Decimal to base :

(a) Decimal to binary :

eg: convert $(12.125)_{10}$ into binary. ($\to 2$)

soln ⇒ for integer part:

$$\begin{array}{r|ll} 2 & 12 & \\ \hline 2 & 6 & 0 \\ \hline 2 & 3 & 0 \\ \hline & 1 & 1 \end{array} \quad \uparrow \text{ bottom to top}$$

$(12)_{10} = (1100)_2$

for fractional part:

0.125×2	$= 0.25$	0 ← (take integer part)
0.25×2	$= 0.5$	0
0.5×2	$= 1.0$	1

↓ top to bottom

$(0.125)_{10} = (0.001)_2$

Hence, $(12.125)_{10} = (1100.001)_2$ Ans

(b) Decimal to octal :

eg: convert $(658.825)_{10}$ into octal. ($\to 8$) ~~For integer part~~.

soln ⇒ for integer part:

$$\begin{array}{r|ll} 8 & 658 & \\ \hline 8 & 82 & 2 \\ \hline 8 & 10 & 2 \\ \hline & 1 & 2 \end{array} \quad \uparrow$$

$(658)_{10} = (1222)_8$

for fractional part: (take only integer)

0.825×8	$= 6.6$	6
0.6×8	$= 4.8$	4
0.8×8	$= 6.4$	6
0.4×8	$= 3.2$	3

↓ (top to bottom)

$(0.825)_{10} = (0.6463)_8$

Hence $(658.825)_{10} = (1222.6463)_8$

(c) decimal to hexadecimal :

eg: Convert $(5386.345)_{10}$ into hexadecimal.

soln ⇒ for integer part

$$\begin{array}{r|ll} 16 & 5386 & \\ \hline 16 & 336 & 10 \to A \\ \hline 16 & 21 & 0 \\ \hline & 1 & 5 \end{array} \quad \uparrow$$

$(5386)_{10} = (150A)_{16}$

for fractional part:

0.345×16	$= 5.52$	5
0.52×16	$= 8.32$	8
0.32×16	$= 5.12$	5
0.12×16	$= 1.92$	1

↓

$(0.345)_{10} = (0.5851)_{16}$

Hence, $(5386.345)_{10} = \cancel{(150A)_{16}}\ (150A.5851)_{16}$.

Base to decimal:

(a) binary to decimal:

~~g. $(110001.1011)_2 = (\text{------})_{10}$~~

~~soln $\Rightarrow (1\times8^0)+(1\times8^1)+(0\times2^2)+(0\times2^3)+(0\times2^4)+(1\times2^5)+(1\times2^{-1})+(0\times2^{-2})+(1\times2^{-3})+(1\times2^{-4})$~~

~~$= 1+$~~

$\xleftarrow{+ve}$ $\xrightarrow{(-ve)}$

eg: $(110001.1011)_2 = (\text{------})_{10}$

soln $\Rightarrow$ $(1\times2^0)+(0\times2^1)+(0\times2^2)+(0\times2^3)+(1\times2^4)+(1\times2^5)+(1\times2^{-1})+(0\times2^{-2})+(1\times2^{-3})+(1\times2^{-4})$

$= 1+0+0+0+16+32+\frac{1}{2}+0+\frac{1}{8}+\frac{1}{16}$

$= 49 + 0.5 + 0.125 + 0.0625$

$= (49.6875)_{10}$

(b) octal to decimal:

eg: $(62.5463)_8 = (\text{------})_{10}$

soln $\Rightarrow$ $(6\times8^1)+(2\times8^0)+(5\times8^{-1})+(4\times8^{-2})+(6\times8^{-3})+(3\times8^{-4})$

$= 48+2+0.625+0.0625+0.01171+0.0007$

$= (50.7)_{10}$ Ans

(c) Hexadecimal to decimal:

eg: $(150\overset{\to 10}{A}.5851)_{16} = (\text{----})_{10}$

soln $\Rightarrow$ $(10\times16^0)+(0\times16^1)+(5\times16^2)+(1\times16^3)+(5\times16^{-1})+(8\times16^{-2})+(5\times16^{-3})+(1\times16^{-4})$

$= 10+0+1280+0.3125+0.03125+0.00122+1.52587$

$= (5386.345)_{10}$

eg: Determination of base number.

eg: Determine the value of base x if $(193)_x = (623)_8$

soln → convert both the numbers into decimal system i.e.

$6\times 8^2 + 2\times 8^1 + 3\times 8^0 = (403)_{10}$

also $(1\times x^2) + (9\times x^1) + (3\times x^0) = 403$

$\Rightarrow x^2 + 9x + 3 - 403 = 0 \Rightarrow x^2 + 9x - 400 = 0$

$\Rightarrow x = 16, -25$

negative base is not applicable, Hence $x = 16$ Ans

Octal to Hexadecimal:

Q. convert $(572)_8 = (\text{------})_{16}$

soln → ~~method 1~~ 1st we find octal to decimal

$(2\times 8^0) + (7\times 8^1) + (5\times 8^2) = 2 + 56 + 320 = (378)_{10}$

~~method 2~~ and then decimal to hexadecimal

$$\begin{array}{r|ll} 16 & 378 & \\ \hline 16 & 23 & 10 \;(A) \\ \hline & 1 & 7 \end{array}$$

$(378)_{10} = (17A)_{16}$

* Binary addition:

Rules: sum carry

		sum	carry
1 + 1	=	0	1
1 + 0	=	1	0
0 + 0	=	0	0
1 + 1 + 1	=	1	1

eg: Add $(11001100)_2$ and $(11011010)_2$

soln →

```
  1      1
  11001100
  11011010
 ~~110010110~~
```

```
 1 11
  11001100
  11011010
 ---------
 110100110
```

~~Octal address~~

Octal addition :

eg: Add $(167)_8$ and $(325)_8$

Soln ⇒

	1 1 → carry
	$(1\,6\,7)_8$
	$(3\,2\,5)_8$
	$(5\,1\,4)_8$

Rough

1 1
1 6 7
3 2 5
5 (9-8)(12-8) → subtract करते हैं, तो carry 1 होगा
1 4

Rules :

if result is greater than base then divide the result by base.

quotient = carry

Remainder = sum

Hexadecimal addition :

Rules :

result or sum is greater than 16. Then divide the result by 16.

quotient = carry

remainder = sum

eg: Add $(3F8)_{16}$ and $(5B3)_{16}$

Soln ⇒

1 → carry
3 F 8
5 B 3
$(9\,A\,B)_{16}$

Rough

0 - 9	
10 - A	
11 - B	
12 - C	
13 - D	
14 - E	
15 - F	
16 ~~16~~	

Rough

1
3 F 8
5 B 3
9 A 11 → B

F = 15
B = 11
26 - 16 = 10 → A

Subtract hoga to Carry 1 होगा

binary subtraction :

eg: perform $(11101100)_2 - (00110010)_2$

soln ⇒

$$\begin{array}{r} 1\;1\;1\;0\;1\;1\;0\;0 \\ -\;0\;0\;1\;1\;0\;0\;1\;0 \\ \hline (1\;0\;1\;1\;1\;0\;1\;0)_2 \end{array}$$

Octal subtraction :

eg: (i) compute $(516)_8 - (413)_8$

soln ⇒

$$\begin{array}{r} (5\;1\;6)_8 \\ -\;(4\;1\;3)_8 \\ \hline (1\;0\;3)_8 \end{array}$$

(ii) compute $(506)_8 - (413)_8$

soln ⇒

$$\begin{array}{r} 4\;\;8\;\; \\ \cancel{5}\;0\;6 \\ -\;4\;1\;3 \\ \hline (0\;7\;3)_8 \end{array}$$

Hexadecimal subtraction :

eg: (i) compute $(CB2)_{16} - (972)_{16}$

soln ⇒

$$\begin{array}{r} C\;B\;2 \\ -\;9\;7\;2 \\ \hline (3\;4\;0)_{16} \end{array}$$

(ii) compute $(CB2)_{16} - (9C2)_{16}$

soln ⇒

$$\begin{array}{r} B\;\;27\;\; \\ \cancel{C}\;\cancel{B}\;2 \\ -\;9\;C\;2 \\ \hline 2\;F\;0 \end{array}$$

* Method to find complement :

(i) $(245)_8$ find 7's complement and 8's complement.
(ii) $(785)_{16}$ find F's complement and 16's complement.
(iii) $(789)_{10}$ find 9's complement and 10's complement.

(i) soln ⇒

$$\begin{array}{r} 777 \\ 245 \\ \hline 532 \end{array} \rightarrow \text{7's complement}$$

$$\begin{array}{r} +1 \\ \hline 533 \end{array} \rightarrow \text{8's complement}$$

(iii)

$$\begin{array}{r} 999 \\ 789 \\ \hline 210 \end{array} \rightarrow \text{9's complement}$$

$$\begin{array}{r} +1 \\ \hline 211 \end{array} \rightarrow \text{10's complement}$$

(ii)

$$\begin{array}{r} FFF \\ 785 \\ \hline 87A \end{array} \rightarrow \text{15's complement}$$

$$\begin{array}{r} +1 \\ \hline 87B \end{array} \rightarrow \text{16's complement}$$

(iv) find 7's complement of $(2456)_8$.

soln ⇒

$$\begin{array}{r} 7777 \\ 2456 \\ \hline 5321 \end{array} \rightarrow \text{7's complement.}$$

0, 1

```
 1111
 1000
10111
```

Trick:

for 9's complement जितने digit होंगे उतने times 9 लिखना है। and उससे given number को subtract करें।

eg: find 9's complement of $(65270)_{10}$

$$\begin{array}{r} 99999 \\ 65270 \\ \hline 34729 \end{array}$$ → 9's complement

and

$$\begin{array}{r} +1 \\ \hline 3472A \end{array}$$ → 10's complement

eg: find 2's complement of $(10101)_2$

soln ⇒

$$\begin{array}{r} 11111 \\ -10101 \\ \hline 01010 \end{array}$$ → 1's complement

$$\begin{array}{r} +1 \\ \hline 01011 \end{array}$$ → 2's complement

Q.) subtract $(57232)_{10} - (03550)_{10}$ using r's complement

soln ⇒ 10's complement of $(03550)_{10}$

$$= \begin{array}{r} 99999 \\ -03550 \\ \hline +96449 \end{array}$$ → 9's complement

$$\begin{array}{r} +1 \\ \hline (96450)_{10} \end{array}$$ → 10's complement

then

$$\begin{array}{r} 57232 \\ +96450 \\ \hline 153682 \end{array}$$

1 → discard

∴ $(57232)_{10} - (03550)_{10} = (53682)_{10}$

Notes: complement +ve number ko −ve karta hai & −ve number ko +ve karta hai.

2	8	0	2	15	1
2	4	0	2	7	1
2	2	0	2	3	1
	1			1	

Q) Subtract $(3550)_{10} - (57232)_{10}$

Soln ⇒ 10's complement of $(57232)_{10}$

$$
\begin{array}{r}
99999 \\
-\ 57232 \\
\hline
42767
\end{array}
\rightarrow 9\text{'s complement}
$$

$$
\begin{array}{r}
42767 \\
+\ 1 \\
\hline
42768
\end{array}
\rightarrow 10\text{'s complement}
$$

$$
\begin{array}{r}
03550 \\
+\ 42768 \\
\hline
46318
\end{array}
$$

No end carry generates.

∴ 10's complement of $(46318)_{10}$ is with (−ve sign)

$$
\begin{array}{r}
99999 \\
46318 \\
\hline
53681
\end{array}
+1 = (53682) \rightarrow 10\text{'s complement}
$$

∴ $(3550)_{10} - (57232)_{10} = -(53682)_{10}$

Q) perform $(15)_{10} - (8)_{10}$ using 4 bit 2's complement representation.

Soln ⇒ $(15)_{10} = (1111)_2$

$(8)_{10} = (1000)_2$

2's complement of $(1000)_2$

i.e
$$
\begin{array}{r}
1111 \\
1000 \\
\hline
0111
\end{array}
\rightarrow 1\text{'s complement}
$$

$$
\begin{array}{r}
0111 \\
+1 \\
\hline
1000
\end{array}
\rightarrow 2\text{'s complement}
$$

i.e
$$
\begin{array}{r}
1111 \\
1000 \\
\hline
1\,0111
\end{array}
$$

discard

∴ $(15)_{10} - (8)_{10} = 0111$

Binary codes :

(i) weighted binary code
* (a) BCD and binary code
* (ii) Non-weighted binary code
* (a) Excess 3 code
* (b) Gray code.

* weighted code : In this, each digit position of the number represents a specific weight.

2^5	2^4	2^3	2^2	2^1	2^0
32	16	8	4	2	1

eg.

decimal to binary

decimal	binary (4 bit)
5	0101
8	1000
9	1001
13	1101

2^5	2^4	2^3	2^2	2^1	2^0	
32	16	8	4	2	1	
0	0	0	1	0	1	→ 5
		1	0	0	0	→ 8
		1	0	0	1	→ 9
		1	1	0	1	→ 13

→ binary coded decimal

BCD code :

Here, each digit of decimal number is represented by a seprate group of 4-bits with ~~wait~~ weightage (8-4-2-1).

eg : BCD code of $(58)_{10} = (\overset{5}{0101}\ \ \overset{8}{1000})_{BCD}$

(a) Excess 3 code : In a simple BCD code, add 3 (0011) to get the excess-3 code.

Note: BCD code only we write for decimal number since we have given any no. in octal or hexadecimal form then 1st we change them in decimal.

Q. Find the excess-3 code for following decimal number.
(a) 592 (b) 403

Soln ⇒ (a) 592

→ add each pair of 4 digit by 3 i.e (0011)

$$
\begin{array}{rccc}
\Rightarrow & 0101 & 1001 & 0010 \\
+ & 0011 & 0011 & 0011 \\
\hline
 & 1000 & 1100 & 0101
\end{array}
$$

(b) Gray code: It is also called reflected code.
In this, the number is changes by only one bit as it is proceeds from one number to the next.

Two bit binary code → change in gray code

2 - 1	→	binary code	gray code
0 0	→ 0		0 0
0 1	→ 1		0 1
1 0	→ 2		1 1
1 1	→ 3		1 0

binary code — mirror — gray code

3 bit gray code

000
001
- - - mirror
011
010
- - - mirror
110
111
- - -
101
100

MSB → most significant bit

* Binary to gray code conversion :

eg: (i) convert binary number $(10110)_2$ to gray code.

$\oplus$ - module 2 addition means (addition without carry)

MSB

1 0 1 1 0

1 1 1 0 1

(ii) convert (11101)Gray to binary equivalent.

MSB

1 1 1 0 1

1 0 1 1 0

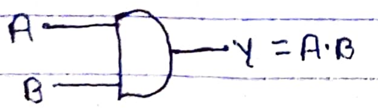
* LOGIC GATES :

(i) Basic Gates (AND, OR, NOT)

(ii) Universal Gates (NAND, NOR)

(iii) Exclusive gates (EX-OR, EX-NOR)

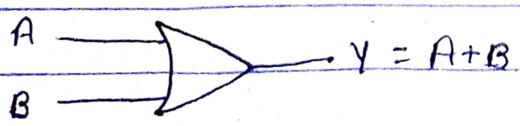
AND Gate :



Truth table

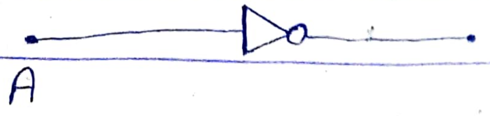
input		output
A	B	Y = A·B
0	0	0
0	1	0
1	0	0
1	1	1

OR Gate :



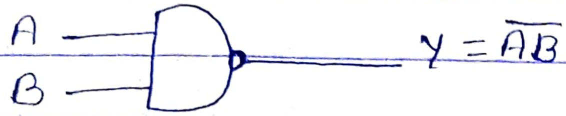
input		output
A	B	Y = A+B
0	0	0
0	1	1
1	0	1

NOT Gate (Inventor):



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

NAND Gate (universal gate):

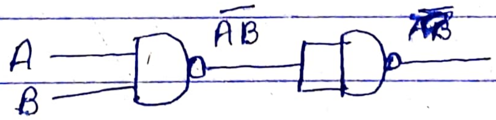


Input		Output
A	B	$Y = \overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

Note: by using NAND gate we can find rest gate like NOT, AND, OR Gate for ~~eg~~ example.

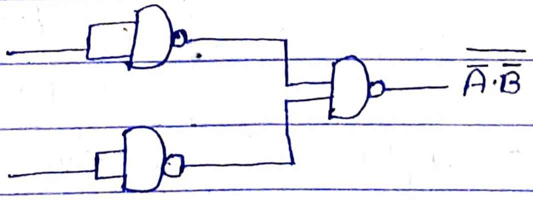
(a) NOT:

(b) AND:



$\overline{\overline{AB}} = AB$

(c) OR:



$\overline{\bar{A}\cdot\bar{B}} = \bar{\bar{A}} + \bar{\bar{B}} = (A+B)$ demorgan's theorem

Demorgan's theorem:

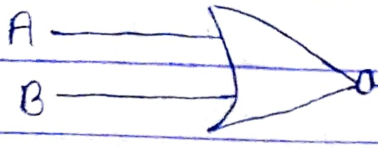
→ change the sign break the line.

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

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

Note: we can say NAND gate as a universal gate because by using this gate we can ~~define~~ design other gate.

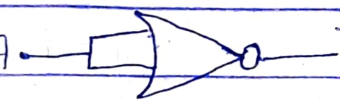
NOR Gate (universal gate) :



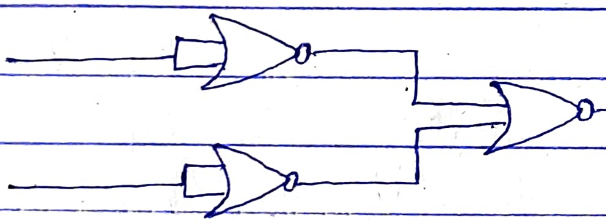
$Y = \overline{A+B}$

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

→ by using NOR Gate we can design NOT, AND, OR Gate.

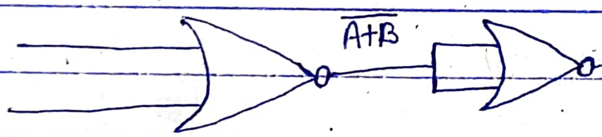
(a) NOT : A  $\bar{A}$

(b) AND :



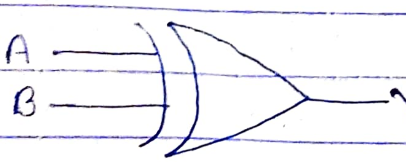
$\overline{\bar{A}+\bar{B}} = \bar{\bar{A}} \cdot \bar{\bar{B}} = AB$

(c) OR :



$\overline{\overline{A+B}}$
↓
$A+B$

- EX-OR Gate (exclusive OR)



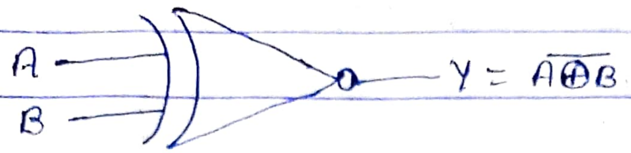
$Y = A \oplus B$

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

Input		Output
A	B	$Y = A\bar{B} + \bar{A}B$
0	0	0
0	1	1
1	0	1
1	1	0

Note: EX-OR Gate is a comparator means this is use for compare input. If both input are same then output is zero and if ~~both~~ input are different then output is one.

Ex - NOR Gate (exclusive NOR) :



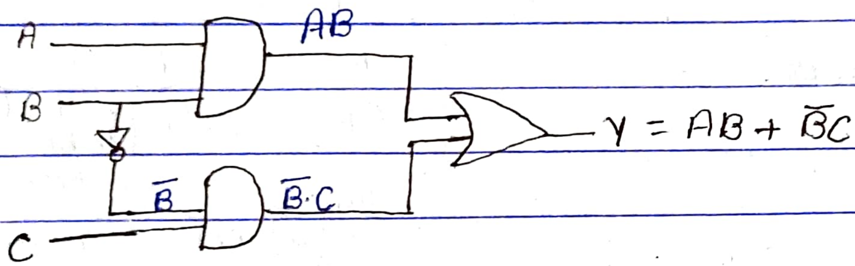
Input		Output
A	B	$Y = \overline{A \oplus B}$
0	0	1
0	1	0
1	0	0
1	1	1

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

Q. Function Realization using basic Gates :

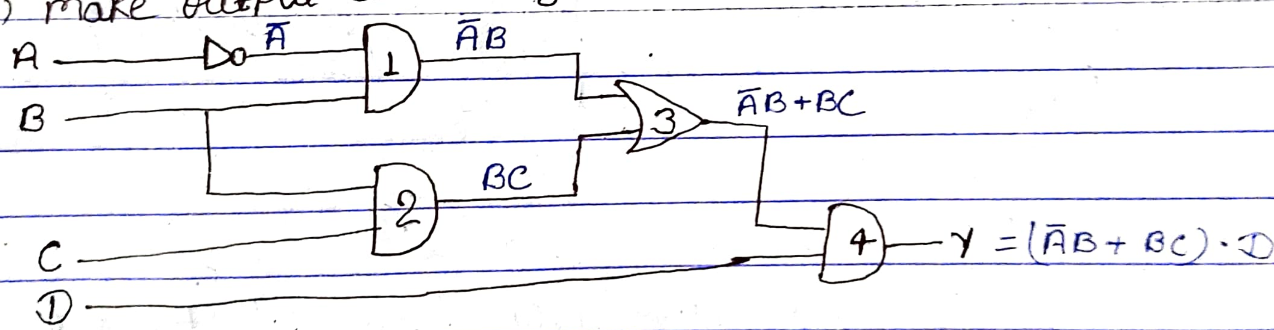
(i) $Y = AB + \bar{B}C$

Soln ⇒



(ii) make output at each gate.

(1)



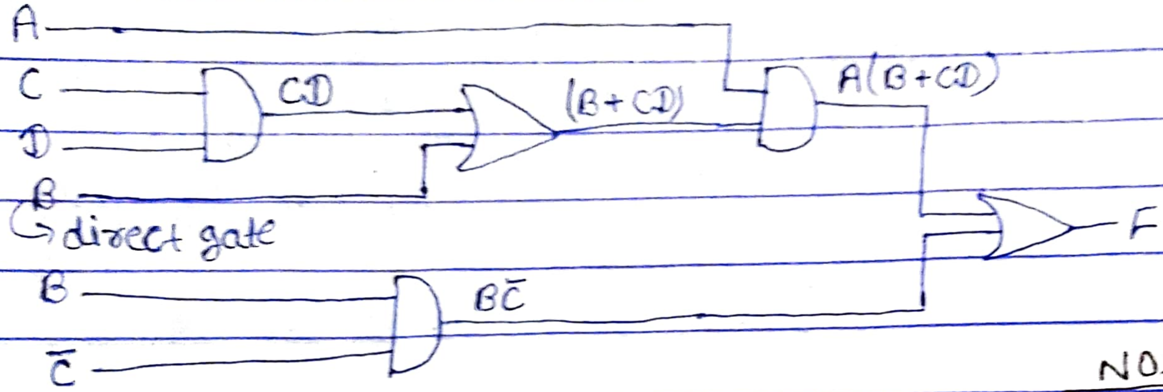
Soln ⇒ 1 Gate output $= \bar{A}B$, 2 gate output $= BC$

3 gate output $= 1 + 2 = \bar{A}B + BC$

4 gate output i.e $Y = 3 \cdot D = (\bar{A}B + BC) \cdot D$

Q. Implement the following circuit using NAND and NOR gates only. • $F = A(B + CD) + B\bar{C}$

Soln ⇒ Step 1 : AND/OR Implementation



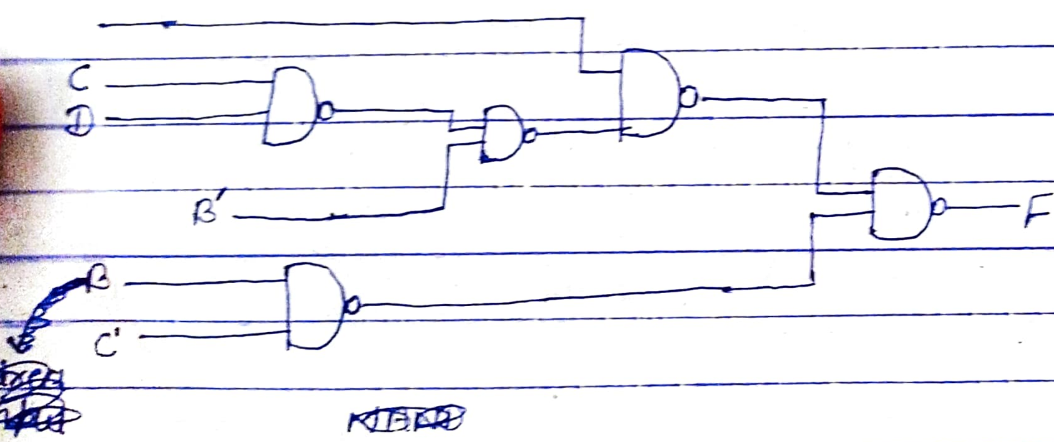
Note :

AND → NAND

OR → NAND (With direct input complemented)

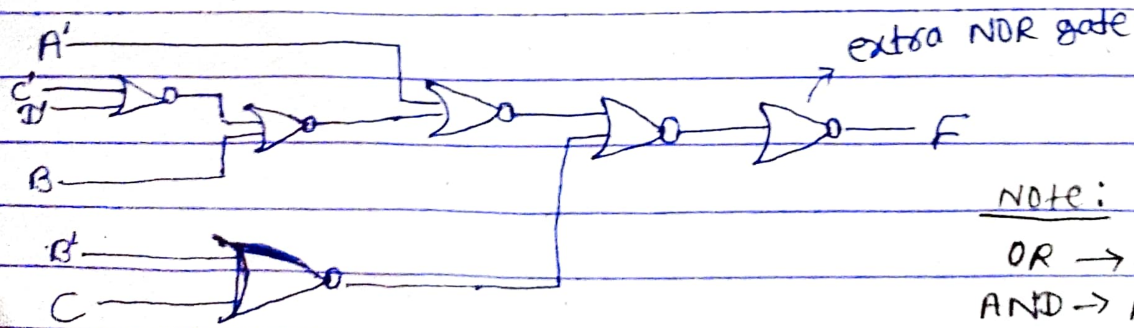
Here complemented means if output is A then take its complement which is A' or $\bar{A}$

Step 2 : NAND Implementation



also we solve by using NOR gate 1st step is as it is

Step 2 :



Note :

OR → NOR

AND → NOR (With direct input complemented)

Note : Whenever we change any gate into NOR gate then we use one extra NOR gate at end to getting final answer.

* Simplification of boolean function using boolean algebra.

* boolean laws:

$A+0=A$	$A\cdot 0=0$
$A+B=B+A$ (commutative)	$A\cdot B=B\cdot A$
$A(B+C)=AB+AC$ (distributive)	$A+BC=(A+B)(A+C)$
$A+\bar{A}=1$ (complement	$A\cdot\bar{A}=0$
$A+A=A$	$A\cdot A=A$
$A+1=1$	$A\cdot 0=0$
$A+AB=A$	$A(A+B)=A$
$A+\bar{A}B=A+B$	$A\cdot(\bar{A}+B)=AB$

* DE-MORGAN'S THEOREM:

→ change the sign, break the line.

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

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

Q. (i) simplify $\bar{A}BC\bar{D}+BC\bar{D}+B\bar{C}\bar{D}+B\bar{C}D$

Soln ⇒ $\bar{A}BC\bar{D}+BC\bar{D}+B\bar{C}\bar{D}+B\bar{C}D$

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

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

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

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

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

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

$(1+yz)$

SCP → standard canonical product

(ii) Simplify $\overline{\bar{A}B} + \bar{\bar{A}} + \overline{AB}$

Soln ⇒ Let $F = \overline{\overline{\bar{A}B} + \bar{\bar{A}} + \overline{AB}}$

$= \overline{\overline{\bar{A}B}} \cdot \overline{\bar{\bar{A}}} \cdot \overline{\overline{AB}}$ $[\because A\cdot\bar{A} = 0]$

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

$= 0$

(iii) Simplify $F(w,x,y,z) = x + xyz + \bar{x}yz + wx + \bar{w}x + \bar{x}y$

Soln ⇒ $F(w,x,y,z) = x(1 + yz + w + \bar{w}) + \bar{x}(yz + y)$

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

$= x + \bar{x}y$

$= x + y$

* SOP (sum of product) and POS (product of sum) :

(i) $F(A,B,C) = AC + AB + BC \xrightarrow{\text{form}} (SOP)$

(ii) $F(A,B,C) = (A+B)(B+C)(A+C) \xrightarrow{\text{form}} (POS)$

* Canonical SOP (sum of product) :

→ In the SOP (sum of product) form, many times all the individual terms do not involve all literals.
For example, in expression $AB + ABC̄$ the first product term do not contain literal C.

→ If each term in SOP form contain all the literals then the SOP form is known as standard or canonical SOP form.

eg: Convert the given expression in standard SOP form.

~~Soln ⇒ F(A) = AB~~ $F(A,B,C) = AC + AB + BC$

Soln ⇒ $F = AC(B+\bar{B}) + AB(C+\bar{C}) + BC(A+\bar{A})$

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

or $F(A,B,C) = ABC + A\bar{B}C + AB\bar{C} + \bar{A}BC$ $[\because A + A = A]$

~~sop~~ In every term three literals are present. Since this is in SCP form replacing $A/B/C$ by 1 or $\bar{A}/\bar{B}/\bar{C}$ by 0:

for minterm: $F(A,B,C) = \overset{4\,2\,1}{111},\ \overset{4\,2\,1}{101},\ \overset{4\,2\,1}{110},\ \overset{4\,2\,1}{011}$

$= \Sigma(7, 5, 6, 3)$

Q. convert the given expression in standard SOP for

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

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

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

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

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

for minterm ⇒ $F(A,B,C) = 111, 110, 101, 100$

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

* canonical POS (product of sum):

In the POS (product of sum) form, many times all the individual terms do not involve all literals for example, in expression $(A+B)\cdot(B+C)$ the literal is missing in both the terms or sum.

→ If each term in POS form contain all the literals then the POS term is known as standard POS or canonical POS.

eg: convert the given expression in standard POS form.

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

soln ⇒ $Y = A(A+B+C)$

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

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

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

for max term: replace A/B/C by 0 and $\bar{A}/\bar{B}/\bar{C}$ by 1.

then $Y = \pi(000, 001, 010, 011)$

$= \pi(0, 1, 2, 3)$ Ans

(ii) $F(A,B,C) = (A+B)(B+C)(A+C)$

soln ⇒ $F(A,B,C) = (A+B+C\cdot\bar{C})(B+C+A\cdot\bar{A})(A+C+B\cdot\bar{B})$

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

~~for max term ⇒ Replace A/B/C by 0 and $\bar{A}/\bar{B}/\bar{C}$ by 1.~~

~~$F(A,B,C) = \pi(000, 001, 100, 010)$~~

~~$= \pi(0, 1, 4, 2)$~~

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

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

for max term ⇒ Replace A/B/C by 0 and $\bar{A}/\bar{B}/\bar{C}$ by 1

then $Y = \pi(000, 001, 100, 010)$

$= \pi(0, 1, 4, 2)$

$= \pi(0, 1, 2, 4)$

$0 + 2^0$

$2^1 \; 2^0$ $\quad$ 1 0

2 1

Simplification of boolean function by K-map.

(i) 2-variable K-map

(ii) 3-variable K-map

(iii) 4-variable K-map.

K-map (Karnaugh map): The method first proposed by Vitch and modified by Karnaugh, hence it is known as the Veitch diagram, or the Karnaugh map.

Steps in K-map:

(i) Representation

(ii) placing

(iii) Grouping

(iv) simplified expression.

(i) Representation:

(a) 2 variable K-map:

→ In 2 variables we'll have $2^n = 2^2 = 4$ combinations are available.

A \ B	0	1
0	0	1
1	2	3

2 variable K-map

AB →
0 0 = 0
1 0 = 2

Decimal	A (2)	B (1)
0	0	0
1	0	1
2	1	0
3	1	1

b) 3 variable k-map:

there are 8 combination (2^3)

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

(cells or block)

Decimal	A (4)	B (2)	C (1)
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

(c) 4 variable k-map: there are 2^4 (16) combination.

Gray code

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

decimal	A (8)	B (4)	C (2)	D (1)
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

minterm → Σ

maxterm → π

(ii) placing (filling) :

→ for minterms we place '1' on the respective block. ~~representation~~

→ for maxterms we place '0' on the respective block.

(iii) Grouping :

It is done in adjacent cell and in power of 2.

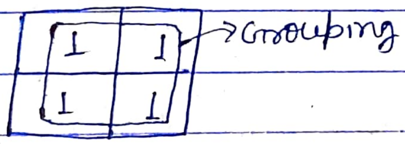
for eg : $2^0 = 1$ (single not paired)

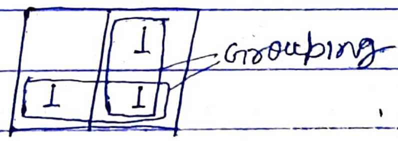
$2^1 = 2$ (double

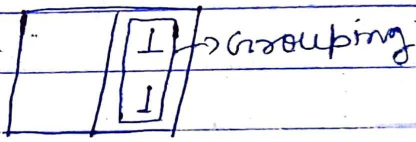
$2^2 = 4$ (quad) , $2^3 = 8$ (oct)

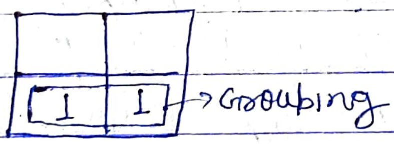
$2^4 = 16$ (Hex)

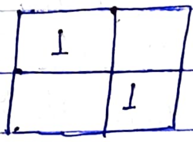
(a) Grouping (2-variable k-map) :





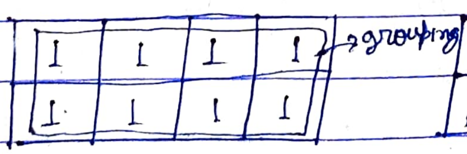


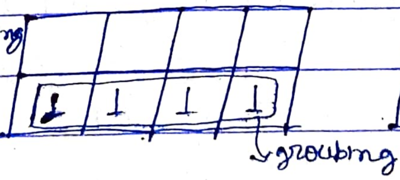


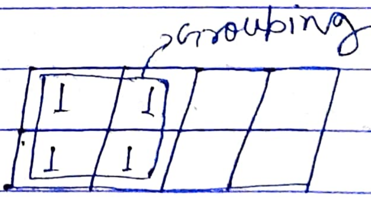


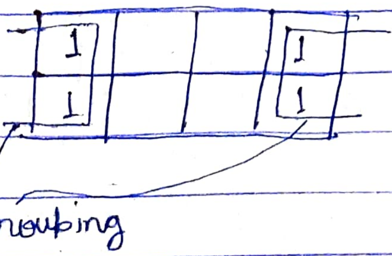
→ here pairing is not possible not form Grouping

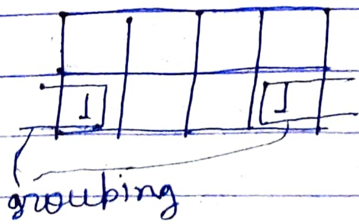
(b) Grouping (3-variable k-map) :

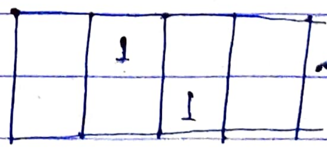






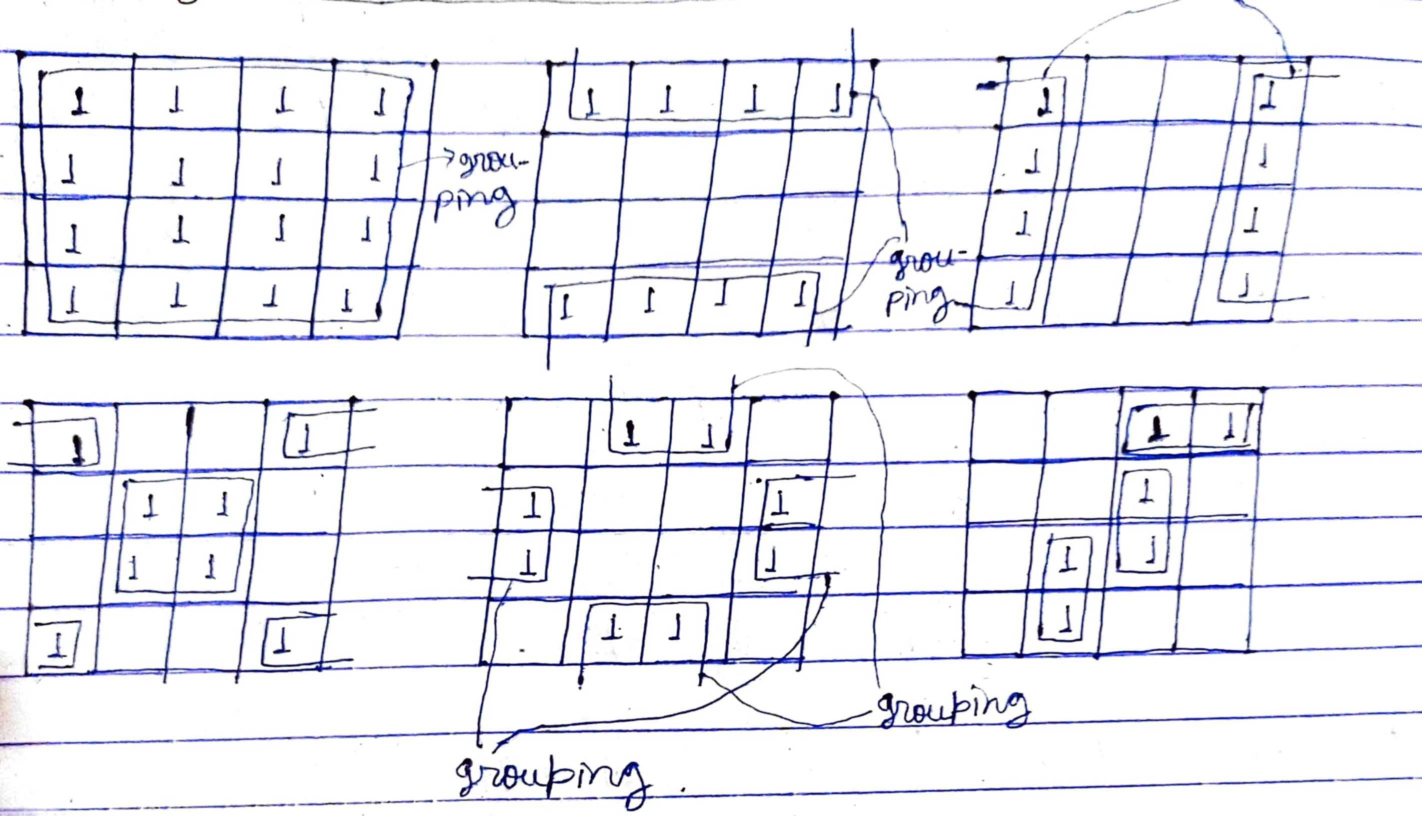






→ here grouping is not possible.

Grouping (4-variable k-map):



$$110 \quad 111 \quad 001$$

Eg:(i) Reduce $Y = AB\bar{C} + ABC + \bar{A}\bar{B}C$ using k-map.

soln ⇒ It is in SOP form. since replacing A/B/C by 1 or $\bar{A}/\bar{B}/\bar{C}$ by 0.

1st find minterm by converting the expression into its standard form if required.

$\therefore Y = \Sigma(110, 111, 001) = \Sigma(6, 7, 1)$

4	2	1	
1	1	0	= 6
1	1	1	= 7
0	0	1	= 1

since it is 3-variable function

B = 1

A \ BC	00	01	11	10
$\bar{A}$ ← 0		1		
A ← 1			1	1

→ grouping

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