Number Systems: There are four types of number systems.

1. Decimal Number system
2. Binary Number System
3. Octal Number system
4. Hexadecimal Number system

Decimal Number system :- Decimal number System is a base/radix 10 number system : 0 to 9 are the numbers available in decimal number system. The point of seperation of real and fractional parts is called decimal point. The positional weights of real part is $10^0, 10^1, 10^2$ --- and fractional part $10^{-1}, 10^{-2}, 10^{-3}$ - - - .

Eg: $(7943.69)_{10}$

$$\begin{array}{ccccccc} 7 & 9 & 4 & 3 & . & 6 & 9 \\ 10^3 & 10^2 & 10^1 & 10^0 & & 10^{-1} & 10^{-2} \end{array}$$

$7\times10^3 + 9\times10^2 + 4\times10^1 + 3\times10^0 + 6\times10^{-1} + 9\times10^{-2}$

$7000 + 900 + 40 + 3 + 0.6 + 0.09$

7943.69

Binary Number System :- Binary number system is a Base/radix 2 number system. 0 and 1 are only the values available in binary number system. The point of seperation of real and fractional part is called Binary point. The positional weights of real part is $2^0, 2^1, 2^2$ - - - and fractional part $2^{-1}, 2^{-2}, 2^{-3}$ - -

Eg: $(101.10)_2$

Octal Number system :- Octal number system is a Base/radix 8 number system. 0 to 7 are the numbers available in octal number system. The point of seperation of real and fractional part is called octal point. The positional weights of real part are $8^0, 8^1, 8^2$ --- and fractional part is $8^{-1}, 8^{-2}$ ----.

Eg:- $(273.12)_8$

Hexadecimal Number System: Hexadecimal Number system is a base/radix 16 number system. 0 to 9 and A to F are the available values in Hexadecimal number system. The point of seperation of real part and fractional part is hexadecimal point. The positional weights of real part is $16^0, 16^1, 16^2$ --- and fractional part is $16^{-1}, 16^{-2}$ ---

Eg: $(2A71.C8)_{16}$

Number Base Conversions:

Finding the decimal equivalent: The decimal value is obtained by sum of all digits (or) coefficients multiplied by their positional weight. As given by

$$a_n r^n + a_{n-1} r^{n-1} + \cdots + a_2 r^2 + a_1 r^1 + a_0 r^0 + a_{-1} r^{-1} + a_{-2} r^{-2} + \cdots + a_{-n} r^{-n} \quad \text{— ①}$$

Here a - coefficient

r - radix

n - place value

Convert Binary to Decimal:

To convert Binary number into decimal number place radix value $r=2$ in equ ①

Convert octal to Decimal:

To convert octal to decimal value place the radix $r=8$ in equ ①

Convert Hexadecimal to Decimal:

To convert Hexadecimal to decimal value place the radix $r=16$ in equ ①

Convert the Binary Value 11101.1011 into decimal

$$\begin{array}{ccccccccccc} 1 & 1 & 1 & 0 & 1 & . & 1 & 0 & 1 & 1 \\ 4 & 3 & 2 & 1 & 0 & & -1 & -2 & -3 & -4 \end{array}$$

$1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-4}$

$16 + 8 + 4 + 1 + 0.5 + 0 + 0.1 + 0.06$

$(29.66)_{10}$

Convert the $(4057.06)_8$ into decimal

$$\begin{array}{ccccccc} 4 & 0 & 5 & 7 & . & 0 & 6 \\ 3 & 2 & 1 & 0 & & -1 & -2 \end{array}$$

$4 \times 8^3 + 0 \times 8^2 + 5 \times 8^1 + 7 \times 8^0 + 0 \times 8^{-1} + 6 \times 8^{-2}$

$2095 + 0.0937 = (2095.0937)_{10}$

Convert hexadecimal number 5C7 into decimal value

$$\begin{array}{ccc} 5 & C & 7 \\ 2 & 1 & 0 \end{array}$$

$5 \times 16^2 + C \times 16^1 + 7 \times 16^0 = 5 \times 256 + 12 \times 16 + 7$

$= 1280 + 192 + 7$

$= (1479)_{10}$

Convert the binary value 11010.11 to decimal value.

$$\begin{array}{cccccccc} 1 & 1 & 0 & 1 & 0 & . & 1 & 1 \\ 4 & 3 & 2 & 1 & 0 & & -1 & -2 \end{array}$$

$1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2}$

$16 + 8 + 2 + 0.5 + 0.25$

$(26.75)_{10}$

Convert the octal value $(127.4)_8$ into decimal value

$$\begin{array}{ccccc} 1 & 2 & 7 & . & 4 \\ 2 & 1 & 0 & & -1 \end{array}$$

$1\times8^2 + 2\times8^1 + 7\times8^0 + 4\times8^{-1}$

$64 + 16 + 7 + 0.5$

$(87.5)_{10}$

Convert hexadecimal value B65FA to decimal

B 6 5 F . A

3 2 1 0 -1

$B\times16^3 + 6\times16^2 + 5\times16^1 + F\times16^0 + A\times16^{-1}$

$11\times16^3 + 6\times256 + 80 + 15\times1 + 10\times1/16$

$45056 + 1536 + 80 + 15 + 0.625$

$(46687.625)_{10}$

Any number system to decimal

Convert $(121.10)_3$ to decimal

1 2 1 . 1 0

2 1 0 -1 -2

$1\times3^2 + 2\times3^1 + 1\times3^0 + 1\times3^{-1} + 0\times3^{-2}$

$9 + 6 + 1 + 0.33 + 0 = (16.33)_{10}$

Convert $(431.24)_5$ to decimal

4 3 1 . 2 4

2 1 0 -1 -2

$4\times5^2 + 3\times5^1 + 1\times5^0 + 2\times5^{-1} + 4\times5^{-2}$

$100 + 15 + 1 + 0.4 + 0.16 = (116.56)_{10}$

Convert $(278.16)_9$ to decimal

2 7 8 . 1 6

2 1 0 -1 -2

$2\times9^2 + 7\times9^1 + 8\times9^0 + 1\times9^{-1} + 6\times9^{-2}$

$162 + 63 + 8 + 0.11 + 0.074 = (233.184)_{10}$

Decimal to Other Number System:

This is quite reverse operation to the other Number system to decimal. The real part can be obtained by dividing the decimal Number by radix or base, fractional part by multiplying the decimal number by radix or base. till the fractional part is zero.

Eg:

(1) Convert $(41)_{10}$ to binary value.

(A)

$$\begin{array}{r|l} 2 & 41 \\ 2 & 20 - 1 \\ 2 & 10 - 0 \\ 2 & 5 - 0 \\ 2 & 2 - 1 \\ & 1 - 0 \end{array}$$

$(41)_{10} = (101001)_2$

(2) Convert $(0.625)_{10}$ to binary Value

(A)

	Real	fractional	coefficient
0.625×2	1	0.25	1
0.25×2	0	0.5	0
0.5×2	1	0	1

$(0.625)_{10} = (0.101)_2$

(3) Convert $(25.625)_{10}$ into binary value

(A)

$$\begin{array}{r|l} 2 & 25 \\ 2 & 12 - 1 \\ 2 & 6 - 0 \\ 2 & 3 - 0 \\ & 1 - 1 \end{array}$$

	Real	fractional	coefficient
0.625×2	1	0.25	1
0.25×2	0	0.5	0
0.5×2	1	0	1

$(25)_{10} = (11001)_2$ $\quad$ $(0.625)_{10} = (0.101)_2$

$(25.625)_{10} = (11001.101)_2$

(4) Convert $(125.201)_{10}$ into (i) octal & (ii) Hexadecimal

Scanned by CamScanner

(A) (i)

$$\begin{array}{r|l} 8 & 125 \\ 8 & 15 - 5 \\ & 1 - 7 \end{array}$$

$(125)_{10} = (175)_8$

	R	F	C
0.201 × 8	1	0.608	1
0.608 × 8	4	0.864	4
0.864 × 8	6	0.912	6

$(0.201)_{10} = (0.146)_8$

$(125.201)_{10} = (175.146)_8$

(ii)

$$\begin{array}{r|l} 16 & 125 \\ & 7 - 13 \end{array}$$

$(125)_{10} = (7\ 13)_{16}$

$= (7D)_{16}$

	R	F	C
0.201 × 16	3	0.216	3
0.216 × 16	3	0.456	3
0.456 × 16	7	0.296	7

$(0.201)_{10} = (0.337)_{16}$

$(125.201)_{10} = (7D.337)_{16}$

Decimal to any other number system:

(1) Convert $(293.16)_{10}$ into base 4

$$\begin{array}{r|l} 4 & 293 \\ 4 & 73 - 1 \\ 4 & 18 - 1 \\ 4 & 4 - 2 \\ & 1 - 0 \end{array}$$

	R	F	C
0.16 × 4	0	0.64	0
0.64 × 4	2	0.56	2
0.56 × 4	2	0.24	2

$(293)_{10} = (10211)_4$

$(0.16)_{10} = (0.022)_4$

$(293.16)_{10} = (10211.022)_4$

(2) Convert radix (i) 4 & (ii) 7 & (iii) 5 of decimal value 123 ?

(i)

$$\begin{array}{r|l} 4 & 123 \\ 4 & 30 - 3 \\ 4 & 7 - 2 \\ & 1 - 3 \end{array}$$

$(123)_{10} = (1323)_4$

(ii) $7 \lfloor 123$
$7 \lfloor 17 - 4$
$\quad 2 - 3$

$(123)_{10} = (234)_7$

(iii) $5 \lfloor 123$
$5 \lfloor 24 - 3$
$\quad 4 - 4$

$(123)_{10} = (443)_5$

Octal to Binary Conversion:

To convert a number from octal to binary just place each octal digit by a 3 bit binary equivalent. The 3 bit binary equivalent for the octal Numbers shown in the table below.

Binary Number	Octal Number
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Convert $(237.7)_8$ into binary

$(237.7)_8 = (010\ 011\ 111 . 111)_2$

Convert $(0.301)_8$ into binary

$(0.301)_8 = (0.011000001)_2$

Binary to Octal conversion:

To convert a binary to octal Number, first split the given number into real and fractional parts of 3 bit groups and replace the 3 bit groups by octal number.

Convert $(001110100 . 010011100)_2$ into octal

(A) $(001\ 110\ 100 . 010\ 011100)_2 = (164.234)_8$

Hexadecimal to Binary Conversion: To convert a hexadecimal number to binary replace each hexadecimal number by a four bit binary group. The four bit binary group for hexadecimal numbers is shown in table below.

HexaDecimal Number	Binary Number 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 (A)	1 0 1 0
11 (B)	1 0 1 1
12 (C)	1 1 0 0
13 (D)	1 1 0 1
14 (E)	1 1 1 0
15 (F)	1 1 1 1

(1) Convert $(47A)_{16}$ into Binary?

4 7 A

(0100) (0111) (1010)

$(47A)_{16} = (0100\,0111\,1010)_2$

(2) Find the Binary equivalent of $(0.B0D)_{16}$

$(0.1011\,0000\,1101)_2$

(3) Convert hexadecimal Number $(17E.F6)$? to Binary

(A) $(000101111110.11110110)_2$

$(17E.F6)_{16} = (000101111110.11110110)_2$

(4) Binary To Hexa Decimal Conversion:-

To convert a Binary number into hexadecimal first split the real and fractional parts, replace each 4 bit binary groups by the hexadecimal number.

(1) Convert $(11011011011)_2$ into hexa decimal. (R→L)

$\underbrace{0110}_{6}$ $\underbrace{1101}_{D}$ $\underbrace{1011}_{B}$

$(11011011011)_2 = (6DB)_{16}$

(2) Find the hexadecimal$(0.010011011)_2$? (L→R)

$0.$ $\underbrace{0100}_{4}$ $\underbrace{1101}_{D}$ $\underbrace{1000}_{8}$

$(0.010011011)_2 = (0.4D8)_{16}$

(3) Convert $(1011001110.011111)_2$ to hexadecimal.

$\underbrace{0010}_{2}$ $\underbrace{1100}_{C}$ $\underbrace{1110}_{E}$. $\underbrace{0111}_{7}$ $\underbrace{1100}_{C}$

$(1011001110.011111)_2 = (2CE.7C)_{16}$

Octal to Hexadecimal Conversions:

We can't convert octal number directly into hexa decimal number. so, use the following methods to convert. They are

(1) octal to decimal to Hexa decimal

(2) octal to binary to Hexa decimal

(1) Convert $(762.013)_8$ to Hexadecimal?

7 6 2 . 0 1 3

(111) (110) (010) (000) (001) (011)

Octal - binary

$(762.013)_8 = (111110010.000001011)_2$

binary - Hexa

$\underbrace{0001}_{1}$ $\underbrace{1111}_{\substack{15 \\ (F)}}$ $\underbrace{0010}_{2}$. $\underbrace{0000}_{0}$ $\underbrace{0101}_{5}$ $\underbrace{1000}_{8}$

$(111110010.000001011)_2 = (1F2.058)_{16}$

$(762.013)_8 = (1F2.058)_{16}$

Hexadecimal to Octal:

Method to convert Hexa decimal to octal

(1) Hexa decimal to binary to octal

(2) Hexadecimal to decimal to octal

(1) Convert $(2F.64)_{16}$ to octal

Hexa to binary

2	F	.	6	4
(0010)	(1111)		(0110)	(0100)

$(2F.64)_{16} = (00101111.01100100)_2$

Binary to octal

$\underbrace{000}_{0}$ $\underbrace{101}_{5}$ $\underbrace{111}_{7}$. $\underbrace{011}_{3}$ $\underbrace{001}_{1}$ $\underbrace{000}_{0}$

$(000101111.011001010) = (057.310)_8$

$(2F.64)_{16} = (057.312)_8$

Any number system to Other number system:

Method:

AnyNumber to decimal to other number.

(1) Convert $(012)_3$ into 7

(A) $(012)_3$ to decimal

$$\begin{matrix} (0 & 1 & 2) \\ 3^2 & 3^1 & 3^0 \end{matrix}$$

$= 0\times3^2 + 1\times3^1 + 2\times3^0$

$= 3 + 2 = (5)_{10}$

$(012)_3 = (5)_{10}$

$(5)_{10}$ to 7 Base

$7\underline{|5}$

$(012)_3 = (5)_7$

(2) Convert $(376.28)_9$ into base 5?

$(376.28)_9$ to decimal

$$\overset{2}{\underset{9}{3}}\ \overset{1}{\underset{9}{7}}\ \overset{0}{\underset{9}{6}}\ .\ \overset{-1}{\underset{9}{2}}\ \overset{-2}{\underset{9}{8}}$$

$= 3\times9^2 + 7\times9^1 + 6\times9^0 \ . + 2\times9^{-1} + 8\times9^{-2}$

$= 243 + 63 + 6 + 0.222 + 0.098$

$= (312.320)_{10}$

$(376.28)_9 = (312.320)_{10}$

$(312.320)_{10}$ to Base 5

		R	F
5 | 312	0.320 × 5	1	0.60
5 | 62 − 2	0.60 × 5	3	0
5 | 12 − 2			
2 − 2			

$(312)_{10} = (2222)_5$ $\qquad (0.320)_{10} = (0.13)_5$

$(312.320)_{10} = (2222.13)_5$

$(376.28)_9 = (2222.13)_5$

Binary Arithmetic: Binary Arithmetic is similar to decimal Arithmetic but it has some rules.

Binary Addition:- The rules for binary addition are following.

	Sum
0 + 0	0
0 + 1	1
1 + 0	1
1 + 1	0 with carry 1

Scanned by CamScanner

Eq: Add 1101.101 , 111.011 ?

```
   1 1 0 1 . 1 0 1
+    1 1 1 . 0 1 1
------------------
 1 0 1 0 1 . 0 0 0
```

Binary subtraction: Rules for binary subtraction are following

0 − 0 = 0

0 − 1 = 1 with borrow of 1

1 − 0 = 1

1 − 1 = 0

Eq:- Subtract 1010.010 , 111.111

```
     1 0 1 0 . 0 1 0
(−)    1 1 1 . 1 1 1
--------------------
     0 0 1 0 . 0 1
```

Binary Multiplication: The rules for binary multiplication are following.

0 × 0 = 0

0 × 1 = 0

1 × 0 = 0

1 × 1 = 1

Eq: Multiply 1101 with 110

```
   1101 x 110
   ----------
     0 0 0 0
   1 1 0 1
 1 1 0 1
 -------------
 1 0 0 1 1 1 0
```

Binary Division:

Eq:- Divide 101101 by 110

```
110) 101101 (111.1
     110
     ----
     1010
      110
     ----
     1001
      110
     ----
      0110
       110
       ---
       000
```

Eg:-(1) Mul 1011.101 by 101.01?

```
    1011.101 x 101.01
    -----------------
          1011101
         0000000
        1011101
       0000000
      1011101
    -----------------
      111000.00001
```

(2) Div 110101.11 by 101?

```
101) 110101.11 (1010.11
     101
     -----
     00110
       101
     -----
      00111
        101
      -----
        0101
         101
        -----
         000
```

Complements: Complements are used to simplify the subtraction operation. Leads to simple and less expensive circuits.

There are two types of complements, they are

1. (r-1)'s complement also called as diminshed radix complement.
2. r's complement also called as radix complement.

(r-1)'s complement (or) diminshed radix complement.

Given a number 'N' with radix or base 'r' and 'n' no. of digits and (r-1)'s is given by $(r^n-1)-N$

eg: Find the 9's of 2435.

$N = (2435)_{10}$

$r = 10$

$n = 4$

$(10^4-1)-2435 \quad [(r^n-1)-N]$

$9999-2435$

7564

from the above example it is clear that 9's can complem-ent can be obtained by subbracting each digit from 9.

(1) Find 9's complement for 546700

$N = (546700)_{10}$

$r = 10$

$n = 6$

$999999 - 546700$

453299

(2) Find the 9's complement of 012398

$N = (012398)_{10}$

$r = 10$

$n = 6$

$999999 - 012398$

987601

(3) Find the 1's complement of $(1101)_2$

$N = (1101)_2$

$r = 2$

$n = 4$

$(2^4 - 1) - 1101$

$15 - 1101 = 1111 - 1101$

$= 0010$

* From the above example 1's complement of the number can be obtained by subbracting each digit by one (or) simple changing zero to one, one to zero.

(4) Find the 1's complement $(1011000)_2$?

$\Rightarrow 1111111 - 1011000$

$\Rightarrow 0100111$

r's complement (or) Radix Complement: Given a number 'N', base (or) radix 'r' and 'n' no. of digits, the r's complement is given by $r^n - N$

Eg: Find 10's complement of a decimal 2389

$N = (2389)_{10}$

$r = 10$

$n = 4$

$(10^4) - 2389$

$\cancel{9999}$ $10{,}000$ $- 2389 = 7610$

10's complement = 9's complement of N + 1

$= 7610 + 1 = 7611$

Find 10's complement of (i) $(012398)_{10}$ (ii) $(246700)_{10}$

(i) 10's complement of N = 9's complement of N + 1

$= 999999 - 012398 + 1$

$= 987601 + 1 = 987602$

(ii) 10's complement of N = 9's complement of N + 1

$= 999999 - 246700 + 1$

$= 753299 + 1 = 753300$

Find the 2's complement of binary (i) $(1101100)_2$? (ii) (0110111)

(i) 2's complement of N = 1's complement of N + 1

$= 1111111 - 1101100 + 1$

$= 0010011 + 1$

$= 0010100$

(ii) 2's complement of N = 1's complement of N + 1

$= 1111111 - 0110111 + 1$

$= 1001000 + 1$

$= 1001001$

Subtraction using Complement's :-

(r - 1)'s complement	r's complement
Eg: 1's & 9's	Eg: 2's & 10's
1. Subtraction b/w two variables a & b i.e., a-b.	1. Subtraction b/w two variab-les a & b i.e., a-b.

2. $a-b = a+(-b)$	2. $a-b = a+(-b)$
3. $(-b)$ is complement of b.	3. $(-b)$ is complement of b.
4. Add $-b$ to a.	4. Add $-b$ to a.
5. ~~If~~ carry exist the result is positive other wise negative.	5. If carry exist then result is positive other wise negative
6. If carry equal to 1 add the carry to the remaining bits.	6. If carry equal to 1 eliminate the carry.
7. If carry equal to zero find the (r-1)'s complement of remaining bits.	7. If carry equal to zero find the r's complement of remaining bits.

Subtraction of two n bits unsigned numbers with base 'r' can be done as follows.

* add min u and 'a' to the subtrahend and 'b', 'b' is a r's (or) (r-1)'s complement.

* If $a \geq b$ the sum will produce an end carry and result is positive.

  Add carry to the remaining bits for (r-1)'s complement & eliminate carry for r's complement.

* If $a \leq b$ the sum doesn't produce any carry & the result is negative. Find the complement of the remaining bits.

eg① using (i) 9's & (ii) 10's complement subtract 72532 − 3250

(Sol) (i) 72532 − 3250
$\quad\quad\quad\quad$ a $\quad\quad$ b

$$\begin{array}{r} 72532 \\ -03250 \\ \hline \end{array}$$

b = 03250

$-b$ = 9's complement of b

$-b$ = 99999 − 03250

$-b = 96749$

$a+(-b) = 72532$

$$\begin{array}{r} 72532 \\ +\ 96749 \\ \hline (1)69281 \end{array}$$

carry

carry = 1, carry exist result is +ve

In 9's complement if carry exist add the carry to remaining bits

$$\begin{array}{r} 69281 \\ +\ \ \ \ \ \ 1 \\ \hline +69282 \end{array}$$

$a+(-b) = +69282$

(ii) $b = 03250$

$-b =$ 10's complement of b

= 9's complement of b + 1

$-b = 99999 - 03250 + 1$

$-b = 96749 + 1 = 96750$

$a+(-b) = 72532$

$$\begin{array}{r} 72532 \\ +\ 96750 \\ \hline (1)69282 \end{array}$$

carry

carry = 1, result is positive

If 10's complement if carry exist, equal to 1 eliminate the carry.

$a+(-b) = +69282$

(2) Using (i) 9's & (ii) 10's complement subtract 3250 − 72532

(Sol) (i) 3250 − 72532
a b

$b = 72532$

$-b =$ 9's complement of b

$-b = 99999 - 72532$

$-b = 27467$

$a+(-b) = \begin{array}{r} 03250 \\ +27467 \\ \hline 30717 \end{array}$

Carry = 0 , result is -ve

In 9's complement If carry=0 , find complement of remaining bits

$\Rightarrow 99999 - 30717$

$\Rightarrow -69282$

$a+(-b) = -69282$

(ii) $b = 72532$

$-b$ = 10's complement of b

$-b = 99999 - 72532 + 1$

$-b = 27468$

$a+(-b) = \begin{array}{r} 03250 \\ +27468 \\ \hline 30718 \end{array}$

carry=0 , result negative

In 10's complement If carry = 0 , Find complement of remaining bits.

$\Rightarrow 99999 - 30718 + 1$

$\Rightarrow 69281 + 1$

$\Rightarrow -69282$

$a+(-b) = -69282$

(3) Given the two binary numbers $x = 1010100$ & $y = 1000011$ perform subtraction (i) $x-y$, (ii) $y-x$ using 1's & 2's complement?

(A) (i) $x-y = x+(-y)$

$y = 1000011$

$-y$ = 1's complement of y

$-y = 1111111 - 1000011$

$-y = 0111100$

$x+(-y) = 1010100$

$+ 0111100$

carry ← ①0010000

carry = 1, result +ve

In 1's complement carry = 1 add carry to the remaining bits

0010000

+ 1

+0010001

$x+(-y) = +0010001$

2's complement :

$y = 1000011$

$-y = 1111111 - 1000011 + 1$

$= 0111100 + 1$

$-y = 0111101$

$x+(-y) = 1010100$

$+ 0111101$

carry ← ①0010001

carry = 1, result +ve

In 2's complement if carry = 1 then eliminate the carry.

$x+(-y) = +0010001$

(ii) 1's complement

$y - x = y + (-x)$

$x = 1010100$

$-x$ = 1's complement of x

$-x$ = 1111111 − 1010100

$-x$ = 0101011

$y+(-x)$ = 1000011
+ 0101011
1101110

carry=0, result −ve

In 1's complement. If carry=0, then find the complement for remaining bits.

⇒ −1101110 + 1111111

⇒ −0010001

$y+(-x)$ = −0010001

2's complement

x = 1010100

$-x$ = 1111111 − 1010100 + 1

$-x$ = 0101011 + 1

$-x$ = 0101100

$y+(-x)$ = 1000011
+ 0101100
1101111

carry=0, result −ve

In 2's complement, If carry=0, then remaining bits complement.

⇒ 1111111 − 1101111 + 1

⇒ 0010000 + 1

⇒ −0010001

$y+(-x)$ = −0010001

Signed Binary Numbers:

Numbers without any sign is known as unsigned numbers. 9 is an example for unsigned numbers.

Numbers with sign is known as signed numbers. The sign can be positive (or) negative. The numbers with +ve sign is known as positive sign numbers. The numbers with -ve sign is known as negative sign numbers.

Generally Decimal Numbers are represented with +ve & -ve sign. i.e., +255, -255. But we can't use +ve & -ve signs to represent binary numbers. So, Instead we use zero to represent the +ve number and one to represent the -ve number. In the left most position of the Binary number.

Negative signed numbers has three forms of representation.

1. Signed magnitude form
2. 1's complement form
3. 2's complement form.

Positive signed numbers has only one form of representation. i.e., signed magnitude form. 1's complement form and 2's complement form are same as Signed magnitude form.

Number	Signed magnitude	1's complement	2's complement
-7	10000111	11111000	11111001
-6	10000110	11111001	11111010
-5	10000101	11111010	11111011
-4	10000100	11111011	11111100
-3	10000011	11111100	11111101
-2	10000010	11111101	11111110
-1	10000001	11111110	11111111
-0	10000000	11111111	—
+0	00000000	00000000	00000000

1	00000001	00000001	00000001
2	00000010	00000010	00000010
3	00000011	00000011	00000011
4	00000100	00000100	00000100
5	00000101	00000101	00000101
6	00000110	00000110	00000110
7	00000111	00000111	00000111

Note:-

No. of bits required to represent a decimal number in binary is given by $2^{n-1}-1 \geq$ Maximum

Eg: Find the sign magnitude representation of decimal numbers using 8 bits (a) +27, (b) −27, (c) −101, (d) −106

(a) +27

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

27 = 11011

27 in 8 bits = 00011011

+27 = 00011011

(b) −27 = 10011011

(c) −101

2 | 101
2 | 50 − 1
2 | 25 − 0
2 | 12 − 1
2 | 6 − 0
2 | 3 − 0
1 − 1

101 = 1100101

101 in 8 bits = 01100101

−101 = 11100101

(d) −106

2 | 106
2 | 53 − 0
2 | 26 − 1
2 | 13 − 0
2 | 6 − 1
2 | 3 − 0
1 − 1

106 = 1101010

106 in 8 bits = 01101010

−106 = 11101010

(2) Represent the following decimal number using 1's complement form. In 8bits

(a) -67 (b) +102 (c) -88 (d) -45 , 2's complement.

(A) -67

2 | 67
2 | 33 - 1
2 | 16 - 1
2 | 8 - 0
2 | 4 - 0
2 | 2 - 0
  | 1 - 0

67 = 1000011

8bits = 01000011

-67 = 11000011

1's complement = 10111100

(b) +102

2 | 102
2 | 51 - 0
2 | 25 - 1
2 | 12 - 1
2 | 6 - 0
2 | 3 - 0
  | 1 - 1

102 = 1100110

8bits = 01100110

+102 = 01100110

1's complement = 00011001

(C) -88

2 | 88
2 | 44 - 0
2 | 22 - 0
2 | 11 - 0
2 | 5 - 1
2 | 2 - 1
2 | 1 - 0

88 = 1011000

8bits = 01011000

-88 = 11011000

1's complement = 10100111

(d) -45

2 | 45
2 | 22 - 1
2 | 11 - 0
2 | 5 - 1
2 | 2 - 1
  | 1 - 0

45 = 101101

8bits = 00101101

-45 = 10101101

2's complement = 11010010 + 1

= 11010011

(3) Subtract 14 from 25 using 8 bit in 1's complement?

(A)

2 | 14
2 | 7 - 0
2 | 3 - 1
  | 1 - 1

14 = 1110

2 | 25
2 | 12 - 1
2 | 6 - 0
2 | 3 - 0
  | 1 - 1

25 = 11001

-14 in 8 bits = 10001110

-14 in 1's complement = 11110001

25 in 8 bits = 00011001

25 in 1's complement = 00011001

25+(-14) = 00011001
(+) 11110001
①00001010

Carry = 1 , add carry to remaining bits.

25+(-14) = 00001010
1
00001011

25+(-14) = 00001011

Rules to perform signed subtraction using 1's & 2's complement:-

* When we add two signed numbers if there is a carry add carry to remaining bits in 1's complement and eliminate carry in 2's complement.
* When we add two signed numbers, if the MSB bit is zero in the result, the result is in true form. and it is a positive number.
* When we add two signed numbers, if MSB bit is '1' in the result, the result is negative, It is in 1's complement (or) 2's complement form. To obtain the true form find the 1's complement (or) 2'complement of the result.

(4) Add -25 to +14 in 8bit's 1's complement?

(A)
2 | 14
2 | 7 - 0
2 | 3 - 1
  1 - 1

14 = 1110

8bits = 00001110

2 | 25
2 | 12 - 1
2 | 6 - 0
2 | 3 - 0
  1 - 1

25 = 11001

8bits = 00011001

14 in 1's complement = 00001110

-25 in 1's complement = 11100110

$14 + (-25) = $ 00001110
(+) 11100110

11110100

sign is one, write complement of remaining.

$14 + (-25) = 10001011$

(5) Find the decimal equivalent of the following binary numbers?
(a) $(10100001)_2$ (b) $(00010011)_2$ assume the given numbers in sign magnitude form.

(A)
(a) $(10100001)_2$ is in sign magnitude form; MSB = 1, so result -ve
Find decimal for remaining bits.
0100001

$0 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$

$\Rightarrow (-32)_{10}$

(b) Given $(00010011)_2$

Given number is in signmagnitude form.

MSB = 0, so result is +ve

Find decimal of remaining bits

$0 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2 + 1 \times 2^0$

$\Rightarrow 16 + 2 + 1 = (+19)_{10}$

(6) Find the decimal equivalent of following binary numbers
(a) $(10100111)_2$ (b) $(01010011)_2$ assume the given numbers in 1's complement form?

(A)
(a) Given number is in 1's complement form

MSB = 1, Given number is -Ve number.

Find the 1's complement of remaining bits and find decimal for them.

$(10100111)_2$

1's complement $= (11011000)_2$

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

$\Rightarrow 64+16+8$

$\Rightarrow (-88)_{10}$

(b) Given number is in 1's complement form

MSB = 0, Given number result is +ve

so, Find decimal of remaining bits

$(0101011)_2$

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

$\Rightarrow 64+16+2+1$

$\Rightarrow (+83)_{10}$

(7) Find the decimal equivalent of following binary numbers assume them in 2's complement form.

(a) $(10011001)_2$ (b) $(01100111)_2$

(A)

(a) Given number is in 2's complement form

MSB = 1, number is -ve

so, to obtain true form find 2's complement of given number and convert to decimal.

$(1001100 1)_2$

2's complement = 1100110 + 1

= 1100111

1100111

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

$\Rightarrow 64+32+4+2+1$

$\Rightarrow (-103)_{10}$

Scanned by CamScanner

(b) $(01100111)_2$

Given number is in 2's complement

MSB = 0 , number is +ve

Find the decimal equivalent

01100111

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

$\Rightarrow 64+32+4+2+1 \Rightarrow (+103)_{10}$

(8) Add -25 to -14 using 8bits 1's complement

(A)
$$\begin{array}{r|l} 2 & 25 \\ 2 & 12-1 \\ 2 & 6-0 \\ 2 & 3-0 \\ & 1-1 \end{array} \quad 25 = 11001 \qquad \begin{array}{r|l} 2 & 14 \\ 2 & 7-0 \\ 2 & 3-1 \\ & 1-1 \end{array} \quad 14 = 1110$$

25 in 8bits = 00011001   14 in 8 bits = 00001110

-25 in 1's complement = 11100110   -14 in 1's complement = 11110001

$$\begin{array}{rr} (-25)+(-14) = & 11100110 \\ (+) & 11110001 \\ \hline & 1\,11010111 \end{array}$$ carry = 1 add carry to remaining bits

$$\begin{array}{rr} (-25)+(-14) = & 11010111 \\ (+) & 1 \\ \hline & 11011000 \end{array}$$ MSB = 1 write complement of remaining

(-25) + (-14) = 10100111

(9) subtract 14 from 46 using 8bits 2's complement

(A)
$$\begin{array}{r|l} 2 & 14 \\ 2 & 7-0 \\ 2 & 3-1 \\ & 1-1 \end{array} \quad 14 = 1110 \qquad \begin{array}{r|l} 2 & 46 \\ 2 & 23-0 \\ 2 & 11-1 \\ 2 & 5-1 \\ 2 & 2-1 \\ & 1-0 \end{array} \quad 46 = 101110$$

46 in 8bits = 00101110   14 in 8bits = 00001110

46 in 2's complement = 00101110   -14 in 2's complement = 11110001+1 = 11110010

46 + (-14) = 00101110 + 11110010 = 100100000

Carry = 1 , In 2's complement eliminate carry.

46 + (-14) = 00100000

(10) Add -75 to +26 using 8bit's 2's complement?

(A)
$$\begin{array}{r|l} 2 & 75 \\ 2 & 37-1 \\ 2 & 18-1 \\ 2 & 9-0 \\ 2 & 4-1 \\ 2 & 2-0 \\ & 1-0 \end{array} \quad 75 = 1001011 \qquad \begin{array}{r|l} 2 & 26 \\ 2 & 13-0 \\ 2 & 6-1 \\ 2 & 3-0 \\ & 1-1 \end{array} \quad 26 = 11010$$

26 in 8 bits = 00011010

75 in 8bits = 01001011

-75 in 2's complement = 10110100 + 1 = 10110101

+26 in 2's complement = 00011010

$$
\begin{array}{rr}
26+(-75) = & 00011010 \\
(+) & 10110101 \\
\hline
 & 11001111
\end{array}
$$

MSB = 1 , write 2's complement of rem bits.

(-75) + 26 = 10110000 + 1 = 10110001

(11) Add -89.75 to 43.25 using 12bit 1's complement?

(A)

$$
\begin{array}{l|l}
2 & 89 \\
2 & 44 - 1 \\
2 & 22 - 0 \\
2 & 11 - 0 \\
2 & 5 - 1 \\
2 & 2 - 1 \\
 & 1 - 0
\end{array}
$$

89 = 1011001

8bits = 01011001

	R	F	C
0.75x2	1	0.5	1
0.5x2	1	0	1

0.75 = 0.11

0.75 in 4bits = 0.1100

-89 in 1's complement = 10100110.0011

$$
\begin{array}{l|l}
2 & 43 \\
2 & 21 - 1 \\
2 & 10 - 1 \\
2 & 5 - 0 \\
2 & 2 - 1 \\
 & 1 - 0
\end{array}
$$

43 = 101011

43 8bits = 00101011

	R	F	C
0.25x2	0	0.5	0
0.5x2	1	0	1

0.25 in 4bits = 0.0100

43.25 in 1's complement = 00101011.0100.

(-89.75) + (43.25) = 10100110.0011 + 00101011.0100 = 11010001.0111

MSB = 1, write 1's complement of remaining bits.

(-89.75) + (43.25) = 1010110.1000

(12) Add 27.125 to -79.65 in 12bit 2's complement?

(A)

$$
\begin{array}{l|l}
2 & 27 \\
2 & 13 - 1 \\
2 & 6 - 1 \\
2 & 3 - 0 \\
 & 1 - 1
\end{array}
$$

27 in 8bits = 00011011

	R	F	C
0.125x2	0	0.25	0
0.25x2	0	0.5	0
0.5x2	1	0	1

0.125 in 4bits = 0.0010

27.125 in 2's complement = 00011011.0010

$$
\begin{array}{l|l}
2 & 79 \\
2 & 39 - 1 \\
2 & 19 - 1 \\
2 & 9 - 1 \\
2 & 4 - 1 \\
2 & 2 - 0 \\
 & 1 - 0
\end{array}
$$

+79 in 8bits = 01001111

	R	F	C
0.65 x2	1	0.3	1
0.3x2	0	0.6	0
0.6x2	1	0.2	1
0.2x2	0	0.4	0

0.65 in 4bits = 0.1010

-79.65 in 2's complement = 10110000.0101 + 1 = 10110000.0110

(-79.65) + (27.125) = 00011011.0010 + 10110000.0110 = 11001011.~~0111~~ 1000

MSB = 1 , write 2's complement of remaining bits.

(-79.65) + (27.125) = 10110100.0111 + 1

= 10110100.1000

-52.125

Binary Codes: The Computer stores and process the data in the form of binary. Hence numerals, alphabets, special characters and control functions are to be converted into binary format. The process of converting them into binary format is known as Binary Codes.

There are different types of binary codes

(1) Weighted and non weighted codes

(2) Numeric and alpha-numeric codes

(3) error detecting and correcting codes

(4) self complementary codes

(5) unit distance codes (cyclic codes)

(6) Reflective codes

(7) Sequential codes

Weighted and Non weighted codes: Weighted codes are the binary codes in which each bit position of the binary number has some weight. Adding the weights when the binary digit is '1' gives the decimal value.

eg: BCD, 8421, 2421 etc.,

Non weighted codes are binary codes which are not assigned with any weight.

Eg: Excess-3 (XS-3), Gray codes

Numeric and alpha numeric codes: Numeric codes are binary codes which represents only numeric data.

Eg: 8421, BCD, XS-3, Gray, 2421 codes

Alpha numeric codes are binary codes which represents numbers, Alphabets, special characters

eg: ASCII (Americal standard code for Information Interchange), EBCDIC (Extended Binary code and Decimal Interchange code).

Error detecting and Correcting codes: When binary information is transmitted for longer distances the error may be introduced i.e., 0 may change as 1 and 1 may change as 0. Due to the presence of noice. Some special codes are used to detect and correct the errors.

eg: parity and Hamming code

parity code is used to detect the errors.

Hamming code is used to detect & correct the errors.

* Error detection and correction involves the addition of extra bit to the transmitted data is known as parity bit. and check bit

Self Complementary Codes: A code is said to be self complementary code. If code word of the 9's complement of N can be obtained by the 1's complement of the code word of N.

eg: Excess-3 is a self complementary code.

* In this code N = 2 is represented as 0101. Hence the code word of 9's complement of N. i.e., 9-2=7, can be obtained by 1's complement of code word of 2 which is 1010 and it is excess-3 code word of 7.

Eg:- 4221, 5211, XS-3 codes.

Unit distance Codes: The name itself indicates that there is a unit distance between the two consecutive codes. i.e., each successive code differs from the preciding one by only one bit. They are also known as cyclic codes. grey code is a cyclic code.

Sequential Codes: In sequential codes each successive code is one number greater than the preceeding one.

eg: 8421, XS-3

Reflective Codes: A reflective code is a binary code in which least n significant bits for the code word 2^n to $2^{n+1}-1$ are the mirror images of 0 to 2^n-1.

Eg: gray code is a reflective code.

BCD (Binary coded Decimal): Numerical codes (8421) used to represent the decimal digits are called binary coded decimal. (BCD). The 8421 code is widely used known as BCD. It is a natural binary code each decimal digit can be represented in BCD by using 4 bits.

The BCD codes for the decimal digits 0 to 9 is shown in table below.

Decimal	BCD (8 4 2 1)
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

There are six invalid codes as listed below.

10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

Eg:- 15 in binary → 1111

15 in BCD → 0001 0101

Advantage:-

It is easy to convert BCD into decimal and vise versa.

Disadvantage:-

BCD is less efficient. Hence more number of bits is used to represent in BCD.

Rules for BCD Addition:-

(1) Add the two BCD numbers using normal binary addition.

(2) If the result (or) sum is equal to 9 (or) less than 9 no correction is required. The sum is a valid BCD.

(3) If the result (or) sum is greater than 9 (or) If a carry is generated from 4bit sum the correction is needed.

(4) To get the correct BCD add 6 i.e., 0110 to the sum. When 6 is added to the sum, If a carry is generated add the carry to the next higher BCD

Eg:(1) Add $(28)_{10}$ with $(18)_{10}$ using BCD addition?

$(28)_{10}$		0010	1000
$(18)_{10}$		0001	1000
(+) 46	(+)	11	
		0100	0000
	(+)		0110
		$\underbrace{0100}_{4}$	$\underbrace{0110}_{6}$

Carry generated add 6 to sum i.e (0110

Corrected BCD $= (46)_{10}$

(2) Add $(147)_{10}$ to $(256)_{10}$ using BCD addition?

$(147)_{10}$	0001	0100	0111
$(256)_{10}$	0010	0101	0110
$(403)_{10}$		1	11
	$\underbrace{0011}_{3<9}$	$\underbrace{1001}_{9\leq 9}$	$\underbrace{1101}_{13>9}$
		11	0110
	$\underbrace{0011}_{3<9}$	$\underbrace{1010}_{10>9}$	$\underbrace{0011}_{3<9}$

(13>9 add 6 (0110) to get correct BCD)

(10>9 add 6 (0110) to get correct BCD)

Scanned by CamScanner

0011 1010 0011
111 0110

0100 0000 0011
4<9 0<9 3<9

∴ Corrected BCD = $(403)_{10}$

(3) Add $(679.6)_{10}$ to $(536.8)_{10}$ using Decimal addition?

$(679.6)_{10}$ 0110 0111 1001 . 0110

$(536.8)_{10}$ (+) 0101 0011 0110 . 1000

(+) 1216.4

1011 1010 1111 . 1110 (All are >9 to get correct BCD add 6(0110))

11>9 10>9 15>9 14>9

0110 0110 0110 . 0110

0001 0010 0001 0110 . 0100

1 2 1 6 4

Corrected BCD

= $(1216.4)_{10}$

Rules for BCD subtraction:

(1) Subtract subtrahend from the minuend using normal binary subtraction.

(2) If no borrow is taken from the next higher BCD number no correction is required.

(3) If a borrow is taken from the next higher BCD Correction is needed. to correct the BCD subtract 6 (0110) from the difference result to get the correct and valid BCD.

Eg①: Subtract $(189)_{10}$ from $(203)_{10}$ using BCD subtraction?

$(203)_{10}$ 0010 0000 0011

$(189)_{10}$ 0001 1000 1001

(−)

0000 0111 1010

0110 0110

(−)

0000 0001 0100

$\underbrace{0000}_{0}$ $\underbrace{0001}_{1}$ $\underbrace{0100}_{4}$

∴ corrected & valid BCD = $(014)_{10}$

② perform subtraction using 8421 code

(a) $(38)_{10} - (15)_{10}$

(b) $(206.7)_{10} - (147.8)_{10}$

(a) $(38)_{10}$ 0011 1000

$(15)_{10}$ 0001 0101

(−)

$\underbrace{0010}_{2}$ $\underbrace{0011}_{3}$

corrected & valid BCD = $(23)_{10}$

(b) $(206.7)_{10}$ 0010 0000 0110 . 0111

$(147.8)_{10}$ 0001 0100 0111 . 1000

(−)

0000 1011 1110 . 1111

0110 0110 0110

0000 $\underbrace{0101}_{5}$ $\underbrace{1000}_{8}$. $\underbrace{1001}_{9}$

corrected & valid BCD = $(058.9)_{10}$

BCD subtraction using 9's & 10's complement:

* Find 9's or 10's complement of subtrahand then represent the decimal in BCD and add the BCD numbers.

Scanned by CamScanner

Eg① Perform the following subtraction in BCD by using 9's complement (a) 305.5 − 168.8 (b) 679.6 − 885.9

(a) 305.5 − 168.8

a = 305.5

b = 168.8

a−b = a+(−b)

−b = 9's complement of b

= 999.9 − 168.8

= 831.1

a = 305.5 ⇒ 0011 0000 0101 .0101

−b = 831.1 ⇒ 1000 0011 0001 . 0001

(+) 136.7

(+)

1011 0011 0110 . 0110

11>9 3<9 6<9 6<9

(+) 0110

carry 1) 0001 0011 0110 . 0110

Carry exist given number is +ve add carry to LSB

0001 0011 0110. 0110

(+) 1

0001 0011 0110. 0111

1 3 6 . 7

Corrected BCD $(+136.7)_{10}$

(b) 679.6 − 885.9

a = 679.6

b = 885.9

a−b = a+(−b)

−b = 9's complement of b

= 999.9 − 885.9

= 114.0

a = 679.6 , −b = 114.0

$a = 679.6 \Rightarrow$ 0110 0111 1001 . 0110

$-b = 114.0$ 0001 0001 0100 . 0000

(+)

0111 1000 1101 . 0110

7<9 8<9 13>9 6<9

(+) 0110

0111 1001 0011 . 0110

7<9 9<9 3<9 6<9

(+) 0110

0111 1001 1001 0110

No carry result is negative find the 9's complement of result.

$\Rightarrow 999.9 - 793.6 = 206.3$

corrected BCD $\Rightarrow (-206.3)_{10}$

(2) Perform the following subtraction in 8421 in 10's complement.

(a) 342.7 − 108.9 , (b) 507.6 − 206.4

(a) 342.7 − 108.9

$a = 342.7$

$b = 108.9$

$a-b = a+(-b)$

$-b$ = 10's complement of b

$= 999.9 - 108.9 + 1 = 891.0+1 = 891.1$

$a = 342.7 \Rightarrow$ 0011 0100 0010 . 0111

$-b = 891.1 \Rightarrow$ 1000 1001 0001 . 0001

+

1011 1101 0011 . 1000

11>9 13>9 3<9 8<9

(+) 0110 0110

1) 0010 0011 0011 . 1000

2 3 3 . 8

carry equal to 1 and result positive. eleminate carry in 10's complement.

Corrected BCD = $(+233.8)_{10}$

(b) 507.6 − 206.4

a = 507.6

b = 206.4

a−b = a+(−b)

−b = 10's Complement of b

= 999.9 − 206.4+1 = 793.5 +1 = 793.6

a = 507.6 ⇒	0101	0000	0111	. 0110
−b = 793.6 ⇒	0111	1001	0011	. 0110
(+)	1100	1001	1010	. 1100
	12>9	9≤9	10>9	12>9
	0110		0110	0110
	1 0010	1001	1011	. 0010
Carry → exist.			12>9 0110	
	1 0010	1010	0001	. 0010
		(10>9) 0110		
result is positive eliminate carry in 10's complement	1 0011	0000	0001	. 0010
	3	0	1	2

corrected BCD = $(+301.2)_{10}$

XS−3 code:- It is a non weighted code, sequential code, self Complementary code. To get the XS-3 number add three to the given number. The table below shows the XS-3 code for decimal number.

Decimal	Excess-3 (8421) (+3)
0	0011
1	0100
2	0101
3	0110
4	0111
5	1000
6	1001
7	1010
8	1011

9	1100

* There are six invalid codes in excess-3

0	0000
1	0001
2	0010
13	1101
14	1110
15	1111

Rules for Excess-3 Addition:

(1) Add 2 excess-3 numbers.

(2) If a carry is generated to the next higher 4bit, add 3 (i.e 0011) to the result.

(3) If no carry is generated to the next higher 4 bit, subtract 3 (i.e., 0011) from the result.

eg:① Add 37 with 28 using XS-3 addition?

(A) 37 → 0110 1010

28 → 0101 ③ 1011

(+)

1100 0101 (carry +3)

(No carry -3)

-0011 +0011

1001 1000

$\underbrace{1001}_{9-3=6}$ $\underbrace{1000}_{8-3=5}$

corrected XS-3 = (65)

② Add 247.6 with 359.4 using XS-3 addition?

247.6 0101 0100 1010 . 1000

359.4 (+) 0110 1000 ③ 1100 . 0100

1011 1101 0110 . 1100

0011

247.6 0101 0111 1010 . 1001

359.4 (+) 0110 ③ 1000 ③ 1100 ③ 0111

1100 0000 0111 0000

+0011 +0011 +0011 +0011 corrected XS-3 ⇒ (607.0)

$\underbrace{1001}_{9-3}$ $\underbrace{0011}_{3-3}$ $\underbrace{1010}_{10-3}$ $\underbrace{0011}_{3-3}$

Rules for XS-3 substraction:

(1) substract two excess-3 numbers

(2) If a borrow is taken from the next higher BCD, then substract 3 (i.e, 0011)

(3) If a borrow is not taken from the next higher BCD, then add 3 (i.e., 0011)

Eg:

① perform following substractions in XS-3 code:

(a) 267 - 175 (b) 27.8 - 57.6

(a)

267	0101 ③	1001	1010
(-) 175	0100	1010	1000
	0000	1111	0010
	+0011	-0011	+0011
	$\underbrace{0011}_{3-3}$	$\underbrace{1100}_{12-3}$	$\underbrace{0101}_{5-3}$

corrected XS-3 = (092)

(b)

27.8	0101	1010	. 1011
(-) 57.6	1000	1010	1001
	1101	0000	0010

Note:

Higher number (or) Larger number can not be substracted from smaller number. In XS-3 substraction with out using complements.

∴ It is not possible.

XS-3 subtraction using 9's & 10's complement:

Subtraction is performed for two XS-3 number using 9's & 10's complement by taking the complement of the subtrahend.

eg ① substract the following

348 from 687 using 9's complement method

(A) $-348 + 687$
b a

$a-b = a+(-b)$

$-b =$ 9's complement of b

$= 999 - 348$

$= 651$

a+(-b) ⇒ 687 = 1001 1011 1010

(+) 651 = 1001 1000 0100

(+) 1 1 1

1 0011 0011 1110

1

In 9's complement carry exist add carry to remaining bits

0011 0011 1111

+0011 +0011 −0011

0110 0110 1100

6−3 6−3 12−3

Corrected XS-3 = (+339)

② perform following substraction using 10's complement method (XS-3)

(a) 597−239 (b) 234 − 672

(A) (a) 597 − 239
a b

$a-b = a+(-b)$

$-b$ = 10's complement of b

= 999 - 239 + 1 = 760+1 = 761

a + (-b)

597 =	1000	1100	1010
761 (+) =	1010	1001	0100
	1		
	1̲0011	0101	1110

In 10's complement carry exist eliminate carry

+0011	+0011	-0011
0110	1000	1011
6-3	8-3	11-3

corrected XS-3 = (+ 358)

(b) 234 - 672
a b

a-b = a+(-b)

$-b$ = 10's complement of b = 999 - 672 + 1 = 328

234 =	0101	0110	0111
328 = (+)	0110	0101	1011
	1011	1100	0010

In 10's complement carry=0 complement of remaining bits.

0100 0011 1101

Gray Code (Reflective code): Gray code is a non weighted code, reflective code, cyclic code, unit distance code & Numeric code.

To obtain the gray code from binary & Binary from gray code X-OR (Exclusive OR) operation need to be performed. XOR operation is given below.

A	B	$A \oplus B$, $\bar{A}B + A\bar{B}$
0	0	0
0	1	1
1	0	1
1	1	0

The Gray code & Its decimal equivalent and binary equivalent is shown in table below.

Graycode	Decimal	Binary code (8421)
0000	0	0000
0001	1	0001
0011	2	0010
0010	3	0011
0110	4	0100
0111	5	0101
0101	6	0110
0100	7	0111
1100	8	1000
1101	9	1001
1111	10	1010
1110	11	1011
1010	12	1100
1011	13	1101
1001	14	1110
1000	15	1111

Binary to Gray code conversion:

Rules:

(1) The MSB in Gray code is same as MSB in binary.

(2) To find the next bit in gray code perform XOR operation of present bit with previous bit.

(3) Repeat step ② until all the binary values are XOR with the previous bits.

Eg:-

① Find the gray code for 1011011

	Binary		Gray
(A)	1011011	=	1110110

② Find the gray code for $(3A7)_{16}$

(A) $(3A7)_{16} = (0011 \quad 1010 \quad 0111)_2$

$(001110100111)_2 =$ Gray 001001110100

Gray code to Binary Code conversion :- Rules

(1) The MSB bit in binary is same as Gray code

(2) To find the Next bit in binary perform XOR operation with the present bit in gray code to the previous bit in binary code.

(3) Repeat step② until all bits in gray code are XOR with previous bit in binary form.

Eg:-

① convert 1110110 into binary?

(A) $(1011011)_2$

② Write the Gray Code equivalent for the octal number $(756)_8$?

(A) $(756)_8 = (111 \quad 101 \quad 110)_2$

$(111101110)_2 =$ Gray 100011001

Scanned by CamScanner

Binary Logic: It deals with the binary variables & logic expressions. The variables are designated by A, B, C and X, Y, Z. With each variable having only two possible values 0 & 1. There are three basic logic operations. They are and, OR, NOT.

(1) AND :- The operation is represented by (•) dot.

eg: $A \cdot B = C$ (or) $AB = C$

It is pronounced as A AND B is equal to C.

C = 1, If and only If A = 1 & B = 1. Otherwise C equal to 0.

(2) OR :- The OR operation is represented by + sign.

Eg: $A+B = C$

It is pronounced as A OR B is equal to C.

C=1, If A = 1 or B = 1 and A = 1 and B = 1. otherwise 0.

(3) NOT :- The operation is represented by prime(') (or) over bar $(\bar{\ })$

eg: $A' = C$ (or) $\bar{A} = C$

C=0, If A = 1 and C = 1, If A = 0

NOT operation is also known as complement operation. Since it changes 0 to 1 and 1 to 0.

* Binary logic looks like Binary arithematic and the operations AND & OR have similarities to multiplication and addition. How ever binary arithematic is different from Binary logic.

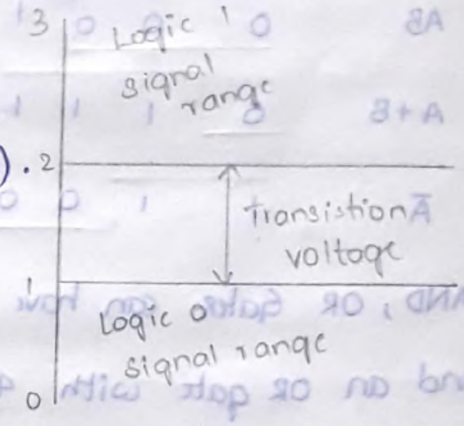
eg: In Binary arithmatic $1+1 = 10$ (carry = 1, sum = 0)

In Binary logic $1+1$ {1 OR 1} $= 1$

* Definitions of logical operations are used in a compact form known as truth tables. The truth tables for AND, OR, NOT is shown in table below.

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

A	A' (or) $\bar{A}$
0	1
1	0



Logical Gates.

Analog input is 0 to 3V (continuous).

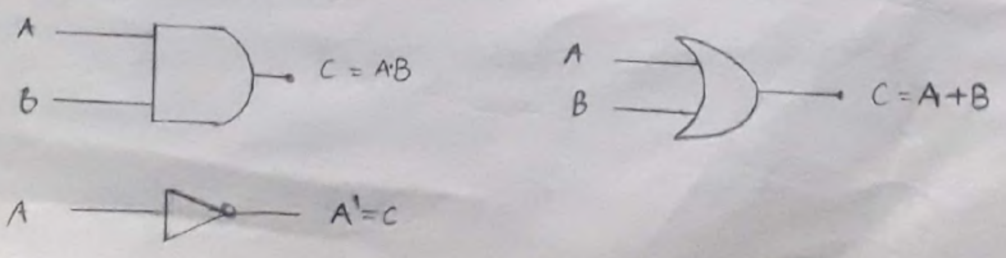
Logic 0 is low input. (0 to 1V)

Logic 1 high input. (2 to 3V)

* logic gates are electronic circuits that operates on one or more inputs to produce a output signal.

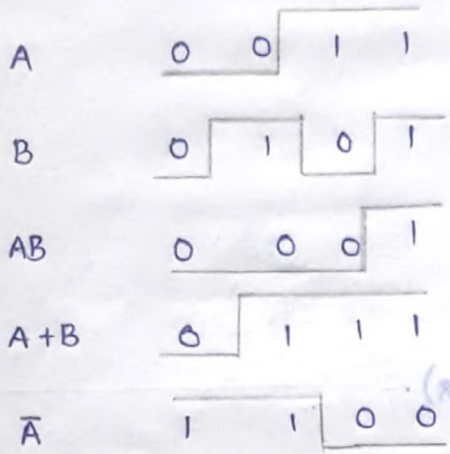
* analog signals are current (or) Voltage signals having values over a given range 0 to 3V. Digital signals are also voltage or current signals but having two voltage levels. logic 0 as a signal equal to 0 to 1V and Logic 1 as a signal equal to 2 to 3V But in practice each voltage level has a range shown in figure above.

* The Graphic symbols for AND, OR and NOR are shown in figure below. 00, 10, 01, 11.

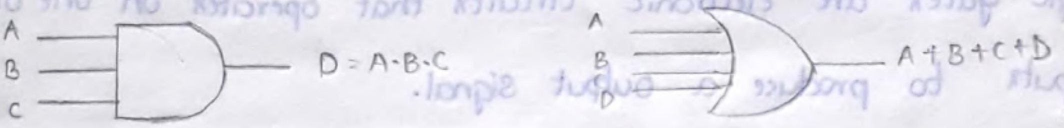


Gates are the basic building blocks of hardware circuit in digital systems. The Input signal exist in one of the following states.

The timing diagrams for Gates is shown in figure below.



* AND, OR Gates can have multiple inputs, AND gate with 3 inputs and an OR gate with 4 inputs are shown in figure below.



Binary storage and Registors:

Part-2 : BOOLEAN ALGEBRA AND LOGIC GATES

Axiomatic Def of Boolean Algebra: The Boolean algebra can be defined as(i) set of elements . {A, B, C - - - . } where there are 0 and 1.

(2) set of operators {+, · , ⊕ , ⊙}

(3) Number of Rules

(4) postulates

(5) Theorems

Rules in Boolean Algebra:

Rule① Any symbol which represents an arbitary element of boolean Algebra is known as a variable.

② A single variable (or) a function of a set of Variables can have single value either zero (0) (or) one (1) .

Eg: (i) variable A has values either 0 (or) 1.

(ii) Function Y = A+BC has values 0 (or) 1

③ complement of any variable (A) is represented by A' or $\bar{A}$.

④ The logical And operation of any two variables is represented by dot. (·)

⑤ The logical OR operation of any two variables is represented by +

Postulates and theorems of Boolean Algebra: The postulates are logical expressions that are accepted without any proof. Postulates are nothing more than the three basic logical operations AND, OR, NOT that we have already discussed. Postulates are also known as axioms.

AND	OR	NOT
0·0 = 0	0+0 = 0	$\bar{0}$ = 1
0·1 = 0	0+1 = 1	$\bar{1}$ = 0
1·0 = 0	1+0 = 1	
1·1 = 1	1+1 = 1	

Theorems are used to simplify the boolean expressions.

Theorem ①: Complementation Law

complement means invert i.e., changing 0 to 1 and 1 to 0. There are five complementation laws.

$\overline{0} = 1$

$\overline{1} = 0$

If $A = 0$ then $\overline{A} = 1$

If $A = 1$ then $\overline{A} = 0$

$\overline{\overline{A}} = A$ (double complementation law)

Theorem ②: AND Laws

There are four AND laws

$A \cdot 0 = 0$

$A \cdot 1 = A$

$A \cdot A = A$

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

Theorem ③: OR Laws.

There are 4 OR Laws

$A + 0 = A$

$A + 1 = 1$

$A + A = A$

$A + \overline{A} = 1$

Theorem ④: Commutative law.

It states that changing the order of variables does not effect the result.

Law 1: $A + B = B + A$ Law 2: $A \cdot B = B \cdot A$

Theorem ⑤: Associative Law

It states that grouping of variables doesn't effect the result.

Law 1: $(A+B)+C = A+(B+C)$

Law 2: $(A \cdot B) \cdot C = A \cdot (B \cdot c)$

Theorem ⑥: Distributive Law.

It allows factoring (or) multiplying out of the expression.

Law1:- $A\cdot(B+C) = (A\cdot B)+(A\cdot C)$

Law2:- $A+(B\cdot C) = (A+B)\cdot(A+C)$

Theorem ⑦: Redundant literal rule (RLR)

OR ing of a variable A with the AND of the complement of that variable ($\bar{A}$) with another variable B is equal to the OR ing of ~~two~~ two variable A+B.

Law1: $A+\bar{A}\cdot B = A\overset{+}{}B$

proof: $A+\bar{A}\cdot B = (A+\bar{A})\cdot(A+B)$

$= 1\cdot(A+B)$

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

Law2: AND ing of a variable A with the OR of the complement of that variable ($\bar{A}$) with another variable B is equal to the AND ing of two variable A.B.

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

proof:- $A\cdot(\bar{A}+B) = (A\cdot\bar{A})+(A\cdot B)$

$= 0+A\cdot B$

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

Theorem ⑧: Idempotence Law.

Idempotence means same.

Law1: $A\cdot A\cdot A\cdot\ldots = A$

Law2: $A+A+A\ldots = A$

Theorem ⑨: Absorption Law

Law1: $A+AB = A$

proof: $A+A\cdot B = A(1+B)$

$= A\cdot 1$

$A+A\cdot B = A$

Law2: $A\cdot(A+B) = A$

Proof: $A\cdot(A+B) = A\cdot A+A\cdot B$

$= A+A\cdot B$

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

$= A$

Theorem ⑩: consensus theorem

Law①

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

Proof:-

$$AB + \bar{A}C + BC(1)$$

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

$$AB + \bar{A}C + BCA + BC\bar{A}$$

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

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

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

Law②: $(A+B)(\bar{A}+C)(B+C) = (A+B)(\bar{A}+C)$

Proof:- RHS

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

$$= 0 + AC + B\bar{A} + BC$$

LHS

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

$$= ABC + AC + \bar{A}B + \bar{A}BC + BC + BC$$

$$= ABC + AC + \bar{A}B + \bar{A}BC + BC$$

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

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

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

Theorem ⑪: Transposition theorm

Law1: $AB + \bar{A}C = (A+C)(\bar{A}+B)$

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

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

from consenses thorem

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

Law2: $(A+B)(\bar{A}+C) = A \cdot C + \bar{A} \cdot B$

Theorem (12): Duality theorem

(1) To find the duality of a given expression, Interchange OR sign to AND sign.

(2) Interchange AND sign to OR sign.

(3) change 0 to 1 and 1 to 0.

eg(1):- obtain the dual of the following expression

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

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

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

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

(iii) $\bar{A}B + \bar{A}B\bar{C} + \bar{A}BCD + \bar{A}B\bar{C}\bar{D}E$

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

Theorem (13): Demorgan's Theorem

Law(1): $\overline{A+B} = \bar{A} \cdot \bar{B}$

The sum of complements is equal to the product of Individual complements.

Truth Table

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

Law(2): $\overline{A \cdot B} = \bar{A} + \bar{B}$

The product of complements is equal to sum of individual complements.

Truth table:-

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

eq①: obtain complement of given Boolean expressions using demorgan's theorem.

(i) $AB'C + AB'D + A'B'eD$

(A) let $F = AB'C + AB'D + A'B'$

complement of F

$F' = (AB'C + AB'D + A'B')'$

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

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

(ii) $ABCD + ABC'D' + A'B'CD$

(A) let $F = ABCD + ABC'D' + A'B'CD$

$F' = (ABCD)' \cdot (ABC'D')' \cdot (A'B'CD)'$

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

② Find the complement of functions F_1 & F_2

$F_1 = x'yz' + x'y'z$, $F_2 = x(y'z' + yz)$

by taking their duals and complementing each literal

(A) $F_1 = x'yz' + x'y'z$

Dual of $F_1 = (x' + y + z') \cdot (x' + y' + z)$

complement of each literal.

$F_1' = (x + y' + z) \cdot (x + y + z')$

Dual of $F_2 = x(y'z' + yz)$

$F_2 = x + (y' + z') \cdot (y + z)$

Complement of each literal

$F_2' = x' + [(y + z) \cdot (y' + z')]$

Demorgan's Theorem can also be obtained by taking the Dual of given expression and complementing each literal.

Boolean Functions:-

construct any boolean expression, It requires Boolean variables & Boolean operators.

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

Literals: Each occurance of a Variable in a given boolean function in complement form (or) uncomplement form is known as literal.

eg:- Simplify the following boolean expressions to minimum number of literals?

(i) $xy + \bar{x}y + x\bar{y}$

$y(x+\bar{x}) + x\bar{y}$

$y + x\bar{y}$ $\quad$ $A+BC = (A+B)\cdot(A+C)$

$(y+x)\cdot(y+\bar{y})$

$x+y$

Minimum No. of Literals $= x+y$

(ii) $xyz + \bar{x}y + xy\bar{z}$

$xy(z+\bar{z}) + \bar{x}y$

$xy + \bar{x}y$

$y(x+\bar{x})$

y

Minimum no. of Literals $= y$

(iii) $\overline{(A+B)} \cdot \overline{(\bar{A}+\bar{B})}$

(A) $(\bar{A}\cdot\bar{B})\cdot(A\cdot B)$

$(\bar{B}\cdot\bar{A})\cdot(A\cdot B)$

$0\cdot 0 = 0$

Min no. of literals $= 0$

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

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

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

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

$B + AC$

min no. of literals $= B + AC$

(v) $A'c' + ABC + Ac' + AB'$

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

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

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

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

Min no of literals $= \bar{C} + A(\bar{B}+C)$

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

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

(vi) $(\bar{A}+C)(\bar{A}+\bar{C})(A+B+CD)$

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

$(\bar{A} + \bar{A}(C+\bar{C}) + 0)(A+B+CD)$

$(\bar{A} + \bar{A}\cdot 1)(A+B+CD)$

$\bar{A}\cdot(A+B+CD)$

$\bar{A}\cdot A + \bar{A}\cdot B + \bar{A}CD$

$0 + \bar{A}B + \bar{A}CD$

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

Min no of literals $= \bar{A}(B+CD)$

(vii) $(\overline{\bar{x}\bar{y}} + z) + z + xy + wz$

$(x+y)\cdot\bar{z} + z + xy + wz$

$(x+y)\cdot\bar{z} + xy + z(1+w)$

$(x+y)\cdot\bar{z} + z + xy$

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

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

$\bar{z}x + y(1+x) + z$

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

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

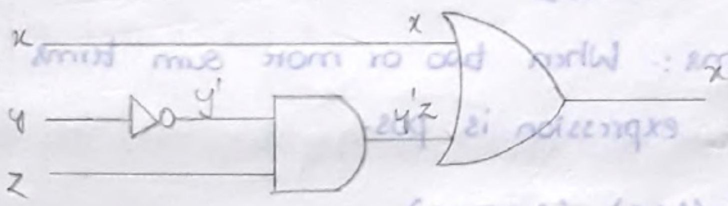
$y + x + z$

* Draw the truth table & logic diagram using basic logical gates for $F = x + y'z$

(A) Truth table:

x	y	z	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Logical gates diagram:



Canonical & standard forms:

Minterms: A minterm is a product term which contains all the literals in complemented (or) uncomplemented form. In minterm representation zero is the complement form & 1 is the true form.

Maxterm: Maxterm is a sum term which contains all the literals in complement form (or) uncomplemented form. In maxterm representation '0' is the true form & '1' is the complement form.

The minterm & maxterm for three variables is shown in table below.

x	y	z	Minterm: term	Minterm: Designation	Maxterm: term	Maxterm: Designation
0	0	0	$\bar{x}\bar{y}\bar{z}$	m_0	$x+y+z$	M_0
0	0	1	$\bar{x}\bar{y}z$	m_1	$x+y+\bar{z}$	M_1
0	1	0	$\bar{x}y\bar{z}$	m_2	$x+\bar{y}+z$	M_2
0	1	1	$\bar{x}yz$	m_3	$x+\bar{y}+\bar{z}$	M_3

1	0	0	$x\bar{y}\bar{z}$	m_4	$\bar{x}+y+z$	M_4
1	0	1	$x\bar{y}z$	m_5	$\bar{x}+y+\bar{z}$	M_5
1	1	0	$xy\bar{z}$	m_6	$\bar{x}+\bar{y}+z$	M_6
1	1	1	xyz	m_7	$\bar{x}+\bar{y}+\bar{z}$	M_7

* Any Boolean expression can be expressed in two ways

(1) sum of min terms (or) sum of products (sop)

(2) product of maxterms (or) product of sums (pos)

(1) sum of products: When two or more product terms are added the resulting expression is sop.

Eg:- $y = B + AC + A\bar{C}D$

(2) product of sums :- When two or more sum terms are multiplied the resulting expression is pos.

Eg:- $y = B \cdot (A+C) \cdot (A+\bar{C}+D)$

Standard sop (or) canonical sop:- If each term in the sop contains all the variables then the expression is known as standard (or) canonical sop.

'Σ' notation is used to represent sop Boolean expression.

Eg:- $F = ABC + A\bar{B}C + AB\bar{C}$

(or)

$F = m_7 + m_5 + m_6$

(or)

$F = \Sigma(5,6,7)$

Standard pos (or) canonical pos:- If each term in the pos contains all the variables then the expression is known as standard (or) canonical pos.

'π' notation is used to represent pos Boolean expression.

Eg:- $F = (A+B+C) \cdot (A+\bar{B}+C) \cdot (A+\bar{B}+\bar{C})$

Scanned by CamScanner

(or)

$F = M_0 \cdot M_2 \cdot M_1$

(or)

$F = \Pi(0,1,2)$

* Represent the function given truth table using
(i) standard sop (ii) standard pos.

x	y	z	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

(A) Standard sop:

$F = \bar{x}y\bar{z} + x\bar{y}\bar{z} + x\bar{y}z$ (or) $F = m_2 + m_4 + m_5$ (or) $F = \Sigma(2,4,5)$

standard pos:

$F = (x+y+z)(x+y+\bar{z})(x+\bar{y}+\bar{z})(x+\bar{y}+\bar{z})$ (or) $F = \Pi(0,1,3,6,7)$

(or) $F = M_0 M_1 M_3 M_6 M_7$.

To convert sop to standard sop :- (1) Identify missing literal then multiply with the sum of the missing literal and its complement

(2) Remove repeated terms if any.

eg: ① Express the boolean function $F = A + \bar{B}C$ as sum of minterms (or) sop form.

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

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

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

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

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

$$F = \overset{111}{ABC} + \overset{110}{AB\bar{C}} + \overset{101}{A\bar{B}C} + \overset{100}{A\bar{B}\bar{C}} + \overset{001}{\bar{A}\bar{B}C}$$

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

$$F = \Sigma(1,4,5,6,7)$$

② Represent the following Boolean expression in standard SOP

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

$$F = \bar{y}(x+\bar{x}) + xy(z+\bar{z}) + \bar{x}y\bar{z}$$

$$F = x\bar{y} + \underset{(z+\bar{z})}{\bar{x}\bar{y}} + xyz + xy\bar{z} + \bar{x}y\bar{z}$$

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

$$F = \overset{101}{x\bar{y}z} + \overset{100}{x\bar{y}\bar{z}} + \overset{001}{\bar{x}\bar{y}z} + \overset{111}{xyz} + \overset{110}{xy\bar{z}} + \overset{010}{\bar{x}y\bar{z}} + \overset{000}{\bar{x}\bar{y}\bar{z}}$$

$$F = m_5 + m_4 + m_1 + m_7 + m_6 + m_2 + m_0$$

(or)

$$F = \Sigma(0,1,2,4,5,6,7)$$

To convert POS to standard POS:

(1) Identify the missing literal and add with the product of missing literal and its complement.

(2) Remove repeated terms if any.

Eg ① Represent the following boolean expression $F = xq + \bar{x}z$ as a product of Max terms?

(A) $F = xq + \bar{x}zy$ (By transposition theorm $AB + \bar{A}C = (A+C)(\bar{A}+B)$)

$$F = (x+y)\cdot(\bar{x}+z)$$

$$F = (x+y+z\cdot\bar{z}) \cdot (\bar{x}+z+y\cdot\bar{y})$$

$$F = \overset{000}{(x+y+z)} \cdot \overset{001}{(x+y+\bar{z})} \cdot \overset{100}{(\bar{x}+y+z)} \cdot \overset{110}{(\bar{x}+\bar{y}+z)}$$

$$F = M_0 M_1 M_4 M_6$$

$$F = \Pi(0,1,4,6)$$

Scanned by CamScanner

② Express the following Boolean expression in standard pos (or) canonical pos.

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

$$= [(a+\bar{b})+(c\cdot\bar{c})]\,[b+d+a\cdot\bar{a}]\,[a+\bar{c}+(b\cdot\bar{b})]$$

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

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

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

$$\Rightarrow M_4 M_5 M_6 M_7 M_0 M_2 M_8 M_{10} M_2 M_3 M_6 M_7$$

$$\Rightarrow M_4 M_5 M_6 M_7 M_0 M_2 M_8 M_{10} M_3$$

$$\Pi(0,2,3,4,5,6,7,8,10)$$

conversion b/w canonical form:- To convert from one canonical form to another canonical form interchange the symbols Σ and Π & list the numbers missing from the original form.

Eg:- ① convert each of following to other canonical forms.

(a) $F(x,y,z) = \Sigma(2,4,5,6)$

(b) $F(x,y,z) = \Pi(0,1,3,7)$

(A) $F(x,y,z) = m_2+m_4+m_5+m_6$

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

The s. pos can be obtained as

$F(x,y,z) = \Pi(0,1,3,7)$

(b) $F(x,y,z) = M_0 \cdot M_1 M_3 M_4$

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

$$= \Pi(0,1,3,7)$$

(2) convert the following expressions into standard sum of minterms & product of maxterms.

(1) $F(A,B,C,D) = \bar{B}D + \bar{A}D + BD$

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

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

$A\bar{B}D + \bar{A}\bar{B}D + \bar{A}BD + \bar{A}\bar{B}D + ABD + \bar{A}BD$

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

$A\bar{B}D(C+\bar{C}) + \bar{A}\bar{B}D(C+\bar{C}) + \bar{A}BD(C+\bar{C}) + ABD(C+\bar{C})$

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

$m_{11} + m_9 + m_3 + m_1 + m_7 + m_5 + m_{15} + m_{13}$

$\Sigma(1,3,5,7,9,11,13,15)$

$\Rightarrow \Pi(0,2,4,6,8,10,12,14)$

$\Rightarrow M_0 M_2 M_4 M_6 M_8 M_{10} M_{12} M_{14}$

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

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

(2)(A) $[(A+\bar{B})+(C\cdot\bar{C})](A+B+\bar{C}+D\cdot\bar{D})(\bar{A}+B+C+D\bar{D})$

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

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

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

$M_4 M_5 M_6 M_7 M_2 M_3 M_8 M_9$

$\Pi(2,3,4,5,6,7,8,9)$

$\Sigma(0,1,10,11,12,13,14,15)$

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

$+ ABCD$

There are three ways to simplify boolean functions can be

(1) using Boolean Expressions (Theorems)

(2) K-map (Karnaugh map)

(3) Quine Mcclusky method

Map Method: The map method is a simple and straight forward method to simplify boolean functions. The map method is also known as k-map (or) karnaugh map method.

An n variable k-map consists of 2^n cells. 2,3,4 variable k-map for sop & pos forms are shown below.

2 Variable k-map

A \ B	0	1
0	$\bar{A}\bar{B}$ m_0	$\bar{A}B$ m_1
1	$A\bar{B}$ m_2	AB m_3

(a) sop form

A \ B	0	1
0	$A+B$ M_0	$A+\bar{B}$ M_1
1	$\bar{A}+B$ M_2	$\bar{A}+\bar{B}$ M_3

(b) pos form

3 variable k-map

A \ BC	00	01	11	10
0	$\bar{A}\bar{B}\bar{C}$ m_0	$\bar{A}\bar{B}C$ m_1	$\bar{A}BC$ m_3	$\bar{A}B\bar{C}$ m_2
1	$A\bar{B}\bar{C}$ m_4	$A\bar{B}C$ m_5	ABC m_7	$AB\bar{C}$ m_6

(a) sop form

A \ BC	00	01	11	10
0	$A+B+C$ M_0	$A+B+\bar{C}$ M_1	$A+\bar{B}+\bar{C}$ M_3	$A+\bar{B}+C$ M_2
1	$\bar{A}+B+C$ M_4	$\bar{A}+B+\bar{C}$ M_5	$\bar{A}+\bar{B}+\bar{C}$ M_7	$\bar{A}+\bar{B}+C$ M_6

(b) pos form

4 variable k-map

AB \ CD	00	01	11	10
00	$\bar{A}\bar{B}\bar{C}\bar{D}$ 0	$\bar{A}\bar{B}\bar{C}D$ 1	$\bar{A}\bar{B}CD$ 3	$\bar{A}\bar{B}C\bar{D}$ 2
01	$\bar{A}B\bar{C}\bar{D}$ 4	$\bar{A}B\bar{C}D$ 5	$\bar{A}BCD$ 7	$\bar{A}BC\bar{D}$ 6
11	$AB\bar{C}\bar{D}$ 12	$AB\bar{C}D$ 13	$ABCD$ 15	$ABC\bar{D}$ 14
10	$A\bar{B}\bar{C}\bar{D}$ 8	$A\bar{B}\bar{C}D$ 9	$A\bar{B}CD$ 11	$A\bar{B}C\bar{D}$ 10

(a) sop form

AB \ CD	00	01	11	10
00	$A+B+C+D$ 0	$A+B+C+\bar{D}$ 1	$A+B+\bar{C}+\bar{D}$ 3	$A+B+\bar{C}+D$ 2
01	$A+\bar{B}+C+D$ 4	$A+\bar{B}+C+\bar{D}$ 5	$A+\bar{B}+\bar{C}+\bar{D}$ 7	$A+\bar{B}+\bar{C}+D$ 6
11	$\bar{A}+\bar{B}+C+D$ 12	$\bar{A}+\bar{B}+C+\bar{D}$ 13	$\bar{A}+\bar{B}+\bar{C}+\bar{D}$ 15	$\bar{A}+\bar{B}+\bar{C}+D$ 14
10	$\bar{A}+B+C+D$ 8	$\bar{A}+B+C+\bar{D}$ 9	$\bar{A}+B+\bar{C}+\bar{D}$ 11	$\bar{A}+B+\bar{C}+D$ 10

(b) pos form

Note: (1) In two variable k-map each term has two adjacent terms

(2) In 3 variable k-map each term has three adjacent terms.

(3) In 4 variable k-map each term has four adjacent terms.

PAIR: Pair is formed by grouping two adjacent minterms (or) maxterms. A pair eliminates one variable in the O/p expression.

eg: Simplify the following expression using k-map.

$f(A,B,C) = \cancel{B}\ \Sigma(1,3)$

A \ BC	00	01	11	10
0	0	1 1	1 3	2
1	4	5	7	6

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

QUAD :- Quad is a group of four adjacent minterms (or) maxterms. A Quad eliminates two variables in the output expression.

eg: Simplify the boolean expression $f(A,B,C) = \Sigma(1,3,5,7)$ using K-map.

A \ BC	00	01	11	10
0	0	1 1	1 3	2
1	4	1 5	1 7	6

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

OCTET:- Octet is a group of 8 adjacent minterms (or) Maxterms. An octet eliminate three variables in the O/p expression.

eg: Simplify following expression in k-map $f(A,B,C,D) = \Sigma(0,1,2,3,4,5,6,7)$

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

$f(A,B,C,D) = \bar{A}$

Redundancy group: A redundant group is a group in which all the elements in a group are covered by some other group

ex:- Simplify the following expression using K-map

$f(A,B,C,D) = \Sigma m(1,5,6,7,11,12,13,15)$

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

Here the QUAD is redundance group. It must be eliminated

$f(A,B,C,D) = \bar{A}\bar{C}D + \bar{A}BC + AB\bar{C} + ACD$

A redundant group has to eliminated because it increases the number of gates required.

eg:- Simplify following Boolean expression

(i) $f(A,B,C) = \Sigma(0,2)$

(ii) $f(A,B,C) = \Sigma(2,3,4,5)$

(iii) $f(x,y,z) = \Sigma(3,4,6,7)$

(i)

A \ BC	00	01	11	10
0	1 (0)	1	3	1 (2)
1	4	5	7	6

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

(ii)

A \ BC	00	01	11	10
0	0	1	1 (3)	1 (2)
1	1 (4)	1 (5)	7	6

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

(iii)

x \ yz	00	01	11	10
0	0	1	1 (3)	2
1	1 (4)	5	1 (7)	1 (6)

$f(x,y,z) = x\bar{z} + yz$

(2) Simplify the boolean expression

(i) $F(x,y,z) = \Sigma(0,2,4,5,6)$

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

(a) Express function as sum of minterms.

(b) Find the minimal sum of product expression.

(A).

(i)

(a) sum of minterms:-

$$f(x,y,z) = \underset{000}{\bar{x}\bar{y}\bar{z}} + \underset{010}{\bar{x}y\bar{z}} + \underset{100}{x\bar{y}\bar{z}} + \underset{101}{x\bar{y}z} + \underset{110}{xy\bar{z}}$$

(b)

x \ yz	00	01	11	10
0	1 (0)	(1)	(3)	1 (2)
1	1 (4)	1 (5)	(7)	1 (6)

$f(x,y,z) = \bar{y}\bar{z} + x\bar{y}$

(ii) (a) sum of minterms:-

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

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

$f = \underset{(3)}{\bar{A}BC} + \underset{(1)}{\bar{A}\bar{B}C} + \underset{(2)}{\bar{A}B\bar{C}} + \underset{(5)}{A\bar{B}C} + \underset{(7)}{ABC}$

$f = \Sigma(1,2,3,5,7)$

(b)

A \ BC	00	01	11	10
0	(0)	1 (1)	1 (3)	1 (2)
1	(4)	1 (5)	1 (7)	(6)

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

(3) Simply $f(A,B,C,D) = \Sigma(0,1,2,3,9,10,11,12,14)$

(A)

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

Scanned by CamScanner

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

(4) Simplify using k-map.

(i) $f(A,B,C) = \Sigma(0,3,5,6,7)$

(ii) $f(A,B,C,D) = \Sigma(2,4,5,9,12,13)$

(iii) $f(A,B,C,D) = \Sigma(0,1,3,7,8,9,11,15)$

(iv) $f(A,B,C,D) = \Sigma(0,1,2,3,5,7,8,9,11,14)$

(v) $f(A,B,C,D) = \Sigma m(0,1,2,3,7,8,9,10,11,12,13)$

(vi) $f(w,x,y,z) = \Sigma m(0,1,2,3,11,12,14,15)$

(A)

(i)

A \ BC	00	01	11	10
0	1 (0)	(1)	1 (3)	(2)
1	(4)	1 (5)	1 (7)	1 (6)

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

(ii)

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

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

(iv)

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

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

(iii)

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

$f(A,B,C,D) = \bar{B}\bar{C} + CD$

(v)

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

$f(A,B,C,D) = \bar{B} + \bar{A}CD + A\bar{C}$

(vi)

wx \ yz	00	01	11	10
00	1 (0)	1 (1)	1 (3)	1 (2)
01	4	5	7	6
11	1 (12)	13	1 (15)	1 (14)
10	8	9	1 (11)	10

$f(w,x,y,z) = \bar{w}\bar{x} + wx\bar{z} + wyz$

Product of sum Simplification:-

(1) Simplify the following expression to pos.

(i) $f(A,B,C) = \Pi(0,2,5,7)$

(ii) $f(A,B,C,D) = \Pi(1,3,5,7,12,13,14,15)$

(i)
(A)

A \ BC	00	01	11	10
0	0 (0)	1	3	0 (2)
1	4	0 (5)	0 (7)	6

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

(ii)
(A)

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

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

(2) obtain the minimal pos expression

(i) $f(A,B,C,D) = \Pi M(2,8,9,10,11,12,14)$

(ii) $f(A,B,C,D) = (A+\bar{B}+\bar{D})(A+\bar{B}+\bar{C})(\bar{A}+\bar{C}+\bar{D})(\bar{A}+\bar{B}+D)(A+B+\bar{C}+\bar{D})$

(i)

(A)

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

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

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

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

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

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

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

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

$\Rightarrow f(A,B,C,D) = \Pi(3, 5, 6, 7, 11, 12, 15, 14)$

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

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

Minimal expression:-

The Boolean expression that can't be simplified further is called minimal expression. The minimal expression contains minimum number of literals, The minimal expression is not unique. There can be more than one minimal expression for a given boolean expression.

Prime implicants (PI) & False prime implicants (FPI):-

Each square or rectangle formed by grouping of adjacent minterms is called PI and grouping of Maxterms is called FPI.

NON-prime Implicant (NPI) & NON-False prime implicant (NFPI):-

A group formed by one minterm is NPI. And one maxterm is called NFPI

Essential prime Implicant (EPI) & Essential false prime implicant (EFPI)

A group (or) prime Implicant which contains atleast one '1' which can't be covered by any other group (or) PI is called EPI.

A group (or) FPI which contains atleast one '0' which can't be covered by any other group (or) FPI is called EFPI.

Redundant prime Implicant (RPI) & Redundant false prime implicant (RFPI):-

All '1's in a group (or) PI are covered by some other group (or) PI is called RPI. All '0's in a group (or) FPI are covered by some other group (or) FPI is called RFPI

Selective prime implicant (SPI) & Selective false prime implicant (SFPI):

A group (or) PI which is neither an EPI nor RPI is called SPI. A group (or) FPI which is neither an EFPI nor RFPI is called as SFPI

Eg① Find all the PI for following boolean functions & determine which are essential:

(A) (i) $f(\omega, x, y, z) = \Sigma(0, 2, 4, 5, 6, 7, 8, 10, 13, 15)$

(ii) $f(\omega, x, y, z) = \Sigma(0, 2, 3, 5, 7, 8, 10, 11, 14, 15)$

(iii) $f(\omega, x, y, z) = \Sigma(1, 3, 4, 5, 10, 11, 12, 13, 14, 15)$

(iv) $f(\omega, x, y, z) = \Sigma(0, 2, 3, 5, 7, 8, 9, 10, 11, 13, 15)$

(i)

wx \ yz	00	01	11	10
00	1 (0)	(1)	(3)	1 (2)
01	1 (4)	1 (5)	1 (7)	1 (6)
11	(12)	1 (13)	1 (15)	(14)
10	1 (8)	(9)	(11)	1 (10)

(0, 2, 8, 10) - EPI = $\bar{x}\bar{z}$

(0, 2, 4, 6) - SPI-1 = $\bar{w}\bar{z}$

(5, 7, 13, 15) - EPI = xz

(4, 5, 7, 6) - SPI 2 = $x\bar{w}$

$f(w,x,y,z)$ = EPI + EPI + SPI-1 (or) SPI-2

= $\bar{x}\bar{z} + xz + \bar{w}\bar{z}$ (or) $\bar{x}\bar{z} + xz + x\bar{w}$

(ii)

wx \ yz	00	01	11	10
00	1 (0)	(1)	1 (3)	1 (2)
01	(4)	1 (5)	1 (7)	(6)
11	(12)	(13)	1 (15)	1 (14)
10	1 (8)	(9)	1 (11)	1 (10)

(0, 2, 8, 10) - EPI = $\bar{x}\bar{z}$

(3, 7, 11, 15) - SPI = yz

(or) (10, 14, 15, 11) - EPI = wy

(5, 7) - EPI = $\bar{w}xz$

(8, 2, 11, 10) - SPI = $\bar{x}y$

$f(w,x,y,z) = \bar{x}\bar{z} + wy + \bar{w}xz + yz$ (or) $\bar{x}\bar{z} + wy + \bar{w}xz + \bar{x}y$

(iii)

wx \ yz	00	01	11	10
00	(0)	1 (1)	1 (3)	(2)
01	1 (4)	1 (5)	(7)	(6)
11	1 (12)	1 (13)	1 (15)	1 (14)
10	(8)	(9)	1 (11)	1 (10)

(1, 3) - EPI = $\bar{w}\bar{x}z$

(4, 5, 12, 13) - EPI = $x\bar{y}$

(10, 11, 14, 15) - EPI = wy

$f(w,x,y,z) = \bar{w}\bar{x}z + x\bar{y} + wy$

(iv)

wx \ yz	00	01	11	10
00	1 (0)	(1)	1 (3)	1 (2)
01	(4)	1 (5)	1 (7)	(6)
11	(12)	1 (13)	1 (15)	(14)
10	1 (8)	1 (9)	1 (11)	1 (10)

(0, 2, 10, 8) - EPI = $\bar{x}\bar{z}$

(5, 7, 13, 15) - EPI = xz

(8, 9, 10, 11) = SPI = $w\bar{x}$

(or)

(9, 11, 13, 15) = SPI = wz

(2, 3, 10, 11) = SPI - $\bar{x}y$

(or)

(3, 7, 15, 11) = SPI - yz

$f(w,x,y,z) = \bar{x}\bar{z} + xz + w\bar{x} + \bar{x}y$

(or)

$= \bar{x}\bar{z} + xz + w\bar{x} + yz$

(or)

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

(or)

$= \bar{x}\bar{z} + xz + wz + yz$

② Reduce using k-map & Identify false prime implicants & EFPI.

(i) $f(A,B,C,D) = \prod M(5,6,7,9,10,11,13,14,15)$

(ii) $f(A,B,C,D) = \prod M(0,1,2,6,8,10,11,12)$

(A) (i)

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

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

EFPI = FPI = (5,7,13,15), (7,6,14,15), (9,11,13,15), (10,11,14,15)

(ii)

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

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

EFPI = FPI = (0,1), (8,12), (10,11), (2,6)

Don't care conditions: The combinations for which the value of the function is not specified is known as don't care condition. The value of the function for such combination is denoted by ϕ (or) x (or) d. In choosing the adjacent squares to simplify the function we assign the value 1 to the don't care combination and the value 0 to others in such a way to increase the size of the selected sub cubes whenever possible. No subcube containing only don't care may be formed because it is not required that the function is equal to 1 for such combinations.

Eg:- ① simplify the following using k-map.

$$f(w,x,y,z) = \Sigma(1,3,7,11,15) + \Sigma_d(0,2,5)$$

(A)

wx \ yz	00	01	11	10
00	X (0)	1 (1)	1 (3)	X (2)
01	(4)	X (5)	1 (7)	(6)
11	(12)	(13)	1 (15)	(14)
10	(8)	(9)	1 (11)	(10)

$$f(w,x,y,z) = \bar{w}z + yz$$

② Simplify following using k-map

(i) $f(w,x,y,z) = \Sigma(1,3,10) + \Sigma_d(0,2,8,12)$

(ii) $f(w,x,y,z) = \Sigma(0,6,8,13,14) + \Sigma_d(2,4,10)$

(iii) $f(A,B,C,D) = \Sigma(4,5,6,7,12,13,14) + \Sigma_d(1,9,11,15)$

(ii)

wx \ yz	00	01	11	10
00	1 (0)	(1)	(3)	X=1 (2)
01	X (4)	(5)	(7)	1 (6)
11	(12)	1 (13)	(15)	1 (14)
10	1 (8)	(9)	(11)	X=1 (10)

$$f(w,x,y,z) = y\bar{z} + \bar{x}\bar{z} + wx\bar{y}z$$

(i)

$wx \backslash yz$	00	01	11	10
00	X (0)	1 (1)	1 (3)	X (2)
01	(4)	(5)	(7)	(6)
11	X (12)	(13)	(15)	(14)
10	X (8)	(9)	(11)	1 (10)

$f(w,x,y,z) = \bar{w}\bar{x} + \bar{x}\bar{z}$

(iii)

$AB \backslash CD$	00	01	11	10
00	(0)	X (1)	(3)	(2)
01	1 (4)	1 (5)	1 (7)	1 (6)
11	1 (12)	1 (13)	X (15)	1 (14)
10	(8)	X (9)	X (11)	(10)

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

③ obtain minimal expression for (i) $F = \Pi M(1,2,3,8,9,10,11,14) \cdot d(7,15)$

(ii) $F(a,b,c,d) = \Pi M(6,7,8,9) \cdot d(10,11,12,13,14,15)$

(A)

(i)

$AB \backslash CD$	00	01	11	10
00	(0)	0 (1)	0 (3)	0 (2)
01	(4)	(5)	d (7)	(6)
11	(12)	(13)	d=0 (15)	0 (14)
10	0 (8)	0 (9)	0 (11)	0 (10)

Check for redundance group not considering don't care.

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

(ii)

$ab \backslash cd$	00	01	11	10
00	(0)	(1)	(3)	(2)
01	(4)	(5)	0 (7)	0 (6)
11	d=0 (12)	d=0 (13)	d=0 (15)	d=0 (14)
10	0 (8)	0 (9)	d=0 (11)	d=0 (10)

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

Other minimization Methods:

Quine Mcclusky method / Tabular Method:-

Limitations of k-map:-

(1) k-map is suitable to solve the boolean functions upto 5 Variables.

(2) Six variable k-map is difficult to visualized using k-map.

so tabular method (or) Quine Mcclusky method is adopted to Solve large variable k-maps.

Tabular Method:-

(1) Represent Minterms (or) Maxterms in binary form.

(2) Arrange the minterms (or) Maxterms as per the grouping.

(3) compare each group with next higher group, If there is one variable difference put '_' in that position and write them in next column.

(4) Repeat the comparision process till no matches were found

(5) Draw the prime Implicant chart. prime Implicant chart gives the relation ship b/w prime Implicants & minterms & maxterms given in boolean expression.

Eg① simplify $f = \Sigma m(0,2,3,6,7,8,10,12,13)$ using tabular method?

Minterms	Binary equivalent	Grouping as per 1's	Forming duals	forming Quads	
m_0	0000	0000(0)	(0,2) 00-0	(0,2,8,10) -0-0	(0,2,8,10) -0-0
m_2	0010	0010(2)	(0,8) -000	(0,8,2,10) -0-0	
m_3	0011	1000(8)	(2,3) 001-	(2,3,6,7) 0-1-	(2,3,6,7) 0-1-
m_6	0110	0011(3)	(2,6) 0-10	(2,6,3,7) 0-1-	
m_7	0111	0110(6)	(2,10) -010		
m_8	1000	1010(10)	(8,10) 10-0		
m_{10}	1010	1100(12)	(8,12) 1-00 PI		
m_{12}	1100	0111(7)	(3,7) 0-11		
m_{13}	1101	1101(13)	(6,7) 011-		
			(12,13) 110- PI		

Prime implicant chart :-

minterms/PI	0	2	3	6	7	8	10	12	13
(8, 12)						X		X	
(12, 13)								X	X
(0, 2, 8, 10)	X	X				X	X		
(2, 3, 6, 7)		X	X	X	X				

$(12,13) = 110- = AB\bar{C}$

$(0,2,8,10) = -0-0 = \bar{B}\bar{D}$

$(2,3,6,7) = 0-1- = \bar{A}C$

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

Simplify the given function using mcclusky method

$F(A,B,C,D) = \Sigma(0,2,3,6,7) + d(5,8,10,11,15)$

Minterms	Binary equivalent	Grouping as per (1's)	Forming duals	Forming Quads	
m_0	0000	0000 (0)	(0,2) 00-0	(0,8,2,10) -0-0	(0,2,8,10) → PI -0-0
m_2	0010	0010 (2)	(0,8) -000	(2,3,6,7) 0-1-	
m_3	0011	1000 (8)	(2,3) 001-	(0,2,8,10) -0-0	
m_5	0101	0011 (3)	(2,6) 0-10	(2,3,10,11) -01-	(2,3,6,7) → PI 0-1-
m_6	0110	0101 (5)	(2,10) -010	(2,6,3,7) 0-1-	
m_7	0111	0110 (6)	(8,10) 10-0	(2,10,3,11) -01-	(2,3,10,11) → PI -01-
m_8	1000	1010 (10)	(3,7) 0-11	(3,11,7,15) --11	
m_{10}	1010	0111 (7)	(3,11) -011	(3,7,11,15) --11	(3,7,11,15) --11 → PI
m_{11}	1011	1011 (11)	(5,7) 01-1 (PI)		
m_{15}	1111	1111 (15)	(6,7) 011-		
			(10,11) 101-		
			(7,15) -111		
			(11,15) 1-11		

Prime Implicant chart

Don't care terms are not listed

minterms / PI	0	2	3	6	7
(5, 7)					X
(0, 2, 8, 10) ✓	X	X			
(2, 3, 6, 7) ✓		X	X	X	X
(2, 3, 10, 11)		X	X		
(3, 7, 11, 15)			X		X

$(0, 2, 8, 10) \rightarrow -0-0 = \bar{B}\bar{D}$, $(2, 3, 6, 7) \rightarrow 0-1- = \bar{A}C$

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

Five variable k-map:-

Five variable k-map can be obtained by using two 4 variable k-maps. Assuming one 4 variable k-map as zero (A = 0). And another 4 variable k-map as A = 1.

eg:- Simplify the boolean expression $F(A,B,C,D,E) = \Sigma m(0,1,5,6,9,13,14,17,21,22,25,29)$ using k-map.

$\bar{A} = 0$

BC \ DE	00	01	11	10
00	1 (0)	1 (1)	3	2
01	4	1 (5)	7	1 (6)
11	12	1 (13)	15	1 (14)
10	8	1 (9)	11	10

A = 1

BC \ DE	00	01	11	10
00	16	1 (17)	19	18
01	20	1 (21)	23	1 (22)
11	28	1 (29)	31	30
10	24	1 (25)	27	26

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

(2) Simplify $F = \Sigma m(0,2,4,6,9,13,21,25,29,31)$ using k-map

$\bar{A}=0$

BC \ DE	00	01	11	10
00	1 (0)	(1)	(3)	1 (2)
01	1 (4)	(5)	(7)	1 (6)
11	(12)	1 (13)	(15)	(14)
10	(8)	1 (9)	(11)	(10)

$A=1$

BC \ DE	00	01	11	10
00	(16)	(17)	(19)	(18)
01	(20)	1 (21)	(23)	(22)
11	(28)	1 (29)	1 (31)	(30)
10	(24)	1 (25)	(27)	(26)

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

(3) simplify $F = \Sigma m(6,9,13,18,19,25,29,27,31) + d(2,3,14,15,17,24,28)$ using k-map?

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

BC \ DE	00	01	11	10
00	(0)	(1)	X=1 (3)	X=1 (2)
01	(4)	(6)	(7)	1 (6)
11	(12)	1 (13)	X=1 (15)	(14)
10	(8)	1 (9)	X=1 (11)	(10)

$A=1$

BC \ DE	00	01	11	10
00	(16)	X (17)	1 (19)	1 (18)
01	(20)	(21)	(23)	(22)
11	X (28)	1 (29)	1 (31)	(30)
10	X (24)	1 (25)	1 (27)	(26)

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

(4) Simplify $F = \Pi M(3,4,7,11,15,19,21,23,27,28,29,31)$

(A) $A=0$

BC \ DE	00	01	11	10
00	(0)	(1)	0 (3)	(2)
01	0 (4)	(5)	0 (7)	(6)
11	(12)	(13)	0 (15)	(14)
10	(8)	(9)	0 (11)	(10)

$\bar{A}=1$

BC \ DE	00	01	11	10
00	(16)	(17)	0 (19)	(18)
01	(20)	0 (21)	0 (23)	(22)
11	0 (28)	0 (29)	0 (31)	(30)
10	(24)	(25)	0 (27)	(26)

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

Digital Logic gates:-

(1) Digital logic gates are the basic building blocks of hardware circuits.

(2) Digital logic gates can have more than one input and only one output.

NOT:-

Statement:- NOT gate gives the complement of the input that is if input is 0 then o/p is 1 & If I/p is 1 then o/p is 0.

Symbol:- A  Ā (or) A'

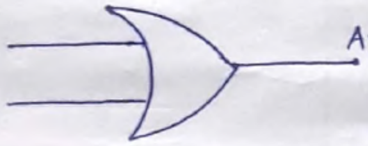
Truth Table:-

I/p	o/p
A	Ā
0	1
1	0

OR :-

Statement:- If any one input is high (or) 1 then output is high (or) 1.

If both inputs are high (or) 1 then o/p is high (or) 1.

Symbol :- 

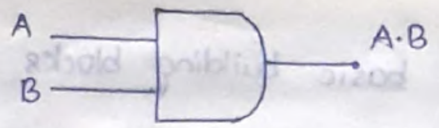
Truth Table :-

I/p		o/p
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

AND :-

Statement:- If both I/p are high o/p is high. When any one of the I/p is low then o/p is low.

Symbol:-



Truth Table :-

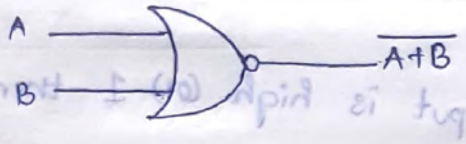
I/p		O/p
A	B	A·B
0	0	0
0	1	0
1	0	0
1	1	1

NOR :- (OR+NOT)

Statement :- When both I/p are low then o/p is high.

When any one of the I/p is high then o/p is low.

Symbol :-



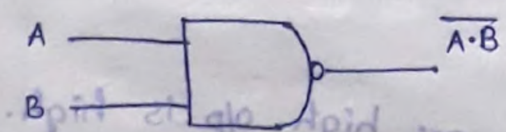
Truth Table :-

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

NAND :- (AND+NOT)

Statement :- When both I/p are high then o/p is low. If any one of the I/p is low then o/p is high.

Symbol:-



Truth Table:-

I/P		O/P
A	B	$\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

~~EX-OR~~

XOR :- (EX-OR) [Exclusive OR gate]

Statement :- If both I/P are same O/P is 0. When both Inputs are difference then O/P is 1.

Symbol:-

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

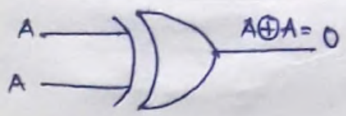
Truth Table :-

I/P		O/P
A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Properties of XOR gate:-

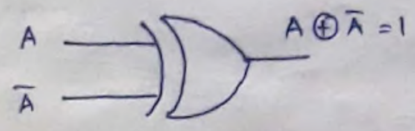
(1) $A \oplus A = 0$

When both I/P are same O/P is low

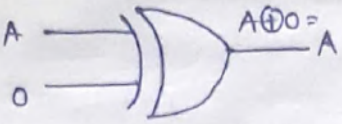


(2) When both I/P are different O/P is high

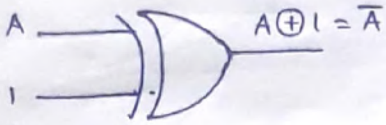
$A \oplus \bar{A} = 1$



(3) $A \oplus 0 = A$ (NON - Inverter property)



(4) $A \oplus 1 = \bar{A}$ (Inverting property)



(5)
* XOR acts as a 2 bit module adder.

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

2 bit adder

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

(6) $AB \oplus AC = A(B \oplus C)$

proof :-

LHS :- $AB \oplus AC$

$\overline{AB}(AC) + AB(\overline{AC})$

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

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

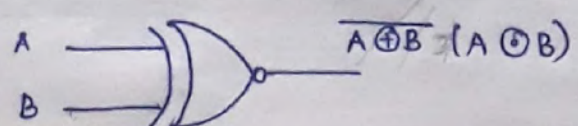
$A\bar{B}C + AB\bar{C}$

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

$\Rightarrow A(B \oplus C) = RHS$

XNOR [Exclusive NOR gate] :- When both i/p are same o/p is 1. When both i/p are different o/p is 0.

Symbol:- XNOR = XOR + NOT



Truth Table :- I/P O/P

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

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

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

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

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

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

$$= A\odot B$$

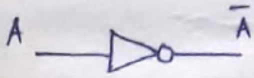
Universal Gates :-

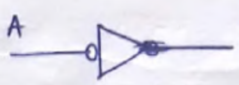
NAND & NOR gates are called as universal gates. Because any other logic function can be implemented by using these gates.

Tricky logic Method :-

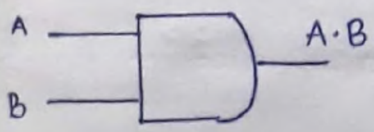
Tricky method gates are obtained by changing AND gate to OR gate and OR gate to AND gate. placing the bubble when there is no bubble, removing the bubble when it is present.

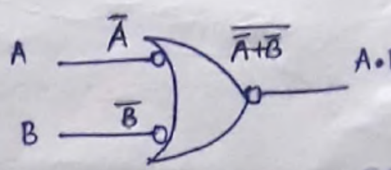
NOT



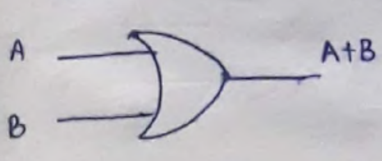


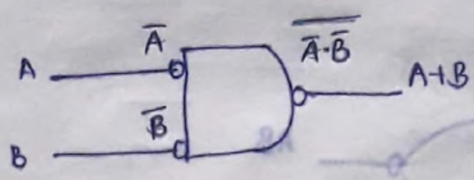
AND



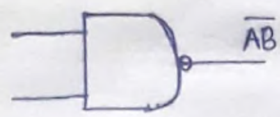


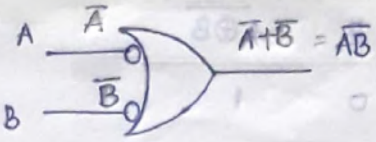
OR



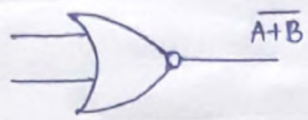


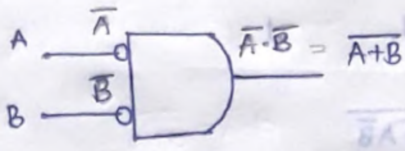
NAND





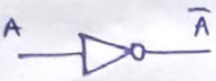
NOR

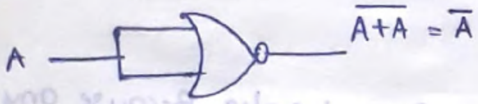




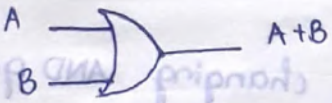
Implementation of other gates by using NOR :-

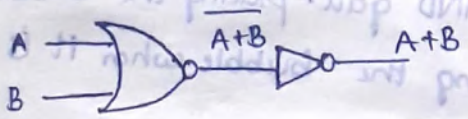
(1) Not by using NOR:-

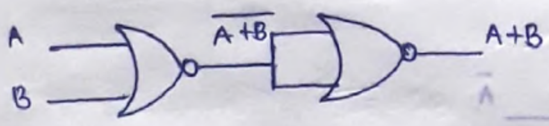




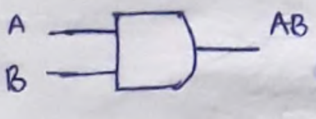
(2) OR by using NOR :-

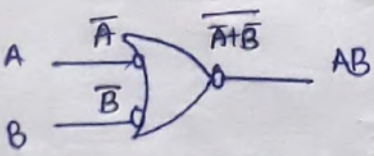


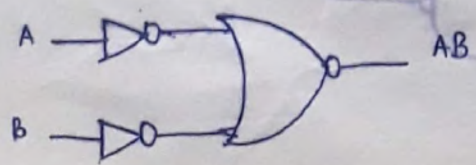


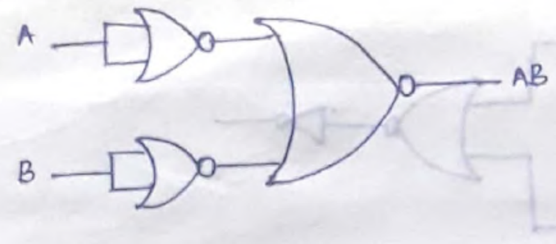


(3) AND by using NOR:-

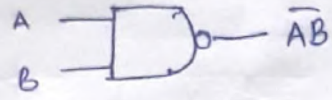


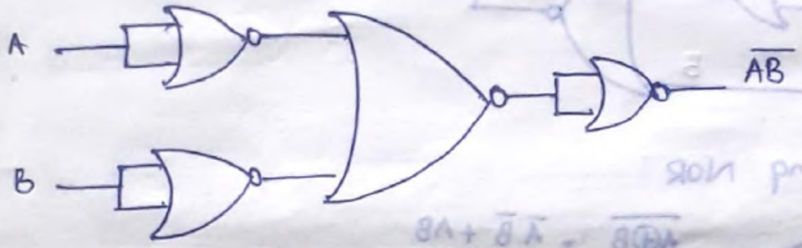






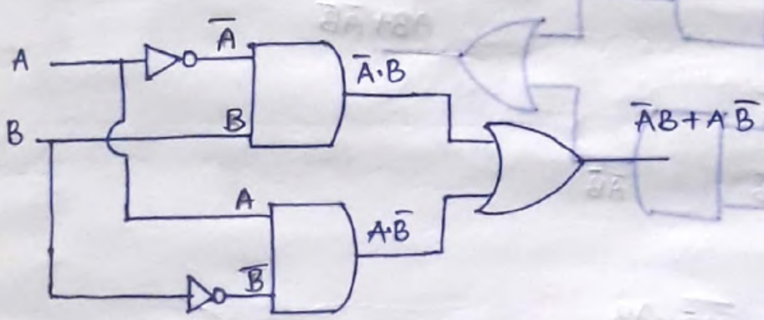
(4) NAND by using NOR :-

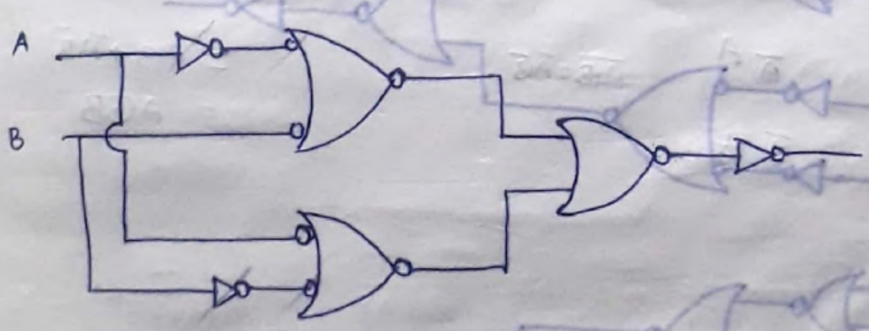


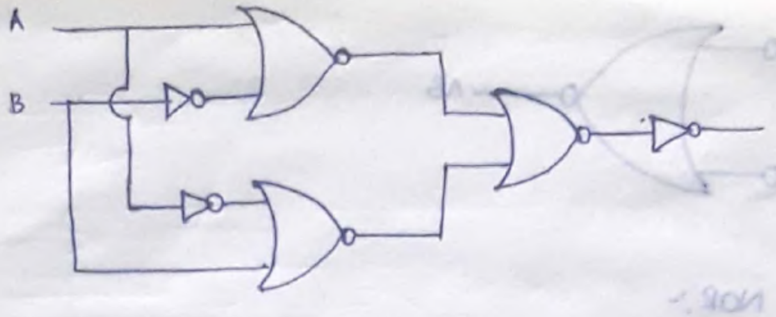


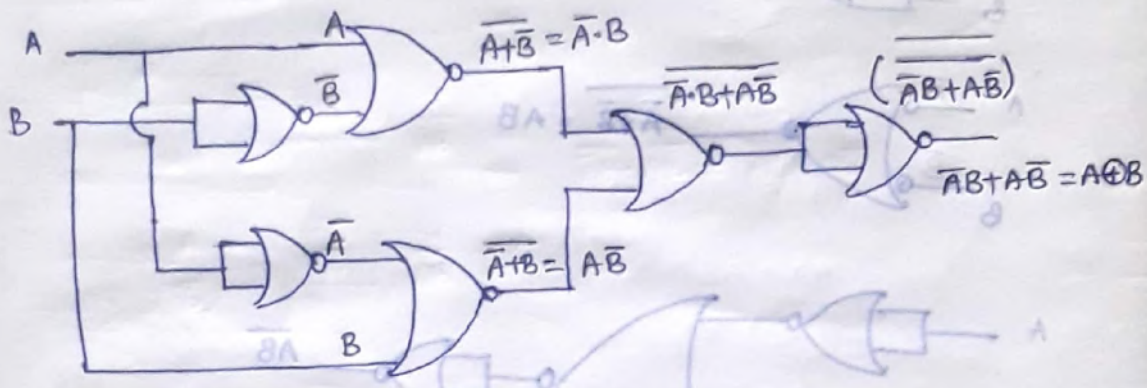
(5) EX-OR by using NOR :- $A \oplus B = \overline{A} \cdot B + A \cdot \overline{B}$

$A \oplus B =$

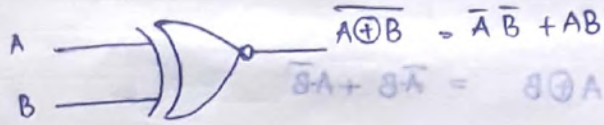


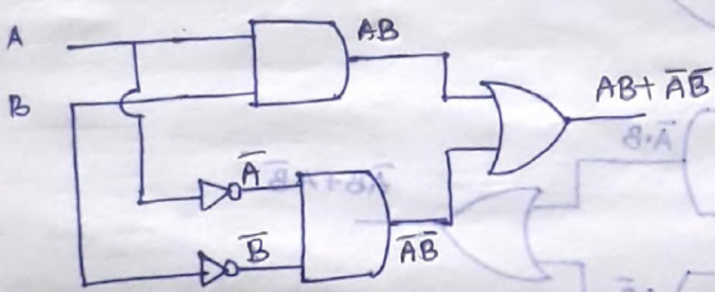


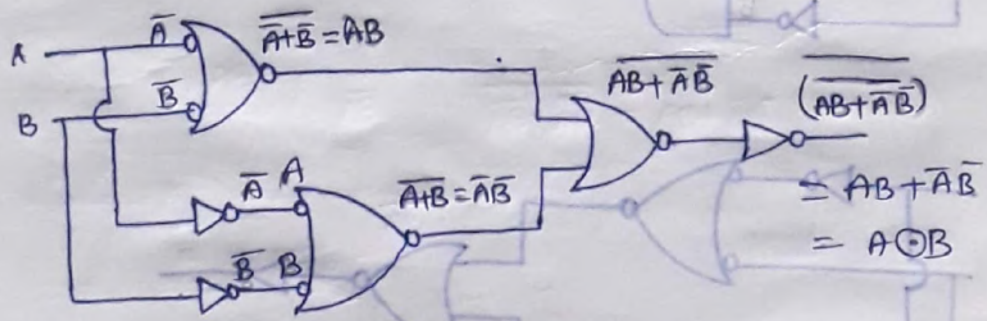


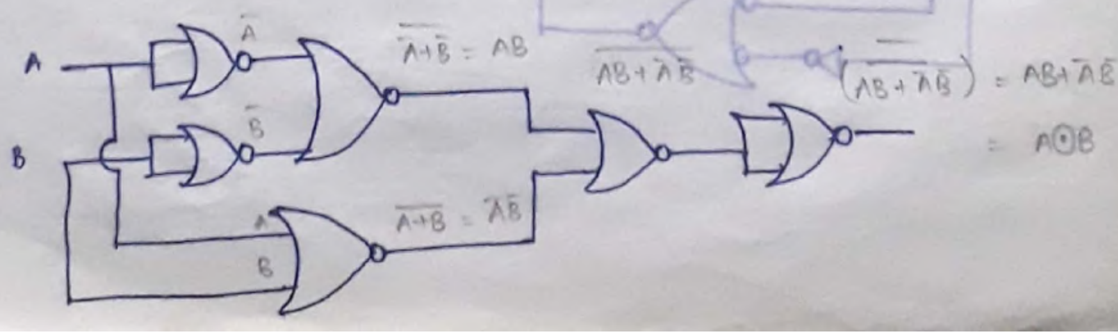


(6) Ex-NOR by using NOR



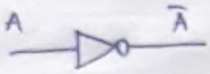


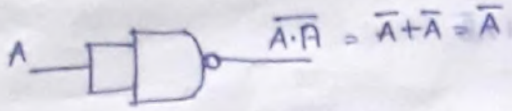




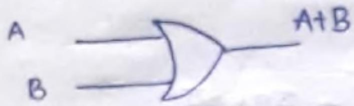
Implementation of other gates by using NAND

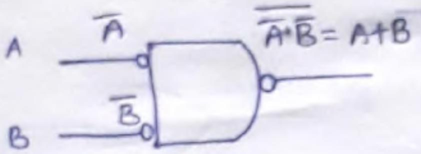
(1) NOT by using NAND

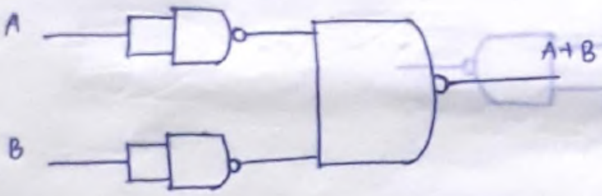




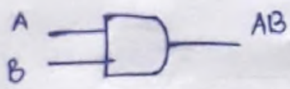
(2) OR by using NAND

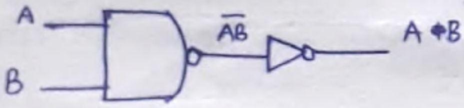


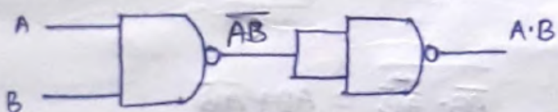




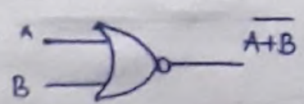
(3) AND by using NAND

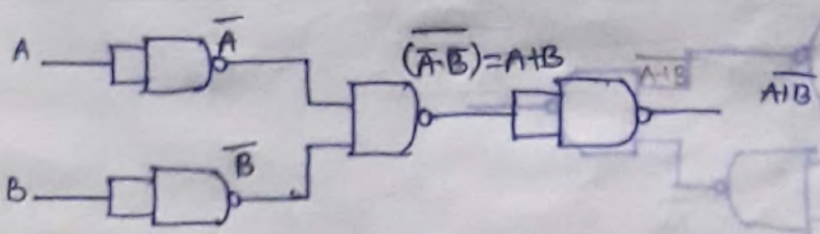




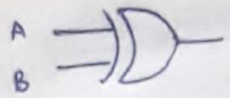


(4) NOR byusing NAND

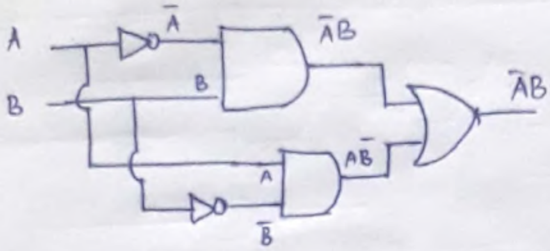




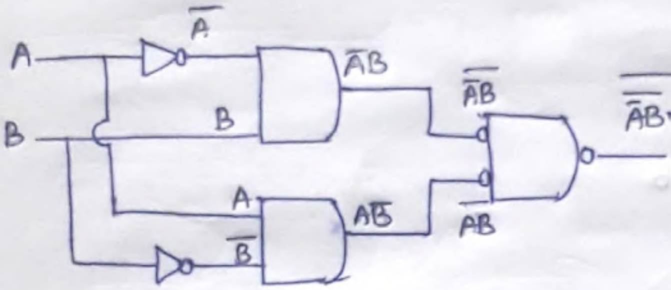
(5) XOR by using NAND



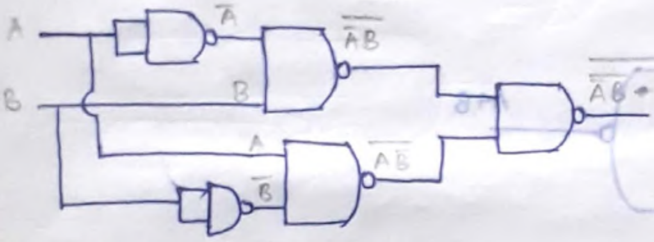
$A \oplus B = \bar{A}B + A\bar{B}$



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



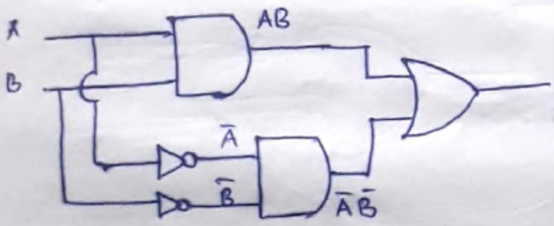
$\overline{\overline{\bar{A}B} \cdot \overline{A\bar{B}}} = \bar{A}B + A\bar{B}$



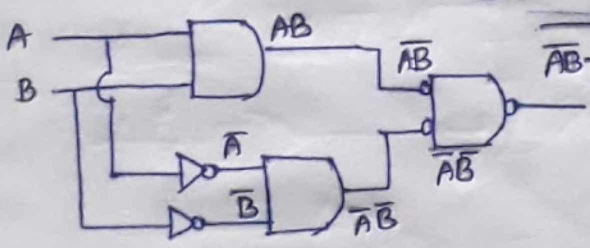
$\overline{\overline{\bar{A}B} \cdot \overline{A\bar{B}}} = \bar{A}B + A\bar{B}$

(6) EX-NOR by using NAND.

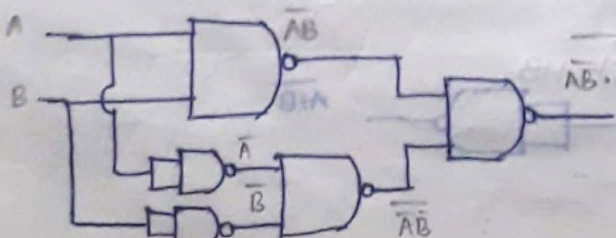
$\overline{A \oplus B} = \bar{A}\bar{B} + AB$



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



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



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

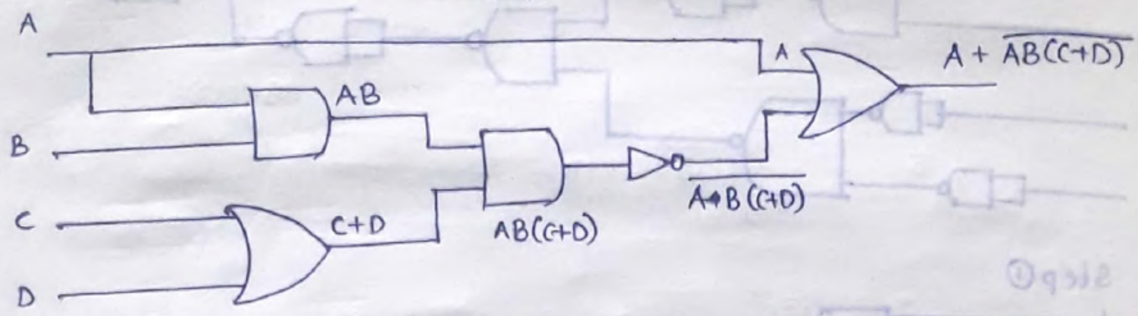
(i) Implement the expression using gates and universal gates.

$A + \overline{AB(C+D)}$

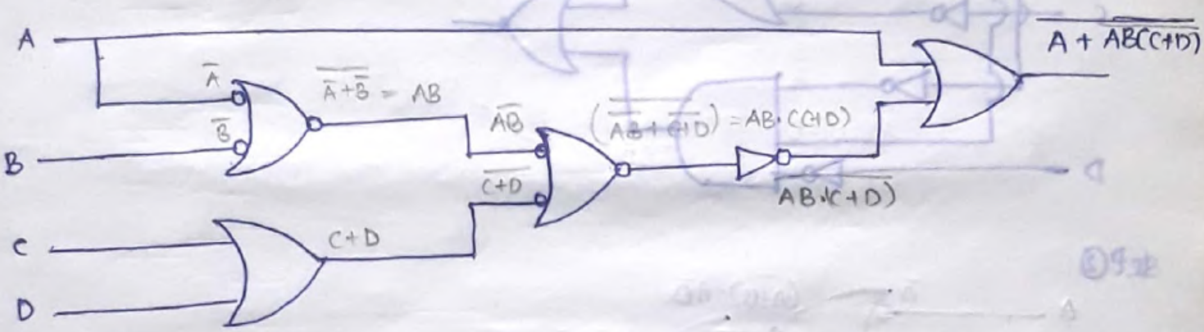
(ii) Implement using universal gates $F = \bar{C} + A\bar{B} + \bar{A}B\bar{D}$

(i) step① Implementation using gates

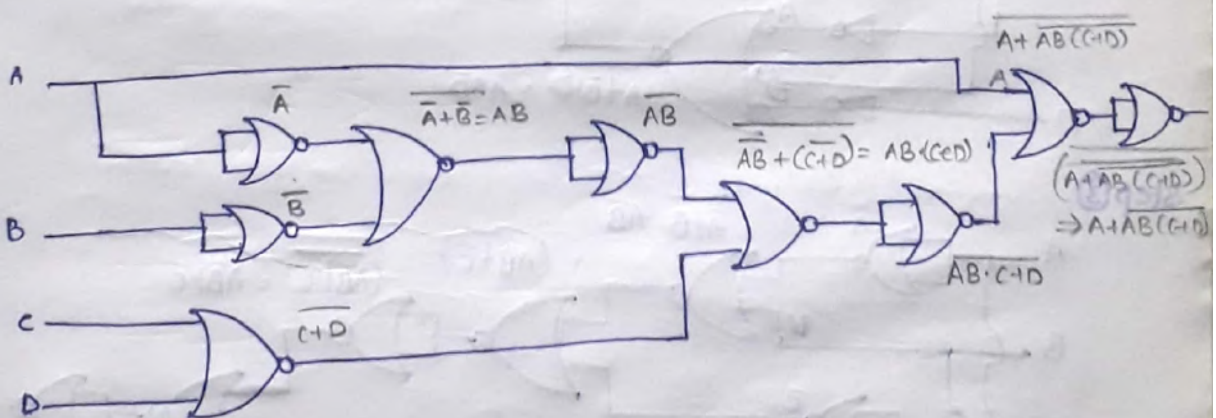
(A)



step② change AND by using tricky method

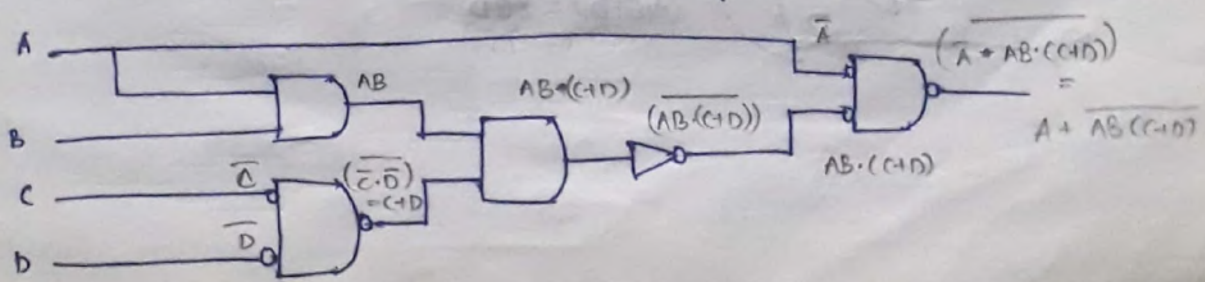


Step③ Add (or) remove bubbles to get the given expression in the form of NOR gates. and convert NOT in terms of NOR.



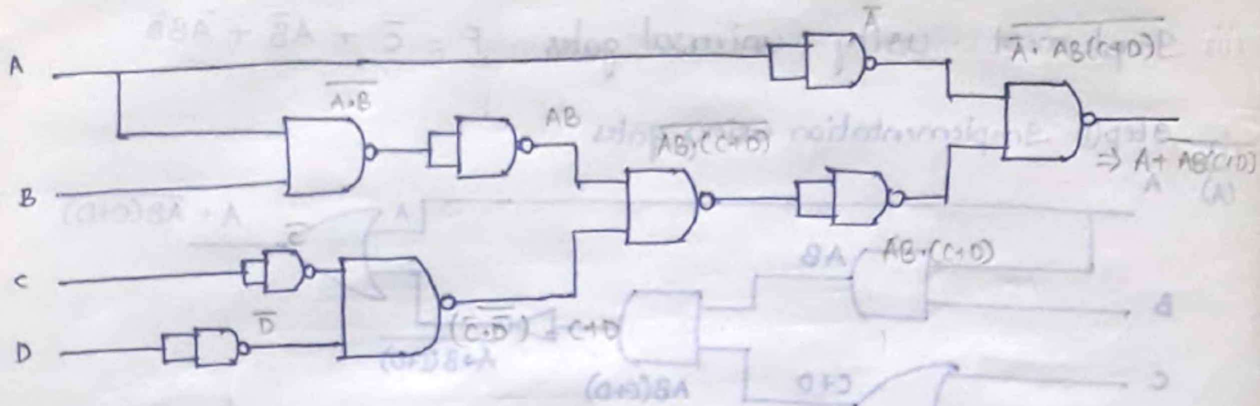
step① Implementation using NAND gates

change OR gates in step① by using tricky method.

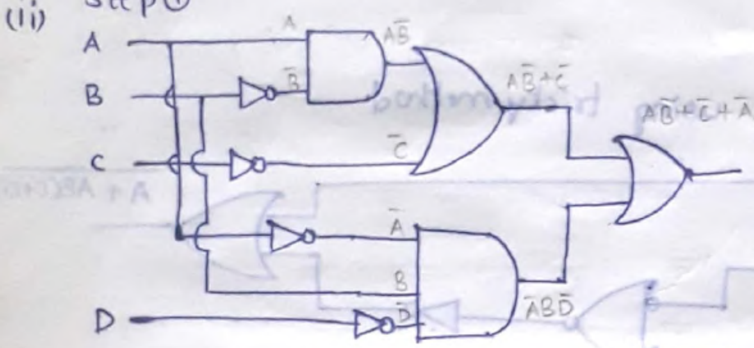


Scanned by CamScanner

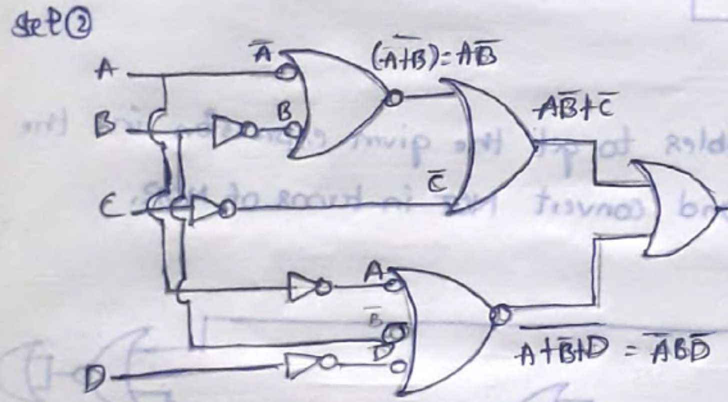
step②:- Add (or) remove bubbles to get the given expression in form of NAND gates and convert NOT in terms of NAND.



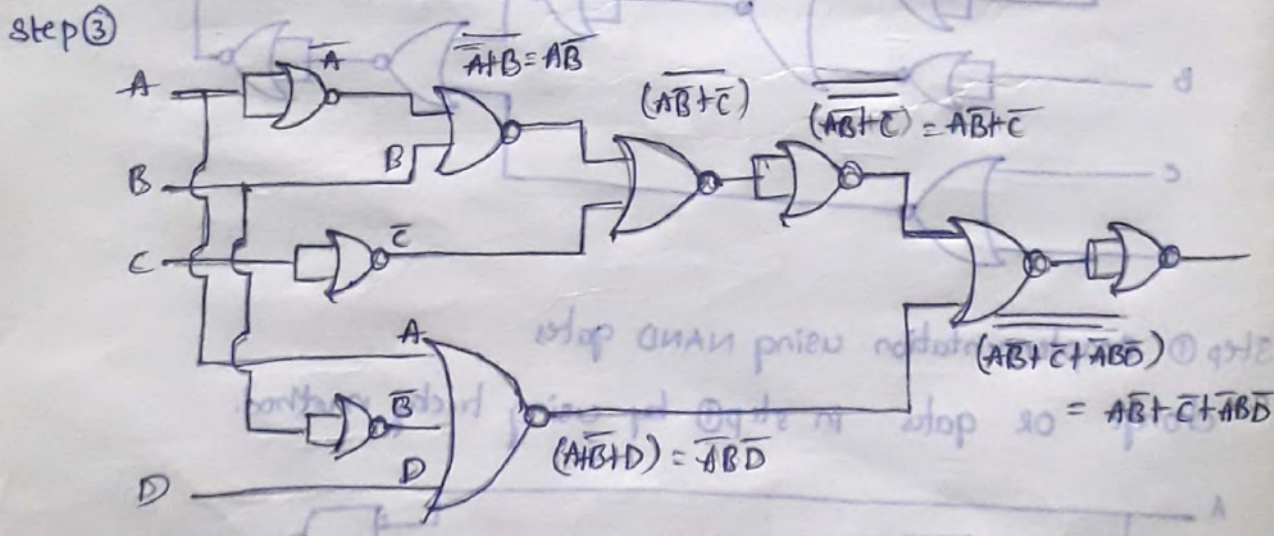
(ii) step①



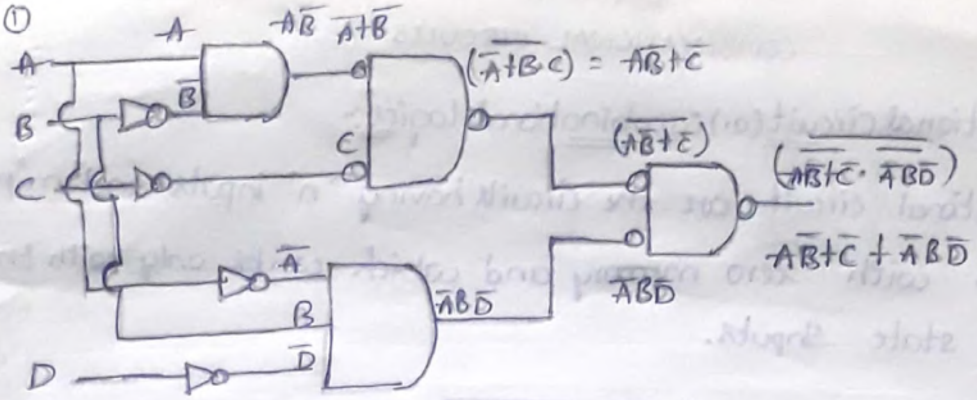
step②



step③

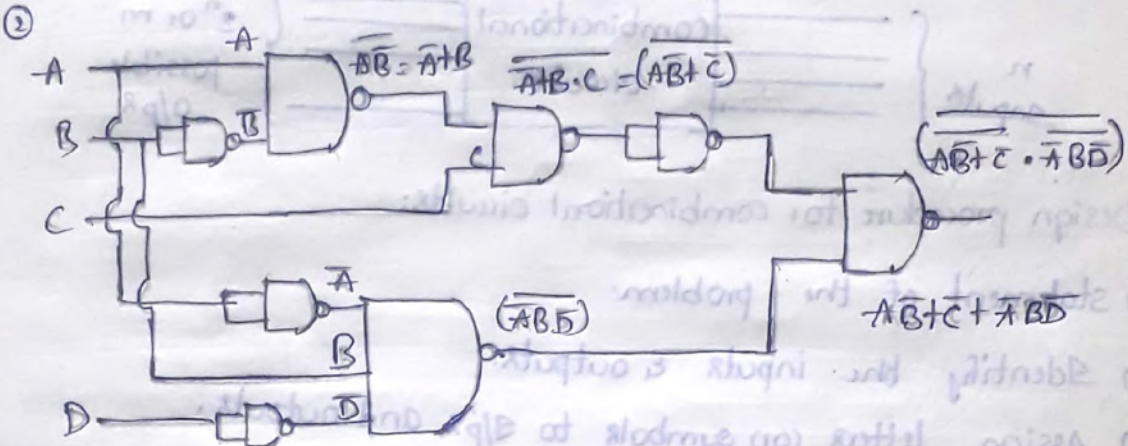


step ①



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

step ②



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

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

$f(A,B,C,D) = \bar{B}\bar{C}\bar{D}$

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

$f(A,B,C,D) = \bar{B}\bar{C}D$

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

$f(A,B,C,D) = \bar{B}C\bar{D}$

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

$f(A,B,C,D) = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}BCD$

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

$f(A,B,C,D) = \bar{A}B\bar{C}D + ABCD$

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

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

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

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

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

$f(A,B,C,D) = \bar{A}\bar{B}CD + ABCD$

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

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

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

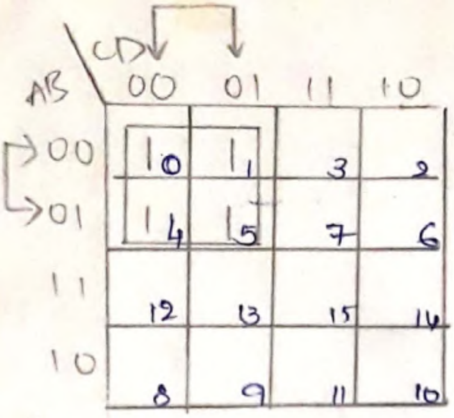
$f(A,B,C,D) = \bar{A}\bar{B}C\bar{D} + ABC\bar{D}$

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

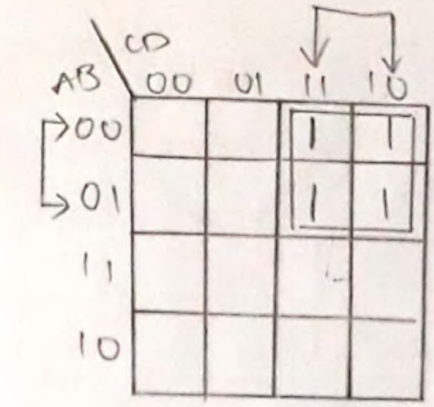
$f(A,B,C,D) = \bar{A}B\bar{C}\bar{D} + \bar{A}BCD$

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

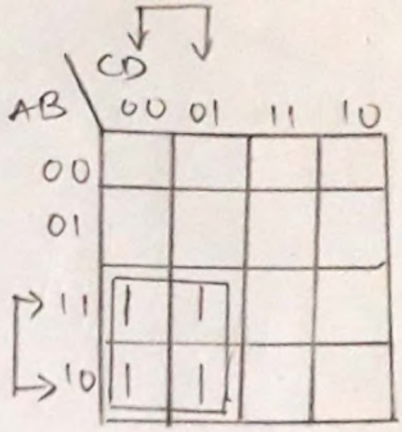
$f(A,B,C,D) = \bar{A}\bar{B}C\bar{D} + AB\bar{C}D$



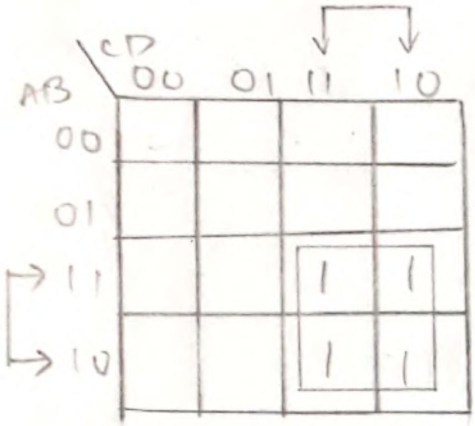
$f(A,B,C,D) = \bar{A}\bar{C}$



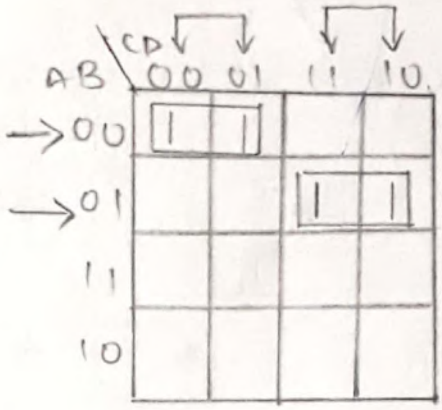
$f(A,B,C,D) = \bar{A}C$



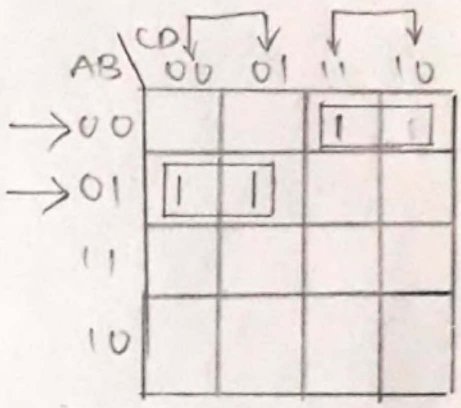
$f(A,B,C,D) = A\bar{C}$



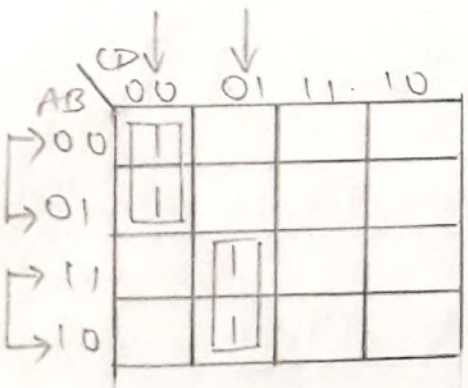
$f(A,B,C,D) = AC$



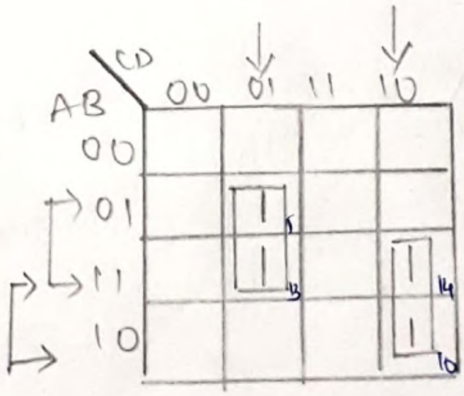
$f(A,B,C,D) = \bar{A}\bar{B}\bar{C} + \bar{A}BC$



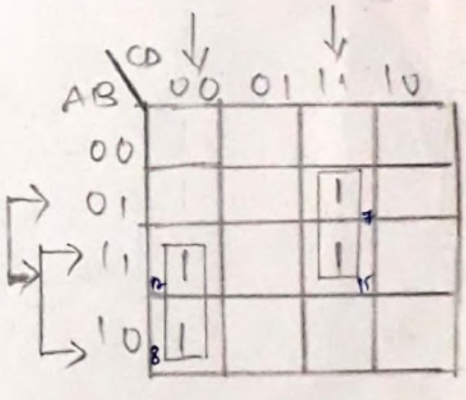
$f(A,B,C,D) = \bar{A}\bar{B}C + \bar{A}B\bar{C}$



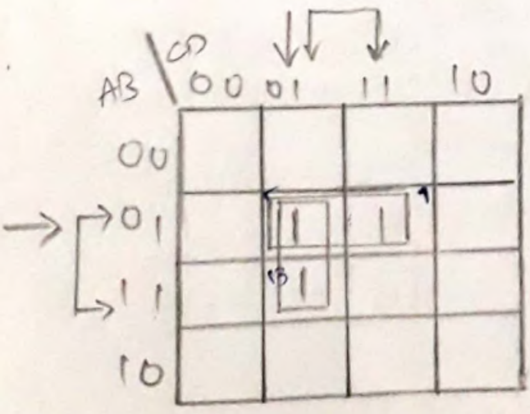
$f(A,B,C,D) = \bar{A}\bar{C}\bar{D} + A\bar{C}D$



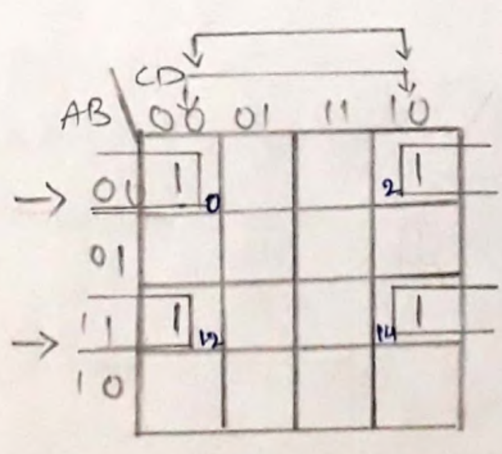
$f(A,B,C,D) = B\bar{C}D + AC\bar{D}$



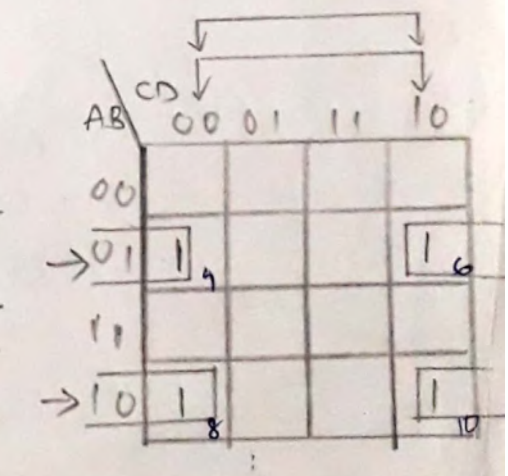
$f(A,B,C,D) = A\bar{C}\bar{D} + BCD$



$f(A,D,C,D) = BC\bar{D} + \bar{A}BD$



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



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

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

$f(A,B,C,D) = \bar{C}\bar{D}$

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

$f(A,B,C,D) = \bar{C}D$

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

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

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

$f(A,B,C,D) = C\bar{D}$

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

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

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

$f(A,B,C,D) = \bar{B}\bar{D}$

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

$f(A,B,C,D) = \bar{B}\bar{C}$

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

$f(A,B,C,D) = \bar{B}D$

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

$f(A,B,C,D) = \bar{B}C$

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

$f(A,B,C,D) = \bar{A}\bar{D}$

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

$f(A,B,C,D) = B\bar{D}$

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

$f(A,B,C,D) = A\bar{D}$

K-MAP GROUPING POSSIBILITIES

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

$f(A,B,C,D) = \overline{A}$

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

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

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

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

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

$f(A,B,C,D) = \overline{C}$

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

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

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

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

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

$f(A,B,C,D) = \overline{B}$

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

$f(A,B,C,D) = \overline{D}$

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

$f(A,B,C,D) = \overline{A}\,\overline{B}$

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

$f(A,B,C,D) = \overline{A}B$

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

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

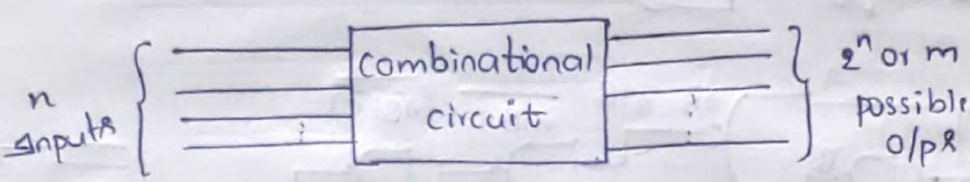
AB \ CD	00	01	11	10
00				
01				
11				
10	1	1	1	1

$f(A,B,C,D) = A\overline{B}$

COMBINATIONAL CIRCUITS

Combinational circuit (or) combinational logics:-

combinational circuit are the circuits having 'n' inputs and 'm' possible outputs with zero memory and which works only with the present state inputs.



Design procedure for combinational circuits:

(1) Statement of the problem.

(2) Identify the inputs & outputs.

(3) Assign letters (or) symbols to i/p's and outputs.

(4) construct the truth table and mention the relation between inputs & outputs.

(5) Write the boolean expression for the o/p variables with the terms of i/p variables.

(6) Simplify using k-map & Tabular method.

(7) Implement the simplified boolean expression using logic gates.

(8) Draw the combinational circuits.

Eg①:- Design a 2bit add,(or) Binary adder (or) Half adder?

(A) step ① :- Design a half-adder

Step 2,3,4:-

	i/p		o/p	
	A	B	S	C
m_0	0	0	0	0
m_1	0	1	1	0
m_2	1	0	1	0
m_3	1	1	0	1

Step ⑤:- Boolean expression for o/p

$S = \Sigma m(1,2)$, $C = \Sigma m(3)$

step ⑥

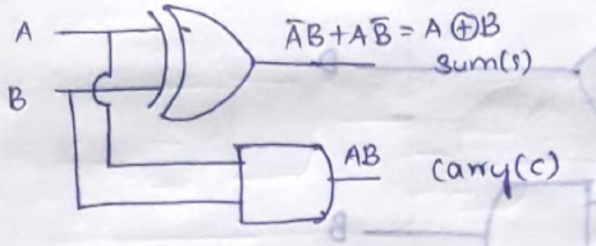
A \ B	0	1
0		1
1	1	

A \ B	0	1
0		
1		1

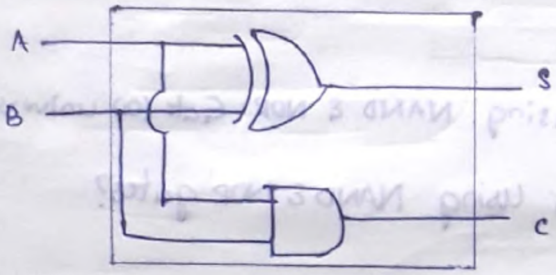
$S = \Sigma m(1,2) = \bar{A}B + A\bar{B}$ $\quad C = \Sigma m(3) = AB$

$= A \oplus B$

step ⑦:



step ⑧:-



② Design a half subtractor (or) 2bit subtractor (or) Binary subtractor.

(A) step①:- Design a half subtractor.

step 2,3,4:-

	I/P		O/P	
	A	B	D	B
m_0	0	0	0	0
m_1	0	1	1	1
m_2	1	0	1	0
m_3	1	1	0	0

step ⑤ Boolean expression for o/p

$D = \Sigma m(1,2)$, $B = \Sigma m(1)$

step⑥

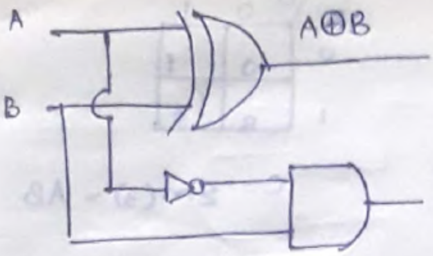
A \ B	0	1
0		1
1	1	

A \ B	0	1
0		1
1		

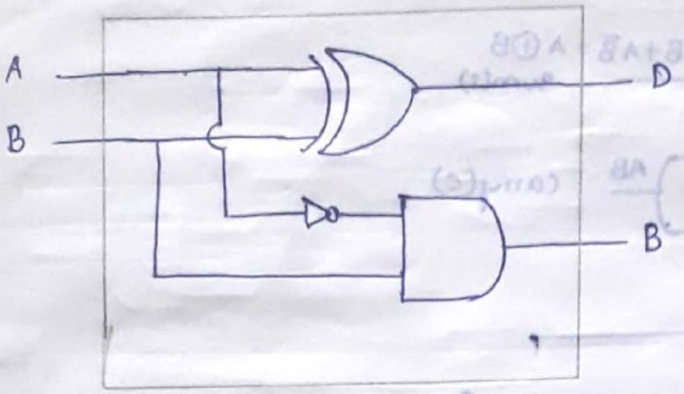
$D = \Sigma m(1,2) = \bar{A}B + A\bar{B}$ $\quad B = \Sigma m(1) = \bar{A}B$

$= A \oplus B$

Step ⑦:-



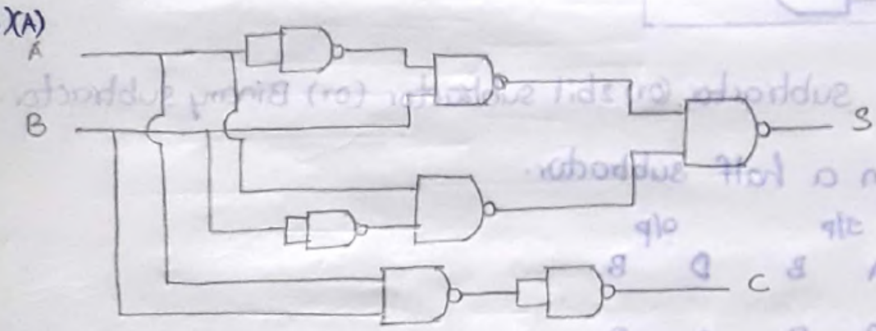
Step ⑧

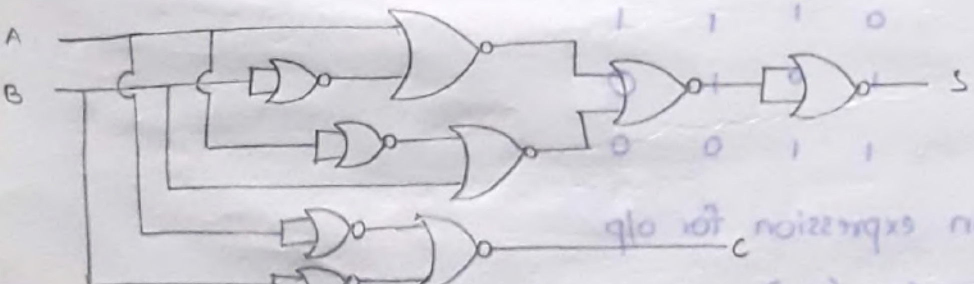


(3) Implement Half adder using NAND & NOR Gate (or) universal gates?

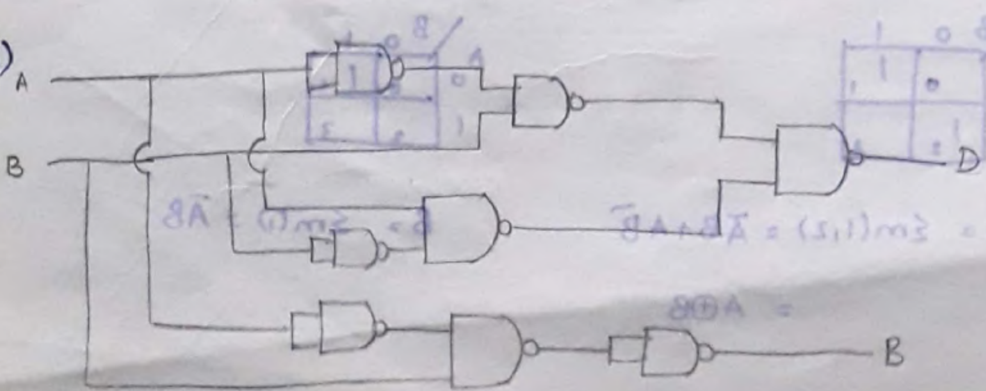
(4) Implement Half subtractor using NAND & NOR gates?

(3)(A)

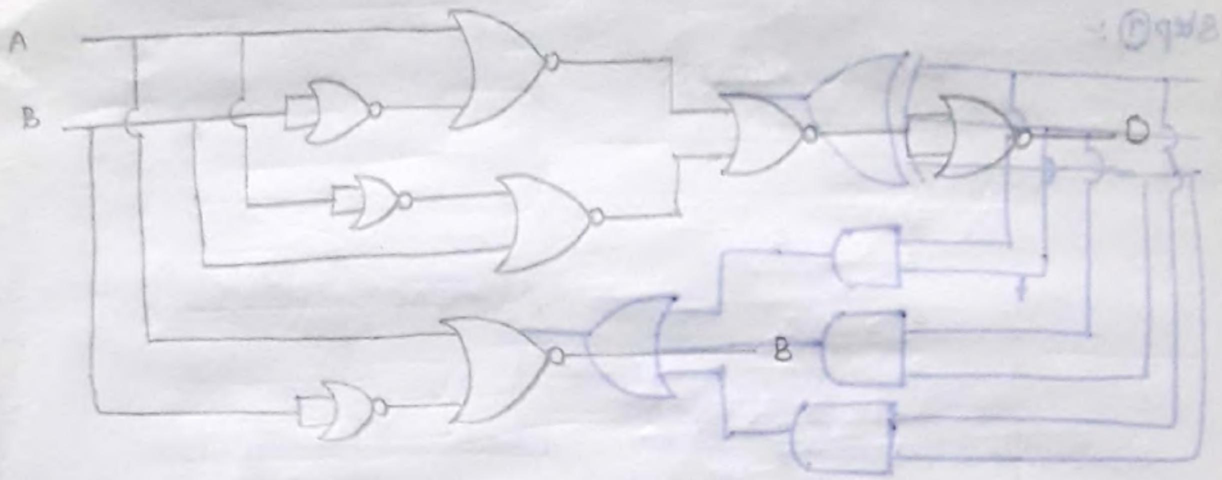




4(A)



Half Subtractor using NOR gate.



(5) Implement full adder?

A) step ① design full adder

Step 2,3,4 :-

	A	B	C_{in}	S	C_{out}
	i/p			o/p	
m_0	0	0	0	0	0
m_1	0	0	1	1	0
m_2	0	1	0	1	0
m_3	0	1	1	0	1
m_4	1	0	0	1	0
m_5	1	0	1	0	1
m_6	1	1	0	0	1
m_7	1	1	1	1	1

step ⑤ :- Boolean expression of o/p

$S = \Sigma m(1,2,4,7)$, $C_{out} = \Sigma m(3,5,6,7)$

step ⑥ :-

A \ BC_{in}	00	01	11	10
0	0	$\boxed{1}$ 1	3	$\boxed{1}$ 2
1	$\boxed{1}$ 4	5	$\boxed{1}$ 7	6

A \ BC	00	01	11	10
0	0	1	1 3	2
1	4	1 5	1 7	1 6

$S = A\bar{B}\bar{C}_{in} + \bar{A}\bar{B}C_{in} + ABC_{in} + \bar{A}B\bar{C}_{in}$

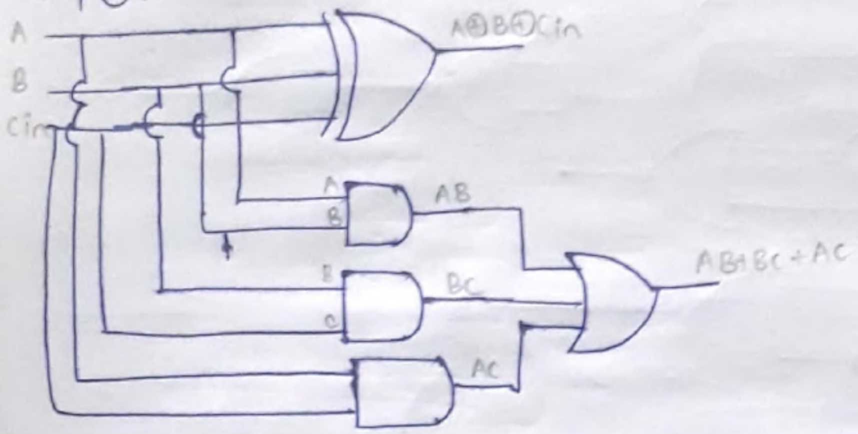
$S = C_{in}(AB + \bar{A}\bar{B}) + \bar{C}_{in}(\bar{A}B + A\bar{B})$

$C_{out} = AC_{in} + BC_{in} + AB$

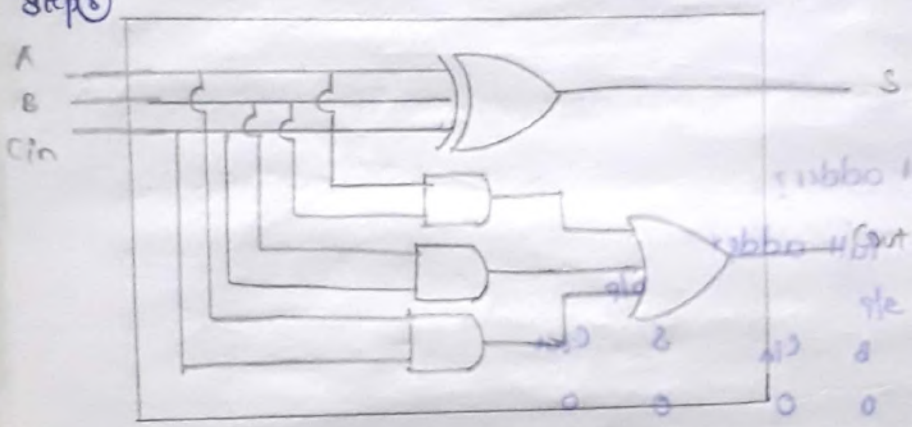
$= C_{in}(\overline{A \oplus B}) + \bar{C}_{in}(A \oplus B)$

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

Step ⑦ :-



Step ⑧



(7) Implementation of full adder using two half adders.

(A) To design full adder ~~using~~ we require two half adder and an OR gate

$S = A \oplus B \oplus C_{in}$

$C_{out} = AC_{in} + BC_{in} + AB$

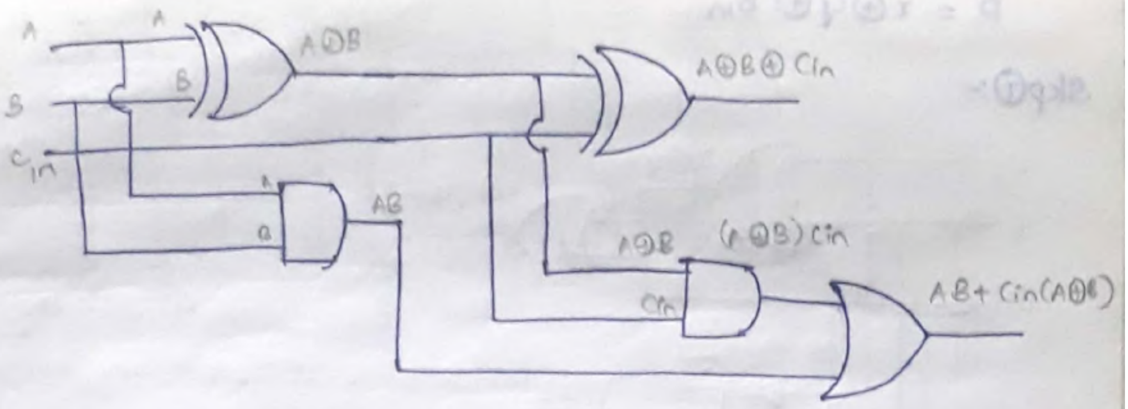
$= AC_{in}(B+\bar{B}) + BC_{in}(A+\bar{A}) + AB$

$= ABC_{in} + A\bar{B}C_{in} + ABC_{in} + \bar{A}BC_{in} + AB$

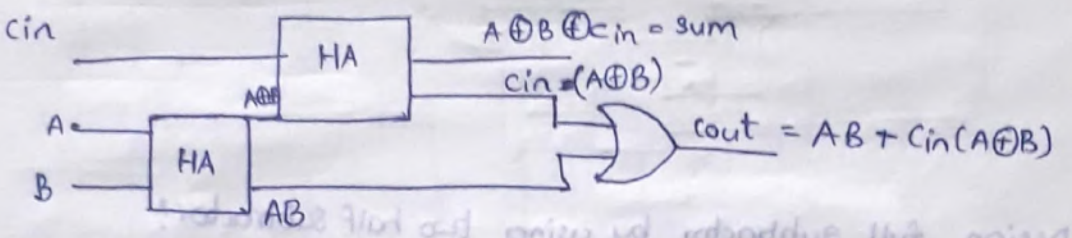
$= ABC_{in} + C_{in}(\bar{A}B + A\bar{B}) + AB$

$= AB(1 + C_{in}) + C_{in}(\bar{A}B + A\bar{B})$

$= AB + C_{in}(A \oplus B)$



combinational circuit



(8) Design full subtractor ?

(A) step① Design full subtractor

Step 2,3,4 :-

	I/P			O/P	
	x	y	Bin	D	Bout
m_0	0	0	0	0	0
m_1	0	0	1	1	1
m_2	0	1	0	1	1
m_3	0	1	1	0	1
m_4	1	0	0	1	0
m_5	1	0	1	0	0
m_6	1	1	0	0	0
m_7	1	1	1	1	1

step ⑤: Boolean expression of o/p

$D = \Sigma m(1,2,4,7)$ $\quad$ $B_{out} = \Sigma m(1,2,3,7)$

Step⑥ :- k-map

x \ yBin	00	01	11	10
0	$_0$	$\boxed{1}_1$	$_3$	$\boxed{1}_2$
1	$\boxed{1}_4$	$_5$	$\boxed{1}_7$	$_6$

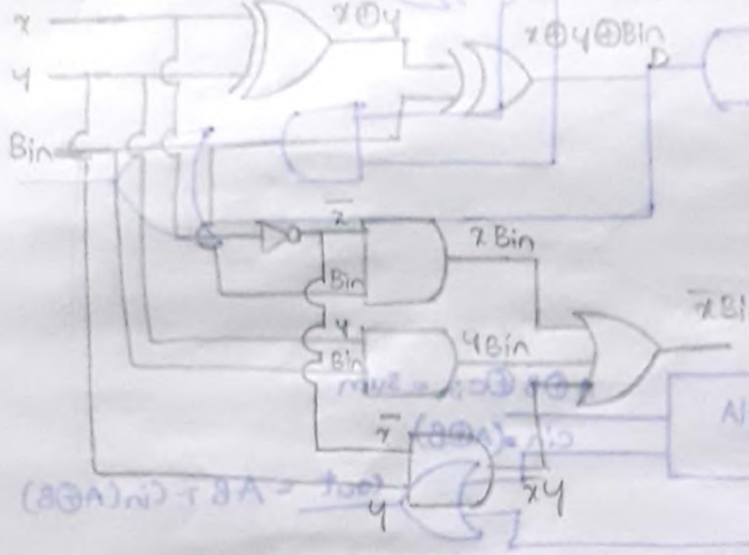
x \ yBin	00	01	11	10
0	$_0$	1_1	1_3	1_2
1	$_4$	$_5$	1_7	$_6$

$D = x\bar{y}\bar{B}_{in} + \bar{x}\bar{y}B_{in} + xyB_{in} + \bar{x}y\bar{B}_{in}$ $\quad$ $B_{out} = \bar{x}B_{in} + yB_{in} + \bar{x}y$

$D = B_{in}(\overline{x \oplus y}) + \bar{B}_{in}(x \oplus y)$

$D = x \oplus y \oplus Bin$

Step ⑦:-



(9) Design full subtractor by using two half subtractor?

(A) This implementation requires two half subtractors and OR gate

$D = x \oplus y \oplus Bin$

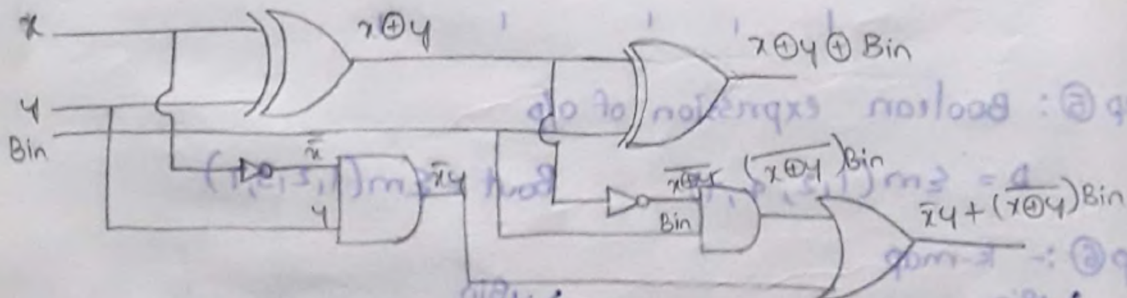
$Bout = \bar{x}y + \bar{x}Bin + yBin$

$= \bar{x}y + \bar{x}Bin(y+\bar{y}) + yBin(x+\bar{x})$

$= \bar{x}y + \bar{x}yBin + \bar{x}\bar{y}Bin + xyBin + \bar{x}yBin$

$= Bin(\bar{x}\bar{y}+xy) + \bar{x}y(1+Bin)$

$= \bar{x}y + Bin(\overline{x \oplus y})$



(10) Desingn 4 bit adder (or) parallel adder by using half adders & full adders.

(A)