Chapter = 1

Introduction:

Introduction to Signal:

A signal is a function that conveys information about a phenomenon. In electronics, it refers to any time varying voltage, current or electromagnetic wave that conveys information.

There are following two types of Signals:

(a) Analog
(b) Digital.

(a) Analog Signal

Analog Signal is a kind of continuous wave form that changes over time. An analog signal is further classified into simple and composite signals. A simple analog signal is a sine wave that cannot be decomposed further. On the other hand, a composite analog signal can be further decomposed into multiple sine waves.

An analog signal is described using amplitude, period of frequency and phase. Amplitude marks the maximum height of the signal. Frequency marks the rate at which signal is changing. Phase marks the position of the wave with respect to time zero.

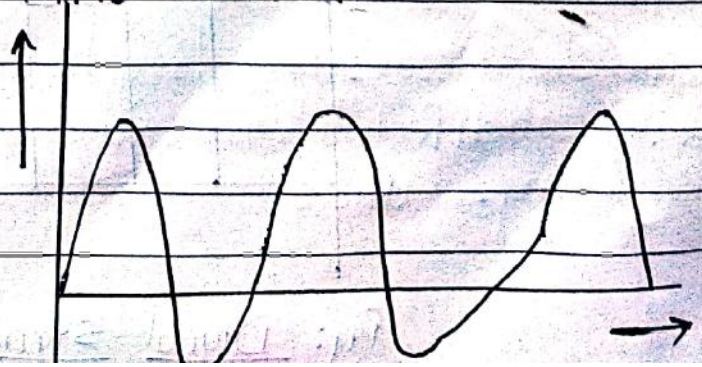


Fig: Analog Signal

An analog signal is not immune to noise hence, it faces distortion and decrease the quality of transmission. The range of value in an analog signal is not fixed.

(b) Digital Signal

Digital signals also convey information like analog signals but is somewhat is different from analog signal. Digital signal is non-continuous, discrete time signal. Digital signal carries information or data in the binary form i.e a digital signal represent information in the form of bits. Digital signal can be further decomposed into simple sine waves that are called harmonics. Each simple wave has different amplitude, frequency and phase. Digital signal is described with bit rate and bit interval. Bit interval describes the time require for sending a single bit. On the other hand, bit rate describes the frequency of bit interval.

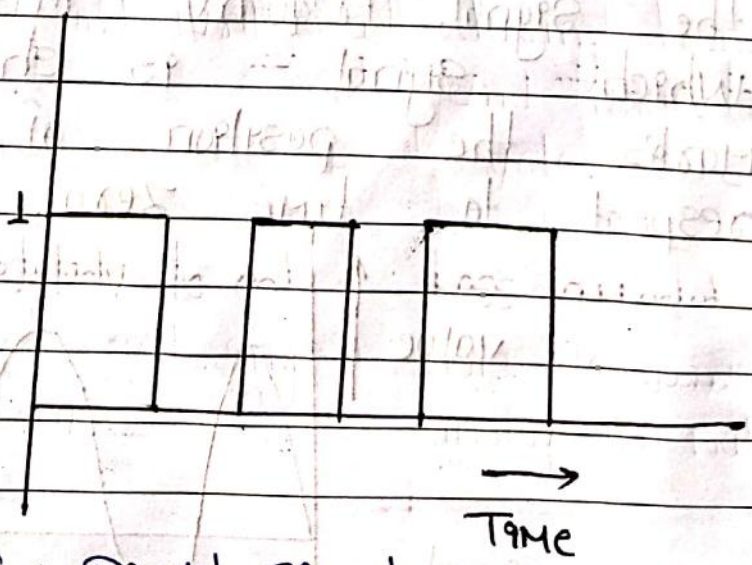


Fig: Digital signal

A digital signal is more immune to the noise; hence, it hardly faces any distortion. Digital signals are easier to transmit and are more reliable when compared to analog signals. Digital signal has a finite range of values. The digital signal consists 0s and 1s.

Comparison Chart;

Basic for comparison	Analog Signal	Digital Signal
Basic	An analog system is a continuous wave that changes over a time period.	A digital signal is discrete wave that carries information in binary form.
Representation	An analog system is represented by a sine wave.	A digital signal is represented by square waves.
Description.	An analog system is described by the amplitude, period of frequency, and phase.	A digital signal is described by bit rate and bit intervals.
Range.	It has no fixed range.	It has a finite number i.e 0 and 1.
Distortion	It is more prone to distortion.	It is less prone to distortion.

Assignment

Flexibility	Analog hardware is not flexible.	Digital hardware is flexible in implementation.
Uses	Can be used in analog devices only. Best suited for audio and video transmission.	Best suited for computing and digital electronics.
Application/ example.	Thermometer.	PCs, PDAs.
Memory	Stored in the form of wave signal.	Stored in the form of binary bit.
Power	Analog instrument draws large power.	Digital instrument draws only negligible power.
Cost	Low cost and portable	Cost is high and not easily portable.
Impedance	Low	High order of 100 Megaohm
Errors	Analog instruments usually have a scale which is cramped at lower end and give considerable observation errors.	Digital instruments are free from observational errors like parallax and approximation errors.

Due to above comparisons the digital systems are more preferred rather than anlog system.

Advantages of digital System;

→ Have made possible many scientific, industrial, and commercial advances that would have been unattainable otherwise.
→ Less expensive
→ More reliable
→ Easy to manipulate.
→ Flexiblity and compactibility.
→ Information Storage can be easier in digital computer systems than in analog ones. New features can often be added to a digital system more easily too.

Disadvantage of digital System;

→ Use more energy than analog circuits to accomplish the same task, thus producing more heat as well.
→ Digital circuits are often fragile, in that if a ~~signal~~ ~~p~~ ~~sig~~ single piece of data is lost or misinterpreted, the meaning of large blocks of related ~~dan~~ can be completely change.
→ Digital computer manipulates discrete elements of information by means of a binary code.
→ Quantization error during analog signal sampling.

Unit 2

Number System and Codes

Number System can simply be defined as a way to represent numbers.

Types of number system:

1) Decimal number System
   Radix (r) = 10
   Symbol = 0 through r-1 = {0, 1, 2 --- 9}
   It is also known as base 10 number system as it uses just '10' symbols i.e 0 to 9.

2) Binary number System.
   Radix (r) = 2
   Symbol = 0 through r-1 = {0, 1}
   Binary System refers to the number System that uses only two symbols i.e 0 and 1. That's why it is called a base '2' number System. It is also known as (BIT) binary digit. This number System is especially used in the internal processing of Computer System.

3) Octal number System
   Radix (r) = 8
   Symbol = 0 through r-1 = {0, 1, 2 --- 7}
   It is base '8' System.

4) Hexa-decimal number System:
   Radix (r) = 16
   Symbol = 0 through (r-1) = {0, 1, 2 --- 9, A, B, C, D, E, F}

Conversion of Number System:

Binary

(1) → binary to decimal (base 2 to base 10)

(a) Convert $(110011)_2$ to decimal.

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

$= 32 + 16 + 2 + 1$

$= (51)_{10}$

(b) $(1011.101)_2 \rightarrow (?)_{10}$

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

$= 8 + 2 + \frac{1}{2} + \frac{1}{2^3} + 1$

$= (11.625)_{10}$

(2) binary to octal:

(a) Convert $(10110101)_2$ into octal.

Here

$=$ 010 110 101

2 6 5

$\therefore (10110101)_2 = (265)_8$

(b) $(11100)_2 \rightarrow (?)_8$

→ 011 100

3 4

$\therefore (11100) = (34)_8$

© $(76.2)_2 \longrightarrow (\quad)_8$

→ 7 6 2

111 110 010

∴ 76.2 → (111110.010)8

(3) binary to hexa-decimal :

@ $(1010.11)_2 \rightarrow (\quad)_{16}$

$(1010.1100)_2 \longrightarrow (\quad)_{16}$

→ 1010 1100

10(A) 12 (C)

∴ (1010.1100 → (A.C)16

ⓓ $(1010101101)_2 \longrightarrow (\quad)_{16}$

Here,

0101 0101 1101

5 5 D

∴ $(1010101101)_2 \rightarrow (55D)_{16}$

Decimal

(4) Conversion of decimal to binary (base 10 to 2) :

(a) $(51)_{10} \rightarrow (\ )_2$

2	51	1
2	25	1
2	12	0
2	6	0
2	3	1
2	1	1
	0	

$\therefore (51)_{10} \rightarrow (110011)_2$

(b) $(72.55)_{10} \rightarrow (?)_2$

2	72	0
2	36	0
2	18	0
2	9	1
2	4	0
2	2	0
2	01	1
	0	

$\cdot 55 \times 2 = 1.10 = 1$
$\cdot 10 \times 2 = 0.20 = 0$
$\cdot 20 \times 2 = 0.40 = 0$
$\cdot 40 \times 2 = 0.80 = 0$
$\cdot 80 \times 2 = 1.60 = 1$

$\therefore (72.55)_{10} = (1001000.10001)_2$

(5) Decimal to octal (base 10 to base 8)

(a)

8	177	1
8	22	6
8	2	2
	0	

$\therefore (177)_{10} \rightarrow (261)_8$

(b) $(25.5)_{10} \rightarrow (\ )_8$

8	25	1
8	3	3
	0	

$\cdot 5 \times 8 = 4.0 = 4$

$\therefore (25.5)_{10} = (31.4)_8$

(6) Decimal to hexa-decimal (base 10 to base 16)

$(77)_{10} \rightarrow (\ )_{16}$

16	77	13 = D
16	4	4 = 4
	0	

$\therefore (77)_{10} = (4D)_{16}$

Conversion of octal

(1) Octal to decimal:

@ $(632)_8 \rightarrow ()_{10}$

$= 6\times8^2 + 3\times8^1 + 2\times8^0$

$= 384 + 24 + 2$

$= (410)_{10}$.

(2) Octal to binary

$(741)_8 \rightarrow ()_2$

7	4	1
111	100	001

$\therefore (741)_8 = (111100001)_2$

(3) Octal to hexa-decimal:

$(75)_8 \rightarrow ()_{16}$

→

7	5
111	101

NOW, 0011 1101

3 D

$\therefore (75)_8 = (3D)_{16}$

Conversion of hexa-decimal :

1.a hexadecimal to decimal :

$(F4C)_{16} \rightarrow ()_{10}$

$= F \times 16^2 + 4 \times 16^1 + C \times 16^0$

$= 3840 + 64 + 12$

$= (3916)_{10}$

$\therefore (F4C)_{16} \rightarrow (3916)_{10}$

(ii) hexadecimal to binary :

@ $(A1D)_{16} \rightarrow (?)_2$

A	1	D
↓		
1010	0001	1101

$\therefore (A1D)_{16} = (101000011101)_2$

Some examples

1) $(75)_8 \rightarrow ()_{16}$

7 ↓ 5 ↓

111 101

Now, for hexadecimal,

0011 1101

3 D

$\therefore (75)_8 = (3D)_{16}$

2) $(751)_8 \rightarrow ()_{10}$

$= 7 \times 8^2 + 5 \times 8^1 + 1 \times 8^0$

$= 448 + 40 + 1$

$= (489)_{10}$

3) $(BCA)_{16} \rightarrow ()_8$

B C A

1011 1100 1010

Now

Binary no.	101	111	001	010
Octal equivalent	5	7	1	2

$(880.70)_{10} \rightarrow (\ )_{16}$

16	880	0
16	55	7
16	3	3
	0	

$\therefore (880.70)_{10} \rightarrow (370.B3)_{16}$

$(0110110.1101)_2 \rightarrow (\ )_8$

Here,

$(000110110.110100)_2 \rightarrow (\ )_8$

000	110	110	110	100
↓	↓	↓	↓	↓
0	6	6	6	4

$\therefore (000110110.110100)_2 \rightarrow (066.64)_8$

Decimal to BCD conversion. (8421)
(Binary coded decimal).

Decimal	BCD 8421	Unused (don't care)
0	0000	1010
1	0001	1011
2	0010	1100
3	0011	1101
4	0100	1110
5	0101	1111
6	0110	
7	0111	
8	1000	
9	1001	

Decimal to excess-3 (XS-3) (X-3)

Decimal	8421	excess-3
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

Self complementing

Q4 $(11011)_2$ → ( ) excess-3

```
  11011
+ 0011
-------
(11110) xs-3
```

Basic term :

MSB (Most Significant Bit)

→ The left most bit of a number is called MSB.

LSB (Least Significant bit)

→ The right most bit of a number is called L.S.B.

Decimal to 2421 Code

	L.S.B ←	MSB →
Decimal	2421	2421
0	0000	0000
1	0001	0001
2	0010	1000
3	0011	1001
4	0100 (1010)	1010
5	0101	1011
6	0110	1100
7	0111	1101
8	1110	1110
9	1111	1111

Note: If the addition of the weight is equal to '9' then it become self-complementing weight.

Qu. Prove 2421 is a self-complementing.

Decimal	2	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	→ LSB
5	1	0	1	1	→ MSB
6	1	1	0	0	
7	1	1	0	1	
8	1	1	1	0	
9	1	1	1	1	

Hence 2421 is self-complementing.

84-2-1 Code

Decimal

Decimal	8	4	-2	-1
0	0	0	0	0
1	0	0	1	1
2	0	1	1	0
3	0	1	0	1
4	0	1	0	0
5	1	0	1	1
6	1	0	1	0
7	1	0	0	1
8	1	0	0	0
9	1	1	1	1

Hence, 84-2-1 is Self-complementing code.

Decimal to XS-3

(a) $(6)_{10} \longrightarrow (\ )_{XS-3}$

$$\begin{array}{r} 0110 \\ +\ \ 11 \\ \hline 1001 \end{array}$$

(b) $(68)_{10} \longrightarrow (\ )_{XS-3}$

$$\begin{array}{rr} 0110 & 1000 \\ +\ \ 11 & +\ \ 11 \\ \hline 1001 & 1011 \end{array}$$

$\therefore (68)_{10} = 10011011$

Q → $(2)_{10} + (7)_{10}$

$$\begin{array}{r} 0010 \\ +\ 11 \\ \hline 0101 \end{array} \qquad \begin{array}{r} 0111 \\ +\ 11 \\ \hline 1010 \end{array}$$

$$\begin{array}{r} 0101 \\ +\ 1010 \\ \hline 11101 \end{array}$$

Now,

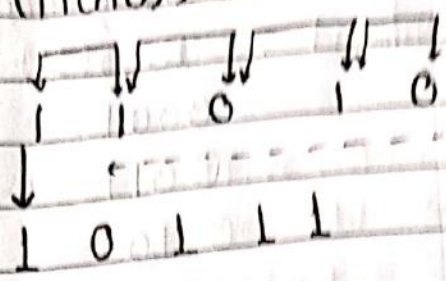
$$\begin{array}{r} 1111 \\ -\ 0011 \\ \hline (1100)_{XS-3} \end{array}$$

$\therefore (2)_{10} + (7)_{10} = (1100)_{XS-3}$

$= (12)_{10}$

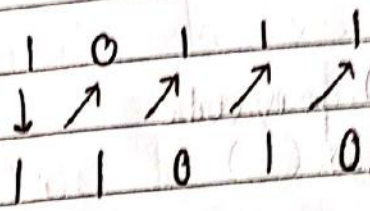
Binary to Gray code and vice versa :

(a) $(11010)_2 \rightarrow$ ( ) graycode.



$\therefore (11010)_2 \rightarrow (10111)$ graycode.

(b) (10111) graycode $\rightarrow$ ( ) binary



$\therefore$ (10111) graycode = $(11010)_2$

BCD to gray code

Decimal	BCD	Gray code
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101

Que

@ $(11001010)_2 \rightarrow (\ )$ gray code

1 1 0 0 1 0 1 0

↓

1 0 1 0 1 1 1 1

$\therefore (11001010)_2 \rightarrow (10101111)$ gray code.

ⓑ $(6)_{10} + (7)_{10}$ using X-S3

$$\begin{array}{r} 0110 \\ +\ 11 \\ \hline 1001 \end{array} \qquad \begin{array}{r} 0111 \\ +\ 11 \\ \hline 1010 \end{array}$$

$$\begin{array}{r} \text{Now } 1001 \\ +\ 1010 \\ \hline 10011 \\ -\ \ 11 \\ \hline (10000)_{XS-3} \end{array}$$

$\therefore (6)_{10} + (7)_{10} = (16)_{10}$.

Complements:

There are two types of complements:
r's complement and (r-1)'s complements.
Where 'r' is the base of a number system.

In binary number system there are two types of complements: 1's complements & 2's complements.
Similarly, decimal number system has 9's and 10's complement.

1's Complement:

We can get 1's complement by simply replacing '1' by '0' and '0' by 1.

eg: 1's complement of 1011 = 0100

Subtraction of binary number using 1's complements:

Steps:

1) Make the both numbers having same number of bits.
2) Determine the 1's complement of the number to be subtracted (subtrahend).
3) Add the 1's complement to the given number from which we subtract (minuend).
4) If carry bit occurs; remove and add it to the results.
5) If there exist no carry determine 1's complement of the result and prefix by a negative sign to get the final result.

eg: Sub 1110000 from 1100000
i.e 1100000 - 1110000

1's Complement of Subtrahend : 0001111

Add it with minuend : 0001111
+ 1100000
1101111

Since, no any additional bits occur,
1's Complement of 1101111 = 0010000
Hence difference will be -0010000.

(2) 111000 - 110000

1's Complement of Subtrahend 001111
Add it with minuend 001111
+ 111000
1000111

Since, there exist one additional bit,
difference 00111
+ 1
001000

Hence, difference will be 001000.

2's Complement

The 2's complement of binary number is obtained by adding 1 to the 1's complement.

eg: 2's Complement of 1101101 = ?

1's Complement = 0010010

2's Complement = 0010010

$$\begin{array}{r} 0010010 \\ +\quad\quad 1 \\ \hline 0010011 \end{array}$$

Subtraction using 2's Complement:

Steps:

1. Make the both numbers having the same number of bits.
2. Determine the 2's complement of the number to be subtracted. (Subtrahend).
3. If there exist no carry determine the 2's complement of the result and prefix by a negative sign to get a final result.
4. If carry bit occur remove ① and the final is ans.

eg: Sub 1110000 from 1100000

Here,

1100000 - 1110000

2's Complement of Subtrahend 0001111

$$\begin{array}{r} 0001111 \\ +\quad\quad 1 \\ \hline 0010000 \end{array}$$

Add it to minuend

$$\begin{array}{r} 0010000 \\ +\ 1100000 \\ \hline \cancel{10000000}\ 1110000 \end{array}$$

Since, there exist no additional bit,

2's Complement of 1110000 = 0001111

$$\begin{array}{r} 0001111 \\ +\quad\quad 1 \\ \hline 0010000 \end{array}$$

Hence difference will be -0010000.

(*) 111000 - 110000

2's Complement of 110000 is 001111

$$\begin{array}{r} 001111 \\ +\quad\quad 1 \\ \hline 010000 \end{array}$$

Add it to minuend

$$\begin{array}{r} 010000 \\ +\ 111000 \\ \hline 1001000 \end{array}$$

Since, there exist additional bit.

Difference will be 001000 (neglect 1).

Examples :

① $(1001)_2 - (1000)_2$
M S

→ For 1's complements,
1's complement of subtrahend $(0111)_2$
Add it to minuend 0111
$+\ 1001$
$(10000)_2$

Since, additional bit occur.
Difference 0000
$+\ 1$
$(0001)_2$ Ans

→ For 2's complements,
2's complement of subtrahend 0111
$+\ 1$
$(1000)_2$

Add it to minuend 1001
$+\ 1000$
$(10001)_2$

Since additional bit occur,
∴ difference $(0001)_2$ Ans

(9) $(10110)_2 - (011)_2$

1st Complements:

At 1st balancing the eqn,

→ $(10110)_2 - (00011)_2$

1st complement of subtrahend $(11100)_2$

Add it to minuend

$$\begin{array}{r} 11100 \\ +\ 10110 \\ \hline (110010)_2 \end{array}$$

Since, additional bit occur,

difference

$$\begin{array}{r} 10010 \\ +\ 1 \\ \hline (10011)_2 \end{array}$$ Ans

→ 2's complement

2's complement of subtrahend

$$\begin{array}{r} 11100 \\ +\ 1 \\ \hline (11101)_2 \end{array}$$

Add it to minuend

$$\begin{array}{r} 11101 \\ +\ 10110 \\ \hline (110011)_2 \end{array}$$

Since additional bit occurs,

difference $(10011)_2$ Ans (neglect 1)

9's & 10's Complements :

The 9's complement of decimal no. can be obtained by subtracting each digit of number from 9.

For example 9's complement of 3 is 6. (9-3).
and 234 is 765 (999 - 234 = 765).

Steps :

1) Make the both numbers having the same number of digits.
2) Determine the 9's complement of the number ~~from which~~ to be subtracted (subtrahend).
3) Add the 9's complement to the given number from which we subtract (minuend).
4) If additional bit occur remove it and add it to the ~~complement~~ of the result.
5) If additional bit doesn't occur determine 9's complement of result and prefix negative to get final result.

eg: Subtract $(123)_{10}$ from $(345)_{10}$

$(345)_{10} - (123)_{10}$

9's complement of subtrahend $999 - 123 = 876$

Add it with minuend $876 + 345 = 1221$

Since, additional bit occur,

difference $221 + 1 = 222$

$\therefore (222)_{10}$ is the required ans.

10's Complement

The 10's complement of decimal number can be obtained by adding 1 to the least significant digit of 9's complement.

eg: 10's complement of 3 is 7 (9-3+1)

Subtraction using 10's complement:

1. Make the both numbers having same number of digits.
2. Determine 10's complement of the number to be subtracted.

3) Add the 10's complement to the given number from which we subtract.

4) If there exists any additional digit in the result after addition, remove it from the result and the remaining digits form the final result.

5) If there exists no any carry then determines 10's complement of the result and prefix by the negative sign to get final result.

Eg: $(345)_{10} - (123)_{10}$

10's complement of Subtrahend $(999-123)$
$= 876+1 = 877$

Add it to Minuend, $877 + 345 = 1222$

Since, additional bit occurs.
So, difference 222 (neglect 1)
$\therefore (022)_{10}$ is the req ans.

Some other examples:

1) $(20)_2 - (10)_2$ using (r-1)'s complement.

Changing into binary

$20 = (10100)_2$
$10 = (1010)_2$
Here, $(10100)_2 - (1010)_2$

balancing equn,
$(10100)_2 - (01010)_2$
m s

1's complement of subtrahend $(1010)_2$
Add it to minuend
$$\begin{array}{r} 10101 \\ +\ 10100 \\ \hline (101001)_2 \end{array}$$

Since additional bit occur,
difference
$$\begin{array}{r} 01001 \\ +\ \ \ 1 \\ \hline (01010)_2 \end{array}$$

Hence, difference will be $(01010)_2$
i.e $(10)_{10}$

2) $(5)_{10} - (7)_{10}$ using (r-1)'s complement.

9's complement of 7 $(9-7) = 2$

Add it to minuend

$$\begin{array}{r} 2 \\ +\ 5 \\ \hline 7 \end{array}$$

Since, no additional bit occurs.

9's complement of 7 $\rightarrow$

$$\begin{array}{r} 9 \\ -\ 7 \\ \hline 2 \end{array}$$

∴ Difference will be $-(2)_{10}$

3) $(789)_{10} - (890)_{10}$ using r's and (r-1)'s complement.

r's complement

10's complement of 890;

$$\begin{array}{r} 999 \\ -\ 890 \\ \hline 109 \\ +\ \ 1 \\ \hline 110 \end{array}$$

Add it to minuend

$$\begin{array}{r} 110 \\ +\ 789 \\ \hline 899 \end{array}$$

Since, no additional bit occurs,

10's complement of 899 is

$$\begin{array}{r} 999 \\ -\ 899 \\ \hline 100 \\ +\ \ 1 \\ \hline 101 \end{array}$$

∴ Difference will be $-(101)_{10}$.

→ (r-1)'s complement.

9's complements of 890 is

$$\begin{array}{r} 999 \\ -890 \\ \hline 109 \end{array}$$

Add it to minuend,

$$\begin{array}{r} 109 \\ +789 \\ \hline 898 \end{array}$$

Since no additional bit occur,

~~diff~~ 9's complement of 898. is

$$\begin{array}{r} 999 \\ -898 \\ \hline 101 \end{array}$$

∴ Difference is $-(101)_{10}$

(4) $(85)_2 - (70)_2$ using r's complement.

Changing to binary

$85 = (1010101)_2$

$70 = (1000110)_2$

Here,

$(1010101)_2 > (1000110)_2$

2's complement of subtrahend $0111001 + 1 = 0111010$

Add it to minuend $1010101 + 0111010 = 10001111$

Since additional bit occur.

difference $(0001111)_2$

↓

$(15)_{10}$

(5) $(555)_{10}$ from $(75)_{10}$ using 1's complement.

75 = $(1001011)_2$
555 = $(1000101011)_2$

Now,
0001 $(001011)_2$ − $(1000101011)_2$

1's complement of subtrahend 0111010100

0111010100
Add it to minuend + 1001011

1000011111

Since no end bit occurs,
1's complement : 0111100000.
Difference = $-(0111100000)_2$
= $-(480)_{10}$

67 $(1101.10)_2 - (1101.00)_2$ using r's and (r-1)

Using (r-1)

→ 1's complement of subtrahend $(0010.11)_2$

Add it to minuend
$$\begin{array}{r} 0010.11 \\ +1101.10 \\ \hline 10000.01 \end{array}$$

Since additional bit occur,

difference
$$\begin{array}{r} 0000.01 \\ +\quad 1 \\ \hline (0000.10)_2 \end{array}$$

Using r's

2's complement of subtrahend
$$\begin{array}{r} 0010.11 \\ +\quad\quad 1 \\ \hline 0011.00 \end{array}$$

Add to minuend
$$\begin{array}{r} 0011.00 \\ +1101.10 \\ \hline 10000.10 \end{array}$$

Since additional bit occurs,

difference will be $(0000.10)_2$ (neglect 1).

(7) $(80.2)_{10} - (70.2)_{10}$ using r's and (r-1)'s complem

$(80.2)_{10} - (70.2)_{10}$

→ Using r's complement

10's complements of 70.2

$$\begin{array}{r} 99.9 \\ -70.2 \\ \hline 29.7 \\ + \quad 1 \\ \hline 29.8 \end{array}$$

Add it to minuend

$$\begin{array}{r} 29.8 \\ +80.2 \\ \hline 110.0 \end{array}$$

Since add. bit occurs,

difference $(10.0)_{10}$ Ans

→ Using (r-1)'s complements.

9's complement of 70.2

$$\begin{array}{r} 99.9 \\ -70.2 \\ \hline 29.7 \end{array}$$

Add to minuend

$$\begin{array}{r} 29.7 \\ +80.2 \\ \hline 109.9 \end{array}$$

Since add. bit occurs.

difference,

$$\begin{array}{r} 09.9 \\ +\quad 1 \\ \hline (10.0)_{10} \end{array}$$ Ans

(523)x. Find the value of x.

Here,

$x-1 = 5$

$x = 6$

$\therefore x = 6, 7, 8 \text{---}$ so on.

If find only minimal value then,

$x = 6$.

Convert $(62)_x = (50)_{10}$

$6 \times x^1 + 2 \times x^0 = 5 \times 10^1$

or, $6x = 48$

$\therefore x = 8$

$\therefore (62)_8 = (50)_{10}$

→ $(40)_x = (50)_8$

$4 \times x^1 = 5 \times 8^1$

$4x = 40$

$x = 10$

$\therefore (40)_{10} = (50)_8$

Parity

1. even parity
2. odd parity

1. even parity

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

2. odd parity

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

Laws of Boolean Algebra:

	Name	AND Form	OR Form
1)	Identity law	$1.A = A$	$0+A = A$
2)	Null law	$0.A = 0$	$1+A = 1$
3)	Idempotent law	$AA = A$	$A+A = A$
4)	Inverse law	$A\bar{A} = 0$	$A+\bar{A} = 1$
5)	Commutative law	$AB = BA$	$A+B = B+A$
6)	Associative law	$(AB)C = A(BC)$	$(A+B)+C = A+(B+C)$
7)	Distributive law	$A+(BC) = (A+B).(A+C)$	$A(B+C) = AB+AC$
8)	Absorption law	$A(A+B) = A$	$A+AB = A$
		$A+(\bar{A}B) = A+B$	
9)	De-Morgan's law	$\overline{AB} = \bar{A}+\bar{B}$	$\overline{A+B} = \bar{A}.\bar{B}$

Logic gate and truth table

A logic gate is an electronic ckt that operates on one or more input signals to produce an output signal. Alogic gate is also known as bulding block of a digital ckt.

Truth table is a table of all possible combinations of the variables showing the relation between the values that variable may take and the result of the operation.

It is divided into three parts;
1) Basic types (AND, OR, NOT).
2) Universal gates (NAND, NOR).
3) Combinational gates (Ex-OR, Ex-NOR).

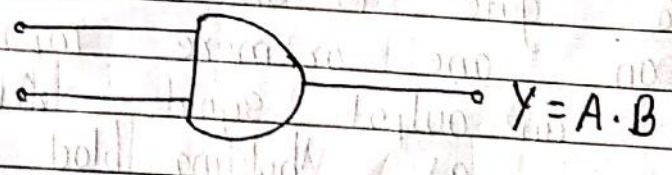
Basic gates

(1) AND gate:

→ AND gate perform logical multiplication.
→ output of AND gate is high if all the input are high otherwise ~~input are low~~. output will be low.

The graphical symbol, logical ckt, algebric expression and truth table of AND gate is shown below;

* Mathematical equn or boolean equn, is
$Y = A \cdot B$.

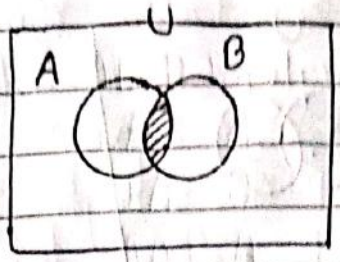
* 

graphical symbol

* Truthtable

A	B	Y = A·B
0	0	0
0	1	0
1	0	0
1	1	1

Venn-diagram



(ii) OR gate

→ It perform logical addition.

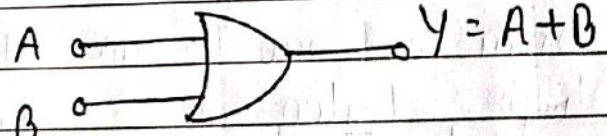
→ The OR Gate contains two or more than two input values which produces only one output value.

→ It's output will be high if any or all input are high.

→ The boolean equation of OR gate is

$$Y = A + B$$

→ Symbol of OR gate is,



→ Truthtable of OR gate,

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

Venn diagram;

iii) NOT gate

→ It performs logical operation or complement.

→ The NOT gate contains only one input value. Each produces only one output value. This gate is also known as inverter.

→ It's output is high if the input is low.

→ The boolean expr. of NOT gate is,

$y = \bar{A}$

→ Symbol;

→ Truth table;

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

→ Venn-diagram:

Universal gate:

NAND and NOR gates are called universal gates because all basic gates can be realized from these gates.

① NOR gate

→ This gate is combination of OR and NOT gate. This gate is complement of the OR function.

→ The output is high if both input are low otherwise output are low.

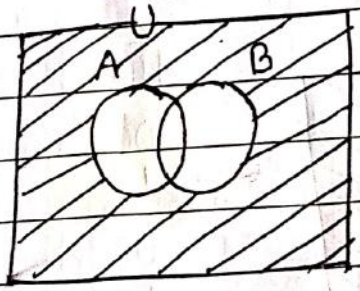
→ It's boolean expression is,

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

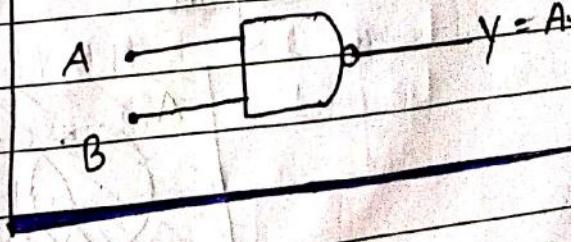
Truth table

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

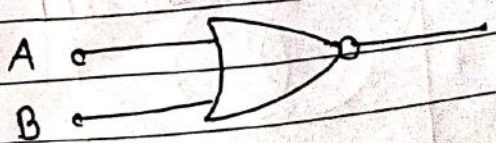
Venne diagram



Symbol NAND



Symbol NOR



NAND gate

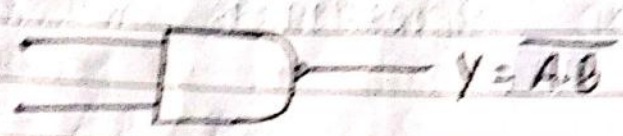
⇒ This gate is the combination of AND gate and NOT gate. This gate is complement of AND function.

⇒ The output is high if any of input or both inputs are low.

⇒ The boolean expression,

$$Y = \overline{A.B}$$

⇒ Symbol :



⇒ Truth table:

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

Venne-diagram

Combination of gates

Ex-OR / X-OR gate

→ It is called combination because all the basic gate function are included in it.

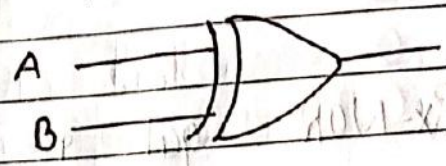
→ the output of exe. exclusive OR gate is high if both input are ~~high~~ different.

→ The boolean expression of Ex-OR gate is

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

or, $Y = A \oplus B$

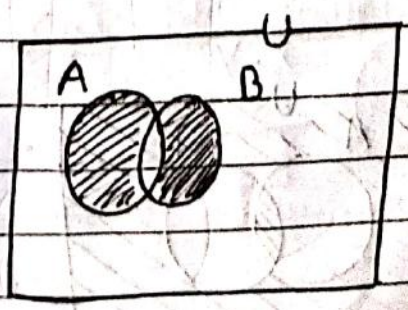
→ The symbol of Ex-OR gate is,



The truth table for Ex-OR gate is,

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

Venne-diagram



$A \oplus B$

7 Exclusive NOR (X-NOR) Gate:

→ The gate contains two or more than two input values which produce only one output value. The 'X-NOR' is the complement of 'X-OR', as indicated by small circle in the graphical symbol.

→ The output of XNOR gate is high if both input are same (i.e high-high or low-low).

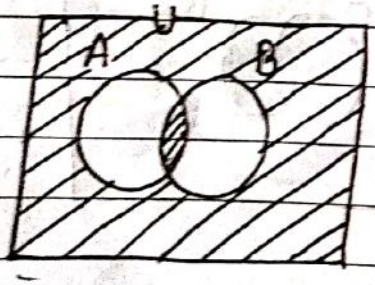
→ The boolean expression for Ex-NOR gate is

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

→ Symbol of Ex-NOR gate is,

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

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



$a \oplus b$

De Morgan's Theorem :

De Morgan's theorem is associated with Boolean algebra, which was given by great and logical mathematician, De-Morgan. So, it is called "De-Morgan's theorem". It consists of first and second theorem.

1) First Theorem (Addition theorem)

It states that "The complement of the sum is equal to the product of complement of individual variable".

If 'x' and 'y' are two variables then,

$$\overline{X+Y} = \overline{X} \cdot \overline{Y}$$

for three variables,

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

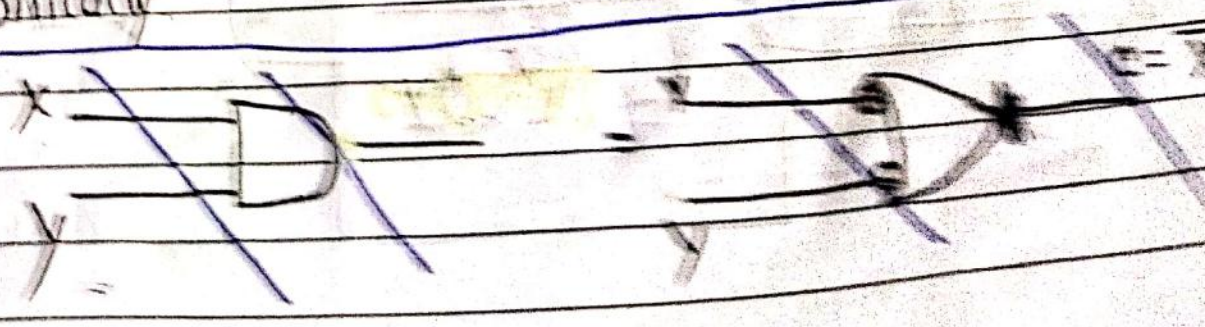
Proof:

X	Y	X+Y	$\overline{X+Y}$	$\overline{X}$	$\overline{Y}$	$\overline{X} \cdot \overline{Y}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

Hence, from above table,

$$\overline{X+Y} = \overline{X} \cdot \overline{Y}$$

graphically



for 3 input: $\overline{X+Y+Z} = \overline{X}\cdot\overline{Y}\cdot\overline{Z}$

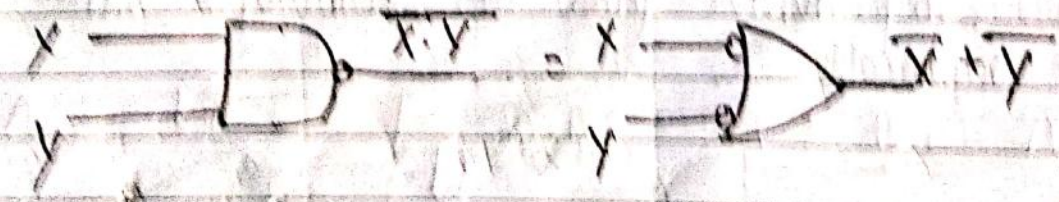
X	Y	Z	X+Y+Z	$\overline{X+Y+Z}$	$\overline{X}$	$\overline{Y}$	$\overline{Z}$	$\overline{X}\cdot\overline{Y}\cdot\overline{Z}$
0	0	0	0	1	1	1	1	1
0	0	1	1	0	1	1	0	0
0	1	0	1	0	1	0	1	0
0	1	1	1	0	1	0	0	0
1	0	0	1	0	0	1	1	0
1	0	1	1	0	0	1	0	0
1	1	0	1	0	0	0	1	0
1	1	1	1	0	0	0	0	0

Proved

87 2nd theorem (Multiplication theorem)

The complement of product term is equal to sum of individual complement. i.e $\overline{X\cdot Y} = \overline{X} + \overline{Y}$

graphically,



Proof:

x	y	$x.y$	$\overline{x.y}$	$\overline{x}$	$\overline{y}$	$\overline{x}+\overline{y}$
0	1	0	1	1	0	1
0	0	0	1	1	1	1
0	1	1	0	0	0	0
1	0	0	1	0	1	1
1						

Hence, $\overline{x}.\overline{y} = \overline{x}+\overline{y}$

For 3-Variable

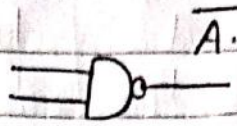
x	y	z	xyz	$\overline{xyz}$	$\overline{x}$	$\overline{y}$	$\overline{z}$	$\overline{x}+\overline{y}+\overline{z}$
0	0	0	0	1	1	1	1	1
0	0	1	0	1	1	1	0	1
0	1	0	0	1	1	0	1	1
0	1	1	0	1	1	0	0	1
1	0	0	0	1	0	1	1	1
1	0	1	0	1	0	1	0	1
1	1	0	0	1	0	0	1	1
1	1	1	1	0	0	0	0	0

Proved

$$\overline{xyz} = \overline{x}+\overline{y}+\overline{z}$$

(#) Realization of universal gate :

1) NAND gate

Symbol 

→ NOT gate

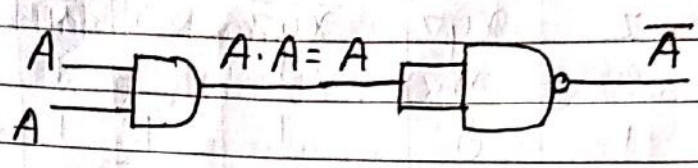
We know for NAND gate,

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

if $A = B$

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

$\therefore Y = \overline{A}$

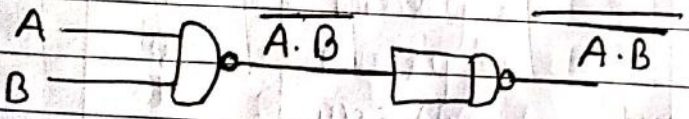
i.e 

→ AND gate

$Y = A \cdot B$

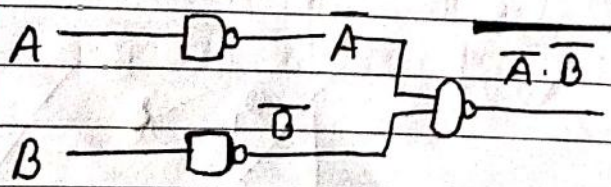
For that the result of NAND gate is

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



→ OR gate; $Y = A + B$

The equivalent ckt using NAND gate.



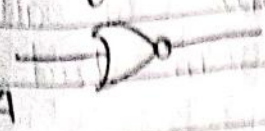
$= \overline{\overline{A} + \overline{B}} = A + B$

(from demorgan's law)

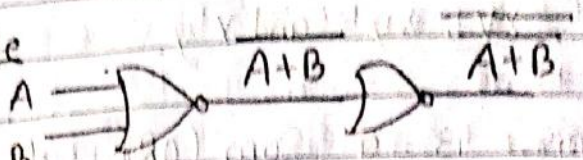
27) NOR gate

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

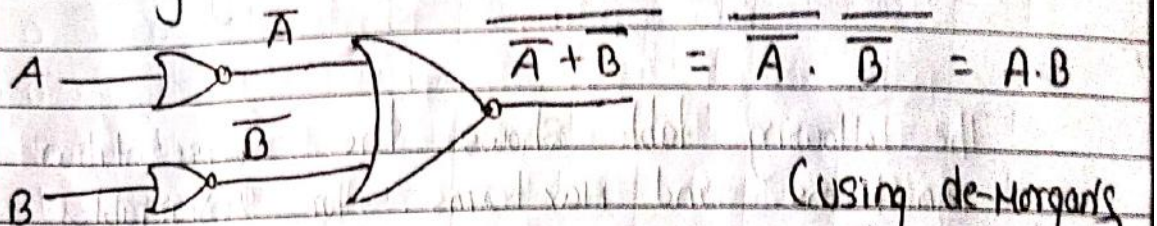
→ NOT gate $Y = \overline{A}$



→ OR gate



→ AND gate $Y = A \cdot B$



(using de-Morgan's law)

Minterms, Maxterms, SOP, POS, Canonical and Standard form:

A minterm is a product (AND) of all variables in the function, in direct or complemented form.

eg: $x'y'$, xy, xy' and $x'y$.

A maxterm is a sum (OR) of all the variables in the function, in direct or complemented form.

eg: $x+y$, $x'+y'$, $x+y'$, $x'+y$.

The following table show the representation of minterms and maxterms for 2-variables,

x	y	Minterms (SOP)	maxterms (POS)
0	0	$m_0 = x'y'$	$m_0 = x'+y$
0	1	$m_1 = x'y$	$m_1 = x+y$
1	0	$m_2 = xy'$	$m_2 = x'+y$
1	1	$m_3 = xy$	$m_3 = x'+y'$

Canonical form

Both Boolean function expressed as a sum of minterms or product of maxterm are called canonical form.
There are two ways of expressing function they are;

1) **Canonical SOP form :**
In this form each product term contains all literals. So, these product terms are nothing but min terms. Hence, canonical SOP form is also called as sum of min terms form.
The corresponding minterms are $p'qr$, $pq'r$, pqr', pqr. By doing logical OR of these four minterms, we will get Boolean function of output.

We can represent this function in following two notations;

$$f = m_3 + m_5 + m_6 + m_7$$
$$f = m_3 + m_5 + m_6 + m_7$$
$$f = \Sigma m(3, 5, 6, 7)$$

2) **Canonical POS form :**
In this form, each sum term contains all literals. Hence, canonical POS is the product of Max-term form.

$$(p+q).(p+q) = 1$$

Corresponding maxterms are $p+q+r$, $p+q'+r$, $p+q'+r'$, $p'+q+r$.
By doing logical 'AND' of these maxterm we will get the Boolean function of output.

Notations:

$F = M0, M1, M2, M4$

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

Both canonical SoP and canonical PoS forms are dual to each other, functionality, these two forms are same. Based on requirements, we can use one of these two forms.

Standard SoP and PoS forms.

These are the simplified version of canonical forms. The main advantage of standard form is that the number of input applied to logic gates can be minimized. Sometime there will be reduction in total number of logic gates required.

Standard SoP form

In this form, each product term need not contain all literals. So the product terms may or may not be in minterms. Therefore the standard SoP form is the simplified form of canonical SoP form.

eg:- Convert the following boolean function into standard SOP form.

$F = pqr + pq'r + pr + p'q'r$

$= p'qr + pq'r + pqr + pq'r + pqr + p'q'r$

$(x + x + x = x)$

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

$= qr \cdot 1 + pr \cdot 1 + pq \cdot 1$

$= pq + qr + pr$

This is simplified boolean function.

* Standard POS form:

In this form, each sum terms need not contain all literals. ie. the sum terms may not be max terms. ∴ the standard POS form is the simplified form of canonical POS form.

Some questions:

To change Canonical into SOP or POS

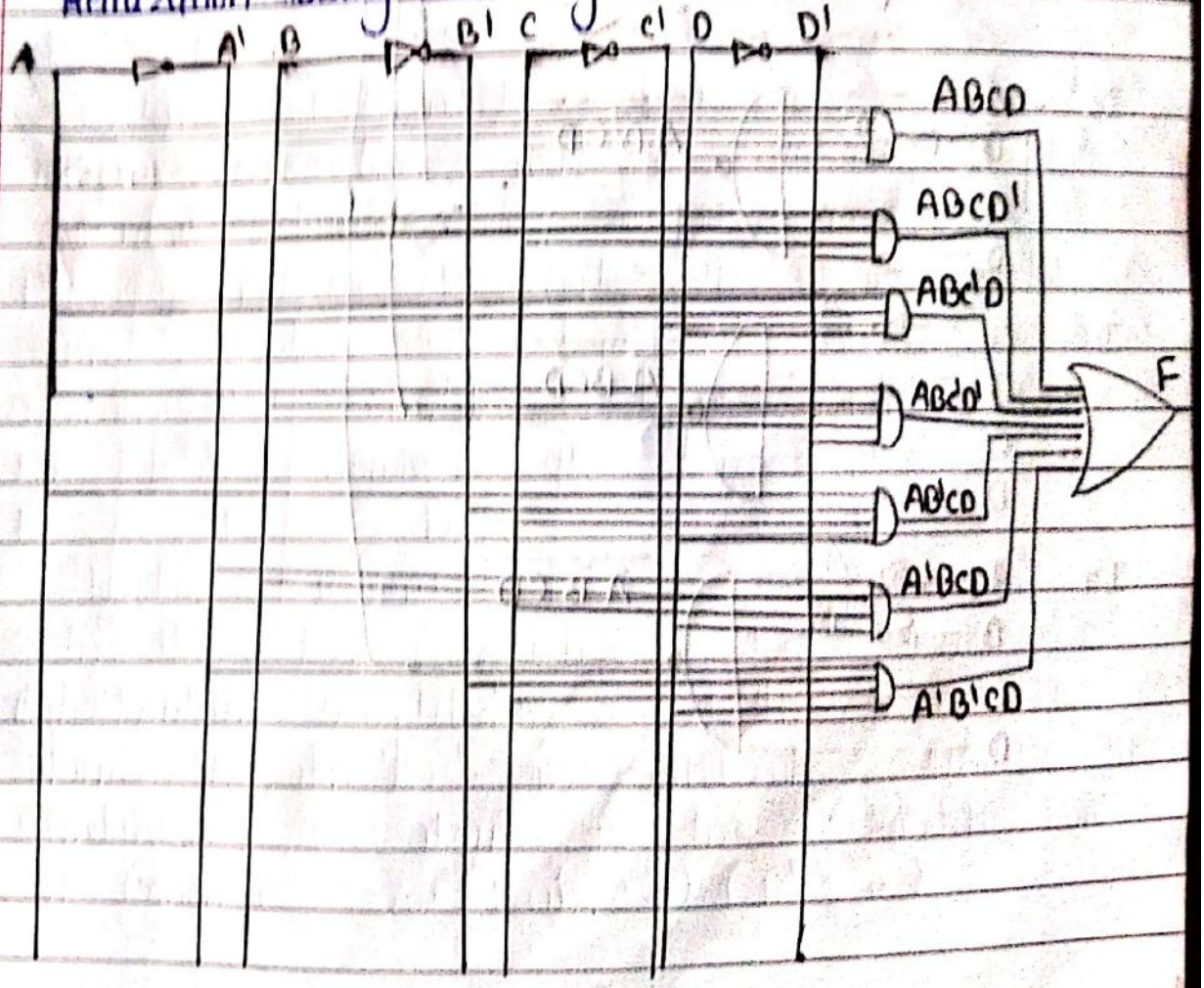
Note: In SOP use X 1
In POS use + 0

1) F = AB + CD change into Canonical SOP

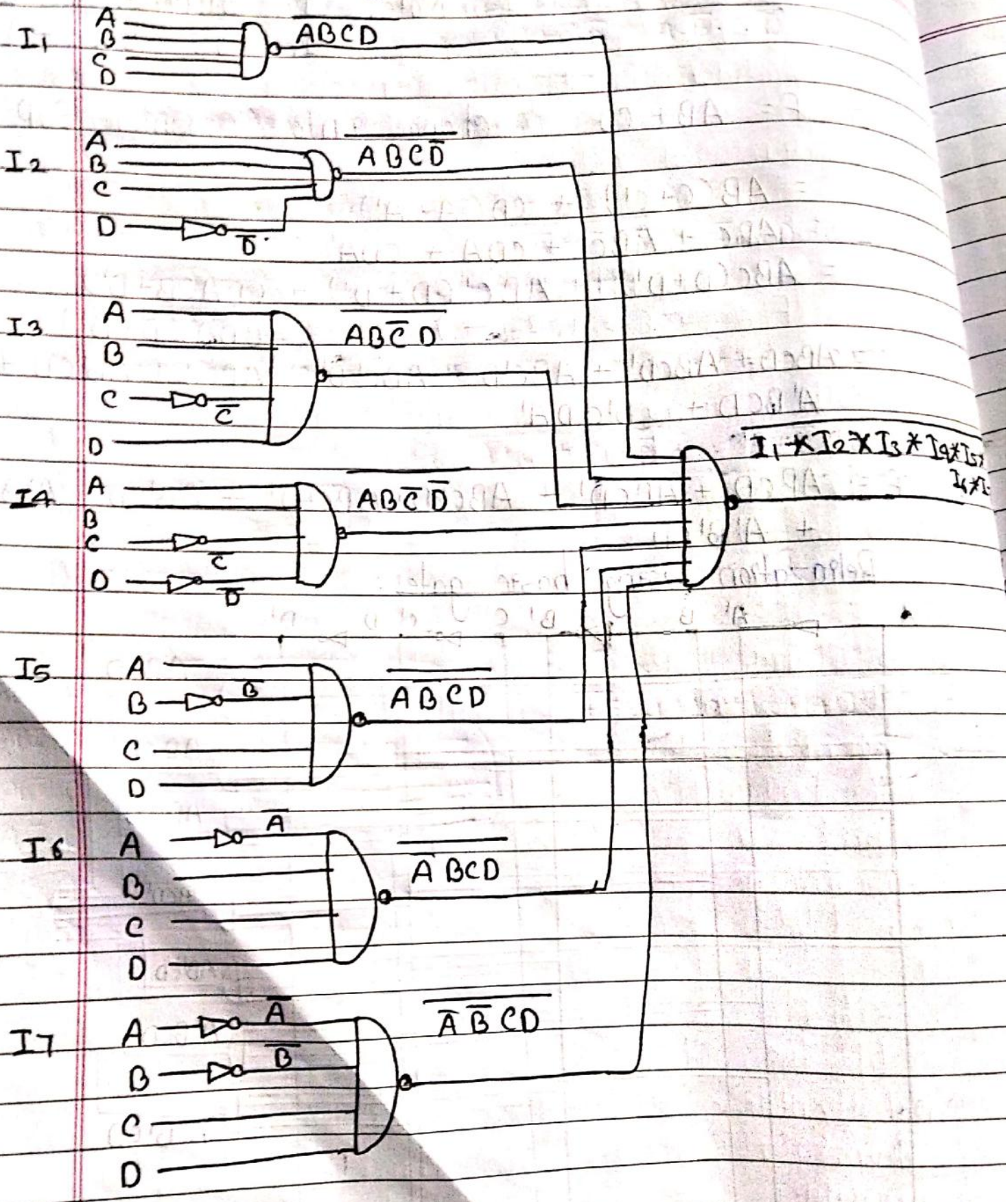
= AB(C+C') + CD(A+A')
= ABC + ABC' + CDA + CDA'
= ABC(D+D') + ABC'(D+D') + CDA(B+B') + CDA'(B+B')
= ABCD + ABCD' + ABC'D + ABC'D' + ABCD + ABCD + A'BCD + B'CDA'

F = ABCD + ABCD' + ABC'D + ABC'D' + AB'CD + A'BCD + A'B'CD

Realization using basic gates:



Realization using NAND gate:



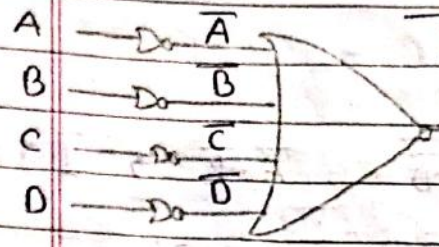
$= \overline{ABCD} \cdot \overline{ABC\bar{D}} \cdot \overline{AB\bar{C}D} \cdot \overline{AB\bar{C}\bar{D}} \cdot \overline{A\bar{B}CD} \cdot \overline{\bar{A}BCD} \cdot \overline{A\bar{B}\bar{C}D}$

Using deMorgan's law,

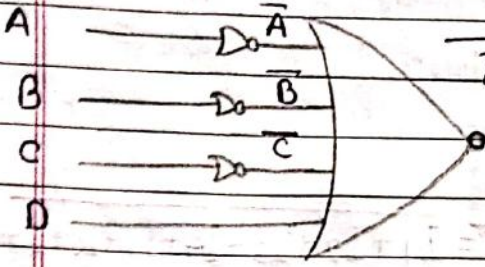
$= \overline{\overline{ABCD}} + \overline{\overline{ABC\bar{D}}} + \overline{\overline{AB\bar{C}D}} + \overline{\overline{AB\bar{C}\bar{D}}} + \overline{\overline{A\bar{B}CD}} + \overline{\overline{\bar{A}BCD}} + \overline{\overline{\bar{A}\bar{B}CD}}$

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

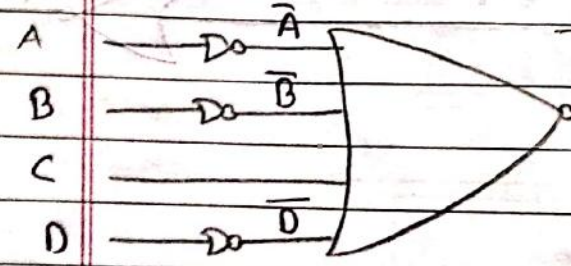
Realization using NOR gate



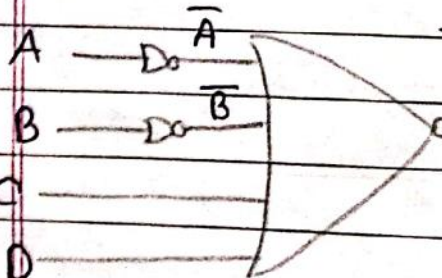
$\overline{\bar{A}+\bar{B}+\bar{C}+\bar{D}} = \bar{\bar{A}} \cdot \bar{\bar{B}} \cdot \bar{\bar{C}} \cdot \bar{\bar{D}} = ABCD = I_1$



$\overline{\bar{A}+\bar{B}+\bar{C}+D} = \bar{\bar{A}} \cdot \bar{\bar{B}} \cdot \bar{\bar{C}} \cdot \bar{D} = ABC\bar{D} = I_2$

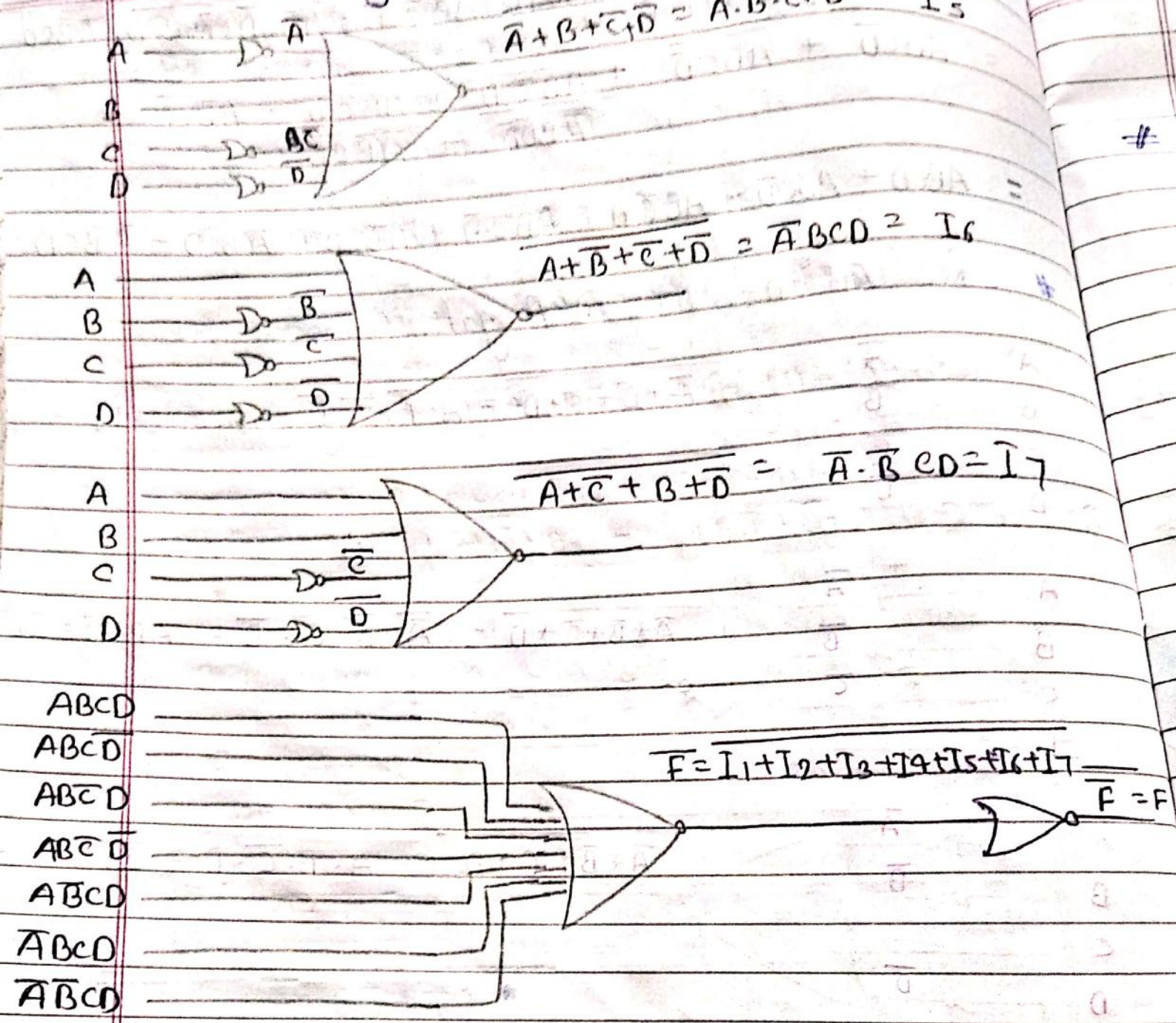


$\overline{\bar{A}+\bar{B}+C+\bar{D}} = A \cdot B \cdot \bar{C} \cdot D = I_3$



$\overline{\bar{A}+\bar{B}+C+D} = A \cdot B \cdot \bar{C} \cdot \bar{D} = I_4$

Realization using NOR Gate:



classmate
Date
Page
otherwise!

Note: To Convert 'SoP to PoS' add '0' for complement
To Convert 'PoS' to SoP ~~add~~ Multiply '0'.

change $F = A+\bar{B}+C+\bar{D}$ into SoP.

$= A\cdot 1 + \bar{B}\times 1 + C\cdot 1 + \bar{D}\cdot 1$

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

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

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

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

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

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

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

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

→ Change into POS

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

$= (A+C+0)\cdot(B+D+0)$

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

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

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

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

(#) Change into POS.

$F = \overline{A+BC}$

$= \overline{A+BC}$

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

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

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

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

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

POS (Product of Sum)

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

$= (A+B+0) \cdot (C+0)$

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

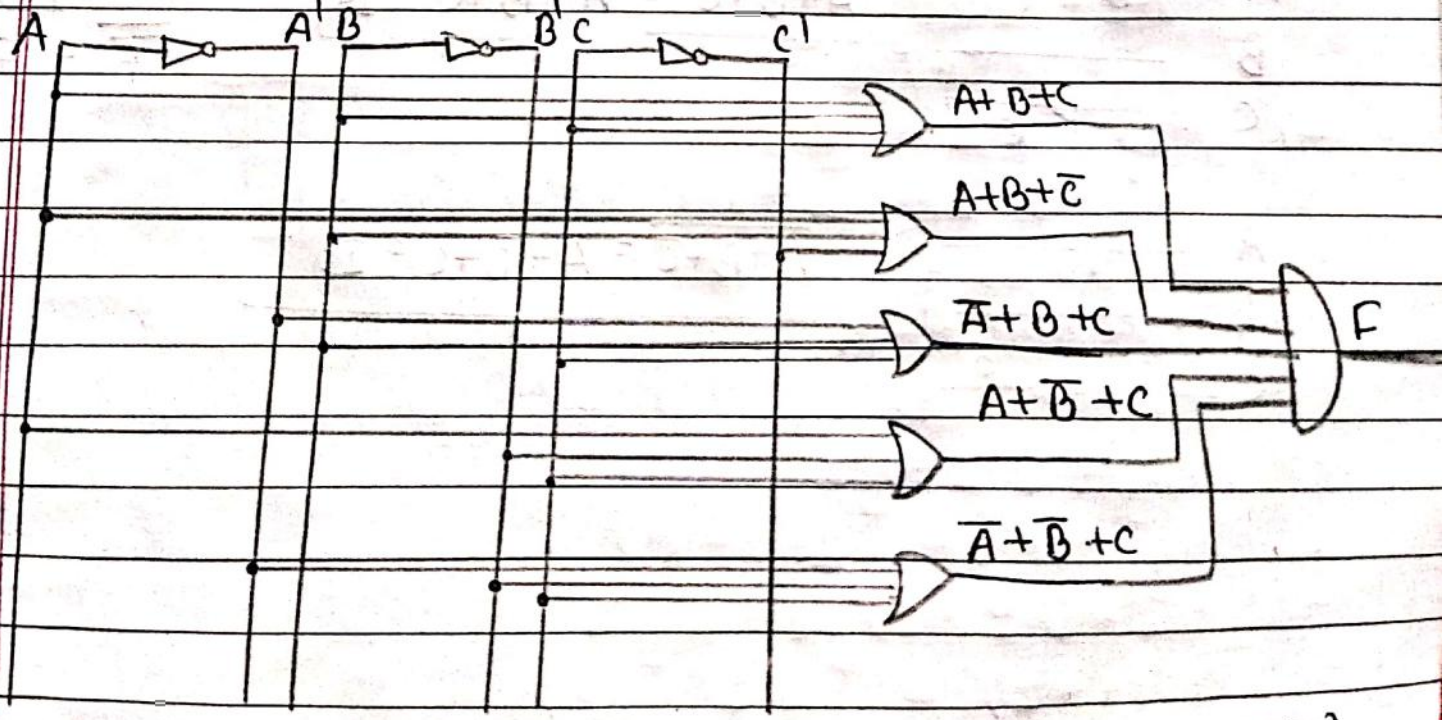
$= (A+B+C) \cdot (A+B+\overline{C}) \cdot (C+B) \cdot (C+\overline{B})$

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

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

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

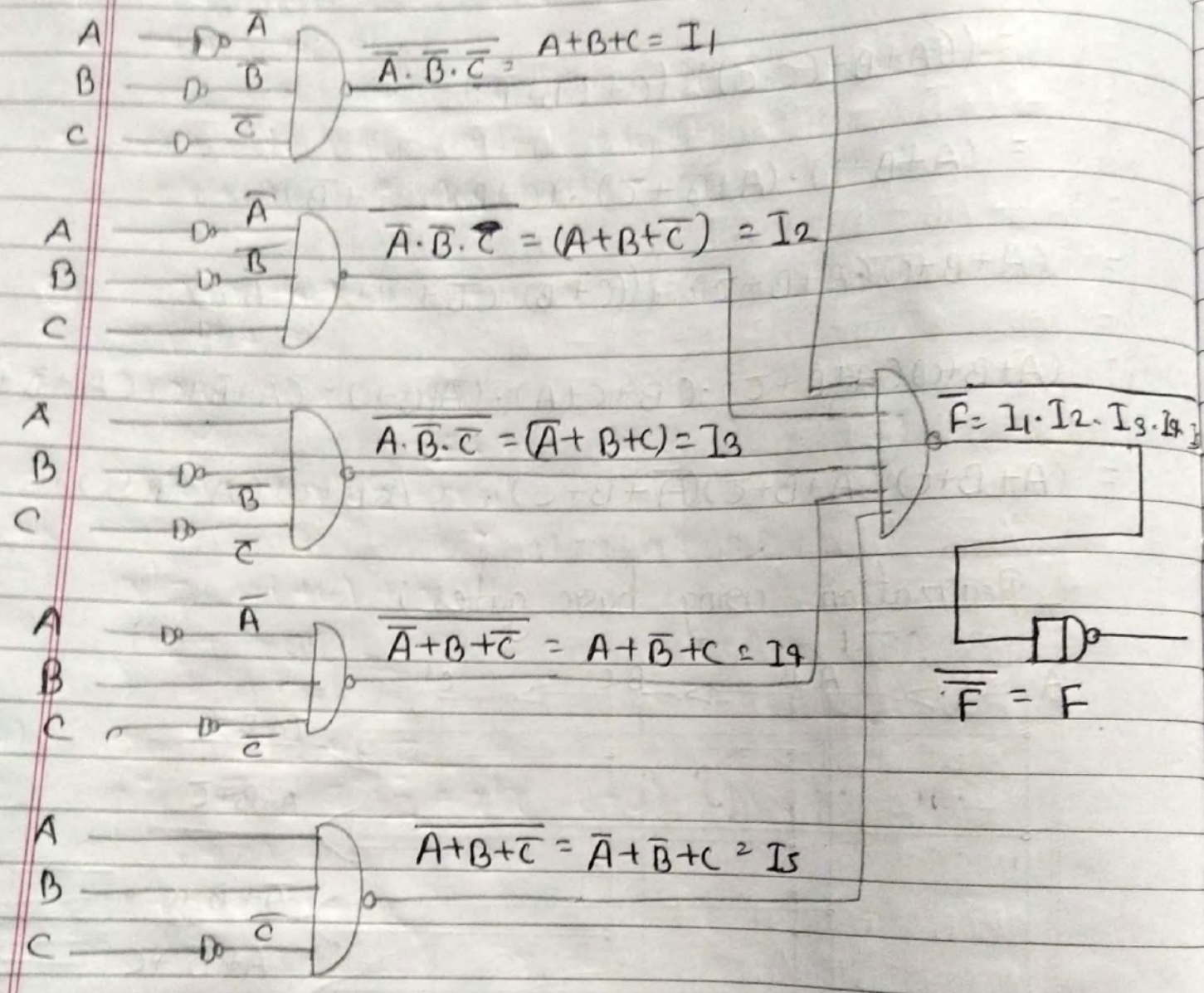
Realization using basic gates;



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

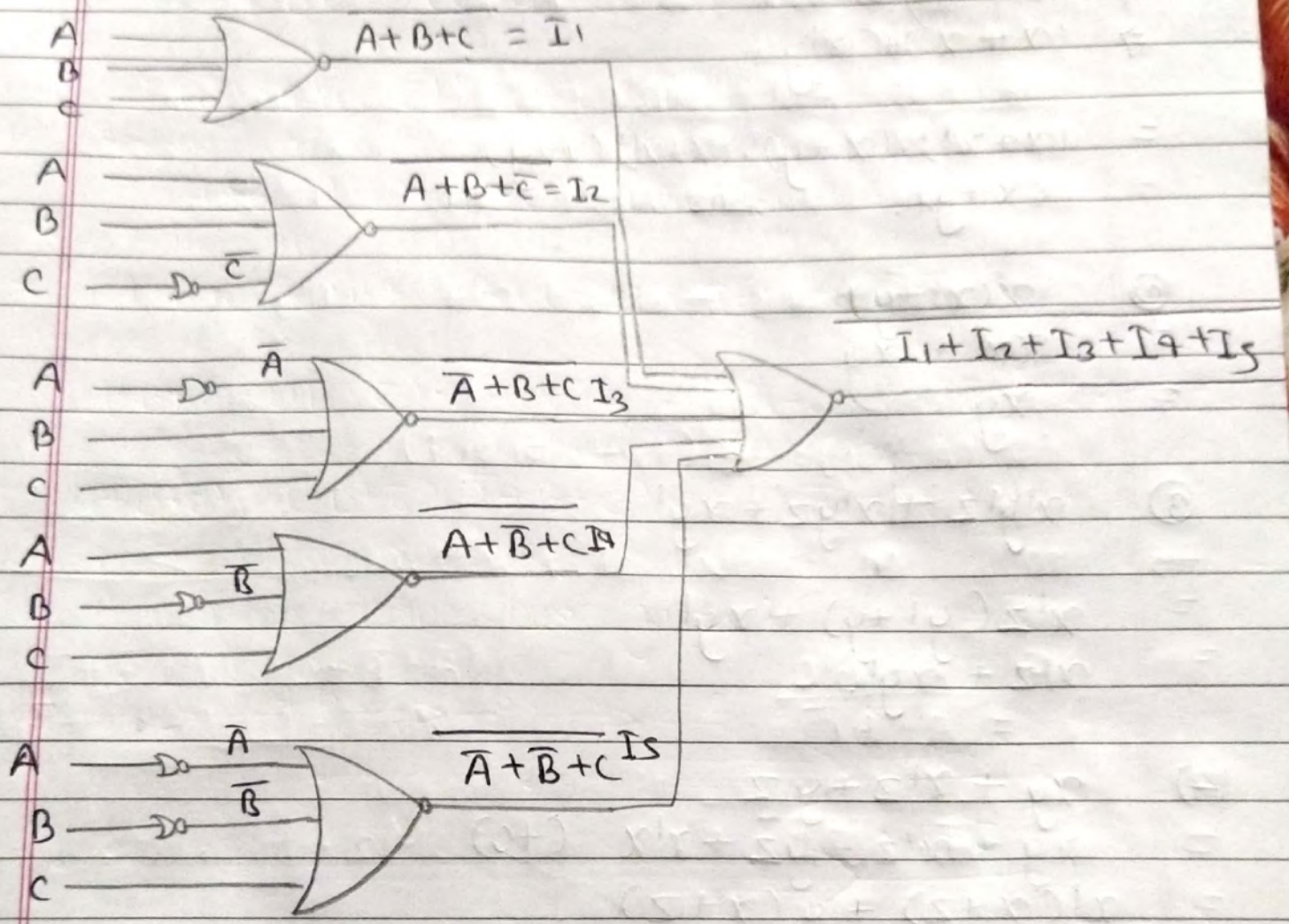
Using NAND gate

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



Using NOR gate :

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



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

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

$$= F$$

Ex+

Simplify the following Boolean function to minimum no of literals:

1) $x + x'y$

$\Rightarrow (x + x') + (x'y)$

$= (x+x').(x+y).(x+x')(x+y)$

$= (x+y)$ Ans

2) $x(x'+y)$

$= xx' + xy$

$= xy$

3) $x'y'z + x'yz + xy'$

$= x'z(y'+y) + xy'$

$= x'z + xy'$.

4) $xy + x'z + yz$

$= xy + x'z + yz + x'x$ $(+0)$

$= x'(x+z) + y(x+z)$

$= (x+z)(x'+y)$.

or,

5) $xy + x'z + yz$

$= xy + x'z + yz(x+x')$

$= xy + x'z + xyz + x'yz$

$= xy(1+z) + x'z(1+y)$

$= xy + x'z$

$(1+z = 1)$

$x + xy = x$

6) $(x+y)(x'+z)(y+z)$

$= (x+y)(x'+z)(y+z+0)$

$= (x+y)(x'+z)(y+z+x.x')$

$= (x+y)(x'+z)(x+y+z).(x'+y+z)$

(X, Y, X, Y)

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

$= x(x+z).\ y(y+y)$

$X.Y$

$= (x+y)(x'+z)$

$[\ x+xy = x$
$\ x.(x+y) = x$
absorption law)

7) $F = (x+y).(x'+y)+(x+y')$

$= xx' + x'y + xy + y + x + y'$

$= y(x+x') + y + x + y'$

$= xy + x'y + x$

$= x$

Express the Boolean function $F = A + B'C$ in a sum of minterms.

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

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

$= ABC + ABC' + AB'C + AB'C' + AB'C + A'B'C$

$= ABC + ABC' + AB'C + AB'C' + A'B'C$

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

$\therefore F(A, B, C) = \Sigma(1, 4, 5, 6, 7)$

→ in a product of maxterms.

$F = A + B'C$

$= \overline{A + B'C}$

$= \overline{A} \cdot \overline{B'C}$

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

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

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

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

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

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

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

$= M_2 \cdot M_3 \cdot M_0$

$\Rightarrow F = \Pi(0, 2, 3)$

Minimum no of literals

→ $xyz + x'y + xyz'$

$= xyz + x'yz + x'yz' + xyz'$

$= xy + x'y$

$= y.$

Converting exp^n to TT.
(POS)

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

input	output	Maxterms
0 0 0	0	A+B+C
0 0 1	1	
0 1 0	0	A+B'+C
0 1 1	0	A+B'+C'
1 0 0	1	
1 0 1	0	A'+B+C'
1 1 0	0	A'+B'+C
1 1 1	1	

$x = \pi(M_0 \cdot M_2 M_3 M_5 M_6)$

(#) SOP

$F = ABC + ABC' + AB'C + AB'C' + A'B'C$

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

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

I/P	O/P	Minterms
000	0	$A'B'C$
001	1	
010	0	
011	0	
100	1	$AB'C'$
101	1	$AB'C$
110	1	$AB'C'$
111	1	ABC

(*) K-Map or (Map) is a diagram made up of squares with each square representing one minterms of a function.
The map method provides a straight forward procedure for minimizing Boolean eqon.