MODULE - 1

NUMBER SYSTEMS

The number systems are classified into 4 types

1) Decimal number system
2) Binary number system
3) Octal number system
4) Hexadecimal number system.

1) DECIMAL NUMBER SYSTEM

In the case of decimal number system we are using 10 unique digits They are from 0 to 9 or 0,1 upto 9 ∴ It consist of 10 digits. Therefore the base or radix of the decimal number system is 10.

Eg: $(25)_{10}$, $(325)_{10}$

2) BINARY NUMBER SYSTEM

In the case of binary number system we are using 2 digits that is 0 and 1. Therefore the base or radix of binary number system is 2.

Eg: $(101.0)_2$, $(1101)_2$

3) OCTAL NUMBER SYSTEM

In octal number system there are 8 unique digits. Starting from 0,1 upto 7. So It consist of 8 digits and therefore base, radix of octal number system is 8.

Eg: $(127)_8$, $(523)_8$

4) HEXADECIMAL NUMBER SYSTEM

These are number systems having unique digits starting from 0,1 upto 15. In this system from 0 to 9 we can write similar to decimal numbers. For 10 we write it as A like that,

11 - B

12 - C

13 - D

14 - E

15 - F

Also this system consist of a total of 16 symbols.

∴ The base or radix of a hexadecimal number system is 16

CONVERSION OF NUMBER SYSTEMS

1) DECIMAL TO BINARY NUMBER SYSTEM. $(()_{10} \to ()_{2})$

1? Convert the decimal number 25 to binary

Ans)
2 | 25 → 1
2 | 12 → 0
2 | 6 → 0
2 | 3 → 1
2 | 1 → 1

$25_{(10)} = (11001)_2$

2? Convert the decimal number 137 to binary

Ans)
2 | 137
2 | 68 → 1
2 | 34 → 0
2 | 17 → 0
2 | 8 → 1
2 | 4 → 0
2 | 2 → 0
1 → 0

$(10001001)_2$

3? Convert the decimal 0.25 to binary

papergrid

Date: / /

0·25×2 = 0·5 ↓ $(0.25)_{10} = (0.01)_2$

0·5×2 = 1·0

HW

1? $(123)_{10} - ()_2$

2? $(457)_{10} - ()_2$

3? $(0.41)_{10} - ()_2$

4? $(0.67)_{10} - ()_2$

5? $(15.32)_{10} - ()_2$

6? $(37.45)_{10} - ()_2$

2 | 123 61

6

2 | 12

03

1

3

2 | 61

6

01

1) 2 | 123

2 | 61 → ~~(1)~~ 1

2 | 30 → 1 ~~$(123)_{10} = (1011)_2$~~

2 | 15 → 1

2 | 7 → 0 $(123)_{10} = (1111011)_2$

2 | 3 → 1

2 | 1 → 1

→ 1

2) 2 | 457

2 | 228 → 1

2 | 114 → 0

2 | 57 → 0

2 | 28 → 1

2 | 14 → 0 $(111001001)_2$

2 | 7 → 0

2 | 3 → 1

1 → 1

3) 0·41×2 = 0·82 $(0.41)_{10}$ = ~~$(0.01)_2$~~ $(0.011)_2$!

0·82×2 = 1·64

0·64×2 = 1·28 0·28×2 = 0·21856

4) 0·67×2 = 1·34 $(0.67)_{10} = (0.1010)_2$

0·34×2 = 0·68

0·68×2 = 1·36

0·36×2 = 0·72

5) $(15.32)_{10}$

$$\begin{array}{l|l} 2 & 15 \\ 2 & 7 \rightarrow 1 \\ 2 & 3 \rightarrow 1 \\ 2 & 2 \rightarrow 1 \\ & 1 \end{array}$$

$(15)_{10} = (1111)_2$

$0.32 \times 2 = 0.64$

$0.64 \times 2 = 1.28$

$(15.25)_{10} = (1111.01)_2$

6) $(37.45)_{10}$

$$\begin{array}{l|l} 2 & 37 \\ 2 & 18 \rightarrow \cancel{1}\ 1 \\ 2 & 9 \rightarrow \cancel{0}\ 0 \\ 2 & 4 \rightarrow \cancel{0}\ 1 \\ 2 & 2 \rightarrow \cancel{0}\ 0 \\ & 1 \quad 0 \end{array}$$

$0.45 \times 2 = 0.9$

$0.9 \times 2 = 1.8$

$(37.45)_{10} = \cancel{(10010.01)_2}\ (100101.011)_2$

BINARY TO DECIMAL CONVERSION

MSB- Most Significant bit   Eg: 456 → MSB-4 , 6 - LSB

LSB- Least Significant bit

1) Convert the binary number $((1101)_2 \rightarrow (\ )_{10})$ to decimal

Ans) 1 1 0 1 →

$\downarrow 2^3 \quad \downarrow 2^2 \quad \downarrow 2^1 \quad \downarrow 2^0$

MSB   LSB

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

$= 1 + 0 + 8 + 4$

$= \underline{\underline{13}}$

2) Convert the binary number $(.1011)_2 \rightarrow (\ )_{10}$

Ans) $(.1011)_2$

$2^{-1} \quad 2^{-2} \quad 2^{-3} \quad 2^{-4}$

$\uparrow \quad \uparrow \quad \uparrow \quad \uparrow$

1 0 1 1

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

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

$= 0.5 + 0 + 1\times0.125 + 1\times0.0625$

$= (0.6875)_{10}$

$(0.1101)_2 \rightarrow (0.6875)_{10}$

3? Convert the binary number $(101.101)_2 - ()_{10}$

$2^2 \; 2^1 \; 2^0 \quad 2^{-1} \; 2^{-2} \; 2^{-3}$

$1 \; 0 \; 1 \cdot 1 \; 0 \; 1$

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

$= 4 + 0 + 1 + 0.5 + 0.125$

$= (5.625)_{10}$

$(101.101)_2 - (5.625)_{10}$

4? Convert the binary number $(1011.111)_2 \rightarrow ()_{10}$ to decimal.

Ans $2^3 \; 2^2 \; 2^1 \; 2^0 \quad 2^{-1} \; 2^{-2} \; 2^{-3}$

$1 \; 0 \; 1 \; 1 \cdot 1 \; 1 \; 1$

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

$8 + 0 + 2 + 1 + 0.125 + 0.25 + 0.5$

$= (11.875)_{10}$

DECIMAL TO HEXADECIMAL CONVERSION $(()_{10} \rightarrow ()_{16})$

1? Convert the decimal number $(75.432)_{10} \rightarrow ()_{16}$

Ans)

$16 \,|\, 75$

$\;\;\; 4 \rightarrow 11 \; (B) \uparrow$

$(75)_{10} \rightarrow (4B)_{16}$

0. 4 3 X 16 = 6.88 → 6

0. 8 8 X 16 = 14.08 – E

.0.8 X 16 = ~~12.0~~1.28 → 1

0.28 X 16 = 4.48 4

$(0.43)_{10} = (6E14)_{16}$

$(75.43)_{10} = (4B \cdot 6E14)_{16}$

Q2. $(29.48)_{10} \to (\ )_{16}$

16|29.
1 → 13

$(29)_{10} = (1D)_{16}$

16 | 29 (quotient 1)
 16
 13

0. 48

0.48 X 16 = 7.68 → 7

0.68 X 16 = 10.88 → A

0.88 X 16 = 14.08 → E

0.08 X 16 = 1.28 → 1

$(0.48)_{10} \to (7AE1)_{16}$

$(29.48)_{10} \to (1D \cdot 7AE1)_{16}$

HEXADECIMAL TO DECIMAL CONVERSION

1? Convert the hexadecimal number to decimal number

1) $(27.42)_{16} \to (\ )_{10}$

$16^1 \quad 16^0 \quad 16^{-1} \quad 16^{-2}$

2 7 . 4 2

$= 2 \times 16^1 + 7 \times 16^0 + 4 \times 16^{-1} + 2 \times 16^{-2}$

$= 32 + 7 + 4 \times 0.06 + 2 \times 0.\cancel{031}$ 3.966

papergrid

Date: / /

$= 32 + 7 + 0.25 + 0.0078$

$= (39.2578)_{10}$

2) $(3FB.4A)_{16} \rightarrow (\ )_{10}$

16^{2} 16^{1} 16^{0} 16^{-1} 16^{-2}

3 F B . 4 A

~~32~~

$3\times16^{2} + 15\times16^{1} + 11\times16^{0} + 4\times16^{-1} + 10\times16^{-2}$

$768 + 240 + 11 + 4\times0.06 + 0.039$

$= (1019.2890)_{10}$

HW 1) $(10111.10111)_2 - (\ )_{10}$

2) $(11001.011)_2 - (\ )_{10}$

3) $(79.88)_{10} - (\ )_{16}$

4) $(CA3.9B)_{16} - (\ )_{10}$

5) $(F39.8AB)_{16} - (\ )_{10}$

1) Ans) 1 0 1 1 1 . 1 0 1 1 1

2^{4} 2^{3} 2^{2} 2^{1} 2^{0} 2^{-1} 2^{-2} 2^{-3} 2^{-4} 2^{-5}

$1\times2^{4} + 0\times2^{3} + 1\times2^{2} + 1\times2^{1} + 1\times2^{0} + 1\times2^{-1} + 0\times2^{-2} + 1\times2^{-3} + 1\times2^{-4} + 1\times2^{-5}$

$16 + 0 + 4 + 2 + 1 + 0.5 +$ ~~0.2~~ $0 + 0.125 + 0.0625 + 0.03125$

$23 + 0.71875 = \underline{23.71875}$

2) $(11001.011)_2 \rightarrow (\ )_{10}$

1 1 0 0 1 . 0 1 1

~~2^{3} 2^{2} 2^{1} 2^{0}~~

$1\times16 + 1\times8 + 0\times4 + 0\times2 + 1\times1 + 0\times2^{-1} + 1\times2^{-2} + 1\times2^{-3}$

1×8 ~~$+ 1\times4 + 0\times2 + 0\times1 + 0\times2^{-1} + 0\times2^{-2} + 0\times2^{-3}$~~

~~$8+4+0+0.25+0.125$~~ $16+8+0+0+1+0+0.25+0.125$

~~12 + 0.375 = 12.375~~ $(25.375)_{10}$

3) $(79.88)_{10} \rightarrow (\;)_{16}$

$16 \,\underline{|\, 79}$

$4 \rightarrow 15$ $\quad (79)_{10} \rightarrow (4F)_{16}$

$0.88 \times 16 = 14.08$

$0.08 \times 16 = 1.28$ ~~14 E.1247~~

$0.28 \times 16 = 4.48$

$0.48 \times 16 = 7.68$

10 - A
11 - B
12 - C
13 - D
14 - E

$(79.88)_{10} = (4F.E147)_{16}$

4) $(CA3.9E)_{16} \rightarrow (\;)_{10}$

C A 3 . 9 E

12^{2} 16^{1} 16^{0} 16^{-1} 16^{-2}

$12 \times 16^{2} + 10 \times 16^{1} + 3 \times 1 + 16^{-1} \times 9 + 16^{-2} \times 14$

$3072 + 16 + 3 + 0.5625 + 0.546$

$3235 + 0.6171875$

$(CA3.9B)_{16} = (3235.61718)_{10}$

5) $(\overset{F}{E}39.8AB)_{16} \rightarrow (\;)_{10}$

$\overset{F}{E}$ 3 9 . 8 A B

16^{2} 16^{1} 16^{0} 16^{-1} 16^{-2} 16^{-3}

~~3584~~ $+ 48 + 9 + 0.5 + 0.390625 + 11 \times 16^{-3}$ (0.0026855

3840

$(F39.8AB)_{16} \rightarrow$ (~~3897.3641 5417~~) $(3897.541748)_{10}$

BINARY TO HEXADECIMAL CONVERSION $()_2 \rightarrow ()_{16}$

1) Convert the binary number $(11011.11101)_2 \rightarrow ()_{16}$

Ans)

Decimal		Binary
0		0000
1		0001
2		0010
3		0011
4		0100
5		0101
6		0110
7		0111
8		1000
9		1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

(By adding $2^3, 2^2, 2^1, 2^0$ we should get the decimal number)

0 0000

1 0010-2

0100-4

0111 → ~~07~~ 1 1000 1100

J → L → R

I → R → L

$$\underbrace{0001}_{1}\ \underbrace{1011}_{B}\ .\ \underbrace{1110}_{\substack{14\\(E)}}\ \underbrace{1000}_{8}$$

$(11011.11101)_2 \rightarrow (1B.E8)_{16}$

$$\begin{array}{r} 1100+ \\ 1 \\ \hline 1101 \end{array} \qquad \begin{array}{r} 1011+ \\ 111 \\ \hline 1100 \end{array}$$

$$\begin{array}{r} 10+ \\ 1 \\ \hline 11 \end{array} \quad \begin{array}{r} 11+ \\ 11 \\ \hline 100 \end{array} \quad \begin{array}{r} 100+ \\ 1 \\ \hline 101+ \\ 11 \\ \hline 110 \end{array} \quad \begin{array}{r} 110+ \\ 1 \\ \hline 111 \end{array}$$

2) Convert the ~~decimal~~ binary number $(11011011.11100101)_2 \rightarrow ()_{16}$

Ans) $\overset{2^3}{1}\overset{2^2}{1}0\overset{2^0}{1}$

$$\underbrace{1101}_{D}\ \underbrace{1011}_{B}\ .\ \underbrace{1110}_{E}\ \underbrace{0101}_{5}$$

$(11011011.11100101)_2 \rightarrow (DB.E5)_{16}$

HEXADECIMAL TO BINARY $()_{16} \rightarrow ()_2$

1? Convert the hexadecimal number $(AB3.7F)_{10} \rightarrow ()_2$

Ans) A B 3 7 F

1010 1011 0011 0111 1111

$(AB3.7F)_{10} \rightarrow (101010110011.01111111)_2$

2) Convert hexadecimal number $(1BE.C2E)_{16} \rightarrow ()_2$

Ans 1 B E C 2 E

0001 0011 1110 1100 0010 1110

$(1BE.C2E)_{16} \rightarrow (000110111110.110000101110)_2$.

BINARY CODED DECIMAL (BCD)

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

BCD is also called as 8421 code

In binary coded decimal each decimal digit is represented with 4 bits. BCD code is also called as 8421 code

1? Write the decimal number $(28.59)_{10} \rightarrow (\ )_{BCD}$

Ans) 2 8 5 9

0010 1000 0101 1001

$(28.59)_{10} \rightarrow (00101000001011001)_{2}$.

2? ADDITION OF BCD NUMBERS

For adding 2 BCD numbers if the result is grless than or equal to 9, the answer is the result itself. But of the result is greater than 9. We have to add +6 (0110) to the result.

Add 25 and 15 BCD

2 5 1 5

0010 0101 0001 0101

```
        1       1
0 0 1 0 0 1 0 1 +
0 0 0 1 0 1 0 1
----------------
0 0 1 1 (1 0 1 0) → >9
         0 1 1 0
----------------
0 0 0 0 0 0 0 0
        1 1
```

1+1=0,1

RULES FOR BINARY ADDITION

A	B	Sum	Carry
0	0	0	0
0	1	1	0

? Perform the action of 24 and 18 using BCD

Ans 2 4 1 8

0010 0100 0001 1000

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

```
        1
0 0 1 1 1 1 0 0 +
  1 1 1 0 1 1 0
----------------
0 1 0 0 0 0 1 0
   4       2
```

Q4+18 in BCD 0100 0010

32 Add 175 and 326 using BCD

1 7 5 . 3 2 6

0001 0111 0101 0011 0010 0110

```
   1 1   1 1     1          1 1
 0 0 0 1 0 1 1 1 0 1 0 1 +  175+
 0 0 1 1 0 0 1 0 0 1 1 0    326
 0 1 0 0 1 0 0 1 1 0 1 1    501
   4       9       11

                 1 1
 0 1 0 0 1 0 0 1 1 0 1 1 +
             1 1 0 1 1 0
 0 1 0 0 1 0 1 0 + 0 0 0 1 +
       1 0 1 1 0       1
 0 1 0 1 0 0 0 0
   5       0       1
```

42 Add 589 and 1 9 9 using BCD

Ans 5 8 9 1 9 9

0101 1000 1001 0001 1001 1001

```
     1 1       1 1     1
 0 1 0 1 1 0 0 0 1 0 0 1 +
 0 0 0 1 1 0 0 1 1 0 0 1       Carry → consider 79
 0 1 1 1 0 0 1 0 0 0 1 0
     ~~8~~7      ~~10~~2     ~~10~~2

             1 1
 0111 0010 0010 +
           0110
 0111 0010 + 1000
      0110
 0111 1000 1000
   7    8    8
```

BCD SUBSTRACTION

Digit	9's complement
0	9
1	8
2	7
3	6
4	5

5	4
6	3
7	82
8	1
9	0

In the case of BCD Substraction for 2 numbers; we have to take the 9's complement of the 2nd number and add to the first number. In this process a carry is generated and add that carry to the answer to get the final result.

Eg: Normal substraction: 8 − 2 = 6 BCD Substraction − 8 + 7 = 5 + 1 = 6

12. Perform the substraction of 79 and 26 in BCD

Ans) 79 − 26 = 53

```
  0111 1001 +
  0111 0011
  1110 1100 (12)
       0110 → 6
  1111 0010
  0110
  0101 0010 +
          1
  0100 0011
   5    3
```

Q. Perform the Substraction of 89 and 54 in BCD

Ans) 89 − 54 = 35

```
      1000 1001 +
4 ← 0100 0100 → 5
      1100 1110 + (14)
           0110
      1101 0100 +
      0110 ~~0110~~
      0001 ~~1010~~

0011 0100 +
        1 → Carry add to that
0011 0101
 3    5
```

papergrid

Date: / /

HW

32. Perform the substraction of 125 and 13 in BCD

```
Ans  125 -
      13 → (9's complement)      0001   0010    0101
     ----                        1000   1000    0110 (6)
     112                         ----------------------
                                 1010   1010    1011
                                                0110
                                 ----------------------
                                 1010   1011 +  0001
                                        0110
                                 ----------------------
                                11011 + 0001    0001 +
                               - 0110              1
                                 ------------   ------
                                 0001   0001    0010
                                   1      1       2.
```

EXCESS 3 CODE

It is a modified form of BCD code in which we have to add 3 to the decimal digit.

Eg the excess code of 592

```
592 - 0101   1001   0010
      1000   1100   0101
```

EXCESS 3 ADDITION

On the case of excess 3 addetion, If we add 2 excess 3 numbers and if a carry is generated add 3 to the answ and if there is no carry generated substract 3 to the number

12. Perform the action of 8 and 6 using excess 3

```
Ans) + 8 -    1000 }  ~~9's complement~~ excess 3
       6      1001 }
     ---      ----
     14 0001  0100
        0011  0011
        ----------
        0100  0111
         4      7  →  ~~9's compleme~~ excess 3 reverse
         1      4
```

55-
32

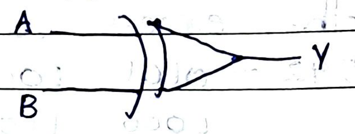
32 11 0 1 1 1
1 1

GRAY CODE

Gray code is an unit distance code in which the bit pattern for 2 consecutive numbers differ in only one bit position. It is also known as cyclic codes

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

Application:-
Truth Table EX-OR (Exclusive OR Gate)



Input	Output
A B	$Y = A \oplus B$
0 0	0
0 1	1
1 0	1
1 1	0

GRAY TO BINARY CONVERSION

1) Convert the gray code $(101011) \rightarrow ()_2$

Ans) Grey code: 101011
MSB

MSB

Grey code: 1 0 1 0 ~~h~~ 1 1 (Write MSB as it is)

Binary code 1 1 0 0 1 0

Ans = $(110010)_2$

2? Convert the grey code 11010111 to its binary equivalent.

Ans) 1 1 0 1 0 1 1 1

1 0 0 1 1 0 1 0 Binary code → $(10011010)_2$.

BINARY TO GREY CONVERSION

1? Convert to binary code 10011010 to its equivalent grey code.

Ans) 1 → 0 → 0 → 1 → 1 → 0 → 1 → 0 MSB as it is

↓

1 1 0 1 0 1 1 1

2) Convert binary number 10111011 To its equivalent grey code

Ans $1 \oplus 0 \oplus 1 \oplus 1 \oplus 1 \oplus 0 \oplus 1 \oplus 1$

↓

1 1 1 0 0 1 1 0

ALPHA NUMERIC CODES

These types of codes consist of both numbers and alphabetic characters The commonly used alphanumeric codes are

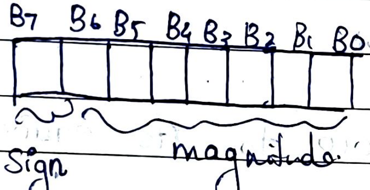
1) ASCII → American standard code for Information and Interchange

2) EBCDIC → Extended Binary Coded decimal Interchange code

- ★ ASCII is a seven bit code in which decimal digits are represented by BCD code preceded by 011
- ★ The EBCDIC code is represented by BCD code preceeded by 1111 in the case of decimal digits → It is a 8 bit code.

SIGNED BINARY NUMBERS

The unsigned numbers represent only magnitude. But in the case of signed binary numbers, they are represented with sign magnitude format which is given below.



Eg +6 → 00000110

−6 → 10000110

1). Eg : +24

B7	B6	B5	B4	B3	B2	B1	B0
0	0	0	1	1	0	0	0

2 | 24 → 0
2 | 12
2 | 6 → 0
2 | 3 → 0
1 → 1

−64

2. Convert dec ~~(24)~~$_{10}$ → ( )$_2$ to its signed binary format (1000)

Ans) ~~0~~ 11000000

max +ve number − 127 − 01111111

min −ve number −128 − 11111111

2 | 64
2 | 32 → 0
2 | 16 → 0
2 | 8 → 0
2 | 4 → 0
2 | 2 → 0
1 → 0

1000000

ONE'S COMPLEMENT REPRESENTATION

One's complement of a binary number when we change ~~one~~ 1's to 0's and 0's to 1's

1) ~~Convert~~ Find ~~fr~~ 1st complement of $(101011)_2$

Ans $(010100)_2$.

2's COMPLEMENT

2' complement of a number is a number when we add 1 to 1's complement.

2) Find 2's Complement of 1010111

Ans 1st's complement → $\begin{array}{r} 0101000 \\ +1 \\ \hline \end{array}$

2's complement 0101001

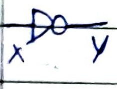
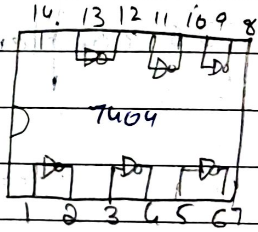
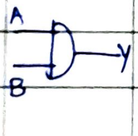
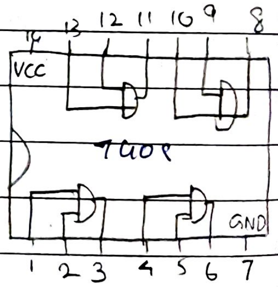
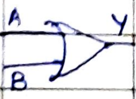
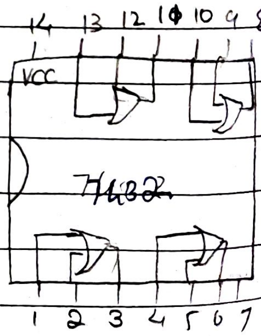
MODULE-2

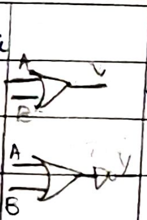
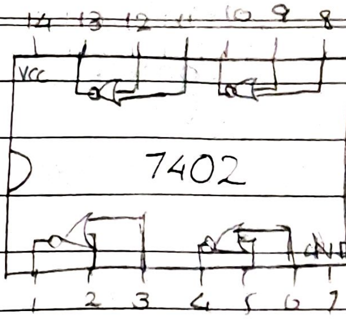
LOGIC GATES

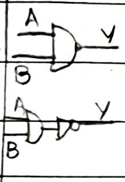
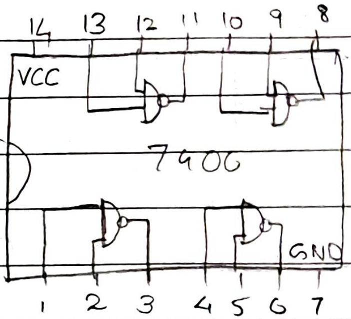
Logic gates are the basic elements that make up a digital System. It is able to operate no. of binary inputs to perform a particular logic function

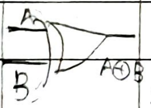
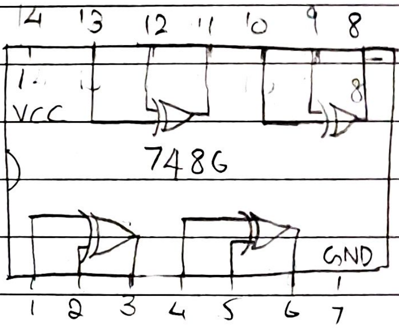
The different gates

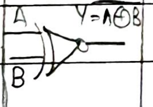
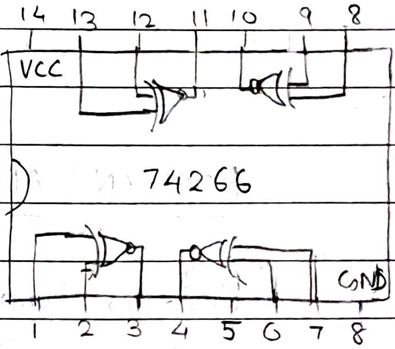
- NOR gate
- AND
- OR
- NAND
- EX-OR
- EX-NOR

	Gates	Symbol	Expression	Truth Table			IC - Structure
1.	NOT gate Inverter gate		$Y = \bar{A}$	Input		Output	
				A		$Y = \bar{A}$	
				0		1	
				1		0	
2.	AND gate		$Y = A \cdot B$	Input		Output	
				A	B	$Y = A \cdot B$	
				0	0	0	
				0	1	0	
				1	0	0	
				1	1	1	
3	OR gate		$A + B$	Input		Output	
				A	B	$Y = A + B$	
				0	0	0	
				0	1	1	
				1	0	1	
				1	1	1	

				Input	Output	
4.	NOR Gate		$Y=\overline{A\$B}$	A B	$Y=\overline{A+B}$	
				0 0	1	
				0 1	0	
				1 0	0	
				1 1	0	

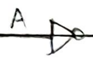
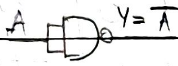
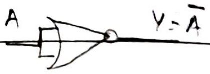
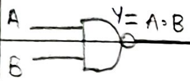
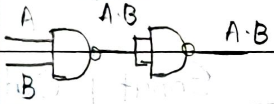
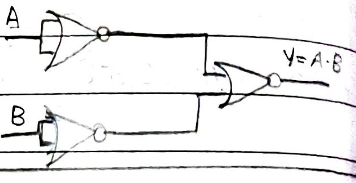
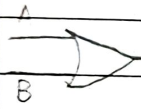
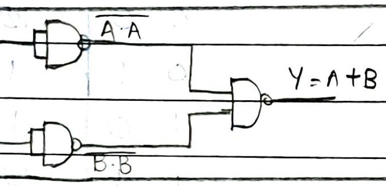
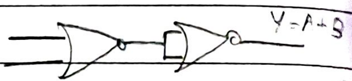
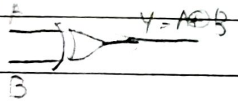
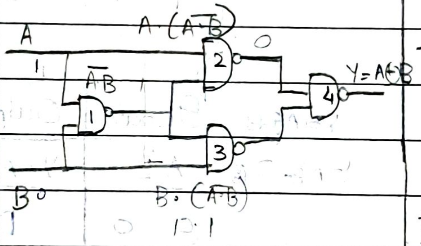
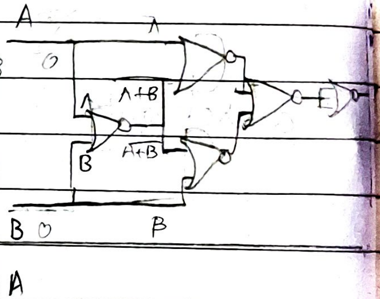
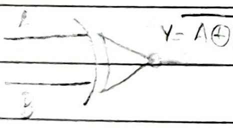
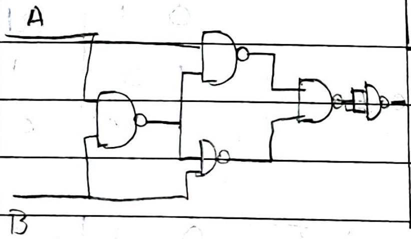
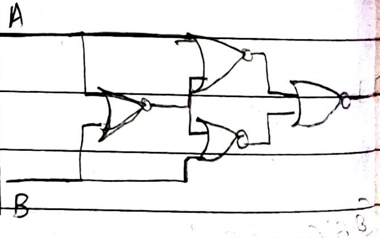
				Input	Output	
5	NAND Gate		$Y=\overline{A\cdot B}$	A B	$Y=\overline{A\cdot B}$	
				0 0	~~0~~ 1	
				0 1	1	
				1 0	1	
				1 1	0	

				Input	Output	
6	XOR gate		$Y=A\oplus B$	A B	$Y=A\overline{B}+\overline{A}\overline{B}$	
		$=A\bar{B}+A\overline{B}$	$=A\overline{B}+\overline{A}B$			
				0 0	0	
				0 1	1	
				1 0	1	
				1 1	0	

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

UNIVERSAL GATES

NAND and NOR are called universal gates because any combination of gate such as not, and or, EX-NOR, EX-OR can be implement with these gates.

	Gate	NAND equivalent	NOR equivalent
1.	 $\overline{A}$ NOT gate	 $Y=\overline{A}$	 $Y=\overline{A}$
2.	 $Y=A\cdot B$ AND gate	 $A\cdot B$	 $Y=A\cdot B$
3.	 OR gate	 $Y=A+B$	 $Y=A+B$
4	 $Y=A\oplus B$	 $Y=A\oplus B$	
5.	 $Y=\overline{A\oplus B}$ X-NOR		

BOOLEAN ALGEBRA

Boolean algebra is used to simplify or rearrange boolean equation to make a simple logic circuit.

LAWS OF BOOLEAN ALGEBRA

1) Commutative law

LAW-1: $A+B=B+A$

Law 2 : $AB = BA$

2. Associative law

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

Law 2: $A(BC) = (AB)C$

3. Distributive Law

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

RULES IN BOOLEAN ALGEBRA

RULE NO	RULES
1	$A+0=A$
2	$A+1=A$ $(A+B+C+E+1=1)$
3	$A\cdot 0=0$
4	$A\cdot 1=A$
5	$A+A=A$
6	$A+\overline{A}=1$
7	$A\cdot A=A$
8	$A\cdot\overline{A}=0$
9	$\overline{\overline{A}}=A$
10	$A+AB=A$
11	$A+\overline{A}B=A+B$
12	$(A+B)(A+C)=A+BC$

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

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

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

DE MORGAN'S LAW

Theorem 1

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

2) $\overline{AB} = \overline{A}+\overline{B}$

$\overline{A\cdot B}\cdot A\cdot\overline{AB}\,B$

$\overline{\overline{AB}A\cdot\overline{AB}\,B}$

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

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

Prove that $\overline{AB} = \overline{A}+\overline{B}$

Ans?

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

LHS=RHS

Q2 Simplify the boolean expression $Y= A\cdot\bar{A}C$

Ans) $Y= A\cdot\bar{A}C$

$= 0\cdot C \quad (A\cdot\bar{A}=0)$

$= 0 \quad \cdot((A\cdot 0)=0)$

3? Simplify $Y= ABCD + ABC$

Ans $Y= ABCD + ABC$

$= ABC(D+1) \quad (A+1=1)$

$= ABC\cdot 1 \quad (A\cdot 1=A)$

$= ABC$

3? Simplify $Y= ABCD + A\bar{B}CD$

Ans) $ACD(\bar{B}+B)$

$= ACD\cdot 1 \quad (A+\bar{A}=1)$

$= ACD \quad (A\cdot 1=A)$

4? $Y= A(A+B)$

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

$= A+AB$

$= A$

5? $Y= AB + ABC + AB(D+E) \quad ABD+ABE$

$= AB(1+C+$ ~~A~~$D+E)$

$= AB \quad (A+1=1)$

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

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

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

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

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

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

$= 1 + \bar{C} + B$

$= 1$

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

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

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

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

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

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

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

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

$= B$

12) Simplify $Y = \overline{\overline{A\bar{B} + ABC} + A(B + \cancel{B}\bar{B})}$

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

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

$\overline{A + B + ABC + AB}$

$\overline{A\cancel{(1+}\,A + B + A\cancel{B}(\overset{AB}{C} + B)}$

$\overline{A + B + AB}$

$\overline{AB} + AB$

$BC + B$

B

$\overline{\overline{AB}} + AB$

$AB + \overline{AB}$

$A(BC + 1) + B$

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

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

$= \overline{\overline{AB} + \overline{ABC} + AB + A}$

$= \overline{\overline{AB} + \overline{A}BC + A}$

$A + B + ABC$

$= A + B' + ABC$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$= \underline{\underline{A+B+C+D}}$

Date: / /

SUM OF PRODUCTS FORM

Sum of productterm is a group of product terms ORed ~~ORed~~ together

Eg: SOP – $Y = AB + BC + AC$, $Y = A\bar{B}C + \bar{A}C + AB\bar{C}$

POS – $Y = (A+B)(A+C)(B+C)$

Group of sum terms ~~and~~ ANDed together thas is POS and SOP.

STANDARD SOP OR CANONICAL SOP

Eg: $AB\bar{C} + A\bar{B}C + \bar{A}BC$

If each term in SOP and POS contain all the literals then they are called as standard canonical SOP AND POS form.

Literal :- Each variable in a SOP or POS form is called a literal.

12. Convert boolean expression $Y = AB + BC + AC$ into standard and canonical form.

Ans) $Y = AB(C+\bar{C}) + BC(A+\bar{A}) + CA(B+\bar{B})$

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

2? $Y = A + AB + ABC$

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

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

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

2 $Y = AB + A\bar{B}C + AC\bar{D} + A\bar{B}CD$ → to standard SOP form

$Y = AB(C+\bar{C})(D+\bar{D}) + A\bar{B}C(D+\bar{D}) + A\bar{C}D(B+\bar{B})$
$+ ABCD$

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

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

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

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

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

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

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

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

MINTERMS AND, MAXTERMS

Each individual term is standard SOP → Minterms

Each individual term is standard POS → Mixterm

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

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

For an n-variable logic fn there are 2^n miniterms and maxterms

for $n=2$, we have $2^2=4$ miniterm / maxterm

$n=3$, We have $2^3=8$ minterm / maxterm

Draw the table for midterm and maxterm for 3 variable logic fn

Let variable be a,b,c

Variables			Midterm	Maxterm
A	B	C	m o	m o
0	0	0	$\bar{A}\bar{B}\bar{C} = m_0$	$(A+B+C) = M_0$
0	0	1	$\bar{A}\bar{B}C = m_1$	$(A+B+\bar{C}) = M_1$
0	1	0	$\bar{A}B\bar{C} = m2$	$(A+\bar{B}+C) = M2$
0	1	1	$\bar{A}BC = m_3$	$(A+\bar{B}+\bar{C}) = m_3$
1	0	0	$A\bar{B}\bar{C} = m4$	$(\bar{A}+B+C) = m4$
1	0	1	$A\bar{B}C = m5$	$(\bar{A}+B+C) = m5$

1 1 0 $\quad A B \bar{C} = m6 \quad (\bar{A} + \bar{B} + \bar{0}) = m6$

1 1 1 $\quad A B C = m7 \quad (\bar{A} + \bar{B} + \bar{C}) = m7$

Q) Write the shorthand notation of the expression

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

$Y = m5 + m2 + m1$

$Y = \Sigma m(1,2,5)$

Q) Write the shorthand notation of

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

$Y = M2 + M5 + M3$

$Y = M(2,3,5)$

$\Sigma \rightarrow$ Minterm

λ — Maxterm

Simplify the following expression using boolean expression

1) $Y = \Sigma M(1,3,5,7)$

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

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

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

$Y = 1 \cdot C$

$Y = C$

2) $Y - \Sigma M(0,1,2,3,4,6)$ Simplify using boolean algebra

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

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

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

$> \bar{A}B + \bar{C}(A + \bar{A}B) \; A+B$

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

?. Simplify $Y = \Pi M(3,5,7)$

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

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

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

OR

$$y = \Sigma m(0,1,2,4,b)$$

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

KARNAUGH MAP OR K-MAP OR VEITCH DIAGRAM

K-map is a systematic approach for simplifying any boolean expression. K MAP is a graphical chart contain boxes called cells. Each cell represents one of 2^n possible combinations that can be formed from n variables. Therefore a 2 variable k-map contains $2^2 = 4$ cells, A 3 variable k-map contain $2^3 = 8$ cells

A 4 variable k-map contains 2^4= cells =16cells and so

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

2 Variable

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

3 Varrable

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

4 Variable

A\B	0	
$\bar{A}$	00	01
A	10	11

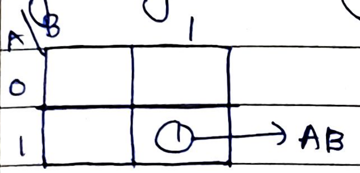
A\B	00	01	11	10
0	000	001	011	011
1	100	101	111	110

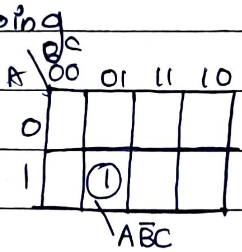
AB\CD	00	01	11	10
00	0000	0001	0011	0010
01	0100	0101	0111	0110
11	1100	1101	1111	1110
10	1000	1001	1011	1010

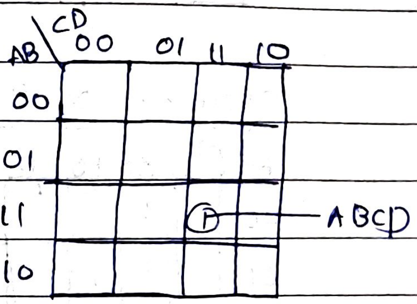
Grouping of cells

We can group the cells in K-MAP in the order of $2^0, 2^1, 2^2, 2^3, 2^4 \ldots$ ie; 1, 2, ~~3~~, 4, 8, 16 ....

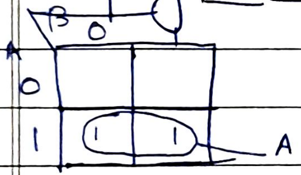
Eg: Single sell grouping

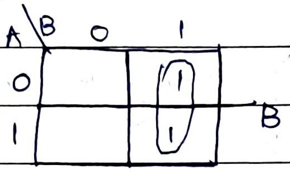






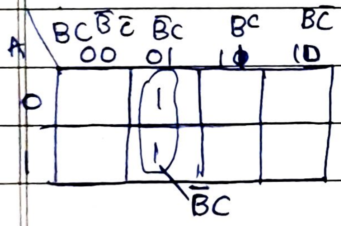
Grouping of 2 cells

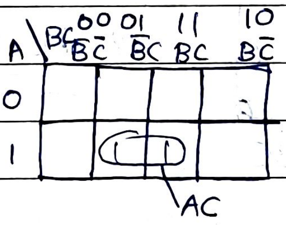


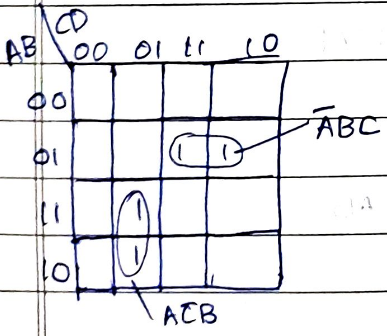


0,1 (0 and its complement) gt cancel each other

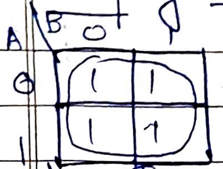
11, 10 (1,0 cancel each other)

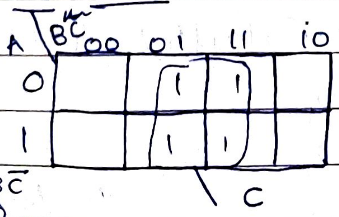


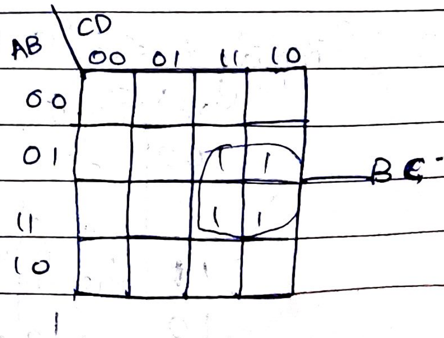


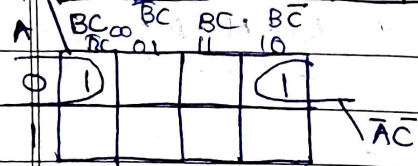


Grouping of 4 cells









Date:

AB \ CD	00	01	11	10
00		1	1	
01				
11				
10		1	1	

$\bar{B}D$

AB \ CD	00	01	11	10
00	1			1
01				
11				
10	1			1

$\bar{B}\bar{D}$

Grouping of 8 cells

AB \ CD	00	01	10	11
00		1	1	
01		1	1	
16		1	1	
11		1	1	

D

1) Simplify $Y = \Sigma m(1,3,7)$ using K MAP

Ans 3 variable K MAP

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

$\bar{A}C$

BC

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

2) Simplify $Y = \Sigma m(1,3,6,7)$ using K MAP

Ans)

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

$\bar{A}C$

AB

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

3) Simplify $Y = \Sigma m(9,10,11,12,13,15)$ 4 Variable K-Map 2^4

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

AC

AB

AD

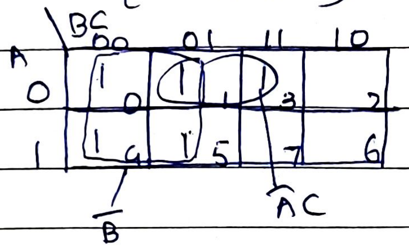
$Y = AB + AD + AC$

4) Minimize the expression $Y = A\bar{B}C + \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}\bar{C} + \bar{A}\bar{B}\bar{C}$ using K MAP

Ans) $Y = \bar{A}\bar{B}C + \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}\bar{C} + \bar{A}\bar{B}\bar{C}$

$= m5 + m1 + m_3 + m4 + m0$

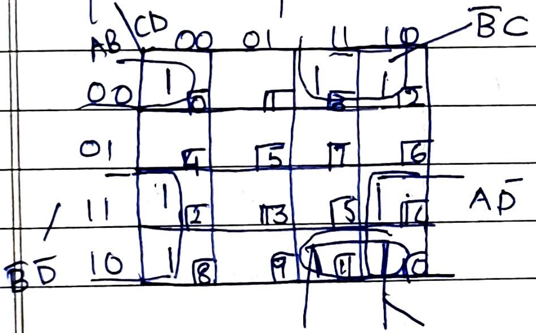
$= \Sigma m(0,1,3,4,5)$



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

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

5) Reduce expression $Y = \Sigma m(0,2,3,8,10,11,12,14)$

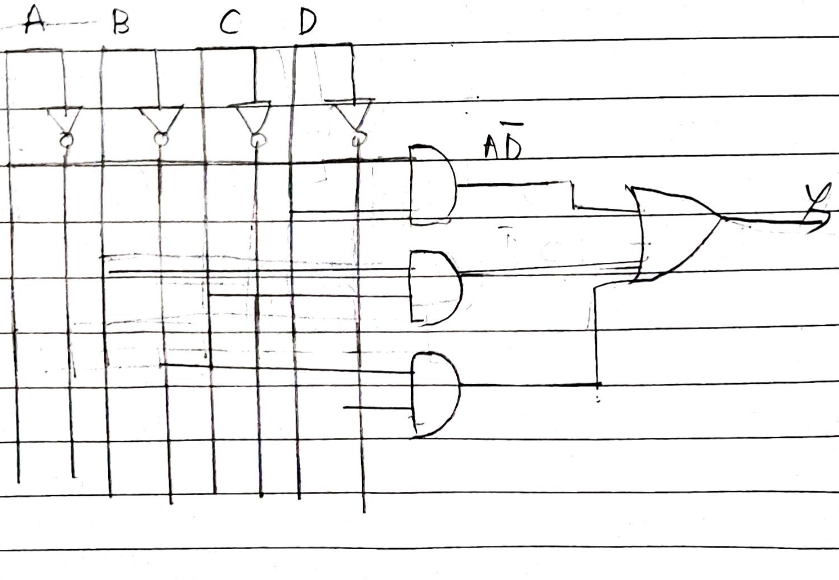


AB \ CD	00	01	11	10
00	1 (0)	(1)	1 (3)	1 (2)
01	(4)	(5)	(7)	(6)
11	1 (12)	(13)	(15)	1 (14)
10	1 (8)	(9)	1 (11)	1 (10)

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

6) Reduce the above expression using K MAP and draw the logic circuit.

Ans) $Y = \Sigma m(0, 2, 3, 8, 10, 11, 12, 14)$



7. Reduce the expression $Y = \Sigma m(1, 3, 4, 5, 11, 12, 13, 15)$

Ans)

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

$\bar{A}D$

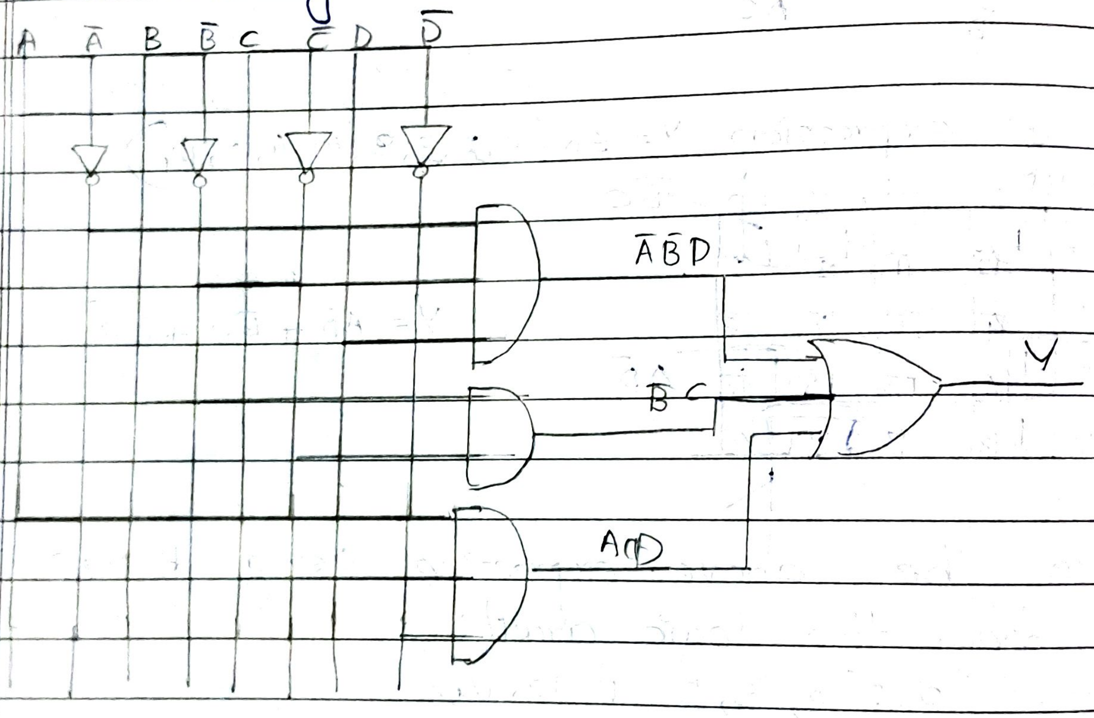
~~$AB\bar{C}D$~~

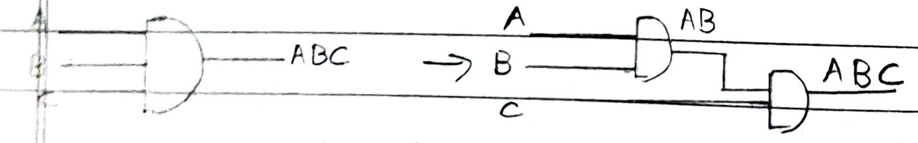
$B\bar{C}$

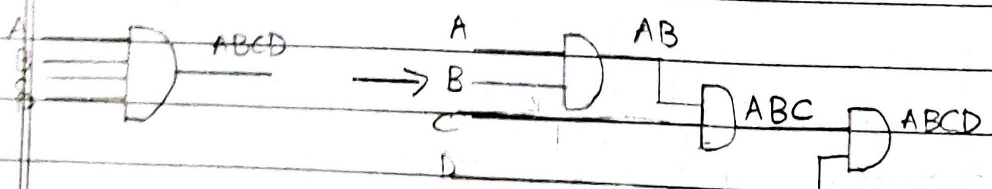
AD

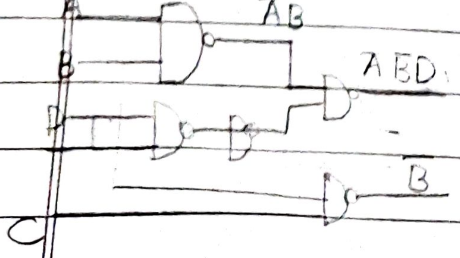
$Y = \bar{A}\bar{B}D + \bar{B}C + AD$

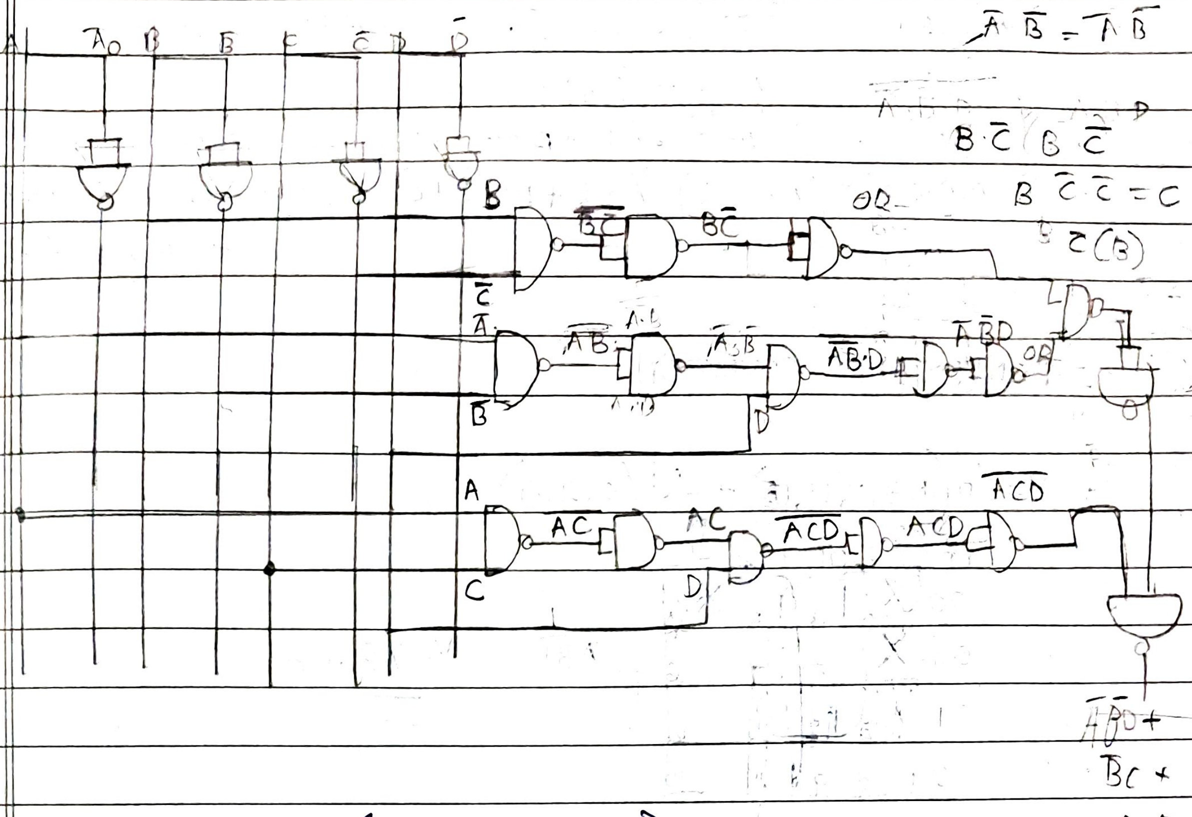
Draw the logic circuit



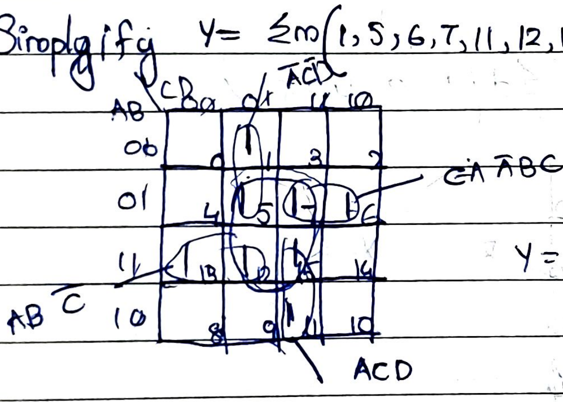






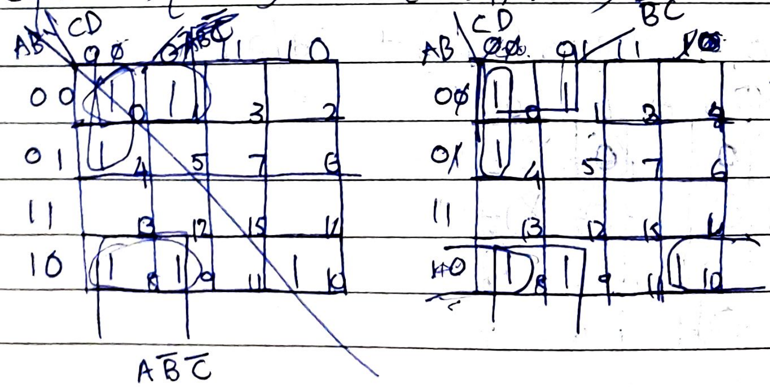


2. Simplify Y = Σm(1,5,6,7,11,12,13,15)



Y = ĀBC + ACD + ABC̄ + ĀC̄D

Reduce f(ABCD) = Σm(0,1,4,8,9,10)



ĀC̄D̄ + B̄C̄ +
~~ĀB̄D̄~~ AB̄D̄

Date:

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

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

$A+C+D$ ~~ACD~~ $\cdot\ \bar{A}+B+D$ ~~$\cdot BC$~~ $B+C$

$f(A,B,C,D) = (A+C+D)(\bar{A}+B+D)(B+C)$ (Demorgan's Theorem) $\pi m = 0$, $\Sigma m = 1$

DON'T CARE CONDITION

1) $Y = \Sigma m(1,3,7,11,15) + \Sigma d(0,2,4)$

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

wait x → 1 x = 1

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

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

2) $Y = \Sigma m(5,6,7,12,13) + \Sigma d(4,9,14,15)$

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

$Y = B$

Reduce the following function
$Y = \pi m(0,3,4,7,8,10,12,14) + \Sigma d(2,6)$

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

AB\CD	00	01	11	10
00	0		0	X
01	0		0	X
11	0			0
10	0			0

→ $\bar{D}$

→ $\bar{A}C$

→ $\bar{D} + \bar{A}C$

σm → X → 1

πm → X → 0

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

$Y = \bar{\bar{Y}} = \overline{\bar{D} + \bar{A}C}$

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

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

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

42 $Y = A + AB + AC$

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

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

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

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

$Y = \pi m(0, 2, 1, 3)$

A\B	00	01	10	10
0	$_0$	$_1$	$_3$	$_2$
1	1_4	1_5	1_7	1_6

→ A

1. MODULE-1

1's and 2's COMPLEMENT SUBSTRACTION

1? Substract 25 − 15

Ans $(25)_{10} \rightarrow (10100)_2$

$(15)_{10} \rightarrow (1011)_2$

110

2 | 15
2 | 7 → 1
2 | 3 → 1
2 | 2 → 1
1 → 1

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

2 | 25 → 1
2 | 12 → 1
2 | 6 → 0.
2 | 3 → 0.
2 | 2 → 1

```
11001+
10000
-----
01001+
    1
-----
01010
```

Steps for one's complement substraction.

CASE-1

If the second number is smaller than the first number

1) Find the one's complement of the smaller number
2) Add this number to the larger number
3) Remove the carry and add it to the the result. This carry is called end around carry.

12. Substract 55 from 67 using one's complement substraction.

Ans) Step-1 → 1's complement of smaller number

55 → 110111   67 - 1000011

55 1's complement ~~010000~~ 001000

```
 1000011+
 1001000
 -------
 0001011
     ①①1
 -------
 0001100 → 1
```

```
2|55          2|55
2|27 → ①      2|27 → 1
2|13 → ①       2|13 → 1
2|6 → ①        2|6 → 1
2|3 → 0         2|3 → 0
 1 → 1            1 → 0

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

10
100
100
0110
```

CASE-2 1^{st} number is smaller than the second number

1) Find 1's complement of the larger number or second number add this to the smaller number

2) The answer is the one's complement of the true result and is opposite in sign also ther is no carry

12. Substract 55-67, using 1's complement method

Ans) Step-1 1's complement of larger number

55 → 110111   67 - 1000011

```
  1 1   1
  0 1 1 0 1 1 1 +
  0 1 1 1 1 0 0
  ─────────────
1 0 1 0 0 1 1   →  - (0 0 0 1 1 0 0)₂
```

2's COMPLEMENT SUBSTRACTION

CASE-1 → Smaller number from larger number.

1) Find 2's complement of smaller number
2) Add this to this larger number
3) Avoid or emit the carry

12 $(13)_{10} - (7)_{10}$ → Substract using 2's complement method

13 → 1101   7 → 111

Ans) 1) 2^{nd} number 2's complement → 0111 → 1000 +

```
1000 +
   1
────
1001
```

```
2 | 13
2 | 6 → 1
2 | 3 → 0
    1 → 1
```

2)

```
 1 1 0 1 +
 1 0 0 1
 ───────
 0 1 1 0  → 6
```

```
13 -
 7
──
 6
```

```
2 | 7
2 | 3   1
    1 → 1
```

3) Discard the carry Ans = $(0110)_2$

CASE-2

If the 1^{st} number is smaller than the second number.

1) Find 2's complement of the larger number ~~add~~

② Add 2's complement to the smaller number

③ Answer is in 2's complement form to get ans in true form, take the 2's complement ~~and~~

? substract 1010 from ~~1001~~ (1000)$_2$

↓10 ↓8

Ans) ① 1010 → 0101 +

1

0110

② 0110 + (Add 2's complement of smaller number)

1000

1110

③ 1110 → 0001 +

1

0010 → (-ve sign → -(0010)

12 Substract 22 from 15 using 1's complement and 2's complement method

Ans) 15 → 1111 22 → 10110

① 22 → 1's complement → 01001

② 01111 +

01001

③ 11000 → 00111 = -(00111)$_2$

15-22 = -7

2|15
2|7 → 1
2|3 → 1
1 → 1

2|22
2|11 -
2|5 -
2|2 -
1

2's complement

① 2's complement 01001 +

1

01010

② 01010 +

01111

11001 → 00110 + = -(00111)$_2$

1

00111

22. Substract 30 − 9 using 1's complement and 2's complement method

Ans) ① CASE-1 → 1's complement

30 − 9 = 21

9 → 1's complement → $(\cancel{1110})_2$ 1001 → 0110

30 → ~~01110~~ → ~~00001~~ (1110)

② ~~11110~~ + ~~00110 +~~
~~01001~~ ~~11110~~
~~110~~ 11110 + 1 ~~00100 +~~
10110 ~~1~~
~~00101~~

③ 10100 +
1

10101

2 | 30
2 | 15 → 0
2 | 7 → 1
2 | 3 − 1
1 → 1

2 | 30
2 | 15 → 0
2 | 7 → 1
2 | 3 → 1
1 → 1

2 | 3
2
1

2's complement.

① 9 → 01001 → 10110 +
1
10111
} 2's complement

② 10111 +
11110

10101 +
~~1~~

~~10110~~

DECIMAL TO OCTAL CONVERSION $(\ )_{10} \to (\ )_8$

1? Convert (25) → (8)

8 | 25
3 → 1

Ans) $(25)_{10} = (31)_8$

2? ~~2~~ $(221)_{10} \to (\ )_8$

$(221)_{10} \to (335)_8$.

8 | 221
8 | 27 → 5
3 → 3

3) $(39.28)_{10} \rightarrow ()_8$.

$8 \,|\, 39$

$4 \rightarrow 7$

$0.28 \times 8 = 2.24$

$0.24 \times 8 = 1.92$

$0.92 \times 8 = 7.36$

$0.36 \times 8 = 2.88$

$(39.28)_{10} \rightarrow (47.2172)_8$

$8 \,|\, 72$

$9 \rightarrow 0$

4) $(72.35)_8 \rightarrow ()_{10}$

Ans) $7\times8^1 + 2\times8^0 + 3\times8^{-1} + 5\times8^{-2}$

$56 + \overset{2}{\cancel{128}} + 0.375 + 0.3125$

$= (58.45)_{10}$

5 $(57.43)_8 \rightarrow ()_2$

Ans) ~~5×8 + 7×8~~ 5 7 4 3 $= (101111.100011)_2$

101 111 100 011

6. $(274.635)_8 - ()_2$

2 7 4 6 3 5 $= (010111100.110011101)_2$

010 011 100 110 011 101

BINARY TO OCTAL

2. $(1011101.1110011)_2 \rightarrow ()_8$

001 011 101 . 111 001 100 $= (135.714)_8$

1 3 5 7 1 4

papergrid

Date: / /

27. $(11011.1101)_2 \rightarrow ()_8 \rightarrow (33.64)_8$

$\underline{011}\ \underline{011} \cdot \underline{110}\ \underline{100}$

3 3 · 6 4

OCTAL TO HEXADECIMAL $()_8 \rightarrow ()_{16}$

12. $(415.373)_8 \rightarrow ()_{16}$

① $()_8 \rightarrow ()_2$ $\underline{4}\ \underline{1}\ \underline{5}\ \underline{3}\ \underline{7}\ \underline{3} \rightarrow (100001101.011111011)_2$ 13 - 10

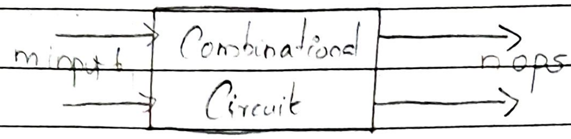
100 001 101 011 111 011

~~0100 00111010 1111 0110~~001|0000|$\overset{8421}{1101}$ · 0111|1101|1000

~~A~~ B ~~A~~ : 1 0 D 7 D 8

MODULE-3

COMBINATIONAL LOGIC CIRCUITS



When logic gates are connected together to produce a specified output or a specified combinations of input variables with no storage involved is called combinational logic circuits.

ADDERS

1) HALF ADDER
2) FULL ADDER

HALF ADDER

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

KMAP

A\B	0	1
0		1 — $A\bar{B}$
1	1	

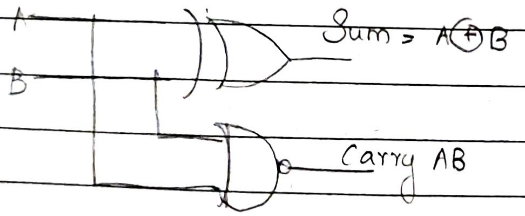
$A \cdot \bar{B}$

Sum $\to A\bar{B} + \bar{A}B$

$= A \oplus B$

A\B	0	1
0		
1		1 — AB

Carry = AB



FULL ADDER

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

KMAP

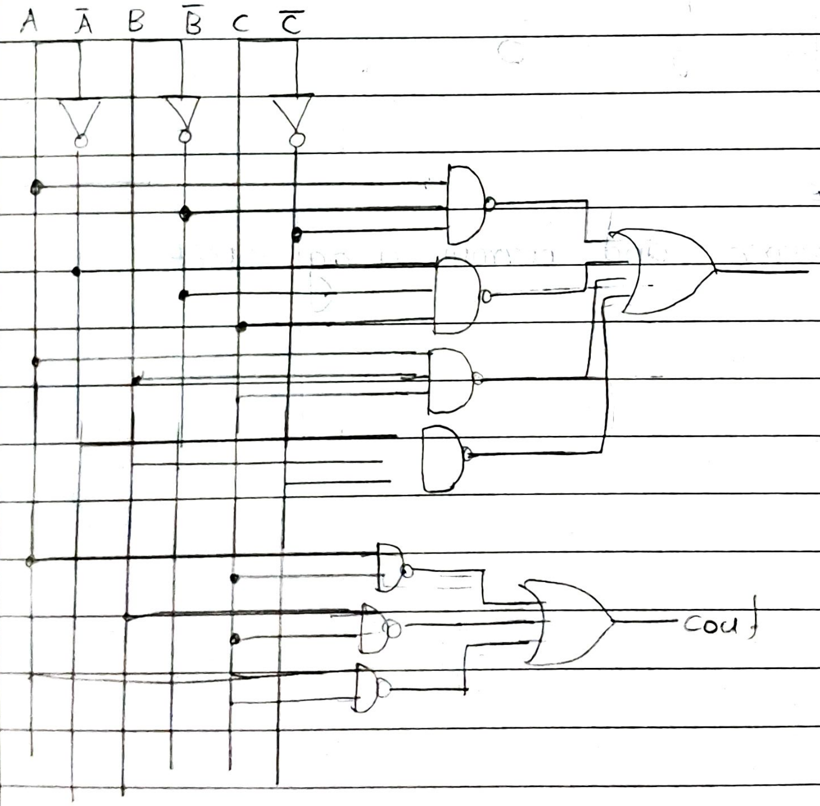
AB\CD	00	01	11	10
0		1	1	1
1	1		1	1

→ AB

Y =√ $A\bar{B}\bar{C}_{in}$ $\bar{A}\bar{B}C$ − ABC_{in}

A\BC	00	01	11	10
0			1	
1		1	1	1

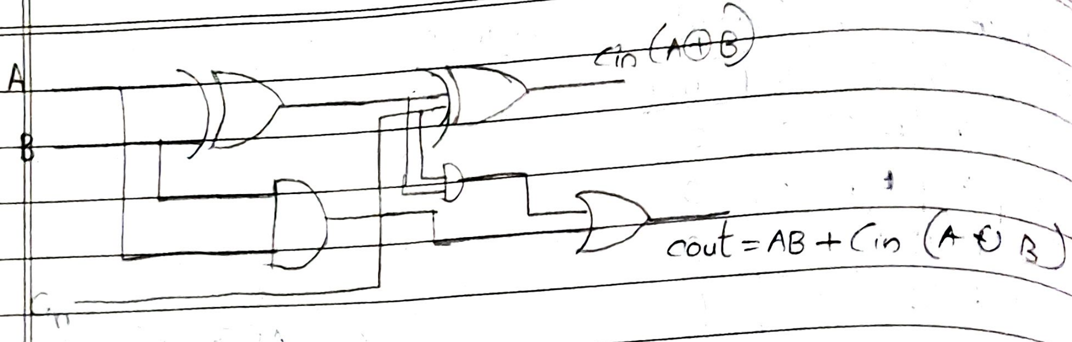
Y = AC_{in} + BC_{in} + AB



Implementation of full adder using 2 half adder

Sum = $C_{in} \oplus (A \oplus B)$

Cout = $AB + C_{in}(A \oplus B)$



Rules for Substraction

Half Substractor

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

Add 9nput and Output = Truth Table

Difference ≠

Find difference and borrow using K-MAP

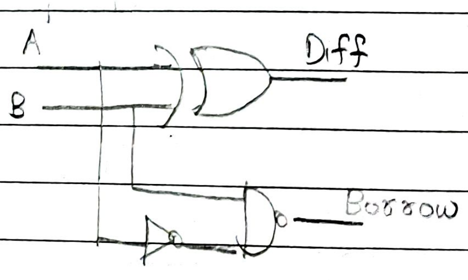
A\B	0	1
0		1 → ĀB
1	1	

VAB̄

= A⊕B

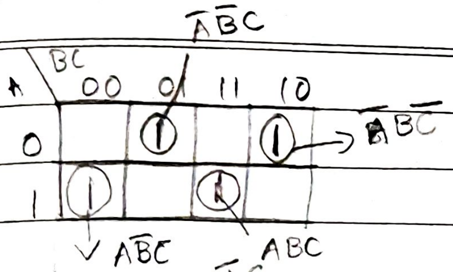
Borrow

A\B	0	1
0		1 → ĀB
1		

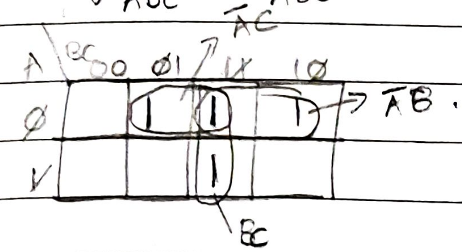


Full Substractor

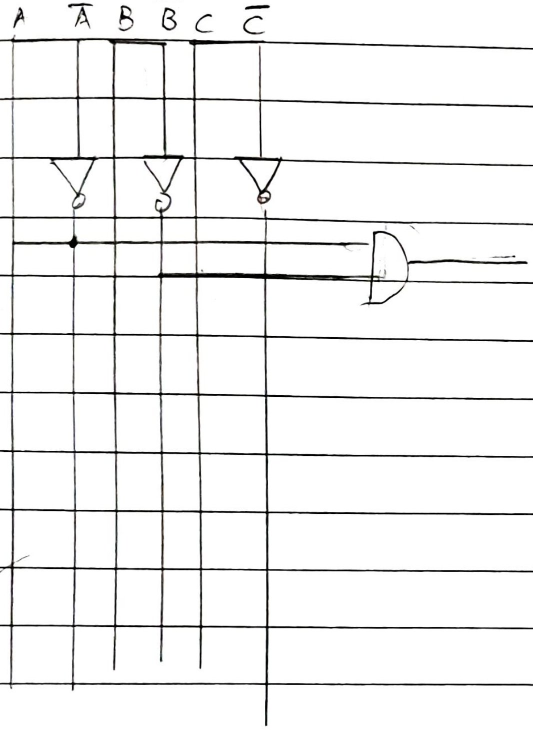
Inputs			Outputs	
A	B	Bin	D	Bout
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1



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



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



Procedure for designing a combinational circuit

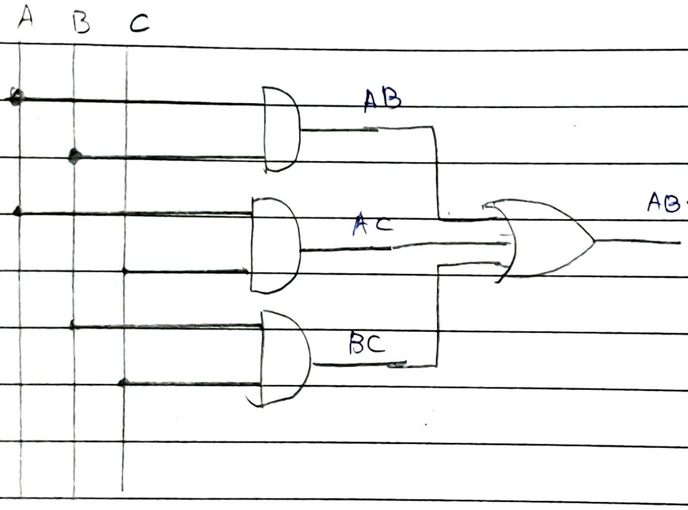
1) Problem definition
2) Determination of input and output variables
3) Assigning letter symbols to input output variable.
4) Derivation of truth table
5) Obtain simplified boolean expression
6) Obtain logic diagram

Design a combination logic circuit with 3 input variables that will produce a logic 1 output more than one input variable are logic 1

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

A \ BC	00	01	11	10
0			1	
1		1	1	1

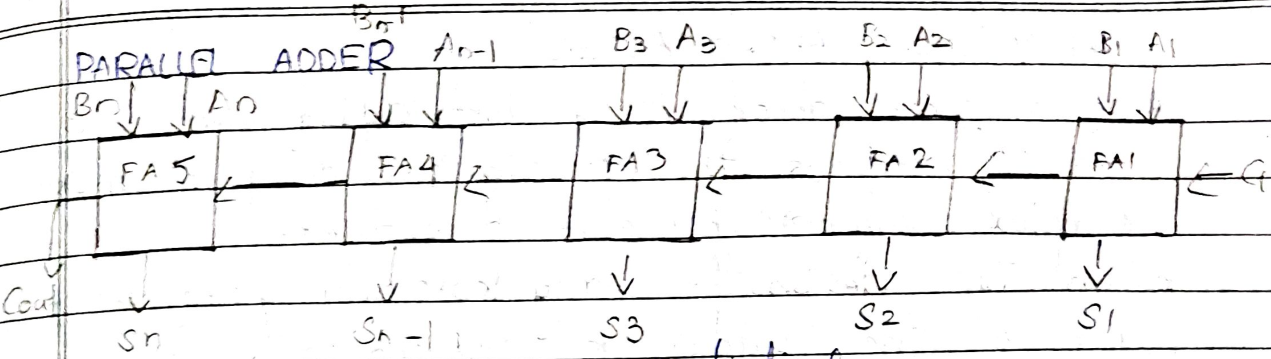
$Y = AB + AC + BC$



ADDER	SUBSTRACTOR	MULTIPLEXER	DEMULTIPLEXER
• Halfadder	• Half Substractor	• 2:1	• 1:2
• Full Adder	• Full Substractor	• 4:1	• 1:4
		• 8:1	• 1:8

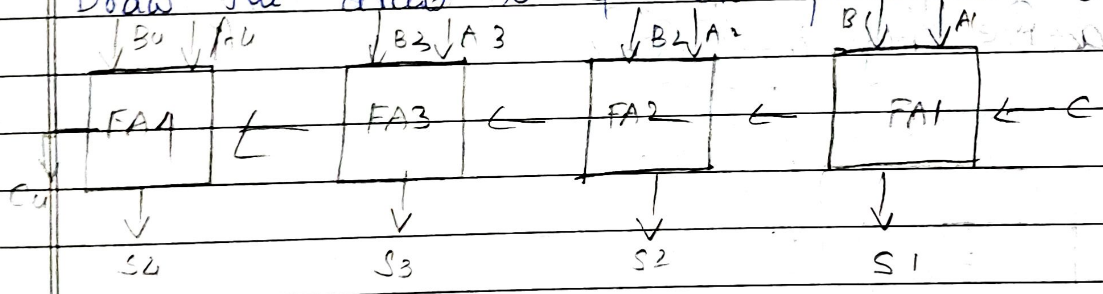
DECODER	ENCODER
$n:2^n$	$2^n:n$
2 to 4	4 to 2
3 to 8	8 to 3
4 to 16	16 to 4

PARALLEL ADDER



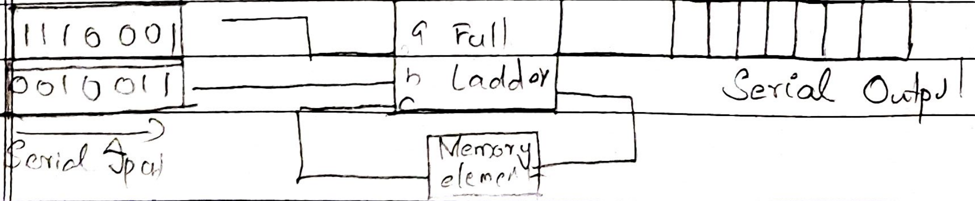
- ~~But a~~ parallel adder is a ~~digital~~ combinational circuit ~~capable~~ used for adding 2 binary numbers greater than 1 bit
- It consist of full adders connected in a chain where the output carry from each full adder is connected to the carry input of the next higher order full adder in the chain
- A n bit parallel adder requires n full adder to perform the operation.

Draw the circuit for 4 bit parallel adder.



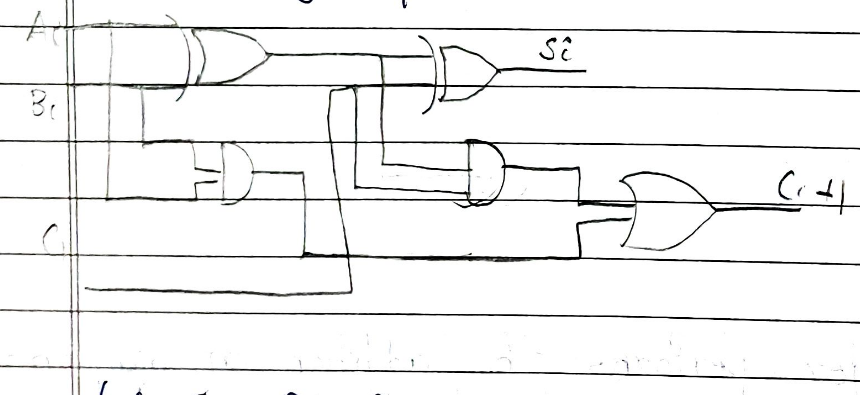
Serial Adder

- Parallel adder performs the addition at high speed but disadvantage is that it requires a large amount of logic circuits
- Serial adder performs bit by bit addition so it require simple circuitry than parallel adder but require low speed of operation.



CARRY LOOK AHEAD ADDER

- A carry-look ahead is a fast added parallel adder as it reduces the propagation delay by more complex hardware, hence it is costlier.
- In this design, the carry logic over fixed group of bits of the adder is reduced to 2 level logic, which is nothing but a transformation of the ripple carry ~~com~~ design
- This method makes use of logic gates so as to look at the lower adder bits of auger and addend to see wheather a higher order carry is to be generated or not
- Carry-lookahead adder ICs A typical carry-lookahead generator IC is 74812 which accept 4 pairs of active low carry propagate (as P_0, P_1, P_2 and P_3) and carry generate (G_0, G_1, G_2 and G_3) signals and an active high input (C_n)



1-1 ⇒ $C_2 = P_1C_1 + G_1$

1-2 → $C_3 = P_2C_2 + G_2$

$P_2(P_1C_1 + G_1) + G_2$

1-3 → $C_4 = P_3C_3 + G_3$

Multiplexer (MUX)

1) Multiplexer is a digital circuit that has multiple inputs and single output

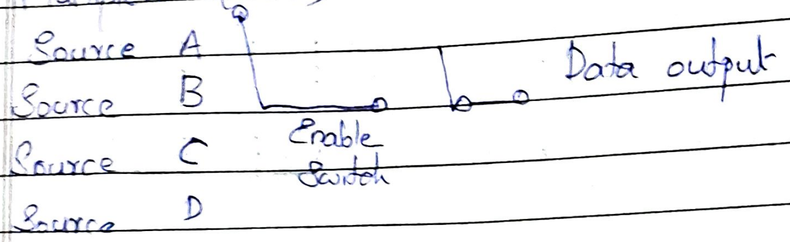
2) It is also knows a data selector

3) The Section of particular input lines is controlled

by set of selection lines.

- Normally there are 2nd Input lines and n Selection lines whose bit combination determines which input is selected
- Multiplexer is (Many one

Multiplexer (MUX)



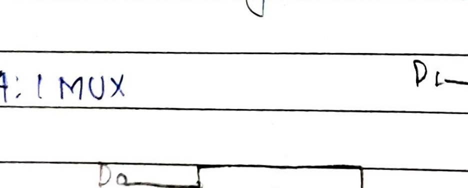
Diff types of MUX

2:1
4:1 } Mux 2^n
8:1

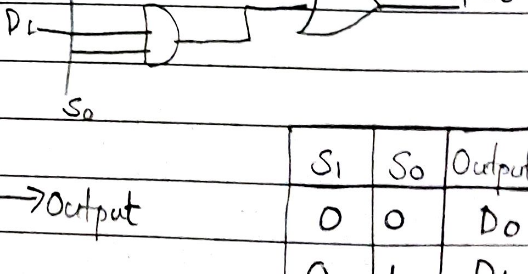
n → Selection Lines
2^n → Input lines
1 → Output lines.

2:1 MUX $Z = D_0\bar{S_0} + D_1 S_0$

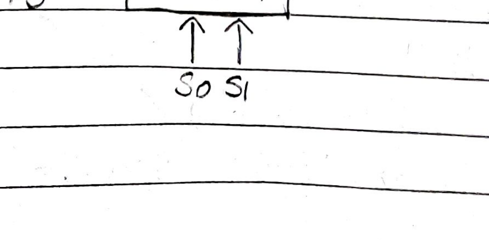
Truth Table



S_0	Output Z
0	D_0
1	D_1

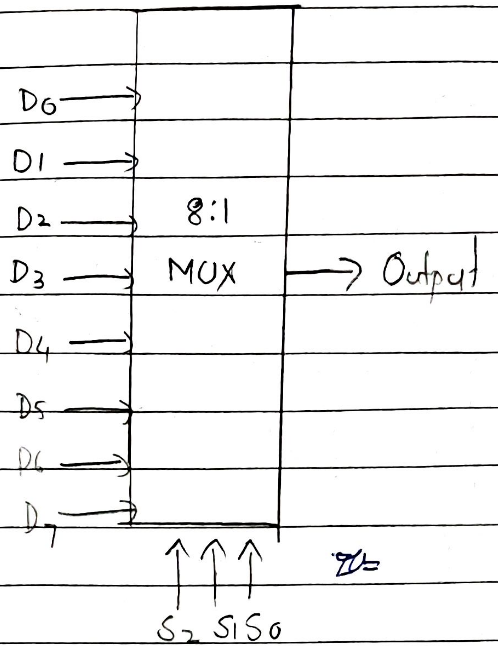


4:1 MUX



S_1	S_0	Output Z
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

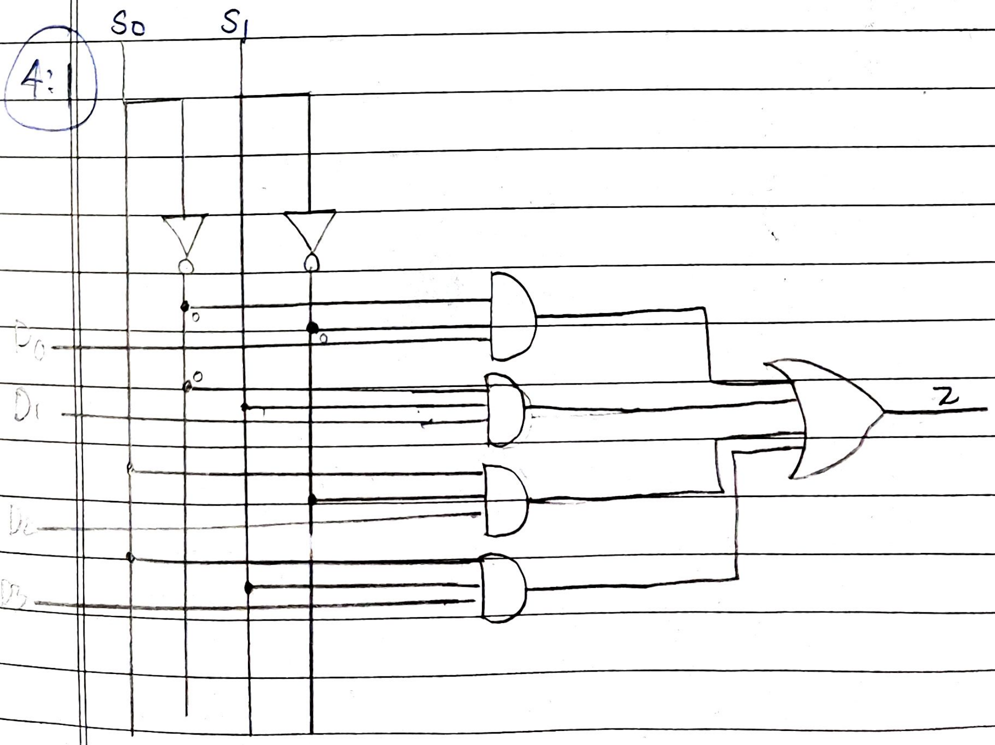
8:1 MUX

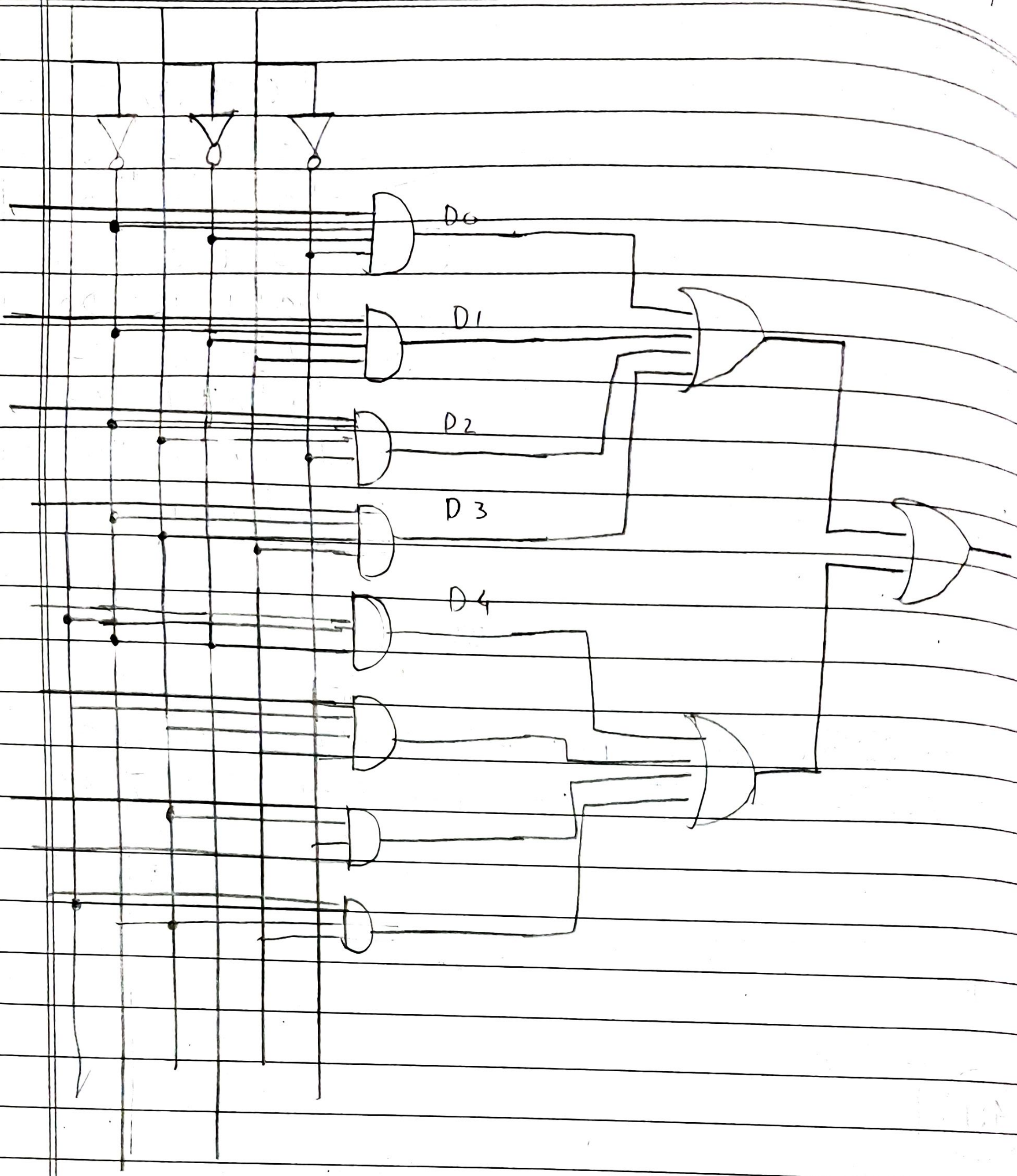


S2	S1	S0	Output Z
0	0	0	D0
0	0	1	D1
0	1	0	D2
0	1	1	D3
1	0	0	D4
1	0	1	D5
1	1	0	D6
1	1	1	D7

~~Z=~~

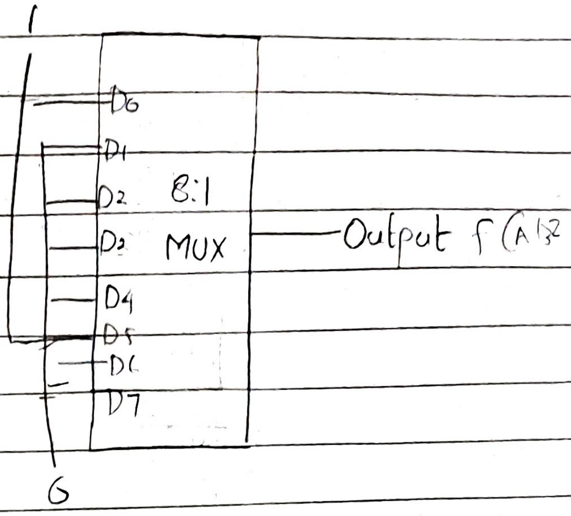
$Z = \overline{S_2}\,\overline{S_0}\,\overline{S_1} D_0 + \overline{S_2}\,\overline{S_1} S_0 D_1 + \overline{S_2} S_1 \overline{S_0} D_2 + S_2 \overline{S_1}\,\overline{S_0} D_3 +$
$S_2 \overline{S_1} S_0 D_4 + S_2 \overline{S_1} S_0 D_5 + S_2 S_1 \overline{S_0} D_6 + S_2 S_1 S_0 D_7$




D0
D1
D2
D 3
D4

1) Implement the following function 8÷1 Mux $f(x,y,z) = \Sigma m(0,2,3,5$

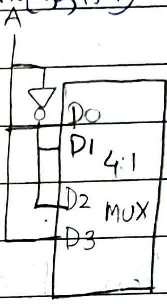
S_2 X	S_1 Y	S_0 Z	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

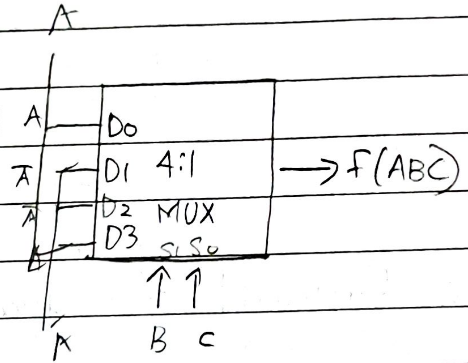


2) Use 4:1 MUX $f(ABC) = \Sigma m(1,2,4,7)$

Implementation table.

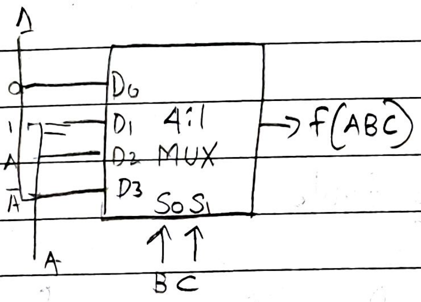
	D0	D1	D2	D3
$\bar{A}$	0	①	②	3
A	④	5	6	⑦
	A	$\bar{A}$	$\bar{A}$	A





3) Implement the expression $f(ABC) = \Sigma m(1,3,5,6)$ using 4:1 multiplexer

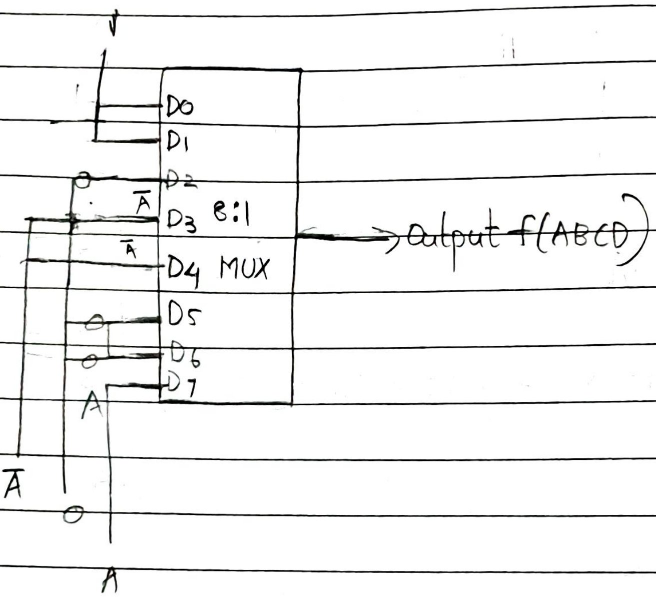
	D0	D1	D2	D3
$\bar{A}$	0	①	2	③
A	4	⑤	⑥	7
	0	1	A	$\bar{A}$



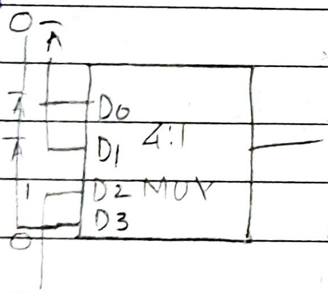
2 Implement boolean expression $f(ABCD) = \Sigma m(0,1,3,4,8,9,15)$

8:1

	D0	D1	D2	D3	D4	D5	D6	D7
$\bar{A}$	0	1	2	3	4	5	6	7
A	8	9	10	11	12	13	14	15
	1	1	0	$\bar{A}$	$\bar{A}$	0	0	A



2 Implement the function $f(ABCD) = \Sigma m(0,1,2,6,9,12,14)$ using 4:1 MUX

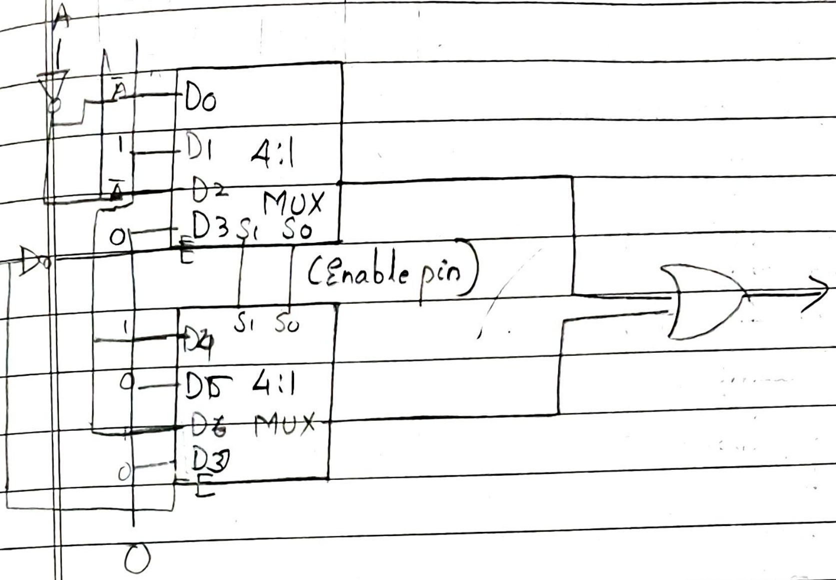


Implementation table

	D0	D1	D2	D3
$\bar{A}$	0	1	2	3
A	4	5	6	7
	$\bar{A}$	$\bar{A}$	1	0

	D0	D1	D2	D3
	0	1	2	3
	4	5	6	7

	D0	D1	D2	D3	D4	D5	D6	D7
Ā	0	1	2	3	4	5	6	7
A	8	9	10	11	12	13	14	15
	Ā	1	Ā	0	1	0	1	0



2. Implement full adder using 4:1 MUX

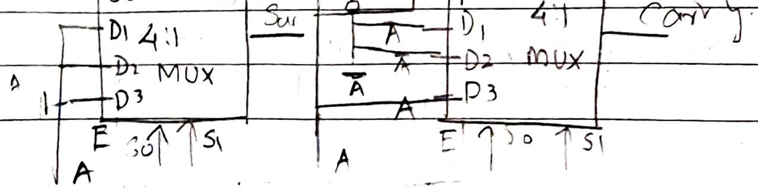
A	B	C	Output Sum	Output C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

	D0	D1	D2	D3
Ā	0	1	2	3
A	4	5	6	7
	A	A	A	1

→ Carry.

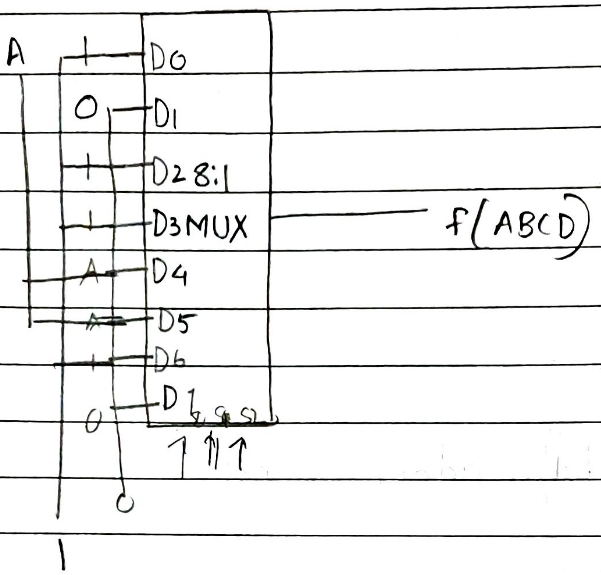
	D0	D1	D2	D3
Ā	0	1	2	3
A	4	5	6	7
	A	Ā	Ā	A

→ Sum



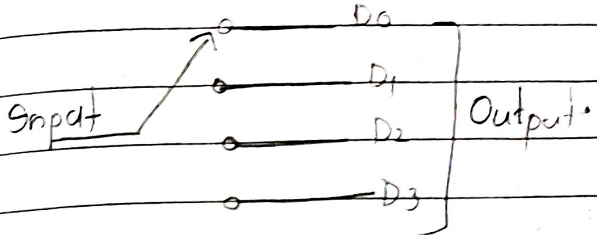
2 Implement the function $f(ABCD) = \Sigma m(0,2,6,19,11,13,12) + \Sigma d(3,8,14)$ use 8:1 multiplexer.

	D0	D1	D2	D3	D4	D5	D6	D7
$\bar{A}$	⓪	1	②	③	4	5	⑥	7
A	⑧	9	⑩	⑪	⑫	⑬	⑭	15
	1	0	1	1	A	A	1	0



DEMULTIPLEXER (DEMUX)

- ✰ Demultiplexer is a digital circuit that has multiple output and a single input
- ✰ It is known as Data distributor
- ✰ The section of particular output line is controlled by set of selection lines.
- ✰ Normally there are 2^n output lines and n selection lines whose bit combination determines which input is selected
- ✰ Therefore demultiplexer is 'One to Many'



TYPES OF MULTIPLEXER

1:2

1:4

1:8

1:16

Multiplexer IC →
↓ 74150 (16:1)

74151 (8:1) → Multiplexer

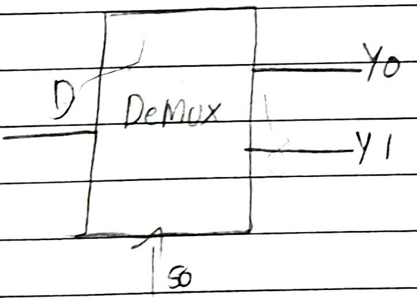
n → Selection Lines
2^n → Output Lines
1 → Snput lines

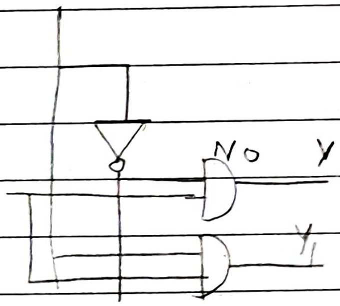
1X2 DEMUX

S0	Y0	Y1
0	F	0
1	0	F

$Y_0 = \overline{S_0}F + S_0 0 = \overline{S_0} E$

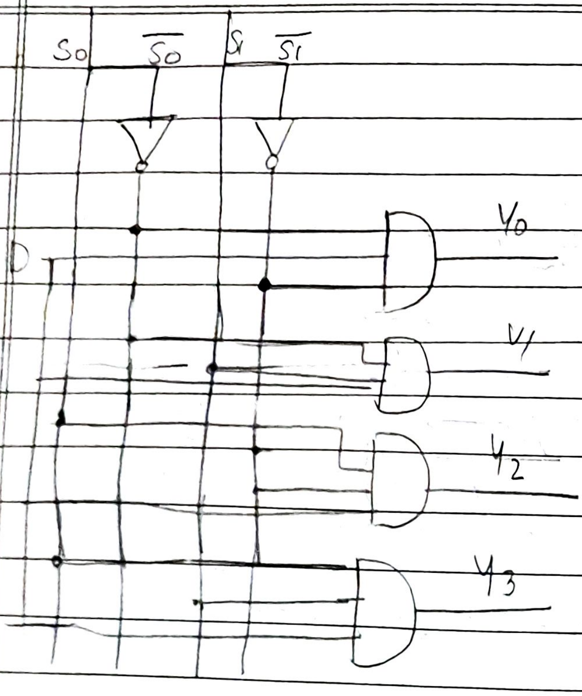
$Y_1 = \overline{S_0} 0 + S_0 F = S_0 F$ DN=0





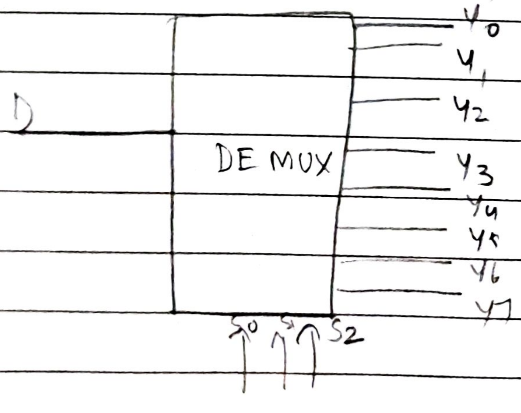
1:4 Demultiplexer

S0	S1	Y0	Y1	Y2	Y2
0	0	D	0	0	0
0	1	0	D	0	0
1	0	0	0	D	0
1	1	0	0	0	D

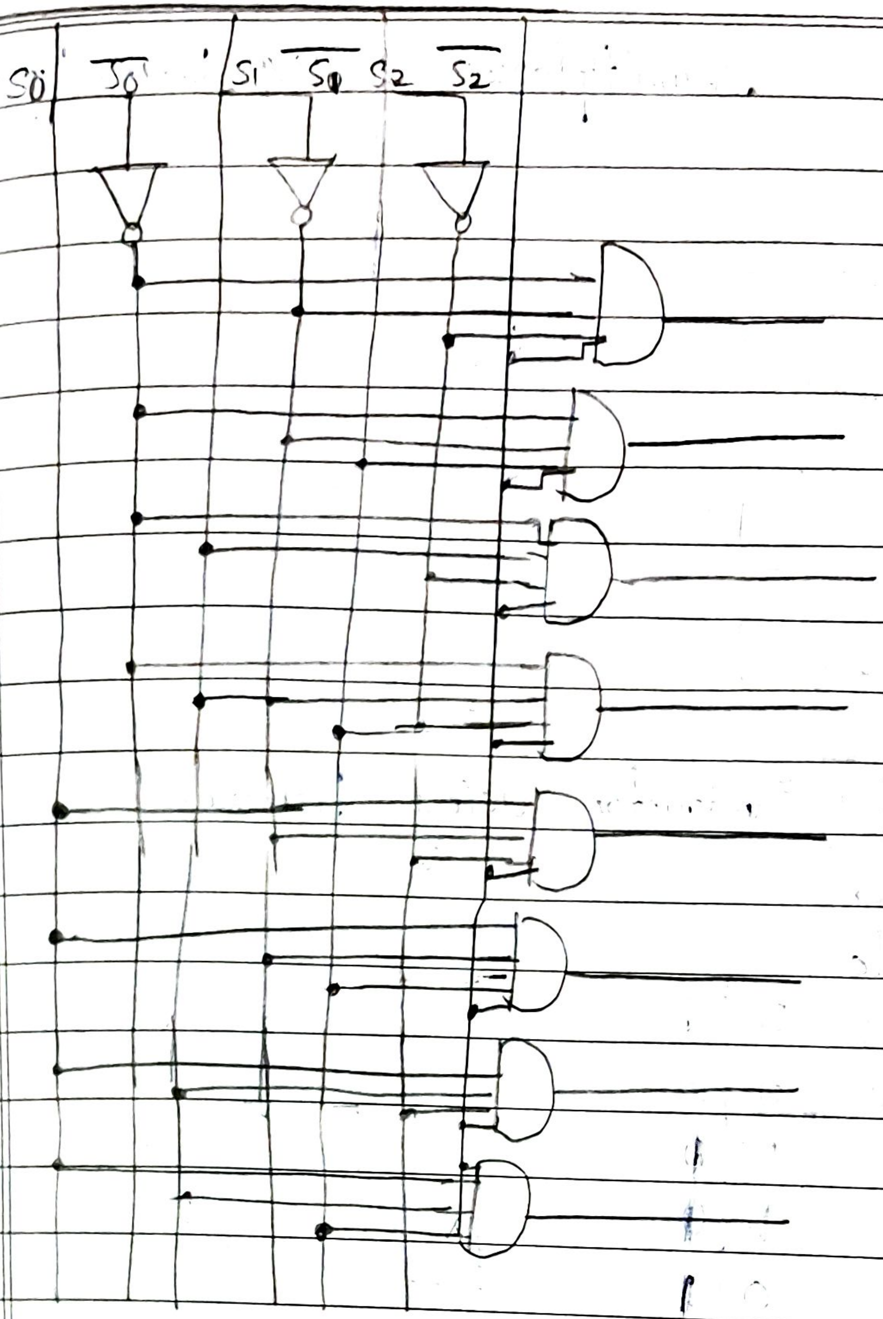


1:8 Demultiplexer

S0	S1	S2	D0	D1	D2	D3	D4	D5	D6	D7
0	0	0	D	0	0	0	0	0	0	0
0	0	1	0	D	0	0	0	0	0	0
0	1	0	0	0	D	0	0	0	0	0
0	1	1	0	0	0	D	0	0	0	0
1	0	0	0	0	0	0	D	0	0	0
1	0	1	0	0	0	0	0	D	0	0
1	1	0	0	0	0	0	0	0	D	0
1	1	1	0	0	0	0	0	0	0	D

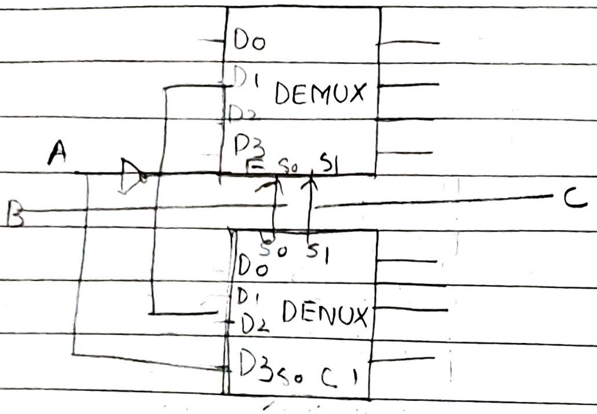


D)



Design a 1:8 Demultiplexer 2 1:4 Demultiplexer

Ans)

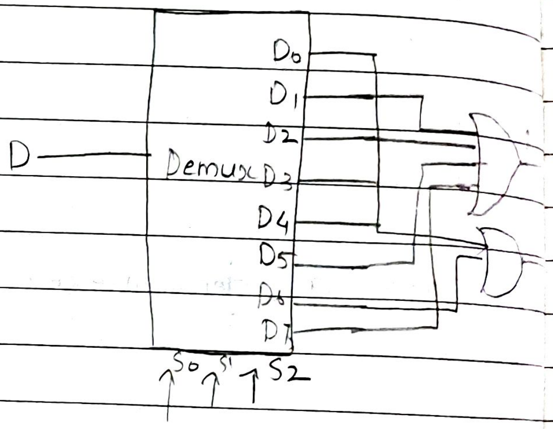


2 Implement full Substractor using Demux.

Full Substractor

Truth Table

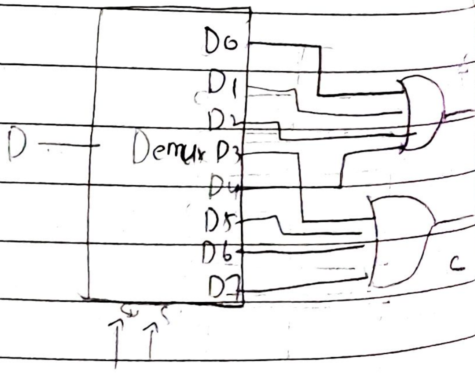
A	B	C	D	B
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	~~0~~ D
1	1	1	1	1



? full adder using demultiplexer

Truth Table

A	B	C	S ~~D~~	C ~~B~~
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



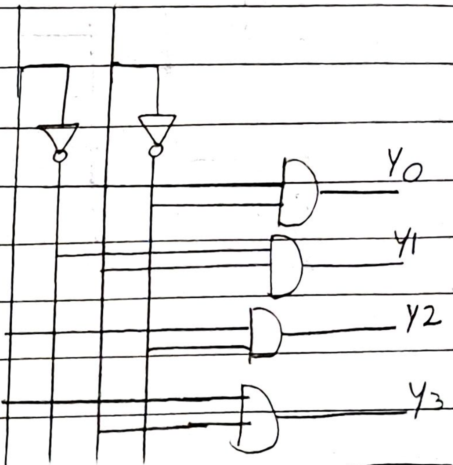
Decoder

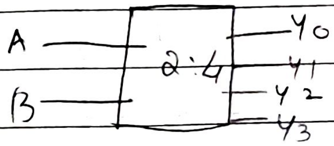
It is a combinational circuits that converts ~~from~~ information from n input lines to a maximum of 2^n output lines,

Eg: 2:4, 3:8, 4:16

2:4 Decoder

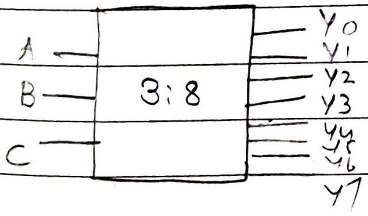
Input		Output			
A	B	Y0	Y1	Y2	Y3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

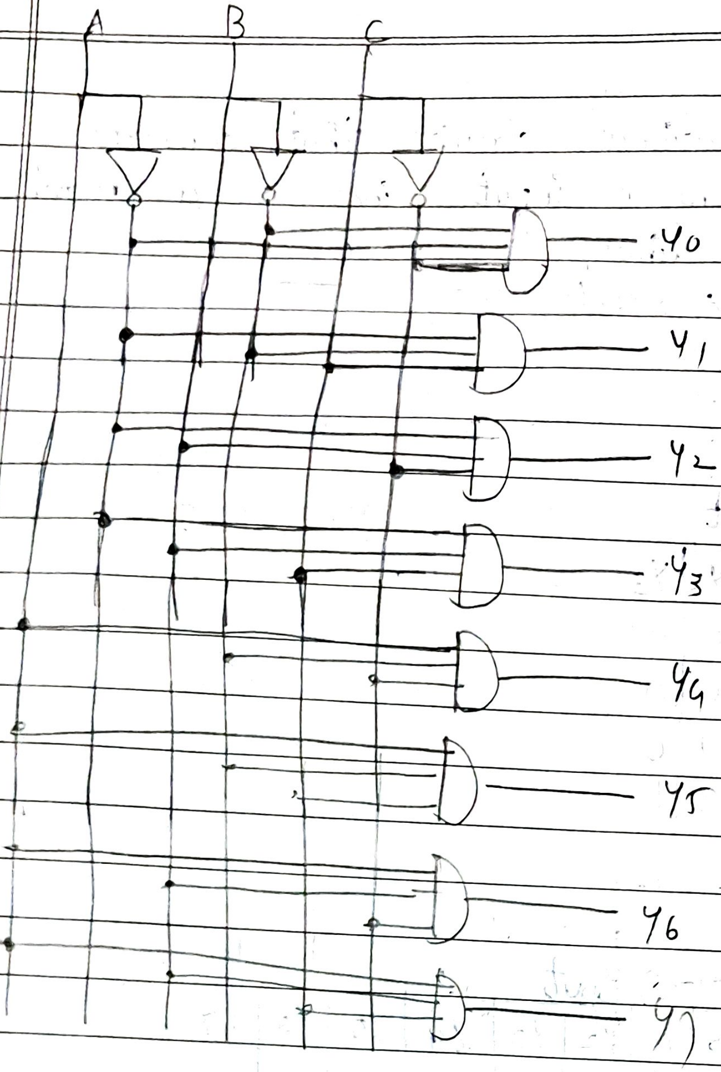




3:8 Decoder

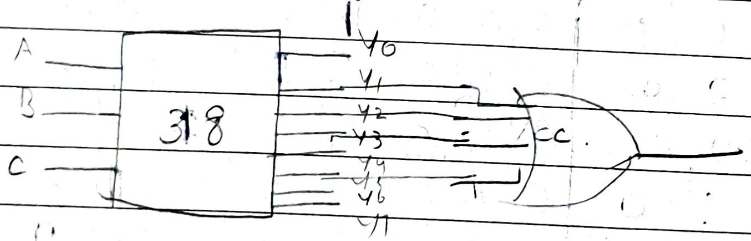
Input			Output							
A	B	C	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1





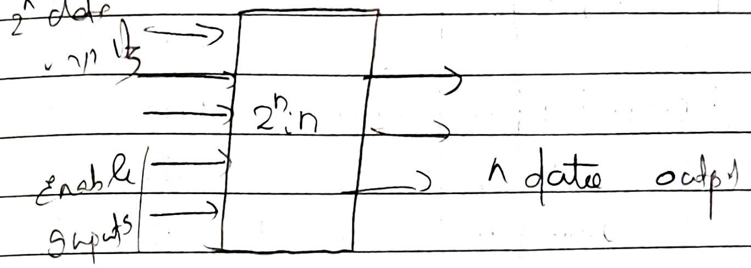
Implement boolean function using decoder

$F_1(ABC) = \Sigma m(1,2,5,7)$



ENCODER

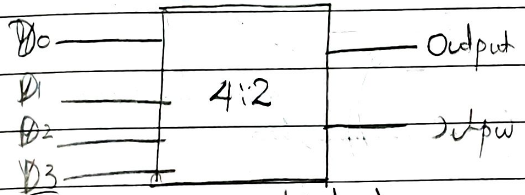
- Encoder is the inverse operation of decoder.
- Multiple input - Multiple output logic circuits which converts coded inputs to coded outputs where the inputs and outputs are different

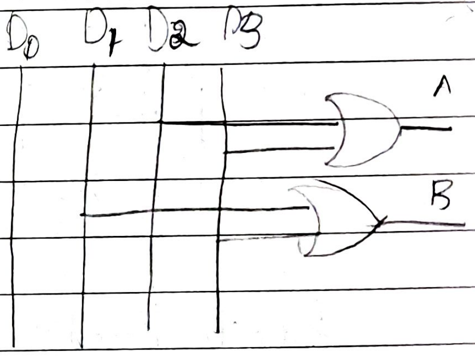


- Decoder consist of n outputs and 2^n maximum possible inputs.

eg: 4 to 2 8 to 3 16 to 4

Logic Circuit

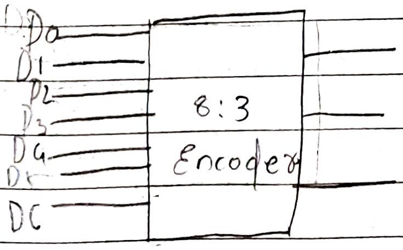




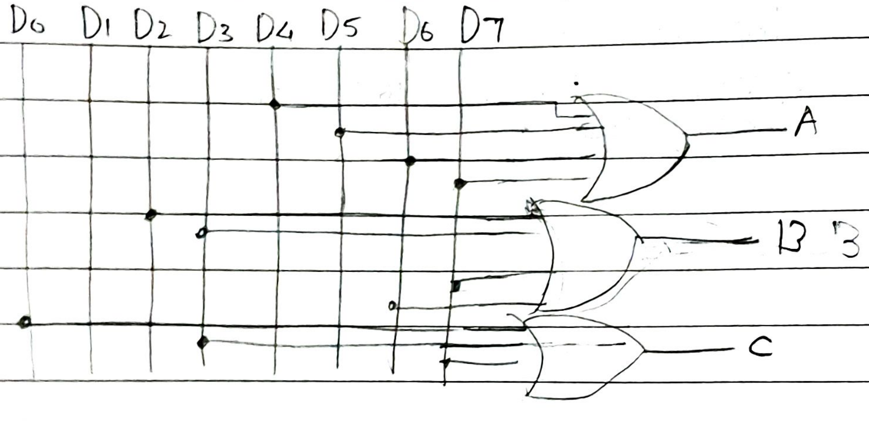
Input				Output	
D0	D_1	D_2	D_3	A	B
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

Truth Table

8:3



Inputs								Output		
D0	D1	D2	D3	D4	D5	D6	D7	A	B	C
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



Priority Encoder

- ★ A priority encoder is one of the types of encoders in which an order imposed to the inputs that means compared with the standard encoder. It includes the priority function

- ★ It is based on relative magnitude of [illegible]

Priority Encoder

Input				Output		
D_0	D_1	D_2	D_3	Y_1	Y_0	V
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

$$Y_1 = D3 + \overline{D3} D2 = D3 - D3$$
$$Y_0 = D_3 + \overline{D3}\ \overline{D2}\ D1 = D2 - \overline{D2} D1$$
$$V = D3 + D2 - D_1 + D_0$$

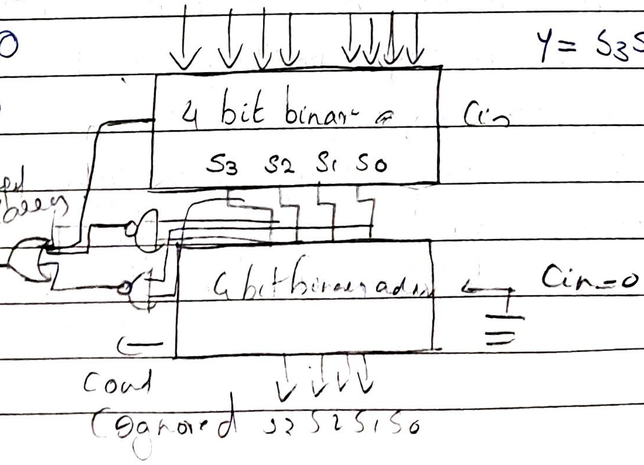
BCD ADDER

Input				Output
S_3	S_2	S_1	S_0	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
0	0	0	0	0
1	0	0	1	0
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

S_3S_2 \ S_1S_0	00	01	11	10
00	0	1	3	2
01	4	5	7	6
10	1 12	1 13	1 15	1 14
16	8	9	1 11	1 10

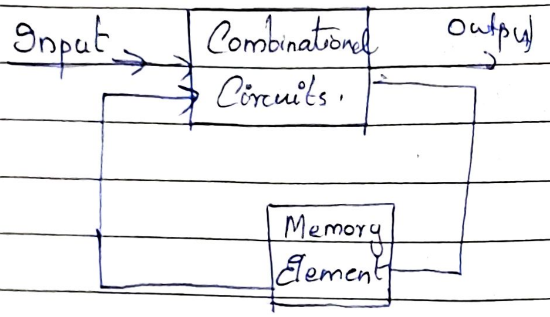
→ S_3S_2
→ S_1S_3

$$Y = S_3S_2 + S_1S_3$$



MODULE- 4

SEQUENTIAL CIRCUITS



As shown in the figure is a sequential circuits in which memory elements are connected to the combinational circuit as a feedback path. Information stored in the memory elements at any given time defines the presence state of sequential circuit. The presence state and external inputs determine the outputs and next stage of the sequential circuits
∴ We can specify the sequential circuit by a line sequence of external inputs. Internal states, (present State and next state) and outputs.

~~LATTICE~~ LATCHES AND FLIP FLOPS

These are bistable elements which are the basic building blocks of most sequential circuits. The main difference between lattice and flipflops is in the method used for changing their state. We use the name flipflop for a sequential device that normally samples its inputs and changes its outputs only at times determined by the clocking signal.

On the other hand the name latch for a sequential device can be used when it chicks all of its inputs countinuosly and changes it's output accordingly at any time independent of clocking signal.

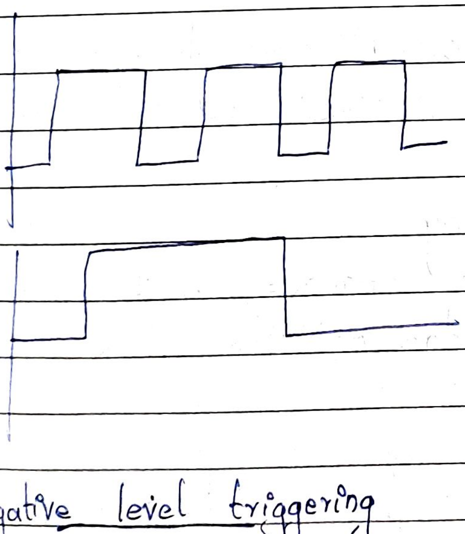
LEVEL TRIGGERING.

- In the level triggering, the output state is allowed to changed according to input when active level is maintained at the enable input.

- There are 2 types of level triggered latches
  1) Positive level triggerng
  2) Negative level triggering

Positive level trigerring

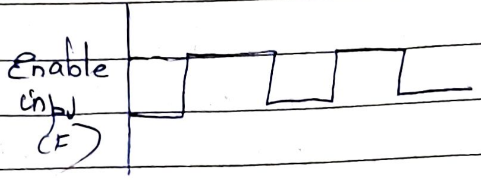
- The output of flip flop responds to input change only when enable input is high



(Edge is not consid...)

Negative level triggering

The output of flip-flop responds to the input changes only when is enable input is 0 (low



Flip flop is enabled only when le
of E input is low.

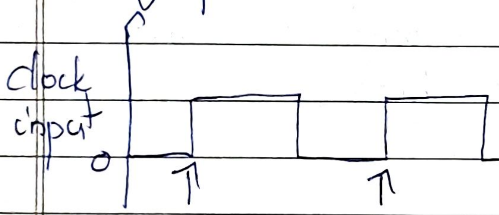
Edge Triggering

In the edge triggering the output responds to the changes in the input only at the +ve or -ve edge of the clock pulse at clock input

- Two types of edge triggering
  1) Positive edge triggering
  2) Negative edge triggering.

Positive edge Triggering

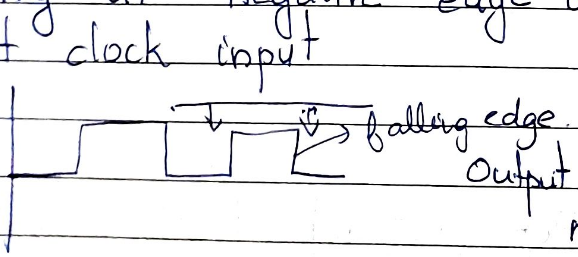
Here the output responds to the changes in the input only at the positive edge of the clock pulse at clock input



output responds only at the possible edges of [illegible]

Negative edge triggering

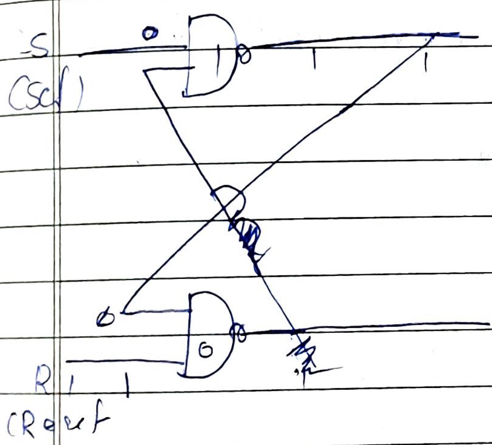
Output responds to the changes in the input only at negative edge of the clock pulse at clock input



Output responds only at the negative edges of pulse

Latch	Flipflop
Building block of sequential circuit, built using logic gates	Building block of sequential circuit build using latches
Check input countinuosly a changes output correspondingly	Check the input continuosly and check the output in a continuous manner only in clock signal
Work with only binary inputs	Work with binary input and clock signal can be used as register
Cannot used as register	Flipflops are edge triggered.
Latches are level triggered	

SR Latch Nand

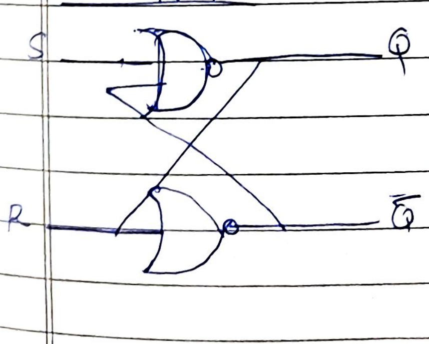


A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

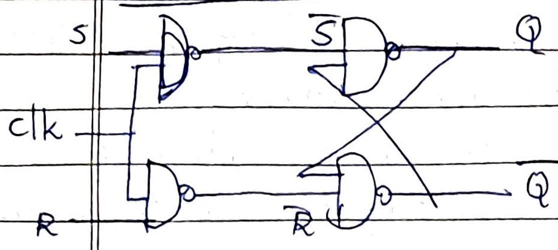
Truth Table (Nand)

S	R	Q	$\bar{Q}$	State
0	0	X	X	invalid
0	1	1	0	set
1	0	0	1	Reset
1	1	NC	NC	Memory

SR Nor



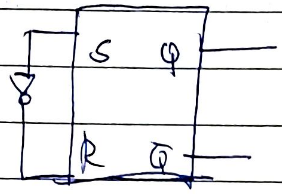
SR FLIPFLOP



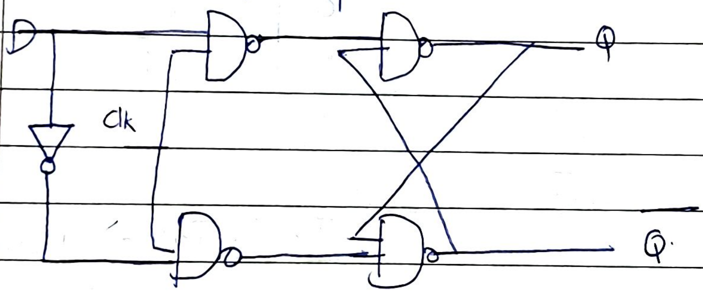
S	R	Q_{n+1}	State
0	0	Q_n	Memory
0	1	0	Reset
1	0	1	Set
1	1	X	Invalid.

D FLIPFLOP (Delay flipflop)

Logic Diagram

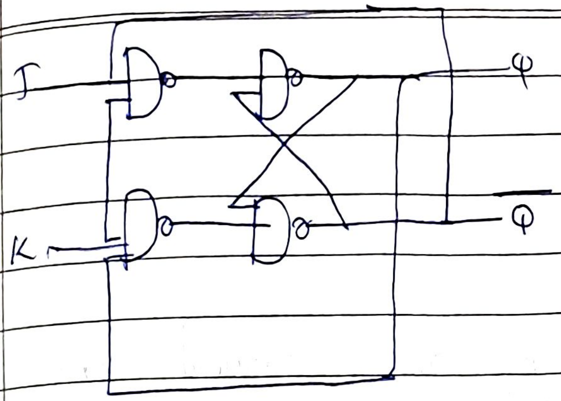


The S input is made high to store 1 in the flipflop and R input is made high to scac 0 in flip-flop



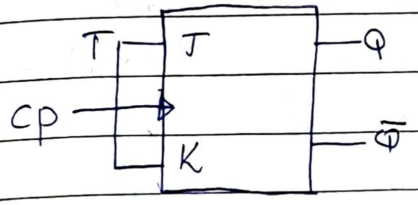
D	Q_{n+1}	STATE
0	0	Reset
1	1	Set

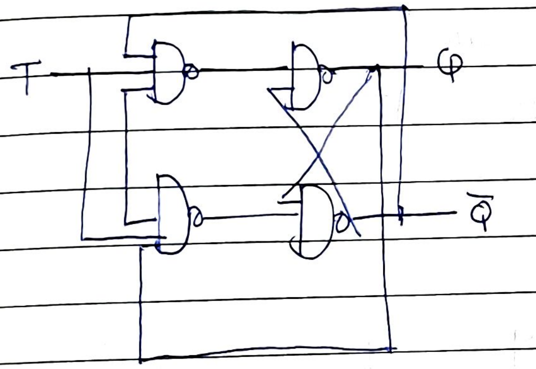
JK flipflop (Jack Kilby)



J	K	Qn+1	State
0	0	Qn	Memory
0	1	0	Reset
1	0	1	Set
1	1	Qn'	~~Togle~~ Toogle

T-FLIPFLOP



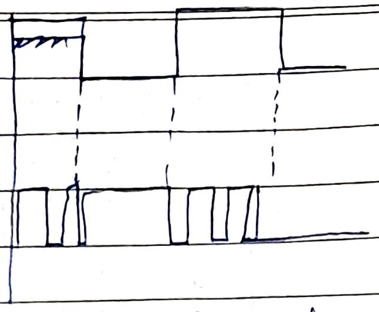


T	Qn+1	State
0	Qn	Memory
1	$Q\bar{n}$	Toggle

Race around Condition

- In $\overset{SR}{\text{JK}}$ flipflop when $\overset{S}{J}=\overset{R}{K}=1$, the output toggles output changes either from 0 to 1 or 1 to 0

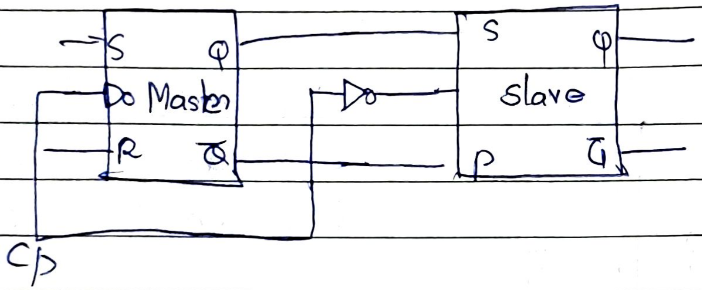
- For JR flipflop when ~~J=K=1~~ J and K = 1 clock is too long then, this state of FF keeps on toggle which leads to uncertainity in determining in ~~de~~ output s/a state of flipflop
This problem is called Race around condition



Methods for eliminating raise around condition

Master slave JKSRflipflop.

A master slave ylipflop is countinued from 2 flipflops: One circuit serves as a master is the other needs a slave and the overall circuit is refered as master slave flipflop



Master slave JK SR flipflop

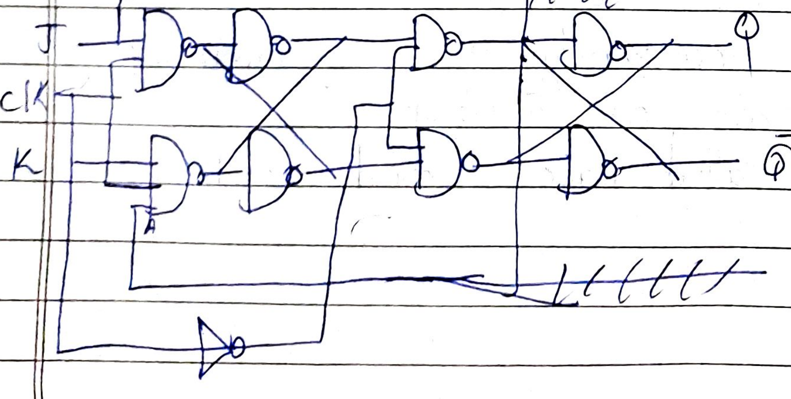
★ when clk=1, 1st FF is enabled is inhibited

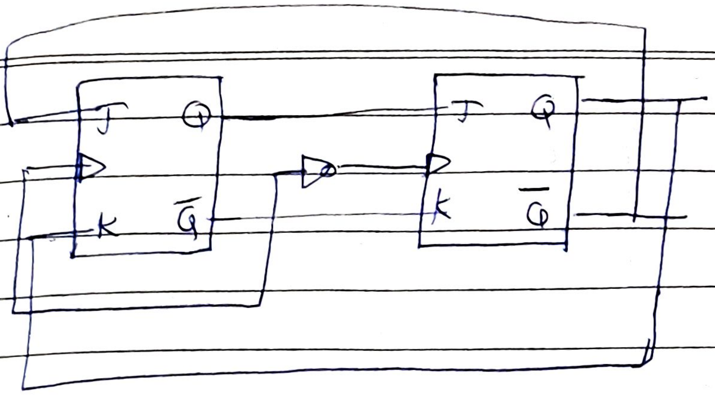
★ when clk=0 1st is inhibited, 2nd is enabled

1st → Master

2nd - Slave

Master slave JK





FLIPFLOP CONVERSION

1) SR to D flipflop.

Excitation table

	Excitation			
D	Qn	Qn+1	S	R
0	0	0	0	X
0	0	1	1	0
1	1	0	0	1
1	1	1	X	0

Characteristic Table

Qn	S	Q	Qn+1
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	Y
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	X

Excitation of SR

Input	PS	NS	FF
D	Qn	Qn+1	S R
0	0	0	0 X
0	1	0	0 ϕ
1	0	1	ϕ 0
1	1	1	X 0

S

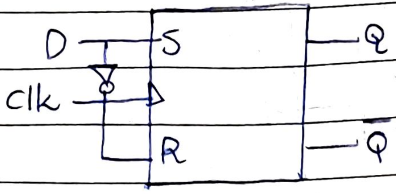
D \ Qn	0	1
0		
1	1	X

S = D

R

D \ Qn	0	1
0	X	1
1		

$R = \overline{D}$



SRFF TO JKFF

Input		PS	NS	FF inputs	
J	K	Qn	Qn+1	S	R
0	0	0	0	0	X
0	0	1	1	X	0
0	1	0	0	0	X
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	1	X	0
1	1	0	1	1	0
1	1	1	0	0	1

For writing Qn and Qn+1 refer truth table
FF Inputs " refer excitation ta

S

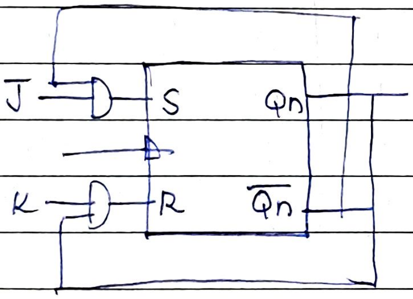
J \ KQn	00	01	11	10
0		X		
1	1	X		1

$s = J\cancel{K}\overline{Qn}$

R

J \ KQn	00	01	11	10
0	X		1	X
1			1	

R = KQn



SRFF TO T

Input	PS	NS	FF inputs	
T	Qn	Qn+1	S	R
0	0	0	0	X
0	1	1	X	0
1	0	1	1	0
1	1	0	0	1

S

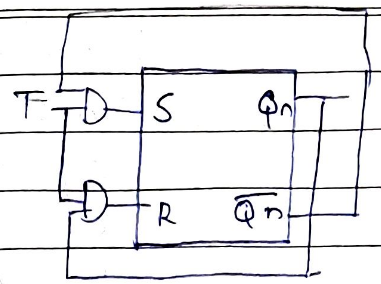
T \ Qn	0	1
0		X
1	1	

$S = T\overline{Qn}$

R

T \ Qn	0	1
0	X	
1		1

R = TQn

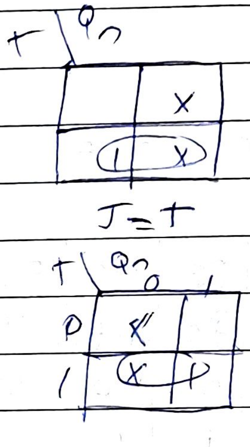


EXCITATCON OF JK FLIPFLOP

Qn	Qn+1	J K
0	0	0 X
0	1	1 X
1	0	X 1
1	1	X 0

JKFF TO TFF

Input	PS	NS	FF inps	
T	Qn	Qn+1	J	K
0	0	0	0	X
0	1	1	X	0
1	0	1	1	X
1	1	0	X	1

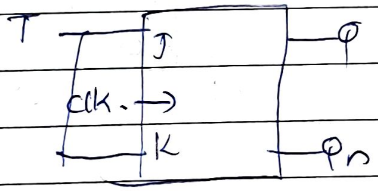


J>IS

J=T

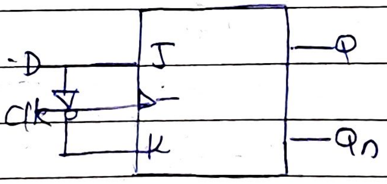
~~K=TQn~~

K=T



JR to DFF

Input	PS	NS	FF inps	
D	Qn	Qn+1	J	K
0	0	0	0	X
0	1	0	X	1
1	0	1	1	X
1	1	1	X	0



DFF TO T FF

Inputs	PS	NS	FF inps
T	Q_n	Q_{n+1}	D
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

Excitation table of D

Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

D

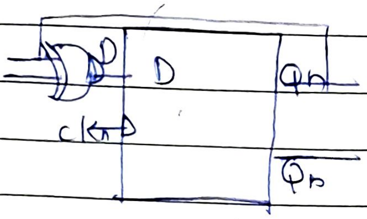
T \ Q_n	0	1
0		1
1	1	

→ $\bar{T}Q_n$

$T\bar{Q}_n$

$D = T\bar{Q}_n + \bar{T}Q_n$

$= T \oplus Q_n$



TFF TO DFF

Inputs	PS	NS	FF inps
D	Q_n	Q_{n+1}	T
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Excitation Table of T

Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

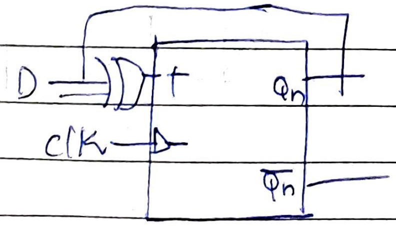
D \ Q_n	0	1
0		1
1	1	

→ $\bar{D}Q_n$

$D\bar{Q}_n$

$T = D\bar{Q}_n + \bar{D}Q_n$

$= D \oplus Q_n$

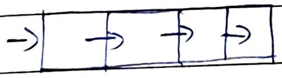


REGISTERS

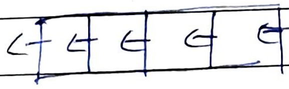
Register is a group of flipflops. A flipflop can store 1 bit of information. So a n bit register as a group of n flipflops and is capable of storing any binary information Containing n bits.

Shift registers

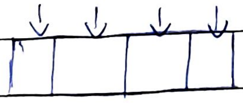
The binary information or data in a register can be moved from one stage to another within the register or into or out of the register, with the application of clock pulse. This type of movement of shifting is used in arithmetic or logical operations. The registers is used for this purpose is called shift registers



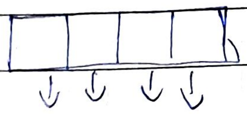
→ Serial right shift



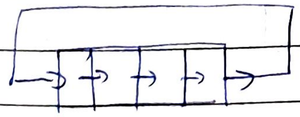
→ Serial left shift



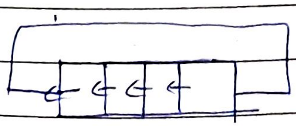
→ Parallel shift in



→ Parallel shift out



Rotate right

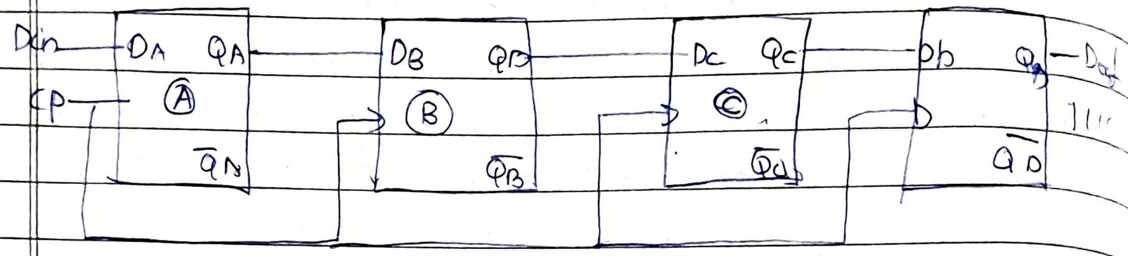


→ Rotate left

TYPES OF SHIFT REGISTER

1) SISO → Serial in serial out
2) SIPO → Serial in parallel out
3) PISO → Parallel In Serial out
4) PIPO Parallel in Parallel out

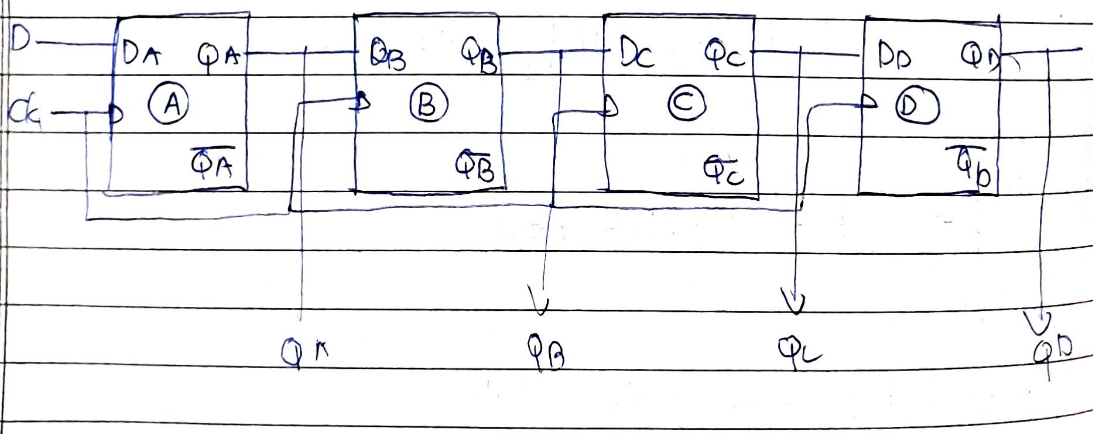
SISO → Serial un Serial ~~out~~ out



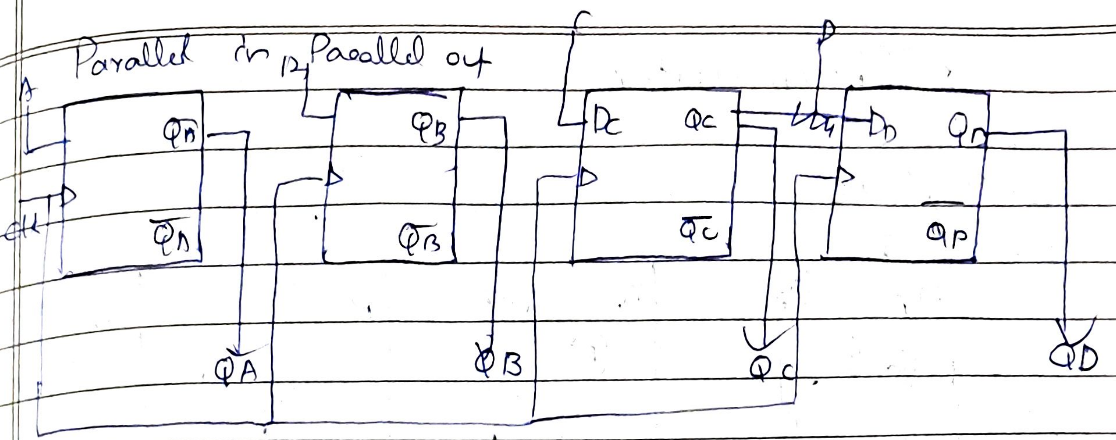
Eg:

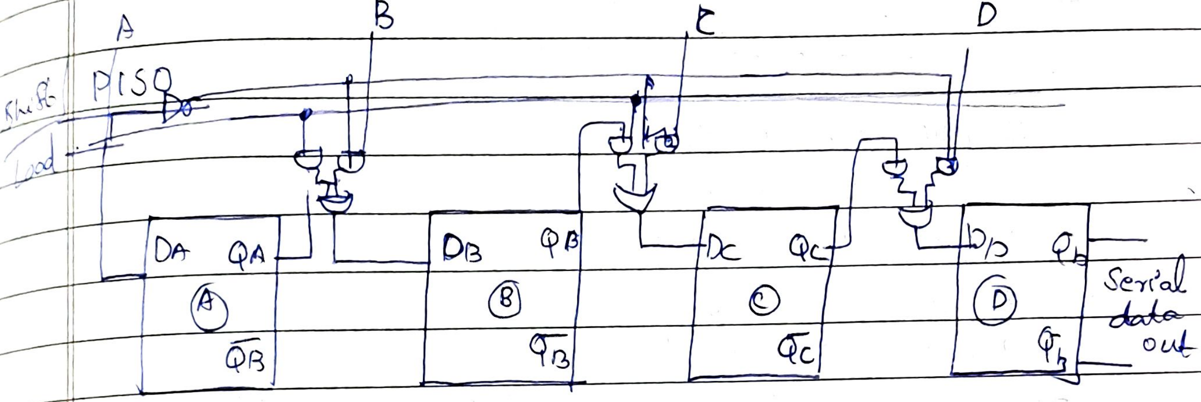
QA	QB	QC	QD	
0	0	0	0	→ ①
1	0	0	0	→ A
1	1	0	0	→ B
1	1	1	0	→ C
1	1	1	1	→ D

SIPO Serial in parallel out



Parallel in 12 Parallel out



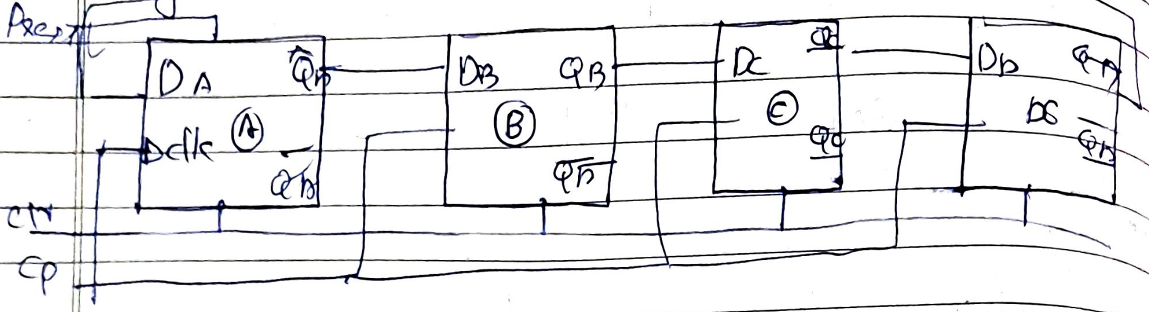


The PISO of shift register it consist of a control signal shift/$\overline{LOAD}$ path.
When shift/Load path is low, G_1, G_2, G_3 will be enabled and G_4, G_5, G_6 will be disabled then the data will be loaded to all trespective flipflops.
When the shift or load path signal is high, G_4, G_5, G_6 will be enabled and G_1, G_2, G_3 will be disabled. After that the data will be shifted in serial manner.

Application of shift register.
1) Delay line
2) Serial to parallel convertr
3) Parallel to Serial converter
4) Shift register counters
 1) Ring counter

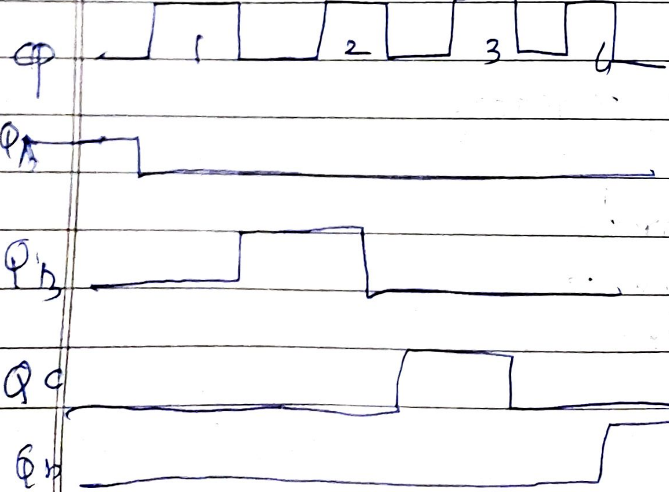
2) Johnson counter

Ring Counter

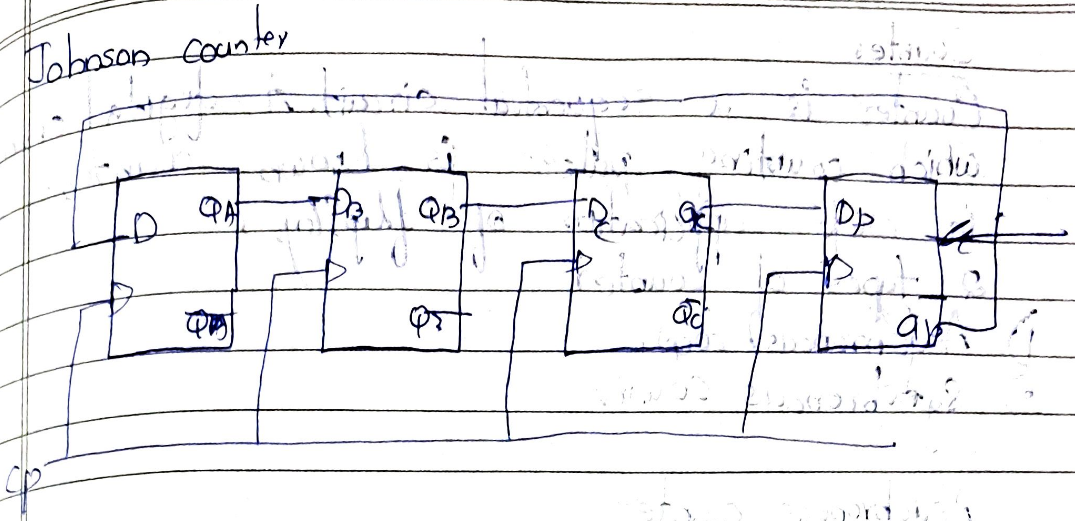


Cp	QA	QB	QC	QD
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1

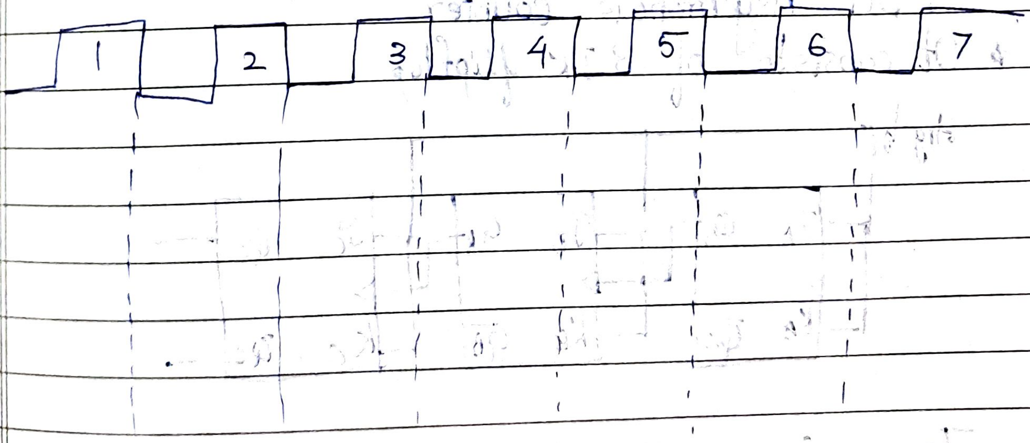
The figure shows the 4 bit ring counter using D ff. It consist 2 additional inputs preset and clear. To the first ff to make the state of the first flipflop as logic 1 and the remaining flipflop are cleared with the help of CLR input. By the application of clock pulse the output of first ff is shifted to next flipflop and so on That means the o/p shifts in a serial manner from 1st ff to last ff in the form of ring.



Johnson counter



Cp	QA	QB	QC	QD
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1



Counter

Counter is a sequential circuit. A digital circuits which counting pulses is known counter. Count is wide application of flipflop.

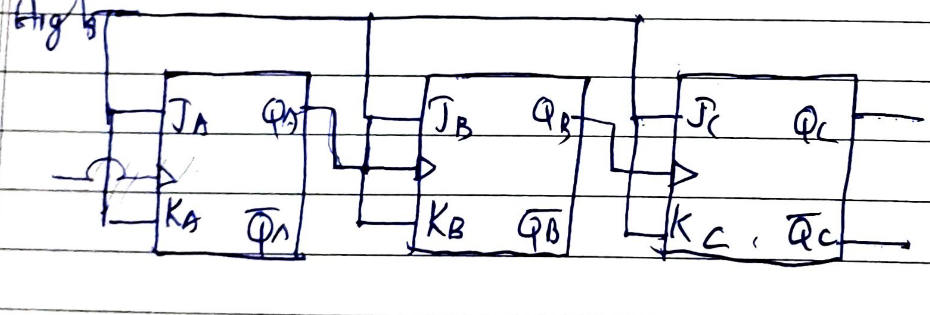
2 types of counter
1) Asychronous count.
2) Synchronous countr

Asychronous counter

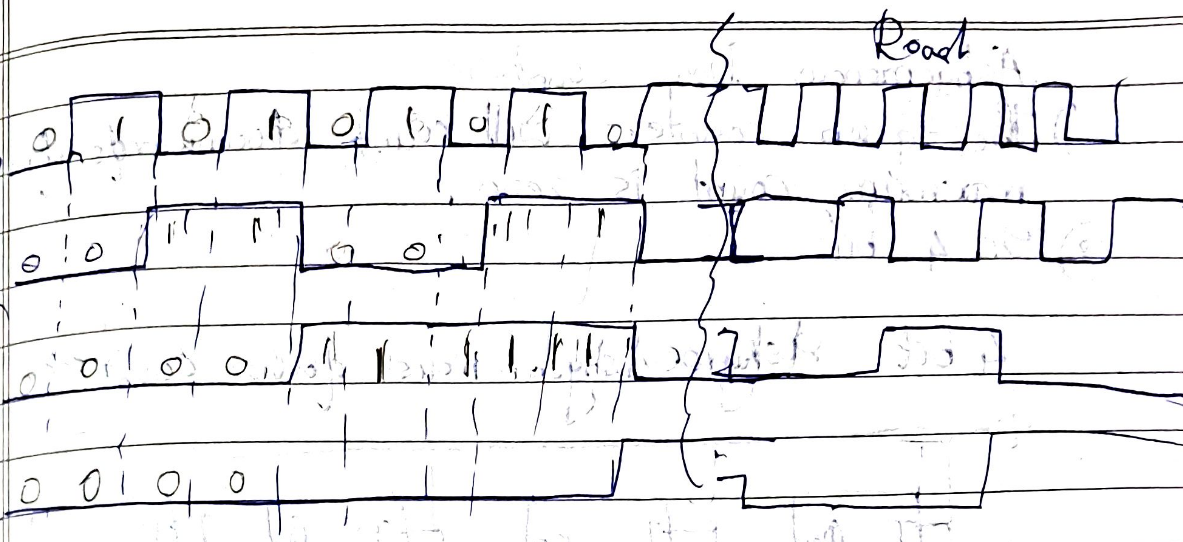
If the flipflop do not recieve the same clock signal called as asychronous counter the output of system clock signal only to first flip flop. The remaining flip-flops flipflops recieve the clock signal from output of its previous stage flip flop. Hence the outputs of all flipflops do not change.

3 BIT Asychronous counter

★ It consists of 3 JK flipflop



Timing Diagram



Clock	QC	QB	QA
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

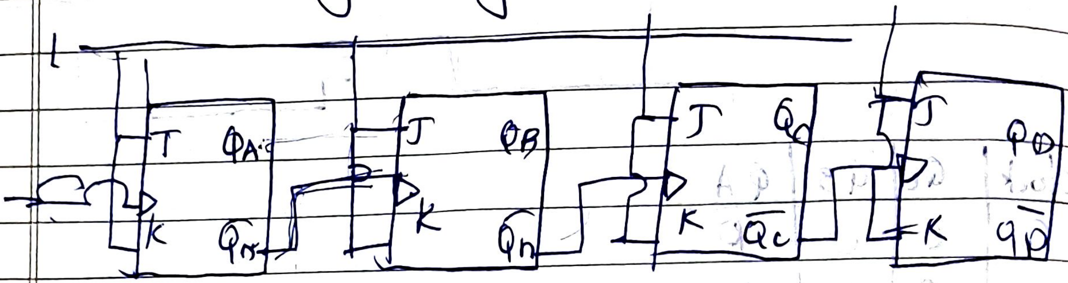
Design 4 bit Asychronous upcounter are ''. flipflops, use 4 flipflops'.

Asbychronous Down countcr:

1) The down counter will count downwards from a maximum count is zero.

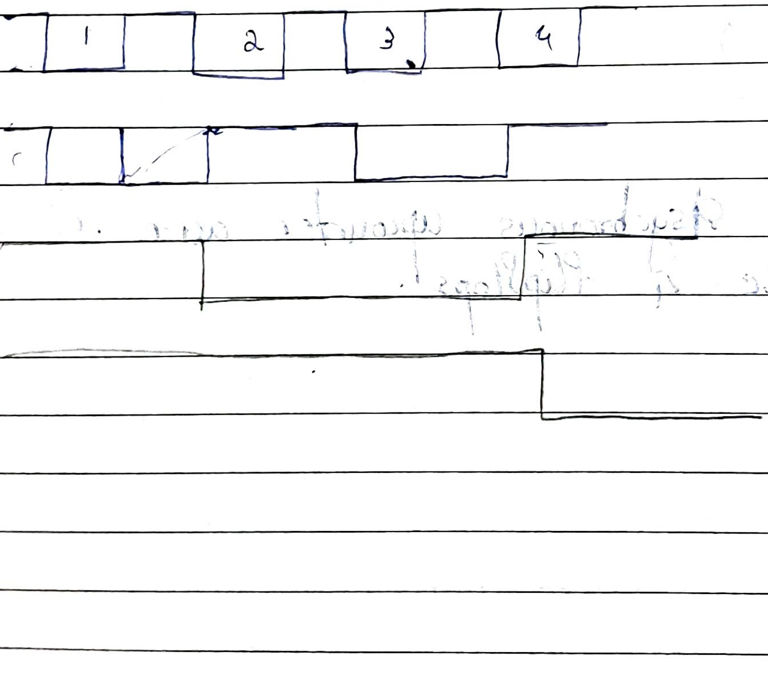
2) On 4 bit

4 bit ~~Ashyror~~ Ashychronous down counter



Modulo Counter ~~Qf~~

Eg: Modulo 0 → ~~5~~ 6 – 0 → 101 count upto 4
Module 0 → 9 → 0 → 1001 Count upto 8



Steps to design sychronous counter.

Decide the number of flipflops

↓

Excitation table of flipflops

↓

State diagram and circuit excitation table

↓

Obtain simplified equation using k-map

↓

Draw the logic diagram.

Design a mod 5 counter using JK flipflop.

① $2^n \geq N$

$2^3 > 5$

$3 \to$ No:of ffps $= 3$

②

PS	NS	J	K
Q_n	Q_{n+1}		
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

③

PS			NS			FF ip S					
Q_C	Q_B	Q_A	Q_{C+1}	Q_{B+1}	Q_{A+1}	J_C	K_C	J_B	K_B	J_A	K_A
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	0	1	1	0	X	X	0	1	X
0	1	1	1	0	0	1	X	X	1	X	1
1	0	0	0	0	0	X	1	0	X	0	X
1	~~0~~	0		OR							

Jc

Qc \ QBQA	00	01	11	10
0			1	
1	X	X	X	X

Jc = QBQA

KC

QC \ QBQA	00	01	11	10
0	X	X	X	X
1	1	X	X	X

Ø ~~KC = $\overline{QBQA}$~~ Kc = 1

JB

Qc \ QBQA	00	01	11	10
0	0	1	X	X
1	0	X	X	X

JB = QA

KB

Qc \ QAQB	00	01	11	10
0	X	X	1	0
1	X	X	X	X

KB = QA

JA

QC \ QBQA	00	01	11	10
0	1	X	X	1
1	0	X	X	X

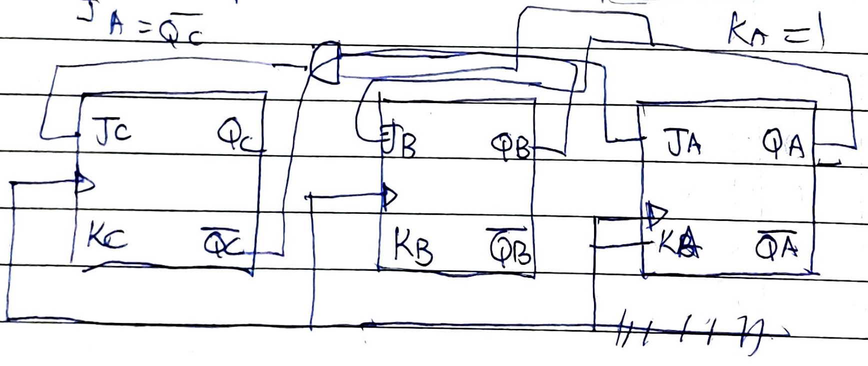
QA

JA = $\overline{Qc}$

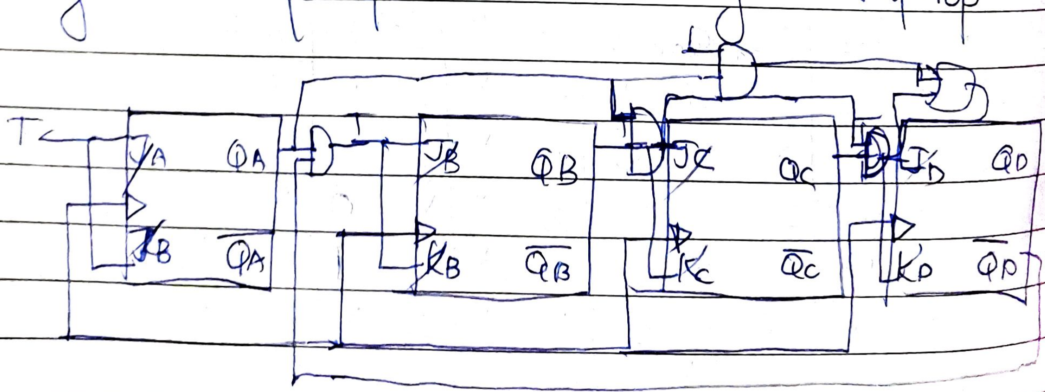
~~Jc~~ KA

QC \ QAQB	00	01	10	11
0	X	1	1	X
1	X	X	X	X

KA = 1



2 Design a decade counter using T flipflop



Redrawn

② Excitation table

Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

③ Circuit excitation table

PS				NS				~~FF i/ps~~			
Q_D	Q_C	Q_B	Q_A	Q_{D+1}	Q_{C+1}	Q_{B+1}	Q_{A+1}	T_D	T_C	T_B	T_A
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	1
0	0	1	1	0	1	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	1
0	1	0	1	0	1	1	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	1
0	1	1	1	1	0	0	0	✓1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	1
1	0	0	1	0	0	0	0	✓1	0	0	1
1	0	1	0	x	x	x	x	x	x	x	x

T_D

Q_CQ_D \ Q_AQ_B	00	01	11	10
00	0_0	0_1	0_3	0_2
01	0_4	0_5	1	0_6
11	X_{12}	X_{13}	X_{15}	X_{14}
10	0_8	1_9	X_{11}	X_{10}

→ ~~$\overline{Q_C Q_B} + Q_C Q$~~

→ ~~$Q_D Q_C + Q_C Q_A Q_B$~~

$Q_A Q_D + Q_A Q_B Q_C$

T_C

Q_CQ_D \ Q_AQ_B	00	01	11	10
00			1	
01			1	
10	X	X	X	X
11			X	X

$T_C = Q_A Q_B$

T_B

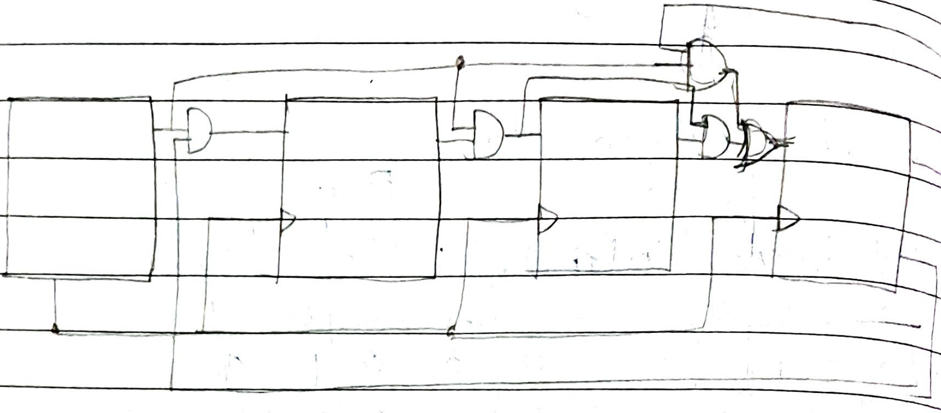
$Q_C Q_D$ \ $Q_A Q_B$	00	01	11	10
00		1	1	
01		1	1	
11	X	X	X	X
10			X	X

$T_B = Q_A \overline{Q_D}$

T_A

$Q_C Q_D$ \ $Q_A Q_B$	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	X	X	X	X
10	1	1	X	X

$T_A = 1$



? Design mod 7 counter using JKFF

? Design a 3 bit down counter (sychronous) using T flipflop

? Design a 3 bit updown counter using JK flipflop. (synchronous)

UD Qc QB QA down count n-1

1

1

Logic family/Level of Integration

Scheme	gates/chip	
Small scale Integration (SSI)	<12	(SSI)
Medium Scale Integrati	12-99	(MSI)
Large Scale Integratio	1000	(LSI)
Very large Integratio	10k	(VLSI)
ultra large Integratio	1000k	(ULSI)
Giga large Integratio	1Meg	(GSI)

~~Terms used in~~ Digital Logic family

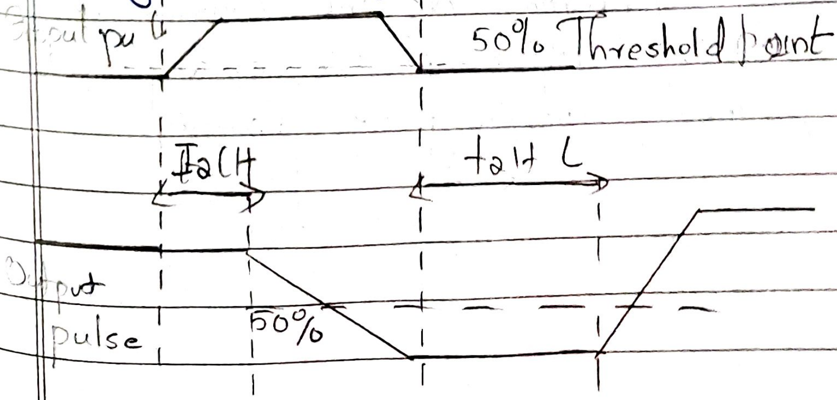
Logic families can be classified broadly according to the technology they are built with.

1) Diode Logic (DL)
2) Resistor-Transistor logic (RTL)
3) Diode-Transistor logic (DTL)
4) Emitter Couple logic (ECL)
5) Transistor-Transistor logic (
6) CMOS logic.

Terms

1) Threshold voltage - Voltage at input of gate that causes a change in state of output from 1 logic level to the another.

2) Propagation delay:



A pulse through a gate takes a certain amount of time to propogate from input to output. This interval of time is called propagation delay of gate. It is expressed as the average transition delay (tpd)

$$t_{pd} = \frac{t_{pLH} + t_{pHL}}{2}$$

Power dissipation

The power dissipation pd of a logic gate vs power required by the gate dut cycle of specified frequency is expressed using millivolt The power dissipation of a gate.

Vcc - gate supply voltage

Icc - Avg current drawn from the supply by the Ic

$$= \frac{I_{CCH} + I_{CCL}}{2}$$

Fan in

Fan in of a logic gate is defined as the number of inputs than the gate is designed to handle.

Fan out (loading factor)

Fan out is the different as the maximum no. of loads that output of the gate can drive without impairing its normal operation.

Current ~~abd~~ and voltage parameters

V_{IH}(min)

High level input voltage is the minimun voltage level required for a logical one at an input. Any voltage below this level will not be accepted as a ~~high~~ HIGH by the logic circuit.

V_{IL}(max)

Low level input voltage it is the maximum voltage level required for a logic zero at an input. Any voltage below this level will not be accepted as low by the logic circuit.

V_{OH}(min)

High level output voltage Is the minimum voltage level at logic circuit output in the logical one state under defined load conditions

V_{OL}(max)

Low level ~~inpu~~ output voltage. It the maximum voltage level at logic circuit output in the logical zero state under defined load conditions.

I_{IH}(max)

High level input current. It is the current that flows into an input when a specified high level voltage is applied to the input

I_{IL}

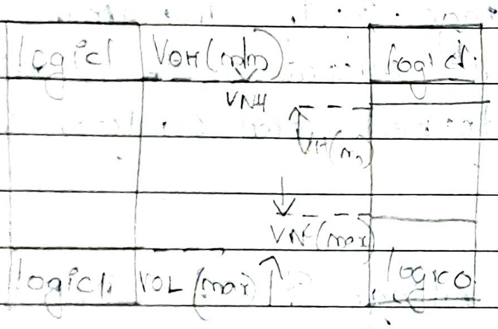
Low level input current. It is the current that

Into an input when a specified low level voltage is applied to the inpu

IOH
It is current that flows from an output in the logical one state and at specified load conditions

IOL
Low level output current. It is the current flows from an output in logical zero state at specified load condit.

NOISE MARGIN



Family	logic 0	logic 1
TTL	0V	+5V
CMOS	0V	3-15V
ECL	-1.7V	-0.9V

Due to ~~straight~~ stray electric and magnetic feild that Induces voltages ~~on~~ the connecting wires between logic circuits. This unwanted signals noise. Sometimes cause the voltage at the input to drop below VH minimum or rise above V_{IL} max.

When the operate in noisy environment the gates may malfanction if noise is beyond certain limits.
Noise immunity of a logic circuit refers do the circuits ability to tolerate noise voltages at its inputs. A quantitative measure of noise immunity is called noise margin. It is expressed in volt and represents maximum noise signal that can be added to the input signal of a digital circuit without causing an undesirable change in circuit output

The noise margin may be highstate noise margin or low state noise margin. High state noise margin is given as

$$V_{NH} = V_{OH(min)} - V_{IH(min)}$$

Low state noise margin is given as $V_{NL} = V_{IL(max)} - V_{OL(max)}$

Speed power product

It is the common mean for measuring and comparing all over all performance of an IC family. It is obtained by multiplying the gate propagation delay by the gate power dissipation.
A low value of speed ~~pow~~ power product is most desirable. It is also called as figure of merit of an IC family.

It is usually expressed in pico joules. (10^{-12})

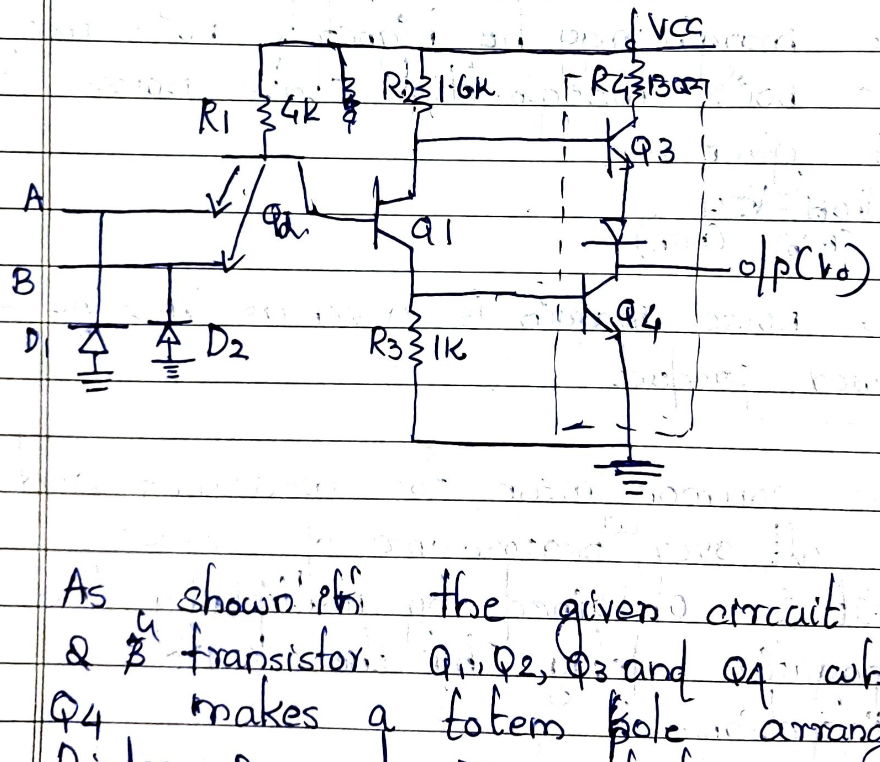
TTL (T^2L) Transistor Transistant Logic

It is the most popular logic family. It is most widely used bipolar digital IC family.

TTL family consist of several sub families.

1) Standard TTL
2) High speed TTC
3) Low power TTC
4) Schottky TTL
5) Low power Schottky TTL
6) Advanced Schottky TTL
7) Advanced low power Schottky TTL
8) Fast TTL

2 INPUT TTL NAND GATE



As shown in the given circuit there are & ~~3~~ 4 transistor Q1, Q2, Q3 and Q4 where Q3 and Q4 makes a totem pole arrangement. Dciodes D1 and D2 protect Q1 from being damaged by the negative spike of voltages at the inputs. By by passing the spikes to the ground. There is ~~an~~ another diode D which ensures Q3 and Q4 are not conducting simultaneously.

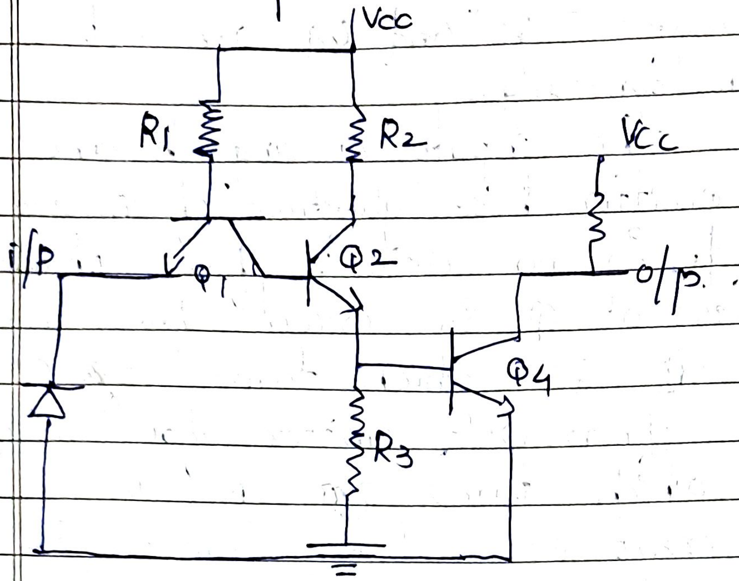
When both the inputs a and b are high both emitter base junctions of Q1 are reverse biased. So no current flows through the emitters of Q1. So current flows through R1 to the base of Q2 which turns Q2 ON. The current from Q2's emitter flows into the base of Q4 which turns Q4 ON. The collector current Q2 flows through R2 producing a drop across it and reduces the voltage and collector of Q2 which makes Q3 off. Since Q4 is ON Vo is ~~s~~ at a low level so the output is at logic 0

When either a or b or both low the base emitter junctions of Q1 are forward biased and emitter base junction of Q is reverse biased. The current flows to ground through the emitters of Q1 Therefore it can't be forward biased to base emitter junction of Q2 which makes Q2 off. When Q2 is off Q4 does not get the required base voltage and ∴ Q4 is off. Since no current flows into the collector of Q2 all current flows current to base of Q3 which makes Q3 ON

Therefore the output voltage Vo is at a logic high level or logic 1

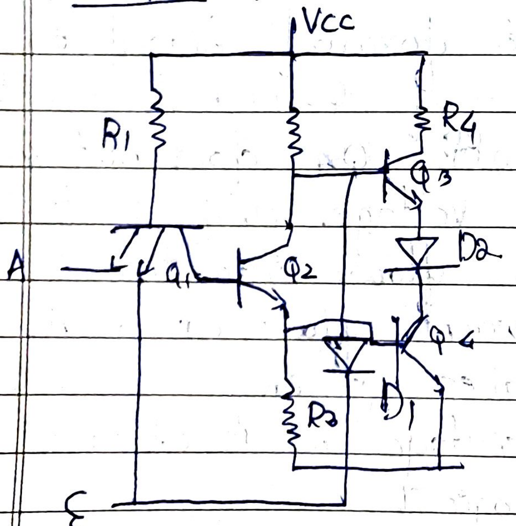
twelvesth..

TTL with open collector invertor



In open collector TTL the output is at the collector of Q4. The Inorder to get the proper high and low logic levels are external pull up resistor is connector to Vcc Q4 so through Q4. When Q4 is off output is pulled to Vcc through the resistor R. When Q4 is ON output is connected to ground through the saturated transistor.

Tristate TTL



The fig shows a tristate TTL because this high low and high impedance State. Both the transistor in the totem pole configuration are turned off. So that the o/p is a high impedance is ground or Vcc

∴ The o/p neither a low or high circuit has 2 inputs. A is the normal logic input and E is the enable i/p which can produce a high impedance.

When E=1 the circuit operates like a normal inverter. Because it has no effect on Q1 or Q2 the output is the inversion of logic input A.

When E=0 circuit goes into high impedance state regardless of the state of logic input A. The low at E forward biases the emitter base junction of Q1 and shunts the current in R1 away from Q2 so that Q2 turns off which interms makes Q4 off.

The low at E forward biases diode D1 to shunt away current at base of Q3. and ∴ Q3 is also off. If both the Q3 and Q4 are off the output terminal is essential and turned off.

CMOS

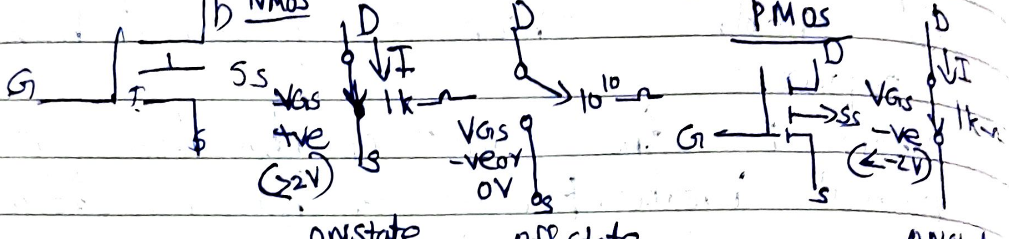
Mos logic

It is simplest fabricate an occupies very small space. Because it requires only 1

It is idealy suited for LSI, VLSI and ULSI applications such as microprocessors, memory elements etc.

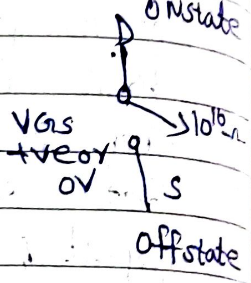
There are 2 types of mosfet

1) Depletion type
2) Enhancement type
3) Mosfet also can be Nmos or Pmos type.



Cmos

Cmos ~~logic family uses p and n channel mosfet in~~ same cir.

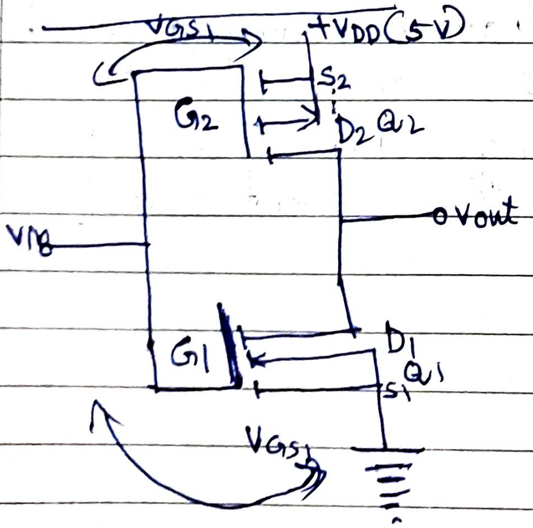


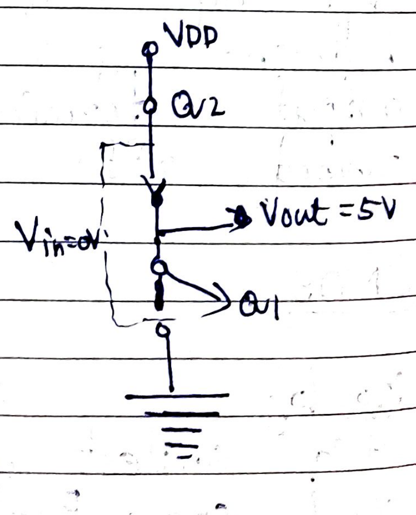
C Mos

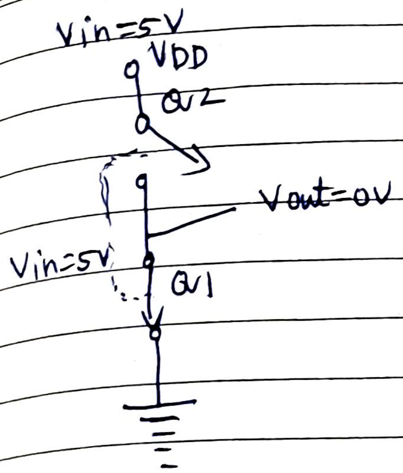
The CMos logic family uses both P&n Channel MosFET in the same Circuit. TheCMos family consume less power thanother Mos families.

CMos inverter

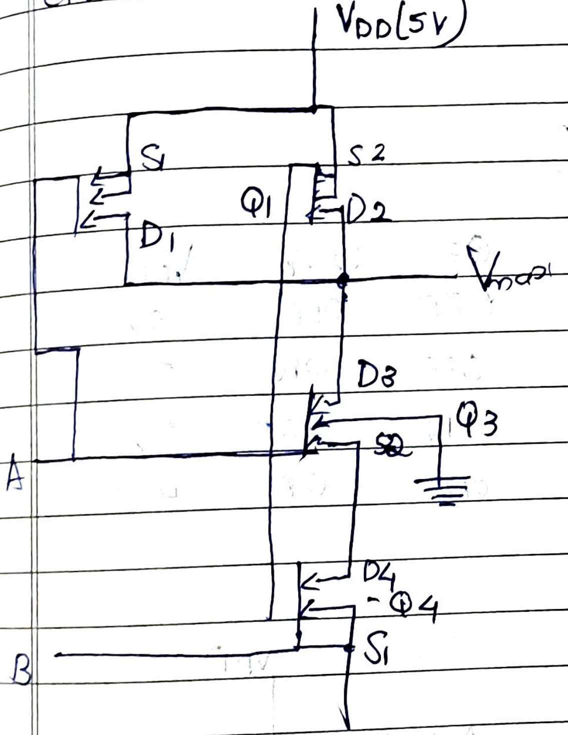
equivalent circuit of CMos

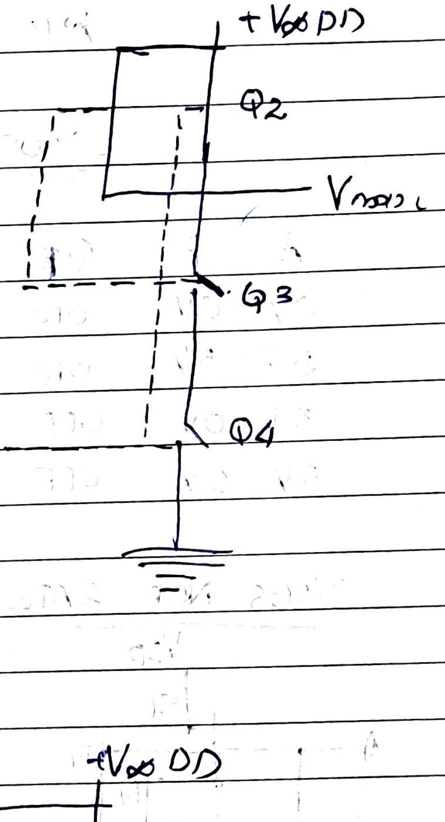


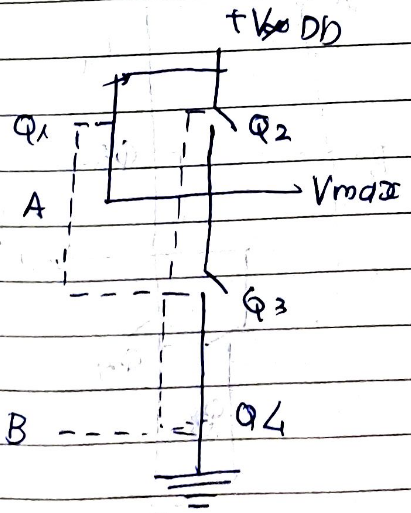


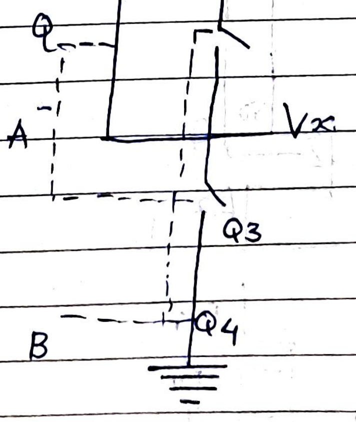


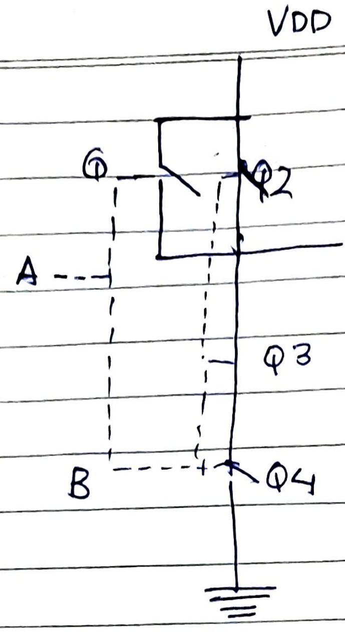
CMOS NAND GATE







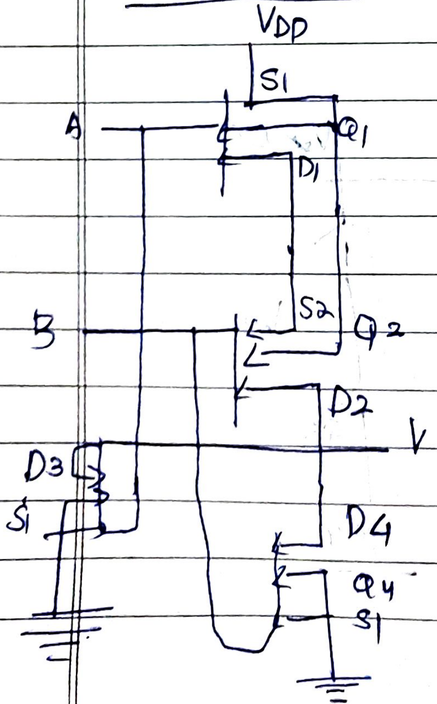


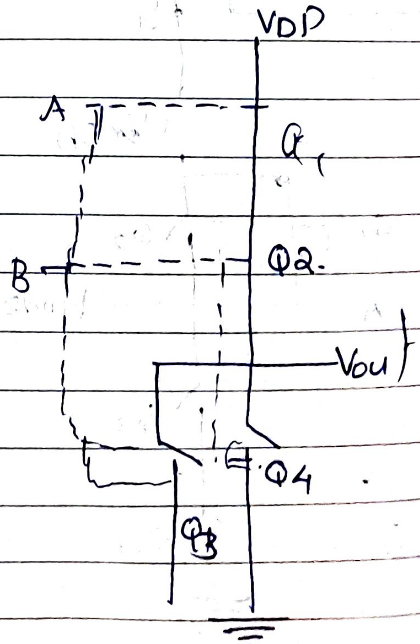


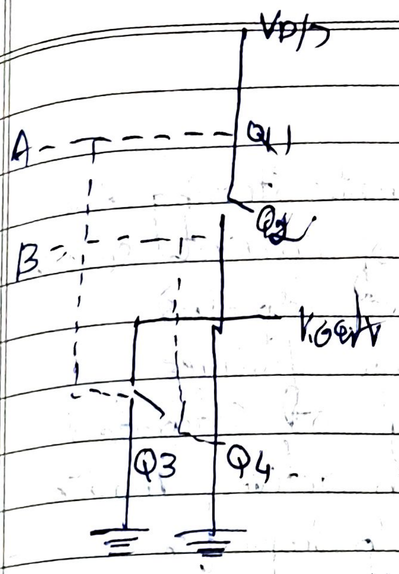
CMOS NAND

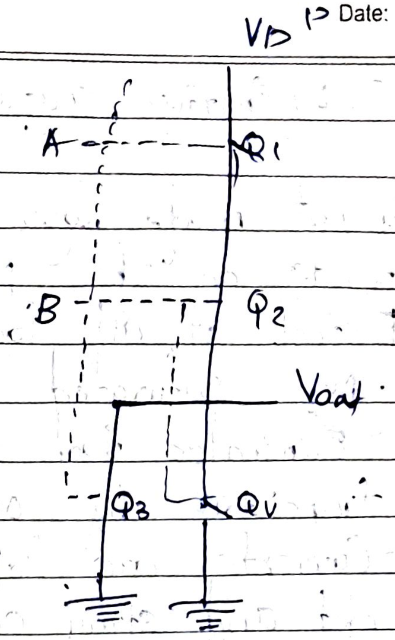
A	B	Q1	Q2	Q3	Q4	VDD
0V	0V	ON	ON	OFF	OFF	5V
0V	5V	ON	OFF	OFF	ON	5V
5V	0V	OFF	ON	ON	OFF	5V
5V	5V	OFF	OFF	ON	ON	0V

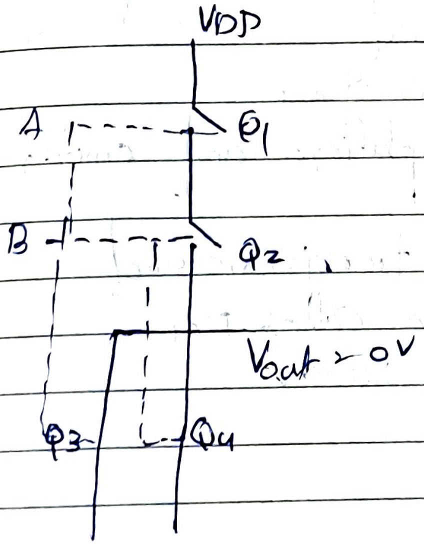
CMOS NOR GATE











A	B	Q1	Q2	Q3	Q4	Vout
0V	0V	ON	ON	OFF	OFF	5V
0V	5V	ON	OFF	OFF	ON	0V
5V	0V	OFF	ON	ON	OFF	0V
5V	5V	OFF	OFF	ON	ON	0V

Truth Table

ECL (Emitter coupled Logic)

1) Current mode logic (CML) / Current-Steering Logic
2) Operates on the principles of current switching
3) Fastest of all logic families (tp = 1ns).
 - Non saturated digital logic families
 - Eliminates turn off delay of saturated transistors by operating in active mode
 - Currents are kept high o/p impedance is low so ckt and stray capacitance can be quickly charged and discharged
 - Has limited voltage swing
4) Consists of different amplifiers and emitter followers
5) Emitter terminals of 2 transistors are tied together and hence called ECL
6) Logic Low +7V Logic high : 0.9 V