SWAMI VIVEKANANDA SCHOOL OF ENGG. & TECH.

MADANPUR, BBSR

LECTURE NOTES

ON

DIGITAL ELECTRONICS

Year & semester: 2^{ND} Year, 3^{RD} Semester

BY- Er. SRIDHARA KUMAR RATH

LECTURER IN DEPARTMENT OF ELECTRONICS & TELECOMMUNICATION ENGINEERING

Number System

① For peprien ting the Information we cell the Number System the digital System.

② Types of Number System.
In digital Computer
used for Kepresanting

① Bainary $(0 — 1)_2$
② Decinal $(10 — 9)_8$
③ Octal $(0 — 7)$
④ Hexa decimal $(0 — 15)_{16}$

ⓐ Bainarey Number System
=> it form only two value (0|1)
That's 0|1 it is also None
of the base 2 Number System

ex => $(10)_2$ $(11)_2$ $(101)_2$

ⓑ Decimal => This Number System
contens 10 digit from (0–9)
ⓤ it is also cold base 10 System

ex => $(90)_{10}$ $(99)_{10}$ $(73)_{10}$

ⓒ Octal Number System this Number System contens 8 digit from (0–7)

(1) It is also cold base 8 System.

Exam => $(75)_8$ $(67)_8$

(ii) octal No

octal No	Binary from
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

(d) Hexa decimal number System =>
It is also none as base 16 Number System it is has 10 digit (0 —— 9) and 6 letters from find Numbers from $(10 —— 15)_{16}$

$(36231)_{16}$ $(2345F)_{16}$

Ex => $(89C945d)_{16}$ $(F20053007003)_{16}$

Hexa Delimal	Binary from
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Hexa Delimal	Binary from
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Binary to Decimal :-

The process Starts from multiplen the Beets of Bainary number with its Corespon ding positnal welds and lastly we add all these produce.

Ex $(10111)_2$

$1\times24+0\times2+1\times2+1\times2+1\times2$

$\Rightarrow$ ~~1~~×16 + 0 + 4 + 2 + 1

$= (23)_{10}$

$(10111011)_2$

$= 1\times2+0\times2+1\times2+1\times2+1\times2+0\times2+1\times2+1\times2$

$= 38+0+32+16+8+0+2+1$

$= 97$

Decemal to octal

in the fost step we pofom the division opponetion on the integen and the susesive quoctient with the base of octal

$(178)_{10} = (B2)_{16}$

$16 \,|\, 178 \quad 2$

$\quad 11$

$(285.625)_{10} = (11D.A)_{16}$

$= 16 \,|\, 285 \quad 13$

$\quad 16 \,|\, 17 \rightarrow 1$

$\quad\quad 1$

octal to Hexa Decimal

* we ite the three dit bynary digit for the given number

* Then my peyars off 4 bet on both side of binary point.

* then we wite the Hexa dicimal diyee which corence pron to his peyar.

$(536)_8 = (55E)_{16}$

$\underbrace{0001}\,\underbrace{0101}\,\underbrace{1110}$

Hexa deimal to other no conversion :-

Hexadelimal to deimal :-

$(2A)_{16} = (\ )_{10}$

$2\times16 + A\times16$

$= 32 + 10\times1$

$= (42)_{10}$

$(BBD)_{16} = (\ \ )_{10}$

$= B\times16^3 + B\times16^1 + D\times16^1$

$= 2816$

① we moltifaly ehes diritof the goven number with its respective poyesinal rew and lasly we ad the produds of all thebreets which its wets

$(3A2F)_{16}$

$\Rightarrow 3 \times 16 + A \times 16^{2} + 2 \times 16^{2-} + 2 \times 16^{-1} F \times 16$

$\Rightarrow 48 + 10 + 2n \frac{1}{16} + 3 \times 16^{1} + A \times 16^{0} + 2 \times 16 + F \times 16^{2}$

$\Rightarrow 48 + 10 + 1 + 2 \times (-16) + 15 \; (\times 256)$

$\Rightarrow 48 + 10 + 2 \times \frac{1}{16} + 15 \times \frac{1}{256}$

$= 58 + 0.125 + 0.9375$

$= (58 + 0.18359375)_{10}$

$= (58.1836)_{18}$

Hexa decimal to Binary:—

① $(2F\;9A)_{16} = (\;)_{2}$

$\Rightarrow (0010111110011010)_{2}$

⑪ $(AF6.30)_{16} = (\;)_{2}$

$\Rightarrow (101011110110.00111101)_{2}$

* The proses of convoting Hexa denimal of Binary is the revors prosses of Binary to Hexa denimal we write the 4 Bit Binary repont of is Hexa denimal number dizid.

① $(7AF)_{16} = ( )_2$

$\Rightarrow (0111\,1010\,1111)_2$

② $(FA9)_{16} = ( )_2$

$(1111\,1010\,1001)_2$

Hexa decimal to octol :-

The process of converting hexadecimal to octal each the reverse octal to hexadecimal we write the 4 bites octal code of hexa decimal.

ex

$(52A71)_{15} = ( )_8$.

$\underbrace{001}\,\overbrace{010}\,\underbrace{010}\,\overbrace{101}\,\underbrace{011}\,\overbrace{110}\,\underbrace{001}$

$(1225361)_8$

Arithmetic Operation :-

- Addition
- Substraction
- multiplication
- division

Binary Addition rule :-

A	B	Sum	Carry	result
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	0	1	10

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 10

1 + 1 + 1 = 11

* addition is carried out just like delimas by adding of the coloums. Starting at the right and working coloumn by coloum.

Binary Substruction rule:-

0 - 0 = 0
0 - 1 = 1
1 - 0 = 1
1 - 1 = 0

A	B	difference	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Substast coloumn by coloum always Start L.S.B (List Significant Bit) from coloumm of.

(ii) If necessery Borrow from the next higher.

(iii) when one is Substuct created from the next most Significant

Binary multiplication:-

it is similar to delimal multiplication rules for Binary multiplication:-

0 x 0 = 0
0 x 1 = 0
0 x 0 = 0
1 x 1 = 1

Binary multcplication

Binary Division

0 ÷ 0 = 0

1 ÷ 1 = 1

```
100 | 1100    | 11
    | 100     |
    -----------
      100
      100
      ----
        0

1001 | 1110101 | 1101
     | 1001    |
     -------------
       11011
        1001
       --------
        100
          0
       --------
        1001
        1001
       ------
          0
```

Binery to Hexa decimad

$$(\underbrace{0101}_{5}\underbrace{1010}_{A}\underbrace{1011}_{B} . \underbrace{0011}_{3})$$

- In the first Step we have to make the pairs of a bets on bothe side of the binery point.
- If there will be one two on there bits left in a pair of 4 bits pair.
- We add the required number of 0 of an any sides.
- In the seceed Step we write the Hexa decimal corresponding each pair.

$$(\underset{A}{1010} . \underset{3}{0011})$$

$$(\underset{3}{0011}\underset{9}{1001} . \underset{2}{0010})$$

$$(\underset{2}{0010}\underset{9}{1001}\underset{A}{1010}\underset{F}{1111})$$

05.11.21

1's Complement & 2s Complement:-

111001

↓↓↓↓↓↓

000110 → 1's comprement

+1

000111 → 2's comprnment

x = 111001

000110

1s comprennt of a binenno is an andan by Co toggle bets in that is trastan neing the 0 b2 1 and 1 b2 0

2s comprment

y = 10010

01101

+1

10010

if we add 1,2 one s comeplemet of binary then the ragetting nomber we wil 2s complement.

Binary Substraation using 1S compement:-

(1010) (1111)

$$\begin{array}{r} 1111 \\ (+)\ 0101 \\ \hline 10100 \\ +\ 1 \\ \hline 0101 \end{array}$$ end eround cany

* conuet the namben towo be Substned one s complet form
* add both the numben
* remove the cany and add ět ~~too~~ the regelt

Subtned

$(1010)_2$

$(1111)_2$

~~1010~~
~~1111~~

0101

$$\begin{array}{r} 1111 \\ 0101 \\ \hline 10100 \\ +\ 1 \\ \hline 0101 \end{array}$$

* 1S complent Sabtracasan is a mated Q to Subtrte (this mated of 2 binary by adison

Binary Sumbstraltion using 2's complement :-

Sumbstraltion falar number form langes number

$(1010)_2$ from $(1111)_2$

```
  1111
  0110
 ------
 10101
omit
  0101 → Result
```

Step 1 - ~~Diff~~ Detormind the 2S complem of So

* add this namben add the lajar namben

* omit the cary

Sumbstraltion of lazar number ~~of~~ for skalar number

$(1010)_2$ $(1000)_2$

```
                              1110
                              0001
                            --------
                           (-)0010 →
 0101          1000
  +1           0110
 -----        ------
 0110          1110
               ~~+1~~
              ------
               0001
```

2 subtract ofonm

* detatmin the 2S complement of lajan number
* add is number form Sumlen number
* theris no Cany in this Core the reget in 2S complent form is nagatev.
* 2 get Answen in ~~frow~~ take 2S Complmet and sign.

Code

Binary code and his Appliction

* the grop of simble is cald hes code
* the degetal is repagetcn and ~~f~~recmetad has grop has bets thes gonp of has Binary code.

* Binary codes canbe classfide ~~weght~~

-> weghted ~~codes~~
-> Non-weghted

weghted code

* this code are those binary code which cobad of pogesan of presepupy. is Pogesan of noember repeesan

Savanal sistem of the code are us to expens the daseral deget 0 to 9 in thes code ~~an~~ is dasemal deget a grop of 4 bites non-weighted code

in this side Binary conds the pogesan has not assing

ex

xx 3 code

gray code

bcd code

22.11.2021

Boolean Algebra

→ Swychting sercite and all so leyeck cenelte
Geat cencite and degetal sencikot

Suychting Algebra and all so BoolRam Algebra
Bolean Algebra is a system is matmatcal
logick ~~it~~ ~~is~~ Algebrace consesof the set
of element (0,1), towo bohery operetas cold
OR and AND unary opeten cold Not.

it is the be tool in the
anlysis AND synthesis of Sanching Sancite.
it is a expnece lac Algebnally

eny complen logick can he expnecS by a boolean
fanson.
The boolean Algebra is goven by Centin
well develored reals and lege.

anioms and Lows BoOlean Algebra

Aniom8 : Of postulates of boolean Algebra

set of lagical expresaty that ~~scept~~ Accep wethut prove. and Openator which we can beyed a set of usefull theorem. actually Anioms are nothing more dhan a ~~daterasa~~ definitions of the 3 baseic logick opera AND, ~~A~~OR, NOT, NAND inventen essn eagen Can be interpreted in the outcome of fun operacen pafanm by a ~~lazick by a~~ logick gate.

AND Operation

(Aniom 1) 0.0 = 0

(Anion 2) 0.1 = 0

(Aniom 3) 1.0 = 0

(Aniom 4) 1.1 = 1

OR Operation

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 1

NOT operation   Bar

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

1. complementation Lows

The complement seeli mens a invalb that is ~~O~~ to changes 0s to 1s ~~O~~ the 5 lows of complemention

Law1 ! $\bar{0} = 1$

Law2 ! $\bar{1} = 0$

Law3 : If A = 0, then $\bar{A} = 1$

Law4 ! If A = 0 , then $\bar{A} = 0$

Law5 ! $\bar{\bar{A}} = 0$ (double complementation low)

OR Lows

~~The fouse~~ The 4 loge

Law 1 ! $A + 0 = A$ (Null Law)
Law 2 ! $A + 1 = 1$ (Identity Law)
Law 3: $A + A = A$
Law 4: $A + \bar{A} = 1$

AND Lows

The 4 AND are 'As follows

Law1: $A \cdot 0 = 0$ (Null Law)

Law2: $A \cdot 1 = 1$ (Identity law)

Law3: $A \cdot A = A$

Law4: $A \cdot \bar{A} = 0$

Commutative law

Associative law

Law 1 $A + B = B + A$

Law 2 $A \cdot B = B \cdot A$

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

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

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

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

$$\text{R.H.S} = (A+B)(A+C) = A(A+C) + B \cdot (A+C)$$

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

$$= A + AC + AB + BC$$

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

$$= A \cdot 1 + BC$$

$$= A + BC$$

23.11.2021

The desebutcb law Lataren ~~of~~ an mateplay out of expresan

Law 1 = $a \times (b+c) = AB + AC$

A	B	C	B+C	A·(B+C)
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	0	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

Law 2 = $A + B \cdot C = A + B \cdot A + C$

Demorgans teoram

Low - 1 = $\overline{A+B} = \overline{A} \cdot \overline{B}$

Law 2 = $\overline{A \cdot B} = \overline{A} + \overline{B}$

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

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

Duality:

The impleecaty of the dualet can sapat ones of ~~co~~ stament the dual all so thus all so proved.

this is called of principle of Duality

$$\left[f(A, B, C, \cdots 0, 1, +, -)\right]_d = f(A, B, C, \cdots, 1, 0, \cdot, +)$$

relasen betuen complement of dual.

* ~~fe~~ ① $f_c(A, B, C \cdots) = \overline{f(A, B, C, \cdots)}$

$= f_d(\overline{A}, \overline{B}, \overline{C}, \cdots)$

* ② $f_d(A, B, C \cdots) = f(\overline{A}, \overline{B}, \overline{C} \cdots)$

$= f_c(\overline{A}, \overline{B}, \overline{C}, \cdots)$

*① the fast relasan Stast that of a complement of a fansan $f(A, B, C..)$ can be actad by complmenteing on the vareabl of the dual fansan $f(A, B, C, \dots)$

*② The secend relasan Stast that the dual can be actade all the litarcals in ~~by~~ $\overline{f(A\ B\ C\ \dots)}$

Duals

① $\bar{0} = 1$ $\bar{1} = 0$

× ~~② 1+0 = 1 0+1 = x~~

② $0 \cdot 1 = 0$ $1 + 0 = 1$

③ $0 \times 0 = 0$ $1 + 1 = 1$

④ $1 \times 1 = 0$ — $0 + 0 = 1$

⑤ $A \cdot 0 = 0$ — $\bar{A} + 1 = 1$

⑥ $A \cdot 1 = A$ — $\bar{A} + 0 = \bar{A}$

⑦ $A \cdot A = A$ — $\bar{A} + \bar{A} = \bar{A}$

⑧ $A \cdot \bar{A} = 0$ — $\bar{A} + A = 1$

⑨ $A \cdot B = B \cdot A$ — $\bar{A} + \bar{B} = \bar{B} + \bar{A}$

⑩ $A \cdot (B \cdot C) = (A \cdot B) \cdot C$ — $\bar{A} + (\bar{B} + \bar{C}) = (\bar{A} + \bar{B}) + \bar{C}$

⑪ $A \cdot (B + C) = (A * B) + C$ — $\bar{A} + (\bar{B} \cdot \bar{C}) = (\bar{A}\bar{B}) \cdot \bar{C}$

⑫ $A \cdot (A + B) = A$ $\longrightarrow$ $\bar{A} + (\bar{A} \cdot \bar{B}) = \bar{A}$

⑬ $A \cdot (A \cdot B) = A \cdot B$ — $\bar{A} + (\bar{A} + \bar{B}) = \bar{A} + \bar{B}$

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

⑮ $(A + B)(\bar{A} + C)(B + C) = (A + B)(\bar{A} + C)$

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

(2) $0 \cdot 1 = 0 \Rightarrow 1 + 0 = 1$

(3) $0 \times 0 = 0 \Rightarrow 1 + 1 = 1$

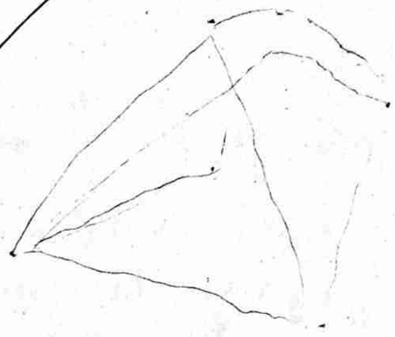
(4) $1 \times 1 = 0 \Rightarrow 0 + 0 = 1$

(5) $A \times 0 = 0 \Rightarrow \bar{A} + 1 = 1$

(6) $A \times 1 = A \Rightarrow \bar{A} + 0 = \bar{A}$

(7) $A \times A = A \Rightarrow \bar{A} + \bar{A} = \bar{A}$

wrong



(8) $A \cdot \bar{A} = 0 \Rightarrow \bar{A} + A = 1$

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

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

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

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

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

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

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

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

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

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

(18) $A + C \cdot C + D = AC + AD + BC + BD$

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

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

$\Rightarrow \bar{A} \cdot \bar{B} = (\overline{AD}) \cdot A\bar{B} \cdot \overline{\overline{AB}}$

(20) $\overline{AB} + \bar{A} + AB = 0$

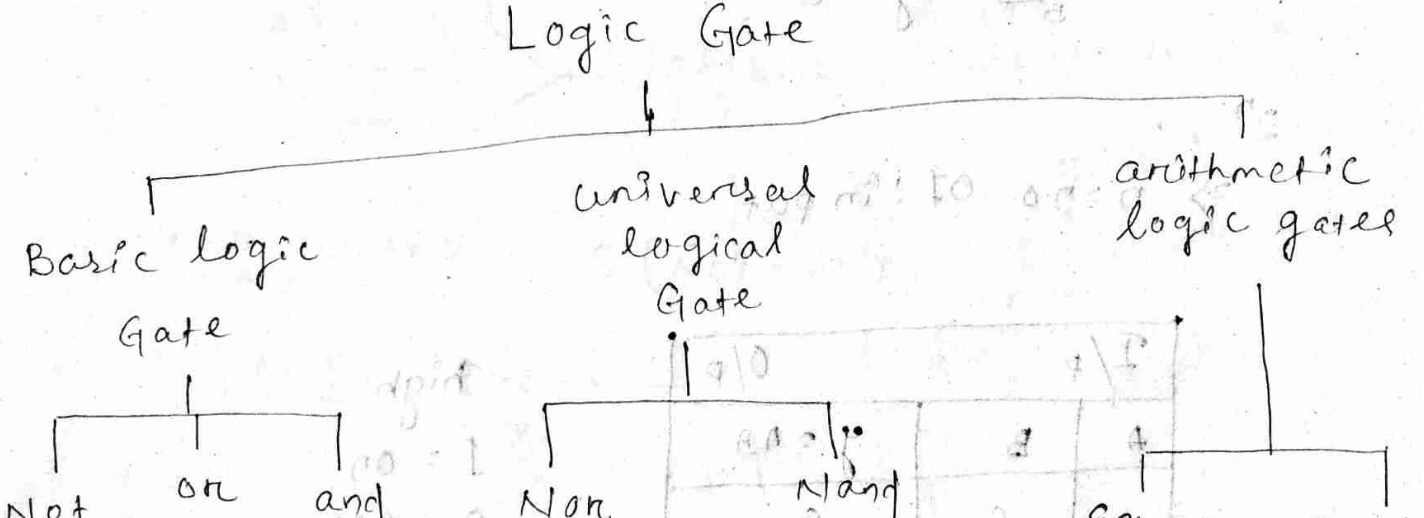
$\Rightarrow \overline{\overline{AB}} \cdot A \cdot \overline{AB} = 1$

Logic gate:-

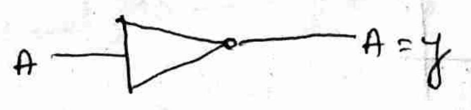
(I) Logic gates are basic bilding blocks of any digital System it is an electronic Sercite have being one or more than in put and ony 1 out put.

(II) The relationship between the input and the output is based on shorturen logic Based on this logic gutes are Named as.

- -> Not gate
- -> or gate
- -> And gate
- -> Nor gate
- -> Ex-or gate
- -> Ex- Nor gate



not Gate



Treuth Table

Not Gate (Inverter)

I/P	O/P
A	$y = \bar{A}$
0	1
1	0

$A = \bar{A}$

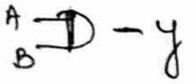
$0 = \bar{0} = P$

$L = T = 0$

And Gate

And Gate

A Sarcite wich potam and aperesan it has ($n > 2$) and I output



AB

2^n

=> n = no . of ! in put

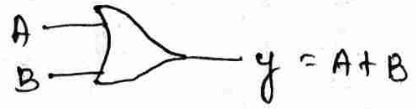
I/P		O/P
A	B	$y = AB$
0	0	0
0	1	0
1	0	0
1	1	1

high
1 = on
0 = OFF
low

Or Gate

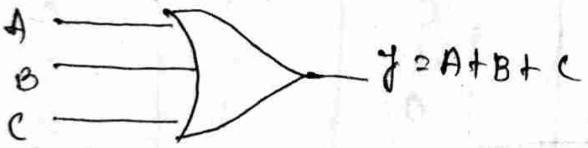
A Sarkite wich an are opersan it has (n>2) and uarm out

Or Logic Dig

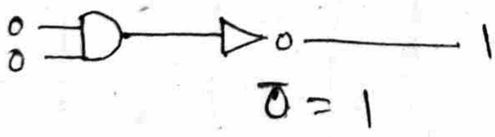


Truth table

I/P		O/P
A	B	y = A+B
0	0	0
0	1	1
1	0	1
1	1	1

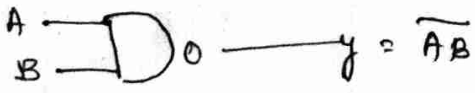


NAND Gate:- S (Not - AND)

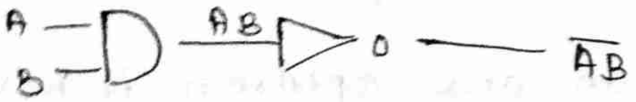


$\overline{0} = 1$

NAND Gate:-

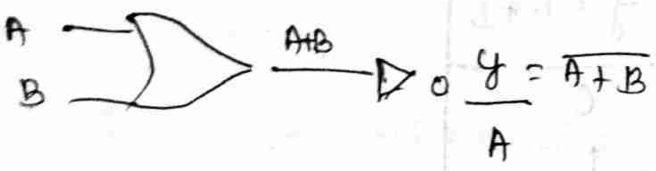


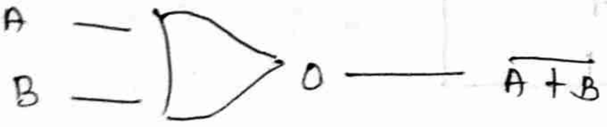
A (Not - And) oprecun non has nand it has (N ≥ 2) and 1



i/p		o/p
A	B	$y = \overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

NoR Gate (Not - OR)



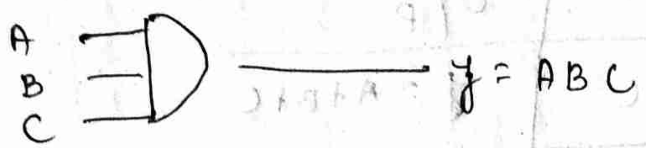


Truth table

i/p		o/p
A	B	$y = \overline{AB}$
0	0	1
0	1	0
1	0	0
1	1	0

I/P		O/P
A	B	$y = \overline{AB}$
0	0	1
0	1	0
1	0	0
1	1	0
0	0	0
0	1	0
1	0	0
1	1	0

And Get

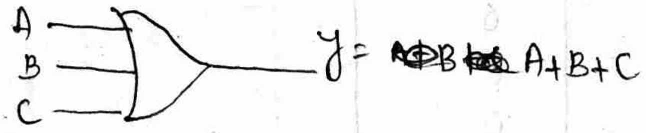


$2^3 = 8$

I/P			O/P
A	B	C	y = ABC
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
0 1	0	0	0
0 1	0	1	0
1	1	0	0
1	1	1	0
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

or Gaete

OR Gate

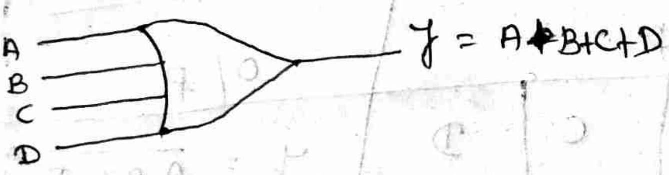


y = ~~A⊕B⊕C~~ A+B+C

Truth table

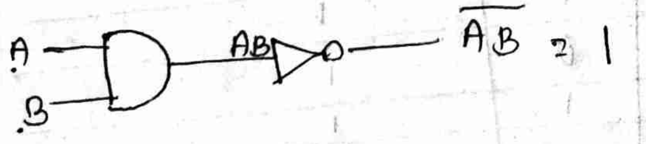
I/P			O/P
A	B	C	y = A+B+C
0	0	1	1
0	0	0	0
0	1	1	1
0	1	0	1
1	0	1	1
1	0	0	1
1	1	1	1
1	1	0	1

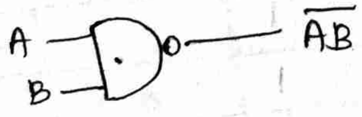
I/p				O/p
A	B	C	D	$y = A+B+C+D$
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1



NAND Gate:-

(Not And)

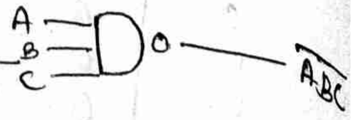




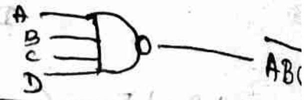
Truth table

I/p		O/p
A	B	$y = \overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

I/p			O/p
A	B	C	$y = \overline{ABC}$
0	0	0	1
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	0



T/p				O/p
A	B	C	D	$y = \overline{ABCD}$
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0



~~NOR Gate~~

x-OR Gate

x-OR Gate

Ex-OR Gate

X-OR Gate

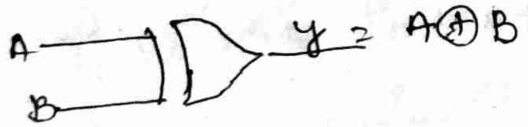
n i/p (n ≥ 2)

$Y = A \text{ XOR } B \text{ XOR } \ldots\ldots n$

$= A \oplus B \oplus C \ldots\ldots$

$AB + \overline{AB}$

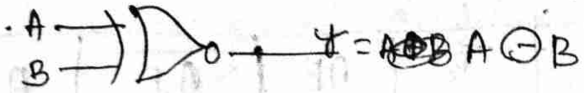
Logic Dig



~~Top~~ TT for X-OR Gate

I/p		O/p
A	B	$y = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Ex - NOR Gate!-



16.11.21

De - morgan's theorem!-

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

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

$\Rightarrow A$

$\Rightarrow B$

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

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

hale Complemet of fo on more on varebale he 2 and of the Pndevyal complement.

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

A	B	$\bar{A}$	$\bar{B}$	A+B	$\overline{A+B}$	A•B
0	1	1	1	0	1	1
0	1	ɸ	0	1	0	0
1	0	D	1	1	0	0
1	0	0	0	1	0	0

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

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

Universal Gate 8

legick Gate are eltronic sancite which Pafam lagical fanson on one ~~oo~~ on noun input to pradus one input.

there are seven legick gate. here input cobenalan of a legick gate are reten a sening And raspanding output reten along them thes input and output Combnalan ~~a~~ is cald truth table.

there are two universal logic Gate one is Nand

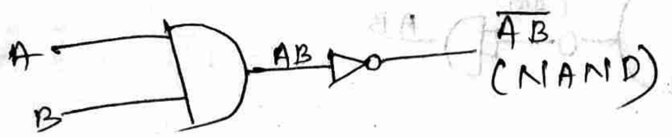
* whey this to logick are cald cenivensal lagek gate ?

why these 2 logicgates arcee

becoge ~~[illegible]~~ this logicgathe can inpleement ~~ant~~ any ony

NAND GAte this

NAND GAte is achive a cobenasan of 2 lagick gate that is and gate folleode by ~~Not~~ And gate.

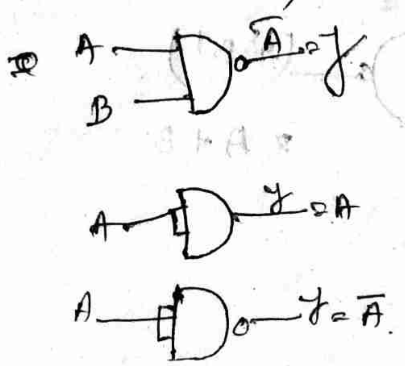


NAND Gate has Not Gate

A Not predusas complement

it can habe , tie the input of a NAND Gate togathen NAND it hade

$$y = \overline{AB}$$
$$= \overline{AA}$$
$$= \overline{A} \quad \text{(NOT)}$$



NAND ~~as~~ AND Gate

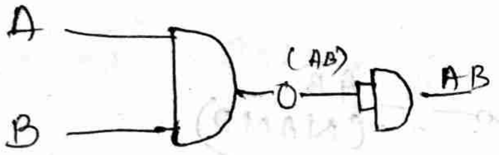
A NAND GATe prodeuses Complent of And gate So if the output of the NAND Gate ovenkeal that of an AND Gate

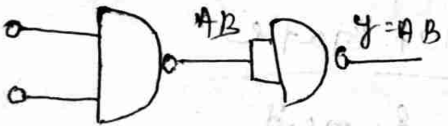
$$y = (AB)'$$

$$= ((A \cdot B)')' \qquad = \overline{\overline{A}} \cdot \overline{\overline{B}}$$

$$= A \cdot B$$

$$= AB$$



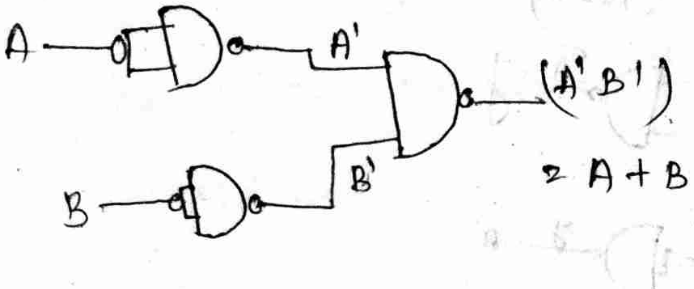


NAND GAte ~~as~~ or GAte

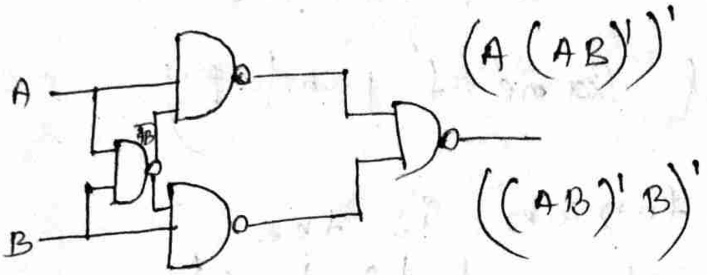
Form D-Morgans tearam

So give the invated input to a nand gate OCte on output

OR GAte



the output of A to input ex-or Gate



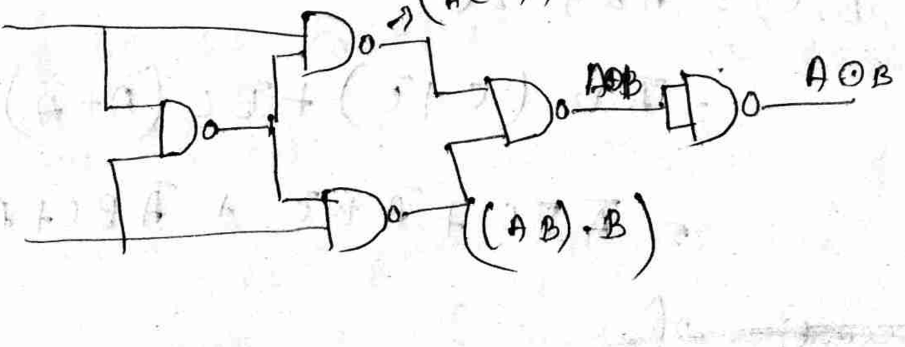
$\left(\overline{AB} + \overline{AB}\right)$

$\left(A\left(\overline{A}B\right)'\right)$

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

~~Nor~~ NAND.

ex - Nor gate



Sop ~~of~~ (Some of preadats ~~for~~) form:-

(1) The is all so clad dis jumtip (canonical form) (DCF) or (extaentd same of pradat) ~~or~~ (Canonical Same of preadats)

(2) in this form, the fansan is the same of a number of preadats huat is preadat form contance all vareabuls of the ~~fan~~ foun san ~~or~~ either in complimented or on complinented

(3) this caine all so be derid ~~to~~ form the truth table find on Some of all the ~~0~~ fones carespens to though canbenasa for which (f) Assumes the valu (1).

~~for~~ exm

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

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

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

~~Som Sume of~~

The preadat toeum his contens all the nareabuls edarcing ing complemata and in comeplematad minterms The min teorm is denotat m_0, m_1, m_2 ...... an 'n' variables fansan can habe 2^n 'n' men torms.

Another way of reepregeing the fansan in canonikal SOP form is the soing the ~~&~~ of menterms for which the fansan equals to 1.

exm

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

Or,

$$f(A,B,C) = \Sigma m(1,2,3,5)$$

Σm repregents the somo of all the men tomos which dasemal codes are given in the parenethesis 2.12.21

POS Pradact - of - Sums Form;

- * this form is alleo cold As conjunctive canonical form (CCF) or expandade pradeict of Sums form or canonical praduct of Sums form;

- * This is by consedaring the canbenasen for which $f = 0$.

- * is trom is a Sum of all the vareabuls.

- * The fansuen $f(A,B,C) = (\overline{A} + \overline{B} + C \cdot \overline{C})$

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

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

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

The Sam freeme which cantaint is outher and varcabeufs in fdarca comeplematad or or on complematad form is cald a max term .

manterm is repragtad → $M_0, M_1, M_2, M_3, M_4 \dots$ ~~tes~~ they (CCF) of 'f' mabe reetanas $f(A,B,C)$

$M_0, M_4, M_6 . M_7$ or

$f \, (A\,B\,C) \Sigma_m(0,4,6,7)$

* repraegtat kuat ahan f of abc is repraegetatd the pradat of all the Max term .

* Canvarsan betwen Canecul form

the compeemat of fansan expresetd some of mentorms = 2 the messing form the origenal fuansan .

Exm

$f(A,B,C) \Sigma m$ ~~(0,4,6,7)~~ $(0,2,4,6\,7)$

this has a complement $f(\overline{A+B+C}) =$

$$\Sigma m(1,3,5) = M_1 + m_3 + M_5$$

if we complmant ~~of~~ De-Morgan's ~~low~~ theorom of 10 'f' in a form

$$f = (\overline{M_1 + M_3 + m_5}) = (\overline{M_1} \cdot \overline{M_3} \cdot \overline{M_5})$$

3/12/21

$= M_1 M_3 M_5 = \Pi\, M(1,3,5)$

sav Ψ

	Sop	pos
①	$f = 1$	$f = 0$
②	minterms	Maxterm
Adeson — ③	$m_0, m_1, \ldots$	$M_0, M_1, \ldots$
④	Σ	Π
⑤	sum	product

~~Sattra~~ Malte purposen

6/12/21

Sop

Sum of products

pos

product of sums

~~Karnaps~~

Karnaugh map

~~or~~ by (K - Map) :-

→ The K-Map is a chart or a graph composed of and arrangement of adjacent cells, each representing a perticular combination of ~~very much~~ variables in sum or product.

→ The K-map is a systemetic method of simplyfing the bolion expression.

> Thrae types of veriable k-mape.

> A Two veriable expretion can have $2^2 = 4$ pasevule Cabenasa of the input veriabul A and B

<u>Mapping of SOP</u> :- expertion

> the two variable k-map has $2^2 = 4$ squeres. these squeres are cold cells

=> A '1' is placd in any squere indecats that Conraspading is in cdudad out put Supnasan and a '0' and o No entry in eny squre indicates thats Conresponding minterm doest appiear in the enpretion for out put

A \ B	0	1
0	$\bar{A}\bar{B}$	$\bar{A}B$
1	$A\bar{B}$	AB

three veriabcale :-

A function in three veriabule (a, b, c) can be expressed in Sop an pos from haveing eight postible combination.

BC / A	00	01	11	10
0	0 $\bar{A}\bar{B}\bar{C}$ (M_0)	1 $\bar{A}\bar{B}C$ (M_1)	3 $\bar{A}B\bar{C}$ (M_3)	2 $\bar{A}B\bar{C}$ (M_2)
1	4 $A\bar{B}\bar{C}$ (M_4)	5 $A\bar{B}C$ (M_5)	7 $\bar{A}BC$ (M_7)	6 $AB\bar{C}$ (M_6)

1

(a) Minterms

→ A three veriabule have 8 Sqares or cells and eace squre on the map repragent a minterm ~~or~~. maxterm Some is figare bellow.

BC / A	00	01	11	10
0	0 $\bar{A}+\bar{B}+\bar{C}$ (m_0)	1 $\bar{A}+\bar{B}+C$ (m_1)	3 $\bar{A}+B+C$ (m_3)	2 $\bar{A}+B+\bar{C}$ (m_2)
1	4 $A+\bar{B}+\bar{C}$ (m_4)	5 $A+\bar{B}+C$ (m_5)	7 $\bar{A}+B+C$ (m_7)	6 $A+B+\bar{C}$ (m_6)

X ✗

maxterms

BC / A	00	01	11	10
0	0 $A+B+C$ (M_0)	1 $A+B+\bar{C}$	3 $A+\bar{B}+\bar{C}$	2 $A+\bar{B}+C$
1	4 $\bar{A}+B+C$	5 $\bar{A}+B+\bar{C}$	7 $\bar{A}+\bar{B}+\bar{C}$	6 $\bar{A}+\bar{B}+C$

2

Maxterms

CD AB	00	01	11	10
00	0 $ABCD$ M0	1 $ABC\bar{D}$ M1	3 $AB\bar{C}\bar{D}$ M3	2 $AB\bar{C}D$ M2
01	4 $A\bar{B}\bar{C}D$ M4	5 $A\bar{B}C\bar{D}$ M5	7 $A\bar{B}\bar{C}\bar{D}$ M7	6 $A\bar{B}\bar{C}D$ M6
11	12 $\bar{A}\bar{B}CD$ M12	13 $\bar{A}\bar{B}C\bar{D}$ M13	15 $\bar{A}\bar{B}\bar{C}\bar{D}$ M15	14 $\bar{A}\bar{B}\bar{C}D$ M14
10	8 $\bar{A}BCD$ M8	9 $\bar{A}BC\bar{D}$ M9	11 $\bar{A}B\bar{C}\bar{D}$ M11	10 $\bar{A}B\bar{C}D$ M10

Wright

Sop	Pos

compliment

1s complement

111001 → x binary

2) 000110 → 1s complement of x

1s complement of a binary no. is an another binary number obtained by toggling all the 0 bit to 1 & the 1 bit too.

2's complement

10010 → y

01101 → 1s complement

+1

01110 2's complement

∴ If weadd 1 to 1's complement of binary no. then the resulting no is known as 2's complement.

For an n-bit

n-bit number max +ve number which can be represent by 2's complement

$= 2^{n-1}-1$

- max (-) ve no. can be

$= -2^{n-1}$

UNIT - 2

A Cobenasahal Sanket conseet of Co logick Gates whose autput at eny time are detarmind form only the Pragent Cobenasan Of inputs ,

Parfans and Oprition they Can be spetelf pofam thet can be Spefad legecal ba a Set of bulen fonstion.

a ef an Intan cunestan

Cobenasal lagick gate raat of to the Valus of the Sinepel end then in put and pradecy the valu out put Singel tras fanning byname infanmasen give an input data to are requend data.

* a block digram is sein in the bilo figen.

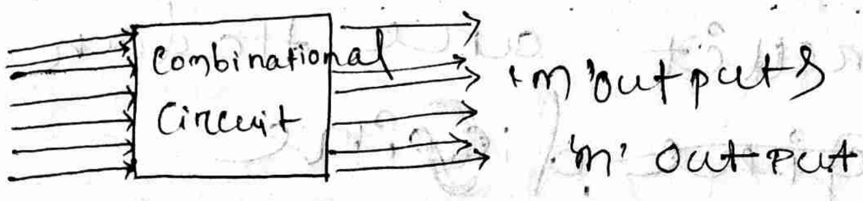
* the 'n' input bynary vareabuls come form and anstronal ; the M output vaneabuls are pradutes Internal Cobenasaned logick Sancit go to etarnnal do ,

Interpented

his input andoutput vareabal ~~intotes~~

las an anlolog lingbal how hale are interpented a bynane singel that repragents logick 1 and 0

logick 1 and logick 0



~~transfer~~

Sequential cireyit it is a Sihous out put defin of on the pragent input, preveos output to the cequans has ben aplied.

who the sequential cireyit is Defanat fore combenasnal cireyit

① in comenasanal cireyit out put deffes of on prajent input at any input and do not us man yelle menary hels. helee preves input does not have any iffect on the cireyit

BINARY ADDER - SUBTRACTOR

* Digital computers perform a variety of information - processing tasks. Among the functions encountered are the various arithmetic operations.

* The most basic arithmetic operation is the addition of two binary digits. This simple addition consists of four possible elementary operations

0 + 0 = 0, 0+1 = 1, 1+0 = 1 and 1+1 = 10

* The first three operations produce a sum of one digit, but when both augend and add~~ee~~end bits are equal to 1; the binary sum Cosisists to two digits. The higher significant bit of this result is called a carry.

* when the augend and addand numbers contain more significant digits, the carry obtained from the addition of two bits is added to the next higher order pair of significant bits.

* A combinational circuit that performs the addition to two bits ~~and~~ is called a half adder.

* one that performs the addition of three bits (two significant bits and a previous carry) is a full adder. the names of the circuits stem form the fact that two half adders can be employed to implement ~~to~~ a full adder.

HALF ADDER

- * This circuit needs two binary input and two binary outputs.
- * The input variables designate the augend and addend bits; the output variables produce the sum and carry. Symbols x and y are assigned to the two input and S and C to the outputs.
- * The truth table for the half adder is listed in the below table.
- * The C output is 1 only when both inputs are 1. The S output represents the least signifiicant bit of the sum.
- * The simplified Boolean functions for the two output can be obtained directily from the truth table

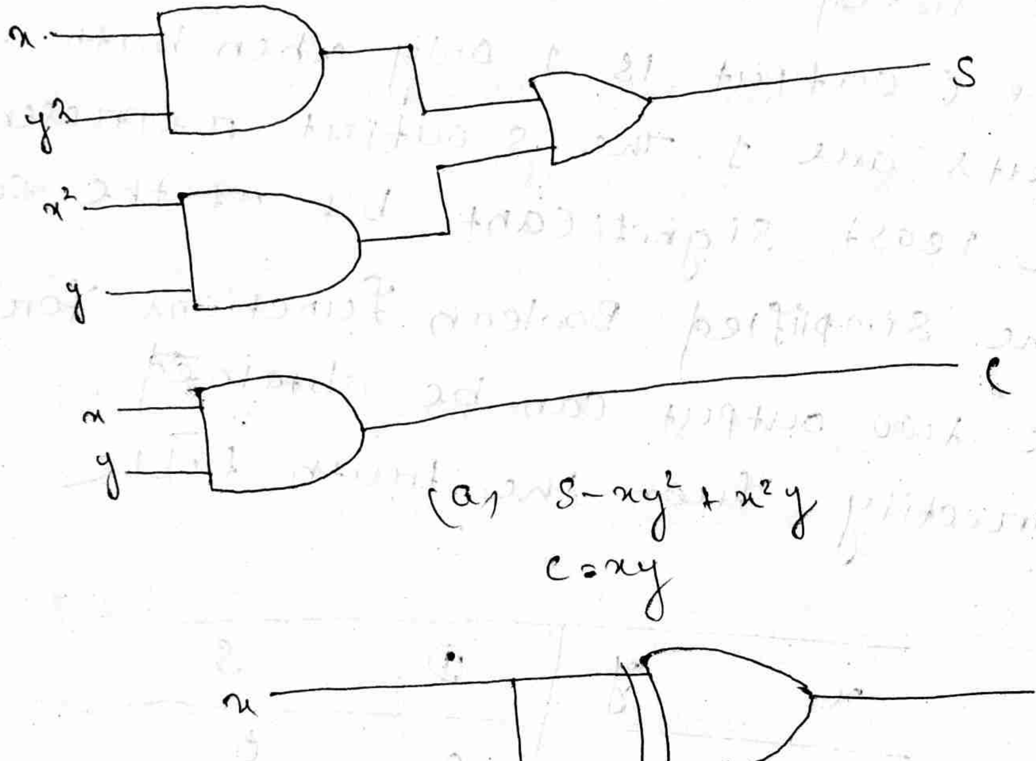
x	y	D	S
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Trough table

* The Simplified Sum-of-Products expressions are $S = x'y + xy'$

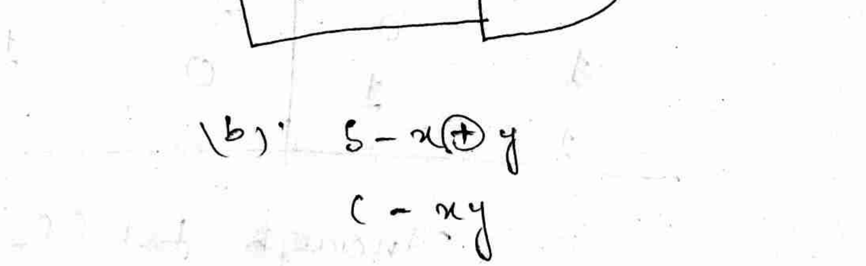
$C = xy$

* The logic diagram of the half adder implemented in Sum of products is shown in the below figure. It can be also implemented with an exclusive-or and an AND gate.



(a) $S - xy' + x'y$

$C = xy$



(b) $S - x \oplus y$

$C - xy$

FULL ADDER

- ★ A full adder is a Combinational Circuit that forms the arithmetic sum of three bits.
- ★ It Consists of three inputs and outputs. Two of the input variables,

FULL ADDER

- ★ A full adder is a combinaional circuit that forms the arlithmetic sum of three bits.
- ★ It Consists of three inputs and two of the input variables, denoted by x and y, represent the two significant bits to be addad. The thired input z, represents the carry from the carry from the previous lower significant position.

x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Truth Table

Two outputs are necessary because the arithmetic sum of three binary digits ranges in value form 0 to 3, and binary representation of 2 or 3 needs two bits. The two outputs are designated by the symbols S for sum and c for carry.

x \ yz	00	01	11	10
0	m_0	m_1 1	m_3	m_2 1
1	m_4 1	m_5	m_7 1	m_6

(a) $S = x'y'z + x'yz' + xy'z' + xyz$

x \ yz	00	01	11	10
0	m_0	m_1	m_3 1	m_2
1	m_4	m_5 1	m_7 1	m_6 1

(b) $C = xy + xz + yz$

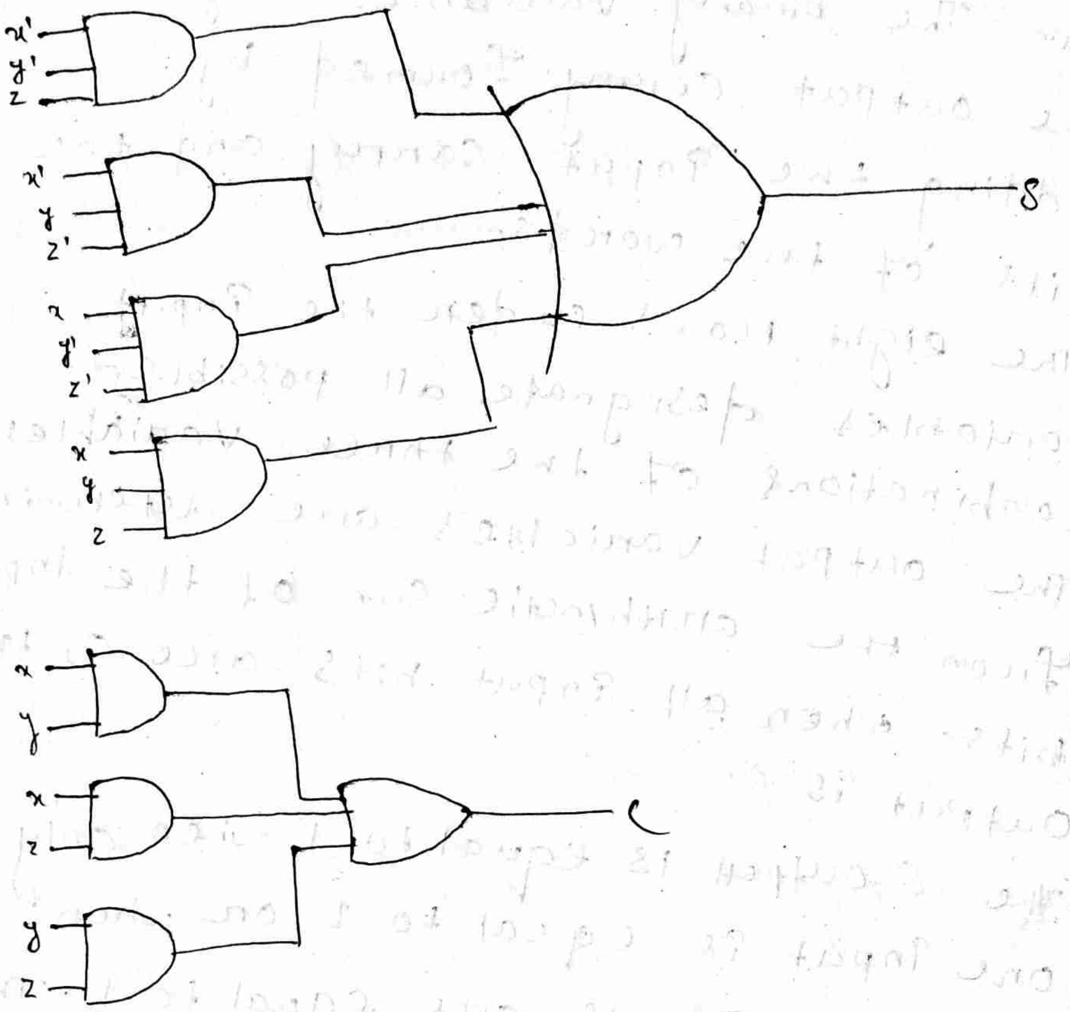
K-map for full adder

- the binary variable S gives the value of the least significant bit of the Sum. The binary varibable C gives the output carry formed by adding the input carry and the bits of the words.
- The eight rows under the input variables designate all possible combinations of the three variables. The output variables are determined from the arithmetic sum of the input bits. when all input bits are 0, the output is 0.
- The S output is equal to 1, when only one input is equal to 1 or when all three inputs are equal to 1. The C output has a carry of 1 if two or three inputs are equal to 1.
- The simplified expressions are

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

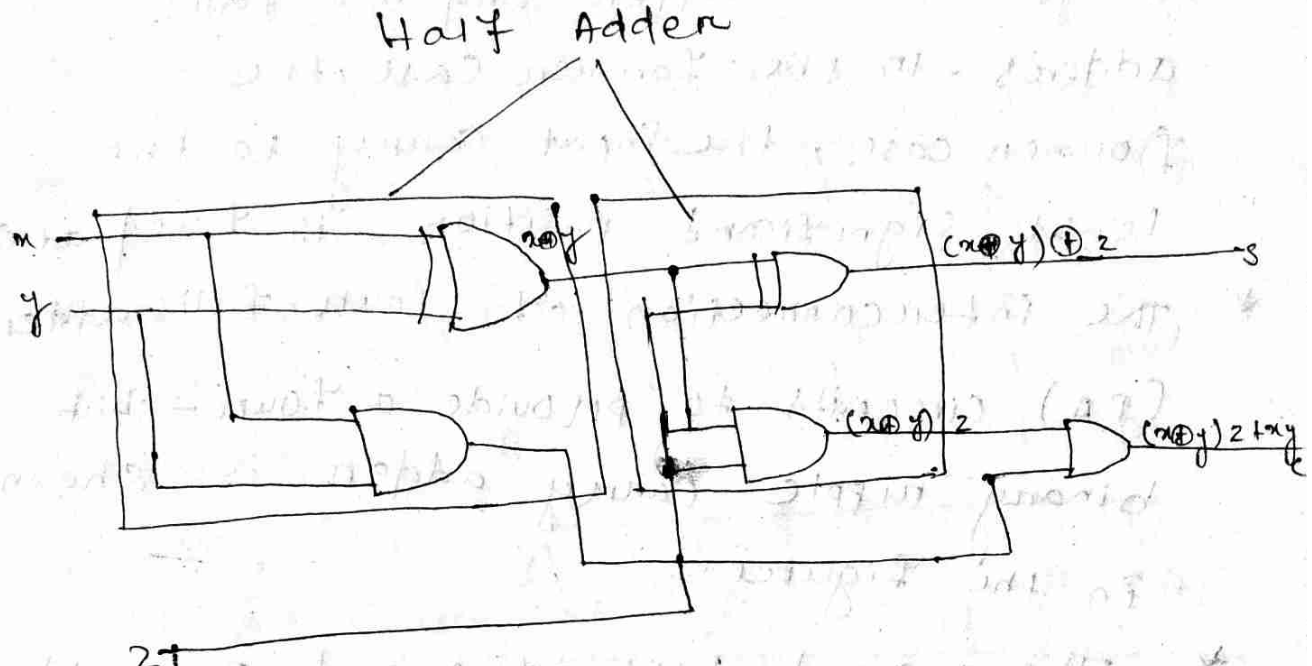
$$C = xy + xz + yz$$

* the logic diagram for the full adder implemented in sum-of-products form is shown in figure.



Implementation of full Adder in SOP form

* It can also be implemented with two half adders and one OR gate as shown in the figure.



implementation of full Adder using Two Half Adders and an OR gate

* A full adder is a combinational circuit that forms the arithmetic sum of three bits.

BINARY ADDER

* A binary adder is a digial that produces the arithmetic sum of two binary numbers.

* It can be constructed with full adders connected in cascade, with the output carry from each full adder connected to the input carry of the next full adder in the chain.

* Addition of n-bit numbers requires a chain of n full adders or a chain of one-half adder and n-1 full adders. In the former case, the former case, the input carry to the least significant position is fixed at 0.

* The interconnection of four full-adder (FA) circuits to provide a four-bit binary ripple carry adder is shown in the figure.

* The augend bits of A and the addend bits of B are designated by subscript numbers form right to left, with subscript 0 denoting the least significant bit.

* The carries are connected in a chain through the full adders. The input carry to the adder is C0, and it ripples through the full adders, with each output carry connected to the input carry of the next higher through order fulll adder.

* Consider the two binary number A = 1011 and B = 0011. Their Sum S = 1110 is formed with the four-bit adder as follows.

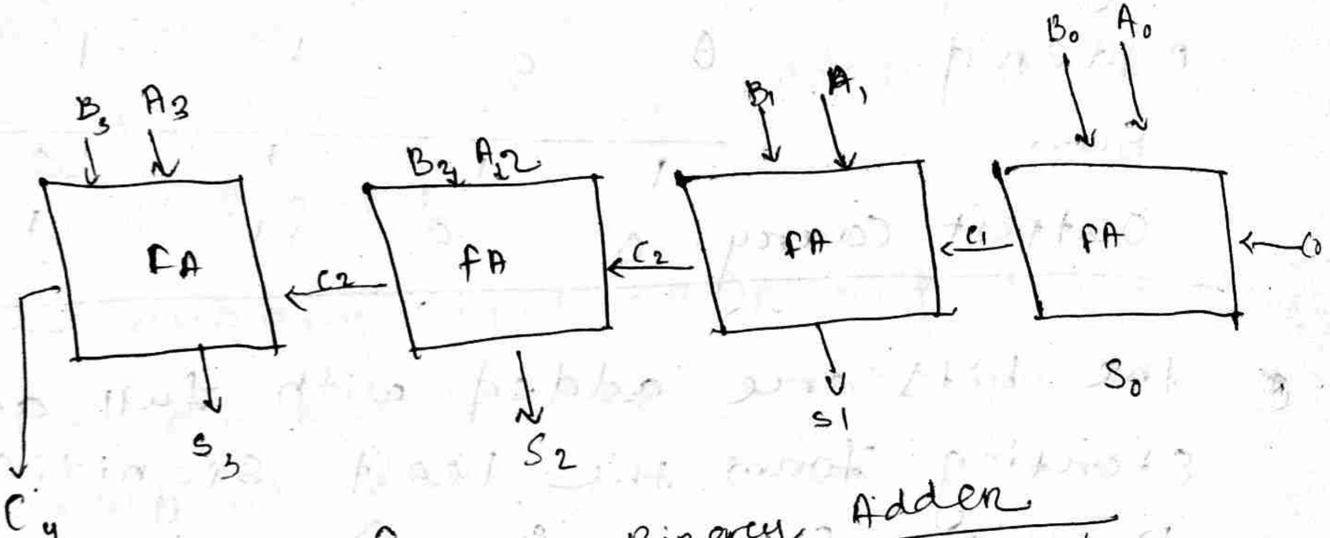
Subscript i;	3	2	1	0	
Input carry	0	1	1	0	C_i
Augend	1	0	1	1	A_i
Addend	0	0	1	1	B_i
Sum,	1	1	1	0	S_i
Output carry	0	0	1	1	C_{i+1}

* the bits are added with full adders starting form the least significant position (subscript 0). to form the sum bit and carry bit. The input carry C_0 in the least significant position must be0.

* The value of C_{i+1} in a given significant position is the output carry of the full adder. This value is transferred into the input carry of the full addar the adds the bits one higher significant position of the left.

* the Sum bits are thus generated Starting form the rightmost position and are available as soon as the corresponding previous Carry bit is generated.

All the Carries must be generated for the Correct Sum bits to appear at the outputs.



four Bit Binary Adder

~~Foure Bit Binary Adder~~

HALE SUBTRACTOR:

- * This circuit needs two binary inputs and two binary outputs.
- * Symbols x and y are assigned to the two inputs and D (for difference) and B (for borrow) to the outputs.
- * The truth table for the half subtracter is listed in the below table.

x	y	D	B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Truth Table

- * The B output is 1 only when the inputs are 0 and 1. The D output repelesents the least significant bit of thee subtraction.
- * The subtreaction operation is done by using the following rules as

0-0=0;

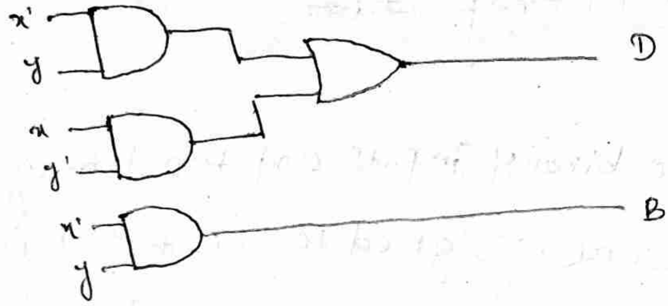
0-1=1 with borrow 1;

1-0=1;

1-1=0;

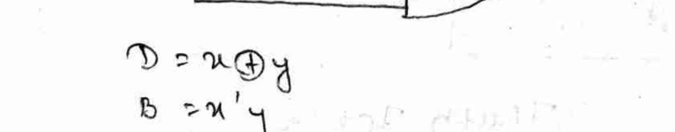
★ The simplified Boolean functions for the two outputs can be obtained directly from the truth table. The simplified sum-of-products expressions are.

$$D = x'y + xy' \text{ and } B = x'y$$



$D = x'y + xy'$

$B = x'y$



$D = x \oplus y$

$B = x'y$

* The logic diagram of the half adder implemented in sum of products ~~is~~ is shown in the figure. It can be also implemented with an exclusive-OR and an AND gate with one inverted input.

full SUBTRACTOR :-

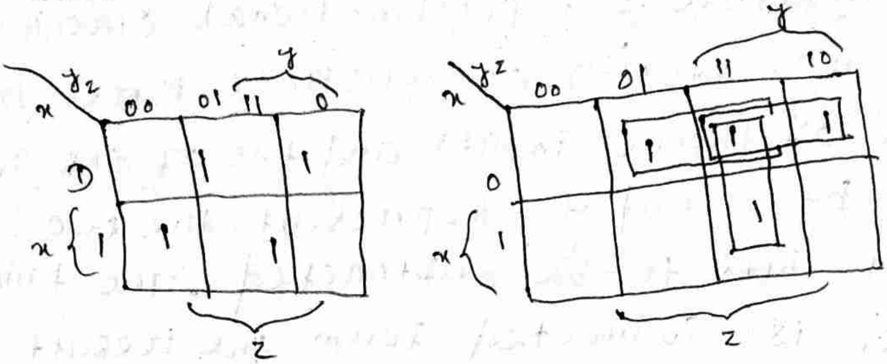
* A full subtractor is a combinational circuit that forms the arithmetion operation of three bits.
* It consists of three inputs and two of the input variables by x and y, represent the two significant bits to be subtracted. The third input z, is subtacted form the result of the first subtraction.

x	y	z	D	S
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

Truth table

Two outputs are necessary because the arithmetic subtraction of three binary ~~re~~ digits ranges in value from 0 to 3. and binary representatio of 2 or 3 needs two bits. The two outputs are designated by the symbols D for difference and B for Borrow.

* The binary variable D gives the value of the least significant bit of the difference. The binary variable B gives the output borrow formed during the subtraction process.



$D = x'y'z + x'yz' + xy'z' + xyz$ $B = x'z + x'y + yz$

K-map for full Subtracton

* The eight rows under the input variables designates all possible combinatins of the three variables. The output variables are determined form the arithmetic, subtraction of the input bits.

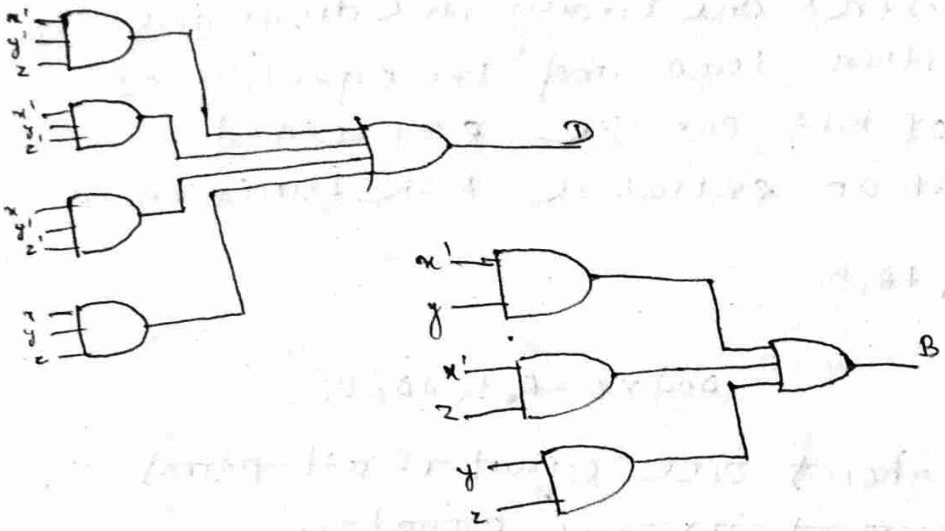
* The difference D becomes 1 when any one of the input is 1 or all three inputs are equal to 1 and the borrow, B is 1 when the input combination (001) or (010) or (011) or (111).

* The simplified expressions are

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

$B = x'z + x'y + yz$

* The logic diagram for the full adder implemented in sum-of-products form is shown in figure.



Implementation of full Subtractor in Sop form

MAGNITUDE COMPARATOR:-

* A magnitude comparator is a Combinational circuit that compares two number A and B and determines their relative magnitudes.

* The following description is about a 2-bit. magnitude comparator circuit.

* The outcome of the comparison is specified by three binary variables that indicate whether A<B, A = B, or A > B

* Consider two numbers A, and B, with two digits each. Now writing the coefficients of the number in descending order of significance:

$$A = A_1 A_0$$

$$B = B_1 B_0.$$

* The two numbers are equal if all pairs of significent digits are equal i.e. if and only if A1 = B1, and A0 = B0.

* when the numbers are binary ,the digits are digits are either 1 or 0 and the equality of each pair of bits can be expriessed logically with an exclusive -NOR function of

$x_1 = A_1 B_1 + A_1 B_1$

And $x_0 = A_0 B_0 + A_0' B_0'$

* ~~The two numbers are equal if all pairs of significant digits are equal~~ i.

* The equality of the two numbers A and B is displayed in a combinational circuit by an output binary variable that we designate by the symbol (A = B)

* This binary variable is equal to 1 if the input number, A and B are equal , and is equal to 0 otherwise .

* for equality to exist , all xi variables must be equal to 1 a condition that dictates an AND operation of all variables.

$$(A = B) = x_1 x_0$$

* The binary variable (A = B) is equal to 1 only if all pairs of digits two numbers are equal.

* TO determine whether A is greater or less than B, we inspect the relative magnitudes of pairs of significant digits , starting form the most significant position . If the two digits of a pair are equal , we compare the next lower significant pair of digits. if the corresponding dight A is 1 and that of B is 0 , we conclute that A > B . If the corresponding digit of A is 0 and that of B is 1 we have A < B . The

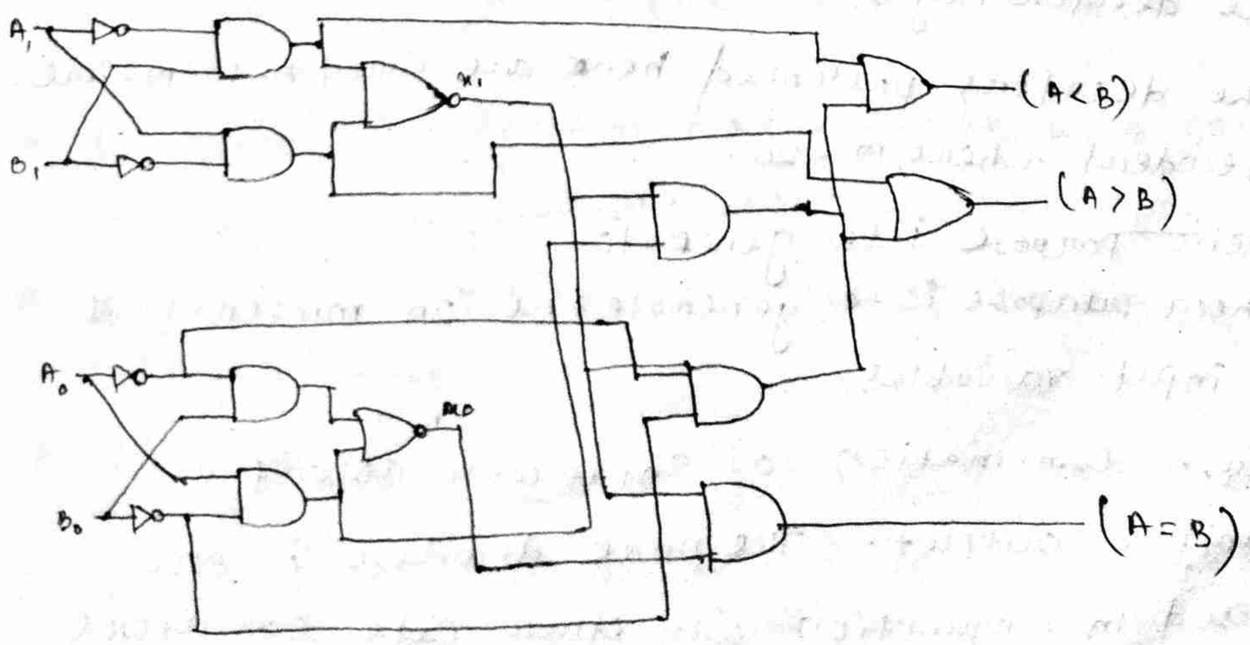
sequential comparison can be expressed logically by the two Boolean functions

$$(A > B) = A_1 B_1' + x_1 A_0 B_0'$$

$$(A < B) = A_1' B_1 + x_1 A_0' B_0'.$$

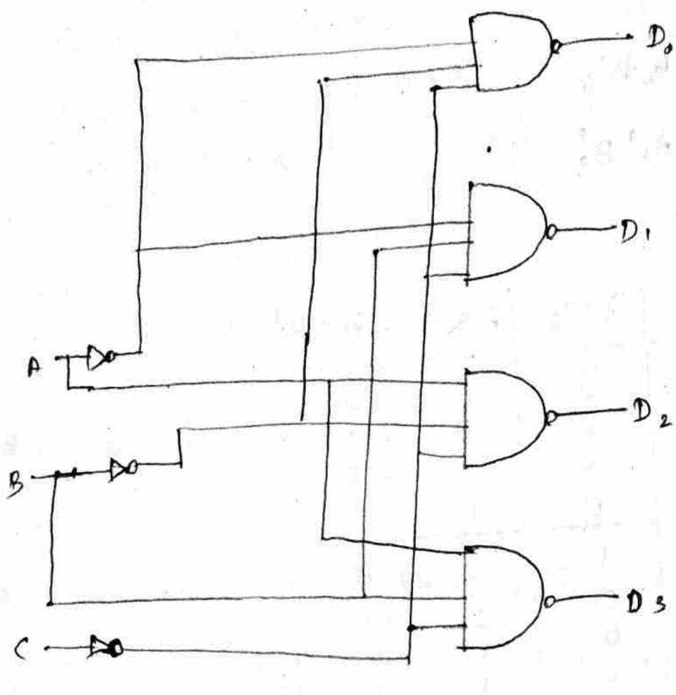
A_1	A_0	B_1	B_0	A>B	A<B	A=B
0	0	0	0	0	0	1
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	0	1	0
0	1	0	0	1	0	0
0	1	0	1	0	0	1
0	1	1	0	0	1	0
0	1	1	1	0	1	0
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	0	1
1	0	1	1	0	1	0
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	0	1

Truth table



Logic Diagram of 2-bit Magnitude Comparator

DECODER :-



E	A	B	D_0	D_1	D_2	D_3
1	x	x	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	1

- A decoder is a combinational circuit that converts binary information from n input lines to a maximam of 2^n unique output lines.
- If the n-bit coded information has unused combinations the decoder may have fewere than 2n outputs.
- The decoders presented hear are called n-to-m-line decoders, where m - 2n.

~~* Their purpose is to generate~~

- Their purpose is to generate the 2n minterms of n input variables.
- Each combination of input will assert a unique output. The name decoder is also used in conjunction with other code converters such as a BCD-to-seven-segment decoder.

* Considere the three -to- eight-line decoder circuit to three inputs are each one of the eight AND gates generates one of the minterms.

* The input variables represent a binary number/and the outputs represent the eight digits of a numberin the octal number system.

* However, a three -to- eight - line decoder can be used for decoding any three- bit code to provide eight out put one for each element of the code.

* A two - to - four - line decoder with an enable input constructed with NAND gates is shown in fig.

* The circuit operates with complemented outputs and a complement enable input. The decoder is enabled when E is equal to 0. As indicated by the truth table, only one output can be equal to 0 at any given times; all other outputs equal to 1.

* The output whose value is equal to 1 regardless of the values of the outher to inputs.

* whene the circuit is disabled, none of the outputs are equal to 0 and none of the minterms are selected.

* In general, a decoder may operate with complemented or un- complemented outputs.

* The enable input may be actrated with a 0 are with a 1 signal.

* Some ~~decoders~~ decoders have two or more enable inputs that must satisfy a given logic condition in order to enable ~~to~~ the circuit.

* A decoder with enable input can function as a demultiplexer - a circuit that receives information from a single line and directs it to one of 2n possible outputs links.

* The selection of a specific output is controlled by the bit combination of n selection lines.

* The decoder of Fig. can function as a one-to-four line demultiplexer when E is taken as a data input line and A and B are taken as the selection inputs.

* The single input variable E has a path to all four outputs, but the input information is directed to only one of the output lines as specified by the binary combination of the two selection lines A and B.

* This feature can be verified from the truth table of the circuit.

* For example, if the selection lines AB = 10, Output D_2 will be the same as the input value E, while all outher ~~out~~ outputs are maintained at 1.

* Since decoder and demultiplexer operations are obtained form the same circuit, a decoder with an enable input is referred to as a decoder-demultiplexer.

* A application of this decoder is binary-to-octal conversion.

ENCODER:-

1 * An encoder is a digital circuity that performs the inverse operation of a decoder.

2 * An encoder has 2n input lines and n output lines.

3 * The output lines as an aggregate, generate the binary code corresponding to the input value.

Inputs								Outputs		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

1 * the above Encoder has eight inputs and three outputs that generate the corresponding binary number.

5 * It is assumed that only one input has a value of 1 at any given time.

6 * The encoder can be implemented with OR gates whose inputs are determined directly from the truth table.

7 * output z is equal to 1 when the input octal digit 1,3,5, or 7.

8 * Output y is 1 for octal digits 2,3,6, or 7, and output x is 1 for digits 4,5,6, or 7.

9 * these conditions can be expressed by the following Boolean output functions.

$$z = D_1 + D_3 + D_5 + D_7$$

$$y = D_2 + D_3 + D_6 + D_7$$

$$x = D_4 + D_5 + D_6 + D_7$$

10 * The encoder can be implemented with three OR gates.

11* The encoder defined above has the limitation that only one input can be active at ~~reg~~ any give time.

12* If two inputs are active simultaneously, the output produces an undefined combination.

13* To resolve this ambiguity, encoder circuits must establish an input priority to ensure that only one input is encoded which is done in the priority Encoder.

PRIORITY ENCODER

1* A priority encoder is an encoder circuit that includes the priority function.

2* The operation of the priority encoder is such that if two or more inputs are equal to 1 at the same time, the input having the highest priority will take precedene.

Inputs				Outputs		
D_0	D_1	D_2	D_3	x	y	2
0	0	0	0	x	x	0
1	0	0	0	0	0	0
x	1	0	0	0	1	1
x	x	1	0	1	0	1
x	x	x	1	1	1	1

3* In addition to the two outputs x and y, the circuit has a third output designated by v; this is a valid bit indicator that is set to 1 when one or more inputs are equal to 1.

4* If all inputs are 0, there is no vaild input and v is equal to 0.

5* The other two outputs are not inspected when v equals 0 and are specified as don't-care conditions.

6 * Here x's in output columns represent donot-care conditions the x s in the input columns are useful for representing a truth in condensed form.

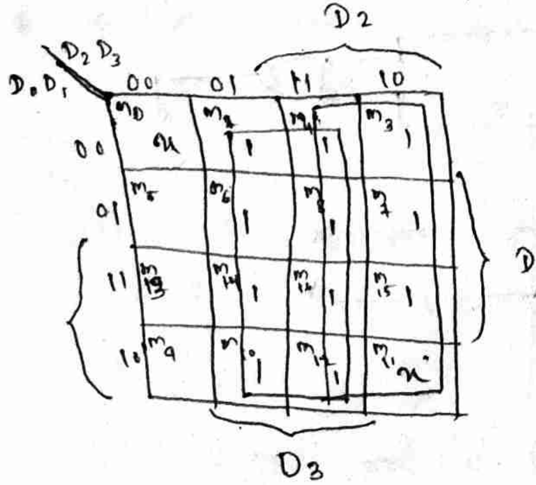
Inputs				Outputs		
D_0	D_1	D_2	D_3	x	y	z
0	0	0	0	x	x	0
1	0	0	0	0	0	1
x	1	0	0	0	1	1
x	x	1	0	1	0	1
x	x	x	1	1	1	1

7 * Higher the subscript number, the higher the priority of the input.

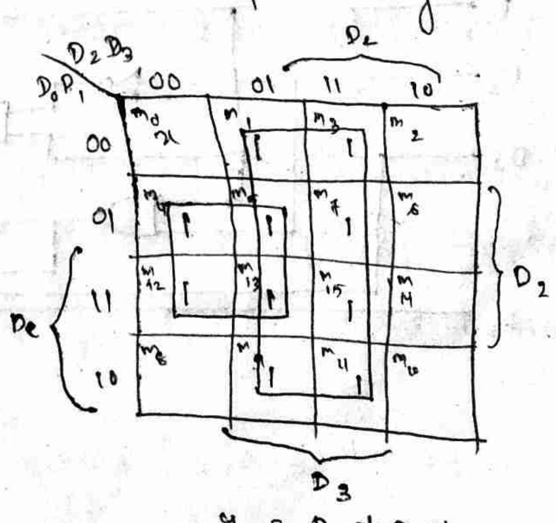
8 * Input D_3 has the highest priority, so, regardless of the values of the other inputs, when this input is 1, the output for xy is 11 (binary 3).

9 * If $D2=1$, provided that $D_3=0$, regardless of the values of the other ~~two~~ lower priority inputs the output is 10.

10 * The output for D_1 is generated only if higher priority inputs are 0, and so on down the priority levels.



$x = D_2 + D_3$



$y = D_3 + D_1 D'_2$

11★ The maps for Simplifying outputs x and y are Shown in above fig.

12★ The minterms for the two functions are derived from its truth table.

13★ Although the table has only five rows, when each x in a row is replaced first by 0 and then by 1, we obtain all 16 possible input combinations.

14★ for example, the fourth row in the table, with inputs xx10, represents the four minterms 0010, 0110, 1010, and 1110, The Simplified Boolean expressions for the priority encoder are obtained from the maps.

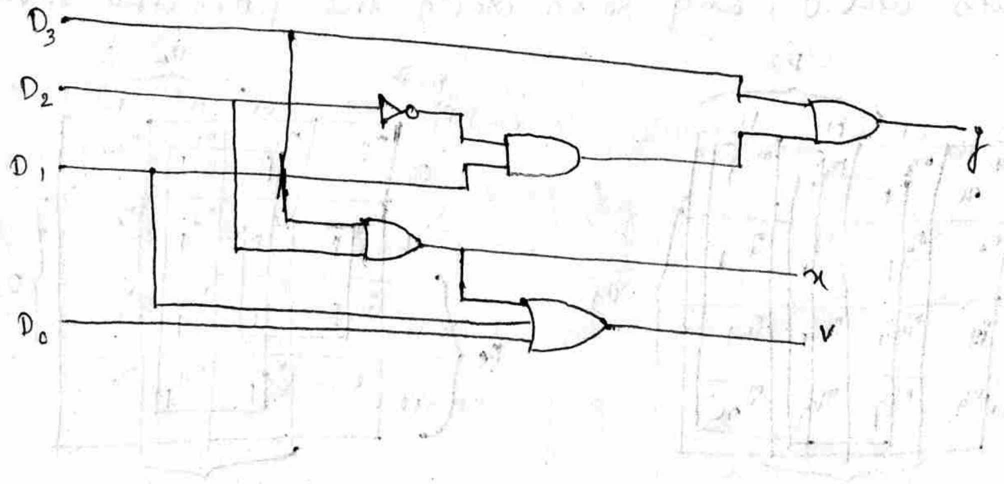
15★ The condition for output V is an OR function of all the input variables.

16★ The priority encoder is implemented according to the following Boolean functions:

$$x = D_2 + D_3$$

$$y = D_3 + D_1 D_2'$$

$$V = D_0 + D_1 + D_2 + D_3$$



MULTIPLEXER:-

1 ★ A multiplexer is a combinational circuit that selects binary information form one of many input lines and directs it to a single output line.

2 ★ The selection of a particuler input line is controlled by a set of selection lines.

3 ★ Normally, there are 2^n input lines and n selection lines whose bit combinations determine which input is selected.

4 ★ A four-to-one-line multiplexer is shown in the below figure. Each of the four inputs, I_0 through I_3, is applied to one input of an And gate.

5 ★ Selection lines S_1 and S_0 are decoder to select a particular AND gate. The outputs of the AND gates are applied to a single OR gate that ~~is~~ provides the one-line output.

6 ★ The function table lists the input that is passed to the output for each combination of the binary selection values.

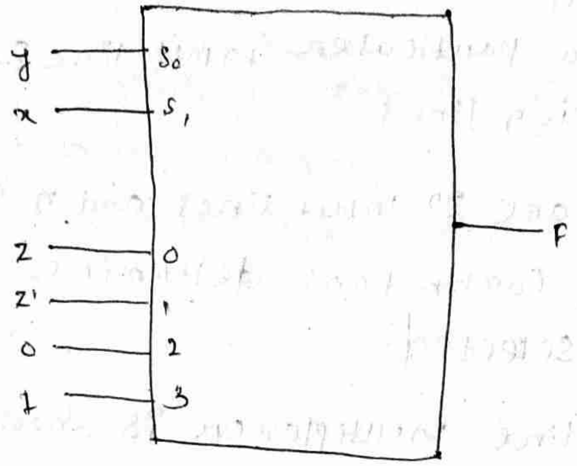
7 ★ To demonstrate the operation of the circuit, Consider the case when

$S_1S_0 = 0$

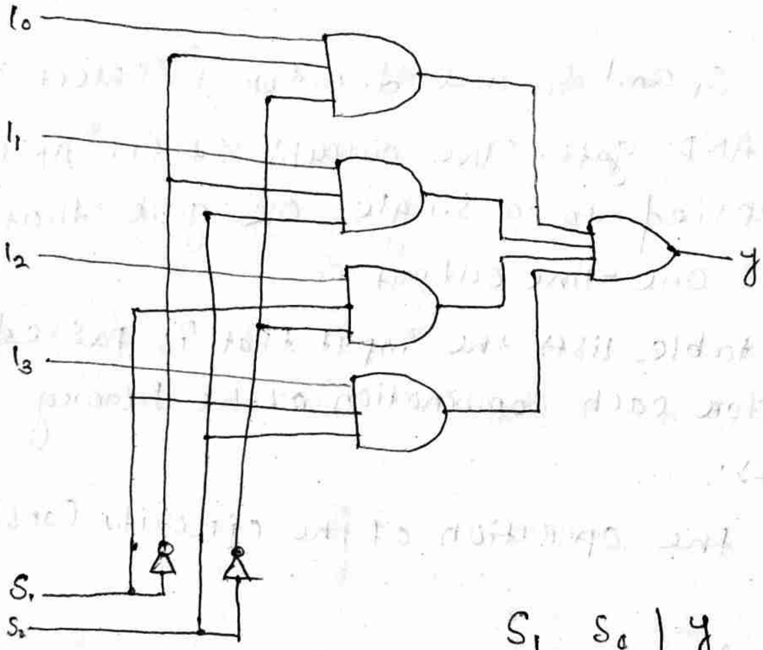
8 ★ The AND gate associated with input I_2 has two of its inputs equal to 1 and the third input connected to I_2.

9 ★ The other three AND gates have at least one input equal to 0, which makes their outputs equal to 0. The output of the OR gate is now equal to the value of I_2, providing a path from the selected input to the output.

10* A multiplexer is also Called a data Selecton, since it selects one of many inputs and Steers the binary information to the output line.



(b) multiplexer implementation



Logic diagram

S_1	S_0	y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

Treuth table

DEMULTIPEXER:-

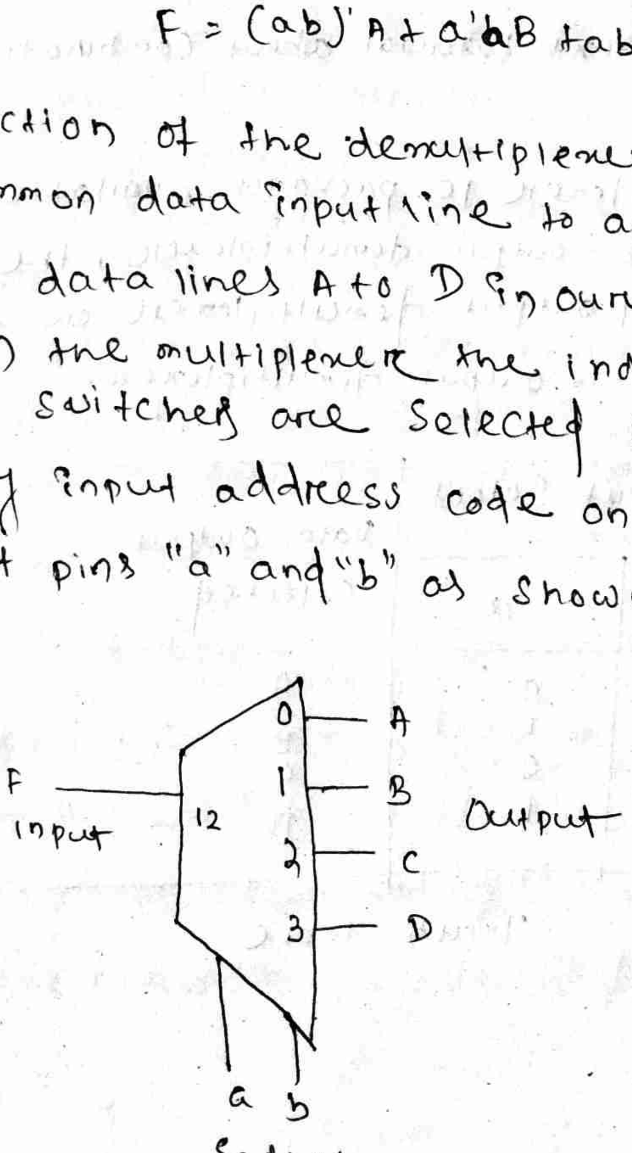
1* The data distributor, known more commonly as a Demultiplexer or "Demux" for short, is the exact opposite of the multiplexer.

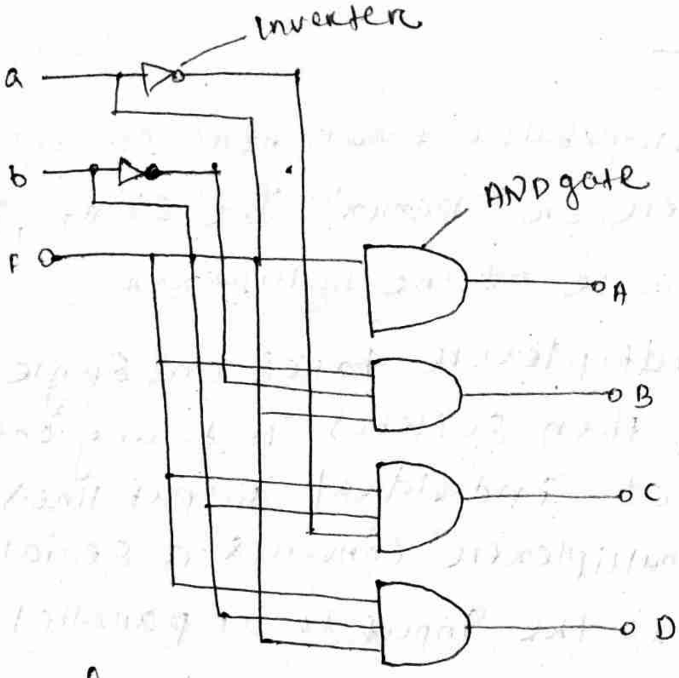
2* The demultiplexer takes one single input data line and then switches it to any one of a number of individual output lines one at a time. The demultiplexer converts a serial data signal at the input to a parallel data at its output lines as shown below.

3* The Boolean expression for this 1-to-4 demultiplexer above with outputs A to D and data select lines a, b is given as;

$$F = (ab)'A + a'bB + ab'C + abD$$

4* The function of the demultiplexer is to swich one common data input line to any one of the 4 output data lines A to D in our example above. As with the multiplexer the individual solid state switches are selected by the binary input address code on the output select pins "a" and "b" as shown.





Logic Diagram

5* unlike multiplexers which convert data form a single data line to multiple lines and demultiplexers which convert multiple lines to a single data line, there are devices available which convent data to and from multiple lines and in the next tutorial about combinational logic devices.

6* Standard demultiplexer IC packages available are the TTL 74LS138 1 to 8 - output demultiplexer, the TTL 74LS139 Dual 1-to-4 output demultiplexer on the cmos CD4514 1-to-16 output demultiplexer.

Output Select		Data Output Selected
b	a	
0	0	A
0	1	B
1	0	C
1	1	D

Truth Table

FLIP-FLOP ANDLATCH :- Ch-3

1* A flip-flop or latch is a circuit that has two stable states and can be used to store information.

2* A flip-flop is a binary storage device capable of storing one bit of information. in a stable state, the output of a flip-flop is either 0 or 1

3* Latch is a non-clocked flip-flop and it is the building block for the flip-flop.

4* A storage element in digital circuit can maintain a binary state indefinitely until derected by an input signal to switch state.

5* storage element that operate with signal level are called latches and those operate with clock transition are called as flip-flops.

6* The circuit can be made to change state by signals applied to ~~be~~ one or more control inputs and will have one or two outputs.

7* A flip-Flop is called so because its output either flips or flops meaning to switch back and forth.

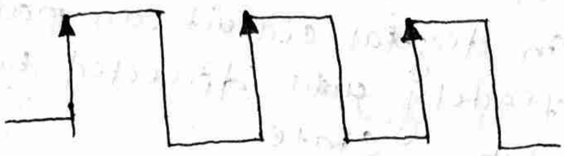
8* A flip-flop is also called a bi-stable multi-vibrator as it has two stable srates. The input signals which command the flip-flop to change state are called excitations.

9* flip-flops are storge devices and can store 1 or 0.

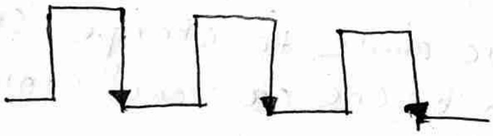
10* flip-flops using the clock signal are called Clocked flip-flops. control signals are effective only if they are applied in synchronization with the clock signal.

11* Clock - signals may be positive - edge triggered or negative - edge triggered.

12* positive - edge triggered flip - flops are those in which state transitions take place only at positive - going edge of the clock pulse



13* Negative - edge triggered flip - flop are those in which state transition take place only at negative - going edge of the clock pulse



14* Some common type of flip - flops include

(a) SR (set - reset) F-F

(b) D (data or delay) F-F

(c) T (toggle) F-F and

(d) JK F-F.

TRIGGERING METHODS :-

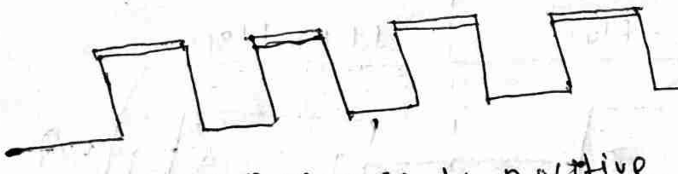
1* the state of a latch or flip-flop is switched by a change in the control input. This momentary change is called a trigger, and the transition it causes is said to trigger the flip - flop.

2☆ Flip-Flop circuits are Constructed in Such away as to make them operate properly when they are part of a sequential circuit that employs a common clock.

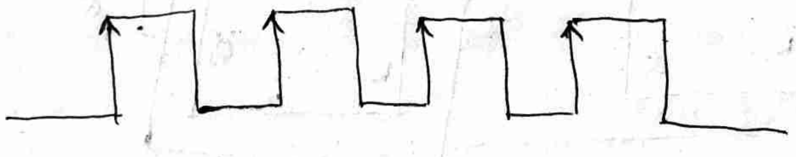
3☆ The problem with the latch is that it responds to a change in the level of a clock pulse. For proper operation of a flip-flop it should be triggered only during a signal transition.

4☆ This can be accomplished by eliminating the feedback part ~~b~~ that is inherent in the operation of the sequential circuit using latches. A clock pulse goes through two transitions: from 0 to 1 and the return ~~or~~ from 1 to 0.

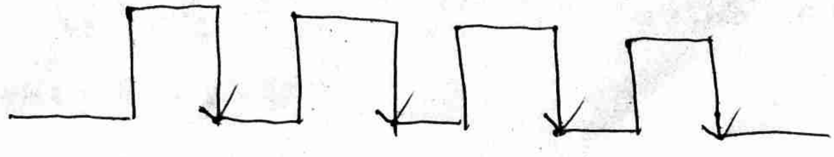
5☆ A ways that a latch can be modified to form a flip-plop is to produce a flip-flop that triggers only during a signal transition (from 0 to 1 or from 1 to 0) of the synchronizing signal (clock) and is disabled during the rest of the clock pulse.



(a) Response to positive level



(b) positive-edge ~~Response~~ Response



(c) Negative-edge ~~Response~~ Response

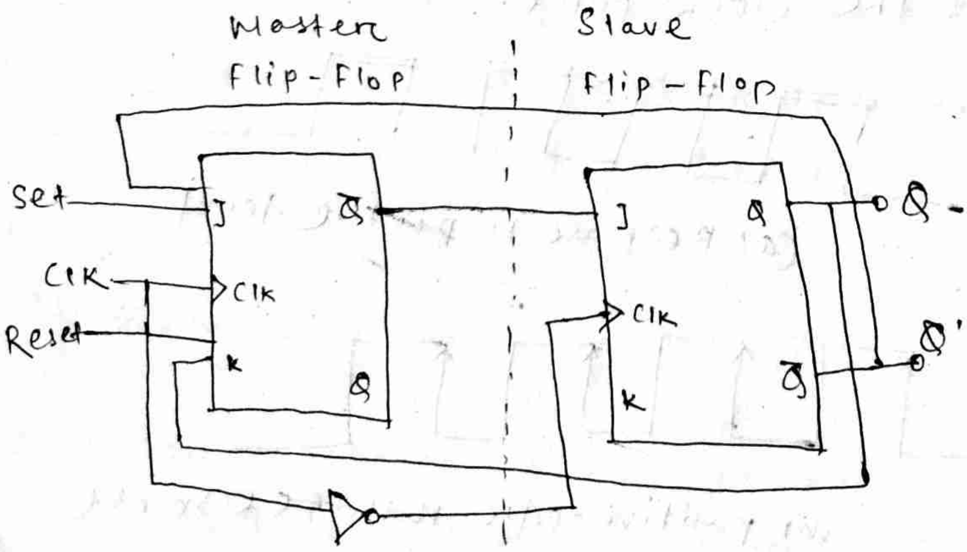
MASTER-SLAVE JK FLIP-FLOP:-

1* The master-slave flip-flop is basically two grated SR flip-flops connected together in a series configuration with the slave having an inverted clock pulse.

2* The outputs from Q and Q from the slave flip-flop are fed back to the inputs of the master with the outputs of the 'master' flip flop being connected to the two inputs of "slave" flip flop

3* This feedback configuration from the slave's output to the master's input gives the characterisic toggle of the JK flip flop as shown below.

The master-slave JK flip flop



4* The input signals J and K are Connected to the gated 'master' SR Filp-flop which 'locks' the input condition while the clock (clk) input is "HIGH" at logic level "1"

5* As the clock input of the "slave" flip flop is the inverse (complement) of the "master" clock input, the "slave" SR fiip-flop does not toggle.

6* The outputs from the "master" filp-flop are only "seen" by the gated "slave" filp-flop when the clock input goes "Low" to logic level "0".

7* when the clock is "Low" the outputs from the "master" flip-flop are latched and any additional changes to its inputs are ignored

8* The gated "slave" flip-flop" now responds to the state of its inputs passed over by the master section.

9* Then on the "Low-to-high" transition of the clock pulse the inputs of the "mester" filp-flop are fed through to the gated inputs of the "slave" filp-flop and on the "High-to-Low" transition the same inputs are reflected on the output of the "slave" making this type of flip-flop edge or pulse-triggered.

10* Then, the circuit accepts input data when the clock signal is "HIGH" and passes the data to the output on the falling-edge of the clock signal.

11* In other words the master-slave JK flip-flop is a "synchronous" device as it only passes data with the timing on the clock signal.

Logic families:— Ch 4 3.1.2022

① A circuit configuration or approach used to produce a tyte of digital integrated circiut is called logic families.

② By using logic familiem we can generate differest logic functions when favricated in the formof an IC a the same approach or in other words belonging to the Same logic family, will have identical electrical characteristic.

③ The set of digital ICs belonging to the same logic families are electricaley comatiele with each outher.

④ Some common charateristic of the same logic family include voltage range, speed of response, poal dissipation input & output logic levels current sourcety & sinking copability for-out noise margig etc.

(5) choosing digital ICs from the same logic family gearanters that there ICs are compatide cuikn respect to eachotres ond that the sustem os a whole portorns the indended cogic funcfions.

~~Types of logic families:-~~

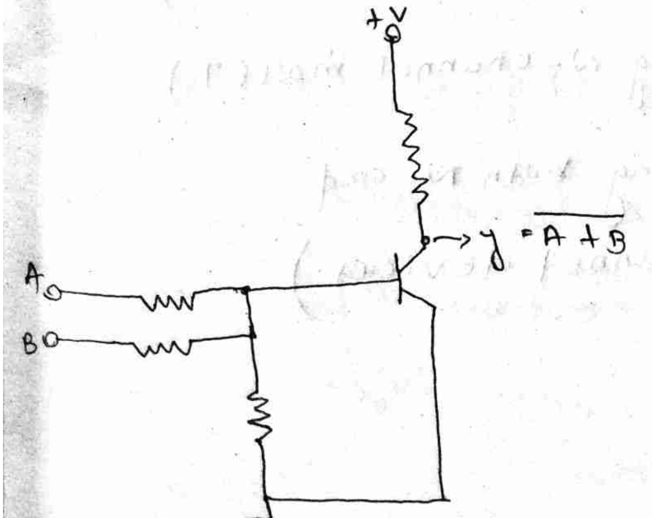
Types of logic family :-

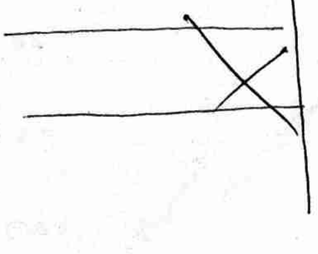
(1) * The entire range of digital ICs is fabricated using either bipolar devices or mos devices or a combination of the two.

(2) * Bipolar families include :-

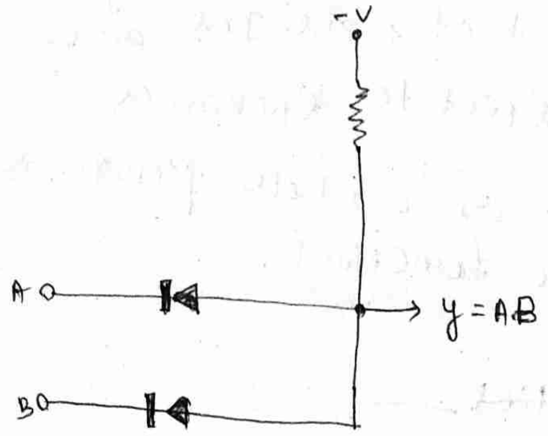
Diode logic (DL)
Resiston -Transiston logic (RTL)
Diode - transiston logic (DTL)
Transiston - Transiston logic (TTL)
Emitten coupled logic (ECL).
(also known as current mode logic (CML))
integrated injection logic (I2L)

* The Bi-mos logic family uses both bipolar and mos devices.

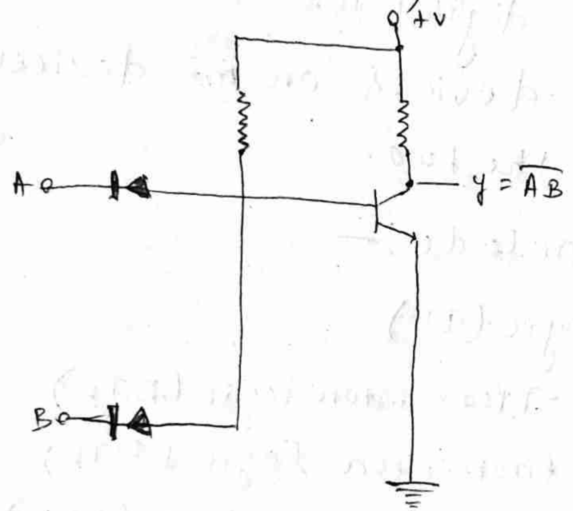




Resistor ~ Transiston logic (RTL)



Diode logic
(DL)



Diode - Transistor logic
(DTL)

(4) Above are some example of DL, RTL and DTL.

(5) mos families include :-

The PMOS family (using P-channel MOSFETs)

The NMOS family (using N-channel MOSFETs)

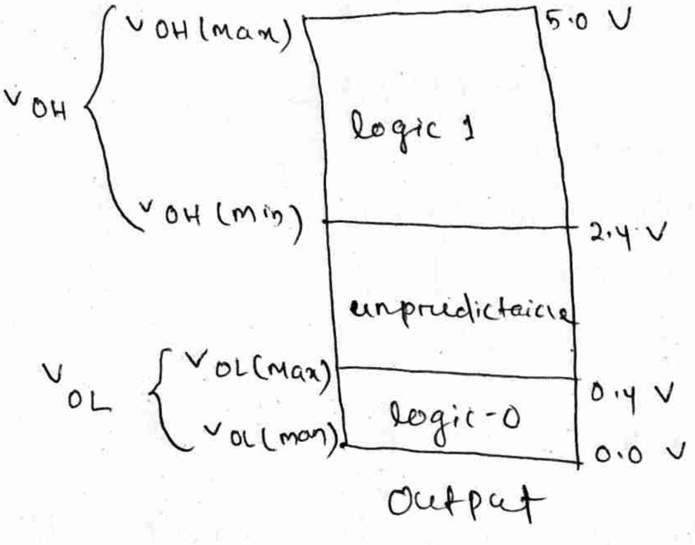
The CMOS family (using both N- and P-channel devices)

Some operational proposties of logic families :-

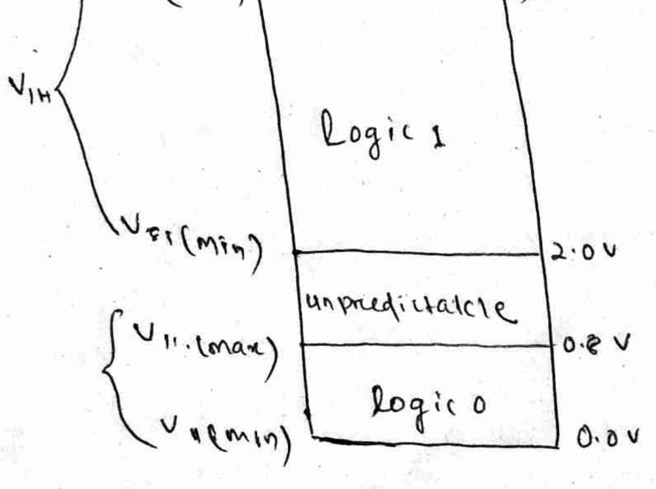
Dc supply voltage:-

The nominal valcde of the DC supply voltage for TTL (Transistor-Transistos-logic) and CMOS (compementary-mental oxide - semiconductior) devsces is +5 volt. Although omnitted from logic dingrams for simplicity, this voltage is connected to Vcc or VDD pin of on IC package and ground is connected to the GND pin.

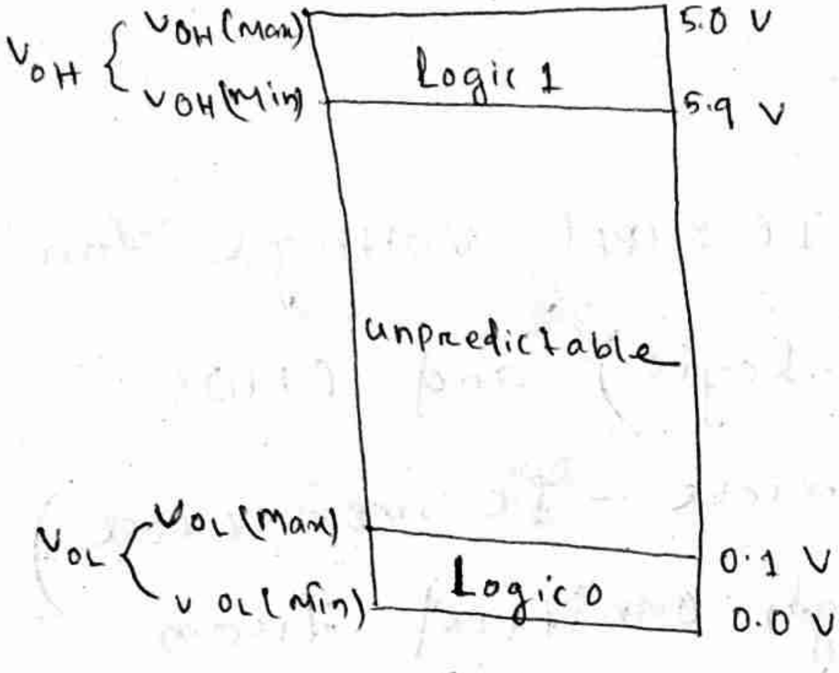
TTL logic levels



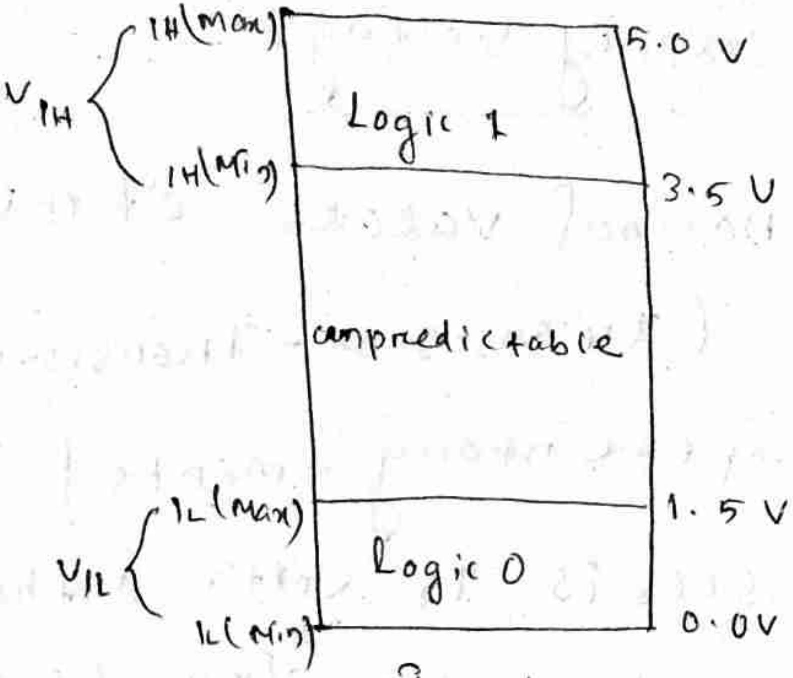
Output



input



Output



Input

CMOS Logic Leveis