

Number System :

In digital electronics, the number system is used for representing the information.

Types of Number Systems :-

1. Decimal Number System

2. Binary Number System

3. Octal Number System

4. Hexadecimal Number System.

1. Decimal Number System

- The number system is having digits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, this number system is known as "Decimal Number System".

Base of decimal number system is 10 (0-9)

2. Binary Number System

- The number system which uses only two digits 0 and 1 is called "Binary Number System".

Base of Binary number system is 2 (0,1).

3. Octal Number System

- The number system which is having digits 0, 1, 2, 3, 4, 5, 6, 7 this number system is known as "Octal Number System".

Base of the Octal number is 8 (0-7).

4. Hexa Decimal Number System

- The number systems which consists the following sixteen numbers of digits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- Base of Hexadecimal n.s is 16 (0-9, A-F)

Table :-

The Table is shown below the Decimal, Binary, Octal and Hexadecimal numbers from 0 to 15 & their equivalent binary numbers.

Decimal	Binary	Octal	Hexadecimal
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Number Conversions

1. Decimal - Binary (2):

Repeatedly divide the number with 2 considering the remainders (Bottom up approach)

$$\text{Ex: } (141)_{10} = (\quad)_2$$

2	141
2	70 - 1
2	35 - 0
2	17 - 1
2	8 - 1
2	4 - 0
2	2 - 0
2	1 - 0

Bottom - Up

0 - 1 - MSB (Most Significant Bit)

$$\therefore (141)_{10} = (10001101)_2$$

2. Decimal to Octal (8)

Repeatedly divide the number with 8, considering the remainders.

$$\text{Ex: } (141)_{10} = (\quad)_8$$

8	141
8	17 - 5
8	2 - 1 0
8	0 - 2

$$\therefore (141)_{10} = (215)_8$$

3. Decimal - Hexadecimal

Repeatedly divide the number with 16, considering the remainders.

Ex:- $(141)_{10} = ()_{16}$

$$\begin{array}{r} 16 \bigg| 141 \\ \hline 8 - 13 - D \end{array}$$

(141) \rightarrow 0 - 8 \rightarrow 1 - 13 \rightarrow 1 - 3 - D

$$\therefore (141)_{10} = (8D)_{16}$$

1. Binary - Decimal

Starting from LSB multiply each positional value with 2^n where n is position, and add all the results.

Ex:- $(1011)_2 = ()_{10}$

$$\begin{array}{r} 1 \ 0 \ 1 \\ 2^3 \ 2^2 \ 2^1 \ 2^0 \end{array}$$

$$(1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0), \therefore 8 + 0 + 2 + 1 = (11)_{10}$$

$$\therefore (1011)_2 = (11)_{10}$$

$$\begin{array}{r} 1 \ 0 \ 0 \ 1 \ 1 \ 0 \ 1 \ 0 \ 1 \ 0 \\ 2^9 \ 2^8 \ 2^7 \ 2^6 \ 2^5 \ 2^4 \ 2^3 \ 2^2 \ 2^1 \ 2^0 \end{array}$$

$$512 + 64 + 32 + 8 + 2 = (618)_{10}$$

$$\therefore (1001101010)_2 = (618)_{10}$$

2. Binary - Octal

Starting from LSB Group 3 bits, convert decimal equivalent for each group of 3 bits.

Ex:- $(10110)_2 = ()_8$

$$\begin{array}{r} 010110 \\ \hline 2^2 2^2 2^2 2^2 2^2 2^2 \\ 110 \quad 010 \quad 1 \end{array} \quad ()_8 = 8^{(EG1)} \times 3 \\ = 26$$

$$\therefore (10110)_2 = (26)_8 = 8^{(EG1)}$$

3. $(10001110)_2 \rightarrow ()_8$ or 10100

$$\begin{array}{r} 01001110 \\ \hline 2^2 2^2 2^2 2^2 2^2 2^2 2^2 \\ 110 \quad 011 \quad 1 \end{array} \quad ()_8 = 8^{(EG1)} \times 3$$

$$= 216 \quad ()_8 = 8^{(EG1)} \times 3 \\ \therefore (10001110)_2 = (216)_8$$

3. Binary - Hexadecimal

Starting from LSB Group 4 bits, convert decimal equivalent for each group of 4 bits

Ex:- $(10101110)_2 = ()_{16}$

$$\begin{array}{r} 0001 \quad 0101 \quad 1110 \\ \hline 2^2 2^2 2^2 2^2 2^2 2^2 2^2 \\ 1 \quad 5 \quad 0 \end{array} \quad ()_{16} = 16^{(EG1)} \times 3$$

$$16^{(EG1)} \times 3 = 16^{(EG1)} \times 1.9375$$

$$\therefore (10101110)_2 = (15D)_{16}$$

$$(10101110) \Leftrightarrow \begin{array}{r} 0 \quad 5 \quad 1 \\ 110 \quad 010 \quad 100 \end{array} = 16^{(EG1)} \times 1.9375$$

$$(10101110) \Leftrightarrow \begin{array}{r} 110 \quad 1010 \quad 0000 \\ 110 \quad 010 \quad 100 \end{array} = 16^{(EG1)} \times 1.9375$$

1. Octal - Binary

Every position must be represented in 3 bit group

$$8^2 \quad 8^1 \quad 8^0 \rightarrow (011\ 011) \quad \text{ex:}$$

$$\text{Ex: } (123)_8 = ()_2$$

$$\begin{array}{r} 1 \\ 001 \\ \hline \end{array} \quad \begin{array}{r} 2 \\ 010 \\ \hline \end{array} \quad \begin{array}{r} 3 \\ 011 \\ \hline \end{array}$$

$$\therefore (123)_8 = (001010011)_2$$

2. Octal to Decimal

From LSB multiply positional value with 8, where n is the position and add all results.

$$\text{Ex: } (123)_8 \rightarrow ()_{10}$$

$$\begin{array}{r} 1 \\ 8^2 \\ 8^1 \\ 8^0 \end{array} \quad \begin{array}{r} 2 \\ 8^2 \\ 8^1 \\ 8^0 \end{array} \quad \begin{array}{r} 3 \\ 8^2 \\ 8^1 \\ 8^0 \end{array} = (8^2 \times 1) + (8^1 \times 2) + (8^0 \times 3)$$

$$= 64 + 16 + 3$$

$$= 83$$

$$\therefore (123)_8 = (83)_{10} \quad \text{ex: } (001010011) \quad \text{ex:}$$

3. Octal - Hexadecimal

Step-1: Convert octal to binary

Step-2: convert binary to hexadecimal

$$\text{Ex: } (123)_8 = (053)_{16}$$

$$\text{Step-1: } \begin{array}{r} 1 \\ 001 \\ \hline \end{array} \quad \begin{array}{r} 2 \\ 010 \\ \hline \end{array} \quad \begin{array}{r} 3 \\ 011 \\ \hline \end{array} \Rightarrow (001\ 010\ 011)_2$$

$$\text{Step-2: } \begin{array}{r} 0000 \\ 0 \\ \hline 0 \end{array} \quad \begin{array}{r} 0101 \\ 5 \\ \hline 5 \end{array} \quad \begin{array}{r} 0011 \\ 3 \\ \hline 3 \end{array} \Rightarrow (053)_{16}$$

II - Method

longest of kernel length.

Step-1: Convert Octal to Decimal

Step-2 Convert a Decimal to Binary

$$\begin{aligned}
 (123)_8 &= \frac{1}{8^0} + \frac{2}{8^1} + \frac{3}{8^2} \quad (8^0 = 1) \\
 &= 64 + 16 + 3 \\
 &= (83)_{10}
 \end{aligned}$$

$$(83)_{10} = ()_{16}^{5+33+323}$$

$$\begin{array}{r}
 16 \overline{)83} \\
 16 \overline{)5-3} \\
 \hline
 0-5
 \end{array}
 = (053)_{16}$$

$$\therefore (123)_8 = (053)_{16} \quad \text{Work of Komalaboykha}$$

1. Hexadecimal - Binary

Every position (value) must be represented in 4 bit group

$$\text{Ex: } (123)_{16} = (\underline{\underline{\underline{1}}})_2$$

$$\frac{1}{1000} \quad \frac{2}{1000} \quad \frac{3}{1000}$$

$$3. (123)_{16} = (000100100011)_2$$

$$\text{Ex-} (A5)_{16} (\text{Ex-16}) = \text{Ex-} (E.61)$$

$$\frac{A}{1010} \quad \frac{5}{0101}$$

$$(A5)_6 = (10100101)_2$$

2. Hexadecimal to Decimal

1019M - 17

Starting from LSB to multiply each positional value with 16^n , where n is the position and add all the results.

$$\text{Ex: } (123)_{16} = ?$$

$$1 \quad 2 \quad 3$$

$$16^2 \quad 16^1 \quad 16^0$$

$$256 + 32 + 3$$

$$= (291)_{10}$$

$$\therefore (123)_{16} = (291)_{10}$$

3. Hexadecimal to Octal

Step-1: Convert Hexadecimal to binary

Step-2: Convert binary to octal

$$\text{Ex: } (123)_{16} = ?$$

$$\begin{array}{r} 1 \\ 0001 \\ \hline 2 \\ 0010 \\ \hline 3 \\ 0011 \end{array} \Rightarrow (0001\ 0010\ 0011)_2$$

$$\begin{array}{r} 0\ 0\ 0 \\ 2^2\ 2^1\ 2^0 \\ \hline 1\ 0\ 0 \\ 2^2\ 2^1\ 2^0 \\ \hline 1\ 0\ 0 \\ 2^2\ 2^1\ 2^0 \\ \hline 0\ 1\ 1 \\ 2^2\ 2^1\ 2^0 \\ \hline 0\ 1\ 0 \end{array} = (0443)_8$$

$$\therefore (123)_{16} = (0443)_8$$

Conversion of Fractional Decimal to Binary

- Divide Integral part and Fractional part.
- Convert Integral part to binary
- Fractional part must be multiplied with 2.
- Divide Integral part & Fractional part
- Multiply fractional part with 2.
- Repeat the same process until fractional part become '0'.

Fractional part \rightarrow '0' \rightarrow Termination.

Fractional part is same with previous values \rightarrow Stop.

Ex: $(34.25)_{10} \xrightarrow{?} (.)_2$ p.0 = $8 \times 2^0.0$

Integral part = $34 \div 2 = 8 \times 2^0.0$

Fractional part = $0.25 \div 2 = 8 \times 2^0.0$

2	34	1	$\leftarrow 8 \div 2 = 8 \times 2^0.0$
2	17-0	0	$34 \Rightarrow 100010 \times 2^0.0$
2	8-1	0	$\leftarrow 8 \div 2 = 8 \times 2^0.0$
2	4-0	1	$\leftarrow 4 \div 2 = 8 \times 2^0.0$
2	2-0	0	$\leftarrow 2 \div 2 = 8 \times 2^0.0$
2	1-0	0	
	0-1		

Integral part Fractional part

$$0.25 \times 2 = 0.5 \rightarrow 010 \quad (0.5 \times 2 = 0.5)$$

$$0.5 \times 2 = 1.0 \rightarrow 1 \quad 0$$

$$0 \times 2 = 0$$

$$\therefore 0.25 \Rightarrow 01$$

$$\therefore (34.25)_2 = 100010.01$$

Ex:- 34.45

Integral part - 34

Fractional part - 0.45

Repeating pattern of 34.45 (0.45)

$$\begin{array}{r} 34 \\ \hline 2 \quad 17 - 0 \\ \hline 2 \quad 8 - 1 \\ \hline 2 \quad 4 - 0 \\ \hline 2 \quad 2 - 0 \\ \hline 2 \quad 1 - 0 \\ \hline 0 \end{array}$$

34 = 100010 (0.45)
0.45 is the repeating part of 34.45

$$\begin{array}{r} 0.45 \times 2 = 0.9 \rightarrow \text{I.P.} \\ 0.9 \times 2 = 1.8 \rightarrow 0 \quad \text{(as F.P.)} \\ 0.8 \times 2 = 1.6 \rightarrow 1 \\ 0.6 \times 2 = 1.2 \rightarrow 1 \\ 0.2 \times 2 = 0.4 \rightarrow 0 \\ 0.4 \times 2 = 0.8 \rightarrow 0 \\ 0.8 \times 2 = 1.6 \rightarrow 1 \end{array}$$

Repetition

$$(0.45) = 0.11100$$

So Stop here.

$$\therefore (34.45) = (100010.011100)_{\text{Ex-264}}$$

$$= 100010.011100$$

$$= 100010.011100$$

100010.011100

100010.011100

Conversion of Fractional Binary to Decimal

Ex:- $(101101.110)_2 = 46.75_{10}$

$$(2^5 \times 1) + (2^4 \times 0) + (2^3 \times 1) + (2^2 \times 1) + (2^1 \times 0) + (2^0 \times 1)$$

$$+ (2^{-1} \times 1) + (2^{-2} \times 1) + (2^{-3} \times 0)$$

$$\text{Digital 8.A = } \begin{array}{c} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{array} \rightarrow \begin{array}{c} 8 \\ A \end{array} = 32 + 8 + 4 + 1 + \frac{1}{2} + \frac{1}{4} + 0$$

$$= 32 + 13 + 0.5 + 0.25$$

$$= 45.75$$

$$\therefore (1011011.110)_2 = (45.75)_{10}$$

Digital 8.A = 45.75		
8	A	
0	1	0
0	0	1
1	1	1

Logic Gates :-

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

Types:-

* Basic Gates

1. AND Gates

2. OR Gate

3. NOT Gate

* Universal Gates:-

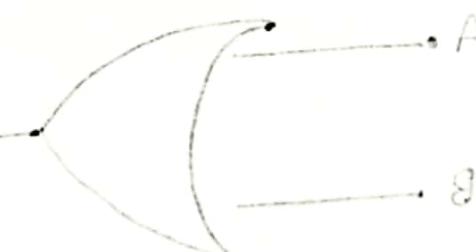
4. NAND Gate

5. NOR Gate

* Special Gates

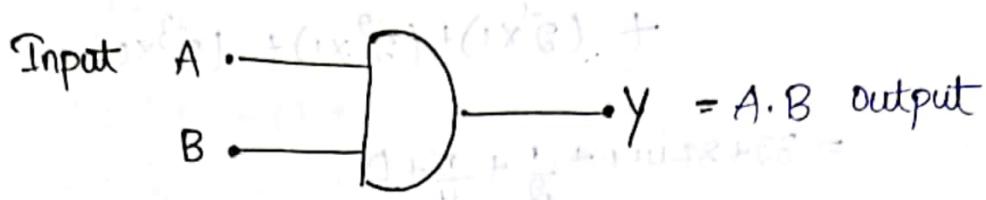
6. EXOR Gate 7. EXNOR Gate

Digital 8.A = 45.75		
8	A	
0	0	0
0	0	1
1	1	0
	0	1
	1	1



i. AND Gate:

- An AND Gate has two or more inputs and only one output which is equal to the product of all inputs.

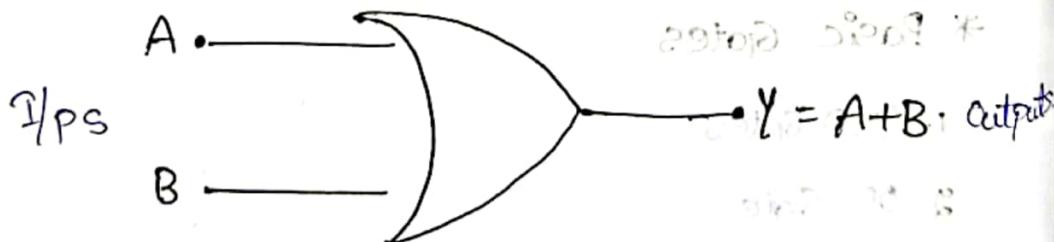


Truth Table:

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

ii. OR Gate:

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

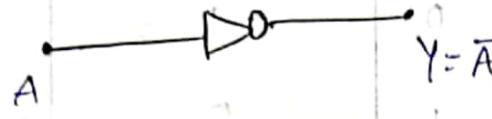


Truth Table:

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

iii. NOT Gate (or) Inverter Gate

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



Truth Table :

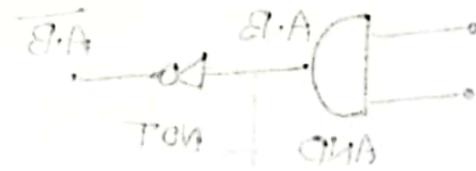
Input (A)	Output (\bar{A})
0	1
1	0

No notand (or) OR gate (OR - NOT).

Universal Gates [NOR & NAND Gates]

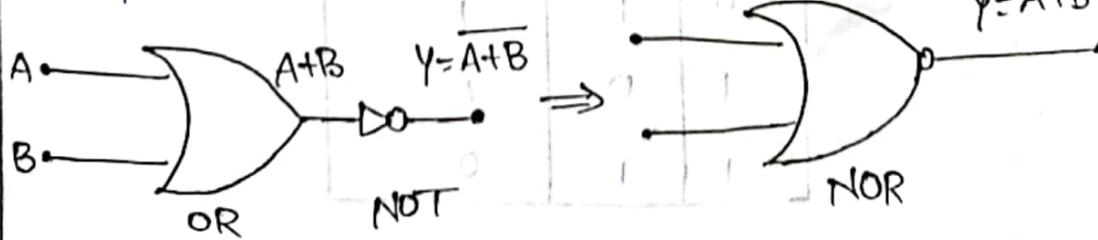
* It is a gate which can be used to implement any other gates without using other types of gates.

Ex:- NOR Gates
NAND Gates



iv. NOR Gate :-

- NOT - OR
- It has two (or) more inputs but only one output.
- The output is the complement of the sum of the inputs.



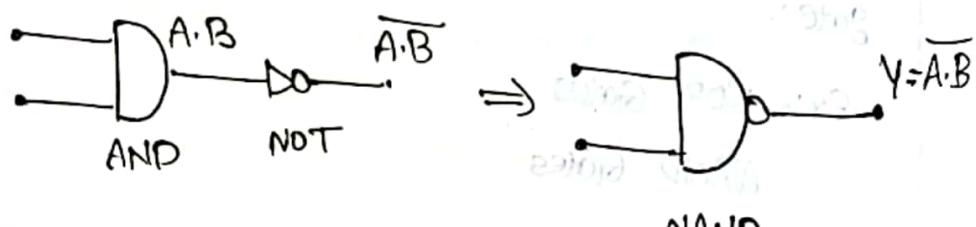
Truth Table :-

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

(A) (A)

V. NAND Gate :-

- (NOT-AND) (NAND Gate is the combination of NOT & AND).
- The output is the complement of the product of inputs.
- It has two or more inputs, but only one output.



Truth Table :-

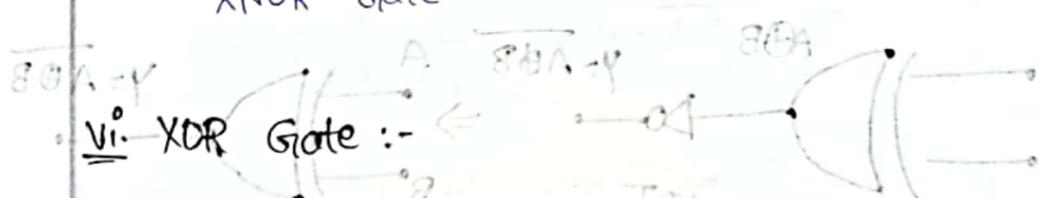
Inputs		Output
A	B	$Y = \overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

Special Gates :-

* These are the exclusive gates used in the particular digital circuits.

Ex:- XOR Gate

XNOR Gate



vi. XOR Gate :-

• Exclusive -OR Gate [Ex-OR]

• It has two or more inputs and only one output.

• Ex-OR operation can be represented with \oplus symbol.

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



XOR GATE

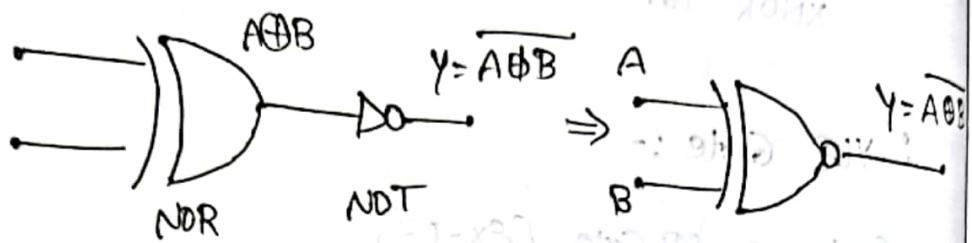
Inputs Output

Truth Table :-

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

vii. XNOR Gate

- Exclusive - NOR Gate [Ex-NOR]
- It has two (or) more inputs and only one output
- The output is the complement of the XOR Gate



- So the operation of the XNOR Gate is given as

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

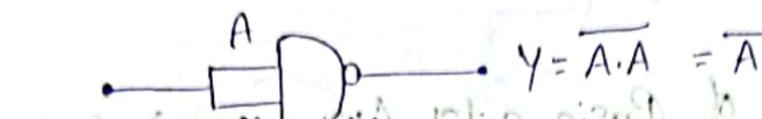
$$Y = A \oplus B$$

Truth Table

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

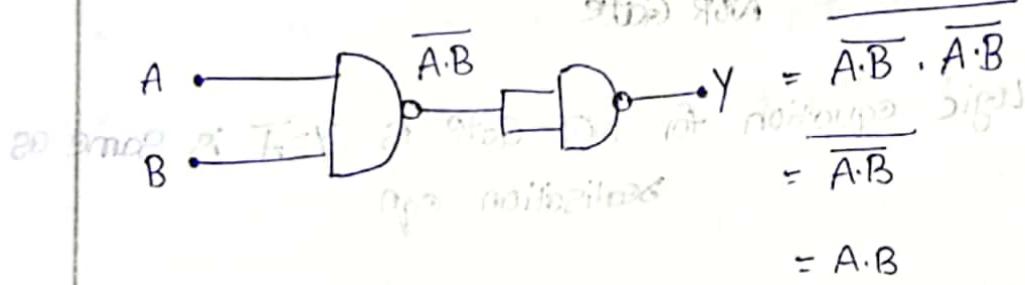
* Realization of Basic Gates (NOT, AND, OR) Using NAND Gate

1. NOT Gate



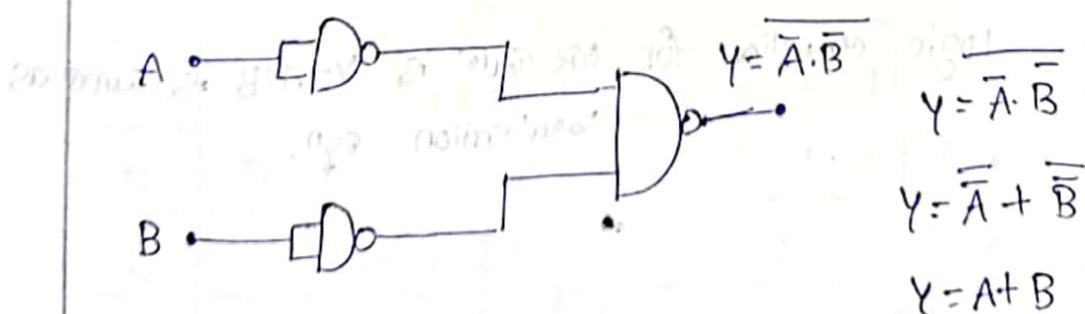
Logic eqn for NOT Gate is $Y = \overline{A}$ is same as realisation eqn.

2. AND Gate



Logic eqn for AND Gate is $Y = A \cdot B$ is same as realisation eqn.

3. OR Gate

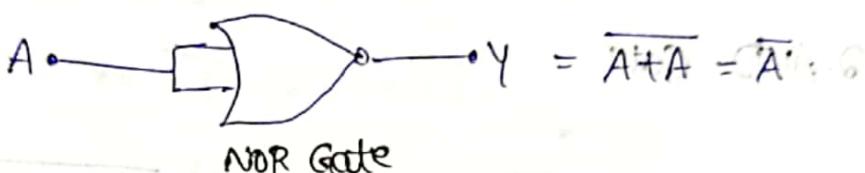


Logic eqn for OR Gate is $Y = A + B$ is same as realisation eqn.

- For designing NOT Gate 1 NAND Gate is required.
- For designing AND Gate 2 NAND Gates are required.
- For designing OR Gate 3 NAND Gates are required.

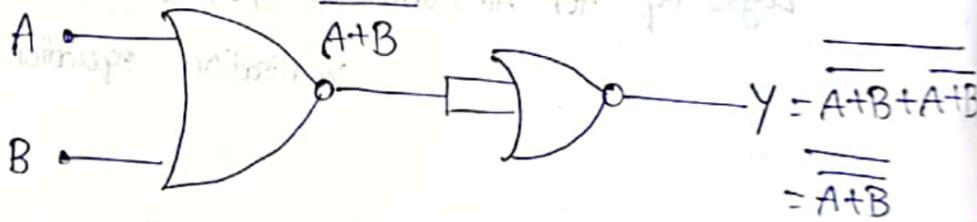
* Realization of Basic gates (NOT, AND, OR) using NOR Gate

1. NOT Gate



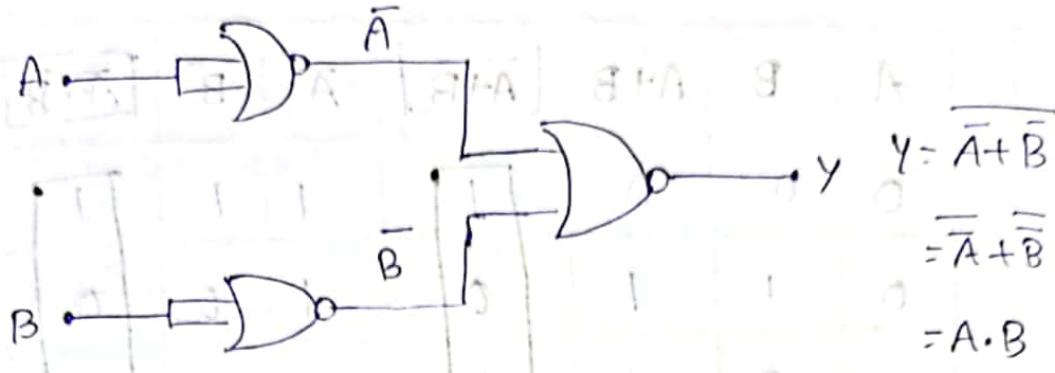
Logic equation for NOT Gate is $Y = \overline{A}$ is same as Realisation eqn.

2. OR Gate



Logic equation for OR Gate is $Y = A+B$ is same as Realisation eqn.

3. AND Gate :-



Logic eqⁿ for AND Gate is $Y = A \cdot B$ is same as realisation eqⁿ.

- For designing the NOT Gate - 1 NOR Gate is required
- " " " OR " 2 NOR " are "
- " " " AND " 3 NOR " " "

Truth Table

De Morgan Theorem

Statement :- If you have two inputs, A & B, Then

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

$$2. \overline{A + B} = \overline{A} \cdot \overline{B}$$

*Proof of $\overline{A \cdot B} = \overline{A} + \overline{B}$

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

According to the Truth Table, we can verify that $\overline{A \cdot B} = \overline{A} + \overline{B}$

* Proof of $\overline{A+B} = \overline{A} \cdot \overline{B}$

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

According to the Truth table we can verify that

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

Boolean Laws

Identities :-

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

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

$$A + 1 = 1$$

$$A + A = A$$

$$A + 0 = A$$

$$A \cdot A = A$$

$$A \cdot 1 = A$$

$$A \cdot 0 = 0$$

$$\overline{\overline{A}} = A$$

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

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

Boolean Equations

Boolean eqn's are of two types

- ## 1. Sum of Product (SOP) expression

- ## 2. Product of sum (POS) expression

$$(6+9+4+7) \cdot (4+3+9) \cdot (7+8+9) \cdot (4+9+7) = 9$$

- ## 1. Sum of Product (SOP)

Ex:-

[Format of SOP]

Each variable = literal (rep'q)

- ## 2. Product of Sum (POS)

Ex:-

$$(A+B) \cdot (B+C) \cdot (A+C) \xrightarrow{\text{Sum Form Produkt}}$$

Format of POS

Canonical & Standard Forms

* Canonical [SOP and POS] Forms

Canonical SOP form

Canonical SOP form is the sum of min-terms in which, each product term contains all literals.

$$\text{Ex: } f = p'q'r + pq'r + pqr' + pqr$$

011 101 110 111
3 5 6 7

$$f = \sum m(3, 5, 6, 7) \quad (\text{sum of minterms})$$

Canonical (POS) Form

canonical POS form is the product of MAX terms, in which each sum term contain all literals.

Ex:-

$$f = (P+q+r) \cdot (P+q+r') \cdot (P+q'+r) \cdot (P+q'+r')$$

$$P+q+r = 0 \quad (q=0) \quad \text{Touch 0} \quad \begin{bmatrix} A=1 \\ A=0 \end{bmatrix}$$

$$0 \quad 0 \quad 0$$

$$P+q+r' = 1 \quad \begin{bmatrix} P=1 \\ q=0 \\ r=0 \end{bmatrix} \quad \text{Touch 1} \quad \begin{bmatrix} A=1 \\ A=0 \end{bmatrix}$$

$$0 \quad 0 \quad 1$$

$$P+q'+r = 2$$

$$0 \quad 1 \quad 0$$

$$P'+q+r = 4 \quad (q=1) \quad \text{Touch 1} \quad \begin{bmatrix} A=1 \\ A=0 \end{bmatrix}$$

$$1 \quad 0 \quad 0$$

$$\bar{f} = \prod M (0, 1, 2, 4) \quad \text{Product of MaxTerms.}$$

Standard SOP & POS Form

• Standard SOP form

Standard SOP form is the standard sum of products form in which each product terms need not contain all literals. So the product terms may or may not be the minterms.

$$Ex = f = pq + qr + pr$$

Q. Convert the following boolean function into standard SOP form

$$f = pq_r + pq'_r + pq_r' + pqr$$

$$= pq_r(r' + r) + pr(q' + q) + qr(p + p')$$

$$\boxed{f = pq_r + pr + qr}$$

• Standard POS form

Standard POS form is the standard product of sums form. In which each sum term need not contain all literals. So the sum terms may or may not be the Max terms.

Ex:- $f = (p+q_r) \cdot (q+r) \cdot (p+r)$

Q. Convert the sum of minterms into standard SOP form

$$f = \sum m(0, 5, 6, 7)$$

$$0 = 000 = \bar{A}\bar{B}\bar{C}$$

$$5 = 101 = A\bar{B}C$$

$$6 = 110 = AB\bar{C}$$

$$7 = 111 = ABC$$

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

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

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

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

Karnaugh Map (K-Map)

- K-map is used for reducing boolean expressions.
- It is a graphical Method.
- It is made up of rectangular boxes.
- Gray codes are used in this K-map.
- There are 2 Variable 3 & 4 variables K-Map.

• Two Variable K-Map

	B	B'	B'
A	0	1	1
A'	0	0	1
A	1	0	1

	B	B'	B'
A	0	0	1
A'	0	1	1
A	1	2	3

$$2^1 \times 2^2 = 4 \text{ cells}$$

$$\begin{aligned} A &\rightarrow 1, A' \rightarrow 0 \quad F = \bar{A}B + A\bar{B} = (\bar{A}B)(A\bar{B}) = 1 - 3 \\ B &\rightarrow 1 \quad B' \rightarrow 0 \end{aligned}$$

$$\text{Ex-1: } F(A, B) = \sum m(0, 3)$$

	B	0	1
A	0	1	1
A'	1	2	3

$$G_{\text{I}} = \bar{A}\bar{B}$$

$$G_{\text{II}} = AB$$

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

Ex-2 :- $F = A'B + A\bar{B} + \bar{A}'B$

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

G.I = $B \quad | \quad (\bar{A} + A) \cdot B$

G.II = $A \quad | \quad (\bar{B} + B) \cdot A$

• Three Variable K-Map

$2^3 = 8$ cells.

A	BC	$B'C'$	$B'C$	BC	$B\bar{C}'$	
	00	01	11	10	0	
A	0	0	1	3	2	

A	$B'C'$	$B'C$	BC	$B\bar{C}'$	
	00	01	11	10	
A	0	0	1	3	

Ex-1 :- $f(A, B, C) = \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}\bar{C} + ABC$

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

$f(A, B, C) = \sum m (1, 3, 4, 6)$

A	BC	00	01	11	10	
		0	1	1	0	
1		0	0	0	1	

G.I - $\bar{A}C$

G.II - $A\bar{C}$

$\therefore f = \bar{A}C + A\bar{C}$

Ex-2: $f = (A+B+C) \cdot (A+B+\bar{C}) \cdot (A+\bar{B}+\bar{C}) \cdot (\bar{A}+B+C)$
 $\cdot (\bar{A}+\bar{B}+C) \cdot (\bar{A}+\bar{B}+\bar{C})$ (POS form)

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

$$= \frac{0 \ 0 \ 0}{0} \cdot \frac{0 \ 0}{1} \cdot \frac{0 \ 1}{3} \cdot \frac{1 \ 0 \ 0}{4} \cdot \frac{1 \ 1 \ 0}{6} (\bar{A}+\bar{B}+\bar{C})$$

$$2^3 = 8 \text{ cells}$$

$$n = \text{no. of variables} = 3$$

$$f = \prod M (0, 1, 3, 4, 6, 7)$$

A		BC		G.I				G.II				G.III			
				00	01	11	10	00	01	11	10	00	01	11	10
0	0	00	01	03	02	10	11	00	01	11	10	00	01	11	10
1	0	00	01	03	02	10	11	00	01	11	10	00	01	11	10

$$G.I = A + \bar{C}$$

$$G.III = B + \bar{D}$$

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

$$G.III = B + C$$

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

• Four Variable K-Map

AB		CD		G.I				G.II				G.III				
				00	01	11	10	00	01	11	10	00	01	11	10	
AB'00	0	1	3	2	00	01	11	10	00	01	11	10	00	01	11	10
A'BD01	4	5	7	6	01	00	01	00	01	00	01	01	00	01	00	01
AB'11	12	13	15	14	11	10	11	10	12	13	15	14	12	13	15	14
AB'10	8	9	11	10	10	11	10	10	08	09	11	10	08	09	11	10

$$\text{Ex:- } f = \overline{A}\overline{B}\overline{C}D + A\overline{B}\overline{C}\overline{D} + \overline{A}\overline{B}C\overline{D} + A\overline{B}C\overline{D} + \overline{A}\overline{B}\overline{C}\overline{D}$$

(SOP form)

$$f = \bar{A}B\bar{C}D + AB\bar{C}D + \bar{A}BCD + ABCD + \bar{A}\bar{B}\bar{C}\bar{D}$$

0101	101	0111	1111	0000
5	13	7	15	0

AB	CD	00	01	11	00	10	01	11
Group-II	00	1	0	1	0	1	3	2
	01	u	1	5	1	7	11	6
	11	12	11	10	11	15	13	14
	10	7	8	9	11	3	0	0

$$G \cdot \mathbb{1} = BD$$

51-A 4 20

$$G \cdot \mathfrak{D} = A' B' \cdot C' D'$$

$$f = A'B'C'D' + BD$$

$$\text{Ex:- } f(A, B, C, D) = \sum (0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14) \text{ SOP}$$

AB	CD	00	01	11	10	
00	00	1	0	1	1	
01	01	1	1	0	0	
11	11	1	1	0	0	
10	10	1	0	1	0	
		8	9	13	11	10
		1	1	1	1	1
		2	2	6	6	4
		3	3	7	7	5
		4	4	12	12	14
		5	5	15	15	16
		6	6	14	14	13
		7	7	13	13	12
		8	8	12	12	11
		9	9	11	11	10
		10	10	10	10	9
		11	11	11	11	8
		12	12	12	12	7
		13	13	13	13	6
		14	14	14	14	5
		15	15	15	15	4
		16	16	16	16	3
						2
						1
						0

Ex-3: $f(A, B, C, D) = \prod (3, 4, 6, 7, 11, 18, 13, 14, 15)$

POS.

AB		CD			
		00	01	11	10
00	0	1	0	2	
01	0	4	5	0	6
11	0	2	0	0	14
10	8	9	0	11	10

G-1

G-2

G-1

$$G-1 = \overline{C} + \overline{D}$$

$$G-2 = \overline{A} + \overline{B}$$

$$G-3 = \overline{B} + D$$

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

*5 Variable K-Map

$A = 0$ $A = 1$ $2^4 = 16$ squares.

BC		DE			
		00	01	11	10
00	0	1	3	2	0
01	4	5	7	6	16
11	12	13	15	14	17
10	8	9	11	10	19
01					18
11					20
10					21
01					23
11					22
10					28
01					29
11					31
10					30
01					25
11					24
10					27
01					26

Minterms 0 through 15 belongs with $A=0$

& 16 through 31 belongs with $A=1$

Ex-1: $f(a,b,c,d,e) = \sum m (0,1,3,4,5,6,10,13,14,18,21,23,24,26,29,30)$

		A						A					
		00	01	11	10			00	01	11	10		
bc		00	10	11	3	12	bc		00	16	17	19	18
G-2		01	14	15	7	16	G-3		01	20	21	23	22
11		12	13	15	14	10	10		11	28	29	31	30
10		8	9	11	10		G-4		10	24	25	27	26

$$G^{-1} = DE$$

$$a + \bar{b} = \delta + \theta$$

$$G-2 = \overline{A} \overline{B} \overline{D}$$

$$G-3 = \overline{CD}e$$

$$G - 4 = A \bar{B} \bar{C} \bar{E}$$

$$(\bar{\sigma}^{\pm 1}) \in \mathbb{P}^1$$

$$\therefore f = D\bar{E} + \bar{A}B\bar{D} + C\bar{D}\bar{E} + AB\bar{C}\bar{E}$$

$$\text{Ex:- } f(a,b,c,d,e) = \sum m(0,4,6,8,12,13,14,15,16,17,18,21,24,25,26,28,29,31) \quad (9)$$

		A				A'					
BC		00	01	11	10	BC		00	01	11	10
00	00	10	18	18	3	3	14	00	10	18	18
01	01	14	18	18	3	3	16	01	10	18	18
11	11	1	13	1	1	1	1	01	1	1	1
10	10	1	8	9	11	18	10	1	20	21	23
		6-4						1	28	29	31
		G-3						1	24	25	27
								G-6			
									G-1		

Ex: $f = \sum m(0, 1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 20, 21, 23, 24, 25, 26, 27, 29, 31)$

$$G-2 : A\bar{C}\bar{E}$$

$$G-3 : \bar{A}\bar{D}\bar{E}$$

$$G-4 : BCE$$

$$G-5 : \bar{A}C\bar{E}$$

$$G-6 : B\bar{D}\bar{E}$$

$$f = A\bar{D}E + A\bar{C}\bar{E} + \bar{A}\bar{D}\bar{E} + BCE + \bar{A}C\bar{E} + B\bar{D}\bar{E}$$

Ex: $f(a, b, c, d, e) = \sum m(0, 5, 6, 8, 9, 10, 11, 16, 20, 24, 25, 26, 27, 29, 31) \{ \text{POS} \}$

		A				B				C				D				E			
		00	01	11	10	00	01	11	10	00	01	11	10	00	01	11	10				
		DE	DE	DE	DE	BC															
00	00	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15				
01	00	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15				
11	00	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15				
10	00	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15				

Labels on the Karnaugh map:

- $G-5$ is circled in the $A=0, B=0, C=1$ cell.
- $G-6$ is circled in the $A=0, B=1, C=0$ cell.
- $G-4$ is circled in the $A=1, B=0, C=0$ cell.
- $G-3$ is circled in the $A=1, B=0, C=1$ cell.
- $G-2$ is circled in the $A=1, B=1, C=0$ cell.
- $G-1$ is circled in the $A=1, B=1, C=1$ cell.

$$G-1 = \bar{B} + C$$

$$G-2 = \bar{A} + \bar{B} + \bar{E}$$

$$G-3 = C + D + \bar{E}$$

$$G-4 = \bar{A} + B + D + E$$

$$G-5 : A + B + \bar{C} + D + \bar{E}$$

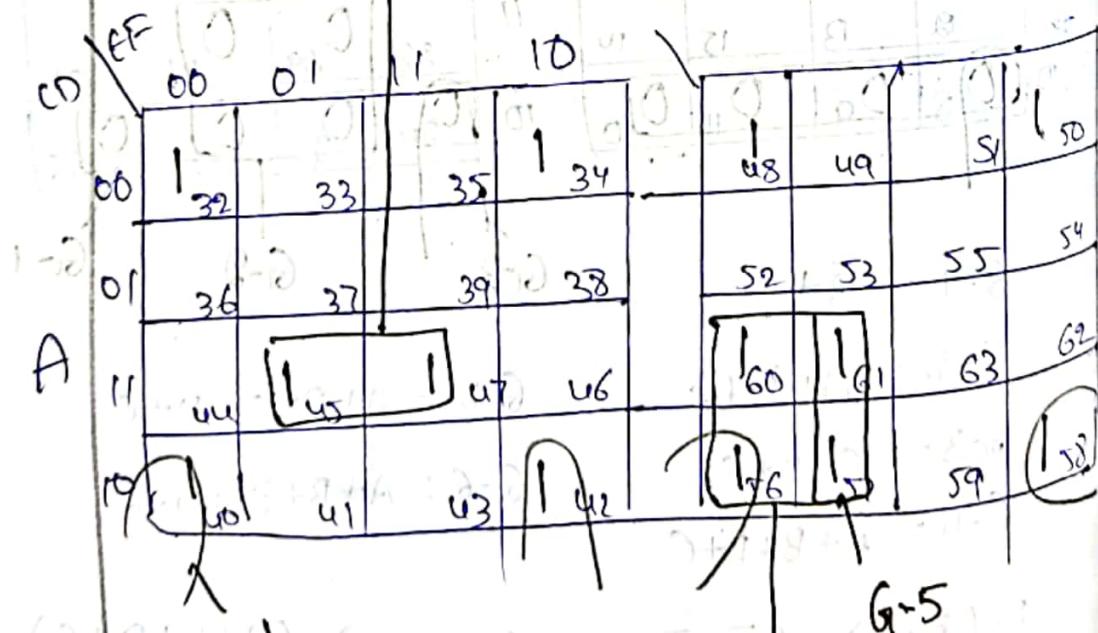
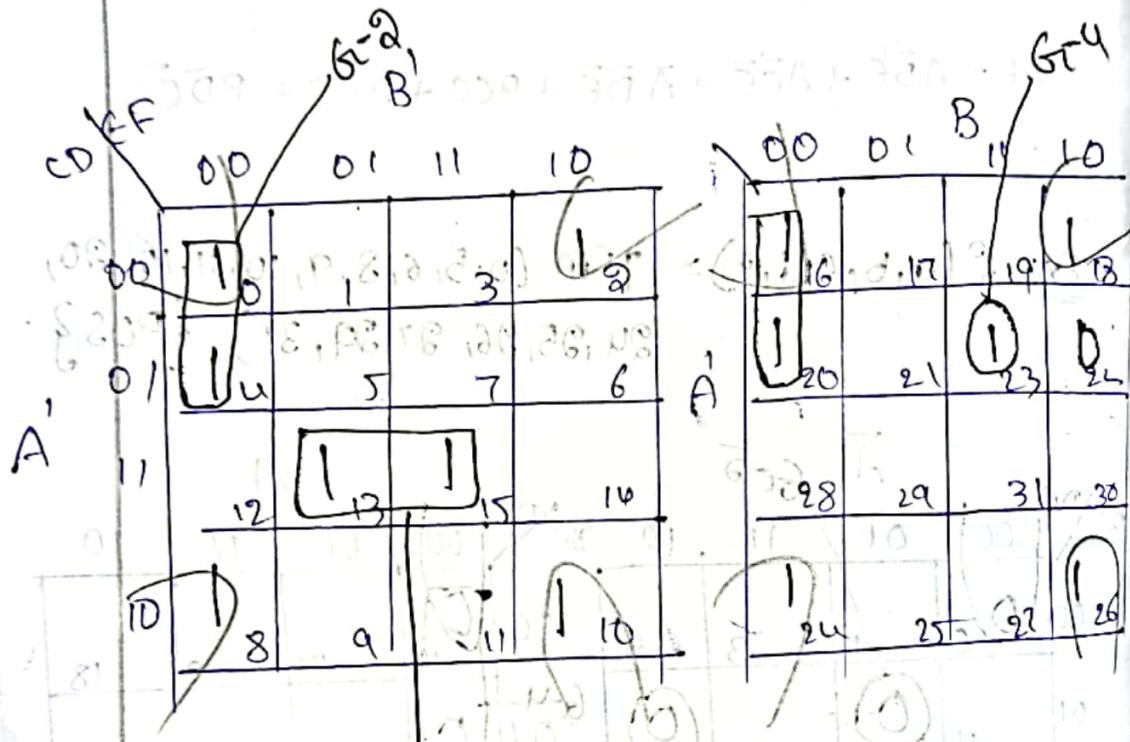
$$G-6 : A + B + \bar{C} + \bar{D} + \bar{E}$$

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

* Six - Variable K-Map (A,B,C,D,E,F).

$$2^6 = 64 \text{ squares}$$

$$f(A,B,C,D,E,F) = \sum (0, 2, 4, 8, 10, 13, 15, 16, 18, 20, 23, 24, 26, 30, 34, 40, 41, 42, 45, 47, 48, 50, 56, 57, 58, 60, 61).$$



$$G-1 = \bar{D}\bar{F}$$

$$G-4 = \bar{A}\bar{B}\bar{C}\bar{D}\bar{E}\bar{F}$$

$$G-2 = \bar{A}\bar{C}\bar{E}\bar{F}$$

$$G-5 = A\bar{C}\bar{E}F$$

$$G-3 = \bar{B}CD\bar{F}$$

$$G-6 = ABC\bar{E}$$

$$\therefore F = \bar{D}\bar{F} + \bar{A}\bar{C}\bar{E}\bar{F} + \bar{B}CD\bar{F} + \bar{A}\bar{B}\bar{E}DEF + A\bar{C}\bar{E}F + ABC\bar{E}$$

* Don't Care Condition in K-Map

- Don't care conditions are denoted by (X) and may be assumed '0' or '1' based on requirement for simplification.

$$\text{Ex:- } Y = \sum m(1, 3, 7, 11, 15) + d(0, 2, 10)$$

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

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

$$\text{Ex:- } Y = \sum m(0, 1, 5, 9, 13, 14, 15) + d(3, 4, 7, 10, 11).$$

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

$$G-1 = D$$

$$G-2 = \bar{A}\bar{C}$$

$$G-3 = AC$$

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

Code Connections

- The Code Converters is useful to convert one type of binary code to another.
 - The Code Conversion is useful to provide security and to reduce the power dissipation in VLSI circuits.
 - There are different types of Binary codes like BCD code, gray code, excess 3 code.
 - BCD - Binary Coded Decimal.

BCD to Excess-3, code Converter

- For this conversion process, 4-bit BCD Code, is considered as input, which is converted into 4-bit excess 3-Code.
 - Since 4 bit is considered for BCD Code, the output is produced only for the inputs from 0 to 9, for the remaining input combinations, the output cannot be predicted. So they are don't care outputs.

(1,0,0,0,0) to (21,0,0,0,0) MS = 4 283

	01	11	15	03	9/30
100 - 100	X	X	X	X	100
100 - 100					100
100 - 100	X		X	X	100
100 - 100					100
100 - 100	X	X	X	X	100
100 - 100					100
100 - 100	X	X	X	X	100
100 - 100					100

Truth Table :-

Decimal Number	BCD Code Input				Excess 3 - 9				
	b_3	b_2	b_1	b_0	c_3	c_2	c_1	c_0	
0	0	0	0	0	0	0	1	1	3
1	0	0	0	1	0	1	0	0	4
2	0	0	1	0	0	1	0	1	5
3	0	0	1	1	0	1	1	0	6
4	0	1	0	0	0	1	1	1	7
5	0	1	1	0	1	0	0	0	8
6	0	1	1	0	1	0	0	1	9
7	0	1	1	1	1	0	0	0	10
8	1	0	0	0	1	0	1	1	11
9	1	0	0	1	1	1	0	0	12

From the Truth Table, the Minterms are obtained for each outputs (c_3, c_2, c_1, c_0)

$$G_0 = \sum m(0, 2, 4, 6, 8) + D(10, 11, 12, 13, 14, 15).$$

For c_0 output:-

b_3	b_2	b_1	b_0	c_3	c_2	c_1	c_0
00	10	1	3	12	00	1	1
01	1	X	X	1X	10	1	1
11	X	X	X	11	X	X	X
10	1	9	X	10	1	X	X

$c_0 = \overline{b_0}$

$G_1 =$

For c_1 Output :-

$$C_1 = \sum m(0, 3, 4, 7, 8) + D(10, 11, 12, 13, 14, 15)$$

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

$$G-1 \Rightarrow \overline{b_1} \overline{b_0}$$

$$G-2 \Rightarrow b_1 b_0$$

$$C_1 = \overline{b_1} \overline{b_0} + b_1 b_0 \quad (\text{or})$$

$$C_1 = b_0 \oplus b_1$$

For c_2 Output

$$C_2 = \sum m(1, 2, 3, 4, 9) + D(10, 11, 12, 13, 14, 15)$$

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

$$G-1 = \bar{b}_2 b_0$$

$$G-2 = \bar{b}_2 b_1$$

$$G-3 = \bar{b}_2 \bar{b}_1 \bar{b}_0$$

$$C_2 = \bar{b}_2 b_0 + \bar{b}_2 b_1 + b_2 \bar{b}_1 \bar{b}_0$$

For C_3 Output

$$C_3 = \text{Im } (5, 6, 7, 8, 9) + D (10, 11, 12, 13, 14, 15)$$

$b_3 b_2$	$b_1 b_0$	00	01	11	10
00		0	1	3	2
01		1	1	1	1
11	X	X ₁₃	X ₁₅	X ₁₄	
10	1 ₈	1 ₉	X ₁₁	X ₁₀	

$b_3 b_2$	$b_1 b_0$	00	01	11	10
00					
01		1	1	1	
11	X	X	X	X	
10	1	1	X	X	

$G-2$

$G-1$

$G-3$

$$G-1 = b_3$$

$$G-2 = b_2 b_1$$

$$G-3 = b_2 b_0$$

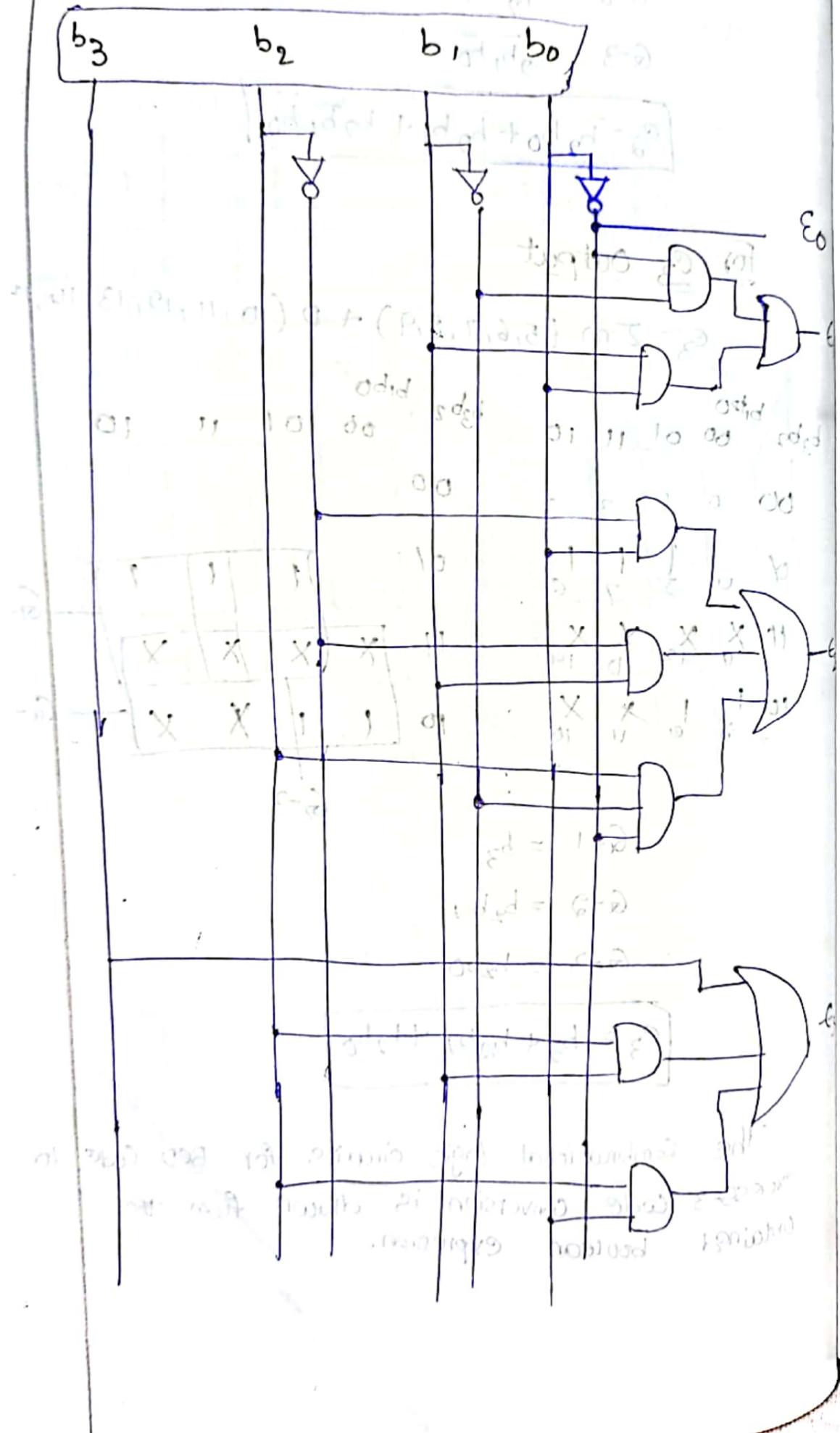
$$C_3 = b_3 + b_2 b_1 + b_2 b_0$$

The Combinational logic circuits for BCD Code to Excess₃ code conversion is drawn from the obtained boolean expression.

BCD Code (Input).

$$10 \text{ g} = 6.3$$

321



24/12/2021

Binary to Gray Code

MSB LSB MSB LSB
 $b_3 b_2 b_1 b_0$ \rightarrow $g_3 g_2 g_1 g_0$

$$g_3 = b_3$$

$$g_2 = b_2 \oplus b_3$$

$$g_1 = b_1 \oplus b_2$$

$$g_0 = b_0 \oplus b_2$$

	0 0 0 0	0 0 0 0
1 (0 0 0 1	0 0 0 1
	0 0 1 0	0 0 1 1
	0 0 1 1	0 0 1 0
2 (0 1 0 0	0 1 1 0
	0 1 0 1	0 1 1 1
	0 1 1 0	0 0 1 0
	0 1 1 1	0 1 0 0

$b_3 \ b_2 \ b_1 \ b_0$

0 0 0 1

0 0 1 0

1 1 1 0

} 2 Transition in binary

$g_3 \ g_2 \ g_1 \ g_0$

0 0 0 1

0 0 1 1

1 1 1 0

} 1 Transition.

binary

0 0 1 1

0 1 0 0

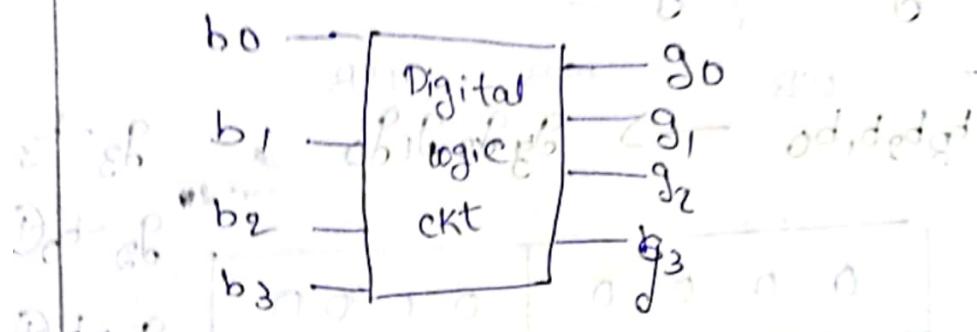
Gray Code.

0 0 1 0

0 1 1 0

19/2021

class prob of parity



$b_7 = 0$ 1 0 0 0 1 0 0 0

$b_7 = 1$ 1 0 0 0 1 1 0 0

Parity bit generator

	b_3	b_2	b_1	b_0	P_0	1	0	0	1
0	0	0	0	0	0	1	0	1	0
1	0	0	1	0	1	0	1	0	1
2	0	1	0	0	1	1	1	0	1
3	0	0	1	0	1	1	1	1	0
4	0	1	0	0	1				
5	0	1	0	1	0	1	0	1	0
6	0	1	1	0	0	0	0	0	0
7	0	1	1	1	1				
					00 10 00 00	00 10 00 00			

minimum 1 { 1 0 0 0

odd prob { 1 1 0 0

1 { 0 1 0 0

8 { 1 1 0 0

b_3, b_2, b_1, b_0 P_0

8	0 0 0	1
9	1 0 0 1	0
10	1 0 1 0	0

11	1 0 1 1	1
----	---------	---

$$(d^0 \oplus d^1 \oplus d^2 \oplus d^3) (d^0 \oplus d^1 \oplus d^2) = 8$$

13	1 0 1 1	1
----	---------	---

14	1 1 1 0	1
15	1 1 1 1	0

$$(d^0 \oplus d^1 \oplus d^2 \oplus d^3) (d^0 \oplus d^1 \oplus d^2) = 8$$

$b_3 b_2$	$b_1 b_0$	P_0
00	00	00
01	01	01
11	11	11
10	10	10

$b_3 b_2$	$b_1 b_0$	P_0
00	00	1
01	01	1
11	11	1
10	10	1

$$P_0 = \bar{b}_3 \bar{b}_2 \bar{b}_1 b_0 + \bar{b}_3 \bar{b}_2 b_1 \bar{b}_0 + \bar{b}_3 b_2 \bar{b}_1 \bar{b}_0 + \bar{b}_3 b_2 b_1 b_0$$

$$+ b_3 \bar{b}_2 \bar{b}_1 b_0 + b_3 b_2 \bar{b}_1 \bar{b}_0 + b_3 \bar{b}_2 \bar{b}_1 \bar{b}_0 + b_3 \bar{b}_2 b_1 \bar{b}_0$$

$$= \bar{b}_3 \bar{b}_2 (b_1 b_0 + \bar{b}_1 \bar{b}_0) + \bar{b}_3 b_2 (b_1 \bar{b}_0 + \bar{b}_1 \bar{b}_0)$$

$$+ b_3 b_2 (b_1 b_0 + \bar{b}_1 \bar{b}_0) + b_3 \bar{b}_2 (\bar{b}_1 b_0 + \bar{b}_1 \bar{b}_0)$$

$$P_0 = \overline{b_3} \overline{b_2} (b_0 \oplus b_1) + \overline{b_3} b_2 (\overline{b_0} \oplus \overline{b_1}) \\ + b_3 \overline{b_2} (b_0 \oplus b_1) + b_3 \overline{b_2} (\overline{b_0} \oplus \overline{b_1})$$

$$P_0 = b_0 \oplus b_1 \left(\bar{b}_3 \bar{b}_2 + b_3 b_2 \right) + \overline{b_0 \oplus b_1} \left(\bar{b}_3 b_2 + b_3 \bar{b}_2 \right)$$

XNOR

XOR

$$P_0 = (b_0 \oplus b_1) (\overline{b_2 \oplus b_3}) + \overline{b_0 \oplus b_1} (b_3 \oplus b_2)$$

x \overline{y} \circ \overline{x} \circ y

$$P_0 = (b_0 \oplus b_1) \oplus (b_2 \oplus b_3)$$

$$P_0 = b_0 \oplus b_1 \oplus b_2 \oplus b_3$$

Decimal to 7-Segment display Code

<u>b_3</u>	<u>b_2</u>	<u>b_1</u>	<u>b_0</u>	<u>a</u>	<u>\bar{b}</u>	<u>c</u>	<u>d</u>	<u>e</u>	<u>f</u>	<u>g</u>	<u>$10a$</u>
0	0	0	0	1	1	1	1	1	1	0	<u>$f \mid g \mid b$</u>
1	0	0	0	1	0	1	1	0	0	0	<u>$c \mid \mid l c$</u>
2	0	0	1	0	1	1	0	1	1	0	<u>$\mid d$</u>
3	0	0	1	1	1	1	1	1	0	1	
4	0	1	1	0	0	1	1	0	0	1	<u>$\mid d = 0$</u>
5	0	1	0	1	1	0	1	1	0	1	
6	0	1	1	0	1	0	1	1	1	1	
7	0	1	1	0	1	0	1	0	0	0	
8	0	0	0	1	1	1	1	1	1	1	
9	0	0	0	1	1	1	1	1	0	1	

$b_3 b_2$	00	01	11	10	
00	1		1	1	$G-I \quad 'a'$
01		1	1	1	$G-IV$
11	X	X	X	X	
10	1	1	X	X	$G-III$
	X	X	X	X	$G-II$
	X	X	X	X	SOP Form.

$$G-II := b_3$$

$$G-III := \overline{b_2} \cdot \overline{b_0}$$

$$G-IV := b_2 \cdot b_0$$

$$a = b_1 + b_3 + \overline{b_2} \cdot \overline{b_0} + b_2 \cdot b_0$$

$$a = b_1 + b_3 + b_0 \oplus b_2$$

$b_3 b_2$	00	01	11	10		
00	1	1	1	1	I	odd
01	1	1	1	1	II	$odd + odd + even$
11	X	X	X	X		
10	1	1	X	X	III	$even + even + even$
	X	X	X	X	IV	$even + even + even$

$$b = \overline{b_2} + b_1 \cdot b_0 + \overline{b_1} \cdot \overline{b_0}$$

$$= \overline{b_2} + b_1 \oplus b_0$$

$$b = \overline{b_2} + b_1 \oplus b_0$$

For C output :-

$$K = \sum m(0, 1, 2, 3, 4, 7, 8, 9)$$

K-Map

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

	00	01	11	10
00	1	1	1	0
01	1	1	1	1
11	X	X	X	X
10	1	1	X	X

$$C = \bar{b}_1 + b_2 + b_0$$

For D output :-

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

	00	01	11	10	11	10	00	01
00	1		1	1			1	
01		1			1			1
11	X	X	X	X				
10	1	1	X	X				

$$G-1 : -b_2 b_0$$

$$G-2 : b_3$$

$$G-3 : \bar{b}_2 b_1$$

$$G-4 : -b_1 \bar{b}_0$$

$$G-5 : b_2 b_0 \bar{b}_1$$

$$d = \bar{b}_2 \bar{b}_3 + \bar{b}_3 + \bar{b}_2 b_1 + b_1 \bar{b}_0 + b_2 \bar{b}_0$$

For C Output :-

$$C = \sum m(0, 2, 6, 8)$$

		00	01	11	10	
		b ₂ b ₀	b ₁ b ₀	b ₁ b ₂	b ₂ b ₁	
		00				
		01				
		11	X	X	X	X
		10			X	X

$$G-1 := \overline{b_2} \overline{b_0}$$

$$G-2: b_1 \overline{b_0}$$

$$C = \overline{b_2} \overline{b_0} + b_1 \overline{b_0}$$

for output $C = \overline{d} \overline{d} + \overline{d} \overline{d} + \overline{d} \overline{d} = 0$

$$g = \sum m(0, 3, 4, 5, 6, 8, 9)$$

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

$$G-1 := b_3$$

$$G-2 := \overline{b_2} b_1$$

$$G-3 := b_1 \overline{b_0}$$

$$G-4 := b_2 \overline{b_1}$$

$$g = b_3 + \overline{b_2} b_1 + b_1 \overline{b_0} + b_2 \overline{b_1}$$

For f output:-

$$f = \sum m(0, 4, 5, 6, 8, 9) \quad m3 = 9$$

	00	01	11	10		
00	1				00	odd
01	1	1		1	10	odd
11	X	X	X	X		
10	1	1	X	X	11	odd

Q-3
Q-11
Q-2, Q-4

$$\begin{aligned} Q-1 &:= b_3 \\ Q-2 &:= \overline{b_1 b_0} \\ Q-3 &:= b_2 b_0 \\ Q-4 &:= b_2 b_1 \end{aligned}$$

$$\therefore f = b_3 + \overline{b_1 b_0} + b_2 \overline{b_0} + b_2 \overline{b_1}$$

Decimal to 7 Segment display Code

	01	11	10	00		
5-10					00	odd
4-5					10	odd
3-4						
2-3						
1-2						
0-1						

a
f | g | b
c | x | c
d | x | d

7-Segment
LED Display

$$1 \overline{d_5 d_4} + 1 \overline{d_3 d_2} + 1 \overline{d_1 d_0} = 6$$

- The logic circuit for the decimal to 7-segment display code is designed with 4 inputs and 7 outputs.

- Truth Table is constructed with the combination of inputs for each decimal number.

- Truth Table is constructed by listing 7 display i/p signals, decimal numbers & corresponding 4 digit binary numbers.

Decimal - 7 segment display

0 - $\begin{array}{c} a \\ f | g \\ \hline b | c \end{array}$

selected pixel $\begin{array}{c} 1 \\ \hline d \end{array}$

1 - $\begin{array}{c} b \\ f | g \\ \hline c \end{array}$

5. $\begin{array}{c} a \\ f | g \\ \hline b \\ \hline c \end{array}$

2. $\begin{array}{c} a \\ \hline g | b \end{array}$

6. $\begin{array}{c} a \\ f | g \\ \hline b \\ \hline c \end{array}$

10. $\begin{array}{c} c \\ \hline d \end{array}$

7. $\begin{array}{c} a \\ f | g \\ \hline b \\ \hline c \end{array}$

3. $\begin{array}{c} a \\ g | b \\ \hline c \end{array}$

8. $\begin{array}{c} a \\ f | g \\ \hline b \\ \hline c \end{array}$

4. $\begin{array}{c} f | g \\ \hline b \\ \hline c \end{array}$

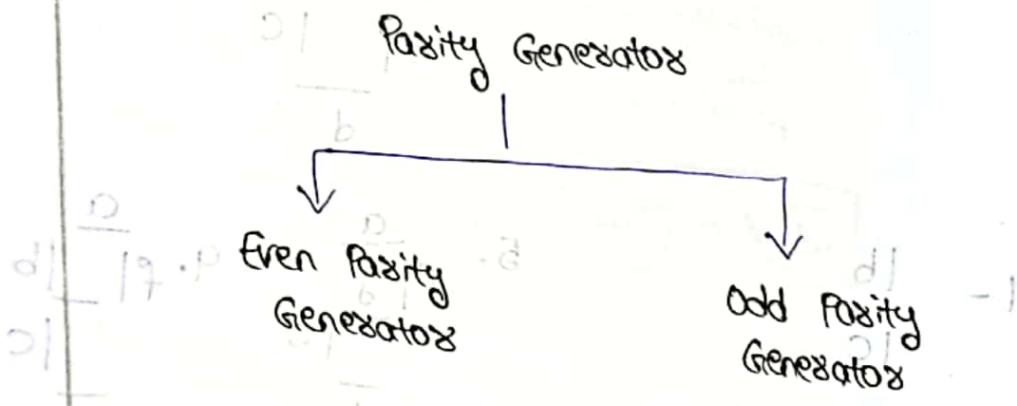
9. $\begin{array}{c} a \\ f | g \\ \hline b \\ \hline c \end{array}$

* Parity bit Generator :-

The Parity bit (or) check bit are the bits added to the binary code to check whether the particular code is in even parity or odd parity by number of 1's.

Parity Generator :- The parity generator is the combinational ckt at the transmitter. It takes an original message as i/p and generates the parity bit for that message.

Types of Parity Generator.



Even Parity Generator :-

1. It maintains the binary data in even no. of 1's.
2. If no. of 1's is even, parity bit value is 0.
3. If no. of 1's is odd, parity bit value is 1.

Odd Parity Generator :-

1. It maintains the binary data in an odd no. of 1's.
2. If no. of 1's is odd, parity bit value is 0.
3. If no. of 1's is even, parity bit value is 1.

* BCD to Gray Code Converter

The logical circuit which converts the binary code to equivalent gray code is known as "Binary to Gray code converter".

Gray Code :- It is a binary number system in which every successive pair of numbers differs in only one bit.

* A binary number is converted to gray code to reduce the switching operation.

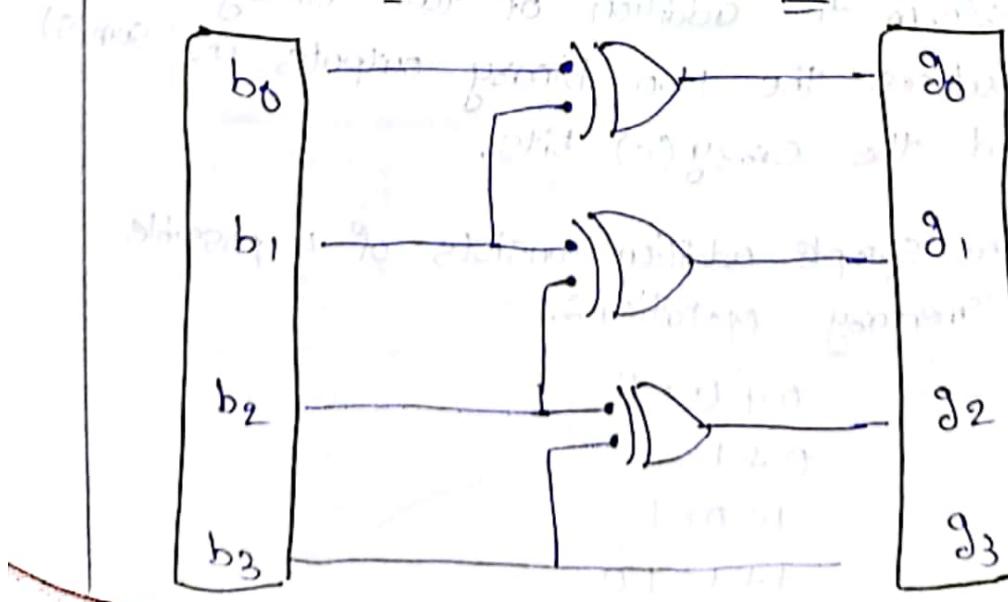
* In this conversion process, 4 bit BCD code is considered as Input which is converted into 4-bit gray code. ~~has 2nd and 4th without bit~~

* For this BCD to gray code conversion.

1. The MSB is kept the same. $g_3 = b_3$
2. Next, take XOR of 1st & 2nd binary bit $g_2 = b_2 \oplus b_3$
3. Next, take XOR of 3rd & 4th binary bit $g_1 = b_1 \oplus b_2$
4. Lastly take XOR of 3rd & 4th binary bit $g_0 = b_0 \oplus b_1$

BCD code (I/P)

Gray code (O/P)



Arithmetic Circuits

Arithmetic circuits are nothing but the logic circuits, which is used to complete some Arithmetic functions like Addition, Multiplication, Subtraction, Division etc.

* Arithmetic circuits are:

- Adders
- Subtractors
- Multipliers, etc....

Adders :- An adder is a device that will add together two bits and result as the output.

These are two kinds of adders

1. Half Adder
2. Full Adder

1. Half Adder

• Half Adder is a combinational circuit used to perform the addition of two binary inputs & produces the two binary outputs as sum(S) and the carry(C) bits.

- The sample addition consists of 4 possible elementary operations.

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 10$$

Truth Table for Half Adder:-

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

K-Map for Carry (C)

A \ B	00	01
0	0	1
1	1	0

↓ ↓

G-1 G-2

Boolean expression for sum

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

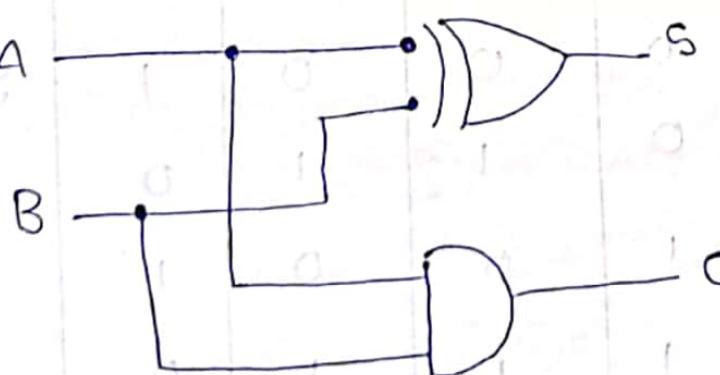
(or)

$$S = A \oplus B$$

$$G-1: \bar{A}B$$

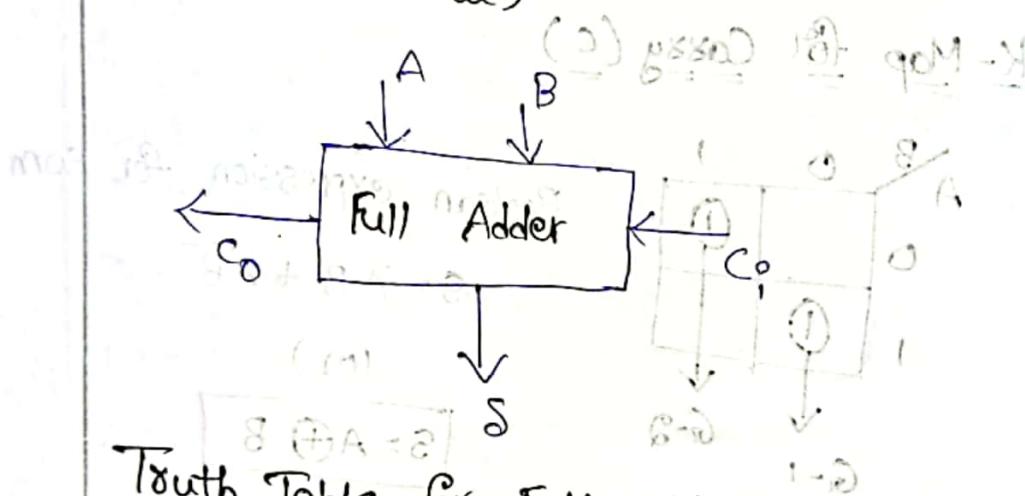
$$G-2: \bar{A}B$$

Logic circuit for Half Adder :-



2. Full Adder :-

- Full Adder is a combinational circuit used to perform the addition of 3 no. of Inputs and produces two outputs.
- The Full Adder adds the bits A & B and the carry from the previous column called the "carry-in (C_{in})" and gives output as the "sum bit (S)" and the carry bit called the "carryout (C_{out})".



Truth Table for Full Adder :-

A	B	C _i	C _o	S
0	0	0	0	0
1	0	0	1	1
2	1	0	0	1
3	1	0	1	0
4	0	1	0	1
5	0	1	1	0
6	1	1	0	0
7	1	1	1	1

K-Map for Sum :-

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

		AC _i	00	01	11	10
		B	0	1	3	2
C _i	B	0	0	1	1	0
		1	1	0	1	1

Boolean expression for Sum :-

$$S = \overline{B} \overline{A} C_i + \overline{B} A \overline{C}_i + B \overline{A} \overline{C}_i + B A C_i$$

$$= \overline{B} (\overline{A} C_i + A \overline{C}_i) + B (\overline{A} \overline{C}_i + A C_i)$$

$$= \overline{B} (A \oplus C_i) + B (\overline{A} \oplus C_i)$$

$$S = (A \oplus B) \oplus C_i$$

K-Map for Carryout (C₀)

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

		AC _i	00	01	11	10
		B	0	1	3	2
C _i	B	0	0	1	1	0
		1	1	0	1	1

Boolean expression for Carryout

$$C_0 = BA + B \overline{A} C_i + \overline{B} A C_i$$

$$= BA + C_i (BA + \overline{BA})$$

$$= AB + C_i (A \oplus B)$$

$$C_0 = AB + C_i (A \oplus B)$$

Logical circuit for the Full Ladder

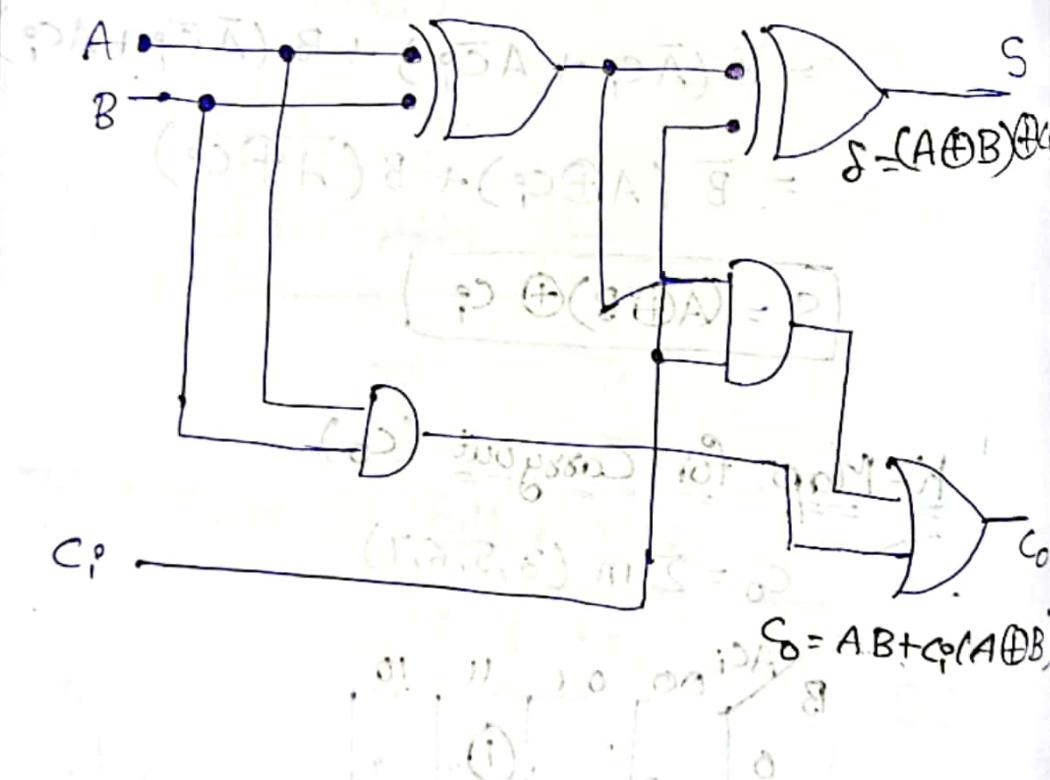
Logical circuit (for the) full ladder from the obtained boolean expression is.

$$S = (A \oplus B) \oplus C_0$$

$$C_0 = AB + C_1(A \oplus B)$$

Here, 2-XOR Gates,

2-AND Gates, 1-OR Gate required.



$$C_0 = AB + C_1(A \oplus B)$$

Augend and Addend :- Augend and addend are added or summed up to find a sum or total.

3. Binary Adder (Ripple Carry Type adder)

- Multiple bits binary adder is obtained by using no. of full adders connected in cascade.
- For the 1st full adder (of LSB) of Augent & Addend along with carryin (C_{in}) are inputs.
- Its sum and carry out are outputs. The carry out is used as carryin for next full adder. Other i/p's for the next full adder are the next significant bits.
- It again generates the sum & carry.
- The process is repeated till the last MSB.

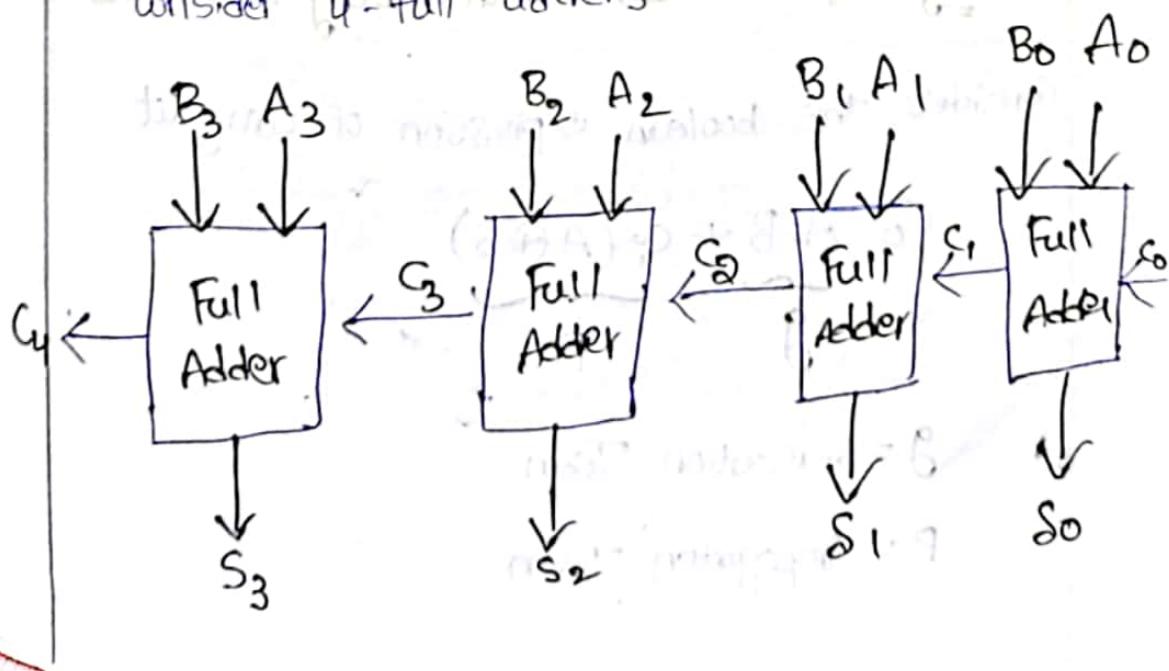
4-bit Binary Adder :-

Consider, A & B contain 4 bits.

$$A = A_3 A_2 A_1 A_0$$

$$B = B_3 B_2 B_1 B_0$$

Consider 4-full adders.



Carry-Lookahead Adder (CLA)

- A carry-lookahead Adder (CLA) or fast adder improves speed by reducing the amount of time required to determine carry bits.

- The CLA adder calculates one or more carry before the sum. It reduces the wait time to calculate the result of larger-value bits of the adder.

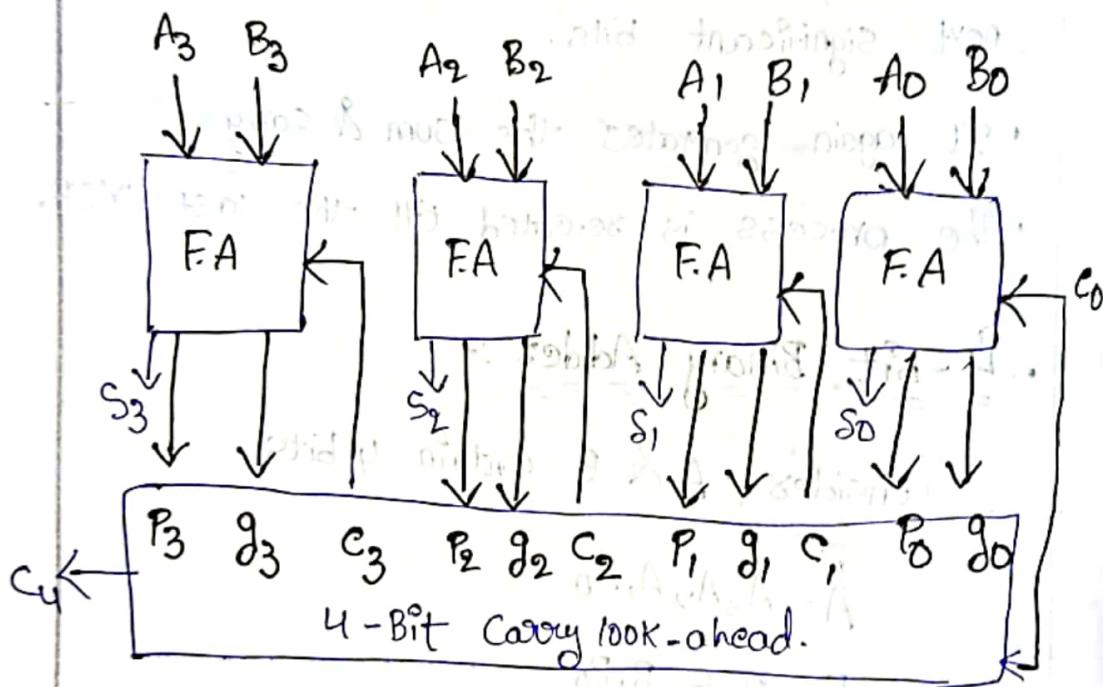


Fig:- 4-bit Adder with Carry Lookahead.

Consider the boolean expression of carryout

$$C_0 = A \cdot B + C_0' (A \oplus B)$$

$\underbrace{g}_{\text{Generation Term}}$ $\underbrace{P}_{\text{Propagation Term}}$

g = Generation Term

P = Propagation Term

The General expression for C_{n+1}

$$C_i = g_i + P_i C_{i-1}$$

$$C_2 = g_2 + P_2 C_{2-1} \Rightarrow C_2 = g_2 + P_2 C_1$$

$$\therefore C_0 = g_0 + P_0 C_{-1} \rightarrow (1)$$

$$\begin{aligned} C_1 &= g_1 + P_1 C_0 \\ &= g_1 + P_1 (g_0 + P_0 C_{-1}) \quad [\text{from eq (1)}] \end{aligned}$$

$$C_1 = g_1 + P_1 g_0 + P_1 P_0 C_{-1} \rightarrow (2)$$

$$\begin{aligned} C_2 &= g_2 + P_2 C_1 \\ &= g_2 + P_2 (g_1 + P_1 g_0 + P_1 P_0 C_{-1}) \quad [\text{from eq (2)}] \end{aligned}$$

$$C_2 = g_2 + P_2 g_1 + P_1 P_2 g_0 + P_1 P_2 P_0 C_{-1} \rightarrow (3)$$

$$C_3 = g_3 + P_3 C_2 \quad [\text{from eq (3)}]$$

$$C_3 = g_3 + P_3 g_2 + P_3 P_2 g_1 + P_3 P_1 P_2 g_0 + P_3 P_1 P_2 P_0 C_{-1} \rightarrow (4)$$

Here

$$g_3 = A_3 \cdot B_3$$

$$P_3 = A_2 \oplus B_2$$

Complements :-

Complements are used in digital circuits, because it is faster to subtract by adding complements than by performing true subtraction.

- Complements are two types.

1. Radix complement ($8's$ complement)

2. Diminished radix complement ($8-1$)'s complement

* $8's$ Complements

Ex. 1: Complement of $(7)_{10}$

Formula : $8^n - N$

$8 = \text{radix/base}$

$n = \text{Total no. of digits}$

$N = \text{Given Number}$

$$(10)^1 - 7 = 3$$

$\therefore \text{Complement of } (7)_{10} = 3$

2. Complement of $(5690)_{10}$

$8 = 10, N = 5690, n = 4$

$$8^4 - N = (10)^4 - 5690$$

$$= 10000 - 5690$$

$$= 4310$$

$\therefore \text{Complement of } (5690)_{10} = 4310$

3. Complement of $(1101)_2$

$$y=2$$

$$N = 1101$$

$$n=4$$

$y^N = 2^4 = 16$ (in octal) $2^4 = 16$

$$= (16)_{10} - 1101$$

$$= 10000 - 01101$$

↓ Complement

$$10010$$

$$+ 1$$

$$\hline 10011$$

$$\Rightarrow \begin{array}{r} 10000 \\ 10011 \\ \hline \boxed{100011} \end{array}$$

∴ Complement of $(1101)_2 = 000011_2$

$(y-1)$'s Complement

Here,

If

$$y=10$$

$$y=2$$

$(y-1)$'s Complement

9's Complement

is 1000

$$y=8$$

7's Complement

$$y=16$$

F's Complement

formula

$$(y-1)'s \text{ Complement} = y^n - N - 1$$

* $(8-1)$'s comp $\stackrel{\text{defn}}{=} 8$'s comp - 1

* $(8-1)$'s comp + 1 = 8's comp

Ex:- Find 7's complement of octal no. $(5674)_8$

Sol $(8-1)$'s complement $\stackrel{\text{defn}}{=} 8^4 - N - 1$

$$\begin{aligned} 8 &= 8 & 10110 - 00001 - \\ n &= 4 & \downarrow \\ N &= 5674 & 01001 \\ && \downarrow + \\ \therefore 8^4 - N - 1 &= 8^4 - 5674 - 1 \\ &= 7777 - 5674 & \begin{array}{r} 00001 \\ 11001 \\ \hline \end{array} \leftarrow \\ &= 2103 & \boxed{11001} \end{aligned}$$

$\therefore (8-1)$'s complement of $(5674)_8$ is 2103.

* 10's Complement $\stackrel{\text{defn}}{=} 2(10-1)$

1. $5896_{10} \rightarrow 2(10-1)$

$$\begin{aligned} 10^4 - N &= 10^4 - 5896 \\ &= 4104 \end{aligned}$$

Subtracting Using 10's complement

1. Find $9-4$

$$\begin{array}{r} 10 \\ - 4 \\ \hline 6 \end{array} \quad \begin{array}{r} 9 \\ - 6 \\ \hline 3 \end{array} \quad \text{Carry} = 10 \rightarrow \text{+ve}$$

$$9-4=5$$

$$9-4=5 \quad 10^1-N$$

Q: (18098).8

$$10+9+(-4)=5+10$$

$$9+(10-4)=15$$

$$9+6=15$$

$$15=15$$

Q: (4598).8

2. find $88-32$.

$$\begin{array}{r} 100 \\ 38 \\ \hline 62 \end{array} \Rightarrow \begin{array}{r} 88 \\ 68 \\ \hline 20 \end{array}$$

$$\therefore 88-32=56$$

$$\begin{array}{r} 100 \\ 88 \\ \hline 12 \end{array} \Rightarrow \begin{array}{r} 100+88-32=56+100 \\ 188-12 \\ 176 \\ 156 \end{array}$$

3. find $72532-3250$

$$\begin{array}{r} 100000 \\ -03250 \\ \hline 96750 \end{array}$$

$$\begin{array}{r} 72532 \\ -3250 \\ \hline 69282 \end{array}$$

$$\boxed{72532-3250=69282}$$

* 9's Complement

$$1, (2457)_{10}$$

$$10^4-N-1 = (10^4-1)-2457$$

$$= 9999-2457$$

$$= 7542.$$

2. $(89031)_{10}$

$$\begin{aligned} 10^7 - N - 1 &= 10^5 - 1 - 89031 \\ &= 99999 - 89031 \\ &= 10968. \end{aligned}$$

3. $(2974)_{10}$

$$\begin{aligned} 10^7 - N - 1 &= 10^4 - 1 - 2974 \\ 1000 + 2974 &= 68 - 88 + 100 \\ &= 9999 - 2974 \\ 1000 - 2974 &= 7025 \end{aligned}$$

$$\begin{array}{r} 68 - 88 \\ \hline 82 \\ 88 \\ \hline 8 \\ 8 \\ \hline 0 \end{array}$$

Subtraction using 9's Complement

1. Find $1234 - 1000$

Minuend 1234 Subtrahend 1000

Step-1: $1234 - 1000$

1234
 1000
 $\hline 234$

1. 9's complement of B.

Step-2: $1234 + 8999$

1234
 8999
 $\hline 10233$

3. Add carry to A.

Step-3: $10233 + 1 = 1234$

$\therefore 1234 - 1000 = 234$

81.2 from 98.3.

Sol: $1. 999$ $\begin{array}{r} 81.2 \\ 18.7 \\ \hline 18.7 \end{array}$

2. $1 - 4.8$

2. Find $1234 - 2000$

Step-1:

$$\begin{array}{r} 9999 \\ - 2000 \\ \hline 7999 \end{array}$$

Step-2:

$$\begin{array}{r} 1234 \\ + 7999 \\ \hline 9233 \end{array}$$

NO carry

Step-3:

$$\begin{array}{r} 110010 \\ - 9999 \\ - 9233 \\ \hline 00766 \end{array}$$

9's complement

$$1234 - 2000 = -766$$

* 1's Complement

① 1's complement of binary number 01001011

Given 01001011

$$\begin{array}{r} 01001011 \\ \hline 10110100 \end{array}$$

1's complement

$$01001011 \xrightarrow{\text{is}} 10110100$$

$$[a_1(P) = a_1(P1) - a_1(S)]$$

backward of 1's complement

Subtraction using 1's Complement

$$1. (28)_{10} - (19)_{10}$$

Step-1: $(28)_{10} = (011100)_2$

$$(19)_{10} = (010011)_2$$

Step-2: $(19)_{10} = \frac{010011}{\downarrow 1's} \quad 1's$

transposed 20

$$\begin{array}{r} 010011 \\ \hline 101100 \end{array}$$

Step-3: Add $(28)_{10}$ & 1's of 19

$$\begin{array}{r}
 011100 \\
 + 101100 \\
 \hline
 1010010
 \end{array}$$

transposed 213
Carry-1 neglect & add.
transposed 213

Step-4: $0010010 \rightarrow (9)_{10}$

transposed 213
Notated 9

$$\therefore (28)_{10} - (19)_{10} = (9)_{10}$$

1. 1's Complement to Subsahand

2. Add to minued

3. Carry generated and add to result

4. m>n carry is generated

5. m<n no. carry then again do complement
do result.

$$Q. (15)_{10} - (28)_{10}$$

Step-1: $(15)_{10} = (1111)_2 = (8)_{10}$

$$(28)_{10} = (11100)_2 = (5)$$

Step-2: $(28)_{10} = \underline{11100}$

\swarrow is Comp.

$$000\underline{11}$$

Step-3: Adding

$$01101001$$

$$00011$$

$$\underline{10010} = -13$$

Step-4: Taking complement of 10010

$$10010$$

$$\boxed{011010} = (13)_{10}$$

3. $(0111)_2 - (0011)_2$ (or) (7-3)

Both the numbers have equal length

Step-1: $(0111)_2 = (0011)_2$

is complement of $(0011)_2 = (1100)_2$

Step-2: 0111 is borrowed as bit

$$\begin{array}{r} +1100 \\ \hline 10011 \end{array}$$

Step-3: 0011

$$\begin{array}{r} 1 \\ \hline 0100 \end{array}$$

$$\begin{array}{r} 01011111 \\ \hline \Rightarrow (4)_{10} \end{array}$$

$$\boxed{7-3=4}$$

4. 3-7.

Step-1:- $(3)_{10} = (0011)_2 = \text{01}(\text{E1})$ 48-998

$(7)_{10} = (0111)_2 = \text{01}(\text{E1})$

Step-2:- Take 1's complement of $(7)_{10}$

$$\begin{array}{r} 0111 \\ \hline \downarrow \text{1's} \\ 1100 \end{array}$$

1000110 48-998

Step-3:- $\begin{array}{r} 11000 \\ 0011 \\ \hline 1000 \end{array}$

0100110 48-998
Carry is not generated.
 01001 is (-ve)

Step-4:- Taking 2's complement.

$$\begin{array}{r} 01(\text{E1}) \\ \hline 1000110 \end{array}$$

$(8-5) (0001001100) = (310) 48-998$

$(0100)_2 = (4)_{10}$ 48-998

$01100 = (110) 48-998$

2's complement to the required 8 bits.

Find 2's complement of 00000101

$$\begin{array}{r} 00000101 \\ \hline \downarrow \text{1's} \\ 11111010 \end{array}$$

$$\begin{array}{r} 11111010 \\ + 1 \\ \hline 11111011 \end{array}$$

Subtraction using 2's Complement

1. $12 - 18$

Step-1: $(12)_{10} = (01100)_2$

$(18)_{10} = (10010)_2$

Step-2: Taking 2's complement of $(18)_{10}$

$$\begin{array}{r} 10010 \\ \hline + 1 \\ \hline 01101 \\ \hline \end{array}$$

Step-3: Adding 01100

$$\begin{array}{r} 01100 \\ + 01110 \\ \hline 11010 \end{array}$$

No carry is generated so the result is (-ve)

Step-4: Taking 2's complement of $(11010)_2$

$$\begin{array}{r} 11010 \\ \hline + 1 \\ \hline 00110 \end{array}$$

1. 2nd Complement to Subtrahend

2. Add to minuend

3. Carry generated neglect it is not generated again
2's complement

Q. 7-2.

Step-1: $(7)_{10} = (111)_2$

$(2)_{10} = (010)_2$

Step-2: Taking 2's complement of $(2)_{10}$.

$$\begin{array}{r} 010 \\ \downarrow \text{is } \downarrow \\ 101 \\ + 11 \\ \hline 110 \\ \hline 0110 \end{array}$$

Step-3: Adding

$$\begin{array}{r} 0111 \\ 0110 \\ \hline 0101 + \underline{\text{tvc}} \end{array}$$

(sum) of 110206 2's complement of 10000000000000000000000000000000

$$(101)_2 = (5)_{10}$$

$(21011) \boxed{7-2=5}$ 2's complement of 10000000000000000000000000000000

$$\begin{array}{r} 01011 \\ \downarrow \text{and } \downarrow \\ 10100 \end{array}$$

1's Complement

" by '0' & '1' by '1'

$$3 = 0011$$

..... branching to lowest and then

$$-3 = 1100$$

..... branching to highest

..... writing the negative sign of 3

Two's complement

Step-1: 1000110110000000

1000110110000000
↓ is complement

1100101000000000

Step-2: 1100101000000000

+ 1

1101000000000000

1101000000000000

present pd behavior gd no 9's complement

9's complement result is 01101000

also 9's complement of 1000110110000000

1234 8765 N=4

N=1234

9's complement: 01101000

9's complement of 1000110110000000

discrepancy found 9's complement of 1000110110000000

discrepancy found 9's complement of 1000110110000000

10's complement

10's complement of 1000110110000000

8634 9999

9's complement 9999

-8634
1365

9's complement of 1000110110000000

10's complement = 1365 + 1

9's complement of 1000110110000000 = 1366

Array Multiplier :-

An Array multiplier is a digital Combinational circuit used for multiplying two binary numbers by employing an array of full adders and half adders.

Multiplication Schemes,

Two Types of Multiplication Schemes.

Serial Multiplication (shift - Add) :- The serial multiplication operation can be solved by finding partial products & then ~~adding~~ ^{partial product} together. The implementations are primitive with simple architecture.

Parallel Multiplication : The parallel products are generated simultaneously in parallel multiplication and a high performance machine parallel implementation is applied, latency is minimized.

* Multiplication process has ~~three~~ ^{three} main steps:

- Partial product generation
- Partial product reduction
- Final addition

* Multiplicand : a quantity which is to be multiplied by another (the multiplier)

* Multiplicand : a quantity by which the multiplicand is to be multiplied.

4x4 Multiplication: at 241 96.00 - m

Multiplicand :- $A = a_3 \ a_2 \ a_1 \ a_0$

$$\text{Multiplied :- } B = b_3 \text{ } b_2 \text{ } b_1 \text{ } b_0$$

$$a(6-a) a_3 a + a_2 a_1 a_0 \rightarrow 0$$

$$b_3 = b_2 \frac{b_1 b_0}{\text{not applicable}} \text{ in 2008}$$

93b0 93b0 93b0 93b0 93b0 93b0 } 93b0

to calculate a budget and then bring it to Product

a_3b_3 a_2b_3 a_1b_3 a_0b_3 of id a_0a b_0b_0 a_1 b_1b_1

2022-03-07 Reitsch Ti been ei gewünscht 2003

Ex:- $\frac{1}{2} \times \frac{1}{3} \times \frac{1}{4} \times \dots \times \frac{1}{n}$ is a decreasing sequence of real numbers.

11 81

Werte $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ \Rightarrow $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ \Rightarrow $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$

PHD 601010010 brand si ji wifos 21
2101010010 2101010010

Condition 1: $\text{biovolume} = \text{robust volume}$

$m = \text{no. of bits in multiplicand}$
 $= \text{no. of bits in multiplier}$
 $D^0, P^0, S^0 = \text{A 4-bit binary word}$
 $\text{No. of AND gates} = m \times n$
 $\text{no. of half adders} = n$
 $\text{no. of Address} = (m-1)n$
 $\text{no. of Full Address} = (m-1)n = (m-2)n$

Steps in Multiplication

The following are the steps of any multiplication

- If LSB of multiplicand is 1, then add the multiplicand into an accumulator. multiplicand bit is shifted one bit to the right and multiplicand bit is shifted one bit to the left.
- Stop when all the bits of the multiplicand is zero.
- Less hardware is used if partial products are added serially. We can add all PP by a parallel multiplier. However, it is possible to use compression technique the number of partial products can be reduced before addition, is performed.

Construction & Working of a 4×4 Array Multiplier

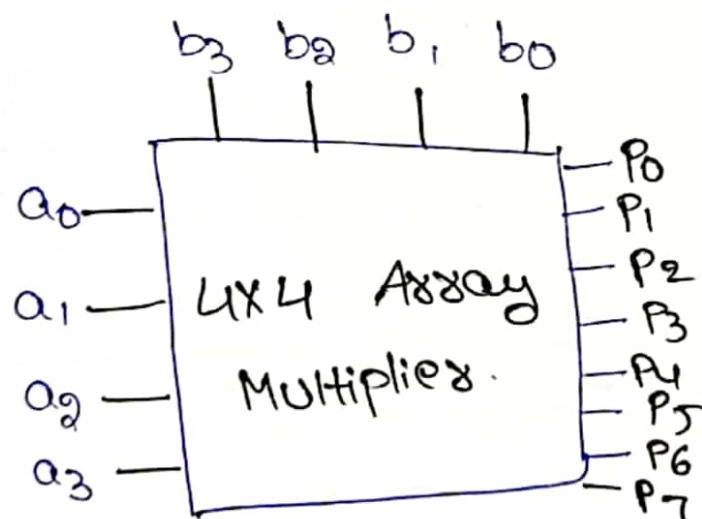
The design structure of the array multiplier is regular, it is based on the add shift algorithm principle

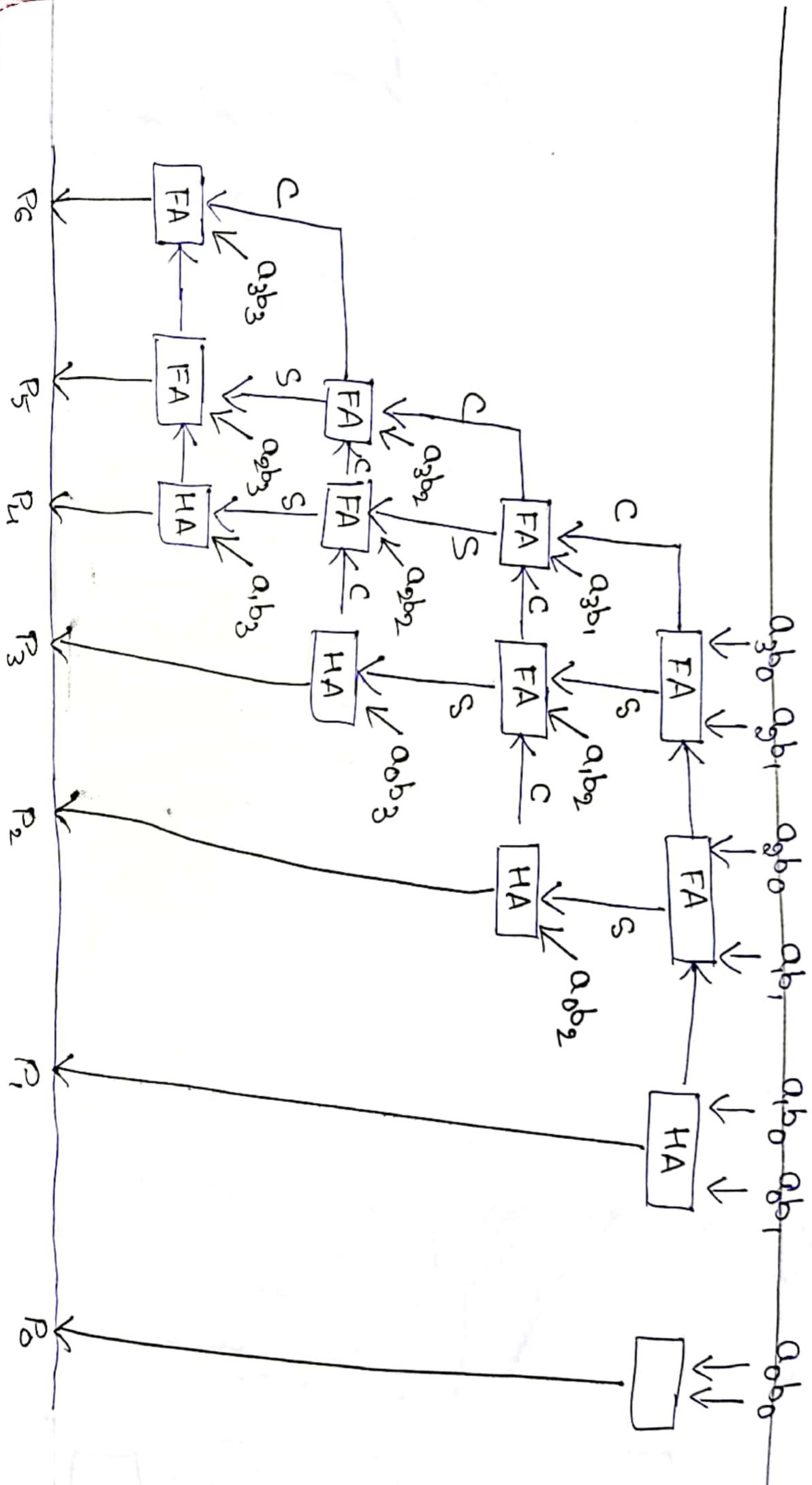
Partial Product = Multiplicand * Multiplier

AND gates are used for the product, the summation is done using full Address and Half Address where the partial product is shifted according to their bit orders.

In an $n \times n$ array multipliers, $n \times n$ AND gates compute the partial products, and the addition of partial products can be performed by using $n \times (n-2)$ Full address and n Half address.

The 4×4 array multiplier shown has 8 inputs & 8 outputs.





BCD Addition

$$1. 9+9=18$$

$$2. 5+6=11$$

$$(9)_{10} = (1001)_2$$

$$0101 \text{ H}$$

$$0110 \text{ H}$$

$$(9)_{10} = (1001)_2$$

$$\begin{array}{r} 1001 \\ + 1001 \\ \hline 10101 \end{array}$$

$$\begin{array}{r} 1001 \text{ H} \\ + 1001 \text{ H} \\ \hline 10010 \end{array}$$

$$\begin{array}{r} 0101 \text{ H} \\ + 1010 \text{ H} \\ \hline 1111 \end{array}$$

$$\begin{array}{r} 1 \ 2 \text{ H} \\ + 6 \text{ [add 6]} \\ \hline 18 \end{array}$$

Problem 2: 18 with 6 to 12 with 6 to 10.

$$3. 7+5=12$$

$$4. 3+2=5$$

$$\begin{array}{r} 011101 \text{ to 1100} \\ + 0101 \text{ to 1010} \\ \hline 1100 \end{array}$$

$$0011$$

$$0010$$

$$0101$$

$$5 \text{ H.}$$

$$\begin{array}{r} 12 \text{ to 1010} \\ + 6 \text{ to 0110} \\ \hline 18 \end{array}$$

Decimal 18 equivalent

Hexa decimal is 12H

$$16 \mid 18$$

Note :-

Digitally 108

In this BCD addition, if the addition result is more than 9, need to add 6 to add BCD.

Otherwise we need to add $(100_2) = 4_{10}$.

BCD Addition :-

$(1001)_2 = 9_{10}$

Step-1 :- At first the given numbers are to be added using the rule of binary

Ex :- Case-1 :- 1010
 $+ 0101$
 $\hline 1111$

Case-2 :- 0001
 $+ 0101$
 $\hline 0100$

Step-2 :- In case-1, the result of addition of two binary numbers is greater than 9, which is not valid for BCD numbers.

In case-2, the result of addition is less than 9, which is valid for BCD numbers.

Step-3 :- If the 4 bit result of addition is greater than 9 and if a carry bit is present in the result then it is invalid and we have to add 6 (0110) to the result of addition.

Step-4 :- Then the resultant that we would get will be a valid binary coded number.

breaking 81 into
16's & 1's binary result

81
16
6

Ex:- sum of 01010 and 01110

$$\begin{array}{r} + 01010 \\ \hline 11110 \end{array} \rightarrow \text{Invalid BCD numbers}$$
$$\begin{array}{r} + 01110 \\ \hline 00010101 \end{array} \rightarrow \text{Valid BCD number.}$$

Ex:- Let, 0101 is added with 0110

$$\begin{array}{r} 0101 \\ + 0110 \\ \hline 1011 \end{array} \rightarrow \text{Invalid BCD numbers}$$
$$\begin{array}{r} 0110 \\ + 0101 \\ \hline 00010001 \end{array} \rightarrow \text{Valid BCD number.}$$

check:-

$$(0101)_2 = (5)_{10}$$

$$(0110)_2 = (6)_{10}$$

$$(5)_{10} + (6)_{10} = (11)_{10}$$

Ex:- Now, let 00010001 is added to 00100110

$$\begin{array}{r} 00010001 \\ + 00100110 \\ \hline 00110111 \end{array} \rightarrow \text{Valid BCD number}$$

check:-

$$(00010001)_{BCD} \rightarrow (11)_{10}$$

$$(00100110)_{BCD} \rightarrow (26)_{10}$$

$$(00110111)_{BCD} \rightarrow (37)_{10}$$

So no need to add 06 as 1 because both

$$(0011)_2 = (3)_{10}$$

$$(0111)_2 = (7)_{10}$$

are less than $(9)_{10}$.

This is the process of BCD Addition.

BCD ~~Binary~~ ^{Definition} : ~~Binary~~ ^{Binary} coded decimal is the storage of numbers in which each decimal digit is converted into a ¹⁰¹⁰ binary number and stored in a single 8-bit byte.

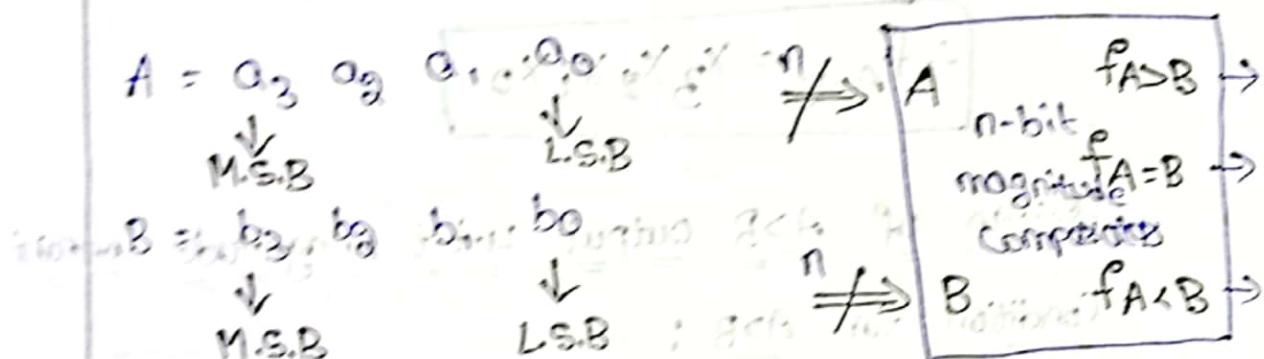
Decimal	Binary number 4 bits	BCD [Binary Coded Decimal]
0	0000	0000
1	0001	0001
2	0010	0010
3	0011	0011
4	0100	0100
5	0101	0101
6	0110	0110
7	0111	0111
8	1000	1000
9	1001	1001
10	1010	00010000
11	1011	00010001
12	1100	00010010
13	1101	00010011
14	1110	00010100
15	1111	00010101

Magnitude Comparators

A magnitude comparator is a combinational circuit that compares two numbers A & B to determine whether:

- $A > B$ (or) $A > B$ $\leftarrow A > B$
- $A = B$ (or) $A = B$ $\leftarrow A = B$
- $A < B$ $\leftarrow A < B$

Let us consider two numbers A & B



(a) Inputs are n -bit binary no. $\leftarrow a_3, a_2, a_1, a_0$

Inputs :-

1. First n -bit number A $\leftarrow a_3, a_2, a_1, a_0$

2. Second n -bit number B $\leftarrow b_3, b_2, b_1, b_0$

Outputs :- 1. $f_{A>B} \leftarrow a_3, a_2, a_1, a_0$

2. $f_{A=B} \leftarrow a_3, a_2, a_1, a_0$

3. $f_{A<B} \leftarrow a_3, a_2, a_1, a_0$

is $f_{A<B} \leftarrow a_3, a_2, a_1, a_0$

if $a_3 = a_2, a_1, a_0$

is $f_{A>B} \leftarrow a_3, a_2, a_1, a_0$

is $f_{A=B} \leftarrow a_3, a_2, a_1, a_0$

Design of Output ($A=B$) in 4-bit Magnitude Comparator.

Condition for $A=B$:

$(A=B) \Leftrightarrow f_f$

$A_3 = B_3 \rightarrow (x_3 = 1) \text{ and } B < A$

$A_2 = B_2 \rightarrow (x_2 = 1) \text{ and } B = A$

$A_1 = B_1 \rightarrow (x_1 = 1) \text{ and } B > A$

$A_0 = B_0 \rightarrow (x_0 = 1)$

$$\therefore f_{A=B} = x_3 x_2 x_1 x_0$$

Design of $A > B$ output 4-bit magnitude Comparator.

Condition for $A > B$:

If $a_3 > b_3 \rightarrow a_3 = 1 \text{ and } b_3 = 0 \Rightarrow a_3 \bar{b}_3 \text{ (or)}$

If $(a_3 = b_3) \text{ and } (a_2 > b_2) \Rightarrow x_3 a_2 \bar{b}_2 \text{ (or)}$

If $(a_3 = b_3) \text{ and } (a_2 = b_2) \text{ and } (a_1 > b_1) \Rightarrow x_3 x_2 a_1 \bar{b}_1$

If $(a_3 = b_3) \text{ and } (a_2 = b_2) \text{ and } (a_1 = b_1) \text{ and } (a_0 > b_0) \Rightarrow x_3 x_2 x_1 a_0 \bar{b}_0$

To summarize, $(A > B) \Leftrightarrow$

1. $a_3 \bar{b}_3 = 1 \text{ or}$

2. $x_3 a_2 \bar{b}_2 = 1 \text{ or}$

3. $x_3 x_2 a_1 \bar{b}_1 = 1 \text{ or}$

4. $x_3 x_2 x_1 a_0 \bar{b}_0 = 1$

$$\therefore f_{A > B} = a_3 \bar{b}_3 + x_3 a_2 \bar{b}_2 + x_3 x_2 a_1 \bar{b}_1 + x_3 x_2 x_1 a_0 \bar{b}_0$$

Design of output (A < B) 4 bit magnitude compare

Condition for $A < B$:

If $a_3 < b_3 \rightarrow a_3 = 0$ and $b_3 = 1 \Rightarrow \bar{a}_3 \cdot b_3$ (or)

If $(a_3 = b_3)$ and $(a_2 < b_2) \Rightarrow x_3 \bar{a}_2 b_2$ (or)

If $(a_3 = b_3) \& (a_2 = b_2) \& (a_1 < b_1) \Rightarrow x_3 x_2 \bar{a}_1 b_1$ (or)

If $(a_3 = b_3) \& (a_2 = b_2) \& (a_1 = b_1) \& (a_0 < b_0) \Rightarrow x_3 x_2 x_1 \bar{a}_0 b_0$

To summarize $(A < B)$ ~~if~~ \Rightarrow ~~if~~ \Rightarrow ~~if~~

1. $\bar{a}_3 \cdot b_3 = 1$ (or)

2. $x_3 \bar{a}_2 b_2 = 1$ or

3. $x_3 x_2 \bar{a}_1 b_1 = 1$ or

4. $x_3 x_2 x_1 \bar{a}_0 b_0 = 1$

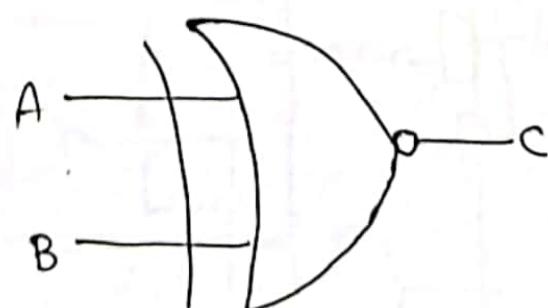
$$\therefore f_{A < B} = \bar{a}_3 \cdot b_3 + x_3 \bar{a}_2 \cdot b_2 + x_3 x_2 \bar{a}_1 \cdot b_1 + x_3 x_2 x_1 \bar{a}_0 \cdot b_0$$

A exclusive NOR gate is easiest way to

compare the equality of other bits.

X-NOR

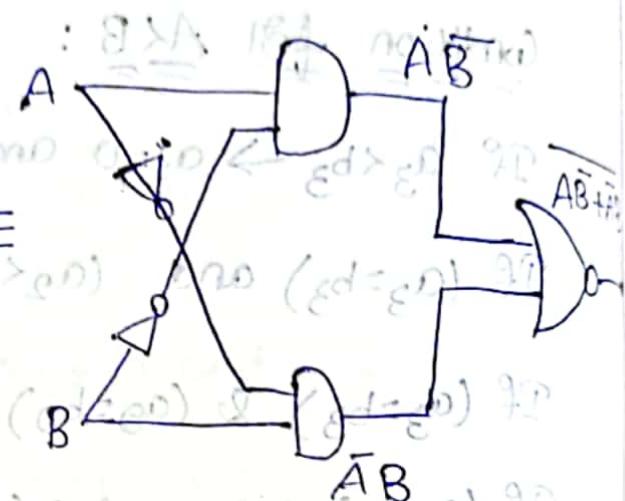
A	B	C
0	0	1
0	1	0
1	0	0
1	1	1



$$\text{EX-NOR} = \text{EX-OR} = \overline{AB} + A\overline{B}$$

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

$$(ii) \quad \overline{AB} = \overline{A} \cdot \overline{B} \cdot \overline{X} = \overline{A} \cdot \overline{B} \cdot \overline{X} \cdot \overline{X}$$



Instead of XNOR circuit we are going to use this circuit to check the given binary numbers are equal or not.

The gate implementation of the three outputs ($A < B$), $A = B$, $A > B$, using the obtained boolean expressions.

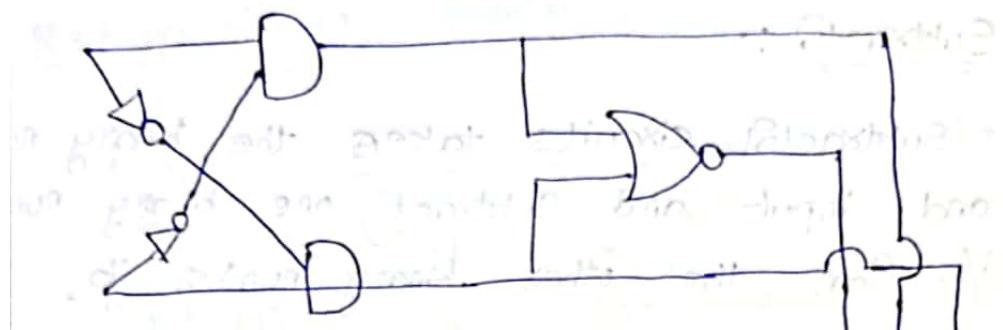
$$f_{A=B} = x_3 \overline{x_2} x_1 x_0 + \overline{x_3} x_2 \overline{x_1} x_0 = x_3 \overline{x_2} x_1 x_0 + \overline{x_3} x_2 \overline{x_1} x_0$$

$$f_{A>B} = \overline{a_3} \overline{b_3} + x_3 \overline{a_2} \overline{b_2} + x_3 x_2 \overline{a_1} b_1 + x_3 x_2 x_1 \overline{a_0} b_0$$

$$f_{A < B} = \overline{a_3} b_3 + x_3 \overline{a_2} b_2 + x_3 x_2 \overline{a_1} b_1 + x_3 x_2 x_1 \overline{a_0} b_0$$

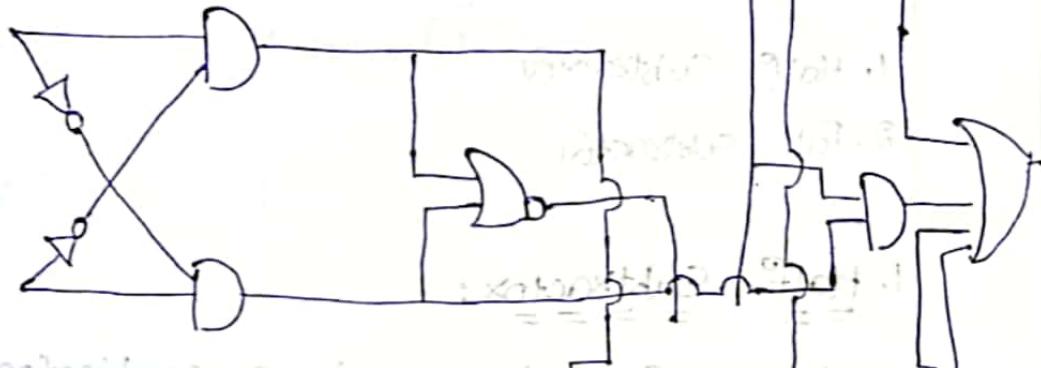


	0	1	0	1
0	0	0	0	0
1	0	1	0	1
0	0	1	1	0
1	1	0	1	1



but consider the following and then see if it is true.

କାନ୍ତିରେ କାନ୍ତିରେ କାନ୍ତିରେ କାନ୍ତିରେ କାନ୍ତିରେ କାନ୍ତିରେ



↳ longitudinal o es extensión? ↳ ↳

go nonpartisan model at least as

low (normal) A. *strobli* S. *min*

10 224-2713 + 2714-2715 2716-2717 2718-2719

so skin the bone today final we'll see you

so older than the today's sign is the one

Diagram 2: A graph with 6 nodes and 7 edges. The nodes are arranged in two rows: the top row has 3 nodes, and the bottom row has 3 nodes. Edges connect the following pairs of nodes: (1,2), (1,3), (2,4), (3,5), (4,6), (5,6), and (5,7). Node 7 is isolated.

A diagram of a hand with fingers and a thumb. A small circle is drawn on the thumb, and a small square is drawn on the index finger.

551053-17

Wenod — structured —

11. *Leucosia* *leucostoma* (Fabricius) *leucostoma* (Fabricius) *leucostoma* (Fabricius)

A line drawing of a classical building facade. It features a central entrance with a triangular pediment above it, flanked by two columns. The facade is supported by a series of columns, and the entire structure is topped with a decorative cornice.

Set in stone, 1992

A hand-drawn diagram of a circuit board. It features a central vertical line with two horizontal lines extending from its top and bottom. The top horizontal line has a small triangle at its left end. To the right of the central line, there is a rectangular component with a small '1' at its top edge and a '2' at its bottom edge. The entire diagram is drawn in black ink on a white background.

1. **What is the name of the first person you met in the room?**

Logic circuit for $f_1 = \overline{a_1} \cdot \overline{a_2} \cdot \overline{a_3} \cdot \overline{a_4}$

of view for magnitude compared.

Subtractor ::

Subtractor circuits takes the binary number and input and subtract one binary number i/p from the other binary numbers i/p.

- It gives out two outputs, difference and borrow.
- These are two types of subtractors:

1. Half Subtractor
2. Full Subtractor

1. Half Subtractor:

The Half Subtractor is a combinational circuit which is used to perform subtraction of two bits. It has 2 inputs A (minuend) and B (subtrahend) and two outputs. Difference and borrow. The logic symbol and Truth Table are shown below.

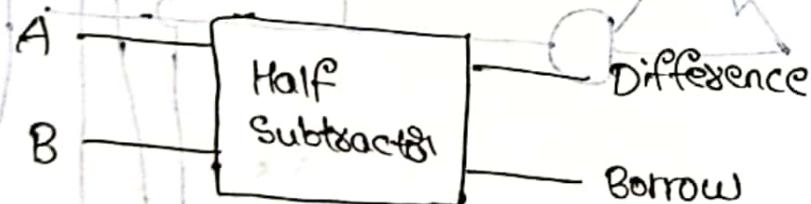


Fig: Logic symbol of Half Subtractor.

Truth Table of Half Subtractor:

Input		Output	
A	B	D	B
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

K-Map for Difference (D)

A\B	0	1
0	0	1
1	1	0

Boolean expression for difference

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

$$D = A \oplus B$$

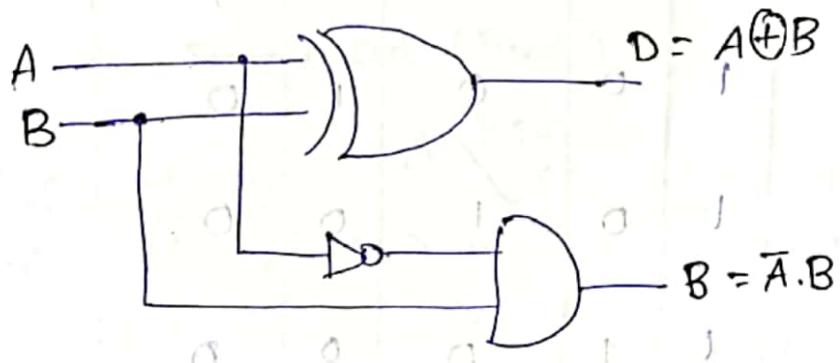
K-Map for Borrow (B)

A\B	0	1
0	0	1
1	1	0

Boolean expression for borrow

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

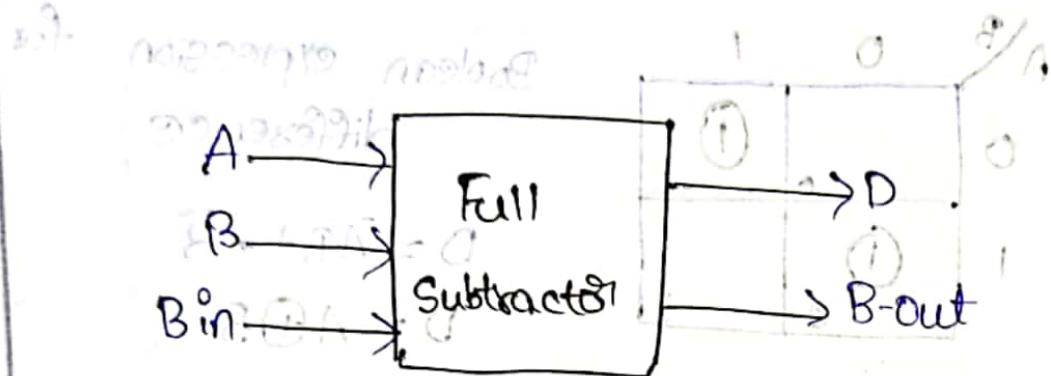
Logic circuit for Half Subtractor:



2. Full Subtractor:

A Full Subtractor is a Combinational circuit that performs subtraction involving three bits, namely A (minuend), B (subtrahend) and Bin (Borrow-in) as inputs and it produces two outputs : difference (D) and borrow out (B out).

Logic Symbol for Full Subtractor:



Truth Table for Full Subtractor:

A	B	B _{in}	D	B _{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	0
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 (D)

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

		00	01	11	10
		0	1	3	2
		1	1	1	1
A	B Bin	0	1	3	2
0	0	1	3	2	
1	1	1	1	1	

$$D = \bar{A}\bar{B}B_{in} + \bar{A}B\bar{B}_{in} + A\bar{B}\bar{B}_{in} + AB{B}_{in}$$

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

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

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

$$\therefore D = A \oplus B \oplus B_{in}$$

K-Map for Borrow-out (B-out)

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

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

Boolean expression for Borrow out (B-out) :-

$$B_{out} = \bar{A}B_{in} + \bar{A}B + B{B}_{in}$$

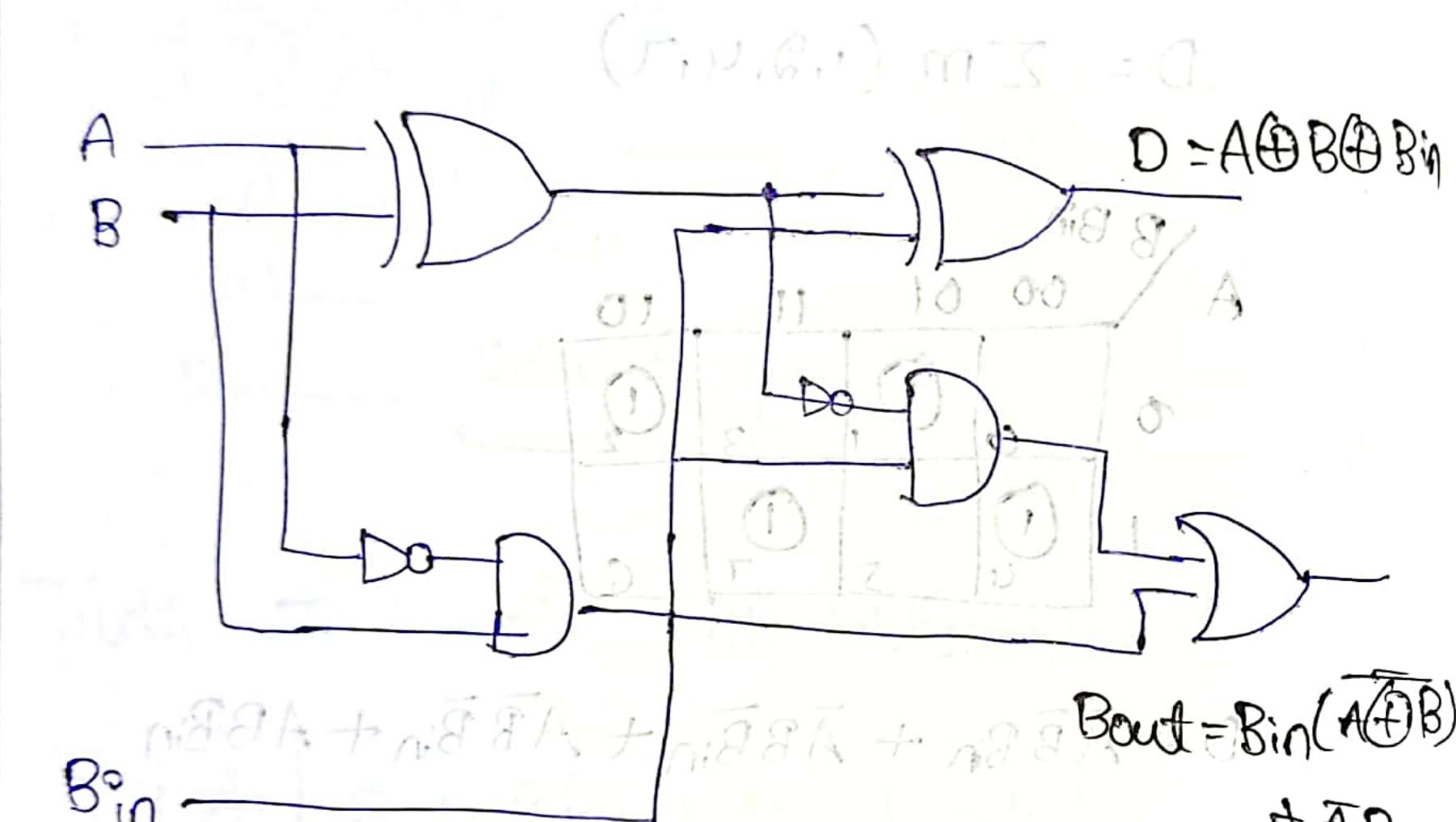
(or)

$$B_{out} = \bar{A}\bar{B}B_{in} + \bar{A}B + AB{B}_{in}$$

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

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

Logic circuit for Full Subtractor:



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