Unit-1

Number System, Boolean Algebra & Logic Gates

Number system:-

Number system is a base for counting various items. The modern computers (or) Digital system communicate operates with Binary numbers which use only digits 0 & 1.

To define any Number system, the following specifications are considered.

⇒ Total no:of digits or symbols

⇒ Based on number system which such as 2, 8, 10, 16

⇒ Here base indicates the total no:of digits available in number system.

Classification:-

There are 4 types. They are,

* Decimal Number System (base 10)
* Binary Number system (base 2)
* Octal Number system (base 8)
* Hexa Number system (base 16)

Decimal Number Systems:-

The total no:of digits (or) symbol present in a decimal number system are 10.

Since the base (or) Radix of the decimal number system is 10. The digits (or) symbols in decimal number system starts with (10).

Ex:- $567_{(10)}$

From the above example we can say that the decimal number 567 can also be written as $(567)_{10}$, where 10 is base. i.e $(567)_{10}$

$5\times10^2+6\times10^1+7\times10^0$.

∴ The decimal number system is the positional weighted system and its weight is expressing in power of 10.

In floating point representation the left side digits have positive powers and right side digits have negative powers.

Ex:- 567.87

$5\times10^2+6\times10^1+7\times10^0+8\times10^{-1}+7\times10^{-2}$

Binary Number system:-

The total no:of digits or symbols are represented in binary numbers (or) are 2.

Since the base (or) Radix of binary number system is 2.

The elements starts from zero to one.

Ex:- 1001

i.e; $1\times2^3+0\times2^2+0\times2^1+1\times2^0$

$= 8+0+0+1$

$= (9)_2$

* 1001.01

$\Rightarrow 1\times2^3+0\times2^2+0\times2^1+1\times2^0+0\times2^{-1}+1\times2^{-2}$

$= 8+0+0+1+0+0.25$

$= 9.25$.

⇒ In a Binary number system each binary digit is known as bit.

⇒ In Binary number system the weight is expressed as powers of 2. for an n-bit number of combinations are 2^n.

Octal Number system:-

Total no:of digits or symbols present in a octal number system is 8. Since the base/radix of the octal number system is 8 i.e; (0-7).

Ex:- $\begin{array}{ccc} 5 & 6 & 7 \\ \downarrow & \downarrow & \downarrow \\ 8^2 & 8^1 & 8^0 \end{array}$

$\Rightarrow 5\times8^2+6\times8^1+7\times8^1$

In floating point:- 567.12

$\Rightarrow 5\times8^2+6\times8^1+7\times8^1+1\times8^{-1}+2\times8^{-2}$

In octal Number system the weight is expressed in powers of 8.

Hexa Decimal Number system:-

The total number of digits are symbols present in a hexa decimal number systems are 16.

Since the base/radix of hexa decimal number system is 16.

Ex:- $\begin{array}{ccc} 4 & A & 5 \\ \downarrow & \downarrow & \downarrow \\ 16^2 & 16^1 & 16^0 \end{array}$

i.e; $4\times16^2+A\times16^1+5\times16^0$.

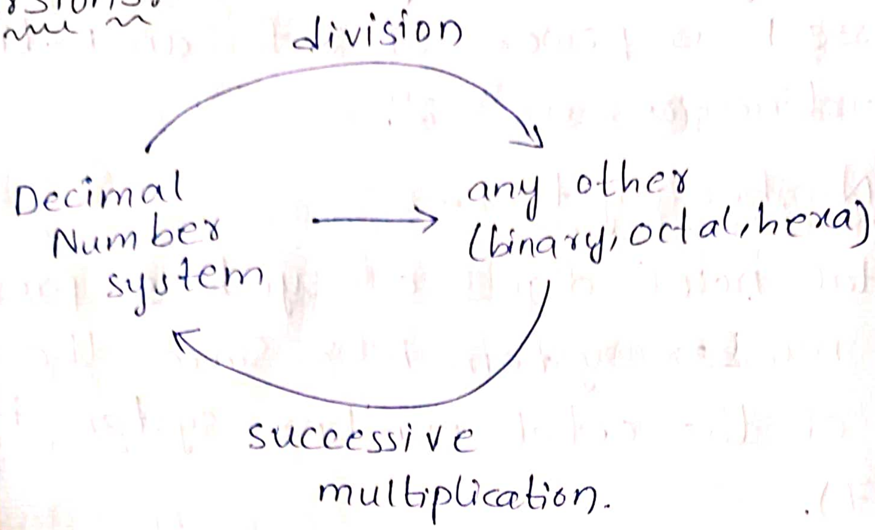
In floating point:-

$\begin{array}{cccc} 4 & A & 5. & 8 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 16^2 & 16^1 & 16^0 & 16^{-1} \end{array}$

i.e; $4\times16^2+A\times16^1+5\times16^0+8\times16^{-1}$

In hexa decimal number system the weight is expressed in powers of 16.

Conversions:-



* Decimal Number system to any other number system.

(i) $(40)_{10}$ (ii) $(18)_{10}$ (iii) $(56)_{10}$

Decimal to Binary system:-

(i)
2 | 40
2 | 20 - 0
2 | 10 - 0
2 | 5 - 0
2 | 2 - 1
1 - 0

$(40)_{10} = (101000)_2$

(ii)
2 | 18
2 | 9 - 0
2 | 4 - 1
2 | 2 - 0
1 - 0

$\Rightarrow (10010)_2$

(iii)
2 | 56
2 | 28 - 0
2 | 14 - 0
2 | 7 - 0
2 | 3 - 1
1 - 1

$\Rightarrow (111000)_2$

Decimal to octal number system:-

(i) $(40)_{10}$
8 | 40
5 - 0

$\Rightarrow (50)_8$

(ii) $(214)_{10}$
8 | 214
8 | 26 - 6
3 - 2

$\Rightarrow (326)_8$

(iii) $(3509)_{10}$
8 | 3509
8 | 438 - 5
8 | 54 - 6
6 - 6

$\Rightarrow (6665)_8$

Decimal to Hexa decimal number system:-

(i) $(40)_{10}$
16 | 40
2 - 8

$(28)_{16}$

(ii) $(214)_{10}$
16 | 214
13 - 6
D = 13

$(D6)_{16}$

(iii) $(3509)_{10}$
16 | 3509
16 | 219 - 5
3 - 11

$\Rightarrow (DB5)_{16}$

Binary to Decimal number system:-

(i) $(1010)_2$

$$\begin{array}{cccc} 1 & 0 & 1 & 0 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 2^3 & 2^2 & 2^1 & 2^0 \end{array}$$

$\Rightarrow 1\times2^3+0\times2^2+1\times2^1+0\times2^0$

$\Rightarrow 8+0+2+0$

$\Rightarrow (10)_{10}$

(ii) $(010110000)_2$

$\Rightarrow 0\times2^8+1\times2^7+0\times2^6+1\times2^5+1\times2^4+0\times2^3+0\times2^2+0\times2^1+0$

$\Rightarrow 2^7+2^5+2^4 \Rightarrow 128+32+16$

$\Rightarrow (176)_{10}$.

Binary to octal number system:-

(i) $(010110001)_2$

$(\underline{010}\,\underline{110}\,\underline{001})_2$

$(261)_8$

(ii) $(010011010)_2$

$(\underline{010}\,\underline{011}\,\underline{010})_2$

$(232)_8$

	2^2	2^1	2^0
	4	2	1
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Binary to Hexa decimal number system:-

(i) $(000010110001)_2$

$(\underset{0}{\underline{0000}}\,\underset{11}{\underline{1011}}\,\underset{1}{\underline{0001}})_2$

$\Rightarrow (0B1)_{16}$

(ii) $(1000101)_2$

$(\underset{4}{\underline{100}}\,\underset{5}{\underline{0101}})_2$

$\Rightarrow (45)_{16}$

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

* Octal to Decimal number system:-

(i) $(324)_8$

i.e; $3\times8^2+2\times8^1+4\times8^0$

$\Rightarrow 192+16+4$

$\Rightarrow (212)_{10}$

(ii) $(473)_8$

$\Rightarrow 4\times8^2+7\times8^1+3\times8^0$

$\Rightarrow 4\times64+56+3$

$\Rightarrow (315)_{10}$

* Octal to Binary number system:-

(i) $(473)_8$

$\Rightarrow (473)_8$

$\Rightarrow (100111011)_2$

(ii) $(273)_8$

$\Rightarrow (273)_8$

$\Rightarrow (010111011)_2$

* Octal to hexa decimal system:-

(i) $(324)_8$ (ii) $(526)_8$ (iii) $(273)_8$ (iv) $(5327)_8$

(i) $(324)_8 = 011\,010100 \Rightarrow (0D4)_{16}$

(ii) $(526)_8 = 101010110 \Rightarrow (156)_{16}$

(iii) $(273)_8 = 010111011 \Rightarrow (0BB)_{16}$

(iv) $(5327)_8 = 101011010111 \Rightarrow (AD7)_{16}$.

* Hexa decimal to decimal system:-

(i) $(56)_{16}$

$\Rightarrow 5\times16^1+6\times16^0$

$= 80+6$

$= (86)_{10}$

(ii) $(3AD)_{16}$

$\Rightarrow 3\times16^2+A\times16^1+D\times16^0$

$= 3\times256+10\times16+13\times1$

$= 768+160+13$

$= (941)_{10}$.

Hexa decimal to Binary system:-

(i) $(FBC)_{16} \Rightarrow (111110111100)_2$

(ii) $(AF25)_{16} = (1010111100100101)_2$

* Hexa decimal to Octal number system:-

(i) $(FBC)_{16} = 111110111100 \Rightarrow (7674)_8$

(ii) $(ABC)_{16} = 101010111100 \Rightarrow (5274)_8$

Fractional point Conversions:-

* Decimal to Binary conversion:-

Ex:- $(0.8125)_{10}$

Division (LCM)

$8125 \times 2 \Rightarrow 1.625$ 1

$0.625 \times 2 \Rightarrow 1.25$ 1

$0.25 \times 2 \Rightarrow 0.5$ 0

$0.5 \times 2 \Rightarrow 1.0$ 1

$(0.1101)_2$

* Decimal to octal:-

Ex:- $(0.640625)_{10}$

$0.640625 \times 8 \Rightarrow 5.125$

$0.125 \times 8 \Rightarrow 1.00$

$\Rightarrow (0.51)_8$

* Decimal to Hexa-decimal:-

Ex:- $(22.64)_{10}$

$16 \,|\, 22$

$\quad 1 - 6$

(16)

$0.64 \times 16 \Rightarrow 10.24$

$0.24 \times 16 \Rightarrow 3.84$

$0.84 \times 16 \Rightarrow 13.44$

$0.44 \times 16 \Rightarrow 7.04$

$0.04 \times 16 \Rightarrow 0.64$

$\Rightarrow (16.A3070)_{16}$

Binary conversions:-

* Binary conversion addition
* Binary subtraction
* Binary multiplication
* Binary division.

Fractional Part Conversions:-

* Binary Addition:-

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

2 = 10
3 = 11

Ex:-

(i)
```
  1010
  1110
 -----
 11000
```

(ii)
```
  101011
  111000
 -------
 1100011
```

* Binary subtraction:-

A	B	A−B	barrow
0	0	0	0
0	1	−1	1
1	0	1	0
1	1	0	0

Ex:- (i)
```
 1110
 0011
 ----
 1011
```

(ii)
```
 111001
 001010
 ------
 101111
```

* Binary multiplication:-

A	B	A·B
0	0	0
0	1	0
1	0	0
1	1	1

Ex:- (i) 1101 × 110

```
 1101 × 110
 ----------
    0000
   1101×
  1101××
 ----------
 100111 0
```

(ii) 1110 × 101

```
 1110 × 101
 ----------
     1110
    0000×
   1110××
 ----------
  1000110
```

Binary division:-

Ex:- 101101 ÷ 110

```
        neglect
 110) 101101 (111.1
      110
      ----
      0010
       110
      ----
      0001
        110
      ----
        110
        110
      ----
          0
```

101110 ÷ 101

```
 101) 101110 (110
      101
      ----
      0101
       101
      ----
        00
         0
      ----
         0
```

Complements:-

1's - 2's, 9's - 10's.

Ex:- $(10010)_2$

```
        10010
 1's →  01101  → compliment of this
            1  → add 1
 2's →  01110
```

```
        562
          1
 2's →  563
```

2's compliment is done with any number ~~can add~~ ~~e to~~

1's compliment is only with 0 and 1

Ex:- 9's - 10's

(i) 468

```
        999
    (-) 468
 9's →  531

        531
          1  — add 1
 10's → 532
```

(ii) 3524

```
        9999
        3524
 9's →  6475

        6475
           1
 10's → 6476
```

(iii) 6234

```
        9999
        6234
 9's →  3765

        3765
           1
 10's → 3766
```

Other binary codes:-

(1) BCD code / 8421 code

(2) Excess-3 code (XS-3)

(3) Gray code

(1) BCD code

	8 4 2 1 (2^3 2^2 2^1 2^0)
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1
10	1 0 1 0
11	1 0 1 1
12	1 1 0 0
13	1 1 0 1
14	1 1 1 0
15	1 1 1 1

(2) XS-3 code { add 3 to decimal no. (0-9)

	8 4 2 1
0+3=3	0 0 1 1
1+3=4	0 1 0 0
2+3=5	0 1 0 1
3+3=6	0 1 1 0
4+3=7	0 1 1 1
5+3=8	1 0 0 0
6+3=9	1 0 0 1
7+3=10	1 0 1 0
8+3=11	1 0 1 1
9+3=12	1 1 0 0

(3) Gray Code:-

(1) Binary to Gray code

(2) Gray code to Binary

if we get different → '1'

same → '0'.

* The gray code is a non-weighted code and it is not suitable for Arithmetic operations.
* The only reason for the popularity of gray code is it is easy to convert binary form.

* Gray code is also called as "Unit distance code".
* There is only 1 bit change b/w the two consecutive numbers.

Q:- D-(0-09).

Binary to Gray conversion:-

0 0 0 0

↓ ↓ ↓ ↓

0 0 0 0

⇒ In this the first bit is constant, and adjacent bits were ~~added~~ compared simultaneously.

⇒ If, we have different bits then ~~the~~ '1' is the answer.

⇒ If, we have same bits the '0' is the answer.

different - 1

same - 0

Convert the decimal number to Gray code (0-9)

0:- 0 0 0 0 → Binary

↓ ↓ ↓ ↓

$(0\ 0\ 0\ 0)_{Gray}$

1:- 0 0 0 1 :- binary

↓ ↓ ↓ ↓

$(0\ 0\ 0\ 1)_G$

2:- 0 0 1 0

↓ ↓ ↓ ↓

constant $(0\ 0\ 1\ 1)_G$

3:- 0 0 1 1

↓ ↓ ↓ ↓

$(0\ 0\ 1\ 0)_G$

4:- 0 1 0 0

↓ ↓ ↓ ↓

$(0\ 1\ 1\ 0)_G$

5:- 0 1 0 1

↓ ↓ ↓ ↓

$(0\ 1\ 1\ 1)_G$

6:- 0 1 1 0

↓ ↓ ↓ ↓

$(0\ 1\ 0\ 1)_G$

7:- 0 1 1 1

↓ ↓ ↓ ↓

$(0\ 1\ 0\ 0)_G$

8:- 1 0 0 0

↓ ↓ ↓ ↓

$(1\ 1\ 0\ 0)_G$

9:- 1 0 0 1

↓ ↓ ↓ ↓

$(1\ 1\ 0\ 1)_G$

Convert the gray to binary (0-15):-

⇒ In this the first bit is constant, the resultant answer ~~added~~ compared to the next bit (second bit).

⇒ If we have different bits then '1' is the answer.

⇒ If we have same bits '0' is the answer.

different - 1

same - 0.

0:- 0 0 0 0 - Gray

$(0\,0\,0\,0)_B$

1:- 0 0 0 1

$(0\,0\,0\,1)_B$

2:- 0 0 1 0

$(0\,0\,1\,1)_B$

3:- 0 0 1 1

$(0\,0\,1\,0)_B$

4:- 0 1 0 0

$(0\,1\,1\,1)_B$

5:- 0 1 0 1

$(0\,1\,1\,0)_B$

6:- 0 1 1 0

$(0\,1\,0\,0)_B$

7:- 0 1 1 1

$(0\,1\,0\,1)_B$

8:- 1 0 0 0

$(1\,1\,1\,1)_B$

9:- 1 0 0 1

$(1\,1\,1\,0)_B$

10:- 1 0 1 0

$(1\,1\,0\,0)_B$

11:- 1 0 1 1

$(1\,1\,0\,1)_B$

12:- 1 1 0 0

$(1\,0\,0\,0)_B$

13:- 1 1 0 1

$(1\,0\,0\,1)_B$

14:- 1 1 1 0

$(1\,0\,1\,1)_B$

15:- 1 1 1 1

$(1\,0\,1\,0$

Binary to Gray code:-

* $(11011001)_B$

1 1 0 1 1 0 0 1

$(10110101)_G$

* $(00100110)_B$

0 0 1 0 0 1 1 0

$(00110101)_G$

* $(11100111)_B$

1 1 1 0 0 1 1 1

$(10010100)_G$

* $(00001111)_B$

$(00001000)_G$

* $(10111000)_B$

$(11100100)_G$

Gray = binary:-

* $(10110101)_G$ → $(11011001)_B$

* $(11100111)_G$ → $(10111010)_B$

* $(10001000)_G$ → $(11110000)_B$

* $(11111010)_G$ → $(10101100)_B$

* $(00001110)_G$ → $(00001011)_B$.

Binary one's compliment for subtraction:-

⇒ Find out the one's compliment of suptrahend and added to minuend (largest no). If there is a carry bring that carry and add to the LSB (lost significant bit).

⇒ Look at the MSB (most significant bit). If it is zero the result is positive and it is in true binary form.

⇒ If the MSB is 1 (whether irrespective of carry the resut is negative and it is in 1's compliment form)

⇒ Take one's compliment of result to get binary value.

Ex:- Add 27.50 from 68.75

Here the minuend is 68.75 / greater number
subtrahend (lesser number.) = 27.50.

2 | 27
2 | 13 - 1
2 | 6 - 1
2 | 3 - 0
  | 1 - 1

(11011)

2 | 68
2 | 34 - 0
2 | 17 - 0
2 | 8 - 1
2 | 4 - 0
2 | 2 - 0

100100

* Add 27.50 from 68.75.

68.75 - 01000100.1100

27.50 - 00011011.1000

1's compliment:-

→ 27.50 - 11100100.0111

68.75 - 01000100.1100

(+)

$\boxed{1}$00101001.0011

carrier: 1

00101001.0100

Ex: 52 - 20.

2 | 52
2 | 26 - 0
2 | 13 - 0
2 | 6 - 1
2 | 3 - 0
2 | 1 - 1

2 | 20
2 | 10 - 0
2 | 5 - 0
2 | 2 - 1
1 - 0

(110100) (10100)

20 - 0 0 1 1 0 1 1

52 - 1 1 0 1 0 0

1 1 1

001001

* Binary subtraction for 2's compliment.

⇒ Find out the 2's compliment of subtrahend and add which through the minuend.

⇒ If there is a carry ignore it. And look at the MSB of the sum.

⇒ If MSB is zero, the result is positive and it is the true binary number.

⇒ If the MSB is "one" (whether there is a carry or no carry). the result is negative and it is in 2's compliment form.

⇒ Take its 2's compliment find out the true binary value.

Ex:- subtract 14 from 46 using 8 bit 2's compliment method. (1446-answer)

2	14	
2	7	0
2	3	1
	1	1

$(1110)_B$

2	46	
2	23	0
2	11	1
2	5	1
2	2	1
	1	0

$(101110)_B$

14 - 00001110

46 - 00101110

⇒ 14 - 11110001

```
        1 1
 11110010
 00101110
---------
1 1010000
```

BCD Addition:-

⇒ Convert a given decimal number into their equivalent BCD codes.

⇒ Add the BCD numbers using the rules of binary addition.

⇒ Check the result, if it is valid (≤ 9) no correction is needed.

⇒ If the result is invalid add 0110 i.e '6' to the four bits sum to get the correct result.

Ex:- perform the BCD addition for 25 and 13.

BCD - 25

```
          25
         /  \
      0010  0101
13 -  0001  0011
(+)
      ---------
      0011  1000
       ↓     ↓
       3     8
```

∵ 3 is <9, it is valid.

∵ 8 is <9, it is valid.

∴ The total sum is '38'.

& 91 and 81

BCD - 91

```
            91
           /  \
       1001    0001
81 -   1000    0001
       -------------
      10001    0010
add '6' ← 0110   ↓
       -------            2
 --000 10111
         ↓
        17
```

91
81
172

∴ '2' is ≤ 9, it is valid

∴ The total sum is 172.

BCD subtraction:-

⇒ The subtraction of BCD numbers can be performed by using 9's & 10's compliment methods.

⇒ find the 9's & 10's compliment of the given -ve number.

⇒ Add 2 numbers to the BCD addition.

⇒ If the carry is generated add that carry to the result for correction.

Ex:- perform the subtraction in BCD by the 9's compliment method. 305.5 − 168.8

9's compliment of 168.8 = 999.9
(−) 168.8
831.1

⇒ 305.5
(+) 831.1

BCD:-

305.5 ⇒ 0011 0000 0101 . 0101

831.1 ⇒ 1000 0011 0001 . 0001

1011 0011 0110 . 0110

↓ ↓ ↓ ↓

11 3 6 6

Here 11>9: 1011

add 6: 0110

①0001 0011 0110 . 0110

1

0001 0011 0110 . 0111

* Note:-

While doing subtraction by using BCD if th is a barrow from the next group then 6 is subtracted from the difference.

⇒ If there is no barrow from the next higher group then no correction is require

Ex:- 679.6 − 885.9

9's compliment of 885.9 ⇒ 999.9

(−) 885.9

114.0

BCD:-

→ 679.6 ⇒ 0110 0111 1001 . 0110

114.0 ⇒ 0001 0001 0100 . 0000

0111 1000 1101 . 0110

↓ 7, ↓ 8, ↓ 13, ↓ 6

~~0111 1000 0110 0110~~

~~0000 0011~~

~~0110~~

0111 1001 0011 . 0110

7 9 3 6.

Signed binary numbers:-

⇒ If the MSB bit is '0' then the resulting value is '+ve', and for remaining have to perform binary addition.

⇒ If the MSB bit is '1' then the resulting value is '−ve', and remaining have to perform binary addition.

Ex:- 0 1 0 1 0 0 1

$2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

32 + 0 + 8 + 1 ⇒ 41

Ex:- 1 1 0 1 0 0 1

$2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

⇒ −41

0101110

+ve 2^5 2^4 2^3 2^2 2^1 2^0

$\Rightarrow 1\times2^5 + 0\times2^4 + 1\times2^3 + 1\times2^2 + 1\times2^1 + 0\times2^0$

$\Rightarrow 32 + 0 + 8 + 4 + 2 + 0$

$\Rightarrow +46$.

Digital logic Gates:-

⇒ Logic gate is a digital circuit which has one or more inputs & only 1 output. The i/p's or o/p's are logic '0' or '1'.

⇒ These logic gates are basic building blocks for digital system.

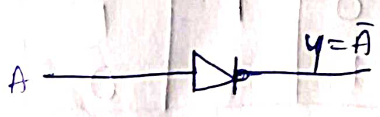
Types of logic gates:-

(1) NOT
(2) AND } – basic gates
(3) OR

(4) NAND } – universal gates
(5) NOR

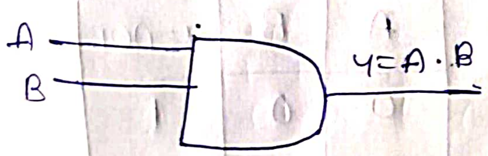
(6) X-OR } – special gates
(7) X-NOR

NOT:-



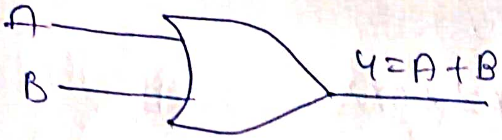
A	$Y=\bar{A}$
0	1
1	0

AND:-



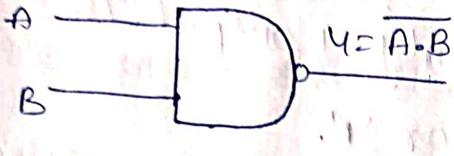
A	B	$Y=A\cdot B$
0	0	0
1	0	0
0	1	0
1	1	1

OR :-



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

NAND :-

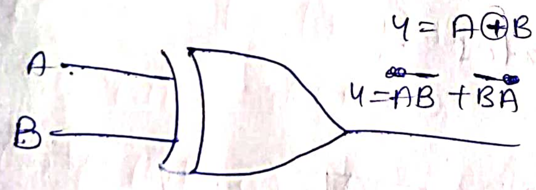


A	B	Y= $\overline{A} \cdot \overline{B}$
0	0	1
1	0	1
0	1	1
1	1	0

NOR :-

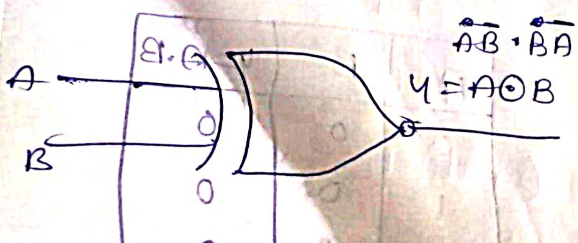
A	B	Y= $\overline{A+B}$
0	0	1
1	0	0
0	1	0
1	1	0

X-OR :- (odd is out)



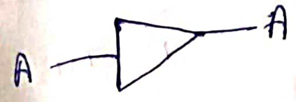
A	B	Y=A⊕B
0	0	0
1	0	1
0	1	1
1	1	0

X-NOR :- (even is out)



A	B	Y=A⊙B
0	0	
0	1	1
1	0	0
1	1	0

Buffer:



0 → 0

1 → 1

Boolean Algebra:-

Rules of Boolean Algebra.

(1) $A+0=A$

(2) $A+1=1$

(3) $A\cdot 0=0$

(4) $A\cdot 1=A$

(5) $A+A=A$

(6) $A+\bar{A}=1$

(7) $A\cdot A=A$

(8) $A\cdot\bar{A}=0$

(9) $\bar{\bar{A}}=A$

(10) $A+AB=A$

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

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

Laws and properties of Boolean Algebra:-

Name	AND form	OR form
(1) Identity law	$1\cdot A=A$	$0+A=A$
(2) Null law	$0\cdot A=0$	$1+A=1$
(3) ~~Idepe~~ Idempotent	$A\cdot A=A$	$A+A=A$
(4) Inverse law	$A\cdot\bar{A}=0$	$A+\bar{A}=1$
(5) Commutative law	$AB=BA$	$A+B=B+A$
(6) Associative law	$(AB)C=A(BC)$	$(A+B)+C=A+(B+C)$

(7) Distributive law $A+BC=(A+B)(A+C)$ $\quad A(B+C) = AB+BC$

(8) Absorption law $A(A+B)=A$ $\quad A+AB=A$

(9) Demorgan's law $\overline{AB} = \bar{A}+\bar{B}$ $\quad \overline{A+B} = \bar{A}\cdot\bar{B}$

Simplify the expressions:-

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

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

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

$= (\bar{A}+B)(A\bar{D}+B\bar{D}+D\bar{D})$ $\quad \{\because D\bar{D}=0\}$

$= (\bar{A}+B)(A\bar{D}+B\bar{D}+0)$

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

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

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

$= (0+\bar{A}B+BA+B)\bar{D}$

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

$= (1+1)B\bar{D}$

$= (1)B\bar{D} \Rightarrow B\bar{D}$

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

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

$= AB+0+A\bar{B}\bar{C}$

$= A(B+\bar{B}\bar{C})$ $\quad$ {by demorgan's law}

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

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

Reduce the expression:-

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

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

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

(2) $f = (\overline{A+\overline{BC}})(A\bar{B}+ABC)$

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

$= (\bar{A}\cdot\overline{\overline{BC}})(A\bar{B}+ABC)$

$= (\bar{A}\cdot BC)(A\bar{B}+ABC)$

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

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

(3) $f = (B+BC)(B+\bar{B}\bar{C})(B+D)$

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

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

$= (BB+B.\bar{B}C)(B+D)$ $\{\because B+D = B+\bar{B}D\}$

$= (B+0C)(B+D)$

$= (B+0)(B+\bar{B}D)$

$= (B.B+B.\bar{B}D) \Rightarrow B+0D \Rightarrow B$

Duality:-

A Dual of Boolean function is obtained by interchanging the logical and operator with logical or operator and interchanging 0's with 1's.

G

Given expression

(1) $\bar{0} = 1$

(2) $0 \cdot 1 = 0$

(3) $0 \cdot 0 = 0$

(4) $1 \cdot 1 = 1$

(5) $A \cdot 0 = 0$

(6) $A \cdot 1 = A$

(7) $A \cdot A = A$

(8) $A \cdot \bar{A} = 0$

(9) $A \cdot B = BA$

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

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

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

(13) $A(AB) = AB$

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

$\bar{1} = 0$

$1+0 = 1$

$1+1 = 1$

$0+0 = 0$

$A+1 = 1$

$A+0 = A$

$A+A = A$

$A+\bar{A} = 1$

$A+B = B+A$

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

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

$A+AB = A$

$A+A+B = A+B$

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

Reduce the expression,

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

(1) $f = A\left[B+\bar{C}\,(\overline{AB+A\bar{C}})\right]$

(2) $f = (B+BC)(B+\bar{B}C)(B+D)$

(3) Show that $A\bar{B}C + B + B\bar{D} + AB\bar{D} + \bar{A}C = B+C$.

(1) $f = A\left[B+\bar{C}\,(\overline{AB+A\bar{C}})\right]$

$= A\left[B+\bar{C}\,(\bar{A}+\bar{B})(\bar{A}+\bar{\bar{C}})\right]$

$= A\left[B+(\bar{A}+\bar{B}) \cdot \bar{C}(\bar{A}+\bar{\bar{C}})\right]$

$= A\left[B+(\bar{A}+\bar{B}) \cdot (\bar{C}\bar{A}+0)\right]$

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

$= A\left[B+(\bar{A}+\bar{B})(\overline{C+A})\right] \Rightarrow \left[AB+(0)(0)\right] \Rightarrow AB$

(2) $f = (B+BC)(B+\bar{B}C)(B+D)$

$= (B+BC)\left[BB + BD + \bar{B}C \cdot B + \bar{B}C \cdot D\right]$

$= (B+BC)\left[B + B \cdot D + 0 + \bar{B}C \cdot D\right]$

$= (B+BC)\left[B + BD + \bar{B}CD\right]$

$= BB + BBD + B\bar{B}CD + BBC + BCBD + BC\bar{B}CD$
$\underset{0}{\downarrow}$

$= [B+BD] + [BC+BCD]$

$= B + BC$

$= B(\underset{\substack{\downarrow \\ 1}}{1+C}) \Rightarrow B(1) \Rightarrow B$

Duali:-

(1) $\bar{A}B + \bar{A}B\bar{C} + \bar{A}BCD + \bar{A}B\bar{C}\bar{D}E$ $\Rightarrow$ if eqn is SOP form it be NAND form

(2) $\bar{X}YZ + \bar{X}Y\bar{Z} + X\bar{Y}\bar{Z} + X\bar{Y}Z$ $\Rightarrow$ if eqn is POS form it be NOR form

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

Implementation by using NAND Gate:-

Boolean functions:-

⇒ Boolean functions are also known functions

⇒ These are constructed by connecting the Bolean constants & variables with Boolean operations.

⇒ The boolean expressions are used to describe the boolean functions.

Ex:- If the boolean expression $\bar{A}B+\bar{B}C$ is used to describe the function 'f'. Then boolean function can be written as based on structure of Boolean expression. It can be catagorized into 2 forms

(1) Sum of product (SOP/normal form)

(2) Product of sum (POS

(1) Sum of product:- This is a group of product terms and summed together.

Ex:- $f(A,B,C) = \underbrace{\bar{A}B+\bar{B}C}_{\text{variable/literals.}}$ → Product terms

⇒ It is also called as disjunctive normal form.

(2) Product of sum:- It is a group of sum terms and multiplied together.

Ex:- $f(A,B,C) = \underbrace{(\bar{A}+\bar{B})(A+B)}_{\text{sum of terms}}$

⇒ It is also called as conjuctive normal form.

Canonical form :-

These are the spl case of SOP & POS forms.

⇒ These r also known as standard SOP form & POS form.

⇒ Standard SOP form is also called as disjunctive canonical form.

⇒ Each product term contains all the variables of the function.

Let us consider a boolean function $f(A,B,C) = (\bar{A}B + \bar{B}C)$

$\bar{A}B(C+\bar{C}) + \bar{B}C(A+\bar{A})$ $\quad \left[\because A = 1,\ \bar{A} = 0\right]$

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

Decimal No	A, B, C	min. terms	f(A,B,C)
0	000	$\bar{A}\bar{B}\bar{C}$	0
1	001	$\bar{A}\bar{B}C$	1
2	010	$\bar{A}B\bar{C}$	1
3	011	$\bar{A}BC$	1
4	100	$A\bar{B}\bar{C}$	0
5	101	$A\bar{B}C$	1
6	110	$AB\bar{C}$	0
7	111	ABC	0

$f(A,B,C) = m_1 + m_2 + m_3 + m_5$

$\Sigma m(1,2,3,5)$

Standard POS form:-

⇒ It is also called as Conjunctive Canonic form.

⇒ Each sum term contains all the variabl of the function.

Max terms:-

⇒ Each individual term in standard POS is called max terms represented with 'M'.

⇒ ~~AAE~~ $\pi M(0, 1 \ldots\ldots)$.

Convert the given expression in standard SOP form.

$$f(A,B,C) = A + ABC$$

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

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

$$\Rightarrow \underset{4}{ABC} + \underset{5}{A\bar{B}C} + \underset{6}{AB\bar{C}} + \underset{7}{A\bar{B}\bar{C}}$$

$$\Rightarrow \pi M(4, 5, 6, 7)$$

* Convert the given expression in standard SOP form. $f(A,B,C) = AB + BC + \bar{A}C$

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

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

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

convert the given expression into standard pos form.

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

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

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

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

~~Convert~~ convert the given expression into stand. pos form. $f(A,B,C) = A(A+B+C)$

Compliments of Boolean function:-

$\Rightarrow (A+B)' = A'.B'$

$(A.B)' = A'+B'$

Ex: 1 :- $XY'Z + XY + YZ'$

$= (XY'Z)' + (XY)' + (YZ')'$

$= X' + Y'' + Z' + X + Y' + Y' + Z''$

$= X' + Y + Z' + X' + Y' + Y' + Z$

2. $ABC' + BC + A'B$

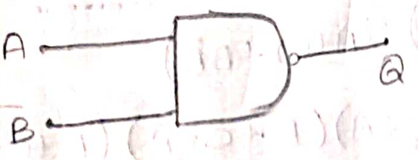
$= (ABC')' + (BC)' + (A'B)'$

$= A' + B' + C'' + B' + C' + A'' + B'$

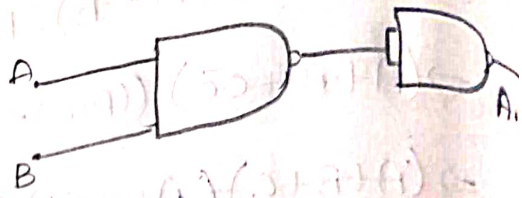
$= A' + B' + C + B' + C' + A + B'$

NAND GATE implem...

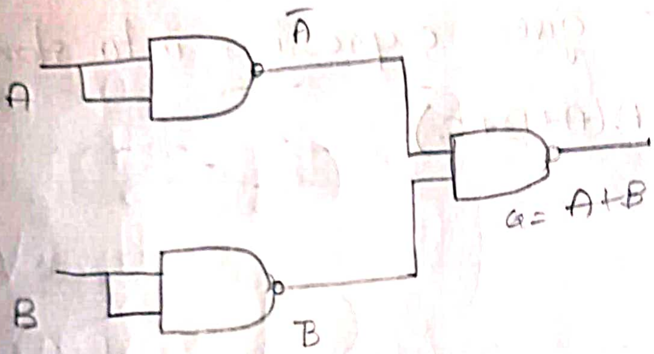
(1) NAND Gate symbol:-



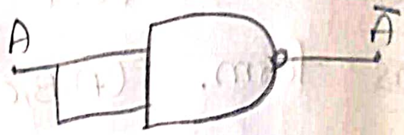
(2) AND Gate:-



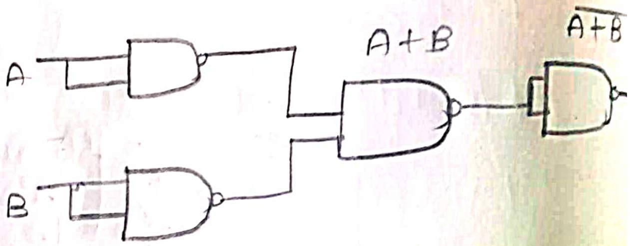
OR Gate:-



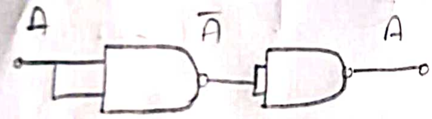
NOT Gate:-



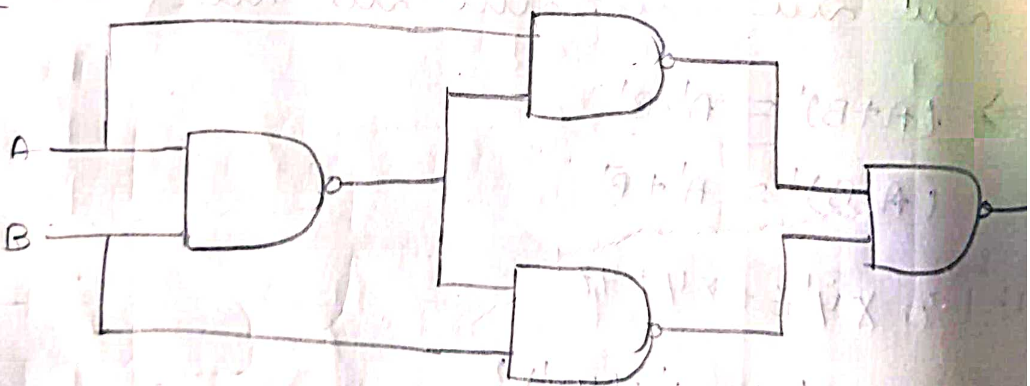
NOR Gate:-



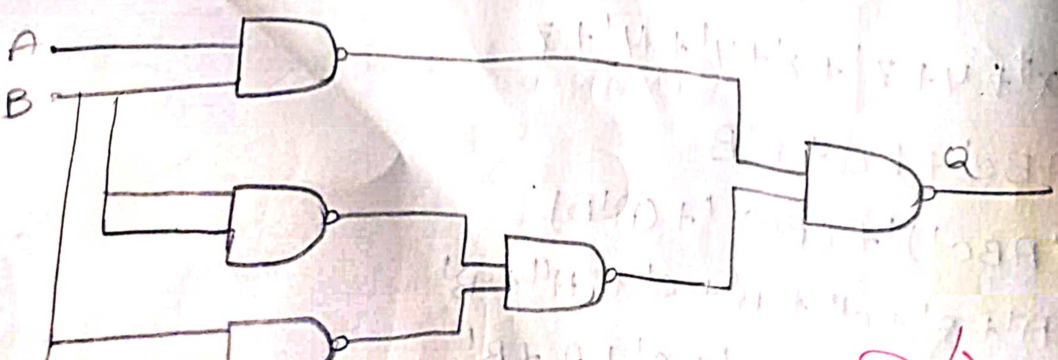
Buffer:-



EX-OR:-

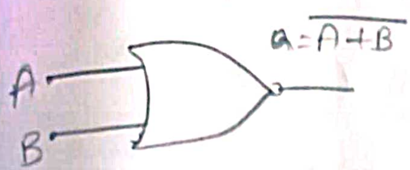


EX-NOR:-

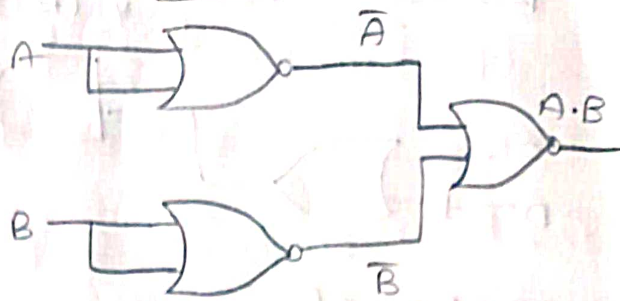


NOR Gate implementation by using gates:-

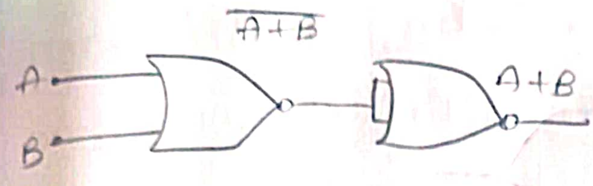
NOR Gate:-



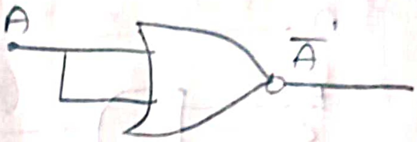
AND Gate:-



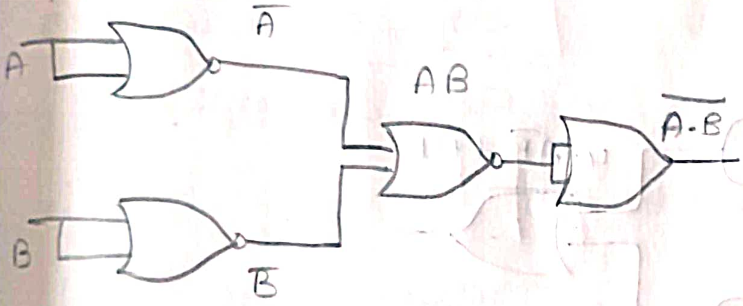
OR Gate:-



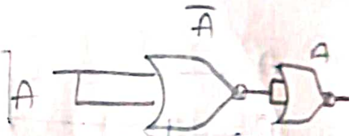
NOT Gate:-



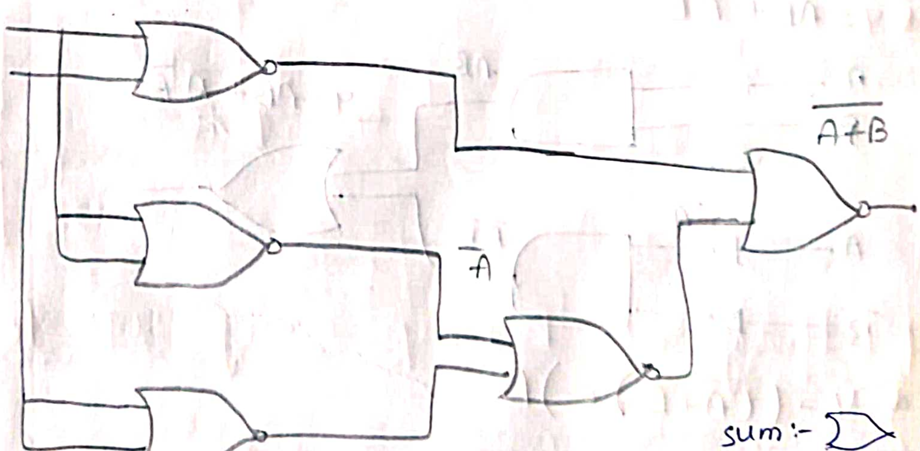
NAND Gate:-



Buffer:-



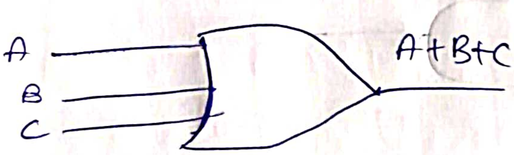
EX-OR Gate:-



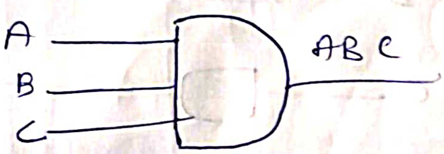
sum :-

product :-

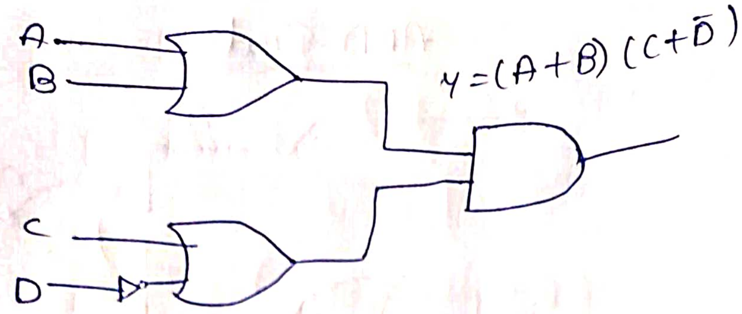
* $Y = A+B+C$



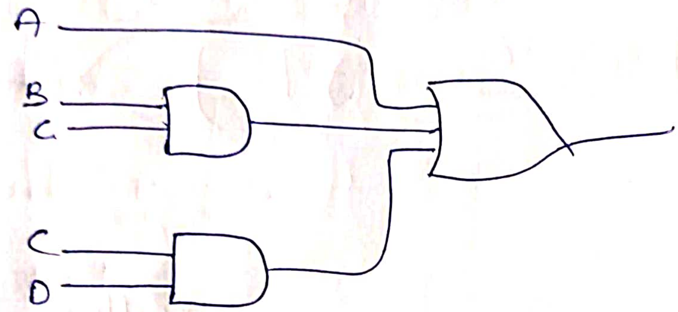
$Y = A \cdot B \cdot C$



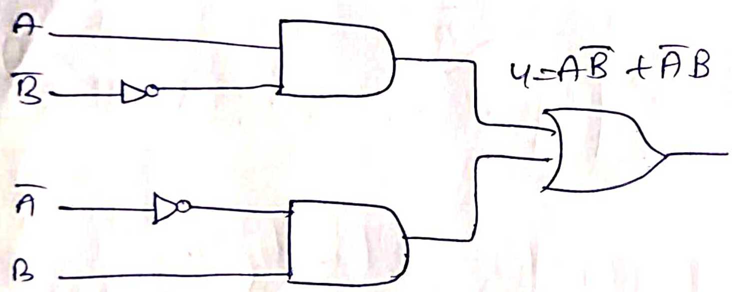
* $Y = (A+B)(C+\overline{D})$



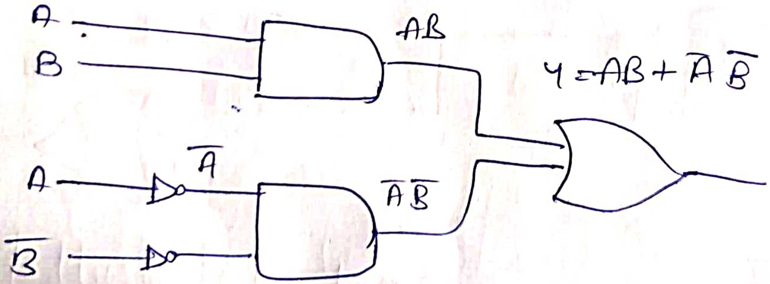
* $A + BC + CD$



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



* $Y = AB + \overline{A}\,\overline{B}$



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