UNIT-1
BASICS OF LOGIC DESIGN

Number system

A number system is a writing system for expressing numbers that is a mathematical notation for representing number of given set using digits or other symbols in a consistent manner.

Types of number system

There are four number systems they are as follows :-

* Decimal number system (0-9) Base 10
* Binary number system (0-1) Base 2
* Octal number system (0-7) Base 8
* Hexa decimal number system [(0-9) A, B, C, D, E, F] Base 16

* Decimal number system (0-9) Base 10

Decimal number system is composed of 10 numbers (symbols) they are (0, 1, 2, 3, 4, 5, 6, 7, 8, 9). By using these numbers we can express any quantity. The decimal system is also called as Base 10 system this is because it ~~is~~ has 10 digits. It is a Positional value system where in each digit has its own value or weight. Expressed in power of 10. The following figure illustrates its concept.

Binary weight	10^4	10^3	10^2	10^1	10^0	. 10^{-1}	10^{-2}	10^{-3}
Decimal value	10000	1000	100	10	1	.0.1	0.01	0.001
	↑ M.S.D							↑ L.S.D

* Binary number system (0-1) Base 2

Binary number system contains 2 numbers or 2 symbols that is 0 and 1. It can be used to express any quantity that can be represented in decimal system or any other system. The binary system is also called as base 2 system. This is because which has only 2 digits 0 and 1. It is a positional value system. where each digit has its own value or weight expressed as power of 2. The following figures illustrates it.

Binary weight	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}
Decimal value	16	8	4	2	1	0.5	0.25	0.125
	↑ M.S.B							↑ L.S.B

* Octal number system (0-7) Base 8

Octal number system contains 8 numbers or symbols they are 0,1,2,3,4,5,6,7 it can be used to express any quantity that can be represented in decimal system or any other system. The octal system is also called as Base 8 system. This is because it has only 8 digits. It is a positional value system. where each digit has its own value or weight expressed as power of the following diagram illustrates it.

Octal weight	8^4	8^3	8^2	8^1	8^0	8^{-1}	8^{-2}	8^{-3}	8^{-4}
Decimal value	4096	512	64	8	1	0.125	0.0156	1.953×10^{-3}	2.441×10^{-4}
	↑ M.S.D								↑ L.S.D

* Hexa decimal number system (0-9 ABCDEF) Base 16

Hexa decimal number system contains 16 numbers or symbols that is 0,1,2,3,4,5,6,7,8,9, a,b,c,d,e,f. It can be used to express any quantity that can be represented in decimal system or any other system. The Hexa decimal number system is also called as Base 16 system. This is because it has only 16 digits. It is a positional value system where each digit has its own value or weight expressed as power of 16. The following figure illustrated it.

Hexadecimal weight	16^4	16^3	16^2	16^1	16^0	. 16^{-1}	16^{-2}	16^{-3}
Decimal Value	655 36	4096	256	16	1	. 0.062	0.0039	0.00024

Conversion from one system to other system

1) Conversion from Binary to Decimal

While converting ~~from~~ binary to decimal each digit must be multiplied by its weight from LSD. And the resulting product should be added.

Ex:- ① convert $(1101)_2 = (13)_{10}$

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

$8 + 4 + 0 + 1$

$= (13)_{10}$

② convert $(101101)_2 = (45)_{10}$

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

$32 + 0 + 8 + 4 + 0 + 1$

$(45)_{10}$ $\therefore (101101)_2 = (45)_{10}$

③ convert $(10110011)_2 = (179)_{10}$

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

$128 + 0 + 32 + 16 + 0 + 0 + 2 + 1$

$(179)_{10}$

$\therefore (10110011)_2 = (179)_{10}$

128
32
16
3
179

④ Convert $(0.1101)_2 = (0.8125)_{10}$

$= .1\times2^{-1} + 1\times2^{-2} + 0\times2^{-3} + 1\times2^{-4}$

$= .0.5 + 0.25 + 0 + 0.625$

$= (0.8125)_{10}$

$\therefore (0.1101)_2 = (0.8125)_{10}$

2
4
8
16
32
64
128

⑤ Convert $(1001.101)_2 = (9.625)_{10}$

⑥ $1\times2^3 + 0\times2^2 + 0\times2^1 * 1\times2^0 . 1\times2^{-1} + 0\times2^{-2} + 0\times2^{-3}$

$8 + 0 + 0 + 1 . 0.5 + 0 + 0.125$

9.625

$\therefore (1001.101)_2 = (9.625)_{10}$

⑥ Convert $(1101.110)_2 = (13.875)_{10}$

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

$8 + 4 + 0 + 1 . 0.5 + 0.25 + 0.125$

13.0875

$\therefore (1101.110)_2 = (13.875)_{10}$

2) Conversion from Octal to Decimal

When converting from Octal to decimal, each digit must be multiplied by its weight (8), and the resulting product should be added.

Problems on converting octal number to decimal numbers

① $(375)_8 = (253)_{10}$

$3\times8^2 + 7\times8^1 + 5\times8^0$

$192 + 56 + 5$

$(253)_{10}$

$\therefore (375)_8 = (253)_{10}$

② $(0.325)_8 = (0.416)_{10}$

$= 3\times8^{-1} + 2\times8^{-2} + 5\times8^{-3}$

$= 3\times0.125 + 2\times0.0156 + 5\times0.001953$

$0.375 + 0.03125 + 0.009765$

0.4160

$\therefore (0.325)_8 = (0.416)_{10}$

③ $(24.6)_8 = (20.75)_{10}$

$2\times8^1 + 4\times8^0 . 6\times8^{-1}$

$16 + 4 . 0.75$

$(20.75)_{10}$

$\therefore (24.6)_8 = (20.75)_{10}$

3) Conversion from Hexa decimal to decimal

When converting from hexa decimal to decimal, each digit must be multiplied by its weight (Base 16) and the resulting product should be added. The digits should be multiplied by power of 16 i.e Base 16.

Convert the following Hexa decimal number into decimal.

① $(185)_{16} = (389)_{10}$

$1 \times 16^2 + 8 \times 16^1 + 5 \times 16^0$

$256 + 128 + 5$

389

$\therefore (185)_{16} = (389)_{10}$

② $(A85)_{16} = (2693)_{10}$

$A = 10$

$10 \times 16^2 + 8 \times 16^1 + 5 \times 16^0$

$2560 + 128 + 5$

2693

$\therefore (A85)_{16} = (2693)_{10}$

③ $(2B7.75)_{16} = (695.4535)_{10}$

$B = 11$

$2 \times 16^2 + 11 \times 16^1 + 7 \times 16^0 \cdot 7 \times 16^{-1} + 5 \times 16^{-2}$

$2 \times 256 + 176 + 7 \cdot 0.4375 + 0.01953$

$695 \cdot 0.457$

$\therefore (2B7.75)_{16} \approx (695.45)_{10}$

④ $(980.65A)_{16} = (2445.39)_{10}$

$9 \times 16^2 + 8 \times 16^1 + 0 \times 16^0 \cdot 6 \times 16^{-1} + 5 \times 16^{-2} + A \times 16^{-3}$

$9 \times 256 + 8 \times 16 + 0 \cdot 6 \times 0.0625 + 5 \times 0.00390 + 10 \times 0.000244$

$2304 + 128 + 0 \cdot 0.375 + 0.0195 + 0.00244$

$2432 \cdot 0.39694.$

Decimal to Binary system.

The conversion from decimal to binary is performed by dividing the decimal numbers successively by 2 and writing the remainder. When the quotient is reached less than 2, the conversion is complete the remainder are taken in the reverse order that represents the equivalent binary number.

Fractional Part Conversion.

Fraction is multiplied by 2 and carry is recorded in the integer position. The process of multiplication is continued until the desired accuracy is reached. Reading the carry downwards represents equivalent Binary fraction.

Convert the following decimal numbers into binary numbers

① $(12)_{10} = (1100)_2$

2	12	- 0
2	6	- 0
2	3	- 1
	1	

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

② $(0.45)_{10} = (0.0111)_2$

$0.45 \times 2 = 0.9$
$0.9 \times 2 = 1.8$
$0.8 \times 2 = 1.6$
$0.6 \times 2 = 1.2$

$\therefore (0.45)_{10} = (0.0111)_2$

③ $(232.52)_{10} = ($

2	232	
2	116	- 0
2	58	- 0
2	29	- 0
	14	- 1

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

(11101000)

0.52 × 2 = 1.04
0.44 × 2 = 0.08
0.08 × 2 = 0.16
0.16 × 2 = 0.32
1000

$(232.52)_{10} = (11101000.1000)_2$

④ $(786)_2 = (518)_{10}$

$7 \times 8^2 + 8 \times 8^1 + 6 \times 8^0$

$448 + 64 + 6$

518

$\therefore (786)_2 = (518)_{10}$

⑤ $(AB2.17B)_{16} = (2738.1166)_{10}$

$A \times 16^2 + B \times 16^1 + 2 \times 16^0 \cdot 1 \times 16^{-1} + 7 \times 16^{-2} + B \times 16^{-3}$

$10 \times 16^2 + 11 \times 16^1 + 2 \times 16^0 \cdot 1 \times 16^{-1} + 7 \times 16^{-2} + B \times 16^{-3}$

$10 \times 256 + 176 + 2 \cdot 1 \times 0.0625 + 0.02734 + 0.00268$

$2738 \cdot (0.0925)$

$\therefore (AB2.17B)_{16} = (2738.0925)_{10}$

Decimal to Octal Conversion

The procedure is same as conversion from decimal to binary except dividing by 8.

Ex:- ① $(359)_{10} = (547)_8$ (high, low)

8 | 359
8 | 44 - 7
8 | 5 - 4
0 - 5

$\therefore (359)_{10} = (547)_8$

② $(0.23)_{10} = (0.1656)_8$

0.23 × 8 = 1.84
0.84 × 8 = 6.72
0.72 × 8 = 5.76
0.76 × 8 = 6.08
0.08 × 8 = 0.64

$\therefore (0.23)_{10} = (0.1656)_8$

Decimal to hexa decimal conversion

The procedure is same as conversion from decimal to Octal except dividing by 16.

① $(423.94)_{10} = (1A7.F0A3)_{16}$

16 | 432		
16 | 26	-7	
1	-10	

$0.94 \times 16 = 15.04$
$0.04 \times 16 = 0.64$
$0.64 \times 16 = 10.24$
$0.24 \times 16 = 3.82$

$\therefore (423.94)_{10} = (1A7.F0A3)_{16}$

Octal to Binary conversion:

While converting from Octal to binary change each octal digit to its binary equivalent for each individual digit that is each digit is replaced with 3 Bit Binary numbers. The 3 Bit Binary equivalent Octal number system is given in the table.

Decimal digit	Binary number
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

Ex ① $(765)_8 = (111\ 101\ 110)_2$
Octal Binary

② $(747.476)_8 = (111\ 100\ 111.100\ 111\ 110)_2$
Octal Binary

Hexa decimal to Binary conversion

Change each hexa decimal digit to its Binary equivalent for each individual digit i.e each digit as replaced with 4 Bit Binary number. The 4 Bit Binary equivalent hexa decimal system is given in the table

Hexa decimal digit	Binary number
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
A	1 0 1 0
B	1 0 1 1
C	1 1 0 0
D	1 1 0 1
E	1 1 1 0
F	1 1 1 1

Ex :-

① $(756)_{16} = (0111\ 0101\ 0110)_2$
Hexa → Binary

② $(AFC8 \cdot 2DE)_{16} = (1010\ 1111\ 1100\ 1000 \cdot 0010\ 1101\ 1110)_2$
Hexa → Binary

③ $(FBD8 \cdot DFA3)_{16} = (1111\ 1011\ 1101\ 1000 \cdot 1101\ 1111\ 1010\ 0011)_2$

Binary to Octal Conversion

While converting from binary to Octal,

Group the bits in three bits

Ex :- ① $(010111)_2 = (27)_8$

② $(011\ 110\ 010)_2 = (362)_8$

③ $(110101011.0101)_2 = (653.24)_8$

④ $(011\ 101\ 010\ 1100 . 0111\ 0101\ 0)_2 = 0(7254.352)$

Binary to Hexa decimal Conversion

While converting from Binary to Hexa decimal, Group the Bits in four bits.

Ex ① $(0111\ 0101\ 0110\ 0 . 0111\ 0101\ 0)_2 = (EAC.75)_{16}$
Binary → Hexa

② $(011\ 1011\ 1011 . 0111\ 01)_2 = (3BB.74)_{16}$

Solve the Problems

① $(671)_{10} = (\quad)_2$

2	671	
2	335	1
2	167	1
2	83	1
2	41	1
2	20	1
2	10	0
2	5	0
2	2	1
	1	0

$\therefore (671)_{10} = (1010011111)_2$

② $(11010)_2 = (\quad)_{10}$

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

$1\times16 + 1\times8 + 0\times4 + 1\times2 + 0\times1$

$16 + 8 + 0 + 2 + 0$

26

$\therefore (11010)_2 = (26)_{10}$

③ $(127662)_8 = (\quad)_{10}$

$1\times8^5 + 2\times8^4 + 7\times8^3 + 6\times8^2 + 6\times8^1 + 2\times8^0$

$32768 + 2\times4096 + 7\times512 + 6\times64 + 6\times8 + 2\times1$

$32768 + 8192 + 3584 + 384 + 48 + 2$

$44,978$

$\therefore (12766\ 2)_8 = (44,978)_{10}$

④ $(010111)_2 = (\quad)_8$

$\frac{010}{2}\ \frac{111}{7}$

$\therefore (010111)_2 = (27)_8$

000 | 0111 | 0111 | 1011 | 1110
1 7 7 B D

⑤ $(5112)_{10} = (\quad)_{16}$

16 | 5112
16 | 319 - 8
16 | 19 - 15
1 - 3

$\therefore (5112)_{10} = (3F8)_{16}$

0101 | 0011 | 1011
0010 | 1101 | 1011
2 D B

⑥ $(894.45)_{10} = (\quad)_8$

8 | 894
8 | 111 - 6
8 | 13 - 7
8 | 1 - 5
0 - 1

$0.45 \times 8 = 3.6$
$0.6 \times 8 = 4.8$
$0.8 \times 8 = 6.4$
$0.4 \times 8 = 3.2$

$\therefore (894.45)_{10} = (1576.3463)_8$

⑦ $(894.45)_{10} = (\quad)_{16}$

8 | 894
8 | 111 - 6
8 | 13 - 7
8 | 1 - 5
0 - 1

16 | 894
16 | 55 - 14
16 | 3 - 7
0 - 3

$0.45 \times 16 = 7.2$
$0.2 \times 16 = 3.2$
$0.2 \times 16 = 3.2$

$\therefore (894.45)_{10} = (37E.733)_{16}$

⑧ $(5274)_8 = (\quad)_2$

$(5274)_8 = (101\ 010\ 111\ 100)_2$

⑨ $(A2F4)_{16} = (\quad)_2$

$(A2F4)_{16} = (1010\ 0010\ 1111\ 0100)_2$

⑩ $(B3A2)_{16} = (\quad)_2$

$(B3A2)_{16} = (1011\ 0011\ 1010\ 0010)_2$

⑪ $(11.10001)_2 = (\quad)_8$

$011 . 100010$

$(11.10001)_2 = (3.42)_8$

⑫ $(011111.101)_2 = (\quad)_{16}$

$00011111 . 1010$

$(011111.101)_2 = (1F.A)_{16}$

⑬ $(1101101)_2 = (\quad)_{10}$

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

$64 + 32 + 0 + 8 + 4 + 0 + 1$

109

$\therefore (1101101)_2 = (109)_{10}$

Other code for Data representation.

* BCD [Binary coded Decimal]
* Gray code
* Excess 3 code
* ASCII code [American standard code for Information Interchange]
* EBCDIC [Extended Binary code decimal Interchange code]

BCD [Binary coded Decimal]

Bit = 1 bit
Nibble = 4 bits
Byte = 8 bits
Word = 16 bits
Double word = 32 bits
Quad word = 64 bits

Decimal digit	BCD			
	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

In Binary coded decimal each decimal digit is expressed with Binary equivalent code. The resulting code is called BCD (Binary coded decimal). In BCD system there are only 10 code groups where in each decimal digit is changed to binary equivalence. The table shown above is for BCD code for equivalent decimal digit

The BCD code is relatively easy to convert to form decimal

8421 code

8421 code is a type of BCD code and it is composed of 4 bits representing the decimal digit (0-9). The name 8421 code indicates the binary weight of 4 bits that is $2^3, 2^2, 2^1, 2^0$

Decimal digit	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

Convert the following number into BCD.

① $(863)_{10} = (1000\ 0110\ 0011)_{BCD}$

8 — 1000
6 — 0110
3 — 0011

② $(723)_{10} = (0111\ 0010\ 0011)_{BCD}$

7 — 0111
2 — 0010
3 — 0011
∴ $(723)_{10} = (0111\ 0010\ 0011)_{BCD}$

③ $(7895.214)_{10} = (\quad)_{BCD}$

7 - 0111
8 - 1000
9 - 1001
5 - 0101
.
2 - 0010
1 - 0001
4 - 0100

$\therefore (7895.214) = (0111\ 1000\ 1001\ 0101 . 0010\ 0001\ 0100)_{BCD}$

Conversion from BCD to decimal

① $(1000\ 1001\ 0001\ 0010\ 0011 . 0111\ 0111)_{BCD} = (\quad)_{10}$

89123.77

$\therefore (10001001\ 00010010\ 00011 . 0111\ 0111)_{BCD} = (89123.77)_{10}$

EXCESS - 3 CODE

Decimal digit	BCD	+3	Excess - 3 code
0	0000	3	0011
1	0001	4	0100
2	0010	5	0101
3	0011	6	0110
4	0100	7	0111
5	0101	8	1000
6	0110	9	1001
7	0111	10	1010
8	1000	11	1011
9	1001	12	1100

The Excess-3 code is a bit code used with binary coded decimal number to convert any decimal number into its excess-3 form i.e to just add 3 to each decimal digit and the sum is converted into BCD number.

The Excess-3 code is an ~~unweighted~~ unweighted code since no definite weights can be assigned to 4 digit position.

The table above shows excess-3 code for each decimal digit

Ex:-

① $(176)_{10} = (\quad)_{EX-3}$

1 - 0100
7 - 1010
6 - 1001

$\therefore (176)_{10} = (0100\ 1010\ 1001)_{EX-3}$

② $(478)_{10} = (\quad)_{EX-3}$

4 - 0111
7 - 1010
8 - 1011

$\therefore (478)_{10} = (0111\ 1010\ 1011)_{EX-3}$

③ $(1599)_{10} = (\quad)_{EX-3}$

1 - 0100
5 - 1000
9 - 1100
9 - 1100

$\therefore (1599)_{10} = (0100\ 1000\ 1100\ 1100)_{EX-3}$

④ $(9213.917)_{10} = (\quad)_{EX-3}$

9 - 1100
2 - 0010
1 - 0100
3 - 0110

.

9 - 1100
1 - 0100
7 - 1010

$\therefore (9213.97)_{10} = 1100\,0101\,0100\,0110 \cdot 1100\,0100\,1010$

Gray Code

Conversion from gray code to binary

Problems on Conversion from binary to Gray code.

① Convert binary 1011 to gray code.

$B_3\ B_2\ B_1\ B_0$

$1\ 0\ 1\ 1 \longrightarrow 1110$

② $0010 \longrightarrow 0011$

③ $0101 \longrightarrow 0111$

④ $1110 \longrightarrow 1001$

⑤ $1111 \longrightarrow 1000$

⑥ $1000 \longrightarrow 1100$

Decimal digit	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
A	1010	1111
B	1011	1110
C	1100	1010
D	1101	1011
E	1110	1001
F	1111	1000

Explaination

Gray code is an unweighted code i.e, the bit positions are not assigned with any specific weights, The important property of Gray code is that, it exhibits only a single bit change from one code number to next. This property is useful many applications. The above table illustrates list of four bit Gray code for decimal numbers 0-15. The Gray cannot be used in arthematic circuits. It is used in instrumentation system.

Conversion from gray code to Binary

* $B_3 = G_3$
* $B_2 = B_2 \oplus G_2$
* $B_1 = B_2 \oplus G_1$
* $B_0 = B_1 \oplus G_0$

(1) 1110 → 1011

(2) 1100 → 1000

Convert the following into binary code

(3) 1111 → 1010

ASCII Code

[American standard code for information interchange]

Alpha numerical codes:

Alpha numeric codes are the combination of alphabets, numbers and special symbols. It represents all the values characters and function that are found on a computer keyboard. The most common Alpha numeric codes used for representing the data are as follows

1) ASCII
2) EBCDIC

ASCII Code

[American standard code for information interchange]

The ASCII is the most widely used alpha numeric code. It is unweighted 7 bit code. It allows manufacture to standardized computer hardware such as keyboards, printers and video displays.

The features of ASCII code are given by

* Each ASCII code consists of 7 bits, hence total in $2^7 = 128$ letters to represent alphabets, numbers and special symbols.
* There is definite order in which the symbols are assigned to each code word as ASCII 7 bit code, but can be considered as a 8 bit with MSD equal to 0
* Extra bits called redundant bits are added with ASCII code for error detection and correction.
* The computer keyboards are ~~standerized~~ standardized with ASCII code i.e, when users press a key on the keyboard, the corresponding 7 bit ASCII code is generated by a computer.

* ~~The ASCII code is generated~~
* The ASCII consists of non-graphic commands that are never printed and used only for control purpose.

EBCDIC

(Extended Binary coded decimal Interchange code)

EBCDIC is an 8 bit alpha numeric code, in which decimal digits are represented by the 8421 code presented by 1111. Both lower case and upper case letters are represented in addition to numbers other symbols and commands.

Each code word consists of 8 bits. Hence we have a total of 2^8 (256) possible combinations to represent the letters in the alphabet, punctuation marks and numbers. The code is divided into two groups. Each group contain 4 bits.

One's complement and two's complement of binary number.

In the binary number system the negative numbers can be represented with the help of one's complement and two's complement. The method of two's complement is used in the computer to handle negative numbers.

Unsigned binary numbers :-

For some application all data is either positive or negative, in such cases positive and negative sign can be ignored and the ~~mag~~ magnitude of number can only be represented.

Ex Smallest 8 bit no. is (0,0,0,0,0,0,0,0) largest (1,1,1,1,1,1,1,1)
This is equivalent to decimal 0-255 and the data of this type is called unsigned binary number because all the bits are used to represent magnitude (value)

Signed binary numbers

Signed binary number or signed magnitude number is when data as both positive and negative values, the binary number is signed. The left most bit is the signed bit and remaining bits are magnitude bits.

Ex :- 10001001

Signed bits — magnitude bits

As shown above the left most bit is signed bit and the remaining bits are magnitude bits.

MSB = 0 → +ve
MSD = 1 → -ve

One's Complement and Two's Complement

* One's Complement :

In One's complement form the numbers are obtained by inverting each bit i.e, complement of each digit. It is used usually for representing negative numbers. The following ex illustrate the conversion of binary to One's Complement.

Ex :- 110111 → binary number
001000 → one's complement

In One's complement the positive numbers are represented the same way as the positive signed magnitude numbers. Negative numbers are the One's complement of the corresponding positive number.

Ex :- +7 → 0111
-7 → 1000

* Two's Complement

The Two's complement is used to represent in negative numbers. The two's complement is obtained in two steps.

Step 1:- Take one's complement of a binary number.
Step 2:- Add 1 to One's complement.

Ex:- $(10101101)_2$

the complement of 10101101 is

$$\begin{array}{r} 10101101 \\ 01010010 \\ + \quad\quad 1 \\ \hline 01010011 \end{array}$$

Problems

① Find Two's complement for following number.

i) 6 ii) 14 iii) 25 iv) 36

i) 6

$(0110)_2$

$$\begin{array}{r} 1001 \\ + \quad 1 \\ \hline 1010 \end{array} \quad \begin{array}{l} \rightarrow \text{one's complement} \\ \\ \rightarrow \text{Two's complement} \end{array}$$

$\therefore 6 = 1010$

ii) 14

(1110)

$$\begin{array}{r} 0001 \\ + \quad 1 \\ \hline 0010 \end{array} \quad \begin{array}{l} \rightarrow \text{One's complement} \\ \\ \rightarrow \text{Two's complement} \end{array}$$

$\therefore 14 = 0010$

iii) 25

$(\overset{16}{1}\ \overset{8}{1}\ \overset{4}{0}\ \overset{2}{0}\ \overset{1}{1})$

$$\begin{array}{r} 0011\ 0 \\ + \quad\quad 1 \\ \hline 0011\ 1 \end{array} \quad \begin{array}{l} \rightarrow \text{One's complement} \\ \\ \rightarrow \text{Two's Complement} \end{array}$$

iv) 36

(1 0 0 1 0 0)

0 1 1 0 1 1 → One's complement

\+ 1

0 1 1 1 0 0 → Two's complement

∴ 36 = 011100

case-1

Addition of numbers using one's complement

i) +6 → 00110

+2 → 00010

+8 01000

ii) +12 → 0 1 1 0 0

+ 8 → 0 1 0 0 0

20 1 0 1 0 0

Case 2 :- positive number and small negative number

i) +6 → 0 0 1 1 0

−2 → 0 1 1 0 1

0 1 0 0 1 1

1

0 1 0 0

carry should be added to the result

ii) 8 → 0 1 0 0 0

−3 → 0 1 1 0 0

0 1 0 0

1

0 1 0 1

Case - 3

Case-3

Positive number and large negative number

i) $-6 \rightarrow 11001$
$+2 \rightarrow 00010$
$-4 \qquad 11011 \rightarrow$ Take complement
10100

ii) $-7 \rightarrow 11000$
$3 \rightarrow 00011$
$-4 \qquad 11011 \rightarrow$ complement
10100

Case-4 (Two negative number)

i) $-6 \rightarrow 11001$
$-2 \rightarrow 11101$
$-8 \qquad 10110$
1 - carry
10111 - complement
11000

ii) $-7 \rightarrow 11000$
$-8 \rightarrow 10111$
$-15 \qquad 11111$

Case 5 (equal and opposite number)

i) $-6 \rightarrow 11001$
$+6 \rightarrow 00110$
$0 \qquad 11111$ - complement
00000

ii) $+8 \rightarrow 0\ 1000$
$-8 \rightarrow 1\ 0111$
$0 \qquad 1111 \rightarrow$ Complement
0000

iii) 1010 → 0101
0100 → 1011

```
  0000
     1   → carry
  0001   - complement
 -1110
```

Find two's complement of binary no.

i) 10101 → 01010
1
01011

ii) 11011011 → 00100100
1
00100101

1
4- nibble
8- Byte
16 word
32- Double word
64- Quad.

Questions

1 Write a note on following
BCD, EBCDIC, ASCII, Excess 3 code, Gray code

2 Explain the following
Octal no. system, Hexa-decimal system

3 Explain the following
Binary system, Decimal system

4 Convert the following
$(101101)_2 = (\ )_{10}$, $(1101.110)_2 = (\ )_{10}$ $(A85)_{16} = (\ )_{10}$
$(232)_{10} = (\ )_2$

5 Convert the following
$(776)_8 = (\ )_{10}$, $(10111)_2 = (\ )_8$, $(5274)_8 = (\ )_2$,
$(B3.A2)_{16} = (\ )_2$

6. Perform addition for the following using two complement

BOALIAN ALGEBRA AND LOGIC GATES

Logic gates

Logic gates is a circuit with one or more input but only one output. The logic gates are the fundamental building blocks of digital system. Logic is the term applied in digital circuits for implementing logic function difines two possible logics i.e,

Logic 0 → Low → 0V
Logic 1 → high → 5V

Types of Logic circuits or logic gates

All the logic gates are grouped into three categories and it includes 7 logic gates.

They are :-

1) NOT gate
2) OR gate
3) AND gate

Basic gates

4) NAND gate
5) NOR gate

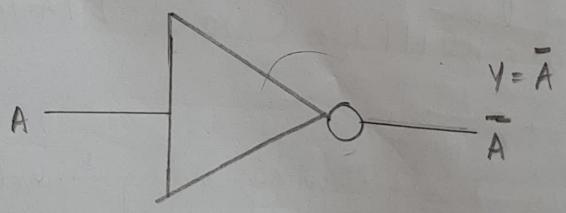
Universal gates

6) XOR gate
7) EXNOR gate

Exclusive gates

1) NOT Gate

a) Logic symbol



b) Truth table

A	Y
0	1
1	0

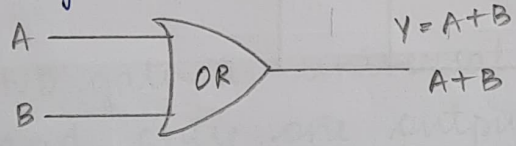
c) Explaination

The NOT gate or inverter produces an output that is the complement of the input which changes one logic level to opposite level i.e, it produce the 1 when input is 0 and produce 0 when input is 1.

The NOT gate is the only logic gate with only 1 input and only 1 output. Figure A shows the 1 input and only 1 output. Figure B shows the truth table and figure C shows logic expression.

2) OR Gate

a) Logic symbol



b) Truth table

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

c) Logic expression

$$Y = A + B$$

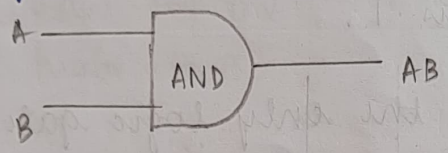
d) Explaination

The OR gate is one of the basic gates with two or more input and only one output. It produces a high output when any one or all the inputs

are high. It produces low output when both the inputs are low. It is also called as additional gate which corresponds to the action of parallel switches for the input. Figure A shows two input OR gate. Figure B shows the truth table and figure C shows the logic expression.

3) AND Gate

a) Logic symbol



b) Truth table

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

c) Logic expression

$$Y = A \cdot B$$

Explaination

The AND gate is one of the basic gates with two or more inputs and only one output. It produces high output when it inputs are high. It produces low output when any 1 of the input is low.

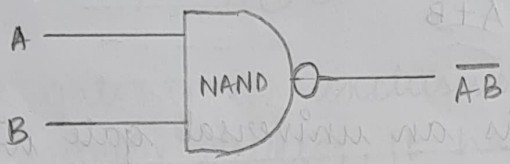
corresponds to the action of serial switches for the input.

Universal Gate

The NAND and NOR gate are called universal gate because using this two gates, we can construct any logic gate.

A) NAND gate

a) Logic symbol



b) Truth table

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

c) Logic expression

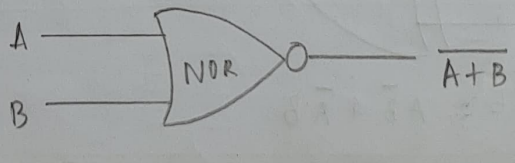
$$Y = \overline{A \cdot B}$$

The NAND gate is universal gate with 2 or more input and only one output. The NAND represents an AND gate followed by the inverted. It can be used to construct any logic gate.

The output of NAND gate is high when either of input is low and output is low when both the inputs are high

B) NOR gate

a) Logic symbol



b) Truth table

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

c) Logic expression

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

The NOR gate is an universal gate with 2 or more inputs and only one output. The NOR represents the OR gate followed by the inverter. It can be used to construct any logic gates. The output of NOR is high when both the inputs are low and output is low when either of the input is high i.e, the operation is opposite to the OR gate.

Exclusive gates

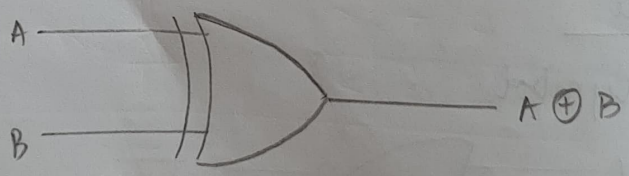
The Exclusive gates are formed by combination of other gates. There are two types of exclusive gates.

- Exclusive OR [EXOR gate]
- Exclusive NOR [EXNOR gate]

---

6) EXOR gate

a) Logic symbol



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

b) Truth table

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

c) Logic expression

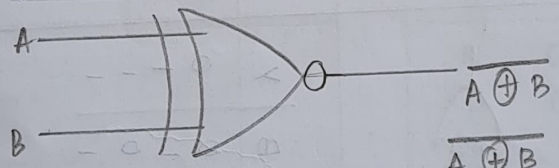
$$Y = A \oplus B$$

The EXOR gate is an exclusive gate. It is used as an inequality detector. It has two or more inputs and only one output. It produces high output when both inputs are different otherwise it produces low output.

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

7) EXNOR gate

a) Logic symbol



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

b) Truth table

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

c) Logic expression

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

The EXNOR gate is an exclusive gate and it is used as an equality detector. It has two or more inputs and only one output. It produces higher out when both the input are same. Otherwise it gives low output.

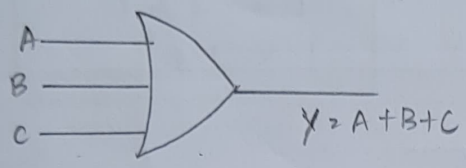
$$Y = \overline{A \oplus B} = \bar{A}\bar{B} + AB$$

Name	NOT	AND	NAND	OR	NOR	EXOR	EXNOR
Algebric Expression	$\bar{A}$	AB	$\overline{AB}$	$A+B$	$\overline{A+B}$	$A \oplus B$	$\overline{A \oplus B}$
Symbol							
Truth table	A | Y 0 | 1 1 | 0	A | B | Y 0 | 0 | 0 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1	A | B | Y 0 | 0 | 1 0 | 1 | 1 1 | 0 | 1 1 | 1 | 0	A | B | Y 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 1	A | B | Y 0 | 0 | 1 0 | 1 | 0 1 | 0 | 0 1 | 1 | 0	A | B | Y 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 0	A | B | Y 0 | 0 | 1 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1

3 input Logic gates

1) OR Gate

a) Logic symbol



b) Truth table

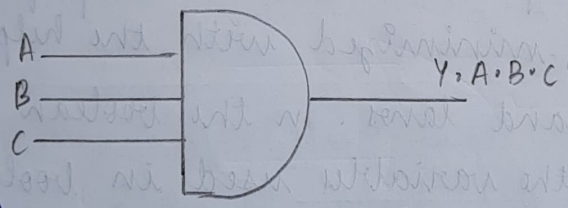
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

c) Boolean algebric expression

$Y = A + B + C$

2) AND gate

a) Logic symbol



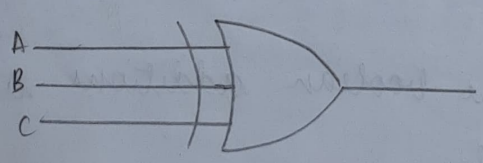
b) Truth table

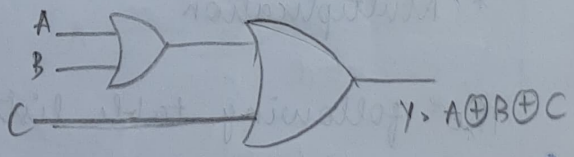
A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

c)

3) EXOR gate

a) Logic symbol





b) Truth table

A	B	C	X
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

c) Logic expression

$X = A \oplus B \oplus C$

Boolean Algebra

Boolean algebra is the mathematical tool used to describe the relationship between the logic circuit outputs and inputs. The operation of the digital circuits can be described an analysed using Boolean algebra. The logical expression can be simplified and/or reduced i.e, minimized with the help of boolean algebra rules and laws. In the boolean algebra terminology the variables used in boolean equation are termed as boolean variable and they assume only two possible states / values i.e Binary 0 or Binary 1

The fundamental Boolean operations are :-

* Inversion
* Addition
* Multiplication

The following table lists the boolean additions and boolean multiplication rules.

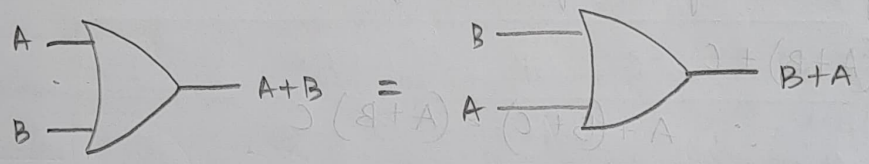
Boolean Addition	Boolean Multiplication
0+0 = 0	0·0 = 0
0+1 = 1	0·1 = 0
1+0 = 1	1·0 = 0
1+1 = 1	1·1 = 1

Laws of Boolean Algebra

There of the basic laws of Boolean Algebra are

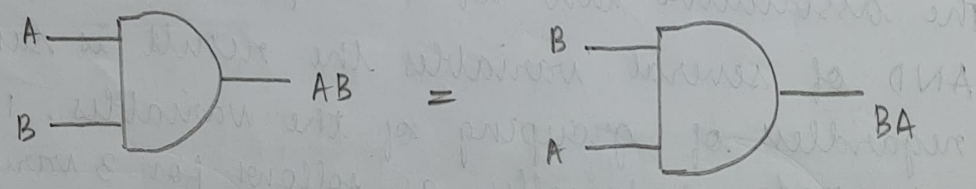
* Commutative laws
* Associative laws
* Distributive laws

Commutative laws of addition



The commutative laws of addition states that the order in which the variable are 0 Red makes no difference.
It is expressed algebraically as A+B = B+A.

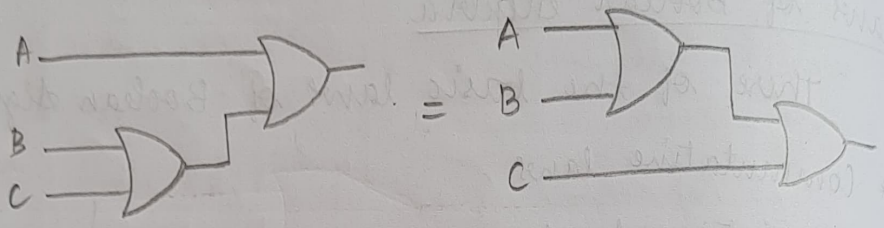
Commutative law of Multiplication



The commutative laws of multiplication states that the order which the variables are anded makes no difference. It is expressed algebrically as follows for two variable

$$A \cdot B = B \cdot A$$

Associative law of addition

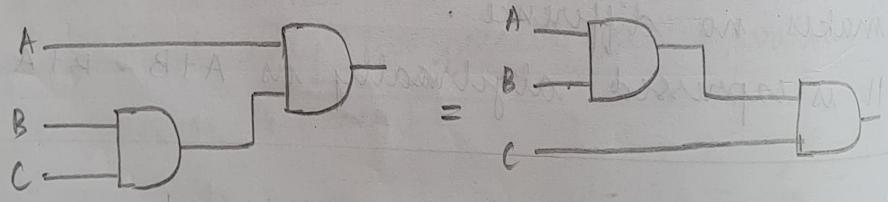


The associative law of addition states in OR of several variables, the result is same regardless of the grouping of the variables. It is expressed algebrically as follows for three variables $A+(B+C)$

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

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

Associative law of Multiplication

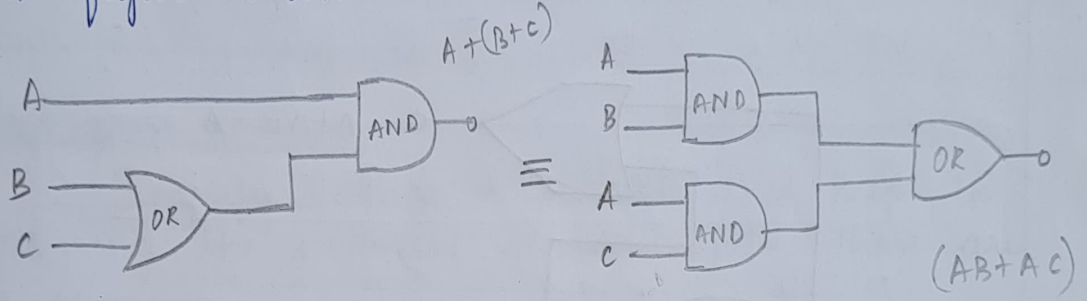


The associative law of multiplication states that AND of several variables the result is same regardless of grouping of the variables. It is expressed algebrically as follows for 3 variables

$$A(BC) = (AB)C.$$

Distributive law for the variable can be expressed as $A(B+C) = AB+AC$

This law stats that OR two or more variables and the AND the result with the single variable is equivalent to ANDing. The single variable with each of two or more ~~and~~ variable and then ORing the product the application of distributive law with OR and AND. Gates is illustrated in the figure below.



Rules of Boolean Algebra

The following table list of 12 basic rules of boolean algebra. That can be used to simplify boolean expression. Rule number 1 to 9 will be viewed in terms of their application to logic gates. where as

rules of boolean algebra. Here A, B and C can be represented as a single variable or combination of variables

Rule no.	Boolean expression
1	$A + 0 = A$
2	$A + 1 = 1$
3	$A \cdot 0 = 0$
4	$A \cdot 1 = A$
5	$A + A = A$
6	$A + \bar{A} = 1$
7	$A \cdot A = A$
8	$A \cdot \bar{A} = 0$
9	$\bar{\bar{A}} = A$
10	$A + AB = A$
11	$A + \bar{A}B = A + B$
12	$(A+B)(A+C) = A+BC$

Verification of Boolean Rules

Rule 10 :- $A + AB = A$

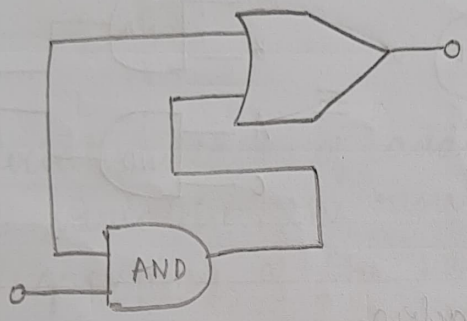
This rule can be verified and proved by applying distributive law boolean rule 2 and 4

$A + AB = A \cdot 1 + AB$ ∵ Rule 4

$= A(1 + B)$ ∵ Distributive law & rule 2

$= A \cdot 1$ ∵ Rule 4

$= A$



A	B	AB	A+AB	= A
0	0	0	0	0
0	1	0	0	0
1	0	0	1	1
1	1	1	1	1

B :- Rule 11 :- $A + \overline{A}B = A$

This rule can be proved as detailed below

$A + \overline{A}B = (A + AB) + \overline{A}B$ ∵ Rule 10 : $A + AB = A$

$= (AA + AB) + \overline{A}B$ ∵ Rule 7 : $A \cdot A = A$

$= AA + AB + \overline{A}A + \overline{A}B$ ∵ Rule 8 :- $A \cdot \overline{A} = 0$ (null)

$= \overline{A}(A + \overline{A}) + B(A + \overline{A})$ ∵ By factoring

$= (A + \overline{A}) + (A + B)$ ∵ Rule 6 $A + \overline{A} = 1$

$= 1 \cdot (A + B)$ ∵ Rule 4 $A \cdot 1 = A$

$= (A + B)$ ∵ $A + \overline{A}B = A + B$

c) Rule 12 :- $(A+B)(A+C) = A+BC$

This rule can be proved as detailed below

$(A+B)(A+C) = A \cdot A + AC + AB + BC$ ∵ Distributive law

$= A + AC + AB + BC$ ∵ Rule 7 :- $A \cdot A = A$

$= A(1+C) + AB + BC$ ∵ Rule 2 :- $A+1=1$

$A \cdot 1 + AB + BC$

$= A(1+B) + BC$ ∵ Factoring (Distributive law)

$= A \cdot 1 + BC$ ∵ Factoring

$= A + BC$ ∵ Rule 2 = $A+1=1$

Rule 4 = $A \cdot 1 = A$

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

Demorgan's theorem

Simplify Boolean Expression and helps in reducing the numbers of logic gates. There are 2 theorems.

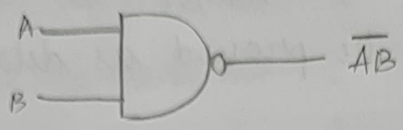
* Demorgan's first theorem (first law)
* Demorgan's second theorem (second law)

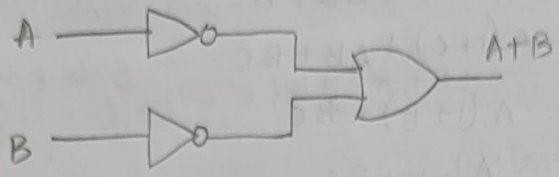
* Demorgan's first theorem

It states that "the complement of a product of variables is equal to the sum of complement of the individual variable logically it is given by.

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

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

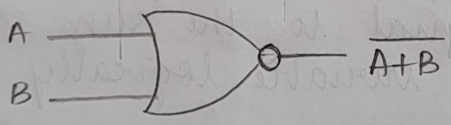


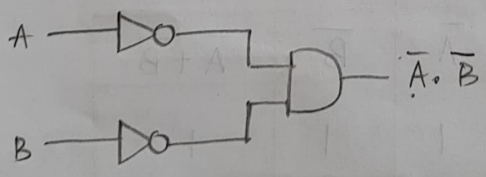


Demorgan's second theorem

It states that the complement of sum of variable is equal to the product of complements

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





The following two forms boolean expression that are commonly used in

Sum of Product (SOP)

Product of Sum form (POS)

Sum of product form (SOP)

When two or more product terms are summed logically by boolean addition the resulting expression is called Sum of Product form.

In other words SOP implementation consting OR ing the O/P of two or more AND gates or functions.

SOP example are :- $A\bar{B} + \bar{A}B$

$AB\bar{D} + AC + AD\bar{C}$

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

Product of Sum (POS)

When two or more sum terms are multiplied logically by Boolean Multiplication. Then the resulting expression is called Product of Sum form. In other words POS implemention consists of AND ing the O/P of 2 or more OR.

POS example are :- $(\bar{A}+\bar{D})(\bar{A}+\bar{B}+\bar{C})$

$D(A+C)$

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

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

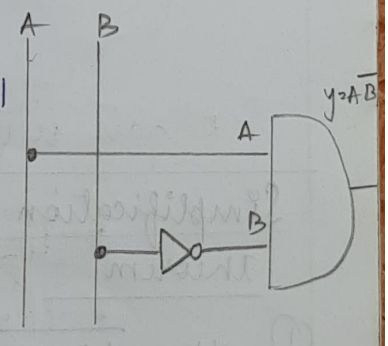
Simplification of logic expression

① $y = A\bar{B}D + A\bar{B}\bar{D}$

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

$A\bar{B}(1)$: Rule 6

$\therefore y = A\bar{B}$: Rule 4



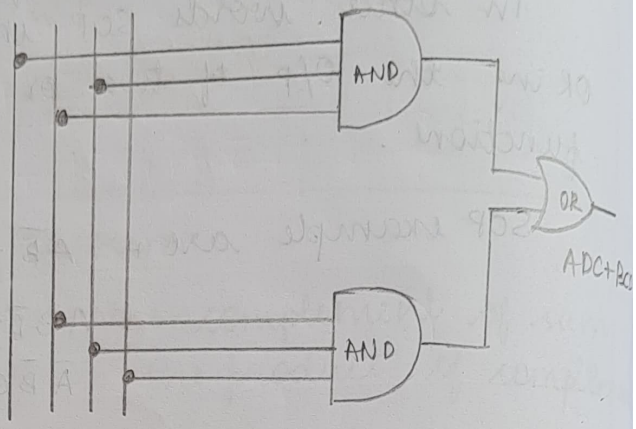
② $y = ACD + A\bar{B}CD$

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

$= CD + (A + A\bar{B})$: Rule 2 $A + \bar{A}B = A + B$

$= CD\,(A + B)$

$y = ADC + BCD$



Simplify the expression and write diagram

① $y = ABC + A\bar{B}C + AB\bar{C} + \bar{A}\bar{B}\bar{C}$

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

$= AC \cdot 1 + AB\bar{C} + \bar{A}\bar{B}\bar{C}$

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

Simplification of Boolean expression using Demorgan's Theorem

① $y = \overline{AB + C}$

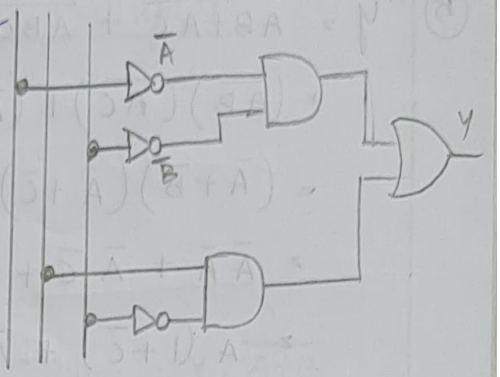
Applying Demorgans 2nd law

$y = (\overline{A}\overline{B})(\overline{C})$

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

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

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



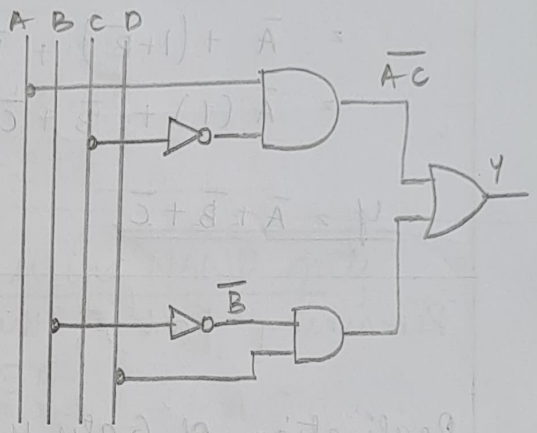
② $y = \overline{(\overline{A}+C)\cdot(B+\overline{D})}$

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

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

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

$y = A\,\overline{C} + \overline{B}\,D$



③ $y = AB + A(B+C) + B(B+C)$ ∵ $AA = A$

$= AB + AB + AC + BB + BC$ ∵ $B+B = B$

$= A(B+B) + AC + B(1+C)$ ∵ $1+C = 1$

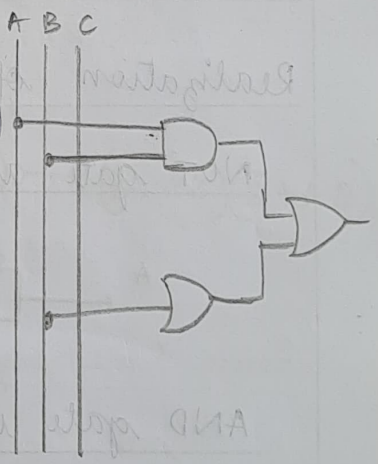
$= AB + AC + B(1)$

$= AB + B + AC$

$= B(A+1) + AC$ ∵ $1+A = 1$

$= B(1) + AC$

$y = AC + B$



④ $y = A\overline{B}\,\overline{C} + A\overline{B}C + ABC$

$= A\overline{B}(\overline{C}+C) + ABC$ ∵ $\overline{C}+C = 1$

$= A\overline{B}(1) + ABC$

$= A\overline{B} + ABC$

(5) $Y = \overline{AB+AC} + \overline{ABC} = 0$

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

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

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

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

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

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

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

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

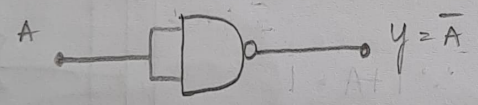
Remaining part of the middle position is continued here

Realization of Gates using universal gates only:

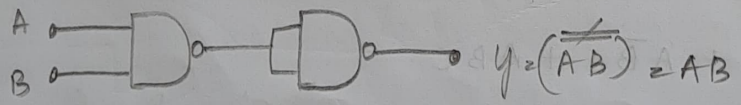
Realization of all gates using universal gates

Realization of all gates using NAND Gates

NOT gate using NAND gate

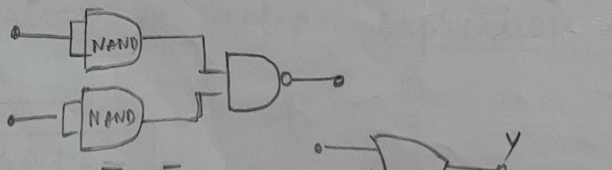


AND gate using NAND gate

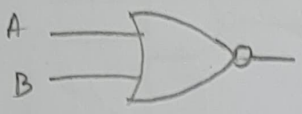


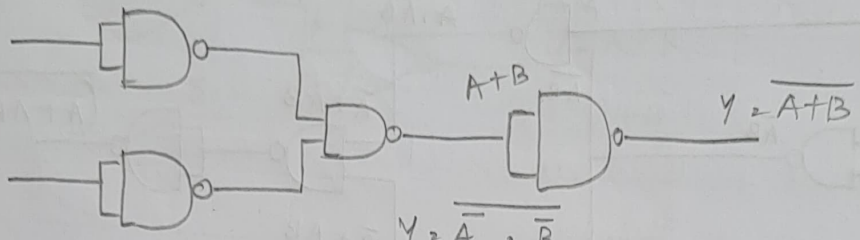
output $Y = AB$

OR gate using NAND gate



Realizing of NOR gate using NAND gate



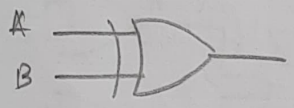


$Y = \overline{\bar{A} \cdot \bar{B}}$

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

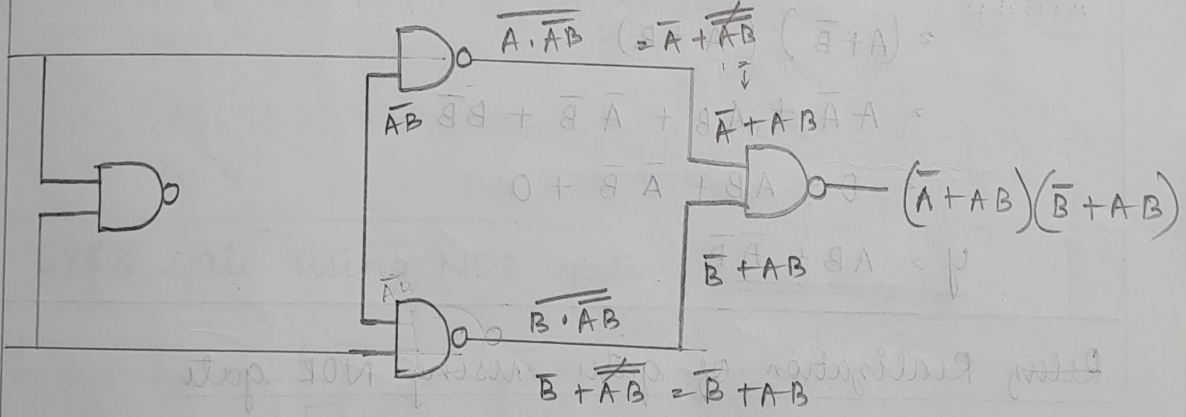
$Y = A + B$

Realization of EX-OR gate using NAND gate



$Y = A \oplus B$

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



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

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

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

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

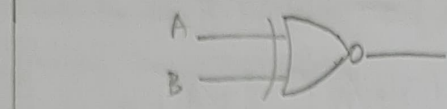
$$= A \cdot \bar{A} + A\bar{B} + B \cdot \bar{A} + B\bar{B}$$

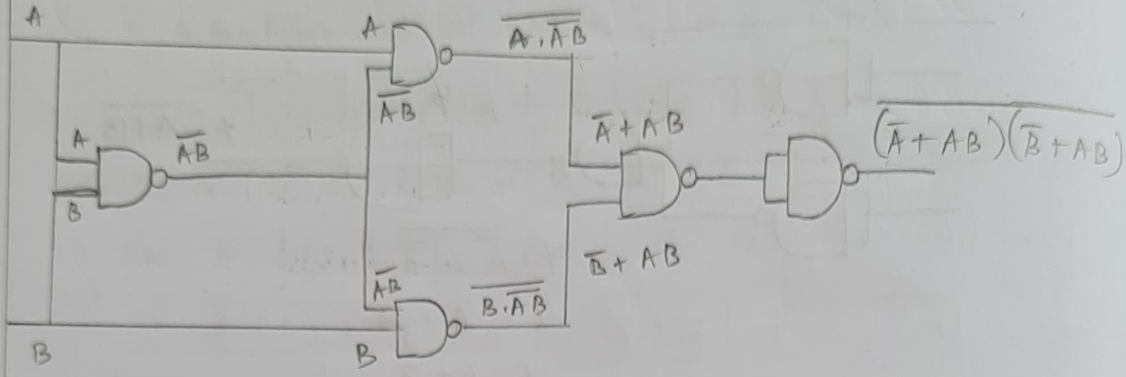
$$= 0 + A\bar{B} + B \cdot \bar{A} + 0$$

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

$$Y = A \oplus B$$

Realization of EX-NOR gate using NAND gate.





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

$$= (\bar{A}B)\,(\overline{A\bar{B}})$$

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

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

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

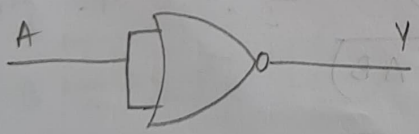
$$= A\bar{A} + AB + \bar{A}\bar{B} + B\bar{B}$$

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

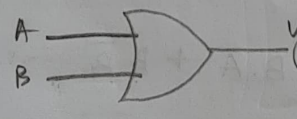
$$Y = AB + \bar{A}\bar{B}$$

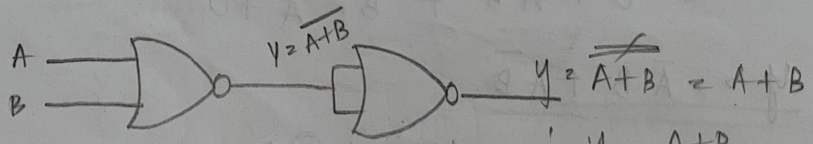
~~Rtay~~ Realization of gate using NOR gate

NOT gate using NOR gate



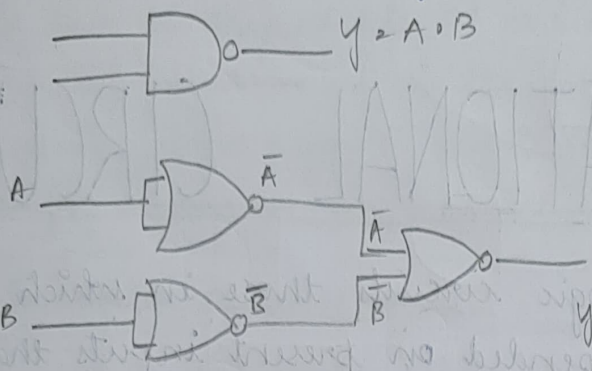
OR gate using NOR gate





$\therefore Y = A + B$

AND gate using NOR gate



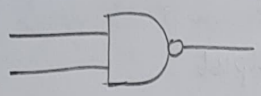
$y = A \cdot B$

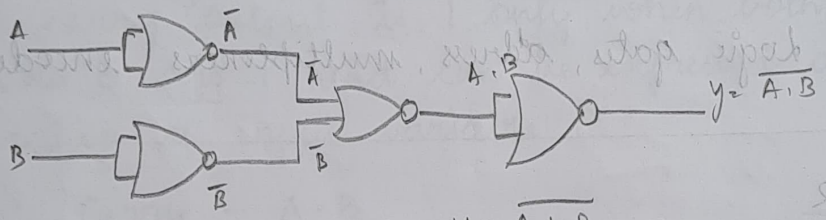
$y = \overline{\bar{A} + \bar{B}}$

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

$y = A \cdot B$

NAND gate using NOR gate



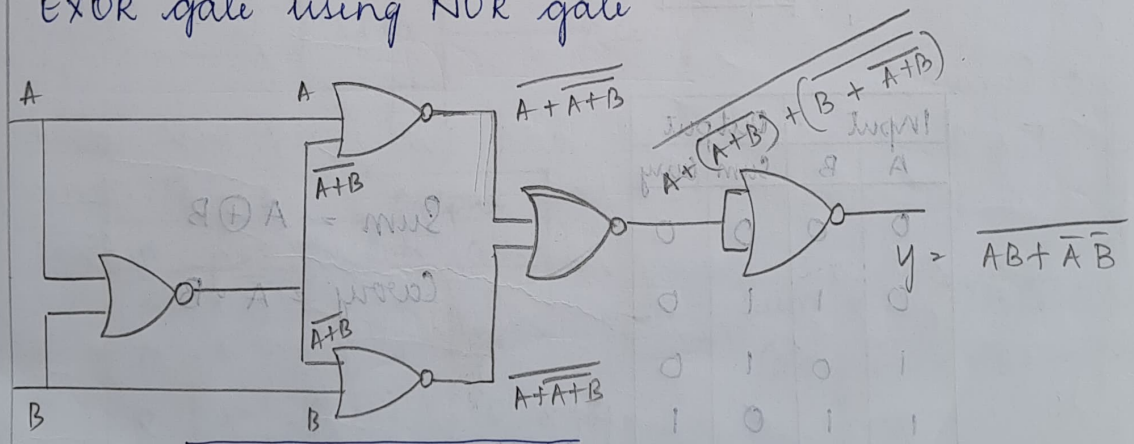


$y = \overline{A + B}$

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

$y = A \cdot B$

EXOR gate using NOR gate



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

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

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

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

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

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

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

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

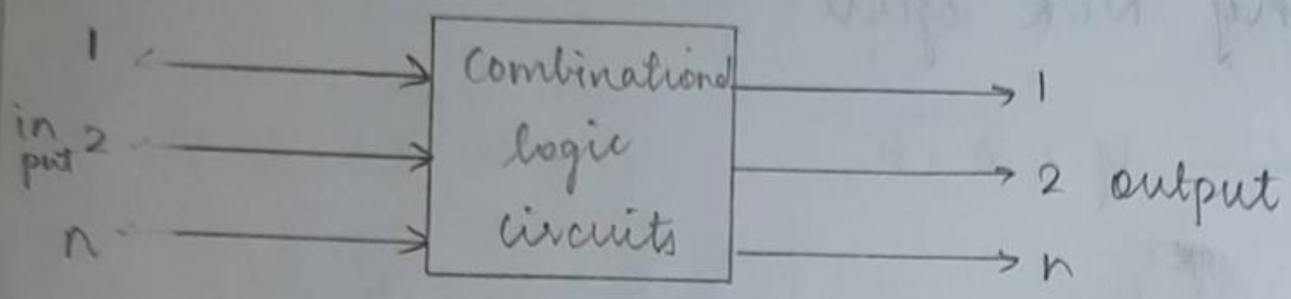
$\overline{A\bar{B}} + \overline{\bar{A}\bar{B}}$

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

UNIT-2

COMBINATIONAL CIRCUITS

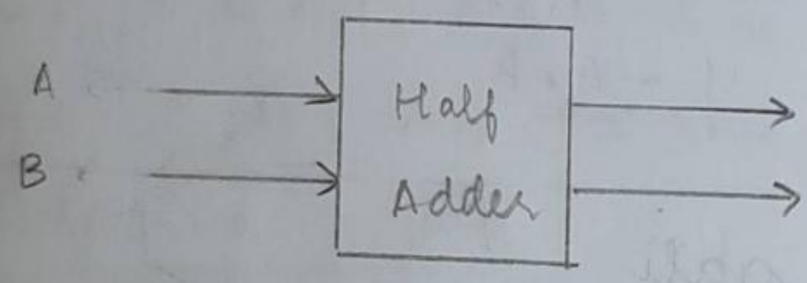
A combinational logic circuits those in which the output will be depended on present inputs that is it has no memory.



Ex:- Logic gates, address, multiplexers, encoders etc.

HALF ADDER

fig A



Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

fig B

$Sum = A \oplus B$

$Carry = A \cdot B$

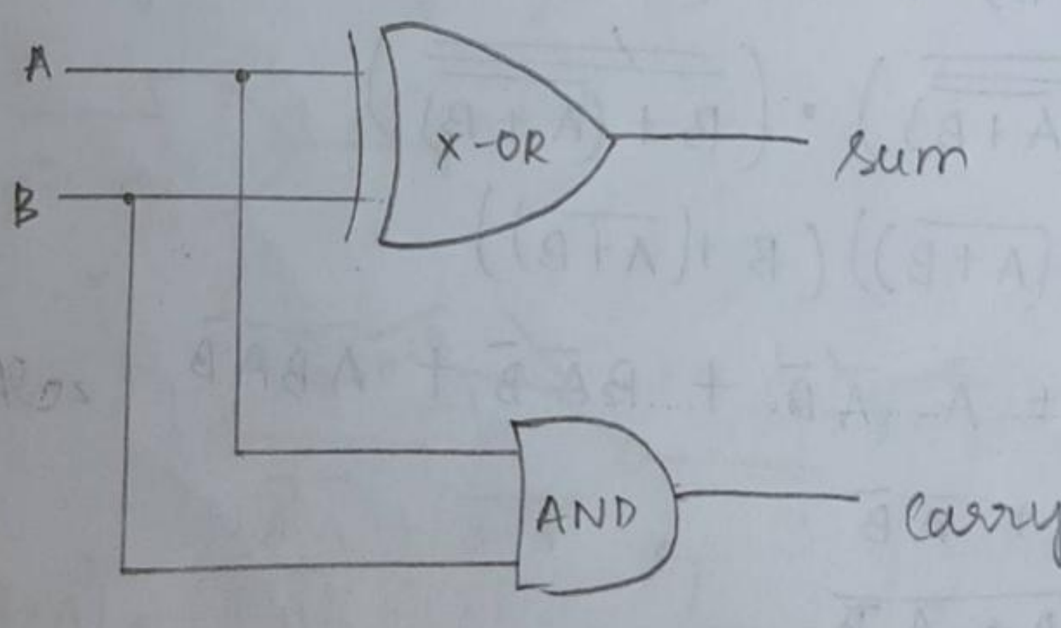


fig c

Half adder is a combinational logic circuit which adds two binary digits at a time and produces two output bits that is it is the sum and carry fig A shows the logic symbol, fig B shows truth table and fig C shows logic circuit.

Functionality of half adder.

* Refering to the truth table the sum output is high only if inputs are not equal. It can be expressed as the EXOR operation of input variable

$$\therefore \text{Sum}(S) = A \oplus B$$
$$= \bar{A}B + A\bar{B}$$

* The carry output is 1 only when both inputs A and B are 1 and can be expressed as AND operation of input variable

$$\therefore \text{Carry} = A \cdot B$$

FULL ADDER

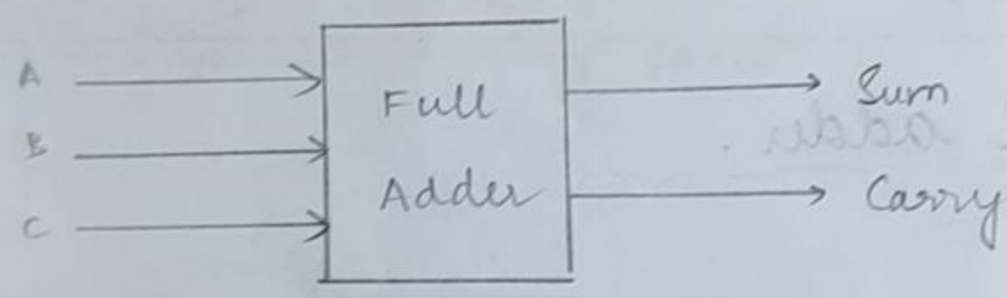
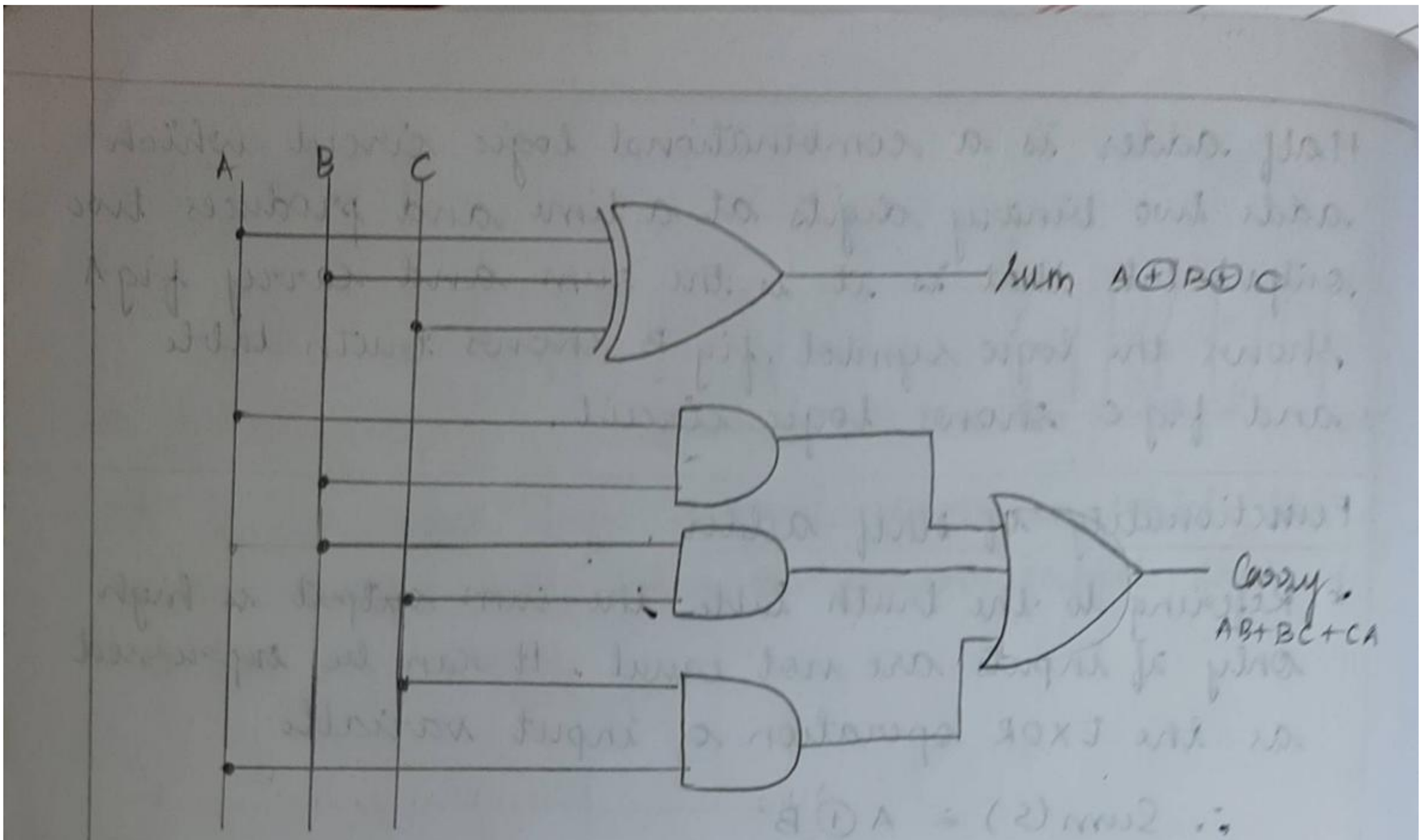


fig 1

Input			Output	
A	B	C	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$\text{Sum} = A \oplus B \oplus C_{in}$

$\text{Carry} = AB + BC_{in} + AC_{in}$



Full adder is a combinational logic circuits that ~~ca~~ access 3 bits at a time and generates 2 outputs [sum, carry]. The difference between half and full adder is the full adder accepts carry input [Cin]. fig A shows logic symbol, fig B shows ~~to~~ truth table, fig C logic circuit

---

Functionality of full adder.

* Sum:

The full adder must add two input bits and input carry. For this inputs A and B exclusive ~~of~~ ore necessary with A ⊕ B to get the equation of the sum given by,

A ⊕ B ⊕ Cin

Thus for the full adder sum function three input exclusive OR gate is necessary.

* Carry:

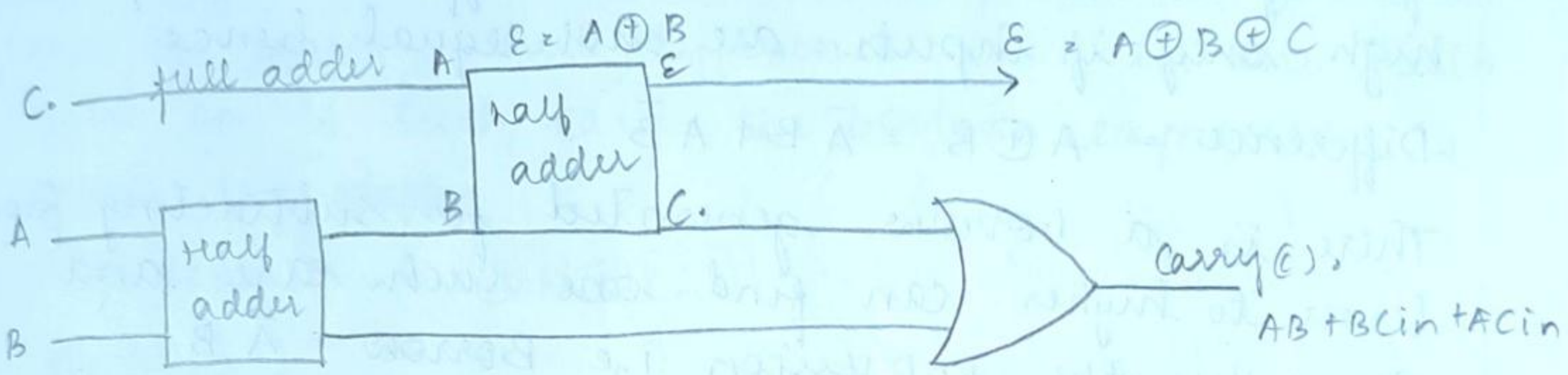
The Carry output is high when two or more inputs are in high condition as shown in the

truth table there are four such cases. Hence thus the simplified Boolean expression for carry from the truth table is given by

$$Carry = AB + BC_{in} + AC_{in}$$

To implement above expression 3 input AND gates are ORed together to obtain carry output

* The full adder can be implemented by using two half adder as shown below



HALF SUBRACTOR

fig (a)

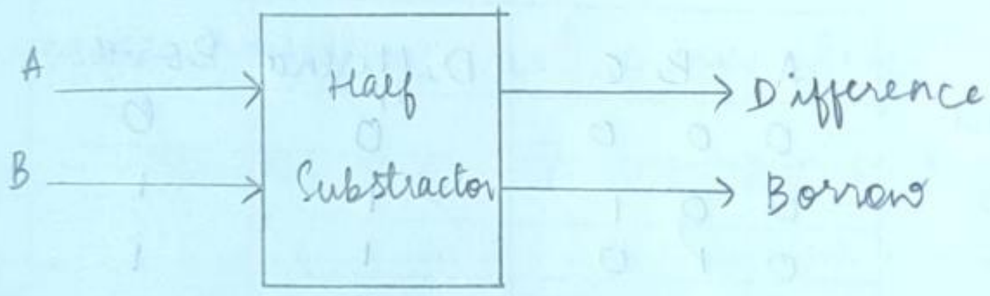


fig (b)

A	B	Difference	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

fig (c)

Difference = $A \oplus B$

Borrow = $\bar{A}B$

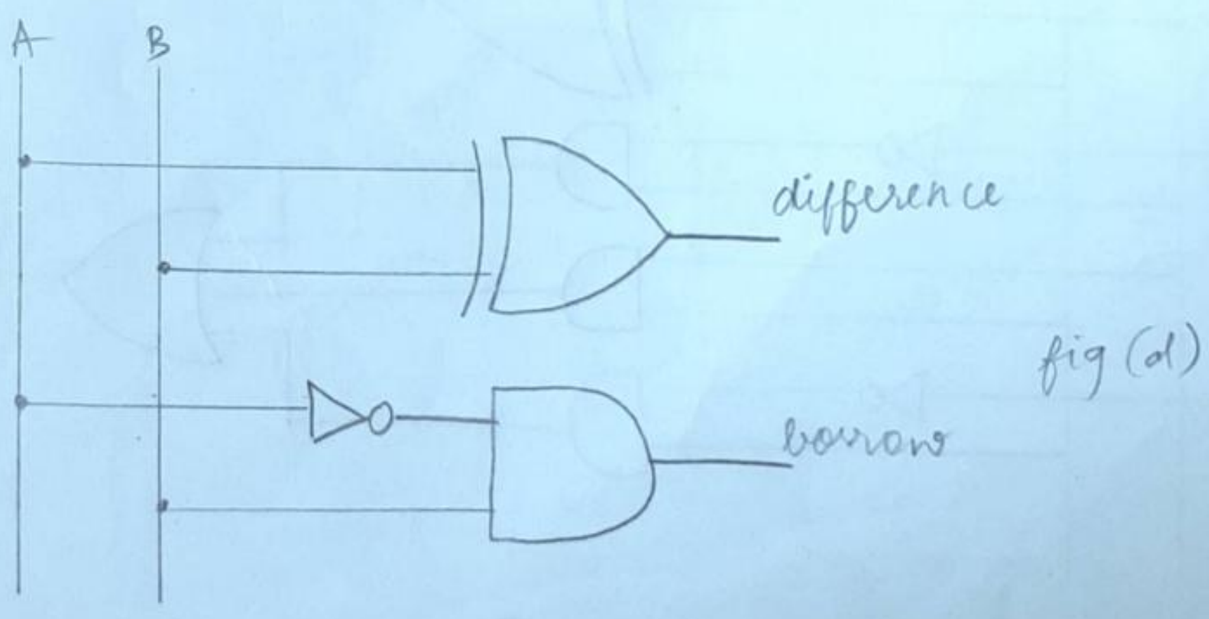


fig (d)

Half subtractor is a combinational logic circuits which subtracts one bit from another bit i.e, b-A and produces two outputs difference and borrow. The half subtractor can only be used for LSB subtraction figure(a) above shows the block diagram, fig(b) shows truth table, fig(c) logic expression and fig(d) logic diagram for half subtractor subtractor

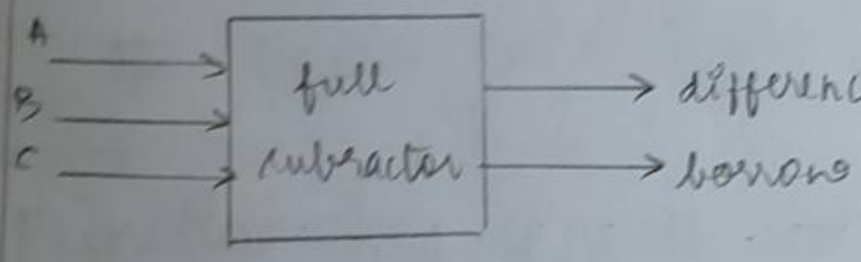
Functionality

Referring to the truth Table the difference output is high only if inputs are not equal, hence

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

There is a borrow generated for subtracting from lower to higher can find one such case and given by the expression i.e Borrow = $\bar{A}B$

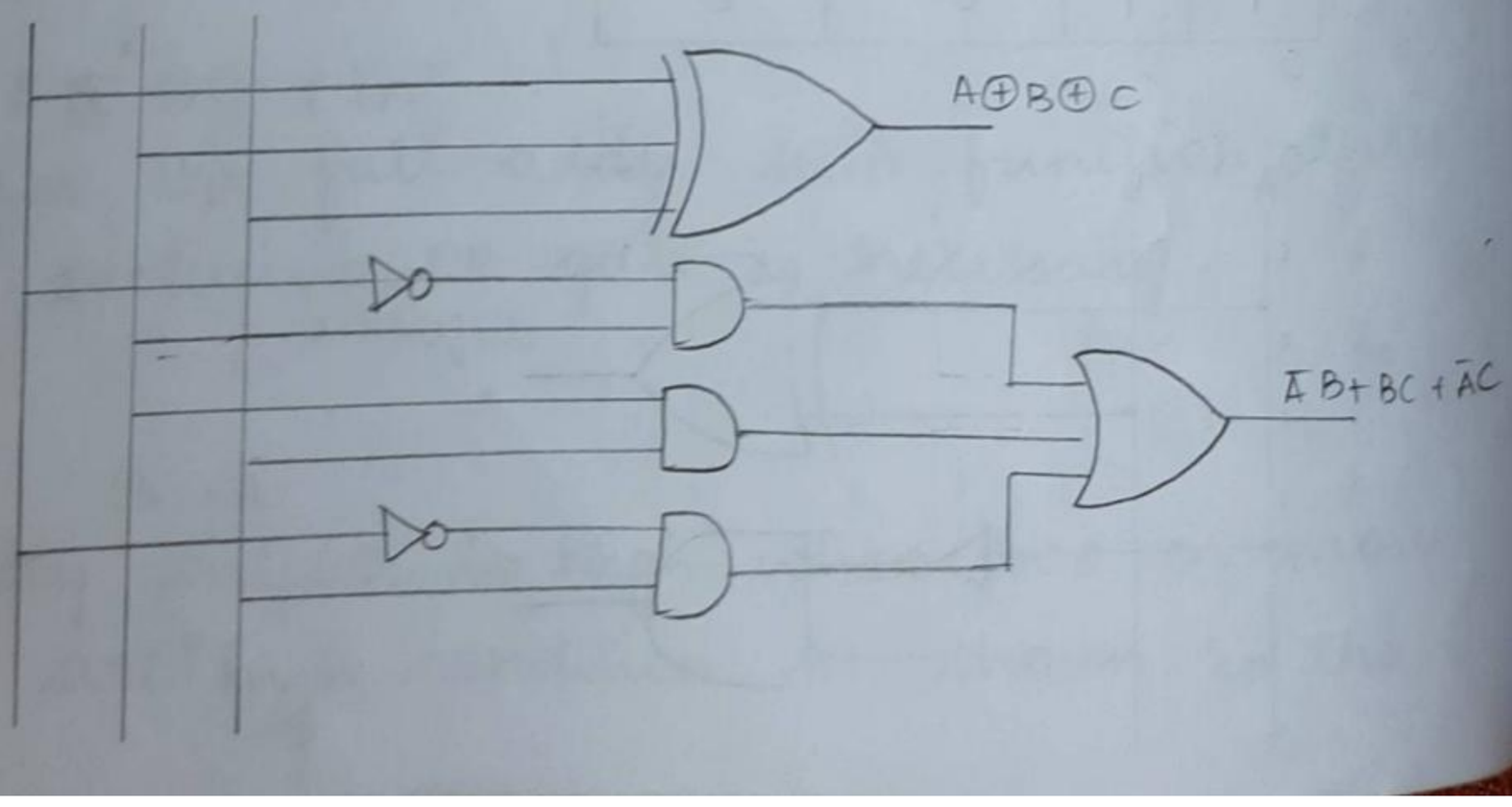
FULL SUBTRACTOR



A	B	C	Difference	Borrow
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Difference = $A \oplus B \oplus C$

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



Full subractor is a combinational logic circuit that accepts subracts 3 bits at a time and generating two outputs difference and borrow. The full subtractor takes care of borrow from the lower column and generates the bit required for next higher column. Fig (a) shows the block diagram, fig (b) shows truth table, fig (c) logic expression and fig (d) logic diagram for full subractor.

Functionality (Difference)

Referring to the truth table the productor for each case is ~~diffe~~ where difference is 1 in the truth table for 4 such case, the boolean expression is given by. ~~differ~~

Difference = $A \oplus B \oplus C$

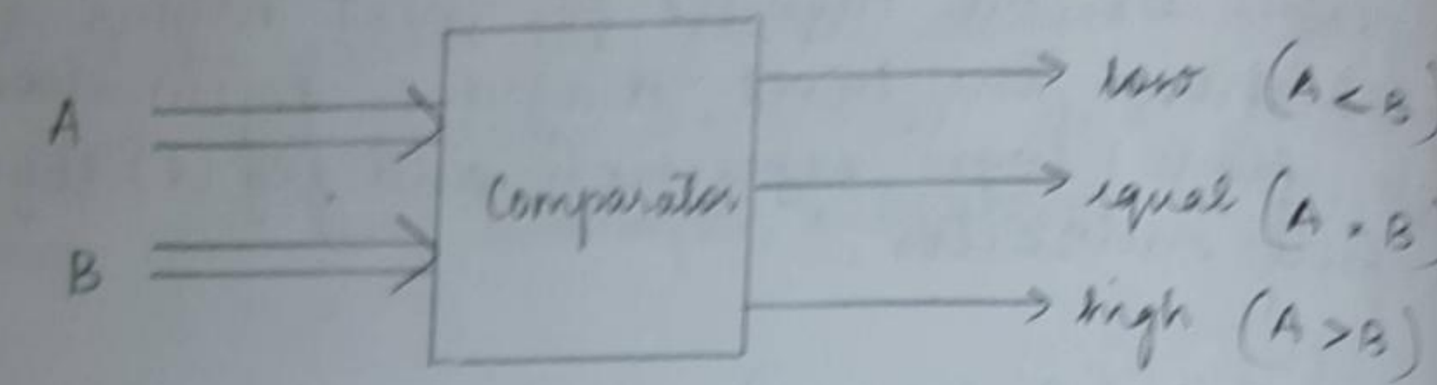
(Borrow)

Considering the productor for each case where Borrow is equal to 1 i,e. Subtracting from, lower to higher in 4 such cases, hence boolean expression for borrow is given by.

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

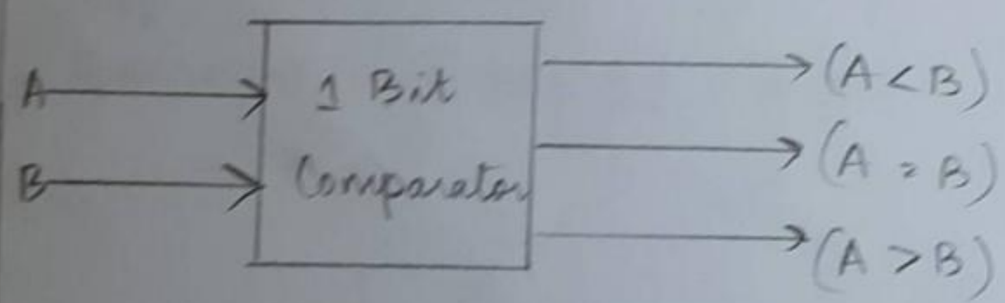
APPLICATION OF ADDER AND SUBTRACTORS

COMPARATORS



The comparator or magnitude comparator is a combinational logic circuit that compares the ~~magnitude~~ magnitude of two digital numbers and indicates whether the magnitude of 1 binary number is greater than, less than or equal to other. The fig above shows the block diagram of comparator which it accepts two binary inputs and produces three single bit output here $(A > B)$; equal $(A = B)$, high $(A < B)$. The wide arrow at the input represents a group of lines or bits.

1 - BIT COMPARATOR.

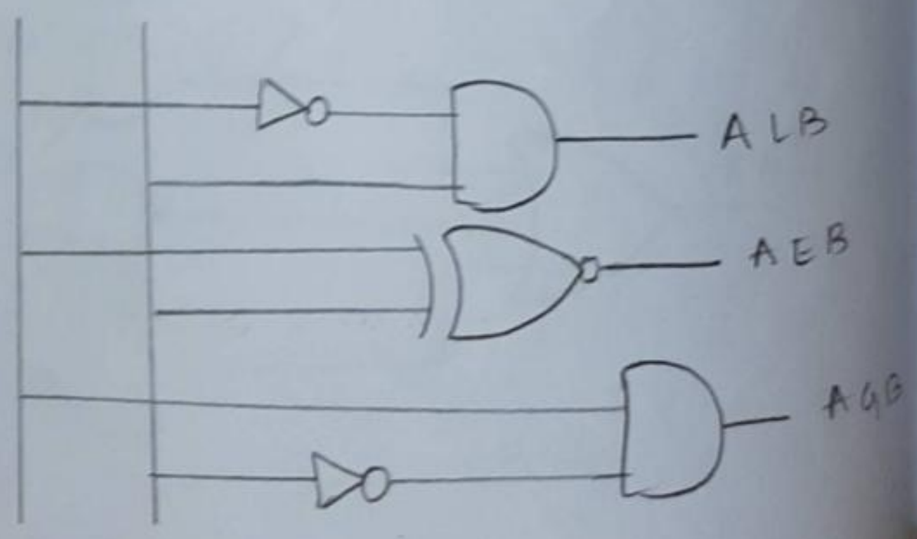


Inputs		Outputs		
A	B	ALB	AEB	AGB
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

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

$A E B = \overline{A \oplus B}$

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



1-Bit comparator is a combinational logical circuit which compares the magnitude of A and B and producing three outputs that is $(A<B), (A=B), (A>B)$ out of three outputs for an Inputs combination any one of the output is high and remaining two output are low. fig (a) shows block diagram, fig (b) shows truth table, fig (c) logic expression, fig (d) logic diagram.

FUNCTIONALITY.

(A L B)

The Productor for each case where A L B is equal to 1 in the truth table is for one such case hence

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

(A E B)

Considering the productor for each case where (A E B) is high in the truth table is for two such cases hence

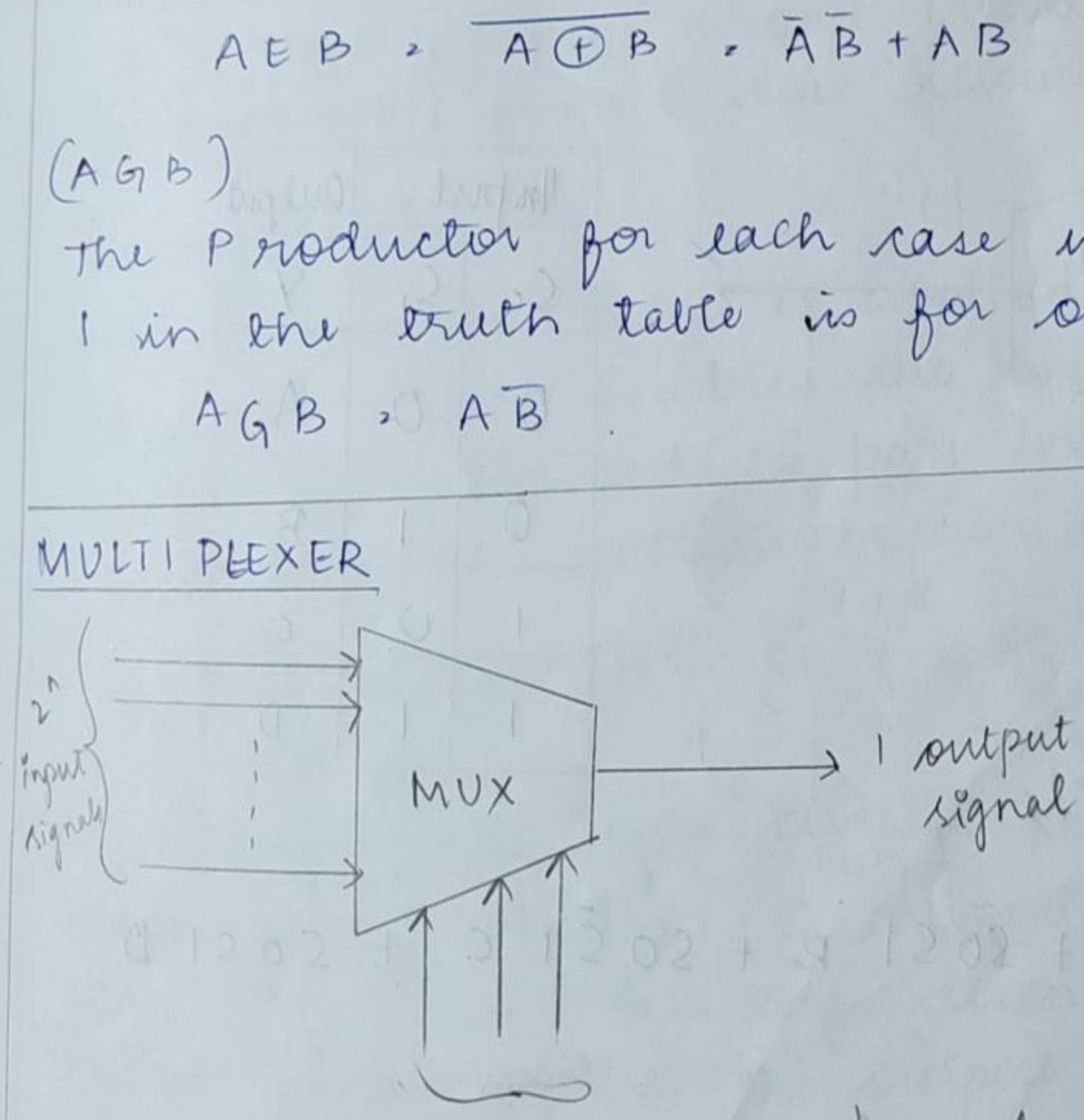
$$A E B = \overline{A \oplus B} = \bar{A}\bar{B} + AB$$

(A G B)

The Productor for each case where A G B is equal to 1 in the truth table is for one such case hence

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

MULTIPLEXER



A digital multiplexer or data selector is a combinational logic circuit with many inputs and only one output

It accepts several data inputs and selects only one of them at a given time and passes to its output. This is done and controlled by select input. It is also called as control input. The fig above shows the fundional block diagram of multiplexer. It has 2^n inputs, n control lines signals and only one output signal. In short we can refer multiplexer as "many to one".

APPLICATIONS OF MULTIPLEXER

* Data selection
* Data routing
* Parallel to serial conversion
* Way form generation
* Logic function generation

4 : 1 Multiplexer

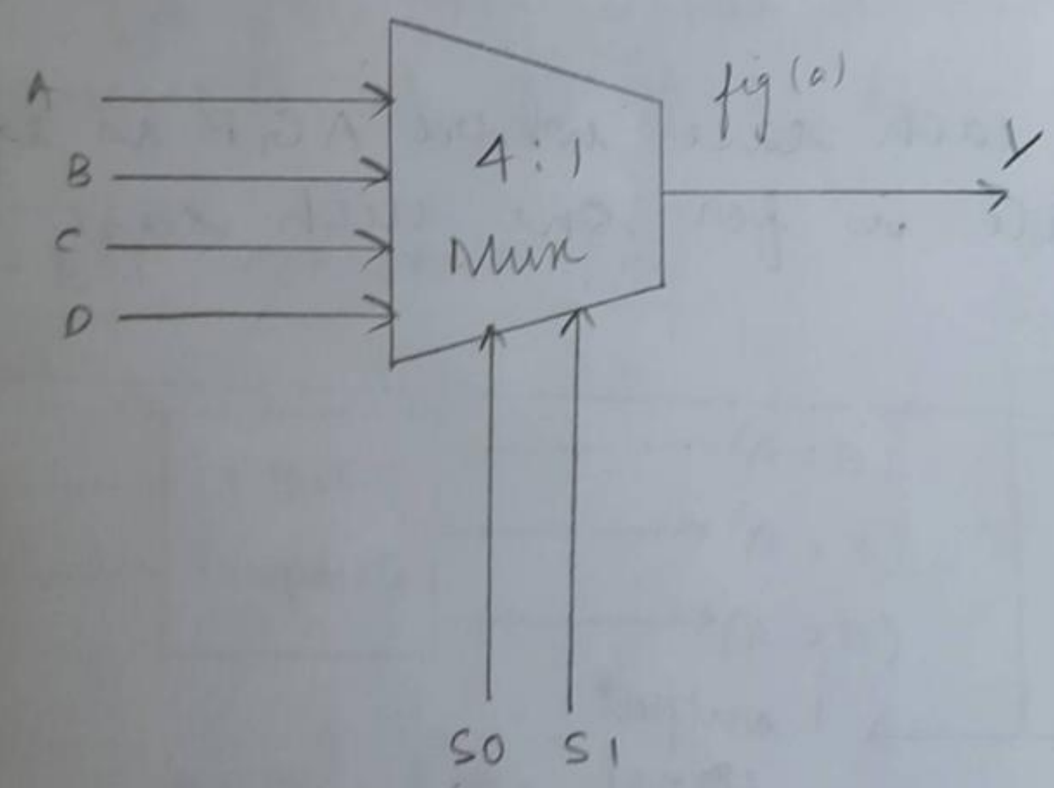


fig (a)

fig (b)

Input		Output
S_0	S_1	Y
0	0	A
0	1	B
1	0	C
1	1	D

fig (c)

$$Y = \bar{S_0}\bar{S_1}A + \bar{S_0}S_1B + S_0\bar{S_1}C + S_0S_1D$$

Logic diagram ⟶

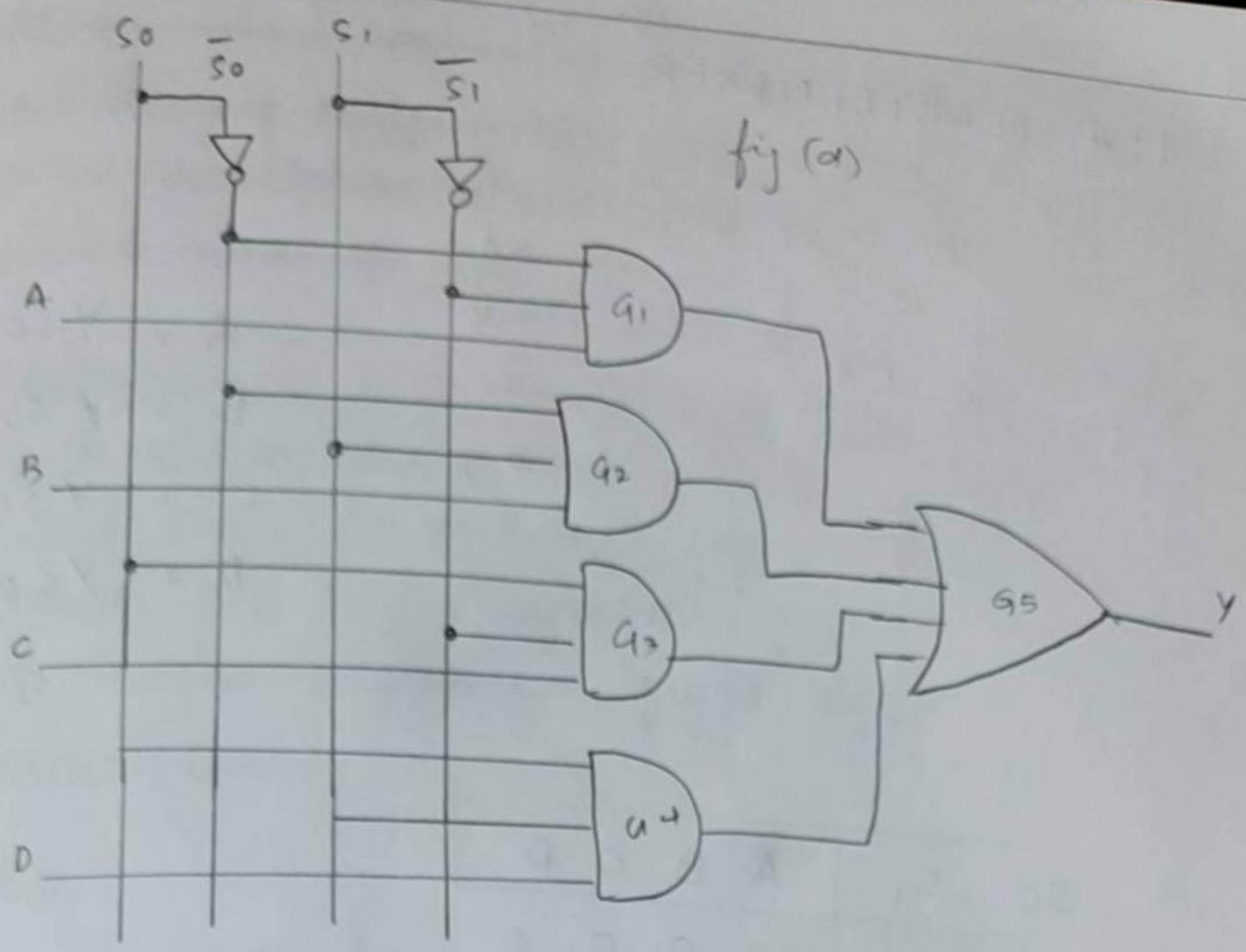


fig (a)

~~Figure~~

4:1 multiplexer is a combinational logic circuit which has four inputs A,B,C,D and only one output. Any one input is guided to output or selected by the two select inputs S0 and S1

Fig (a) shows ~~the~~ block diagram or logic symbol for 4:1 mux. fig (b) shows truth table, fig (c) logic expression, fig (d) logic circuit.

FUNCTIONALITY

When the logic level is applied to the select inputs S0 and S1 that enables the one of the four data inputs is allowed to pass through the output y. Hence the boolean expression for y is

$$Y = A\bar{S0}\bar{S1} + B\bar{S0}S1 + C S0\bar{S1} + D S0S1$$

Referring to the logic circuit

When S0 = 0 and S1 = 0 then Gate G1 is enabled and G2, G3, G4 are disabled. Data from input line A goes to the output line Y and inputs from lines B, C, D are block. The above process is same for remaining logic accordingly.

1 : 4 DEMULTIPLEXER

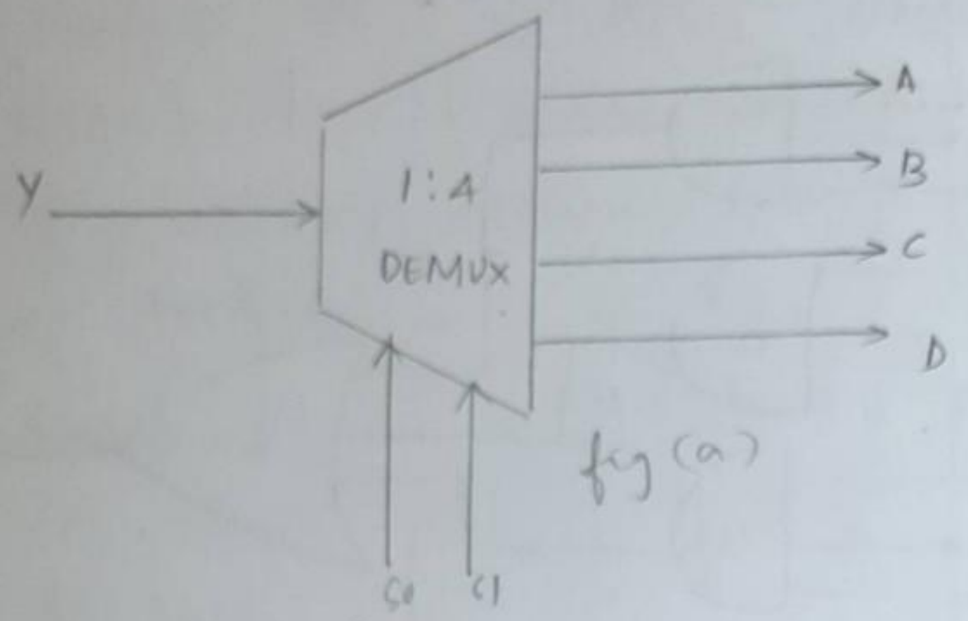


fig (a)

fig (b)

S0	S_1	A	B	C	D
0	0	Y	0	0	0
0	1	0	Y	0	0
1	0	0	0	Y	0
1	1	0	0	0	Y

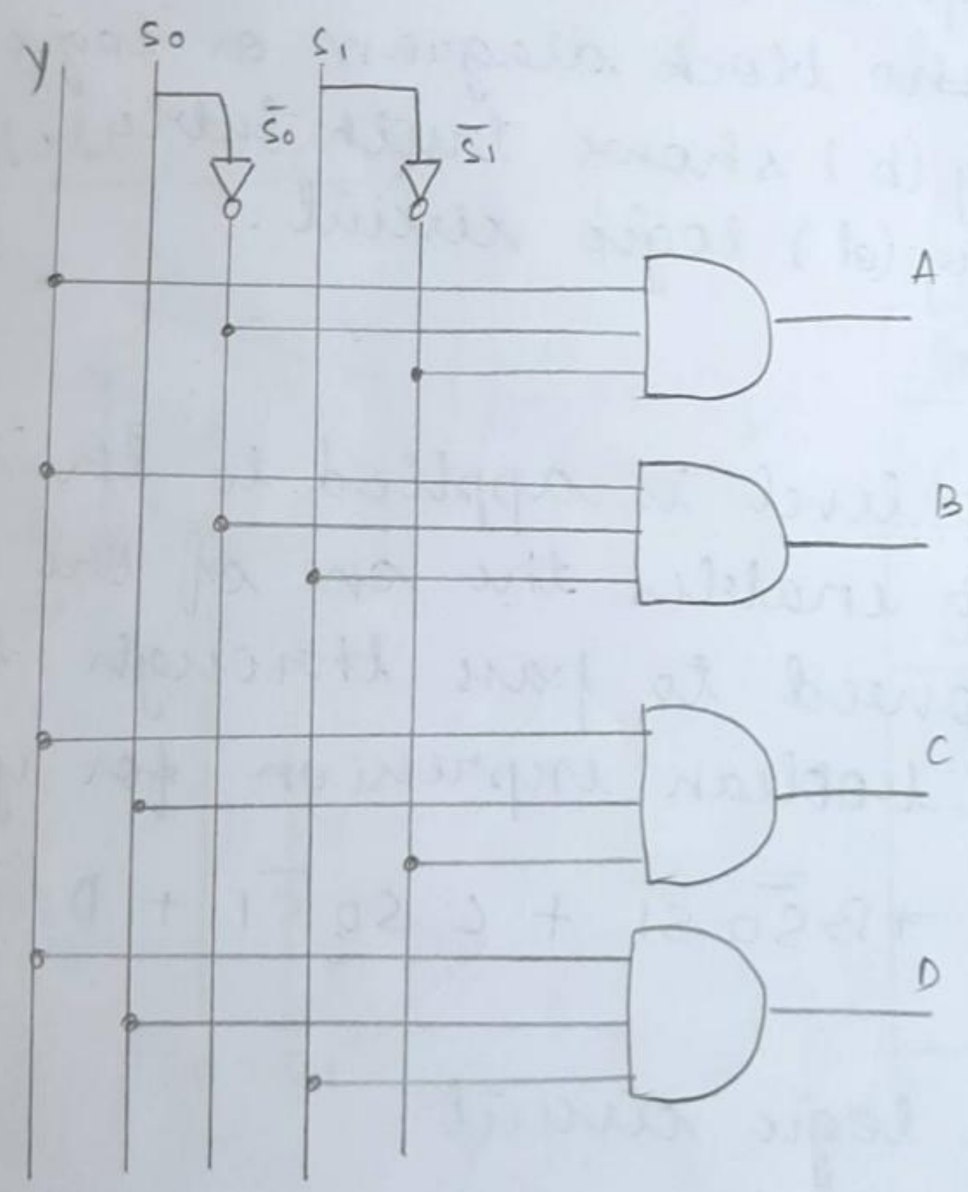
fig (c)

$A = Y \bar{S_0} \bar{S_1}$

$B = Y \bar{S_0} S_1$

$C = Y S_0 \bar{S_1}$

$D = Y S_0 S_1$



~~De Multiplexer is combinational logic circuit with one input and many outputs it performs the reverse operation of multiplexer. The one input is selectively distributed to one of the~~

1:4 De Multiplexer is a digital logic circuit with one input and many outputs. It consists of input bit Y, two select lines S0, S1 and four output lines A,B,C,D Fig (a) shows the logic symbol, fig (b) shows truth table, fig (c) logic expression. fig (d) logic circuit for 1:4 De Multiplexer.

Functionality

The two select lines S0 and S1 enables only one gate at a time and input data 'Y' will pass through the enabled gate to the output line.

when S1 = 0, S0 = 0 the upper AND gate is enabled and remaining AND gates are disabled. Hence the data bit 'Y' is transmitted only to the output line 'A'.

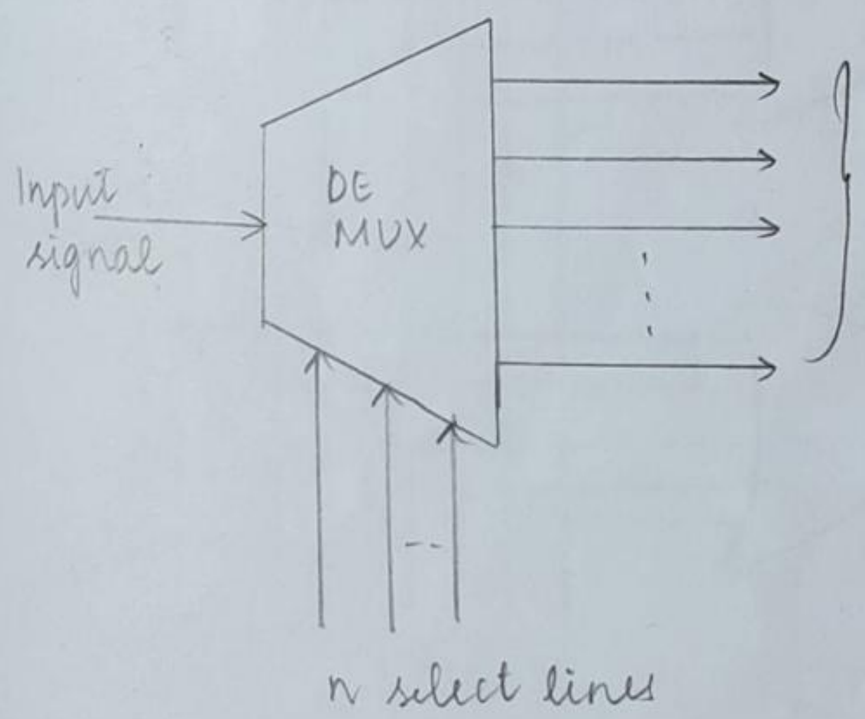
i.e If $y = 0 \Rightarrow A = 0$

$y = 1 \Rightarrow A = 1$

The outputs for remaining line will be obtained in the same manner

---

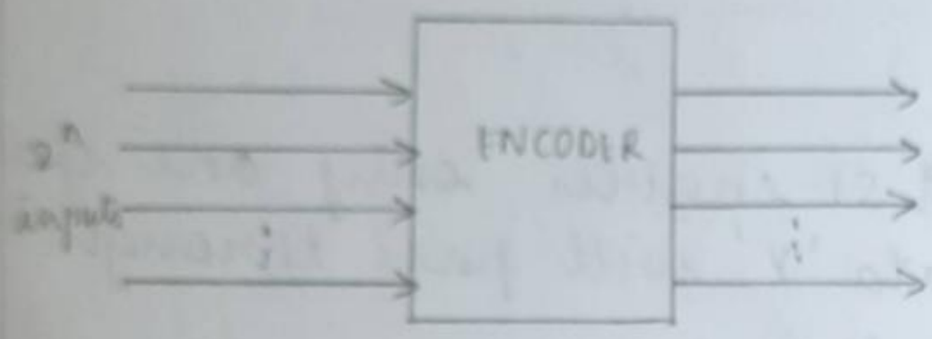
DE MULTIPLEXER.



DE multiplexer is a combinational logic circuit with one input and many outputs it performs the reverse operation of multiplexer. The ~~one~~ input is selectively distributed to one of the 2^n outputs lines like a switch

The above diagram shows the block diagram of demultiplexer which has one data input signal, n select lines and 2^n output lines.

ENCODERS



Encoder is a combinational logic circuit which converts the data or number from one form to another form, which is also called at translator. An encoder has 2^n of input lines, n output lines. ~~out of which any one line~~
The encoder produces A an bit code depending on which input is activated i.e, in encoder only one input line is activated at a given time. The figure above shows the logic symbol of encoder.

The applications of encoder

* Data privacy and security
* Data communication
* Data compression
* Data scanning

Types of encoder

1 Decimal to BCD encoder

2 Octal to Binary encoder

3 Decimal to binary encoders
4 Decimal to Octal encoders
5 Priority encoders etc

1) Decimal to BCD encoder.

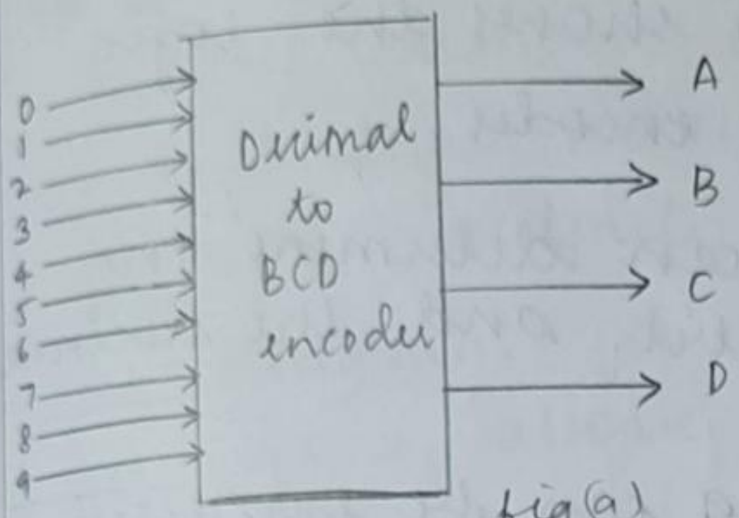
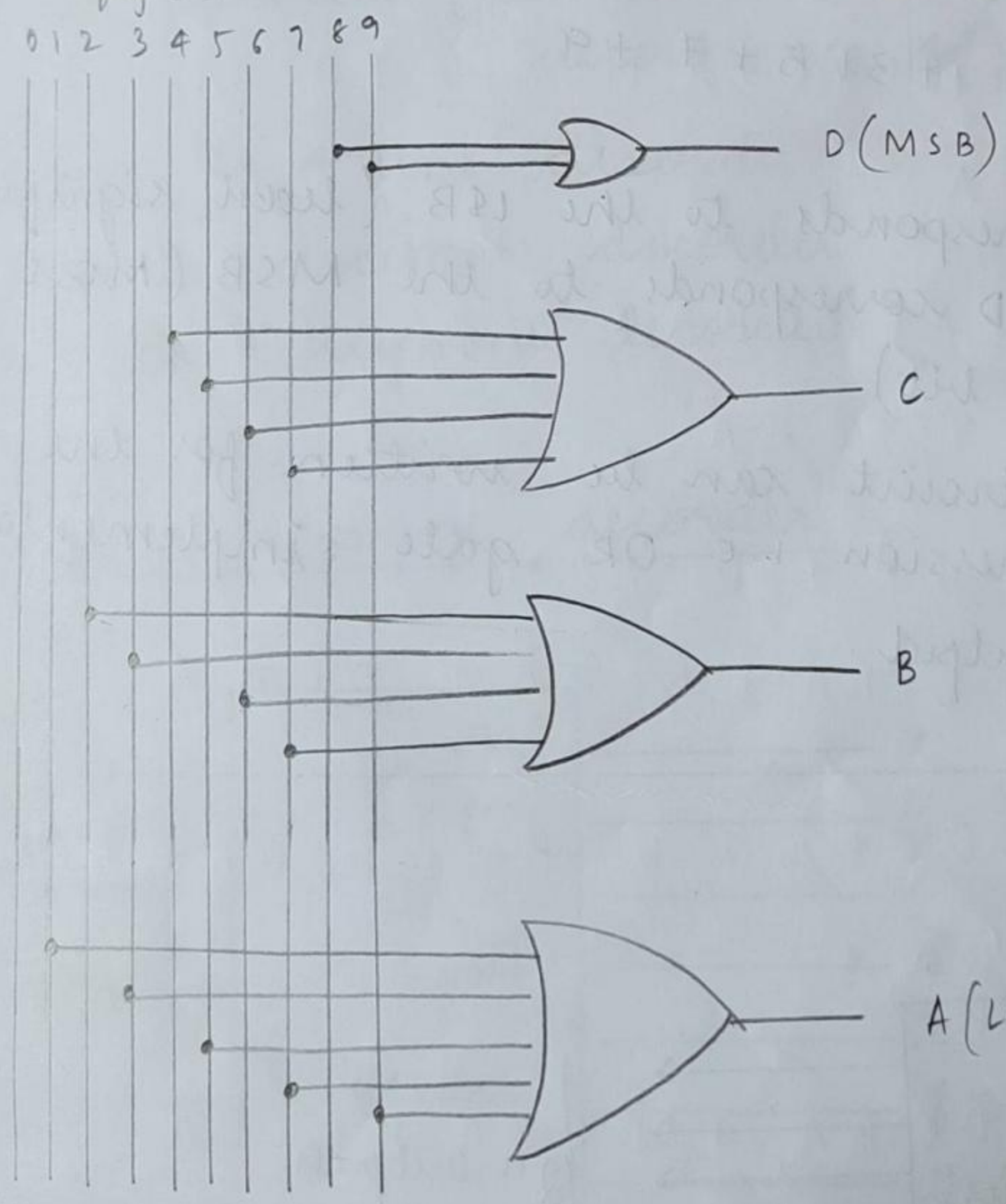


fig (a)

Input	Outputs
	D C B A
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

fig (b)

fig (c)



The Decimal to BCD encoder converts each of the decimal digits 0 to 9 to a binary code. It accepts 10 input lines and produces 4 bit output code corresponding to the activated input. This is also called 10 line to 4 line encoder. The fig (a) shows logic symbol and fig (b) represents the truth table and fig (c) shows the logic circuit of decimal to BCD encoder.

From the truth table we can determine the relationship b/w each BCD bit and the decimal digit.

For decimal digits 8 and 9 can be expressed as OR function as follows

$$D = 8+9$$

$$111^{th} C = A+5+6+7$$

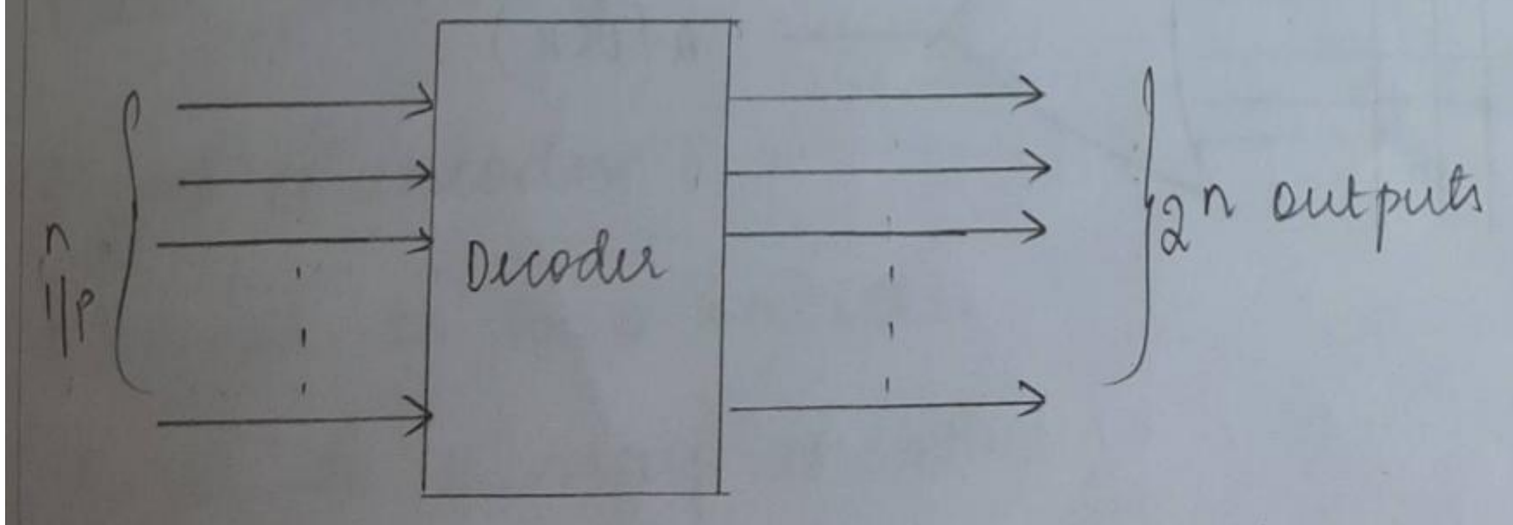
$$111^{th} B = 2+3+6+7$$

$$111^{th} A = 1+3+5+7+9$$

where A corresponds to the LSB (least significant bit) and D corresponds to the MSB (Most significant bit)

The logic circuit can be written for the above expression i.e OR gate implementations for each output.

Decoder



The Decoder converts coded information such as Binary into are recognisable form such as decimal. for example:- BCD to Decimal decoder converts to a BCD code into a appropriate decimal digit.

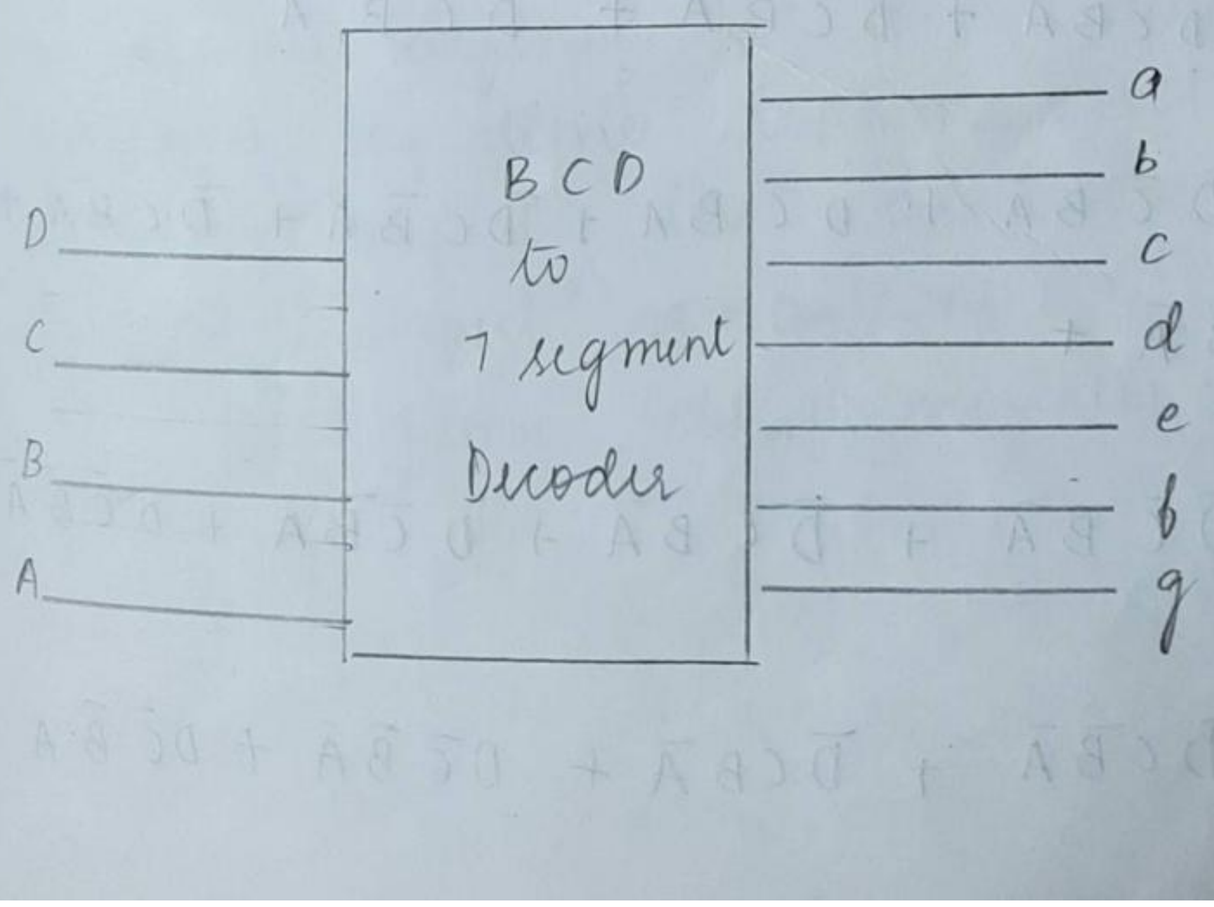
A decoder is a logic circuit that accepts the binary number and activates only the output that corresponds to that input number. all other outputs remain inactive.

the diagram for a general decoder is shown in figure above with 'n' inputs and 'n' outputs. there are 2^n possible input codes for each code, only one of n outputs will be active all other outputs are low.

Examples or types

* 4 bit decoder (Also called as 4 line or 16 line decorder or 1 of 16 decoder)
* 2 line to 4 line decorder
* BCD to decimal decorder
* BCD to 7 segment decorder

BCD to 7 segment decorder



arrangements of segments

INPUTS				OUTPUTS							
D	C	B	A	a	b	c	d	e	f	g	Digit displayed
0	0	0	0	1	1	1	1	1	1	0	0
0	0	0	1	0	1	1	0	0	0	0	1
0	0	1	0	1	1	0	1	1	0	1	2
0	0	1	1	1	1	1	1	0	0	1	3
0	1	0	0	0	1	1	0	0	1	1	4
0	1	0	1	1	0	1	1	0	1	1	5
0	1	1	0	0	0	1	1	1	1	1	6
0	1	1	1	1	1	1	0	0	0	0	7
1	0	0	0	1	1	1	1	1	1	1	8
1	0	0	1	1	1	1	0	0	1	1	9

$$a = \bar{D}\bar{C}\bar{B}\bar{A} + \bar{D}\bar{C}B\bar{A} + \bar{D}\bar{C}BA + \bar{D}C\bar{B}A + \bar{D}CBA + D\bar{C}\bar{B}\bar{A} + D\bar{C}\bar{B}A$$

$$b = \bar{D}\bar{C}\bar{B}\bar{A} + \bar{D}\bar{C}\bar{B}A + \bar{D}\bar{C}B\bar{A} + \bar{D}\bar{C}BA + \bar{D}C\bar{B}\bar{A} + \bar{D}CBA + D\bar{C}\bar{B}\bar{A} + D\bar{C}\bar{B}A$$

$$C = \bar{D}\bar{C}\bar{B}\bar{A} + \bar{D}\bar{C}\bar{B}A + \bar{D}\bar{C}BA + \bar{D}C\bar{B}\bar{A} + \bar{D}C\bar{B}A + \bar{D}CB\bar{A} + D\bar{C}\bar{B}\bar{A} + D\bar{C}\bar{B}A$$

$$d = \bar{D}\bar{C}\bar{B}\bar{A} + \bar{D}\bar{C}B\bar{A} + \bar{D}\bar{C}BA + \bar{D}C\bar{B}A + \bar{D}CB\bar{A} + D\bar{C}\bar{B}\bar{A}$$

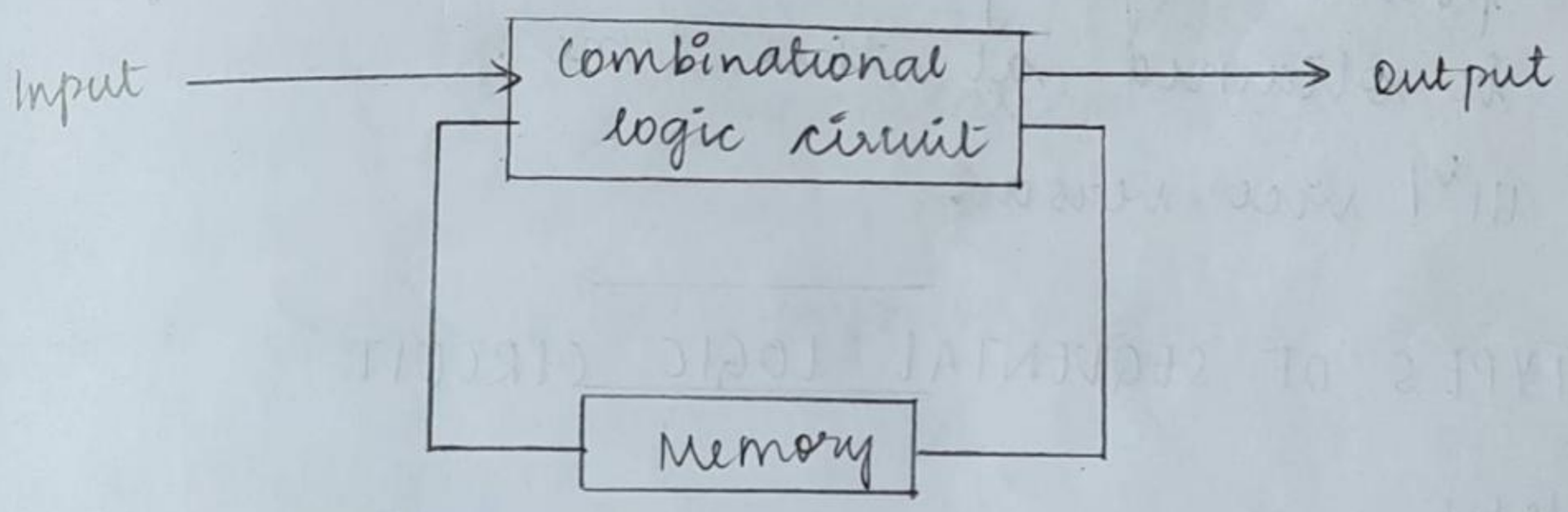
$$e = \bar{D}\bar{C}\bar{B}\bar{A} + \bar{D}\bar{C}B\bar{A} + \bar{D}C\bar{B}\bar{A} + D\bar{C}\bar{B}\bar{A} + D\bar{C}B\bar{A}$$

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

$q = \bar{D}\bar{C}B\bar{A} + \bar{D}\bar{C}BA + \bar{D}C\bar{B}\bar{A} + \bar{D}C\bar{B}A + \bar{D}CB\bar{A} + D\bar{C}\bar{B}\bar{A} + D\bar{C}\bar{B}A$

fig (a) shows the 7 segment indicator and fig (b) shows the logic symbol and fig (c) shows the truth table for BCD to 7 segment decoder it translate 8421 BCD code to 7 segment display that switch on the proper segment on the display. The display will be decimal number 7 LEDS labelled as a, b, c, d, e, f, g by forward bias different leds. we can display the digit from 0 to 9 i.e for eg:-

SEQUENTIAL LOGIC CIRCUIT

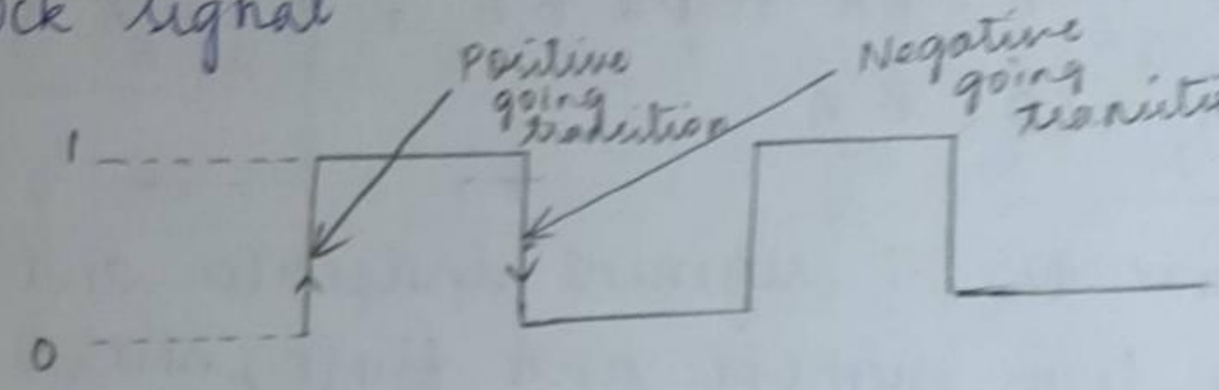


The sequential logic circuits have memory. It is made up of combinational logic circuit and memory elements.

It can be defined as the output level at any instant of time depends not only on present inputs but also on the previous output / input conditions.

Eg:- Flip flops, Shift register, counters etc

Clock signal



Clock is a rectangular pulse train (or) square wave as shown in above figure. The clock input determines when the output will change state i.e, the change of transition. There are two types of transition.

1) Positive going transition
2) Negative going transition.

When the clock changes from 0 i.e is low state to 1 i.e high state then it is called positive going transition. And output will be obtained at this time.

111ly vice versa.

TYPES OF SEQUENTAL LOGIC CIRCUIT

1) Latch
2) Flip flop
3) Shift register
4) Counters etc

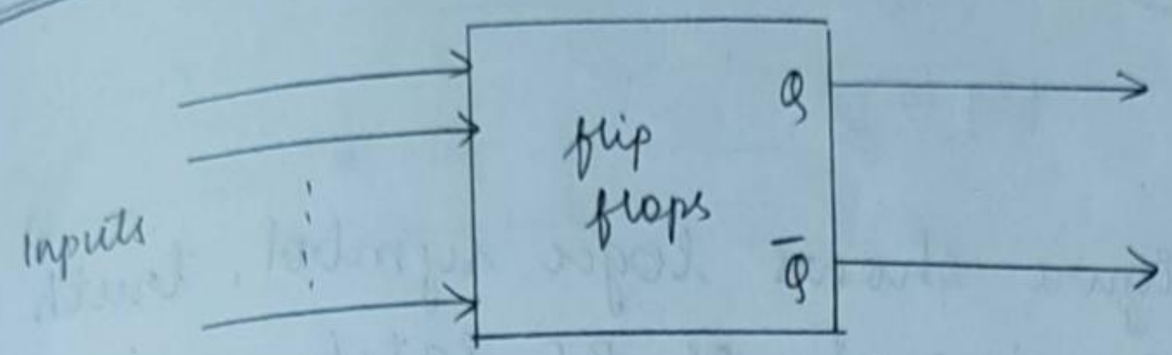
Latch.

The basic building block of a sequential logic circuit that stores single bit of data is called Latch.

Latch is similar to flip flops but without clock input signal. It can remain in either two status i.e, SET (1) , RESET (0)

In short we can call Latch as asynchronous ~~flip flop~~.

Flip flops.



A flipflops are synchronous sequential logic circuit (or) bistable electronics circuit that can be SET (1) or RESET (0).

Fig above shows the logic symbol for flipflops there can be more than two inputs but only two outputs Q and $\bar{Q}$. The outputs in flip flops are always complement to each other the output are obtained in synchronisation with the clock pusle as a trigar, hence called as clocked flipflops (or) synchronous flip flops

Types of flip flops.

1) RS flip flops
2) JK flip flops
3) Master slave JK flip flop
4) D flip flop
5) T flip flop

SR Latch

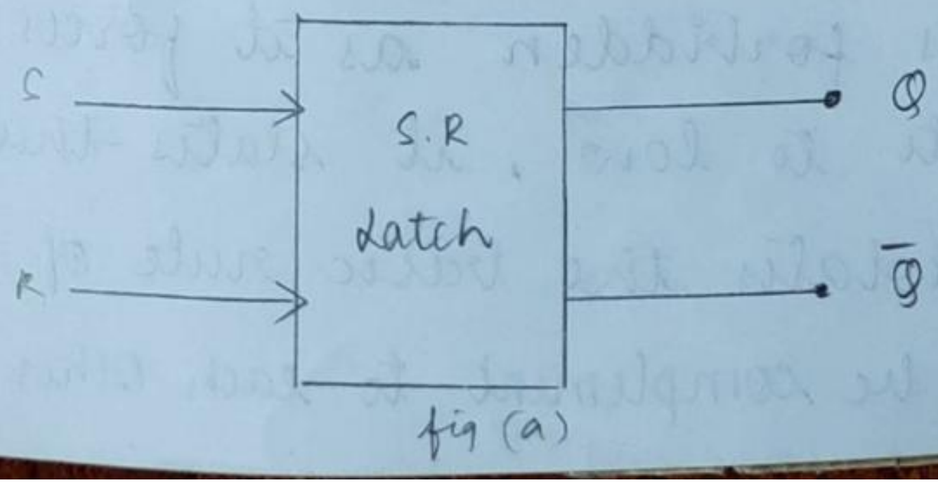


fig (a)

fig (b)

I/P		O/P	
S	R	Q	$\bar{Q}$
0	0	Previous state	
0	1	0	1
1	0	1	0
1	1	Not change ~~valid~~	

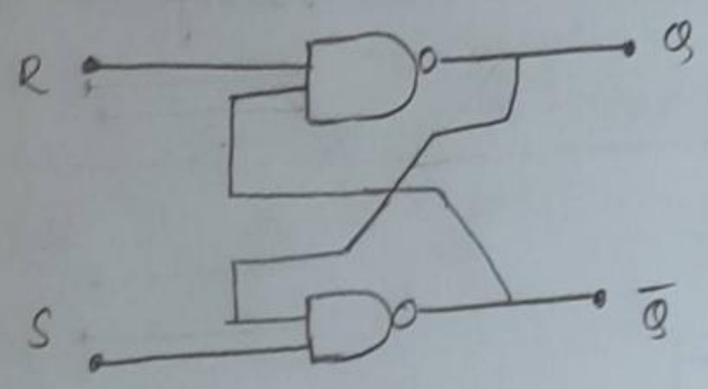


fig (c)

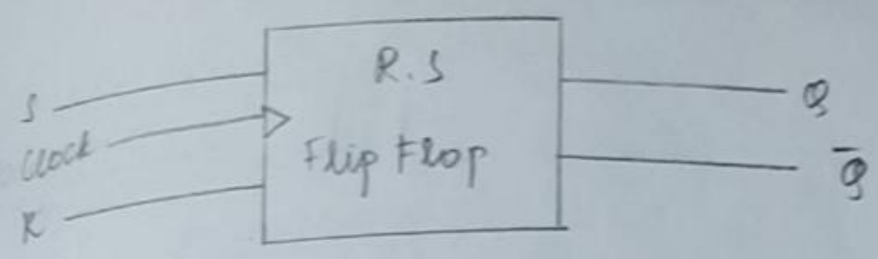
In the above figure shows logic symbol, truth table and logic circuit of RS latch using NOR gates. RS is a RESET and SET latch that as two outputs difined as Q and $\overline{Q}$ and the outputs are complement to each other.

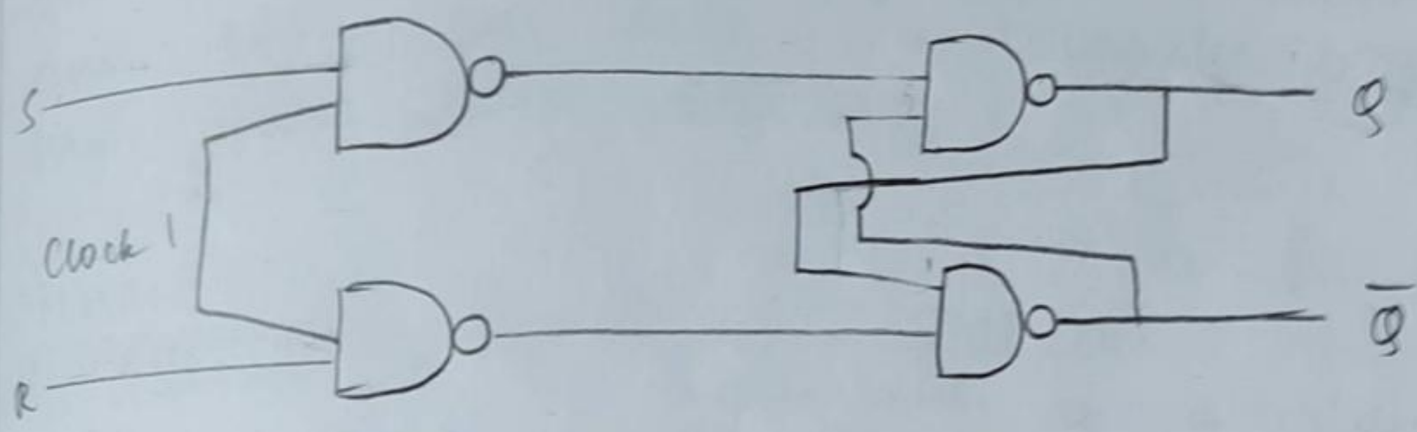
The working of RS latch is given by when $\underline{R = 0}$ and $\underline{S = 0}$ since both inputs are zero the NOR gates have no effect on the output i.e, the latch remains in the present state is unchanged, when R = 0 this condition forces the output of low. Both inputs of 'A' are now low and hence output is high. Thus '1' at S input is ~~set~~ said to SET and hence Q = 1.

when R = 1, S = 0 this condition forces the output of NOR gate 'A' low and NOR gate 'B' inputs are now low and hence the output must be high. thus '1' at the R input RESETS the latch i.e, Q = 0. when R = 1 and S = 1 this input condition is forbidden as it forces the output of NOR gate to low, it states that $\overline{Q} = Q = 0$ it violates the basic rule of latch i.e output should be complement to each other and it

is the forbidden condition.

SR Flip flops.





Inputs			Outputs		Comments
Clock	S	R	Q	$\bar{Q}$	
⊓	0	0	Previous state		Reset State
⊓	0	1	0	1	Set State
⊓	1	0	1	0	forbidden State
⊓	1	1	1	1	Not allowed state

UNIT-3 INTRODUCTION TO COMPUTER CONCEPTS

COMPUTER

A computer is a electronic device or a machine which performs input, output operations and stores the data. The computer performs with high speed and accuracy.

CHARACTERISTICS OF A COMPUTER

Some of the important characteristics are :-

1) **Speed** :-

Computer can process the data and solve the problems at fast rate extremely. i.e millions of instructions are executed per second. A computer can execute three millions of instructions per second.

2) **Accuracy** :-

A computer accuracy is very high. The level of accuracy depends on the instructions and the type of processor used in the computer.

3) **Reliability** :-

The output generated by the computers are very reliable because computers will never make mistakes of their own.

4) **Versatility** :-

The multi processing feature allows it to perform multiple tasks simultaneously with equal time, making it versatile.

5) **Storage capacity** :-

The computers are capable of storing huge amount of data that can be retrived very quickly

6) Diligence :-

The computers can work continously for hours without a single error and it performs all the operation with same speed and accuracy.

APPLICATIONS OF COMPUTER

The various applications of computers are :

1) Education :-

The computers are very much useful in teaching learning procss of education with a system known as computer based education (CBE). CBE involves control, delivery and valuation of learning.

2) Business :-

The computers are used in business organisations for salary calculation, budgets calculations, sales analizes, managing employe data base, etc.

3) Banking :-

The computers are used in the various banks for providing online account facility to maintain the accounts of costumers, used as ATM machines for providing money, etc.

4) Science :-

Due to high speed and accuracy the computers have immense application in the scientific field. Computers are used in the research field, study of earthquakes effects, for satellites based applications, weather monitoring, etc.

5) Military :-

Computers are used in the defence, modern tanks, missiles launching, automated weapons, etc. The military employs computerized control system used for military communication, operating and planning etc

6) Computers are also used in the various fields such as marketing, Hospitals, Networking, Agriculture, Communication etc.

GENERATIONS OF COMPUTER

The computers may be classified into 5 number of generations.

They are :-

1)	1st generation computer	vaccum tubes	(1945 - 1955)
2)	2nd generation computer	Transistors	(1955 - 1965)
3)	3rd generation computer	IC's	(1965 - 1980)
4)	4th generation computer	Micro processor	(1988 - 1989)
5)	5th generation computer	Micro processor	till date

First generation computers

The first generation computers are univac (universal automated computers) from IVM. In this generation vaccum tubes were used. The machine can perform business data and process it, due to vaccum tubes. Vaccum tubes occupied huge space and more energy but slow in input and output It suffered from heat and maintanenace problem It uses the machine language for programming.

Second generation computers

* To overcome the difficulties faced in the first generation computers due to vaccum tubes, transistors were used in the second generation computers.
* Transistor is a small component made up of semi - conductor material, due to which the size was reduced.
* The computers perform operation faster with improved storage capacity.

* This computers can work with high level language and also with machine level language.

Third generation of computers

The third generation computers used integrated circuits instead of transistors, in which the circuits are fixed on a silicon chip

* As more number of components like transistors are moulded on the IC, the computers work, smaller, faster, and flexible in terms of input and output.

* As this computer satisfy the needs of small business, it became more popular as a mini computer

Fourth generation of computers

The fourth generation computers used IC's like VLSI (Very large scale integration)

* Due to VLSI technology the computers were faster, smaller, reliable and it became more user friendly and widely used as a personal computers.

* The fourth generation computers also include super computers that are best in processing, speed, storage and cost. It can process upto billions of instructions per second.

* The fourth generation computers are used in the year (1988 - 1989).

Fifth generation of computers

The fifth generation computer use more advanced technology with ultra large scale integration. This computers are portable, powerful variety of storage mechanisms and advanced software technology.

* The fifth generation computers includes desktops, notebooks (laptops), artifical intellegence that use pentium processor for processing.

* In future, the users can except even more faster and advanced computer technology like nano technology

CLASSIFICATION OF COMPUTERS

The computers can be classified mainly into three groups. They are

1) Classification according to purpose
2) Classification according to data handling
3) Classification according to functionality

1) Classification according to purpose

Depending upon the purpose served the computers can be classified further into two types

a) Specific purpose
b) General purpose

The general purpose computers are designed to perform various tasks. They have the ability to store the large ~~problems~~ programs and data but lacks in speed and ~~effiency~~ efficiency

The specific purpose computers are designed to perform a single task or specific task. This computers are high in speed and ~~efficiency~~ efficiency

2) Classification of according to data handling

Here computers are classified into three types

a) Analog computers
b) Hybrid computers
c) Digital computers

Analog computers

The Analog computers work on the principle of measuring, in which the measurments obtained translated into data. The Analog computers are employed with voltage, resistances etc that represents quantities being manuculated.
The accuracy is very less.

Digital Computers

Digital computers are the computers which operates on digital data. This computers process the data into digital values that is 0's and 1's. They are fast and accuracy machines. Here millions of instructions are executed very fastly.

Hybrid Computers

The Hybrid computers combines the measuring features of analog computer and counting feature of digital computers. These computers use the analog component for computational purpose and digital components for storage of data. It is used effectively for scientific and industrial applicat

3) Classification according to size and capability/functionality

The computers are classified based on size and capability. They are

a) P C — Personal computer
b) Palmtops
c) Laptops
d) Super Computers
e) Mainframe computers
f) Mini computers
g) Server

PC

Personal computers are small and single user computer which is known as micro computers. This computers are used in the most of the organisations like business, offices, home etc. This computers are portable which has its own keyboard, mouse, storage data etc

Palmtops

Palmtops are commonly called as personal digital assisstance (PDA). They are lightly integrated computer which are small and light in weight

Laptops

Laptops are nothing but notebook computers and they are portable.

Server

Server is a computer which provides service to other computer and especially used in networking.

Mainframe Computer

Mainframe computers are integrated by using multiple of CPU. This computer supports hundred to the word users simultaneously. This computers are used in small companies, institutions, scientific calculators and military operation.

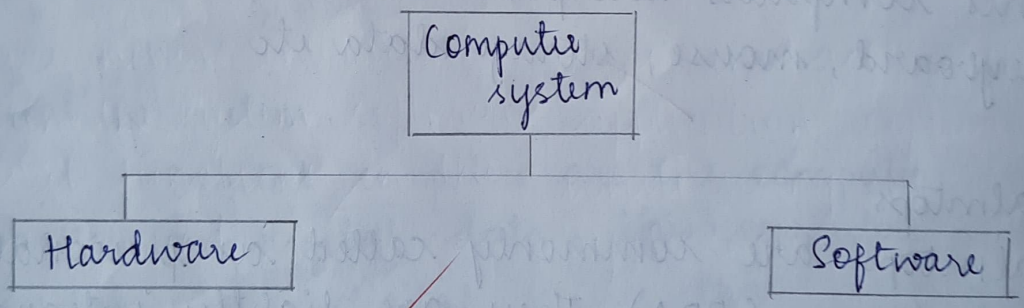
Minicomputer

Minicomputers is a multi user computer which supports many user at a time.

Supercomputer

Super computers are very expensive which cause around mission declares. This computers are used in scientific, aerospace, robotic application and satellite communication.

COMPONENTS OF COMPUTER



Hardware

The Physical parts that make up a computer which are inter connected is called computer hardware.

Ex:- Mouse, keyboard, CPU etc

Software

A software is a collection of programmes where programme is a set of instructions given to the

computer to perform specific tasks or particular task. The software controlls and operates the ~~ha~~ computer hardware to get desired output. Every action and the function of the hardware is given by ~~hardware~~ software.

Types of software

The computer software is divided into two main types

1) System software
2) Application software

* System software

System software is a set of programmes which are required for controlling and managing the hardware components of a computer system. It should be loaded in the computer system before using the system to perform ~~particular~~ any task. In the absence of the software, the computer will be of no use.

Ex:- Operating system, Drivers, compiler, assembler etc

* Application software

The softwares that are designed to fulfill the requirements of user for performing specific tasks.

Differences between system software and application software;

P.T.O

System Software	Application Software
* They are designed to manage the resources of the system like memory, process management etc	They are designed to fulfill the requirements of user for performing a specific task
* It is written in a low level language like machine language or assembly language.	High level language is used to write the application ~~level~~ software. Ex:- Java, C++
* The system software is a general ~~software~~ purpose software	The application software is a specific purpose software
* It is capable of running independently	It cannot run independently.
* Users never interact with the system software as it functions in the background	Users interact with the application software while using the specific applications.
* The system software starts running when the system powered on and runs until the system is powered off	The application software starts when the users begins and it ends when the user stops it.
* The system software is installed in the computer system at that time when the operating system is installed.	Installed as per the requirements.

E - Accessibility software (Electronic accessibility)

Electronic accessibility refers to the ease of use of information and communication technology such as the internet by the people with disability. Websites need to be developed so that the disabled users can access the information.

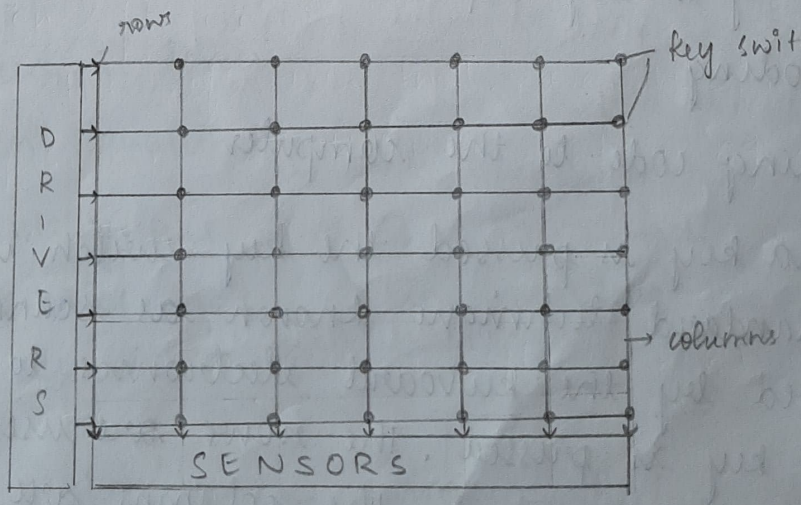
For example,

1) For people who are blind, websites need to be able to be interpreted by programmers which read the text a loud and describe any visuale images

2) For people who have low vision, webpages need adjustable sized font and sharply contresting colours.

3) For the people who are deaf, audio content should be accompanied with the text version of the dialogue.

Working principle of Keyboard.



Keyboard is an input device through which data, programmes and certain commands to a software can be keyed. It translates numbers, letters and symbols into a signal that can be interpreted by a computer. Most of the english key words are based on the "QWERTY" designed. The standard keyboard have ~~one or~~ 102 to 114 keys. The functions can be programmed by the software designer. Many special keys are provided including right, left and up, down keys, Shift key, Inter key, Alter key, backspace, delete, Homepage, insert, Print screen, number lock, scroll, caps lock, etc. Keyboard consists of key switches, there is one key switch for each letter, number, and symbols much like a type writer. Generally key switches are connected in a matrix of rows and columns.

The functions to be performmed by the keyboard ~~are~~

* Sensing a key depression
* Encoding
* Sending code to the computer

When a key is pressed the key switch is activated. The standard technique known as scanning is followed by the keyboard electronics to determine which key is pressed. The rows are used as input to the matrix. The columns are sensed by electronic circuits. The 8 bit code is generated and sent to the computer. Drivers are the external installed software. Any external device connected

to CPU needs drivers to be installed.

Printer

A Printer is an electro mechanical device it has both electronic circuit and mechanical assemblies. The electronic circuits in a printer are usually referred to as printer electronics. A computer interface links the printer with the computer commands and data from the computer are sent to printer through this interface. The printer sends its status to the computer through this interface. The mechanical assemblies include print head assembly, print carriage motor, ribbon assembly, paper movement assembly, sensor assemblies etc. There are two types of printer

1) Impact printer
2) Nonimpact printer

Explain Impact printer with an example.

* It prints the character and image by striking printer hammer wheel against linked ribbion.

* Its speed is lower
* Its print quality is lower
* It normally uses continous paper sheet while printing
* It generates noise during printing
* It is less expensive

Ex:- Dot matrix printer, daisy wheel printer, drum printer, band printer and chain printer etc.

Explain Nonimpact printer with an example.

→ * It prints characters and images without ~~doubt~~ striking the paper.
* Its speed is faster
* Its print quality is higher
* It normally uses individual paper sheet while printing.
* It does not generate noise during printing
* It is more expensive

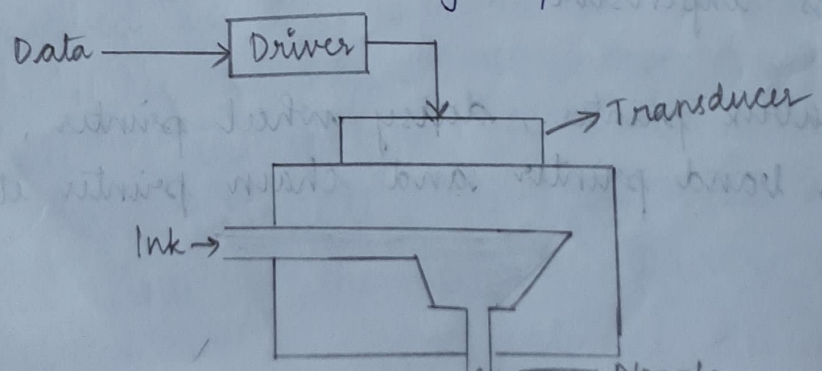
Ex:- Laser printer, Thermal printer, inkjet printer and electro static printer.

Explain dotmatrix printer

→ Dot matrix printer is an impact printer in which the character is formed by large space dots in the form of matrix. It uses tiny hammers in its prints. Dot matrix printers are relatively expensive and does not produce high quality output. However they can print carbon copies thats something laser and inkjet printer cannot do. The printing quality is determined by no. of ~~inks~~ ~~in~~ pins in the print head. It can varry from 9 to 24.

Write a note on inkjet printer.

→



Inkjet printer is peripheral that produces hard copy by spraying the ink on the paper. The inkjet printer produces copy with a resolution of atleast 300 dots per inch. Instead of metal needles they use 100 of tiny guns to fire the dots of ink on the paper In the inkjet printing mechanism has print head which has tiny nozzles called as jets. The principle advantage of ink jet printer is fast. Transduced converts the electrical energy into another form. There are two types of print head based on the transducer they are

1) Thermal
2) Piezo electric

Laser Printer

The line, dot matrix, and ink jet printers need a head movement on a ribbon to print characters. This, mechanical movement is relatively slow due to high inertia of mechanical elements. In laser printers these mechanical movements are avoided. In these printers, an electrinically controlled laser beam traces out the desired character to be printed on a photo conductive drum. The exposed areas of the drum gets charged, which attracts an oppositely charged ink from the ink toner on to the exposed areas. This image is then transferred to the paper when it comes in contact with the drum with pressure applied by the pressure roller. The charge on the drum decides the darkness of the print. when charge is more, more ink is attracted and we get a dark print. This pressure roller transfer the black toner onto the paper. Since the paper is moving at the same speed as the drum the paper picks up the Image pattern precisely. Finally, the printer passes the paper through a pair of heated rollers called fuser. As the paper passes through these rollers. The loose ~~toner~~ toner power gets melted and fuses with the fibres in the paper. The paper is then brought out of the printer.

Proprietary Software

Proprietary software is a software that is copy writed, which means it can only be obtained by paying for a license. Proprietary software has many advantages like the product should be free of bugs. If bugs still exist, updates know as patches are often provided free of charge, which fix these bugs. Ex: Windows

Open Source

The term open source refers to the software whose source code freely available on the internet. Ex:- firefox, Open office, Gimp etc

Firefox - A ~~bew~~ web browser that compits with internet explorer.

Open office - A compitetor to microsoft office.

Gimp - A graphic tool with features founding photoshop.

Freeway software

Freeway is a software, most often proprietary that is distributid at no cost to the end user. There is no agreed upon at set of rights, license, EULA that defines freeway unambiguously, every publisher defines its on rules for the freeway it offers.

Ex:- Adobe reader, Free Studio, Skype etc

COMPUTER NETWORK CONCEPTS

A network can be difined as a group of computer and other devices connected in some ways so as to able to exchange data.

Each of this devices on the network can be taught of a node.

CATEGORIES OR TYPES OF NETWORK

A computer network is mainly of four types

* LAN (Local Area network)
* PAN (Personal Area network)
* MAN (Metropolitan Area network)
* WAN (Wide Area network)

LAN

A local area network (LAN) is the collection of devices that are connected together in one physical location, such as building, office or hom

PAN

A personal area network (PAN) is formed when two or more computers are inter connected to one another wirelessly over a short range, typically less than about 30 feet

MAN

Metropolitan area network (MAN) is a computer network that connects no. of LAN's to form larger network, so that computer resources can be shared. This type of network covers larger area than a LAN but smaller than the area covered by WAN.

MAN is specially designed to provide high speed connectivity to the user in which speed ranges in terms of MBPS

Examples of MAN are ~~copa~~ cabale TV network, used in government agencies, University campus etc

---

Protocol

A Protocol is the set of rules that governs the communication between computers on a network

Protocols in Application Layer

TELNET - Telnet stands for telecommunication network. TELNET is a network Protocol used to virtually ~~acc~~ access the computer and to provide a two way, collaborative and text based communication channel between two machines.

FTP - FTP stands fore File Transfer Protocol. It is the standard network protocol used for the transfer of computer files from a server to a client on a computer network

NFS - NFS stands for Network file system. It is a client server application that lets a computer user view and optionally store and update files on remote computer.

SMTP - SMTP stands for Simple mail transfer Protocol. It is the set of communication guide lines that allows software to transmit an electronic mail over the internet is called simple mail transfer Protocol

User datagram protocol

SNMP - SNMP stands for semple network management Protocol. This protocol uses UDP port no. 161/162. SNMP used to monitor the network, detect network faults and some times even it is used to configur remote devices

ADVANTAGES OF PROTOCOL

- Flexibility
- Less time to transfer files
- Transfers data to different systems
- Allows programmes to run on different systems
- High speed

WAN
Wide Area Network is a telecommunication network it extends over a large geographical area for primary purpose of computer networking
Ex: Internet

DATA PROCESSING

Collection, manuculation and processing collected data for required use is known as Data Processing.

Its a techaique normally performed by a computer

METHODS OF DATA PROCESSING

The different method of Data Processing are

1) Single user Programming
2) Multiple Programming
3) Real time Processing
4) Online Processing
5) Time sharing Processing

Proces is nothing but Task

6) Distributed Processing

Single user Programming

It is usually done by a single person for his personal use. This technique is suitable for small offices.

Multiple Programming

* This technique provides facility to store and execute more than one programme in the central processing unit (CPU) simultaneously.
* Further, the multiple Programming technique increases the overall working efficiency of the respective computer.

Real time Processing

* This technique facilitates the user to have direct contact with the computer system.
* This technique made easy of data Processing. This technique is also known as direct mode or the interactive mode technic and it is developed exclusively to perform one task. It is sort of online processing, which always remains under execution

Online Processing

* This technique facilitates the entry and execution of data directly. Hence it does not store or accumulate first and last process. This technique is developed in such a way that reduces

the data using terms, as it validates data at various points and also ensures that only correct data is stored.

* This technique is widely used for online applications

Time sharing Processing

* This is another form of online data processing that facilitates several users to share the resources of an online computer system.
* This technique is adopted when results are needed fastly.
* As the name suggest, this system is time based. Following are some of the major advantages of time sharing processing
  1) Several users can be served simultaneously
  2) All the users have almost equal amount of processing time
  3) There is the possibility of interaction with the running programme.

Distributed Processing

* This is the specialised data processing technique in which various computers (which are located remotely) remain interconnected with a single host computer making a network of computers

---

COMPUTER SECURITY

* It is required for the following major reasons
  1) To prevent the damage of the hardware
  2) To prevent theft of damage of installed software

3) To Prevent thieft or damage of store data and information

4) To Prevent the disruption of service

TYPES OF THREAT

Following are the most common types of computer threats

* Physical damage - It includes fire, water, pollution etc.
* Natural events - It includes climatic, earthquake volcanic activity etc.
* Loss of services - It includes electrical power, air conditioning, telecommunication etc
* Technical failures - It includes problems in equipments, software capacity saturation etc
* Deliberate type - It includes spying, illegal procur-ing of data etc

SOURCES OF THREAT

Possible sources of a computer thread maybe

* Internal - It includes employees, partners, contracters and vendors.
* External - It includes cyber criminals (proffesional hackers), non-proffesional hackers, activists, malware (virus) etc.

COMMON TERMS

Following are the common terms frequently used to define computer threat - Viruses threats

Virus threat

A computer virus is a programme design to distrupt the normal functioning of the computer without the permission of the user.

Spy ware threat

It is a computer programme that monitors the users online activity or installs programmes without users consent for profit or theaft of personal information.

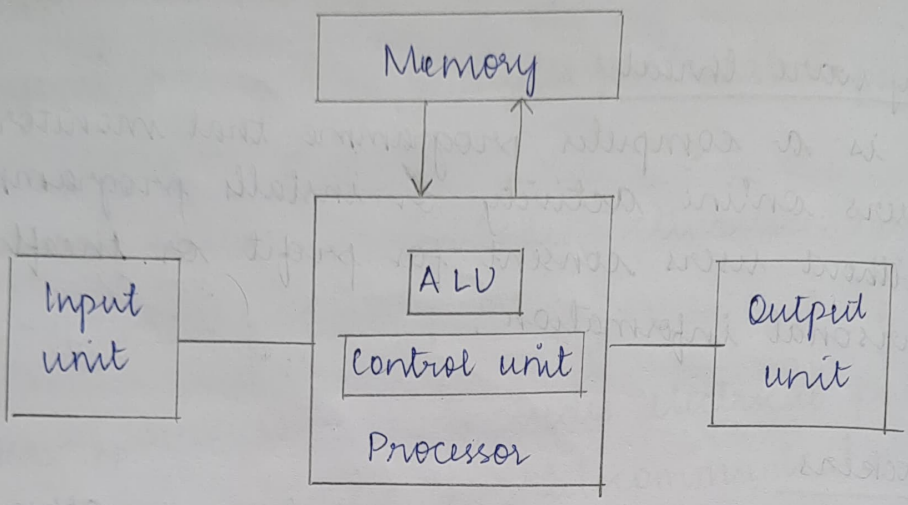
Hackers

They are the programmers who put other on threats for their personal gain by breaking into computer systems with the purpose to steal, change or destroy information

Phishing threats

Its an illegal activity through which phishers attempt to stial sensitive, financial or personal data by means of fraud email or instand messages.

UNIT-4 INTRODUCTION TO COMPUTER ORGANISATIONS AND OPERATING SYSTEMS



Memory hierachy

A memory unit is a essential component of any digital computer since it is needed for storing programmes and data.

The various types of memory are:

* Primary memory / Main memory
* Cache memory
* Auxiliary memory

Main Memory / Primary memory

The Main memory in a computer system is often referred to as random access memory (RAM). This memory unit communicates with CPU [central Processing unit] directly, and also with auxiliary memory through input/output processor. The programs that are not currently required in the main memory are transferred to the Auxiliary

memory to provide a space to currently used programs and data.

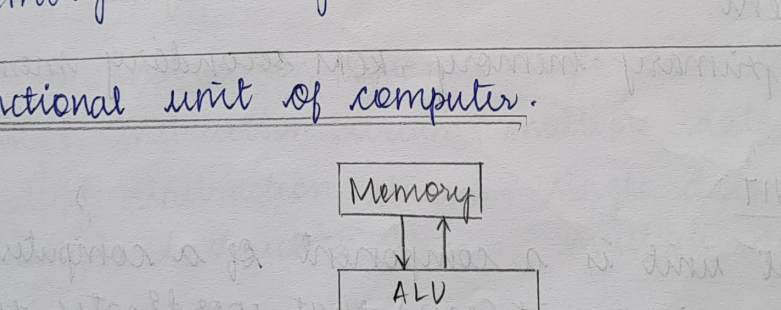
Cache Memory

The data or contents of the main memory that are used frequently by the CPU are stored in the cache memory so that the processor can easily access that data in shorter time. Whenever the CPU requires accessing memory, it first checks the required ~~ta~~ data in the cache memory. If the data is found in the cache memory, it reads from the fast memory. Otherwise the CPU moves on to the main memory for the required data.

Auxiliary Memory

Auxiliary memory is known as permanent memory which is low cost, highest capacity and slowest access storage in the computer system. Auxiliary memory provides the storage for programmes and data that are kept long term storage. The most common examples of auxiliary memory are magnetic tapes and magnetic discs

Functional unit of computer.



A computer consists of five functionally independent main parts input, memory, arithmetic logic unit (ALU) output and control unit.

Input units

Input units are used by the computer to read the data The most commonly used input devices are keyboards, mouse, joysticks, trackballs, microphones etc

- However, the most well-known input device is a keyboard whenever a key is pressed, the corresponding letter or digit is automatically translated into its corresponding binary code and transmitted over a cable to either to memory or the processor.

Memory unit

- The memory unit can be referred to as the storage area in which programs are kept i.e are running, and that contains data needed by the running programs.
- The memory unit can be categorized in two ways namely, primary memory and secondary memory
- It enables a processor to access running execution application and services that are temporarily stored in a specific memory location.
- Primary storage is the fastest memory that operates at electronic speeds.
- Primary memory temporarily secondary memory is permanent
  eg. RAM primary memory, ROM secondary memory.

CONTROL UNIT

The control unit is a component of a computer's central processing unit (CPU) that coordinates the operation of the processor. It tells computer's memory, ALU and input and output devices how to respond to a program's instruction.

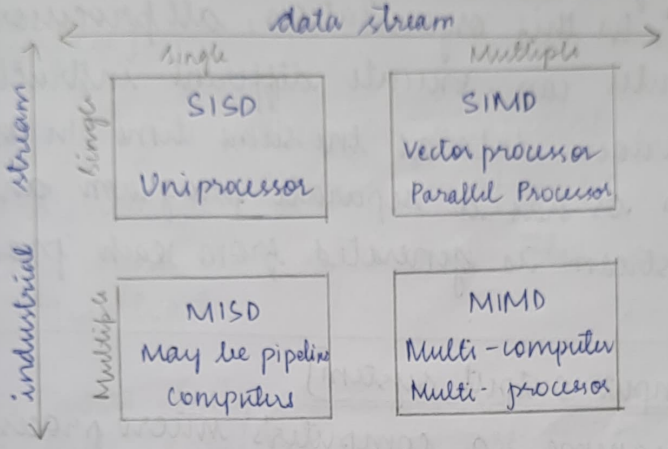
ALU

The actual processing of the data and instruction are performed by ALU. The major operation performed by the ALU are addition, subtraction, multiplication, division, logic and comparison.

- Output - The Primary function of the output unit is to send the processed results to the user. Output devices display information in a way that the user can understand.

- Output devices are pieces of equipment that are used to generate information or any other response processed by the computer
  Eg :- monitor, printer etc

FLYNN's CLASSIFICATION OF COMPUTER.



Flynn's classification divides computer into 4 major group.

1. Single instruction stream, single data stream (SISD)
2. Single instruction stream, multiple data stream (SIMD)
3. Multiple instruction stream, single data stream (MISD)
4. Multiple instruction stream, multiple data stream (MIMD)

SISD

SISD stands for 'Single Instruction and single data stream'. It represents the organization of a single computer containing a control unit, a processor unit, and a memory unit. Instruction are executed sequentially and the system may or may not have internal parallel processing capabilities.

SIMD

SIMD stands for "Single instruction and multiple data stream'. It represents the organization that includes many processing units under the supression of a

common control unit. All processors recieve that same instruction from the control unit but operate on different items of data.

MISD:

MISD stands for 'multiple instruction and single data stream'. In MISD, multiple processing units operate on one single data stream. Each processing unit operates on the data independently via separate instruction stream.

MIMD

MIMD stands for 'multiple instruction and multiple data stream'. In this organisation, all processors in a parallel computer can execute different instruction operate on various data at the same time. In MIMD, each processor has a separate program and an instruction stream is generated from each program.

BIOS [Basic input output system]

It is a programme, a computer's micro processor uses to start the computer system after it is powered ON. It also manages data flow b/w operating system and attached devices such as hard disk keyboard, mouse, printer etc.

FUNCTION OF BIOS

BIOS has four functions.

1) POST [Power on self test]

This tests the hardware of the computer before loading operating system.

2) Boot strap loader

It is a programme which locates the operating system

3) Software / drivers:

This locates the software and drivers that interface with the Operating i.e running

4) CMOS setup :

CMOS stands for complementary metal oxide of semi - conductor. This is a configuration program that enables the users to alter hardware and system settings.

UEFI

UEFI stands for unified extensible firmware interface. UEFI does the same job has BIOS, but with the one difference i.e it stores all the data about the inialization and start up in an EFI file, instead of storing it on the firmware [(EFI) extensible firmware interface]

The EFI file is stored on a special partiation called EFI system partiation [ESP], EFI system which is on the hard disk. The ESP also contains the boot strap loader.

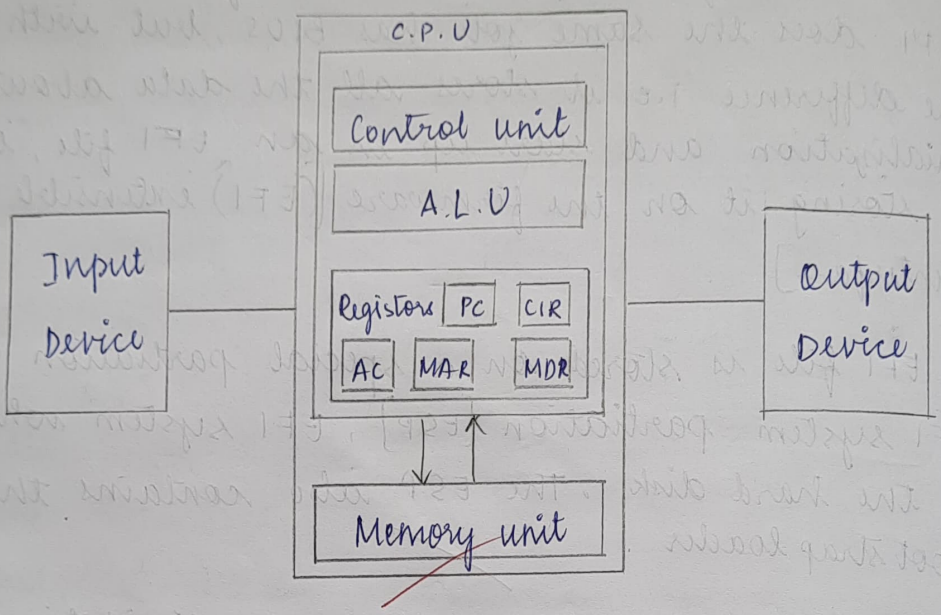
UEFI was designed to overcome the limitations of old BIOS. Some of them are

* UEFI supports drive sizes upto nine zetta ~~bits~~ bytes, whereas BIOS only support only 2.2 tera byte.
* UEFI provides faster boot time
* UEFI has discrete driver support, while BIOS has drive support stored in its ROM, so updating BIOS firmware is bit ~~different~~ difficult
* UEFI offers security like "secure boot", which prevents the computer from booting from unauthorized / ~~unsafe~~ unsigned application.
* UEFI ~~it~~ runs in 32 bit or 64 bit mode, where BIOS runs in 16 bit mode.

Stored Program Concept

Storage of instructions in a computer memory to enable it to perform a variety of tasks in sequence manner.

Von Neumann Architecture / Model



Von Neumann Architecture is based on stored Programme computer concepts, where instructions data and programme data are stored in save memory. This design is still used in most computers produced today. The Von Neumann Architecture is showed in the above diagram.

CPU is an electronic circuit responsible for executing the instruction of a computer programme. It is sometimes refered to as micro processor or processor.

Registors are high speed storage area in the CPU. All data must be stored in registor before it can be processed.

PC [Program Counter]
It contains the address of next instruction to be executed.

CIR [Current instruction registor]
It contains the current instruction during processing

AC [Accumulator]
It stores arithematic and logic results.

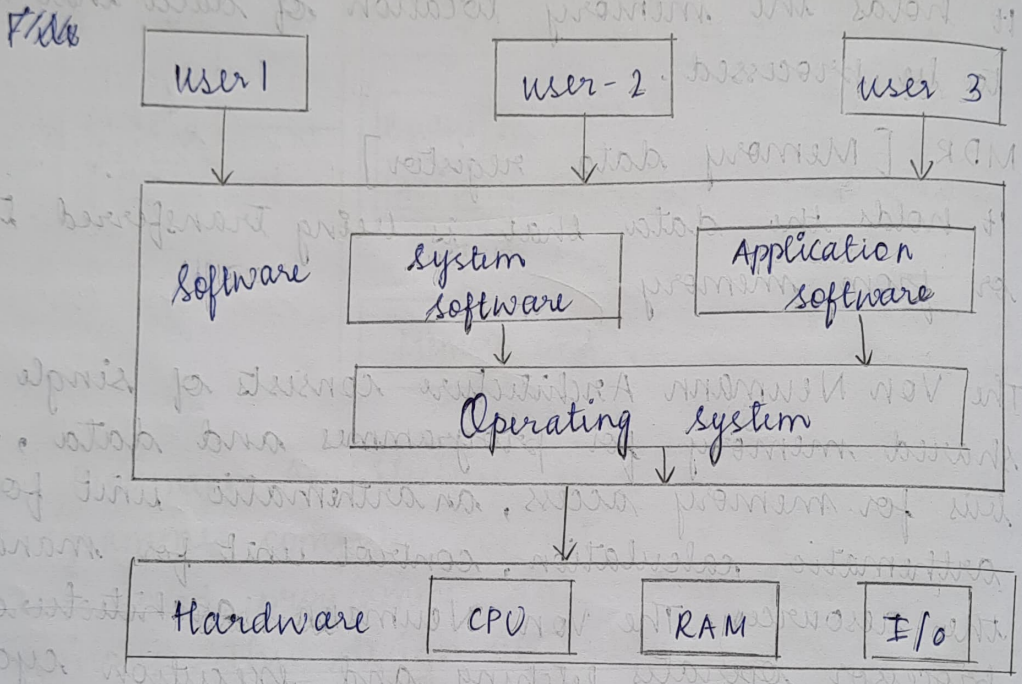
MAR [Memory address registor]
It holds the memory location of data that needs to be processed.

MDR [Memory data registor]
It holds the data that is being transferred to or from memory

The Von Neumann Architicture consists of single, shared memory for programmes and data, single bus for memory access, an arithematic unit for arithematic calculation, control unit for managing the resources. The Von Neumann ~~architicture~~ processor operates fetching and execution cycles seriously

OPERATING SYSTEM

An operating system is an interface between a computer user and computer hardware. An operating system is a software which performs all the basic tasks like file management, memory management, process management, handling input and output, controlling hardware devices.



File management

A file system is normally organized into directories for easy navigation and usage. An operating system does the following activities for file management.

a) Keeps track of information, location, uses, status etc. All these facilities collectively known as file system.

b) Decides who gets the resources and allocates the resources

c) It deallocates the resources

Memory Management

Memory management referres to the manage-ment of primary memory or main memory. Main memory has fast storage that can be accessed directly by the CPU. The Operating system does the following activities for the memory management.

a) Keeps track of primary memory
b) In multiprograming, the OS decides which process will get memory [process = task] when and how much.
c) Allocates memory when a process request.
d) Deallocates memory when a process does not require memory.

Process management

In Multi ~~programing~~ programming environment, the OS decides which process gets the processor when and for how much time. The function is called process scheduling. An operating system does the following activities for processor management.

- Keeps tracks of processor and status of process. The program responsible for this task is known as traffic controller
- Allocates the processor (CPU) to a process
- De-allocates processor when a process is no longer required

Device Management

An operating system manages device communi-cation via their respective drivers. It does the following activities for device management.

- Keeps ~~track~~ tracks of all devices, program respons-ble for this task is known as the input controller
- Decides which process gets the device when and for how much time.
- Allocates the devices in the effecient way.

File Management

Repeated

A file ~~Management~~ system is normally organized into directories for easy navigation and usage thes directories may contain files and other directions.

An operating system does the following activities for file management :-

- Keeps tracks of information, location, uses, status etc. The collective facilities are often known as file system.
- Decides who gets the resource
- Allocates the resources
- De-allocates the resources

TYPES OF OPERATING SYSTEM

Batch Operating System

The users of a batch operating system do no interact with the computer directly. Each user prepares his job on an off-line to device like punch cards and submits it to the computer operato

To speed up processing, jobs with similar needs to are batched together and run as a group. The programmers leave their programs with the operator and the operator then sorts the programs with similar requirements into batches.

The Problems with batch systems are as follows

- Lack of interaction between the users and the job
- CPU is often idle, because the speed of the mechanical input devices is slower than the CPU.
- Difficult to provide the desired priority.

Time Sharing Operating systems

Time-sharing is a technique which enables many people located at various terminals, to use a particular computer system at the same time. Time-Sharing or multitasking is a logical extension of multiprogramming. Processors time which is shared among multiple users simultaneously is termed as time-sharing.

→ Advantages of time-sharing operating systems are as follows:

- Provides the advantages of quick response
- Avoids duplication of software
- Reduces CPU idle time.

→ Disadvantages of time-sharing operating systems are as follows:

- Problem of reliability
- Security and integrity of user programs and data
- Problem of data communication.

Distributed Operating system

Distributed systems use multiple, central processor to serve multiple real-time applications and multiple users. Data processing jobs are distributed among the processors accordingly.

The Processors communicate with one another through various communication lines (such as high-speed buses or telephone lines). These are referred as loosely coupled systems or distributed systems. Processors in a distributed system may vary in size and function. These processors are referred as sites, nodes, computers and soon.

The advantages of distributed system are as follows

- With resource sharing facility, a user at one site may be able to use the resources available at another.
- Speed up the exchange of data with one another via electronic mail.
- If one site fails in a distributed system, the remaining sites can potentially continue operating.
- Better service to the customers
- Reduction of the load on the host computer
- Reduction of delays in data processing.

Network Operating System

A network operating systems runs on a server and provides the server the capability to manage data, users, groups, security, applications and other networking functions. The primary purpose of the network operating system is to allow shared file and printed access among multiple computers in a network, typically a local area network

LAN a private network or to other networks.

The advantages of network operating system are as follows:

- Centralized servers are highly stable.
- Security to server managed.
- Upgrades to new technologies and hardware can be easily integrated into the system.
- Remote access to servers is possible from different locations and types of systems.

The disadvantages of network operating system are as follows:

- High cost of buying and running a server
- Dependency on a central location for most operations
- Regular maintenance and updates are required.

Real time ~~sharing~~ Operating system

A Real time system is defined as a data processing system in which the time interval required to process and respond to an input and display of required updated information is termed as the response time.

There are two types of real time operating systems:

Hard real time systems guarantee that critical tasks complete on time. In hard real time systems, secondary storage is limited or missing and the data is stored in ROM.

Soft real time systems

Soft real time system are less restrictive. A critical real time task gets priority over other tasks and retain the priority until it completes. Soft-real time systems have limited utility than hard real time systems. For example :- Multimedia, Virtual reality

Mobile Operating system

Mobile operating system (OS) as the name itself indicates is used for operating hand held cellphones, tiny electronic gadgets like smartphones, Digital tabs or Notepads (PDAS) or touch pads.

- These mobiles OS software are proprietary [company ownership] in nature and are light and smart (climb to run utility softwares, applications software, mobileapps (applications) etc., on cell (mobile) phones of companies of different make
- Mobile OS of different make and versions vary and become popular by extending good number of utility services to an average endusers.
- Here response time (i.e., how fast while giving / delivering services) how many ~~tipps~~ apps it supports, speedy data display at the push of a button (i.e, finger touch, storage capacity, virtual store, connectivity or link to other cell phones with high speed response, database supports etc, are the few parameters to determine the efficiency to the corresponding mobile OS.

Examples of mobile OS include

- Symbian OS -(on Nokia)
- iphone OS (iOS) -(from apple)
- RIM's black berry OS
- Bada (SAMSUNG) OS
- Android OS
- Web OS (palm/ tip)
- Windows mobile OS (from microsoft)
- Palm OS [for PDAS)

The OS service

A typically operating system contains number of management routines and they could vary from one OS make to another. Thus the operating system services can be summarised as follows: (OS management activities summary)

① Program execution: (the Runtime environment)

→ Accomplish the task of loading a program into main memory partitions.

→ And initiate program execution

→ Provide for normal termination of program after successful execution

→ Provide for displaying error messages in case of abnormal termination (aborting a program).

② I/O operations

→ Accomplish the task of device allocation and control I/O devices.

→ Provide for notifying device errors, devices status etc

③ File system manipulation: (File Handling)

→ Accomplish the task of opening a file, choosing a file

→ Provide for creating a file, deleting a file

→ Allow file manipulation such as reading a file, writing a file, Applying a file etc

④ Communications

→ Accomplish the task of inter-process communications either on the same computer system or between different computer system on a computer network.

→ Provide for message passing and shared memory access in safe mode.

⑤ Accounting

→ Accomplish the task of record keeping the system usage (the resources being used) by how many users and for how long (the duration) for the billing and accounting purpose.

→ Maintain the logs of system activities for performance analysis and error recovery.

→ keep track of history of user logins

⑥ Error detection

→ Accomplish the task of error detection and recovery if any for instance paper jam or out of paper in a printer

→ Keep track status of CPU, memory, I/O devices, storage, devices, file system networking etc.

→ Abort execution in case of fatal errors such as RAM parity error, power fluctuation, if any.

→ Report and for deliver error messages in case of arithmetic overflow, divide by zero errors, etc.

⑦ Resource Allocations

→ Accomplish the task of resource allocation to multiple jobs.

→ Reclaim the allocated resources after their use as and when the job terminates.

→ Provides for resources allocation and scheduling policies (such as round robin, shortest job first,

polling and hand shaking, CPU bound, I/O bound for prioritising jobs for resource allocation and dtaining them for execution).

⑧ Protecting the system

→ Accomplish the task of protecting the system resources against malicious use

→ Provide for safe computing by employing security scheme against unauthorized access / users.

→ Authenticate legitimate (rights) users with logic passwords and registrations.

→ To conclude, the operating must contain sufficient service routines to support ~~all~~ an average user of a computer system with an user Interface that serve as a front end (for accessing various system resources and valuable services). At the same time the OS should not compromise the system protection and security against malicious access by imposters or conspirators (malwares). Not needed

UNIT-5
INTRODUCTION TO COMPUTER PROGRAMS

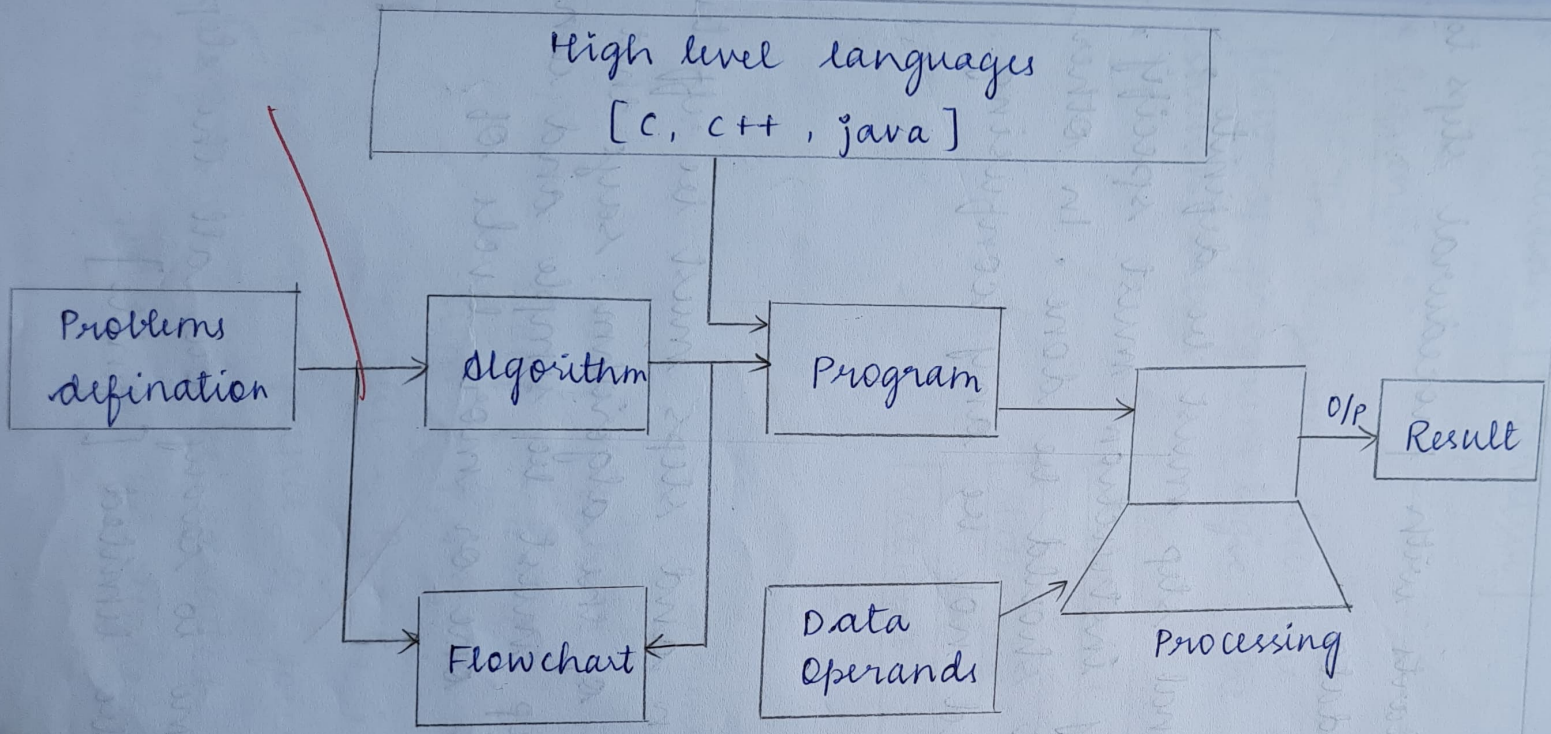
Introduction :

Basic structure of a computer and ~~even humans~~ Von Neuman's stored program concept revealed that the computer has a storage unit called memory. The memory holds set of instructions called Programs, that to be executed and a set of data to be processed.

Problem solving with computer involve preparing a set of procedural instructions for determining the solution using these instructions, The computer can solve the problem to compute the results. Therefore computer programming is consend with designing a set of ~~instru~~ instruct- and expressing it using a precise notation known as computer program. Program develop -ment can be better understood and program logic can be convinced with the introduction of algorithms and flowcharts before writing the programs.

Developing a program

A programming is an art as well as science. A computer professional who knows ~~it~~ the compu- programming like C, C++, java etc can design,



The above following figure illustrates, how to develop a computer program.

analize, write and modify a computer program. The task of writting program code together with testing, debugging and implementing it in an given computer system is called program development. - ment and the content is referred as computer programming. A person who develops a computer programs is known as computer programmer or software developer. Such programmers can develop two major categories of programs i.e, system programs and application programs.

Algorithms:

An algorithm can be defined as a finite number of well defined procedural steps used to solve the problems

Characteristics of Algorithm

A well defined algorithm has five (5) characteristics.

*. **Input**

Algorithm starts with procedural steps to accept the input data.

* **Definite**

Each operational step must be definite, i.e each and every instruction must specify, clearly that, what should be done. In other words there should not be any confusing instruction.

* **Effective**

Each operation and steps must be effective. Effectiveness property of the algorithm says that each operational step must be simple and can be carried out at one or more levels of complexity.

* **Terminate**

The total time to carry out all the steps in algorithm must be limited [finite].

Output

An Algorithm is written to solve the problem, therefore it must produce one or more computer results called output.

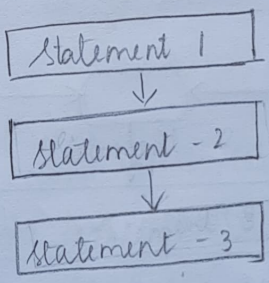
Basic Control Structures in Algorithms.

There are three structures in Algorithms

* Sequencing
* Selection [Branching]
* Repeatation [Interative]

* Sequencing

The sequence structure indicates the sequencial flow of program logic. The sequencing can be shown below



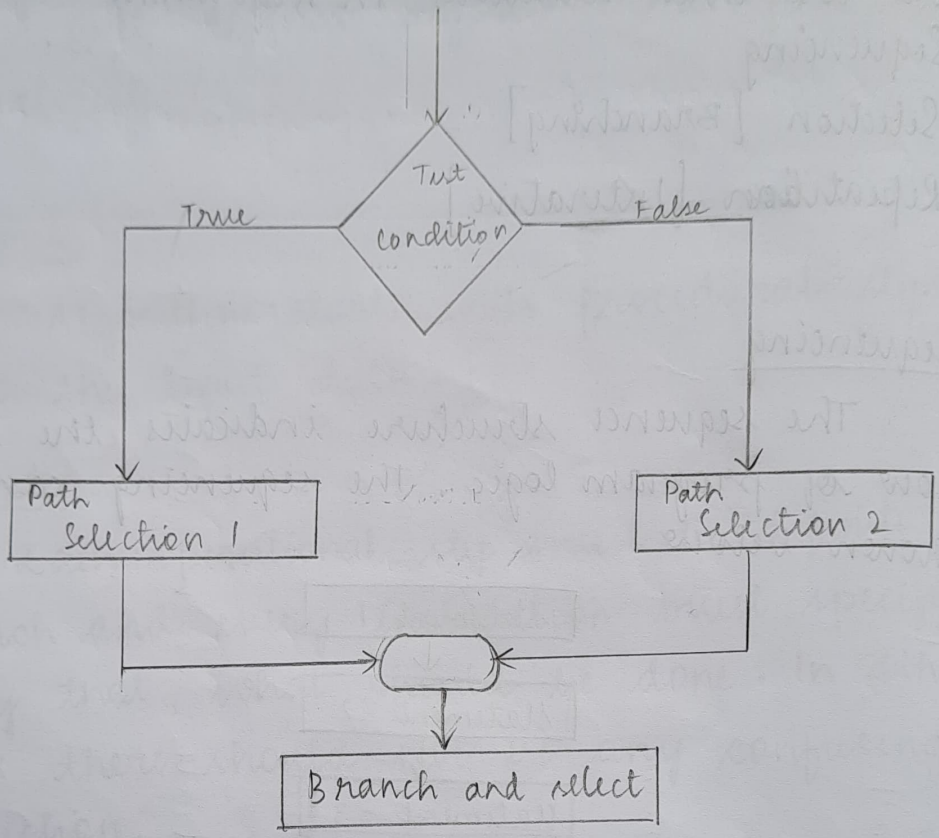
n statements

```
Printf ("Enter two nos.");
Scanf ("%d %d", P1 & P2);
Sum = P1 + P2 ;
Product = P1 * P2 ;
```

A group or sequence of four statements in a 'C' program will be executed in series one after the other. This is called sequencing.

* Selection

It defines conditional program flow. If-then-else statement indicates branching. The path of execution is selected based on whether the condition is TRUE or FALSE. The selection or branching is shown in the below flowchart



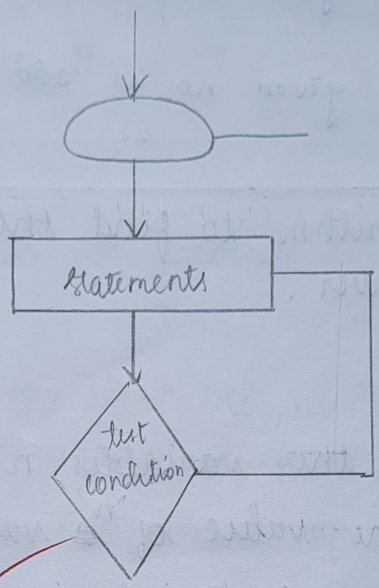
If (num$_1$ > num$_2$)
then
Print "First number is biggest"
else
Print "Second number is biggest"

* Repeatition

The do while statement indicates repeatition or looping. Repeatition means execution of group of statements 'n' number of times. It takes place as long as the loop condition is true.

when the condition becomes false the loop or repeatition will be terminated. This is shown below.

```
do
{
Pf ("
}
while (count <00);
```



As long as the count greater than '0' is true the loop statements will be executed repeatedly. When the count is greater than '0' is false the loop will be terminated.

Examples of Algorithm

Write an algorithm for swaping contents of two variables.

Step 1 :- Start

Step 2 :- Declare the variables a, b, temp

Step 3 :- Read the values a, b

Step 4 :- temp = a
a = b
b = temp

Step 5 :- Print the swapped values

Step 6 :- Stop

Write an algorithm for whether the given no. is odd or even.

Step 1 :- Start

Step 2 :- Read num.

Step 3 :- Check whether the given no. is odd or even

Step 4 :- If the given no. is even then print even

Step 5 :- If the given no. is odd then print odd.

Write an algorithm to find the factorial of a given number.

Step 1 :- Start

Step 2 :- Declare the variables n, i, fact

Step 3 :- Read the value of 'n' variable

Step 4 :- Initialize factorial to 1

Step 5 :- Calculate the factorial by successive multiplications

fact = fact * i ;

Step 6 :- Print the factorial of a number

Step 7 :- Stop

Write an algorithm for adding two 2 numbers

Step 1 :- Start

Step 2 :- Declare variable num 1, num 2 and sum

Step 3 :- Read values num 1 and num 2

Step 4 :- Add num 1, num 2 and assign the result to sum

Step 5 :- Print the sum

Step 6 :- Stop

Write an algorithm for multiplying two numbers

Step 1 :- Algorithm to multiply two numbers [Start]
Step 2 :- Declare variable num1, num2 and product
Step 3 :- Read values num1, num2
Step 4 :- Multiply num1, num2 and asign the result to product
Step 5 :- Print the Product
Step 6 :- Stop

Write an algorithm for subtracting of two numbers.

Step 1 :- Start
Step 2 :- Declare the variable, num1, num2, Result
Step 3 :- Read the values of num1, num2
Step 4 :- Find the difference of two numbers.
Step 5 :- Display the result
Step 6 :- Stop

Write an algorithm to find the biggest of two nos.

Step 1 :- Start
Step 2 :- Declare variable num1 and num2
Step 3 :- Read the values num1 and num2
Step 4 :- Check whether the given no. is greater or smaller
Step 5 :- If num1 is greater than num2, Print num1 is greater
Step 6 :- Otherwise print num2 is greater.

Write an algorithm to find the greatest of three numbers.

~~Star~~

Step 1 :- Start
Step 2 :- Declare variables a, b, c, max
Step 3 :- Input a, b, c
Step 4 :- Let max = a
Step 5 :- If b > max, then Max is B
Step 6 :- If C > Max, then Max is C
Step 7 :- Display the largest value
Step 8 :- Stop.

Write an algorithm to print even numbers from 1 to 100

Step 1 :- Start
Step 2 :- Declare variable i
Step 3 :- Read numbers 1 to 100
Step 4 :- for i = 1 to 100
Step 5 :- if i % 2 == 0
Step 6 :- Print i
Step 7 :- Stop

Write an algorithm to print sum of n numbers

Step 1 :- Start
Step 2 :- Declare variable i, n, sum, a[i]
Step 3 :- Intialize sum = 0
Step 4 :- Read n
Step 5 :- for i=0 ; i<n ; i++
Read a[i]
Step 6 :- for i = 0 to i<n
sum = sum + a[i]

Step 7 :- Print the sum of n numbers

Step 8 :- Stop.

Write an algorithm to find the smallest of three numbers

Step 1 :- Start

Step 2 :- Declare variables a, b, c, min

Step 3 :- Input a, b, c

Step 4 :- Let min = a

Step 5 :- If b ≤ min, then min is B

Step 6 :- If C < min, then min is C

Step 7 :- Display the smallest value

Step 8 :- Stop.

Flow Chart

A flow chart can be defined as a pictorial representation of an algorithm.

Characteristics of flowchart

* Diagrametic representation using standard conventions / notations or symbols.
* Depicting flow of control via arrow marks (symbols)
* Presentation of program logic.
* Each symbol in a flowchart must represent an activity, such as input and output of data,
* processing of data, decision making etc

Advantages of flowchart

The advantages of flowcharts are

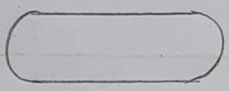
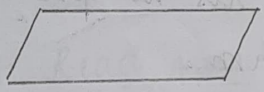
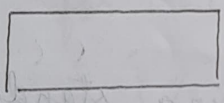
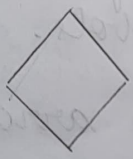
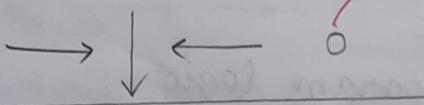
* Easier to understand the diagrametic flow of operations of an algorithm.

* Assist in verifying the algorithm and the program.

* Assists in error detection and correction both in algorithm and program writing.

* It is easier to discuss and share the ideas on problem solving methods and steps while explaining it to others.

Basic symbols used in drawing flowchart.

Symbols	Uses
	Start / Stop
	Input / Output
	Process / calculation
	Decision making
	connectivity

Limitations in flowchart

The drawbacks or disadvantages of flowcharts are as follows:

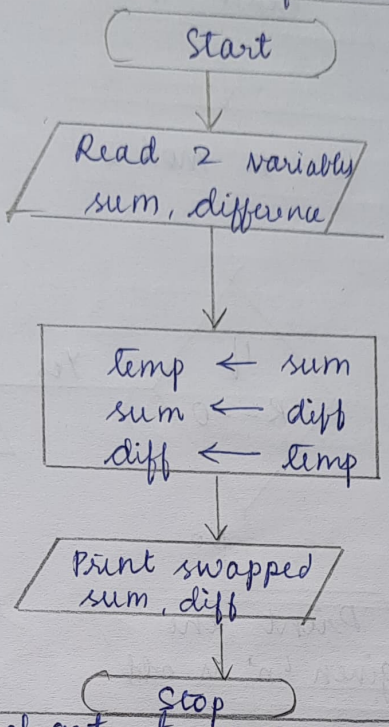
* Flowchart appear to be complicated especially in large complex problems.

* Flowcharting tends to be costly in terms of time and effort spend for drawing the large

flowchart.

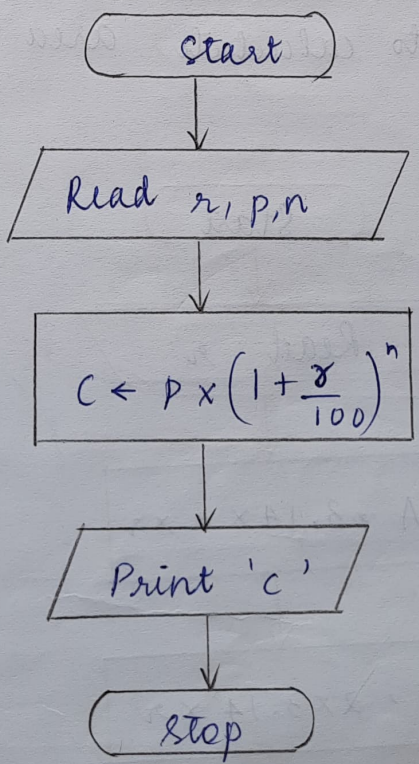
* Flowcharts are difficult to modify once they are drafted.
* Flowchart periodical updation is impossible.

Write a flowchart to swap contents of two variables.

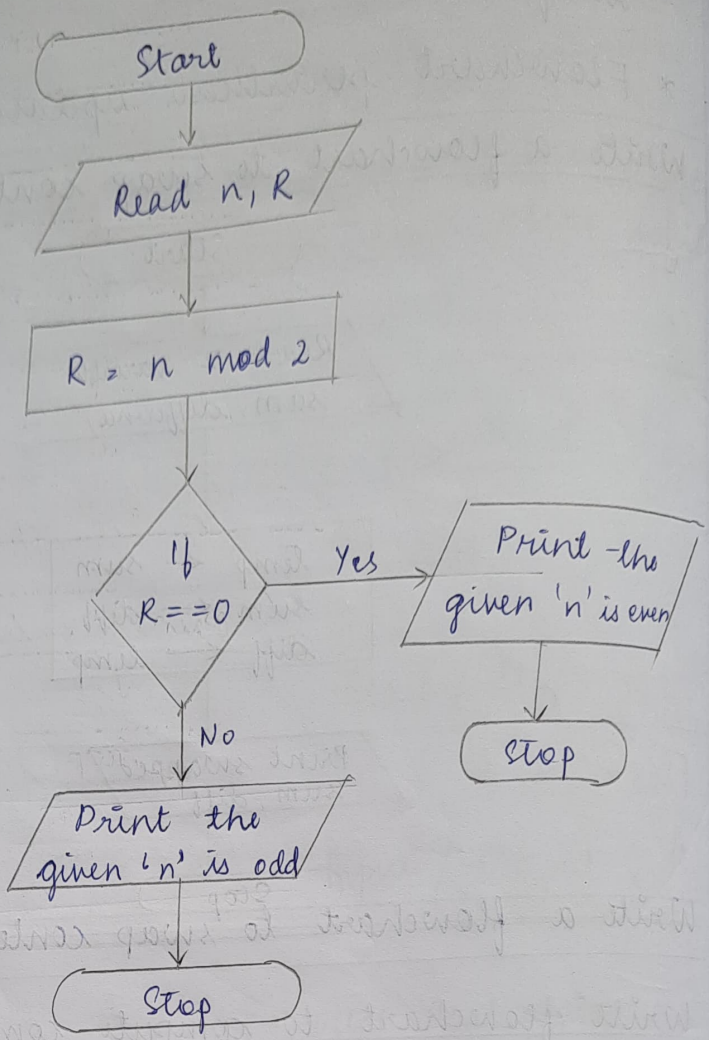


~~Write a flowchart to swap contents of two variables.~~

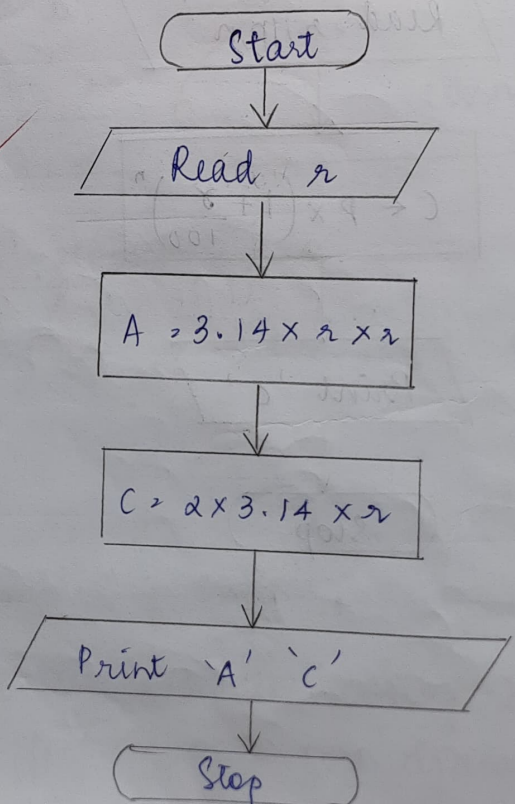
Write flowchart to compute compound interest.



Write a flowchart for whether the given no. is odd or even.



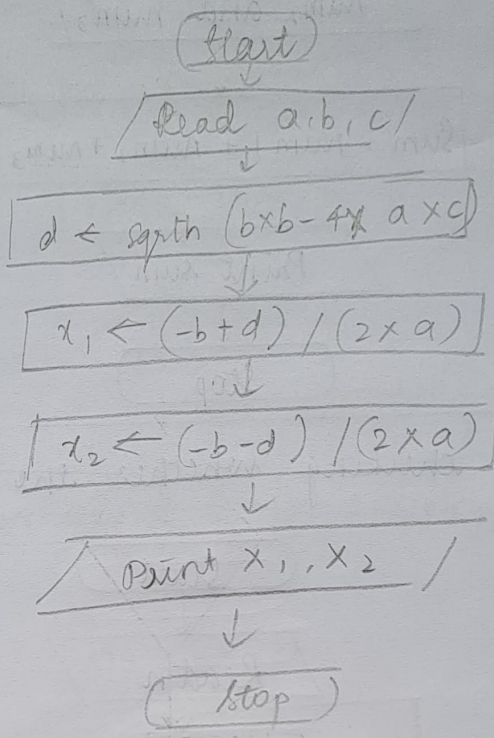
Write a flowchart to calculate area and circumferen of a circle.



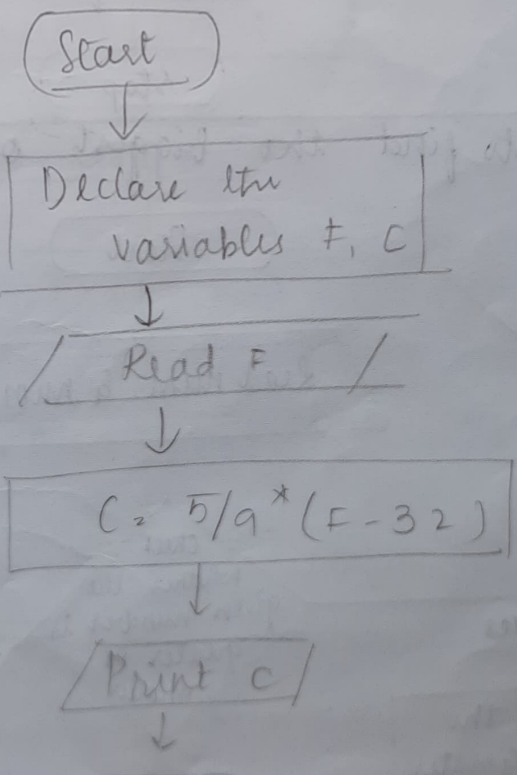
Write the flowchart to compute all possible roots of given quadratic equation $ax^2+bx+c=0$

~~Classification~~

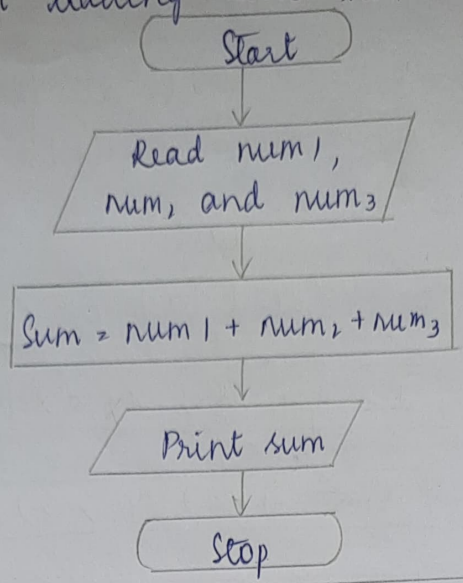
Write the flowchart to compute all possible roots of given quadratic equation $ax^2+bx+c=0$



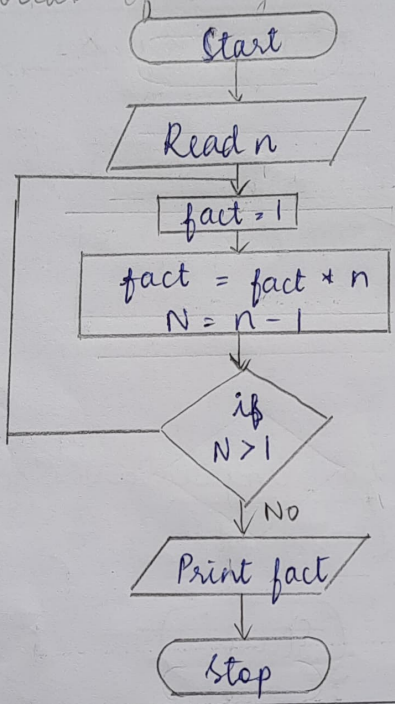
Write flowchart for temperature conversion from farenheit to centigrat



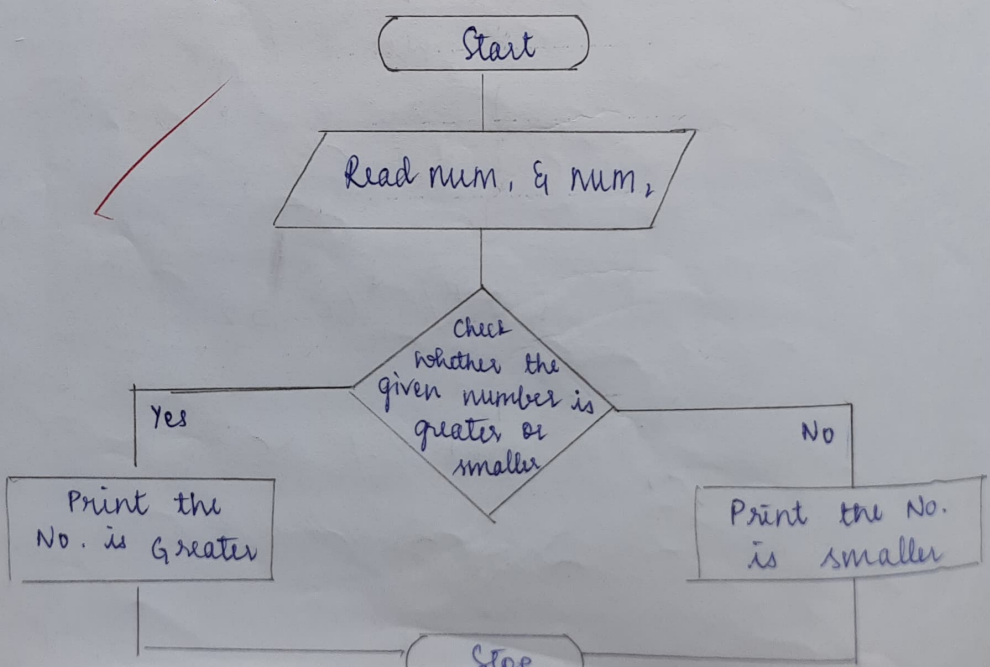
Flowchart for adding three numbers.



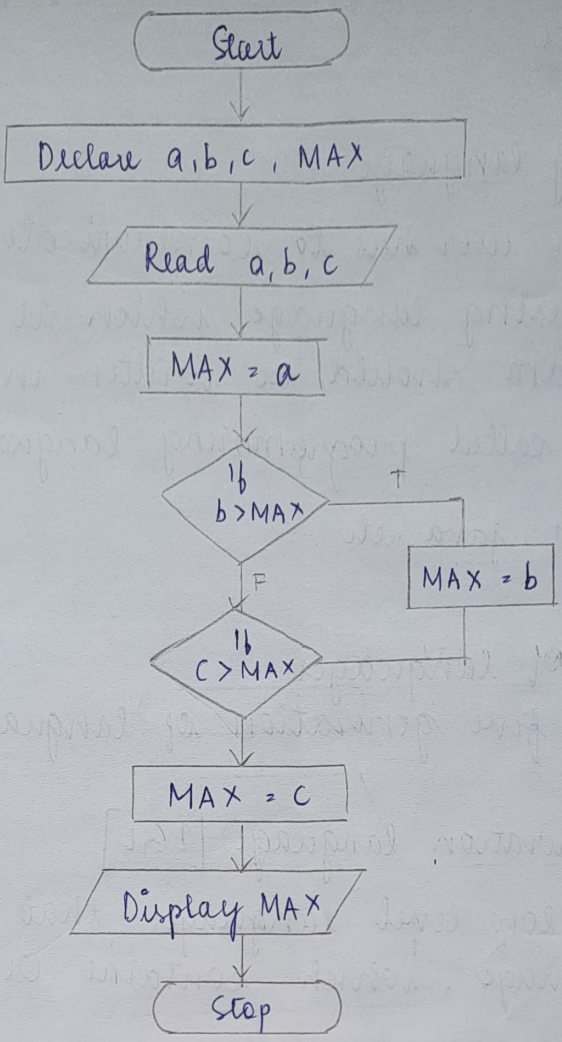
Flowchart for ~~checking whether the given no. is odd or even~~ factorial of a given number.



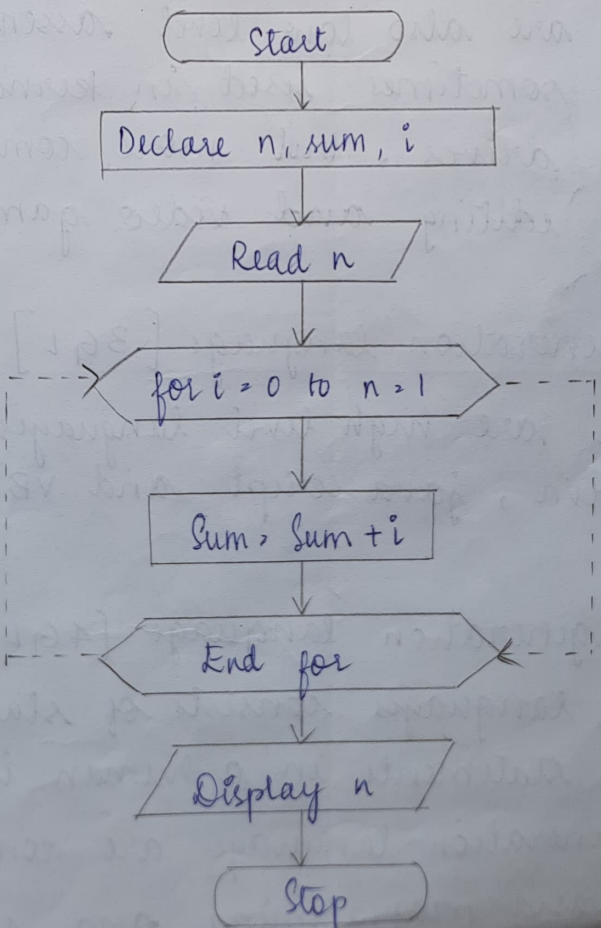
Flowchart to find the biggest of 2 numbers.



Flowchart to find the greatest of 3 numbers.



Flowchart for finding sum of 'n' numbers.



Classification and generation of programming languages

Programming language

The computer user has to communicate with a computer using language which it can understand i.e, program should be written in computer languages called programming language.

Ex:- C, C++, java etc.

Generation of languages

There are five generation of languages:

* First generation language [1GL]

These are low level languages that are machine level language, which contains 0's and 1's.

* Second generation language [2GL]

These are also low level assembly language. They are sometimes used in kernals and hardware drives, but more commonly used in video editing and video games.

* Third generation language [3GL]

These are high level languages such as C, C++, java, java script and VB.

* Fourth generation language [4GL]

These languages consists of statements similar to statements in a human language. Fourth generation language are commonly used in database programming and scripts.

Some of the languages are perl, php, python, ruby and SQL

* Fifth generation language [5GL]

These are the programming languages that contains visual tools to help develop a program. Examples of 5GL are Mercury, OPS5 and Prolog

Types of programming languages

There are three types of programming languages. They are:

* Machine level language

The computer can understand only machine level language only. It is also called as binary language which contains 0's and 1's. In this language, the different instruction to the computer are formed by taking different combinations of 0's and 1's

* Assembly level language:

It uses symbolic instruction and executable machine codes. The meaningful abbrevations are used for machine specific instruction. An assembler is required for translating assembly language instructions to machine language instructions to run a program on a computer. Machine and assembly languages are low level languages.

* High level language

This language consists of set of english like words and symbols the specified rules are to be

followed while writing programs in high level language. This are called source programs. The translators like compilers and interpreters are used to convert into machine language.

ASSEMBLER

Assembler is a translator which translates assembly level language programmes into machine ~~text~~ code.

Functions of assembler

- It translates ~~symbolic~~ mnemonic operations codes to machine codes and corresponding register address to system address.
- It checks the syntax of assembly level language program and generates messages on syntax errors.
- It assembles all the codes in the main memory for the execution
- If the assembly language program is large, it provides linking facilities among the sub routines
- It facilitates the generation of output on required output median.

Advantages and Disadvantages of assembler

Advantage

* easy to understand and use
* Less error prone
* Efficiency is more than higher level language
* More control on hardware components.

Disadvantages

* Machine independent
* It is harder to learn
* Development time is slow
* Less efficient than machine level languages
* No standardisation, no support for modern soft - ware engineering technology.

Difference between compiler and interpreter

Compiler	Interpreter
It translates entire source board into a set of machine language instruction.	It translates high level language statement into a machine code and executes immediatly
The object code is permanently saved and it is used every time and the program is to be executed.	The object code is not saved for future use.
Repeated compilation is not necessary for repeated execution of a program	An instruction is interpreted and translated into machine language every time it is executed.
A compiled program runs much faster than an interpreted program	Interpreters are slower than compilers.

Difference b/w high level languages and low level languages

High level language	Low level language
The high level language are machine Endependent.	The low level language are dependent

High level language	Low level language
High level language are easy to learn	Low level languages are difficult to learn
They are near to human languages	They are far from human languages.
Programs in high level languages are slow in execution.	Program in low level languages are fast in execution.
High level language do not provide much facility at hardware level	Low level language provide much facility at hardware level.
Knowledge of hardware is not required to write programs.	Knowledge of hardware is required to write programs.
Programs in high level languages is easy to modify	Programs in low level languages is difficult to modify.
Examples of high level language is C, C++, java, javascript etc	Examples of low level languages is machine language and assembly language.