A \bar{B} + \bar{A} B) + c_{in} (A B + \bar{A} \bar{B})$$

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

$$\therefore S = \bar{c}_{in} x + c_{in} \bar{x}$$

$$= c_{in} \oplus x$$

$$= c_{in} \oplus (\bar{A} B + A \bar{B})$$

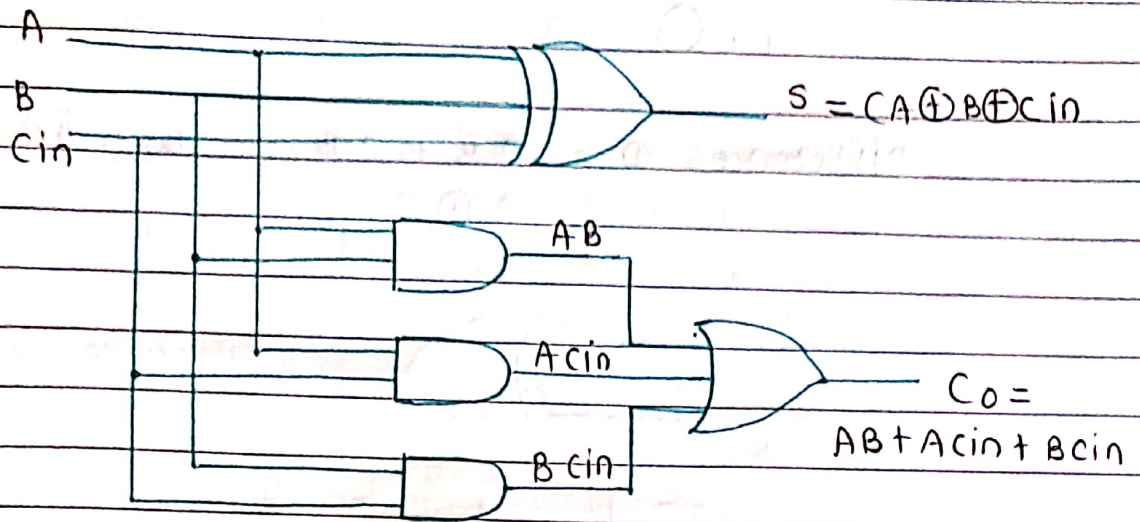
$$= c_{in} \oplus A \oplus B$$

For carry output :-

A \ B cin				
	00	01	11	10
0	0	0	1	0
1	0	1	1	1

$$C_o = AB + A c_{in} + B c_{in}$$

Logic diagram for full adder :-

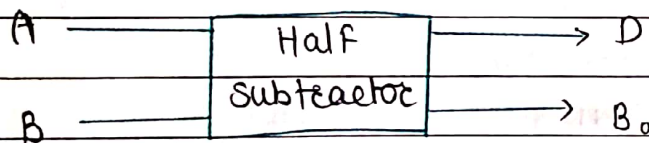


* Half Subtractor :-

Half subtractor is a combinational circuit with two inputs and two outputs (difference & borrow).

It produces the difference betⁿ the two binary bits at the input and also produces output (Borrow) to indicate if a 1 has been borrowed.

In the subtraction $(A - B)$; A is called a minused bit and B is called as subtracted bit.



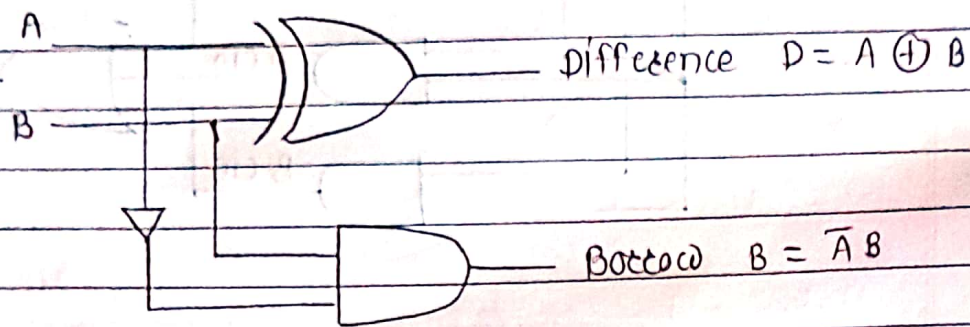
(a) Block diagram

Inputs		Output
A	B	$D = A - B$
0	0	0
0	1	1
1	0	1
1	1	0

(b) Truth table

A \ B	0	1
0	0	1
1	1	0

Difference $D = A\bar{B} + \bar{A}B$ $B_0 = \bar{A}B$
 $= A \oplus B$



* Full Subtractor :-

It is a combinational circuit with three i/p A, B and Bin and two outputs D and Bo.

A is the minuend, B is subtrahend, Bin is the borrow produced by the previous stage, D is the difference output and Bo is the borrow output.

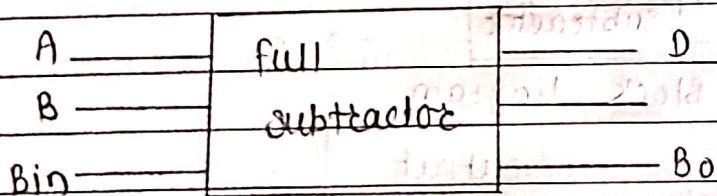


fig @ Block diagram

Truth table for full subtractor :-

Inputs			Output D (A - B - Bin)	Bo
A (Minued)	B (Subtrahend)	Bin Previous Borrow		
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

K-map :

For Difference output :-

A \ B Bin					
	00	01	11	10	
0	0	①	0	①	$D = \bar{A}\bar{B}Bin + \bar{A}B\bar{B}in + A\bar{B}\bar{B}in + ABBin$
1	①	0	①	0	

For Borrow output :-

A \ B Bin					
	00	01	11	10	
0	0	①	①	①	$B_o = \bar{A}Bin + \bar{A}B + BBin$
1	0	0	①	0	

The difference output (D) can be simplified as

$$\begin{aligned}
 D &= \bar{A}\bar{B}Bin + \bar{A}B\bar{B}in + A\bar{B}\bar{B}in + ABBin \\
 &= Bin(\underbrace{\bar{A}\bar{B} + AB}_{\text{Ex-NOR}}) + \bar{B}in(\underbrace{\bar{A}B + A\bar{B}}_{\text{Ex-OR}})
 \end{aligned}$$

Ex-NOR

Ex-OR

$$D = B \text{ in } (A \oplus B) + \bar{B} \text{ in } (A \oplus B)$$

Let $A \oplus B = C$

$$\begin{aligned} \therefore D &= B \text{ in } \bar{C} + \bar{B} \text{ in } C \\ &= B \text{ in } \oplus C \\ &= B \text{ in } \oplus A \oplus B \end{aligned}$$

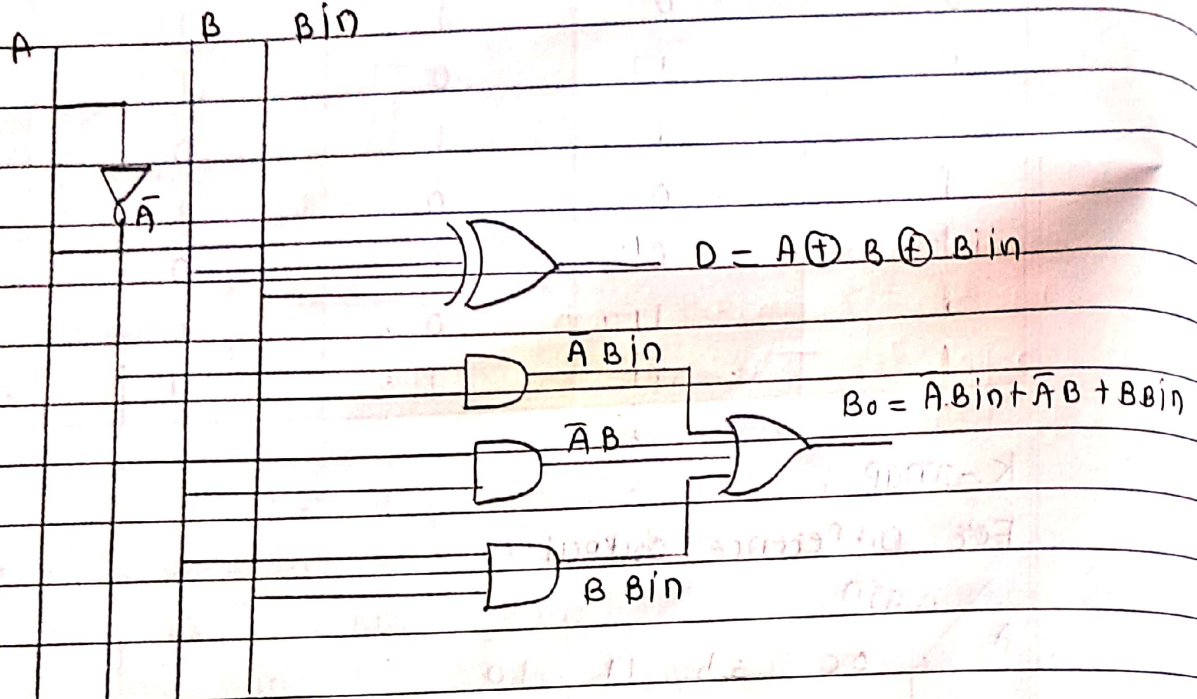


fig:-

o BCD to 7 segment decoder :-

- IC 7446 A, 7447 A or 74LS47 are 16 pin TTL ICs. They accept four inputs A_3, A_2, A_1, A_0 with A_3 represent MSB and A_0 LSB.
- They provide seven outputs $\bar{a}, \bar{b}, \dots, \bar{f}$ which are active low outputs. Each of these outputs are capable of driving the LED segments of a seven segment display directly.

- Each output is capable of sinking a current of 40 mA, in its active (low) state.
- As the output are active low, these ICs are suitable for driving the common anode seven segment LED displays.

The pin configuration of these ICs is shown in fig:

				Pin name	Description
A ₀	1		16	VCC	
A ₁	2		15	$\bar{F}$	
LT	3	IC 7446A /	14	$\bar{g}$	A ₀ , A ₁ , A ₂ , A ₃ BCD input
	4	7447 A /	13	$\bar{a}$	$\bar{a}$ to $\bar{g}$ Active low o/p
	5	74LS47	12	$\bar{b}$	$\bar{LT}$ Lamp test
	6		11	$\bar{c}$	$\overline{RBI}$ Ripple blanking i/p
	7		10	$\bar{d}$	$\overline{BI}$ Blanking i/p
GRND	8		9	$\bar{e}$	$\overline{RBO}$ Ripple blanking o/p

Fig. Pin Configuration of 7446 A, 7447 A & 74LS47

- $\bar{a}, \bar{b}, \dots, \bar{g}$ are seven active low open collector o/p's. We have to connect an external pull up resistor to each o/p pin.
- $\overline{RBI}$ (Ripple blanking input) :-
For the normal decoding operation, this i/p, should be connected to logic 1.
- If $\overline{RBI}$ is connected to ground, then it switch off the display when BCD inputs corresponds to for non-zero BCD inputs, the decoder o/p will be normal & BCD no. will be displayed.
- $\overline{RBI} = 0$ is connected for blanking out the leading zero's in multidigit displays.

* BI (Blanking input) :-

- If BI is connected to 0, then the display will be switched off irrespective of the BCD i/p.
- This feature is used in the multiplexed display in order to save power.

* RBO (Ripple blanking output) :-

- This o/p is normally at logic 1.
- But it goes to logic 0 during the zero blanking interval when RBI is forced to a low level.

* Lamp test (LT) :-

- This pin can be used to check whether all the segments of the display are working properly or not.
- If $\overline{LT}$ is forced low with $\overline{RBO}$ at logic 1 or open, then all the o/p terminals ($\overline{a}$ to $\overline{g}$) will be forced to their active state i.e. Low.
- If a display is connected then all its segments will glow simultaneously.
- Now if $\overline{RBO}$ / $\overline{BI}$ pin is forced to a logic 0, then all the outputs will be blanked.
- Ripple blanking :- The pin $\overline{RBO}$ / $\overline{BI}$ & $\overline{RBI}$ are used for achieving ripple blanking.

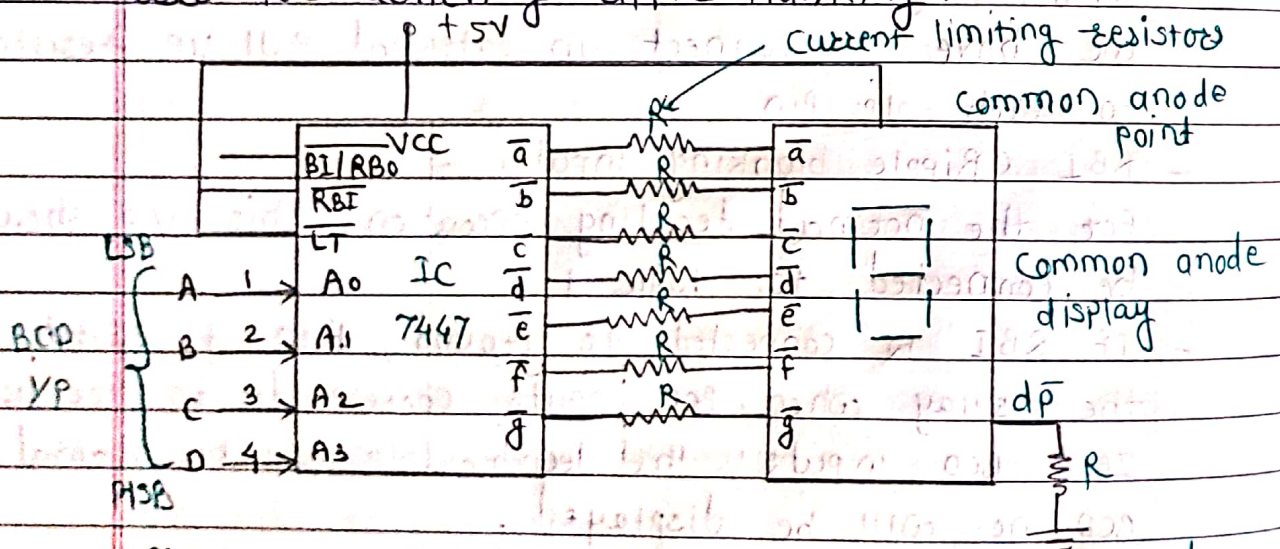


fig:- Use of 7447 to drive a common anode display.

o Code Converters :-

1. Binary to Gray code converter

Step-I Truth Table

Decimal	Binary inputs				Gray outputs			
	B ₃	B ₂	B ₁	B ₀	G ₃	G ₂	G ₁	G ₀
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	0	0	0
9	1	0	0	1	1	0	0	1
10	1	0	1	0	1	0	1	1
11	1	0	1	1	1	0	1	0
12	1	1	0	0	1	1	0	0
13	1	1	0	1	1	1	0	1
14	1	1	1	0	1	1	1	1
15	1	1	1	1	1	1	1	0

Step-II K-map for each gray output & obtain simplified exprⁿ.

B ₃ \ B ₂	B ₁ B ₀ 00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

$$G_3 = B_3$$

K-map for G₃

K map for G_2

$B_3 B_2 \backslash B_1 B_0$	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	0	0	0	0
10	1	1	1	1

$$G_2 = \bar{B}_3 B_2 + B_3 \bar{B}_2$$

$$G_2 = B_3 \oplus B_2$$

K-map for G_1

$B_3 B_2 \backslash B_1 B_0$	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	1	1	0	0
10	0	0	1	1

$$G_1 = B_2 \bar{B}_1 + \bar{B}_2 B_1$$

$$= B_2 \oplus B_1$$

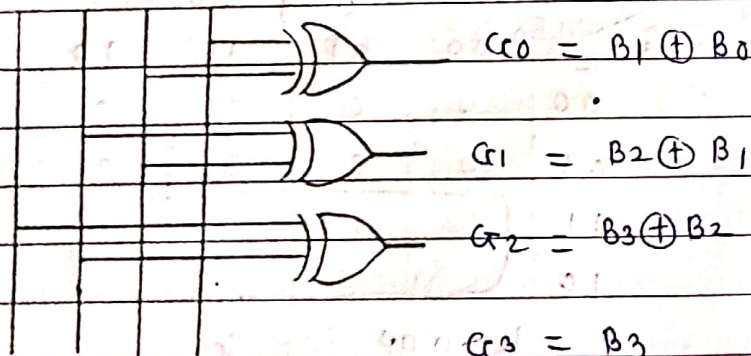
K map for G_0

$B_3 B_2 \backslash B_1 B_0$	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	0	1	0	1
10	0	1	0	1

$$G_0 = \bar{B}_1 B_0 + B_1 \bar{B}_0$$

$$= B_1 \oplus B_0$$

Step-III Realize Binary to Gray code converter :



o BCD to Gray code converter :-

Step-I Truth Table

Decimal	BCD inputs				Gray output			
	D ₃	D ₂	D ₁	D ₀	G ₃	G ₂	G ₁	G ₀
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1

Step-II K-map for each output & get simplified eqⁿ

For output G₃

D ₃ D ₂	D ₁ D ₀			
	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	X	X	X	X
10	1	1	X	X

$$G_3 = D_3$$

For output G₂

D ₃ D ₂	D ₁ D ₀			
	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	X	X	X	X
10	1	1	X	X

$$G_2 = D_2 + D_3$$

For output C_1

$D_3 D_2$ \ $D_1 D_0$		00		01		11		10	
		00	01	11	10	00	01	11	10
00		0	0	1	1				
01		1	1	0	0				
11		X	X	X	X				
10		0	0	X	X				

$$C_1 = D_2 \bar{D}_1 + \bar{D}_2 D_1$$

$$= D_2 \oplus D_1$$

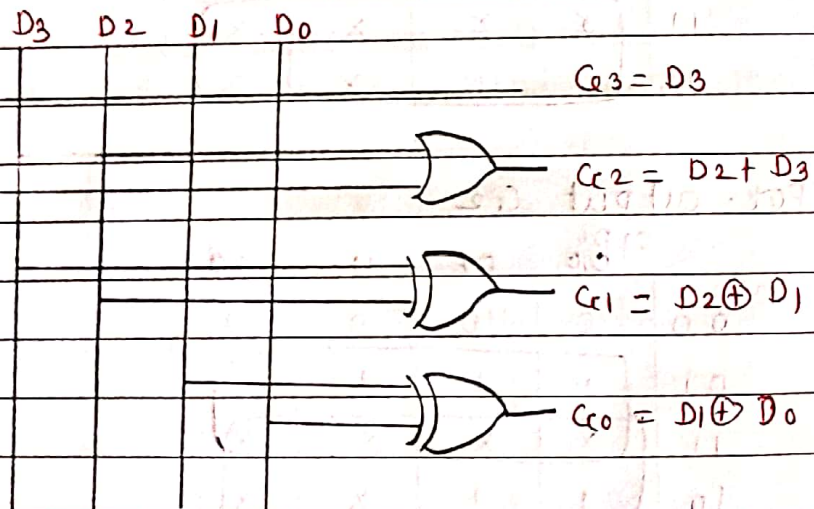
For output C_0

$D_3 D_2$ \ $D_1 D_0$		00		01		11		10	
		00	01	11	10	00	01	11	10
00		0	1	0	1				
01		0	1	0	1				
11		X	X	X	X				
10		0	1	X	X				

$$C_0 = \bar{D}_1 D_0 + D_1 \bar{D}_0$$

$$= D_1 \oplus D_0$$

Step-III Realization :-



o Gray to BCD Code Converter :-

Step-I Truth Table :-

Decimal	Gray Inputs				BCD output			
	G ₃	G ₂	G ₁	G ₀	D ₃	D ₂	D ₁	D ₀
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	1	0	0	1	0
3	0	0	1	0	0	0	1	1
4	0	1	1	0	0	1	0	0
5	0	1	1	1	0	1	0	1
6	0	1	0	1	0	1	1	0
7	0	1	0	0	0	1	1	1
8	1	1	0	0	1	0	0	0
9	1	1	0	1	1	0	0	1

Step-II K map for each output :-

For output D₃

G ₃ G ₂ \ G ₁ G ₀	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	x	x	x	x
10	1	1	x	x

$$D_3 = G_3$$

G ₃ G ₂ \ G ₁ G ₀	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	x	x	x	x
10	0	0	x	x

$$D_2 = G_2$$

For output D_1

$C_3C_2 \backslash C_1C_0$	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	X	X	X	X
10	0	0	X	X

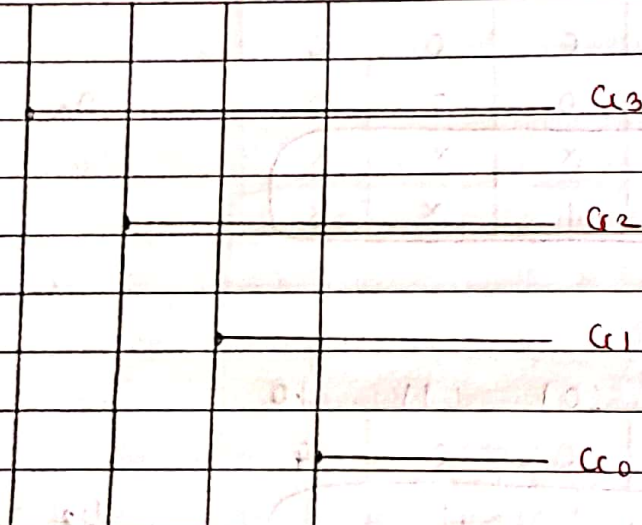
$$D_1 = C_1$$

$C_3C_2 \backslash C_1C_0$	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	X	X	X	X
10	0	1	X	X

$$D_0 = C_0$$

Step - III Realization :-

$D_3 \quad D_2 \quad D_1 \quad D_0$



o Adders and their use as subtractor :-

We can perform both addition & subtraction using only adders because the problem of the subtraction becomes that of an addition when we use 1's and 2's complement representation of negative numbers.

An adder circuit for addition of two n -bit binary number consists of n full adder circuits. It accepts two n -bit binary number as input & produces an $(n+1)$ bit binary no. as the sum.

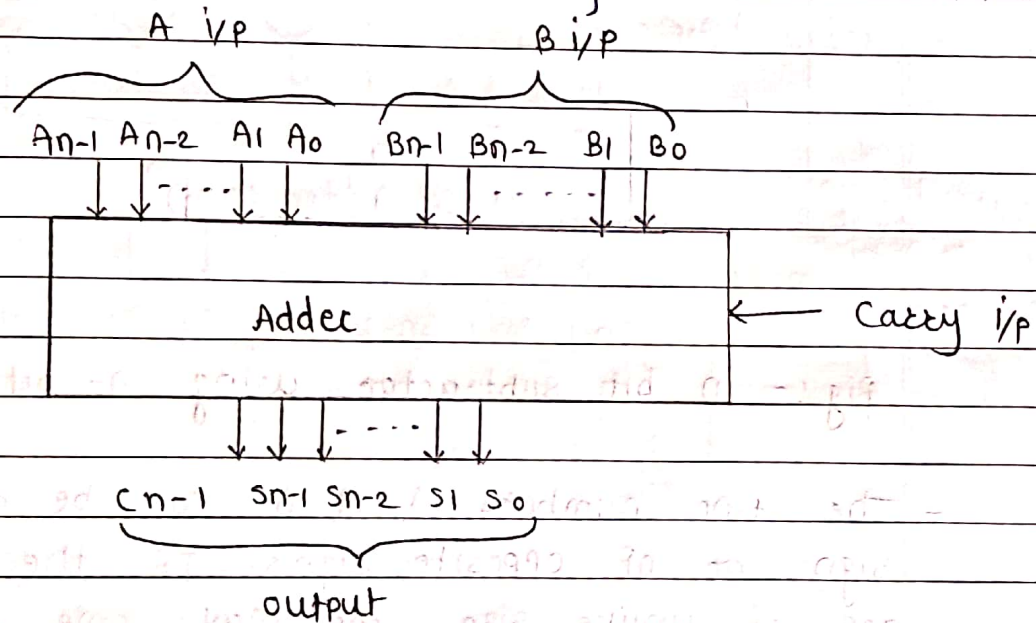
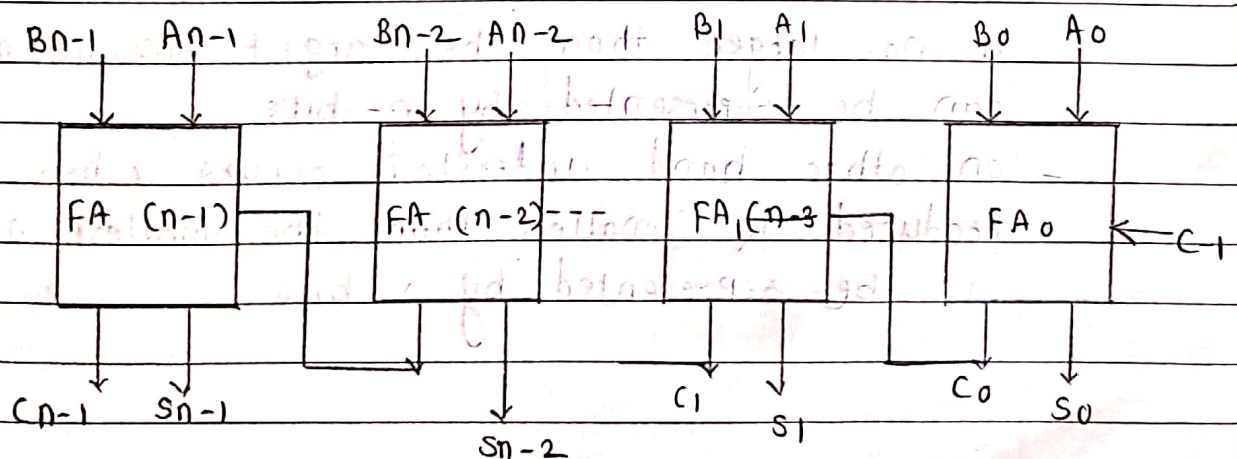


fig :-



Subtraction using adder :-

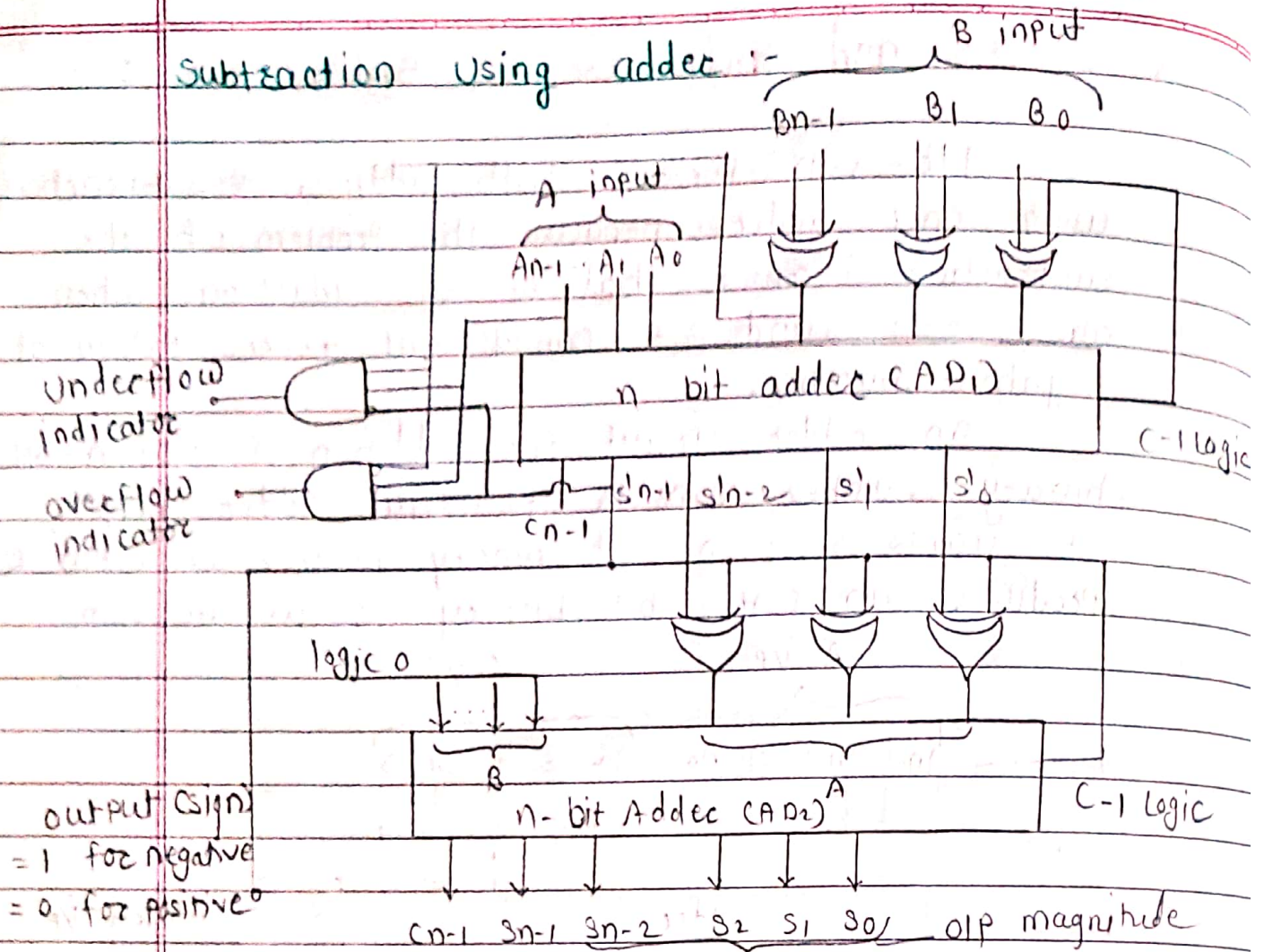


fig:- n bit subtractor using n-bit Adder.

- The two numbers A & B can be of the same sign or of opposite signs. If the two numbers are of unlike sign, we may come across the problem of overflow or underflow.
- overflow occurs when the subtraction operation produces a no. larger than the largest possible no. which can be represented by n-bits.
- on other hand underflow occurs when the result produced is smaller than the smallest no. which can be represented by n-bits.

o Look Ahead Carry :-

- This is the addition technique that eliminates the problem due to interstage carry delay. The look ahead carry addition will therefore speed up the addition process.
- The adder with look ahead carry needs additional hardware but the speed of that adder is independent of the no. of bits.
- Consider the block diagram of full adder & its AND - OR - EXOR realization.

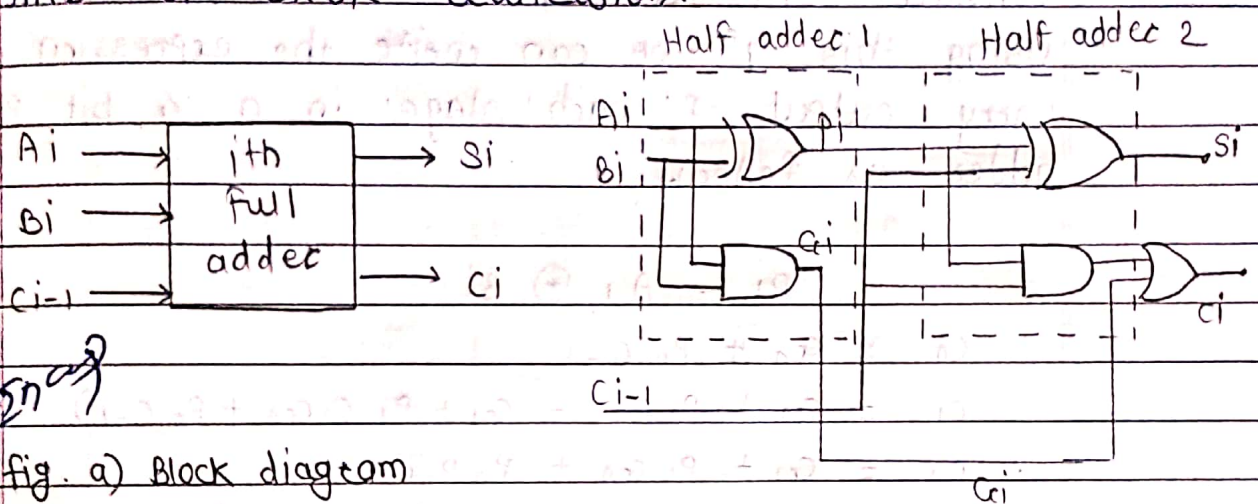


fig. a) Block diagram of i th full adder

fig. b) Realization using AND - OR - EX-OR gates.

Here $P_i = A_i \oplus B_i$ and

$$C_i = A_i B_i$$

$$S_i = P_i \oplus C_{i-1} = A_i \oplus B_i \oplus C_{i-1}$$

$$\text{and } C_i = C_i + P_i C_{i-1}$$

- The output c_i of the first half adder is equal to 1 if $A_i = B_i = 1$ and a carry is generated at the i th stage of the parallel adder.
- This variable c_i is known as carry generate & its value does not depend on the input carry i.e. c_{i-1} .
- The variable p_i is called as carry propagate because this term is associated with the propagation of carry from c_{i-1} to c_i .
- Consider $c_i = c_i + p_i c_{i-1}$ using this eqⁿ we can write the expression for the carry output of each stage in a 4 bit parallel adder as follows.

$$p_i = A_i \oplus B_i$$

$$c_0 = G_0 + P_0 c_{-1}$$

$$c_1 = G_1 + P_1 c_0 = G_1 + P_1 (G_0 + P_0 c_{-1})$$

$$\therefore c_1 = G_1 + P_1 G_0 + P_0 P_1 c_{-1}$$

$$c_2 = G_2 + P_2 c_1$$

$$= G_2 + P_2 (G_1 + P_1 G_0 + P_0 P_1 c_{-1})$$

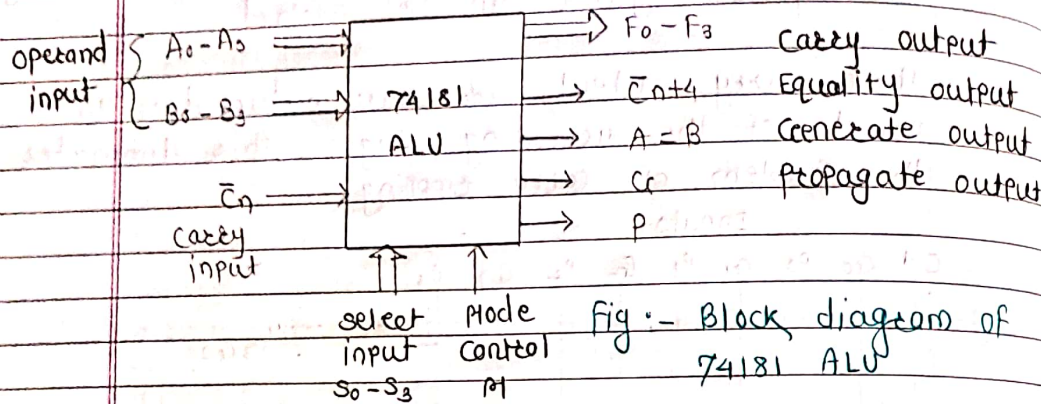
$$\therefore c_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_0 P_1 c_{-1}$$

$$c_3 = G_3 + P_3 c_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_0 P_1 c_{-1}$$

Here G variables are generated from A & B inputs using AND gates and P variables are obtained again directly from A and B inputs using EX-OR gates.

o Arithmetic Logic Unit (ALU) :-

- ALU is a very widely used and popular combinational circuit. It is capable of performing the arithmetic as well as logical operations.
- ALU is the heart of any microprocessor.



B ₀	1	24	VCC
A ₀	2	26	A ₁
S ₃	3	22	B ₁
S ₂	4	21	A ₂
S ₁	5	20	B ₂
S ₀	6	19	A ₃
C _n	7	18	B ₃
M	8	17	G
F ₀	9	16	C _{n+1}
F ₁	10	15	P
F ₂	11	14	A=B
C _{out}	12	13	F ₃

Dual Inline Package

The functions of lines are given

A and B : 4-bit

C_n : carry

F : 4-bit

C_{n+1} : carry

for subtraction

the o/p logic

indicated result

A = B : Logic

C_c : carry

Table :- Functions

Line	Selection				M = Logic Function
	S ₃	S ₂	S ₁	S ₀	
0	0	0	0	0	F = 1
1	0	0	0	1	F = 0
2	0	0	1	0	F = A
3	0	0	1	1	F = B
4	0	1	0	0	F = A + B
5	0	1	0	1	F = A - B
6	0	1	1	0	F = A * B
7	0	1	1	1	F = A / B
8	1	0	0	0	F = A + B + C _n
9	1	0	0	1	F = A - B - C _n
10	1	0	1	0	F = A * B
11	1	0	1	1	F = A / B
12	1	1	0	0	F = A + B + C _n
13	1	1	0	1	F = A - B - C _n
14	1	1	1	0	F = A * B
15	1	1	1	1	F = A / B

The functions of various input, output and control lines are given below

A and B : 4 bit binary data input,

$\bar{C}_n$: Carry input (active low)

F : 4 bit binary data output

$\bar{C}_{n+4}$: Carry output (active low)

For subtraction operation it indicates the sign of the o/p. Logic 0 indicates positive result & Logic 1 indicates result expressed in 2's complement form.

A = B : Logic 1 on this line indicates A = B

cc : Carry generate output.

Table :- Function table of 74181 ALU :-

Line	Selection $S_3 S_2 S_1 S_0$				Active high data		
					M = 1 Logic Function	M = 0 ; Arithmetic operations $\bar{C}_n = 1$ (no carry)	$\bar{C}_n = 0$ (with carry)
0	0	0	0	0	$F = \bar{A}$	$F = A$	$F = A$ plus 1
1	0	0	0	1	$F = \bar{A} + B$	$F = A + B$	$F = (A + B)$ plus 1
2	0	0	1	0	$F = \bar{A} \cdot B$	$F = A + B$	$F = (A + B)$ plus 1
3	0	0	1	1	$F = 0$	$F = \text{Minus 1 (2's comp)}$	$F = \text{Zero}$
4	0	1	0	0	$F = \bar{A} \bar{B}$	$F = A$ plus $A \bar{B}$	$F = A$ plus $A \bar{B}$ plus 1
5	0	1	0	1	$F = \bar{B}$	$F = (A + B)$ plus $A \bar{B}$	$F = (A + B)$ plus $A \bar{B}$ plus 1
6	0	1	1	0	$F = A \oplus B$	$F = A$ minus B minus 1	$F = A$ minus B
7	0	1	1	1	$F = A \bar{B}$	$F = A \bar{B}$ minus 1	$F = A \bar{B}$
8	1	0	0	0	$F = \bar{A} + B$	$F = A$ plus $A \bar{B}$	$F = A$ plus $A \bar{B}$ plus 1
9	1	0	0	1	$F = \bar{A} \oplus B$	$F = A$ plus B	$F = A$ plus B plus 1
10	1	0	1	0	$F = B$	$F = (A + B)$ plus $A \bar{B}$	$F = (A + B)$ plus $A \bar{B}$ plus 1
11	1	0	1	1	$F = A \bar{B}$	$F = A \bar{B}$ minus 1	$F = A \bar{B}$
12	1	1	0	0	$F = 1$	$F = A$ plus A	$F = A$ plus A plus 1
13	1	1	0	1	$F = A + \bar{B}$	$F = (A + B)$ plus A	$F = (A + B)$ plus A plus 1
14	1	1	1	0	$F = A + B$	$F = (A + B)$ plus A	$F = (A + B)$ plus A plus 1
15	1	1	1	1	$F = A$	$F = A$ minus 1	$F = A$

P : carry propagate output

C & P outputs are used when a number of 74181 circuits are used in cascade along with 74182 look ahead carry generator circuit to make the arithmetic operations faster.

select (S_0, S_1, S_2) : Used to select any operations.

Mode control (M) : $M = 0$ Arithmetic operations
 $M = 1$ Logic operations.

• Digital Comparators :-

- Digital comparator is a combinational circuit designed to compare the two n -bit binary applied at its input.

- The comparator has three outputs namely $A > B$, $A = B$ and $A < B$.

- Depending on the result of comparison one of these outputs will go high.

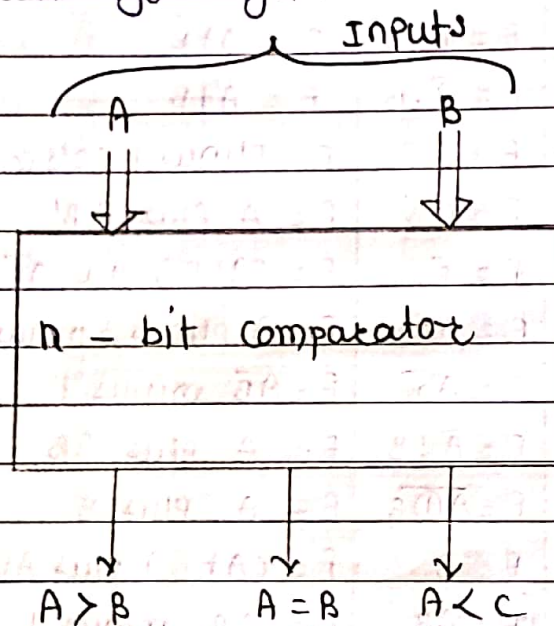


Fig:- Block diagram of n -bit comparator.

fig. shows the block diagram of n-bit comparator

- It receives two n-bit numbers A and B as inputs and the outputs are $A > B$, $A = B$ and $A < B$.
- depending upon the relative magnitude of the two numbers, one of the outputs will be high.

Table :- Truth table of 2 bit - comparator :-

Inputs				Outputs		
A ₁	A ₀	B ₁	B ₀	A > B	A = B	A < B
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	0	1
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	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	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0

A ₁ A ₀ \ B ₁ B ₀	00	01	11	10
00	0	1	1	1
01	0	0	1	1
11	0	0	0	0
10	0	0	1	1

k-map for $A < B$

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

For $A > B$:-

A ₁ A ₀	B ₁ B ₀			
	00	01	11	10
00	0	0	0	0
01	1	0	0	0
11	1	1	0	1
10	1	1	0	0

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

For $A = B$

A ₁ A ₀	B ₁ B ₀			
	00	01	11	10
00	1	0	0	0
01	0	1	0	0
11	0	0	1	0
10	0	0	0	1

$$\begin{aligned} (A=B) &= \bar{A}_1 \bar{A}_0 \bar{B}_1 \bar{B}_0 + \bar{A}_1 A_0 \bar{B}_1 \bar{B}_0 \\ &+ A_1 A_0 B_1 B_0 + A_1 \bar{A}_0 B_1 \bar{B}_0 \\ &= \bar{A}_0 \bar{B}_0 (\bar{A}_1 \bar{B}_1 + A_1 B_1) + \\ &+ (A_1 B_1 + \bar{A}_1 \bar{B}_1) (A_0 B_0 + \bar{A}_0 \bar{B}_0) \\ &= (A_1 \odot B_1) (A_0 \odot B_0) \end{aligned}$$

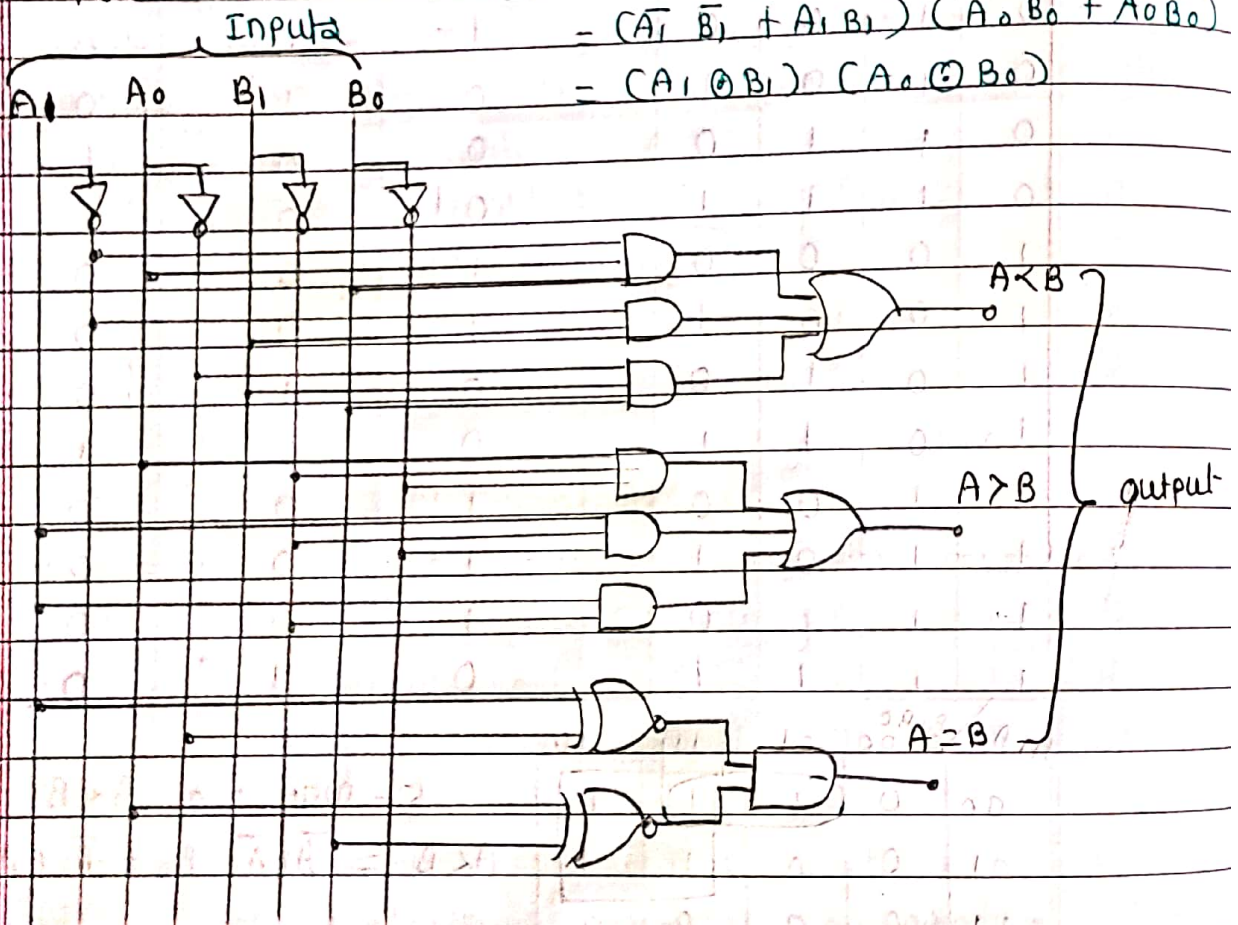


Fig :- Logic diagram for 2 bit Comparator

- 4 bit comparators are available in MS1 (7485) which can compare straight binary & natural BCD codes. These IC's can be cascaded to compare words of greater length without external gates.

- The $A > B$, $A = B$ & $A < B$ outputs of a stage handling less significant bits are connected to the corresponding $A > B$, $A = B$ & $A < B$ cascading i/p of the next stage handling more significant bits.

Table :- Function Table of 7485 4 bit Comparator:-

Comparing Inputs	Cascading inputs			Outputs		
	$A > B$	$A = B$	$A < B$	$A > B$	$A = B$	$A < B$
A, B						
$A > B$	X	X	X	1	0	0
	1	0	0	1	0	0
	X	1	X	0	1	0
$A = B$	0	0	1	0	0	1
	0	0	0	1	0	1
	1	0	1	0	0	0
$A < B$	X	X	X	0	0	1

o Parity generator / checker :-

* Parity generator :-

- Parity generator is a logic circuit which generates the parity bit for even parity or odd parity.
- The digital word is applied at the input and a parity bit for either the even parity or odd parity.

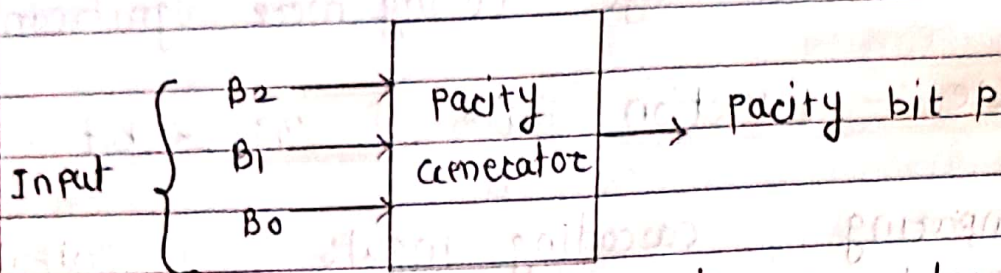


Fig. Block diagram of parity generator.

* Parity checker :-

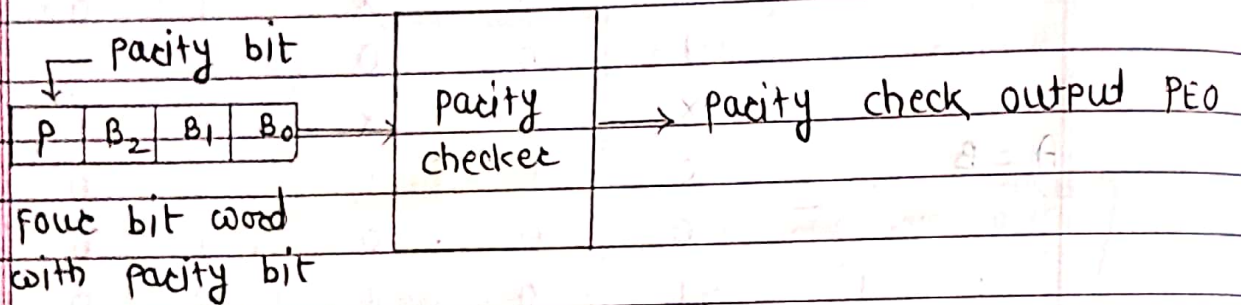


Fig :- Block diagram of parity checker.

- The parity checker circuit is at the receiver. The received word with a parity bit is applied to the parity checker.
- The parity checker will check the parity of the received word and produce its output. Since the parity of transmitted words is "odd" the parity

checker will treat the received words with odd parity as the correct words and those with even parity as the incorrect words.

- The output of parity checker circuit is denoted by parity error output PEO.

$PEO = 1$... If error is present i.e. if the received four bit word has an even parity.

$PEO = 0$... If there is no error i.e. if the received four bit word has an odd parity.

Truth table of parity checker :-

Inputs				output
P	B ₂	B ₁	B ₀	PEO
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

PB ₂	B ₁ B ₀		01	11	10
	00	01			
00	1	0	1	0	
01	0	1	0	1	
11	1	0	1	0	
10	0	1	0	1	

K-map for a Parity checker output.

$$PEO = \bar{P}\bar{B}_2\bar{B}_1\bar{B}_0 + \bar{P}\bar{B}_2B_1B_0 + \bar{P}B_2\bar{B}_1B_0 + \bar{P}B_2B_1\bar{B}_0 + P\bar{B}_2\bar{B}_1\bar{B}_0 + P\bar{B}_2B_1B_0 + P\bar{B}_2\bar{B}_1B_0 + P\bar{B}_2B_1\bar{B}_0$$

$$= \bar{B}_1\bar{B}_0 (\bar{P}\bar{B}_2 + P\bar{B}_2) + B_1B_0 (\bar{P}\bar{B}_2 + P\bar{B}_2) + \bar{B}_1B_0 (\bar{P}B_2 + P\bar{B}_2) + B_1\bar{B}_0 (\bar{P}B_2 + P\bar{B}_2)$$

$$= (\bar{P}\bar{B}_2 + P\bar{B}_2) (\bar{B}_1\bar{B}_0 + B_1B_0) + (\bar{P}B_2 + P\bar{B}_2) (\bar{B}_1B_0 + B_1\bar{B}_0)$$

$$= (P \odot B_2) (B_1 \odot B_0) + (P \oplus B_2) (B_1 \oplus B_0)$$

$$= (\overline{P \oplus B_2}) (\overline{B_1 \oplus B_0}) + P \oplus B_2 + B_1 \oplus B_0$$

Let

$$x = P \oplus B_2 \text{ and } y = (B_1 \oplus B_0)$$

$$\therefore PEO = \bar{x}\bar{y} + xy = x \odot y = \overline{x \oplus y}$$

substituting value we get,

$$PEO = \overline{(P \oplus B_2) \oplus (B_1 \oplus B_0)}$$

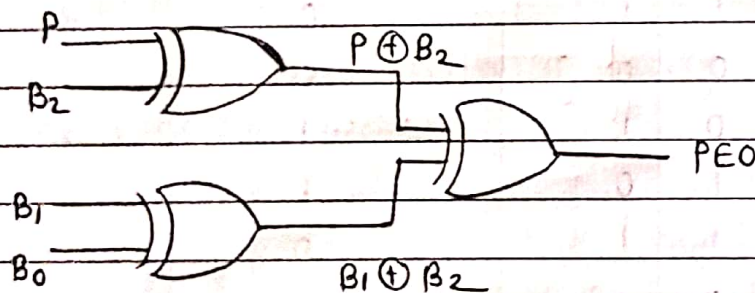


Fig:- Logic diagram.

o Design of Multiplexers & Demultiplexers:-

* Multiplexer :-

- A Multiplexer is a combinational circuit which have many inputs and single output depending upon select lines.
- select lines = n
inputs = 2^n
output = 1
- Many to one circuit.
- It is also called as "Data selector".
- Parallel to serial converter.

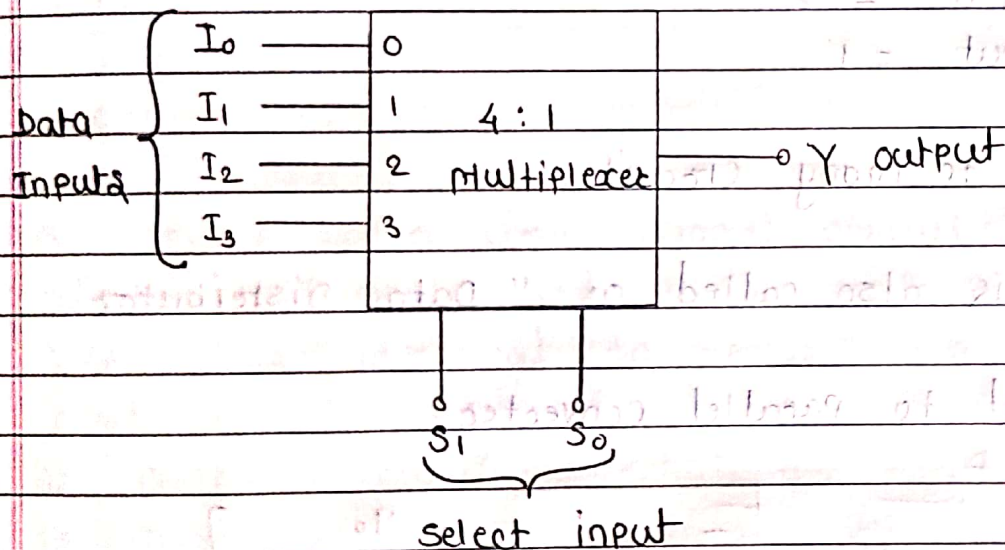
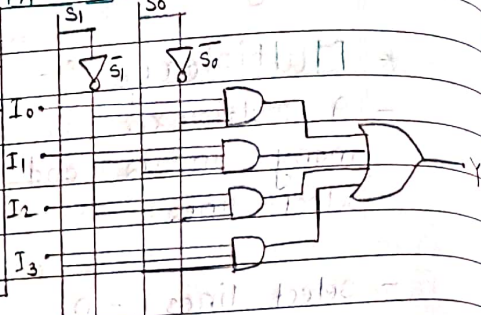


Fig:- Block diagram of 4:1 Multiplexer.

- It has four data input lines ($I_0 - I_3$), two select lines (S_1, S_0) and Y is the data output line. one of the four data inputs will appear at the o/p depending on the values of S_1, S_0 .

Truth table of 4:1 multiplexer :-

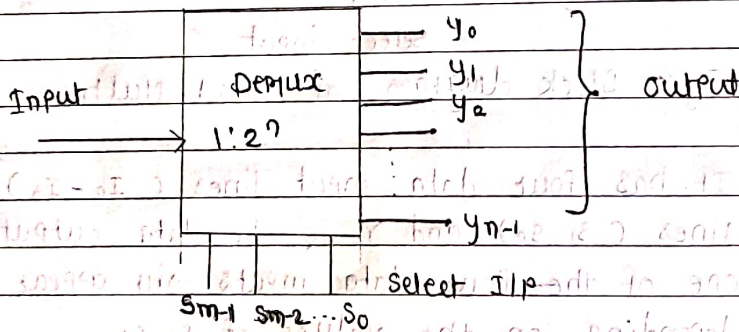
Select	Input	output
S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3



$$Y = \bar{S}_0 \bar{S}_1 I_0 + \bar{S}_0 S_1 I_1 + S_0 \bar{S}_1 I_2 + S_0 S_1 I_3$$

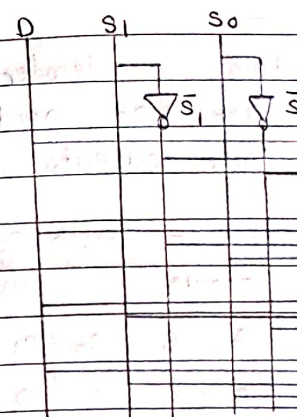
* Demultiplexer :-

- A demultiplexer performs the reverse operation of a multiplexer i.e. it receives one input and distributes it over several outputs.
- select lines = n
output = 2^n
input = 1
- one to many circuit.
- It is also called as "Data distributor".
- serial to parallel converter.



Truth Table of

Data	select line
D	S_1 S_0
0	0 0
0	0 1
0	1 0
0	1 1

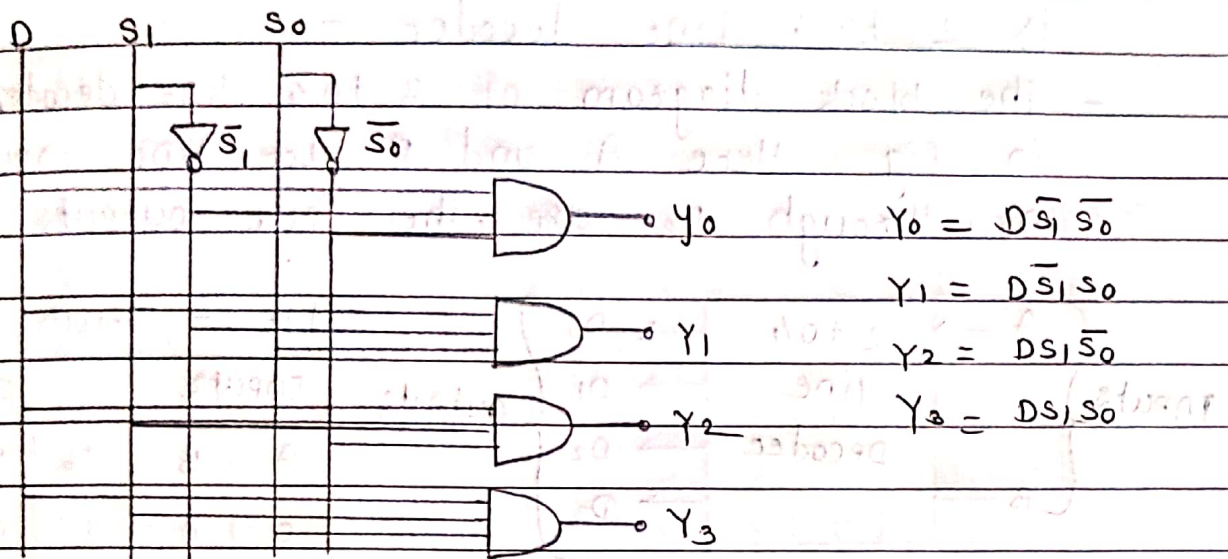


o Decoder :-

- A decoder is a combinational circuit that converts binary information from n inputs to 2^n outputs.
- It has n inputs and 2^n outputs.
- It performs the reverse operation of a multiplexer.
- It is also called as "Data distributor".
- It is a serial to parallel converter.
- Input = n
output = 2^n
 $\therefore$ Input = 6

Truth Table of 1:4 demultiplexer:-

Data	select line		output			
	S_1	S_0	Y_0	Y_1	Y_2	Y_3
D	0	0	D	0	0	0
D	0	1	0	D	0	0
D	1	0	0	0	D	0
D	1	1	0	0	0	D



Decoder :-

- A decoder is a combinational circuit.
- It has n inputs and to a max^m 2^n outputs. Decoder is identical to demux without any data input.
- It performs operations which are exactly opposite to those on an encoder.

- Some to many circuit.

Input = n

output = 2^n

$\therefore$ Input = 6, O/p = $2^6 = 64$.

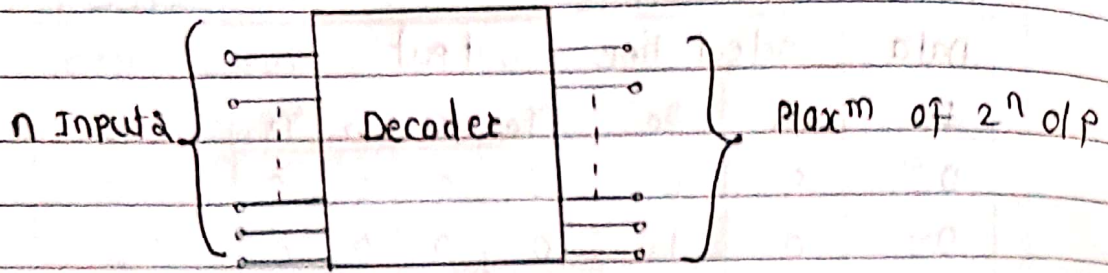


fig :- Block diagram of decoder.

i) 2 to 4 line decoder :-

- The block diagram of 2 to 4 line decoder is shown in fig. Here A and B are two inputs whereas D_0 through D_3 are the four outputs.

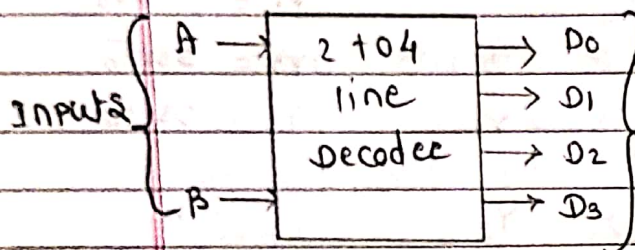


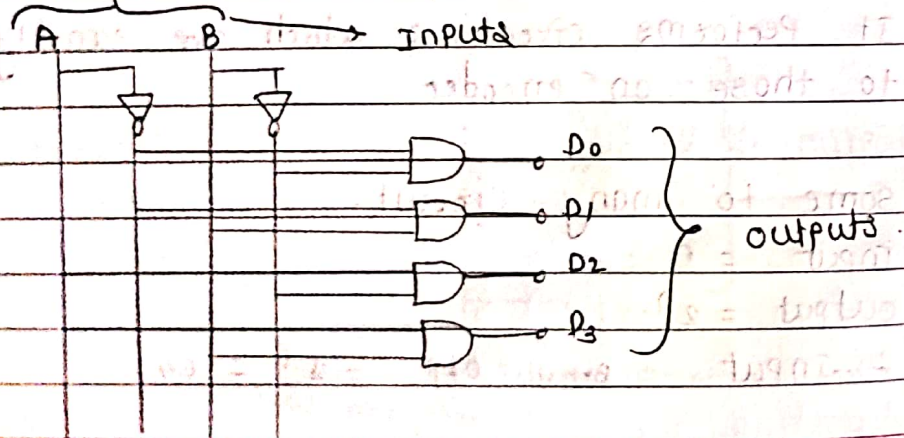
Table :- Truth Table

Inputs		Outputs			
A	B	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

fig :- Block diagram

- Truth table shows that each output is 1 for only a specific combination of inputs. The boolean expressions for the o/p is

$$D_0 = \bar{A} \bar{B}, \quad D_1 = \bar{A} B, \quad D_2 = A \bar{B}, \quad D_3 = A B$$



ii) 3 to 8 Line decoder :-

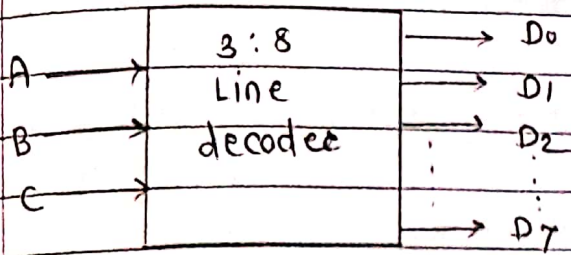


Fig :- Block diagram

Table :- Truth table for 3:8 Line decoder :-

Inputs			Outputs								
A	B	C	D ₀	D ₁	D ₂	D ₃	D ₄	D ₅	D ₆	D ₇	
0	0	0	1	0	0	0	0	0	0	0	$D_0 = \bar{A}\bar{B}\bar{C}$
0	0	1	0	1	0	0	0	0	0	0	$D_1 = \bar{A}\bar{B}C$
0	1	0	0	0	1	0	0	0	0	0	$D_2 = \bar{A}B\bar{C}$
0	1	1	0	0	0	1	0	0	0	0	$D_3 = \bar{A}BC$
1	0	0	0	0	0	0	1	0	0	0	$D_4 = A\bar{B}\bar{C}$
1	0	1	0	0	0	0	0	1	0	0	$D_5 = A\bar{B}C$
1	1	0	0	0	0	0	0	0	1	0	$D_6 = AB\bar{C}$
1	1	1	0	0	0	0	0	0	0	1	$D_7 = ABC$

