NUMBER SYSTEM

The are two main categories of the data used in digital computers are :-

(a) NUMBERS.

(b) CHARACTORS.

There are four types of Number System:

1) Decimal Number System
2) Binary Number system
3) Octal Number System
4) Hexadecimal Number System

1. DECIMAL NUMBER SYSTEM :-

In a decimal number system there are 10 digits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 the base of this number is 10.

(1000th) (100th) (Tenth)

Ex:- $2345 = 2\times10^{3} + 3\times10^{2} + 4\times10^{1} + 5\times10^{0}$

$45.23 = 4\times10^{1} + 5\times10^{0} + 2\times10^{-1} + 3\times10^{-2}$

(Tenth) (unit) ($\frac{1}{10^{th}}$) ($\frac{1}{100^{th}}$)

2 BINARY NUMBER SYSTEM:

In a Binary number System these are only two digits 0 and 1

The base of this number is 2.

Ex:

$(101)_2 \rightarrow 1\times2^{2} + 0\times2^{1} + 1\times2^{0}$

$= 4 + 0 + 1 = 5$ Ans.

3. Octal NUMBER SYSTEM :-

In a Octal number system there are 8 digits 0, 1, 2, 3, 4, 5, 6, 7 The Base of the this number is 8

Ex: $(25)_8 = 2 \times 8^1 + 5 \times 8^0$

$= 16 + 5 = 21$

4. HEXADECIMAL NUMBERS SYSTEM :

In hexadecimal number system the digits are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 and A, B, C, D, E, F which corresponds to 10, 11, 12, 13, 14, 15

The Base of the this number is

Ex: $(31)_{16}$ —— $3 \times 16^1 + 1 \times 16^0$

$= 3 \times 16 + 1 = 48 + 1 = 49$ dus

CONVERSIONS :-

1. DECIMAL TO BINARY

$(81)_{10} \rightarrow (1010001)_2$

2	81	1
2	40	0
2	20	0
2	10	0
2	5	1
2	2	0
	1	

2. DECIMAL TO OCTAL

8	266	2
8	33	1
	4	

$(266)_{10} \longrightarrow (412)_8$

$(258)_{10} \longrightarrow (402)_8$

3. DECIMAL TO HEXADECIMAL

16	423	7
16	26	10
	1	

$(423)_{10} \longrightarrow 1\ 10\ 7$

$(1\ A\ 7)_{16}$

$(214)_{10} \longrightarrow (D6)_{16}$

4. BINARY TO DECIMAL

$(\overset{4}{1}\overset{3}{1}\overset{2}{0}\overset{1}{1}\overset{0}{1})_2 \longrightarrow (27)_{10}$

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

$= 16 + 8 + 2 + 1 = 27$

$(101001)_2 \longrightarrow (41)_{10}$

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

$= 32 + 8 + 1 = 41$

5. BINARY TO OCTAL

$(100\ 111\ 010)_2 \longrightarrow (4\ 7\ 2)_8$

$(111\ 010\ 001)_2 \longrightarrow (721)_8$

6. BINARY TO HEXADECIMAL:

$(\overbrace{10}^{2}\overbrace{1011}^{B}\overbrace{010}^{A})_2 \longrightarrow (2\ B\ A)_{16}$

$(2\ \ 11\ \ 10)_2$

7. Octal FO DECIMAL

$(372)_8 \longrightarrow (250)_{10}$

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

$= 3\times64 + 56 + 2 = 250$

8. OCTAL TO BINARY

$(472)_8 \longrightarrow (100\ 111\ 010)_2$

0	1	2	3	4	5	6	7
000	001	010	011	100	101	110	111

(9) HEXADECIMAL TO DECIMAL:

$(\overbrace{3}^{2}\overbrace{5}^{1}\overbrace{6}^{0})_{16} \longrightarrow (854)_{10}$

$= 3\times16^2 + 5\times16^1 + 6\times16^0$

$= 768 + 80 + 6$

$= 854$

$(432)_{16} \longrightarrow (1074)_{10}$

$= 4\times16^2 + 3\times16^1 + 2\times16^0$

$= 4\times256 + 48 + 2$

$= 1024 + 50 = 1074$

10. HEXADECIMAL TO BINARY:

$(9F2)_{16} \longrightarrow (1001\ 1111\ 0010)_2$

9 → 1001

F=15 → 1111

2 → 0010

$(3A6)_{16} \longrightarrow (0011\ 1010\ 0110)_2$

3 → 0011

A=10 → 1010

6 → 0110

FRACTIONS :-

1. DECIMAL TO BINARY

$(0.375)_{10} \longrightarrow (0.011)_2$

$0.375 \times 2 = 0.750$

$0.75 \times 2 = 1.50$

$0.50 \times 2 = 1.00$ ↓

2	38	0
2	19	1
2	9	1
2	4	0
2	2	0
	1	

$(38.21)_{10} \rightarrow (100110.0011\ 0101\ 110)_2$

$38 = (100110)_2$

$(0.21)_{10} = (0011\ 0101\ 110)_2$

$0.21 \times 2 = 0.42$	0
$0.42 \times 2 = 0.84$	0
$0.84 \times 2 = 1.68$	1
$0.68 \times 2 = 1.36$	1
$0.36 \times 2 = 0.72$	0
$0.72 \times 2 = 1.44$	1
$0.44 \times 2 = 0.88$	0
$0.88 \times 2 = 1.76$	1
$0.76 \times 2 = 1.52$	1
$0.52 \times 2 = 1.04$	1
$0.04 \times 2 = 0.08$ ↓	0

$(0.21)_{10} \rightarrow (0011010111 0)_2$

$(38.21)_{10} \longrightarrow (100110.0011010111 0)_2$

2. DECIMAL TO OCTAL :

$(0.375)_{10} \longrightarrow (0.3)_8$

$0.375 \times 8 = 3.0$

$(0.015625)_{10} \longrightarrow (0.01)_8$

$0.015625 \times 8 = 0.125$ $\quad 0.0$

$0.125 \times 8 = 1.0$ ↓

3. DECIMAL TO HEXADECIMAL :

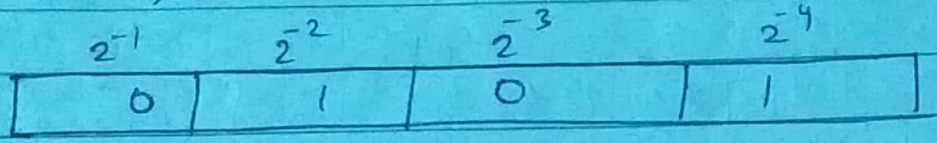
$(0.03125)_{10} \longrightarrow (0.08)_{16}$

$0.03125 \times 16 = 0.5$

$0.5 \times 16 = 8.0$ ↓

4. BINARY TO DECIMAL:

$(0.0101)_2 \longrightarrow (0.375)_{10}$



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

$= 0 + 0.25 + 0.0625$

$= 0.375$

$(1101.000101)_2 \longrightarrow (13.078125)_{10}$

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

$\Rightarrow 8 + 4 + 0 + 1 + 0 + 0 + 0 + \frac{1}{16} + 0 + \frac{1}{64}$

$= 13 + 0.0625 + 0.015625$

$= 13.078125$

5. BINARY TO OCTAL

$(010110.0101)_2 \longrightarrow (26.24)_8$

010 110 . 010 100

2 6 . 2 4

6. BINARY TO HEXADECIMAL

$(10101110.01011)_2 \longrightarrow (AE.5C)_{16}$

1010 1110 . 0101 1100

10 14 . 5 12

7. OCTAL TO DECIMAL:

$(24.6)_8 \longrightarrow (20.75)_{10}$

$2 \times 8^{1} + 4 \times 2^{0} + 6 \times 8^{-1}$

$16 + 4 + \frac{3}{4}$

$20 + 0.75 = 20.75$

8. OCTAL TO BINARY

$(3.1)_8 \rightarrow (011.001)_2$

$3 = 011$

$1 = 001$

9. HEXADECIMAL TO DECIMAL:

$(56.08)_{16} \longrightarrow (86.03125)_{10}$

$5 \times 16^{1} + 6 \times 16^{0} + 0 \times 16^{-1} + 8 \times 16^{-2}$

$= 80 + 6 + 0 + \frac{8}{256}$

$= 86 + 0.03125 = 86.03125$

10. HEXADECIMAL TO BINARY

$(3BF.5C)_{16}$ $(0011\,1011\,1111.0101\,1100)_2$

3 → 0011

B → 1011

F → 1111

5 → 0101

C → 1100

Sign and magnitude Representation

This is the convential form for number representation. Integers are identified by their signs (+ or −) and a string of digits which represent the magnitude.

For example: +17 or 23 are positive integers
−14 −15 are negative numbers.

To represent sign of a number (i.e., + or −), utmost bit (called as Most significant Bit) MSB is used. If it holds value 0, the sign is + and if holds 1 the sign is −.

For example, if in a computer, the word size is 1 byte (8 bits), then +15 will be represented as follows.

Number in binary notation.

0	0	0	0	1	1	1	1

Most significant bit (MSB) (0 for + sign)

Least significant bit (LSB)

Binary equivalent of 15 is 1111

−23 will be represented as

1	0	0	1	0	1	1	1

MSB
1 for − sign.

Binary equivalent of 23 is 10111.

In above binary words of 8 bits, MSB is reserved for sign notation. Therefore the maximum magnitude which can be represented is of 7 bits.

Maximum number which can be represented in 8 bits signed notation is $2^7 = 128$. An 8 bit word can represent total $2^8 - 1 = 255$ numbers i.e., -127 to 0 to +128. 16 bits (2bytes) binary word can represent maximum $2^{15} = 32768$ no. One bit MSB is reserved for sign notation. And total $(2^{16} - 1 = 65535)$ no. it can present are -32767 to 0 and 0 to 32767.

ONE's complement Representation :-

It represents positive numbers by their binary equivalent (called true forms) and negative no. by their 1's complements (called 1's complement forms).

To calculate 1's complement of a binary number, just replace every 0 with 1 and every 1 with 0.

EX:- Find the one's complement form of -13.

Sol +13 = 0000 1101

-13 = 1111 0010

Find the one's complement representation of -13.

Sol:

```
  1111 1111
- 0000 1101
-----------
  1111 0010
-----------
```

Two's complement Representation:

Two's complement method represents positive number in their true forms i.e., their binary equivalent and negative numbers in 2's complement form.

2's complement of a number is calculated by adding 1 to its 1's complement.

6 — 0110 2's complement will be calculated as follows.

1's complement of 0110 = 1001
2's complement of 0110 = +1
1010

NUMBER OF REPRESENTATION:

(i) signed magnitude.
(ii) one's complement.
(iii) Two's complement.

Sign magnitude

1 Bit 6 Bit

7 BIT.

(i) SIGNED MAGNITUDE :-

	+6	−6
	0 000110	1 000110

(ii) One's complement

	+6	−6
	0 000110	1 111001

(iii) Two's complement

	+6	−6
	0 000110 ←	1 111010

Binary	Decimal	
0111	+7	1011 — −5
0110	+6	1010 — −6
0101	+5	1001 — −7
0100	+4	1000 — −8
0011	+3	
0010	+2	
0001	+1	
0000	0	
1111	−1	
1110	−2	
1101	−3	
1100	−4	

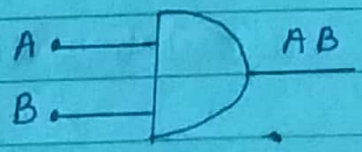
LOGIC GATES :-

Def: It is a electronic circuit which have number of input and it gives us one output is called logic gates.

There are so many logic Gates :-

(1) AND
(2) OR
(3) NOT
(4) NAND
(5) NOR
(6) XOR

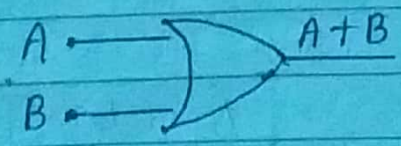
1) AND LOGIC GATE :-



TRUTH TABLE

A	B	AB
T	T	T
T	F	F
F	T	F
F	F	F

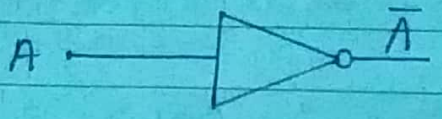
2) OR LOGIC GATE :-



TRUTH TABLE

A	B	A+B
T	T	T
T	F	T
F	T	T
F	F	F

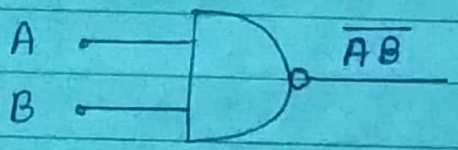
3) NOT LOGIC GATE :-



A	$\bar{A}$
T	F
F	T

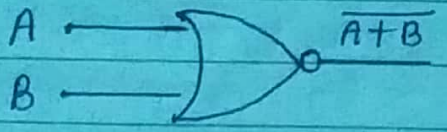
4) NAND LOGIC GATE :-

This Gate is a combination of NOT and AND



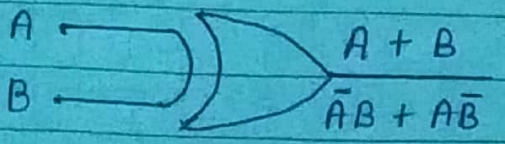
A	B	AB	$\overline{AB}$
T	T	T	F
T	F	F	T
F	T	F	T
F	F	F	T

5) NOR LOGIC GATE :- Combination of NOT OF OR



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

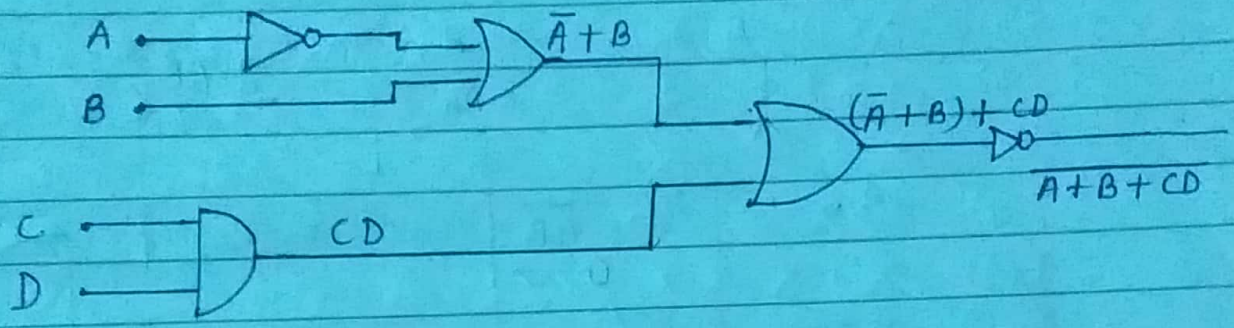
6) XOR LOGIC GATE :-



A	B	A⊕B
T	T	F
T	F	T
F	T	T
F	F	F

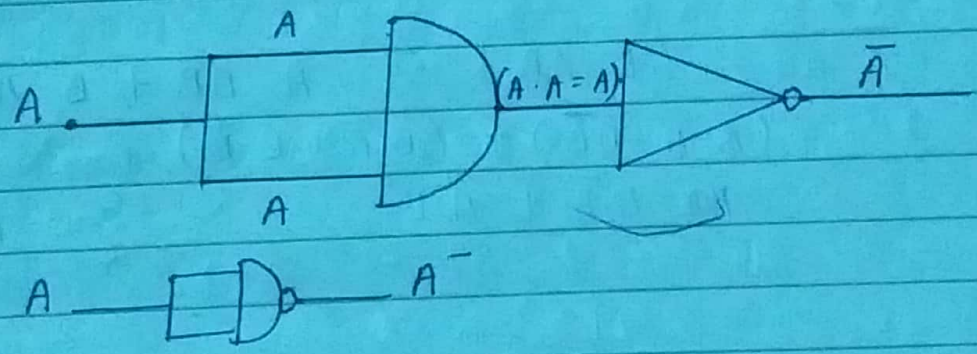
DRAW A CIRCUIT DIAGRAMME :-

$\bar{A} + B + CD$

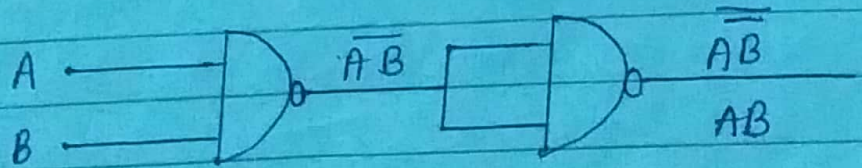


PROVE NAND IS UNIVERSAL GATE :-

(i) NOT

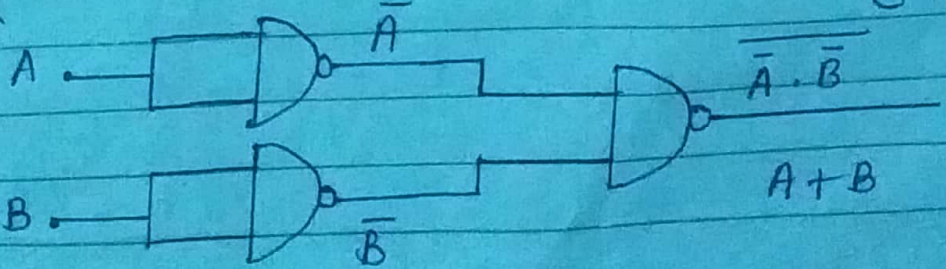


(ii) AND

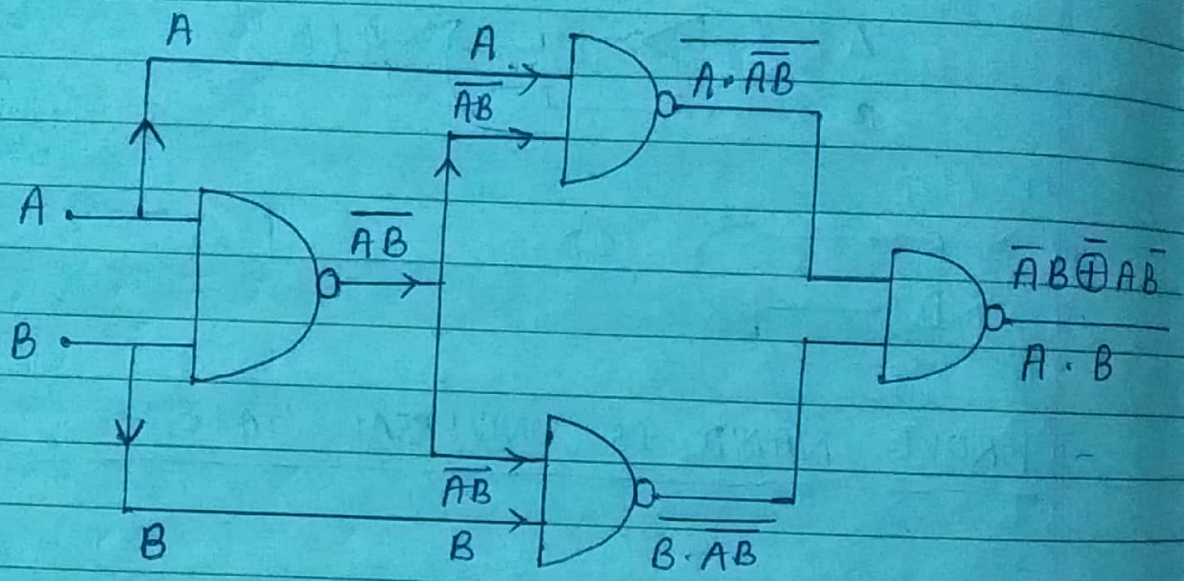


$\{\bar{\cdot} = +$
$\bar{+} = \cdot\}$

(iii) OR



Write a circuit Diagramme for XOR Gate using NAND GATE.



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

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

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

$\{\because A \cdot \bar{A} = F(0)$

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

Date
Page

CODE CONVERSION

(i) Binary to Gray code.

(ii) Gray code to Binary.

Binary to Gray code :-

Binary → 1 1 0 0 0 1 0 1

Gray code → 1 0 1 0 0 1 1 1

Gray code → 1 0 1 0 0 1 1 1

↓ ↑ ↓ ↑ ↓ ↑ ↓ ↑

Binary : → 1 1 0 0 0 1 0 1

BOOLEAN ALGEBRA :

George Bool, George Bole, Boole.
He is enverted in 1854
It (Boolean algebra) states that human logic can be represented mathematically using true or false 0 or 1.

BOOLEAN THEROM :-

(i) $0 \cdot X = 0$
(ii) $1 \cdot X = X$
(iii) $X + 0 = X$
(iv) $X + 1 = 1$
(v) $X \cdot X = X$
(vi) $X \cdot \overline{X} = 0$
(vii) $X + X = X$
(viii) $X + \overline{X} = 1$
(ix) $\overline{\overline{X}} = X$
(x) $X + XY = X$
(xi) $X(X + Y) = X$
(xii) $X + \overline{X}Y = X + Y$

Q1. PROVE THAT $X + XY = X$ BY Boolean Algebra

→ In Algebric method :- $X + XY = LHS$

$X(1 + Y) = X$

$X \cdot 1 = X$

$X = RHS$

$X + XY = X$ proved.

In Truth Table

X	Y	XY	X+XY
T	T	T	T
T	F	F	T
F	T	F	F
F	F	F	F

$X = X + XY$ proved

Q2. Prove that $X+\bar{X}Y = X+Y$

L.H.S = $X+\bar{X}Y$

$\Rightarrow X+XY+\bar{X}Y$

$\Rightarrow X+Y(X+\bar{X})$

$\Rightarrow X+Y = R.H.S$

$\boxed{X+\bar{X}Y = X+Y}$ proved.

Q3. SIMPLYFY

$X+\bar{X}Y+\bar{Y}+(X+\bar{Y})\cdot\bar{X}Y \quad (\because X+\bar{X}Y = X+Y)$

$\Rightarrow X+Y+\bar{Y}+YX\bar{X}+\bar{X}\bar{Y}Y$

$\Rightarrow X+1+X\cdot\bar{X}Y+\bar{X}\bar{Y}\cdot Y$

$\Rightarrow X+1+0\cdot Y+\bar{X}\cdot 0$

$\Rightarrow X+1 = 1$

$\boxed{X+\bar{X}Y+\bar{Y}+(X+\bar{Y})\cdot\bar{X}Y = 1}$ Ans.

Q. CONVERSION of Sum of products expression into canonical form & introduce missing terms.

Sol. convert $X+X\bar{Y}$ into canonical form.

$X+X\bar{Y}$

$\Rightarrow X\cdot 1+X\bar{Y}$

$\Rightarrow X(Y+\bar{Y})+X\bar{Y}$

$\Rightarrow XY+X\bar{Y}+X\bar{Y} = XY+X\bar{Y}$ Ans.

K-MAP

Boolean Function

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

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

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

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

$$= AB.$$

K- MAP

Karnaugh Maps

(i) SUM OF PRODUCTS (Σ)

(ii) PRODUCT OF SUM (π)

Q1. $F(A,B,C,D) = \Sigma(0,5,6,9,10,11,12,13,14,15)$

Sol

AB \ CD	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	(1) 0	1	3	2
$\bar{A}B$	4	(1) 5	7	(1) 6
AB	(1) 12	(1) 13	(1) 15	(1) 14
$A\bar{B}$	8	(1) 9	(1) 11	(1) 10

PAIR 1 = P_1

AB \	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
AB	(1) 12	(1) 13	(1) 15	(1) 14

$AB\bar{C}\bar{D} + AB\bar{C}D + ABCD + AB C\bar{D}$

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

$= AB\bar{C} + ABC$

$= AB(\bar{C}+C)$

$\boxed{P_1 = AB}$

PAIR 2 = P_2

AB \	CD	$C\bar{D}$
AB	(1) 5	(1) 14
$A\bar{B}$	(1) 11	(1) 10

$P_2 = ABCD + ABC\bar{D} + A\bar{B}CD + A\bar{B}C\bar{D}$

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

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

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

$\boxed{P_2 = AC}$

P_3 =

	$\bar{C}D$	CD
AB	$(1)_{13}$	$(1)_{15}$
$A\bar{B}$	$(1)_9$	$(1)_{11}$

$P_3 = AB\bar{C}D + ABCD + A\bar{B}CD + A\bar{B}\bar{C}D$

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

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

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

$\boxed{P_3 = AD}$

Pair 4 = P_4 =

	$\bar{C}D$
$\bar{A}B$	$(1)_5$
AB	$(1)_{13}$

$P_4 = \bar{A}B\bar{C}D + AB\bar{C}D$

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

$\boxed{P_4 = B\bar{C}D}$

Pair 5 = P_5

	$C\bar{D}$
$\bar{A}B$	$(1)_6$
AB	$(1)_{14}$

$P_5 = \bar{A}BC\bar{D} + ABC\bar{D}$

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

$\boxed{P_5 = BC\bar{D}}$

Pair 6 = P_6

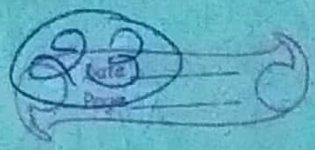
AB \ CD	$\bar{C}\bar{D}$
$\bar{A}\bar{B}$	$(1)_0$

$\boxed{P_6 = \bar{A}\bar{B}C\bar{D}}$

$\therefore$ Sum of Product = $P_1 + P_2 + P_3 + P_4 + P_5 + P_6$

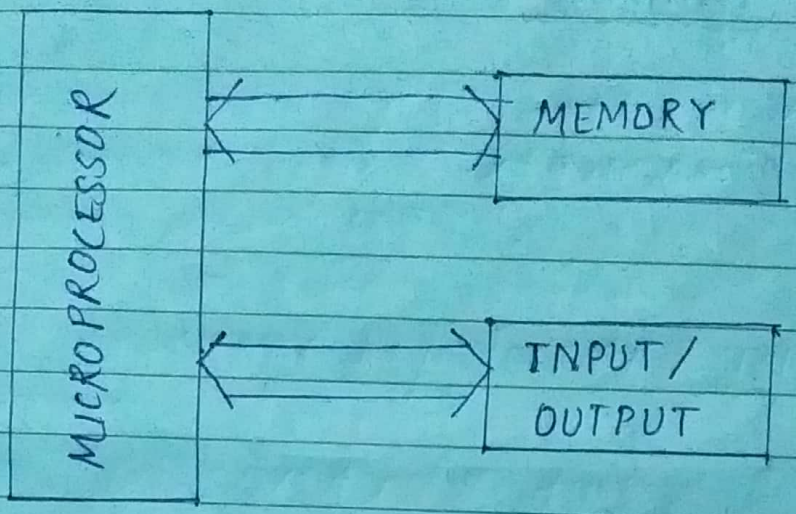
$\boxed{S.O.P = AB + AC + AD + BC\bar{D} + B\bar{C}D + \bar{A}\bar{B}C\bar{D}}$

Ans



CIRCUIT DIAGRAM :-

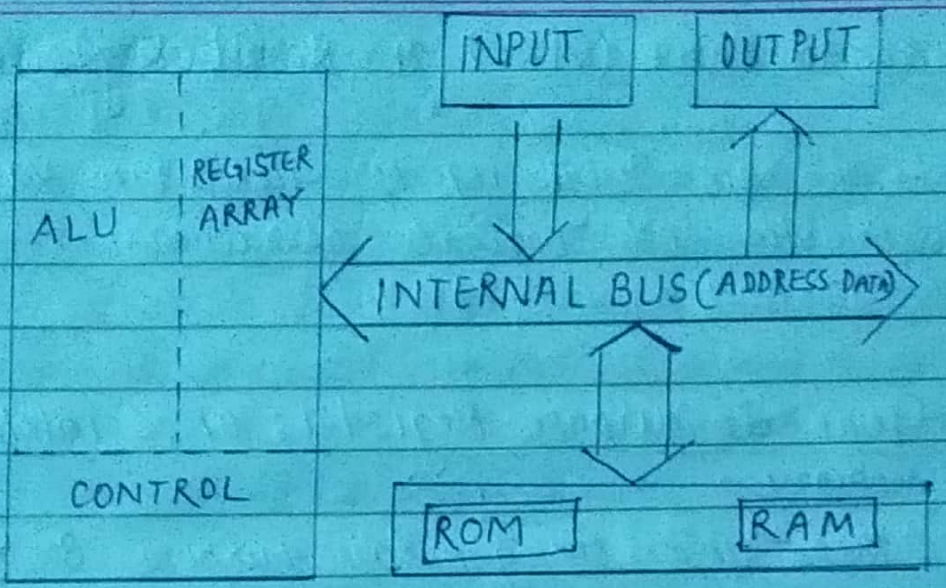
MICRO PROCESSORS



DEFINITION :-

The microprocessor reads each instruction from the memory, decodes it and executes it. It processes the data as required in the instructions. The processing is in the form of arithmetic and logical operations. The data is retrieved from memory or taken from an input device and the result of processing is stored in the memory or delivered to an appropriate output device, all as per the instructions.

First microprocessor : 1971 by Intel corp. U.S.A
It is 4004 Intel.
It was a single chip & 4-bit μp (i.e. operated on 4 bits of data at a time).



Architecture of Microprocessor.

Arithmetic - Logic Unit (ALU)

The arithmetic - logic unit is a combinational network that performs arthimatic and logical operations on the data.

Internal Registers:

A number of registers are normally included in the microprocessor. these are used for temporary storage of data, instructions and and addresses during execution of a program.

8085 microprocessor are typical and are described below:

(I) Accumulator (Acc) or Result Register.

This is an 8 bit register used in various arithmetic and logical operations.

(II) General purpose Registers or Scratch Pad Memory.

There are six general purpose 8-bit registers that can be used by the programmer for a variety of purposes. These registers, labelled as B, C, D, E, H and L can be used individually (e.g.): ~~when~~

When a 16-bit address is to be stored only B-C, D-E & H-L pairs are allowed.

(III) Program Counter (PC)

This is a 16-bit register which holds the address of the next instruction that has to be fetched from the main memory and loaded into the instruction register.

Examples are instructions in the "Jump" and 'Call Subroutine' groups.

(iv) Instruction Register (IR):

This 8-bit register stores the next instruction to be executed. At the proper time this store word (instruction) is fed to an instruction decoder which decodes it and supplied appropriate signals to the controle unit.

(v) Stack pointer (SP):

This is also a 16-bit register and is used by the programmer. To maintain a stack in the memory while using subroutines.

(vi) Status Register or Conditions Flags:-

A status register consisting of a few flip-flops. called as condition flags (in 8085 the number of flags is five).

is used to indication of certain conditions that arise during arithmetic and logical operations. These are.

'zero' → Flag is set if result of instruction is 0.
'sign' → set if MSB of result is 1.
'parity' → set if result has even parity.
'carry' → set if carry or borrow resulted.
'auxiliary-carry' set if instruction caused a carry out of bit 3 and into bit 4 of the resulting value.

(i) ZERO FLAG REGISTERS

Present of any zero called zero flag.

```
  1101 1101
  1111 1001
1 1101 0110
```

ZERO FLAG (ZF) = [0]

(ii) SIGN FLAG REGISTERS

S·F [1] carry is 1
carry is 0

(iii) PARITY FLAG REGISTERS

Number of 1 are Even then P·F is [1]
1 are odd then P.F is [0]

(iv) CARRY FLAG REGISTERS

Remainder or carry is 1 C·F [1]
Remainder or carry is 0 C·F [0]

(v) AUXILIARY CARRY FLAG REGISTERS

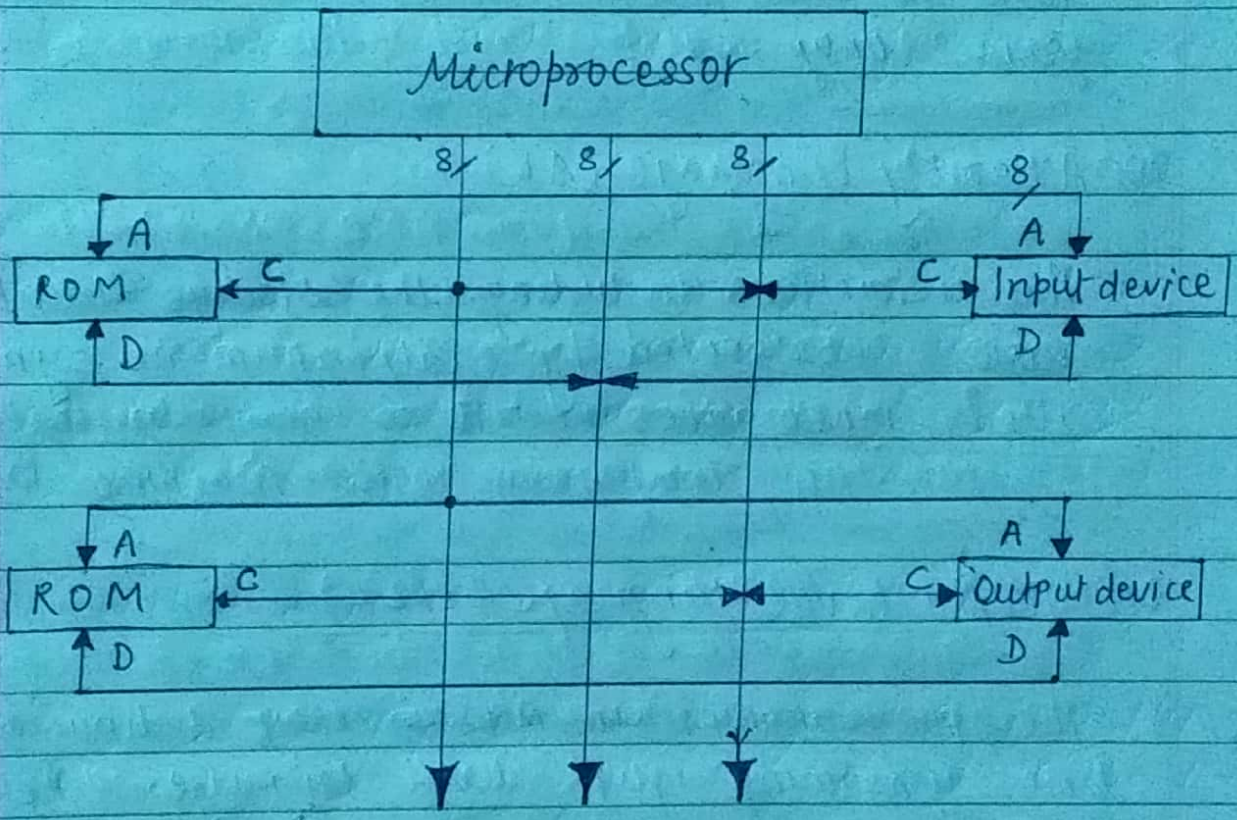
Between carry is possible in D3 to D4 then value of Carry is set in box.

A·C·F [1]

(VII) Dedicated Registers:

several other registers are incorporated in the μp (micro processor) for its internal operation.

They cannot be accessed by the programmer and hence do not concern much as μp user.



Connection of Input/Output Devices And Memory.

1.1 SOFTWARE AND PROGRAM LANGUAGES

Programming Languages

(I) Machine Language (ML)

It store and process information in binary form only.

(II) Assembly Language (AL)

The instructions including the storage locations are represented by alphanumeric symbols, called mnemonics. It is easy to write in comparison with Machine languag

(III) High level language (HLL):

The programming in AL is very tedious and time consuming High level languages like Fortran, Cobol, Algol, Pascal & PL/M programmes translated into (M.L) program

SOFTWARE TOOLS

(1) Assembler:

It is a computer programe that translates an AL program to ML program (also called object code).

(ii) Compiler:

A computer program that translates a HLL program to ML program (object code).

(iii) Editor

An editor make changes in the program text in order to correct any errors or modify the logic.

(iv) System Monitor:

It is a integrated component the monitor (a ML program) resides in ROM.

It helps to performing such functions are entering a program into memory, getting a program executed, madifying contents one to more memory locations.

ML program in convenient hexadecimal format, etc.

MICROPROCESSOR INSTRUCTIONS SET

Five groups of instructions

(i) Data Transfer Group.

These instructions help to move data between registers within the microprocessor, between a register and a memory location or b/w memory location.

(ii) Arithmetic Group.

In this group add, subtract, increment or decrement data in registers or in memory

Eg. an instructions to add the contents of two registers within the microprocessor.

(iii) Logical Group.

These are used to AND, OR, EXCLUSIVE-OR, compare, rotate or complement data in registers or in memory.

Eg. OR the contents of two registers within the microprocessor.

(iv) Branch Group.

This group include conditional and unconditional jump instructions.

A conditional instruction specifies that a certain operation be performed only if a certain condition has been met.

eg - (Jump to a particular instruction if the result of the last operation was zero).

* 4004, 4040, 8008 based on PMOS technology there are speed limitations.

* NMOS is the main technology today in use for low cost µps.

Eg. 8080, 8085, Z-80, 6800, 808, 6800, 8085, Z-8000, 68000).

* Intel 4004 First μp by Intel Corp in 1971

* Intel 4040 - 4 bit μp Ex: Rock well International's PPS4, NEC's μCOM4 and Toshiba's T3472.

* First 8-bit Microprocessor in 1973 by Intel Crop. This was intel 8008.
Improved versions 8080,
Today's the better known 8 bit mps are Intel's 8085, Motorola's M6800, NEC's μCOM85AF,

Then followed 12-bit & 16-bit μps.
Ex: 12-bit μps are Intel's IM6100 and Toshiba's T3190

16-bit μps Intel's 8086, Fairchild's 9440, Texas Instrument's TMS 9940 and TMS 9980 Zilog's Z8000. Motorola's M68000.

Bipolar μps usually made in bit-slice configuration
Ex: Intel's 3002 (2-bit slice, TTL),
Transitron's 1601 (4-bit slice, TTL)
Texas Instruments' SBP 0400 (4-bit slice, TTL).

* The CMOS tech. based μps eg (RCA's COSMAC) have limited application because of lower packing density and higher cost.

ARCHITECTURE OF INTEL 8086 MICROPROCESSOR

It is first announced in 1978, was the first 16 bit microprocessor introduced by Intel corporation. It has been steam g family components such as the 8088 µps, 8087 numeric coprocessor and 8089 I/O propessor.

8086 is manufactured using high performance metal oxide semi conductor (HMOS) technology. and its chip is equivalent to 29000 transistors.

It is enclosed in a 40 pin package as shown right side of figure

A0 through A15 are Address line
D0 through D15 are Data line. multiplexed.

The 8086 is a 16 bit µps. with 16 bit internal and external data paths.

It has ability to address upto 1MB. of memory via, a 20 bit wide address bus.

It can address upto 64K of byte wide input/output ports or 32K of word wide ports.

GND	1	GROUND	40 Vcc
AD14	2		39 AD15
AD13	3		38 AD16/S3
AD12	4		37 A17/S4
AD11	5		36 A18/S5
AD10	6		35 A19/S6
AD9	7		34 $\overline{BHE}$/S7
AD8	8		33 MN/$\overline{MX}$
AD7	9		32 $\overline{RD}$
AD6	10		31 $\overline{RQ}$/$\overline{GT0}$ (HOLD)
AD5	11		30 $\overline{RQ}$/$\overline{GT1}$ (HLDA)
AD4	12		29 $\overline{LOCK}$ ($\overline{WR}$)
AD3	13		28 $\overline{S2}$ (M/$\overline{IO}$)
AD2	14		27 $\overline{S1}$ (DT/$\overline{R}$)
AD1	15		26 $\overline{S0}$ ($\overline{DEN}$)
AD0	16		25 QS1 (ALE)
NMI	17		24 QS0 ($\overline{INTA}$)
INTR	18		23 $\overline{TEST}$
CLK	19		22 READY
GND	20		21 RESET

Pin Configuration of Intel 8086 microprocessor.

READY — This input is controlled to insert wait state into the timing of the microprocessor.

If the Ready pin is placed at a logic 0 level the microprocessor enters into wait states. If ready pin is placed at logic 1 level it has no effect on the operation of the microprocessor.

INTR — Interrupt request is used to request a hardware interrupt. If INTR is held high when IF = 1, 8086 enters an interrupts acknowledge cycle after current instruction has completed execution.

$\overline{TEST}$ — It is an input that is test by wait instruction. If $\overline{TEST}$ is a logic, WAIT instructions function as a NOP. If $\overline{TEST}$ is a logic 1, then WAIT instruction waits for $\overline{TEST}$ to become logic 0.

NMI — NON-maskable interrupt is similar to INTR except that the NMI interrupt does not check to see if the IF flag bit is a logic 1.

CLK — clock pin provides basic timing signal the microprocessor.

RESET - The input causes the microprocessor to reset itself if this pin is held high for a minimum of four clocking periods.

VCC — This power supply provides +5V to the microprocessor.

GND - ground connection is the return for the power supply

MN/$\overline{MX}$ — minimum/maximum mode pin selects either minimum mode or maximum mode operation for the microprocessor. If minimum mode is selected. MN/MX should connected to 5V.

$\overline{BHE}$/S7 → Bus high enable pin is used to enable the most significant data bus bits during read or write operation S7 is always at logic 1.

Minimum Mode Pins

M/$\overline{IO}$ → This pin selects memory or I/O. It gives memory address or an I/O port address.

$\overline{WR}$ → Write line indicates that 8086 is outputting data to memory or I/O device.

$\overline{INTA}$ → Interrupt acknowledge signal is a response to the INTR interrupt pin. $\overline{INTA}$ is normally used to gate the interrupt vector no. onto the data bus.

ALE → Address latch enable shows that 8086 address/data bus contains address information.

DT/$\overline{R}$ → Data transmit signal shows that the microprocessor data bus is transmitting (DT/$\overline{R}$ = 1) or receiving (DT/$\overline{R}$ = 0) data.

DEN → data bus enable activates external data bus buffers.

HOLD → Hold input requests a direct memory access (DMA).

HLDA – Hold acknowledge indicates 8086 microprocessor has entered hold state.

Queue status Bits.

QS1	QS0	Functions
0	0	queue is idle
0	1	first byte of opcode
1	0	queue is empty
1	1	subsequent byte of opera

maximum mode pins.

$\overline{S_2}$ $\overline{S_1}$ $\overline{S_0}$ → status bits indicate the functions of the current bus cycle.

$\overline{RQ}$ / $\overline{GT1}$ → request / grants pins request direct memory access and

$\overline{RQ}$ / $\overline{GT0}$ – These pins both bi directional and are used to request and grant a DMA operation.

QS_1 and QS_0 → Queue status bits show the status of the internal instruction queue. These pins are provided for access by numeric coprocessor (8087)

S_2	S_1	S_0	functions
0	0	0	Interrupt acknowledge
0	0	1	I/O read
0	1	0	Halt
1	0	0	Opcode fetch
1	0	1	memory read
1	1	0	memory write
1	1	1	passive.

MAXIMUM MODE AND MINIMUM MODE SYSTEMS

The 8086 µp can be work in two modes minimum system logic = 1 and maximum system logic 0.

Maximum mode signals ($MN/\overline{MX} = 1$)

Name	Function
HOLD	Hold request
HLDA	Hole acknowledge
$\overline{MX}$ / $\overline{WR}$	write control
M / $\overline{IO}$	Memory / IO control
DT / $\overline{R}$	Data transmit / receive.
$\overline{DEN}$	Data enable
ALE	Address latch enable
$\overline{INTA}$	interrupt acknowledge

Common signals

Name	function
AD-15 - AD 0	Address / data bus
A19/S6 - A16/S3	Address / status
$\overline{BHE}$ / S7	Bus High enable / status
MN/$\overline{MX}$	Minimum / maximum mode control.
$\overline{RD}$	Read control.
$\overline{TEST}$	wait on test control
READY	wait state control
RESET	system reset
NMI	Non-maskable interrupt

INTR	Interrupt request
CLK	system clock
VCC	5 + 5V
GND	Ground.

Maximum Mode Signals.

Name	function
$\overline{RQ}$ / $\overline{GT1,0}$	Request / Grant bus access control
$\overline{LOCK}$	Bus priority lock control
$\overline{S2} - \overline{S0}$	Bus cycle status
$QS_1 - QS_0$	Instruction Queue status.

INTERNAL REGISTERS OF 8086

The 8086 has four groups of external registers.

They are instruction pointer four data registers, four pointer and index registers.
four segment registers.

Instructions pointer is a 16-bit register.

(a) Data Registers :-

There are four general purpose data registers that are located within the EU of the 8086.

these registers are :

Accumulator (A), base register (B),
count register (C) and data register (D).

Each register can be accessed as either a 16 bit word or two 8 bits bytes.

(b) pointer and Index Registers.

They are used to store offset addresses of memory location relative to the segment registers.

There are two ~~stack~~ registers in this register (i) stack pointer :-

It permit easy access to locations in the stack segment of memory.

(ii) Base pointers is used to access data within the stack segment.

Index registers are used to hold the offset address for instruction that access data stored in the data segment of memory.

(SI) source INDEX Registers is used to store an offset address for a source operand. and the destination index Register (DI) is used for storage of an offset that identifies the location of a destination operand

(c) SEGment Registers:

The physical address of the 8086 is 20 bits wide, but its registers and memory location contain address which are 16 bits wide. This gives 1 M byte address space. This address space is segmented into 64 K bytes.

Register	Operations
AX	word multiply, divide
AL	Byte multiply, divide translate
AH	Byte multiply.
BX	Translate
CX	String operations, Loop.
CL	Variable shift and rotate
DX	word multiply, word Divide & Indirect I/O.

Register functions.

There are four segment registers in code segment (CS) register, data segment (DS) register, stack register (SS), and extra segment register (ES). They are loaded with 16 bit.

(SS) register is a identifier of the starting location of the current stack segment in memory.

(D.S) register holds the data, constants & work areas needed by the programe.

(ES) register is moved string one area to an other area.

(d) Flag register:

It is 16 bit register within the EU. Six flags represent status flags and three flags represent condition flag.

status flags :-

They indicates the conditions that are produced as a result of executing an arithmetic or logic instruction.

Details of the flags are as give below:

① carry flag (CF): This is set if there is a carry out or borrow in for the most significant bit of the result

② parity flag (PF): This is set if the result produced by the instruction has even parity.

③ Auxiliary carry flag (AF): This is set if there is carry out from the low nibble into the high nibble.

④ Zero flag (ZF): This is set if the result of an arithmetic or logical operation is zero.

⑤ sign Flag (SF): This set if result is a negative number.

⑥ overflow flag (OF): This is set if the signed result is out of range.

Condition Direction Flag -

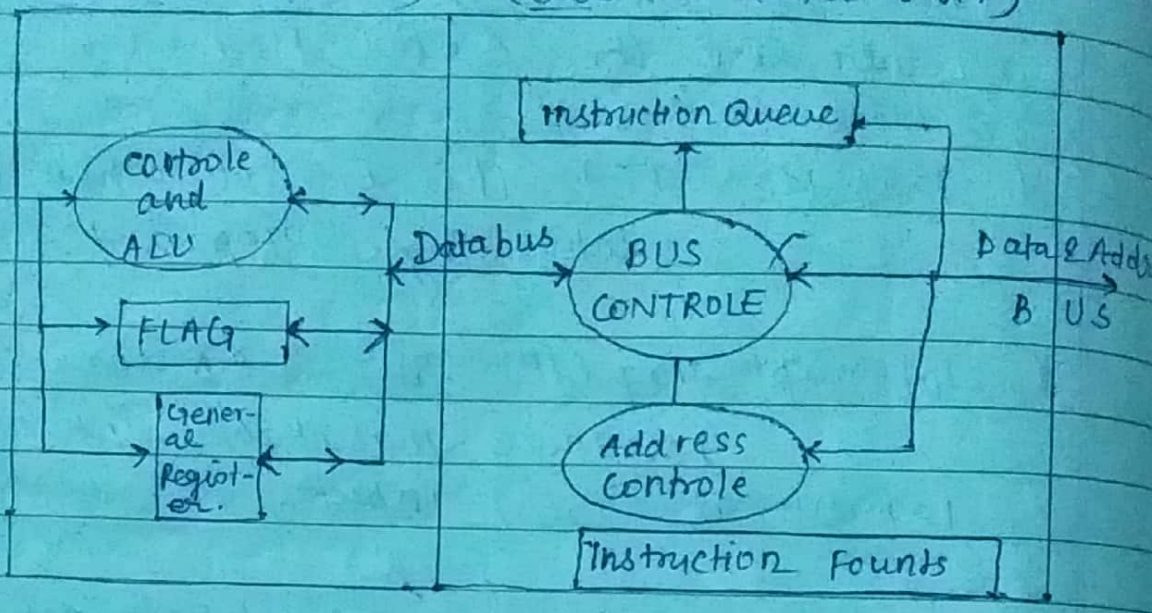
They are the control flag.

1. Trap flag (TF): This is set when 8086 goes into single step mode.

2. Interrupt flag (IF): This is set for 8086 to recognize masakable interrupt requests at INT input.

3. Direction flag (DF): This determines the direction in which string operations will occur. When DF is reset, the starting address of SI is to be incremented i.e., D=0 ; 8086 goes to autoincrement mode.

INTERNAL Architecture of 8086



Execution unit is where actual posseive data takes place:

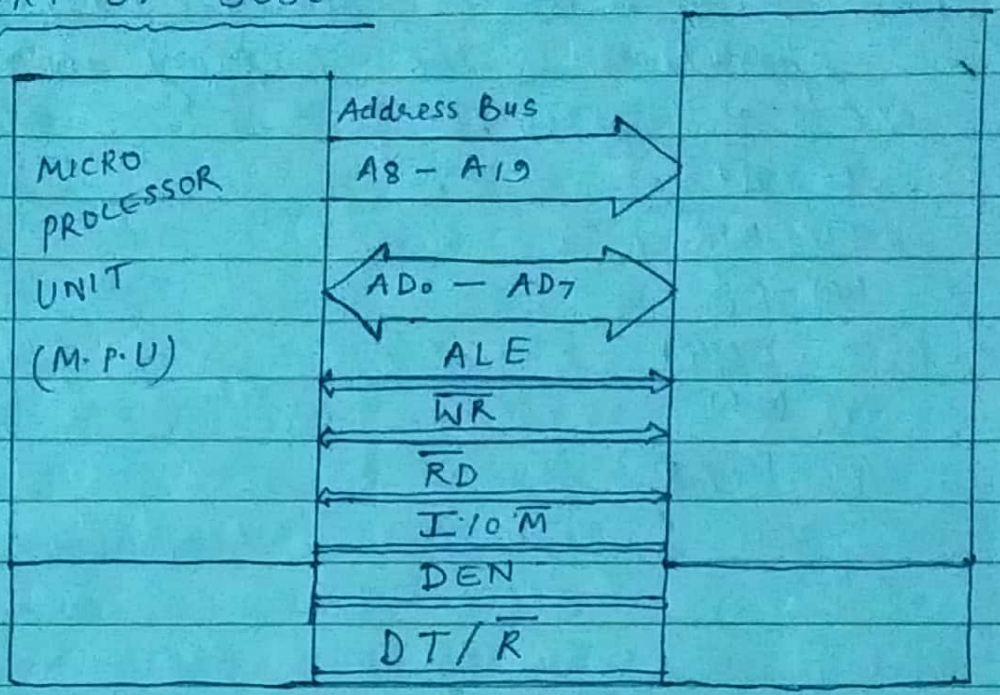
Arithmetic logic unit manipulate data and store intermediate results.

Bus Interface unit is made up of address generation and bus controle unit and instuction queue and the instruction pointer it has task of making sure the bus is to used to its in order to speedup operation.

Controle Unit of 8086 determine how and where an operation is to be performed In consists of

(i) power lines
(ii) Address Status line
(iii) Data line
(iv) data / address controle line
(v) interrupt controle line
(vi) Operation controle line

MEMORY OF 8086



In a micro computer System the memory location are used to store data and instruction that tell the microprocessor. When it is suppose to do in order for the microprocessor to used the information stored in memory the must fetch or read the information from memory the information is read into the Bus interface Unit of the microprocessor over the data Bus.

One's the data bits are process the result can be stored and return back into the memory for use at a letter type this is called writting to memory again the information follows back into the memory on the data Bus.

Data Bus must be bidirectional the operation of the memory control signals are as follows.

(i) ALE
(ii) $\overline{WR}$
(iii) $\overline{RD}$
(iv) IO/M
(v) DEN
(vi) DT/$\overline{R}$

K- MAP

ADDER : It is a sum of Binary digits.

(I) HALF Adder:
(II) FULL Adder:

(I) HALF Adder:

2 bit Adder is called half adder

TRUTH TABLE

A	B	carry	sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

K- MAP FOR SUM

$\therefore$ SOP for sum = $\bar{A}B + A\bar{B}$

A \ B	$\bar{B}$	B
$\bar{A}$	0_0	1_1
A	1_2	0_3

K- MAP FOR CARRY

$\therefore$ SOP for carry = AB

A \ B	$\bar{B}$	B
$\bar{A}$	0_0	0_1
A	0_2	1_3

BLOCK DIAGRAM

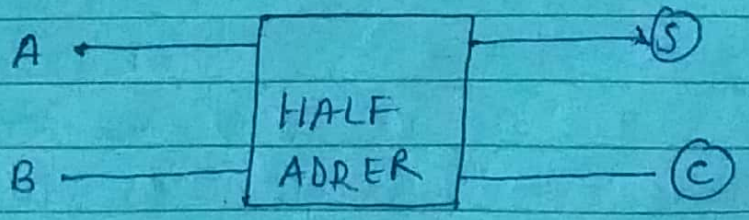
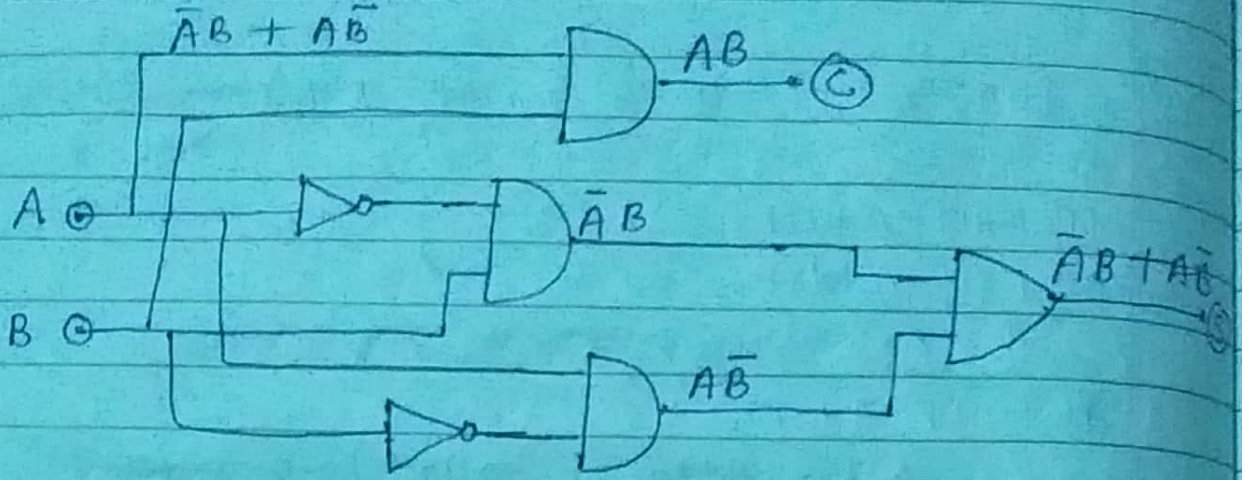


Fig. Block Diagram of Half Adder

CIRCUIT DIAGRAM



Circuit Diagram for Half Adder

circuit diagram for half adder using XOR Gate

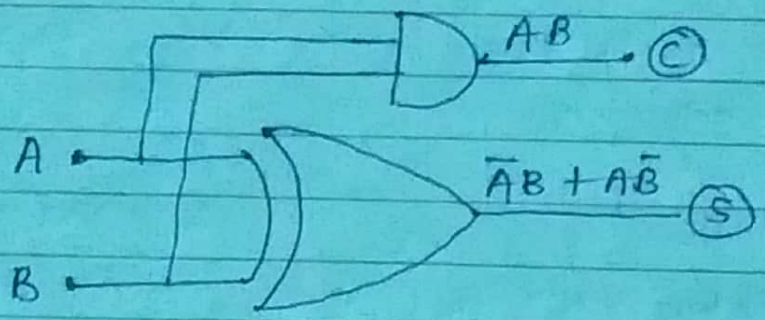


FIG: HALF ADDER

8085	8086
8 bit	16 bit
Add bus 16 bit	20 bit
Data Bus 8 bit	16 bit
A, B, C, D, E, H & L	

B	C (8bit)
D	E (8 bit)
H	L

AX	AH	AL (16 bit)
BX	BH	BL (16 bit)
CX	CH	CL (16 bit)
DX	DH	DL (16 bit)

FLAG REGISTERS	FLAG REGISTERS	
ZF	ZF	IF
CF	CF	TF
AF	AF	DF
PF	SF	
SF	PF	

Addressing Mode of 8086 :-

(1) Immediate :-

In this type of addressing mode immediate data is a part of a instruction and appear in the form of succesive Byte.

Ex: mov AX, 0005H

② Direct :-

In the direct addressing mode, 16 bit memory address is directly specified in the instruction

Ex: MOV AX 0005H

[ ] = Data resider in the memory location in the data segment where is effective address may be computed. using 0005H an a offered address and content of data segment.

10H * DS + 0005H {Effective Address}

③ REGISTERS :-

In the register addressing mode of the data is stored in a register and it is refered using a particular register.

Ex: MOV AX, BX

④ REGISTER INCLINED :-

The address of the memory location which contain data or operate is determind in a direct way.

in this mode the offered addressing of data is either BX, SI or DI

EX: MOV AX [BX]
Effective address: 10H * DS + BX

(5) INDEXED REGISTER:

In this addressing mode offered of the operate is stored in indexed register

EX: MOV AX [SI]
EA = 10H * DS + SI

(6) REGISTER RELATIVE:

In this A.M the data is avilabel ran effective address ferm by adding 8 bit and 16 bit displacement with the contain of BX, BP, SI & DI.

EX: MOV AX, 50H [BX]
EA = 10H * DS + 50H + BX

(7) BASED INDEXED:

Effective address of data is formed by adding content of based register with the content of indexed register (SI or BI)
EX: MOV AX, [BX], [SI]
EA: 10H * DS + BX + SI.

(8) RELATIVE BASED INDEXED :

The E·A is formed by adding 8 or 16 bit displacement

Ex: MOV AX, 50H[BX]. [SI]

EA = 10H * DS + BX + SI * 50H.

INSTRUCTION SETS OF 8086

(1) Data Transfer instruction
(2) LOGICAL instruction
(3) Arithemetic instruction
(4) Machine control instruction.

(1) Data Transfer instruction :-

(i) MOV & (ii) LOAD

MOV AX 0101H

MOV AX BX

MOV 0101H AX

(2) ARITHEMETIC INSTRUCTION :-

(1) Addition (3) Multiply.
(2) subtraction (4) Divide
(5) INCREMENT (6) DECREMENT.

Q1 Write a programe in 8086 to add two Numbers?

→
```
MOV AX, 01H
MOV BX, 02H
ADD AX, BX
MOV 0101H, AX
```

Q2. write a programme add 1+2+3+4 and show it answer?

→
```
MOV AX, 01H
MOV BX, 02H
ADD AX, BX
INC BX
ADD AX, BX
INC BX
ADD AX, BX
MOV ANS, AX
```

```
AX 01H
BX 02H
-------
   03H
   03H
-------
   06H
   04H
-------
   10H
```

Q3 Write a programme an A.L.P in 8086 to subtract two numbers.

MEM1 = 03H − 02H

Sol
```
MOV AX 03H
MOV BX 02H

SUB AX, BX

MOV MEM1, AX
```

Multiply

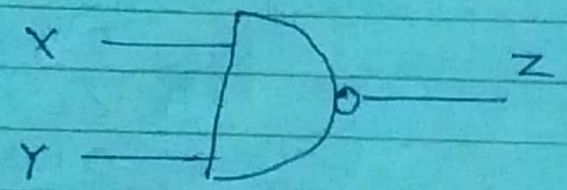
Q4. ANS = 03H * 02H

```
MOV AX , 03H
MOV BX , 02H
MUL BX
MOV ANS AX
```

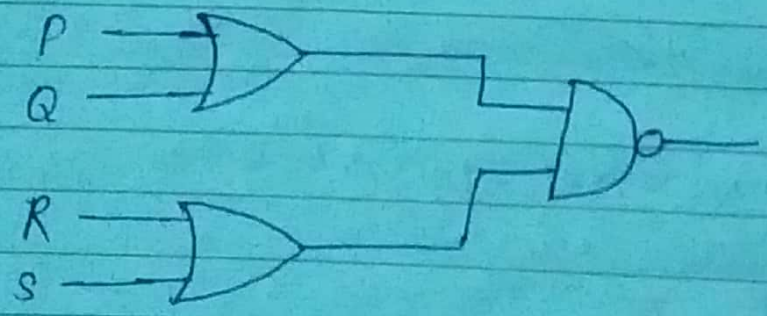
(3) LOGICAL INSTRUCTION:

```
MOV AX , 01H
MOV BX , 02H
AND AX , BX
```

```
0000 0001
0000 0010
0000 0000
```




```
MOV AX , X
MOV BX , Y
AND AX , BX
NOT AX
```



PRAKASH SIR 16-09-17

Page 57

ADDER

(I) Half Adder (II) Full Adder

HALF ADDER :-

2 Bit Adder is called half adder

$\bar{A} = 0$

$A = 1$

$\bar{B} = 0$

$B = 1$

Truth Table

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

K-MAP FOR SUM

A \ B	$\bar{B}$	B
$\bar{A}$	0 (0)	1 (1)
A	1 (2)	0 (3)

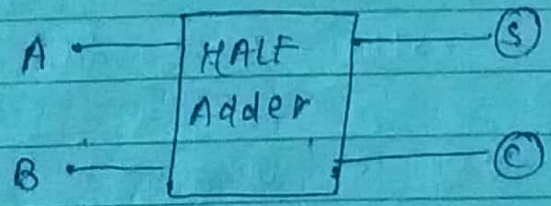
K-MAP FOR CARRY

A \ B	$\bar{B}$	B
$\bar{A}$	0 (0)	0 (1)
A	0 (2)	1 (3)

S.O.P FOR SUM = $\bar{A}B + A\bar{B}$

S.O.P for Carry = AB

BLOCK DIAGRAM FOR HALF ADDER :



CIRCUIT DIAGRAM.

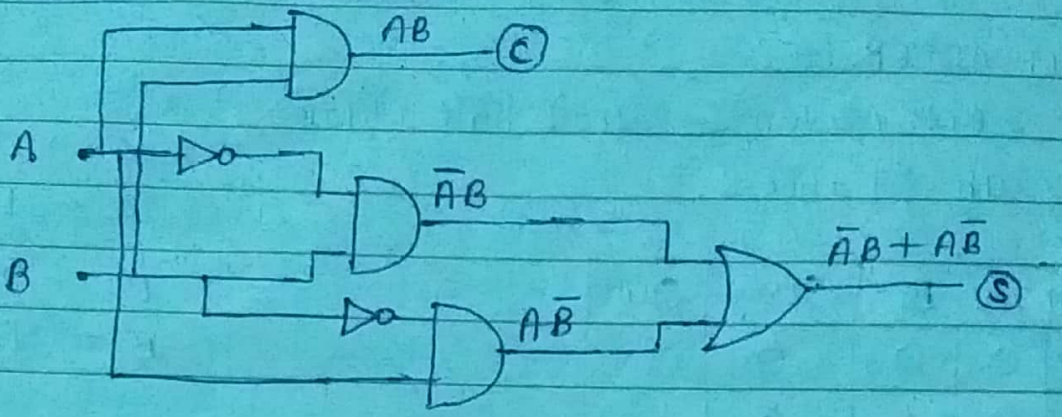
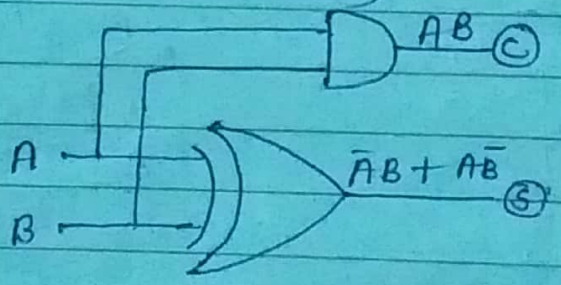


FIG: HALF ADDER CIRCUIT DIAGRAM

CIRCUIT DIAGRAM OF HALF ADDER OF EXCLUSIVE 'OR GATE (XOR Gate).



(2) FULL ADDER :

3 Bit Adder is called Full Adder.

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

K MAP FOR SUM

A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	$B\bar{C}$ 11	$B\bar{C}$ 10
$\bar{A}=0$	0	1	0	1
$A=1$	1	0	1	0

$S.O.P = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC$

K MAP FOR CARRY

A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}=0$	0	0	1	0
$A=1$	0	1	1	1

P_1, P_2, P_3

S.O.P of carry $= P_1 + P_2 + P_3$

$P_1 = \bar{A}BC + ABC$

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

$\boxed{P_1 = BC}$

$P_2 = A\bar{B}C + ABC$

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

$\boxed{P_2 = AC}$

$P_3 = ABC + AB\bar{C}$

$= AB(C + \bar{C})$

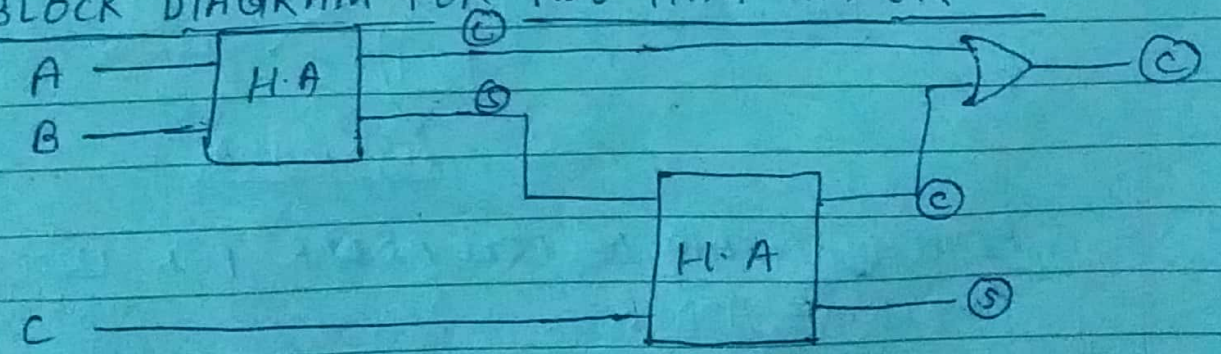
$\boxed{P_3 = AB}$

$\boxed{S.O.P_{(C)} = BC + AC + AB}$

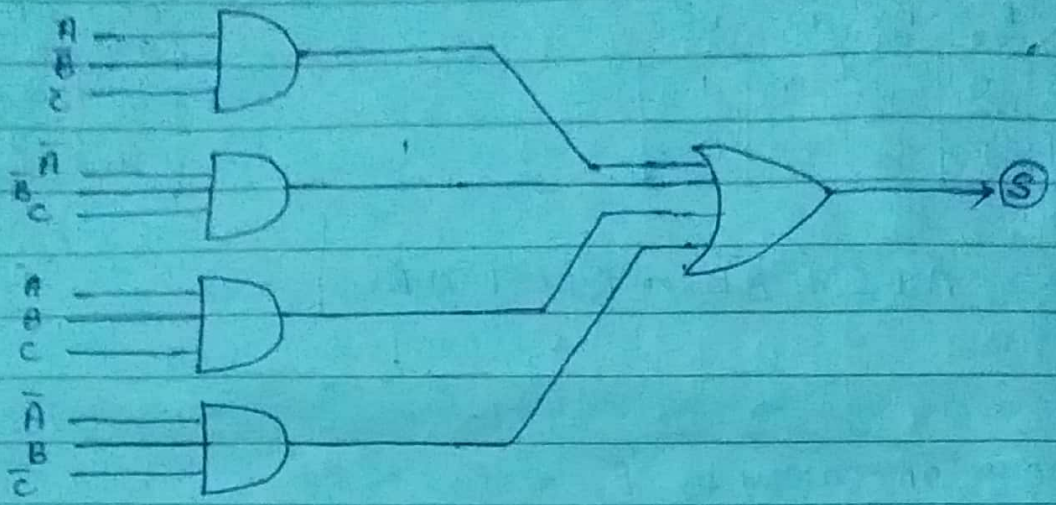
$\boxed{S.O.P_{(S)} = \bar{A}\bar{B}C + \bar{A}BC + AB\bar{C} + ABC}$

{ FULL ADDER USING TWO HALF ADDER

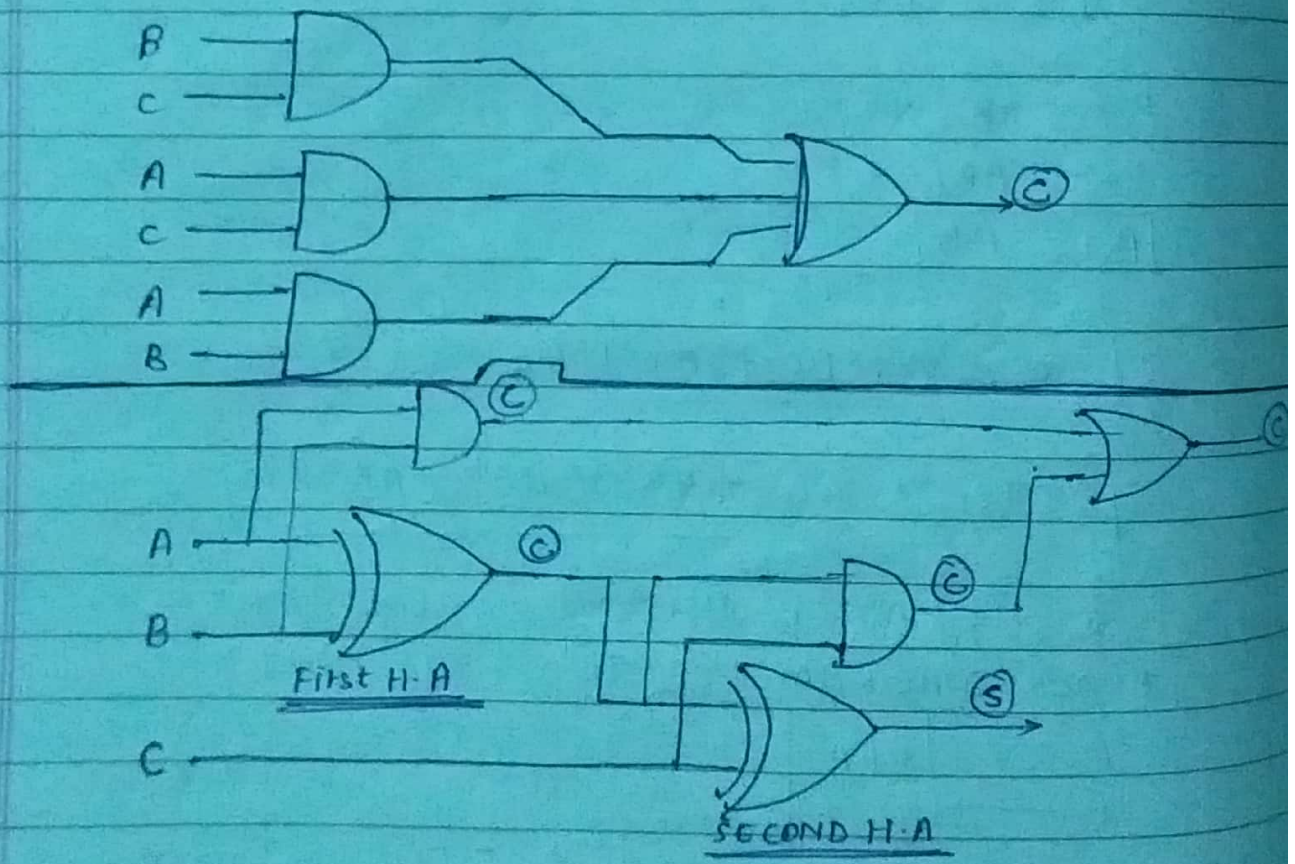
{ BLOCK DIAGRAM FOR TWO HALF ADDER



CIRCUIT DIAGRAM FOR SUM



CIRCUIT DIAGRAM FOR CARRY



CIRCUIT DIAGRAM OF FULL ADDER FOR USING TWO HALF ADDER :

HALF SUBTRACTOR

2 Bit Adder is called Half subtractor.

Truth Table for Half Subtractor

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

K-MAP FOR DIFFERENCE

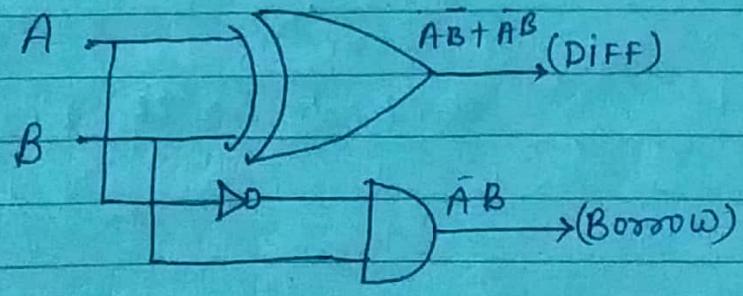
A \ B	$\bar{B}$	B
$\bar{A}$	0	1
A	1	0

S.O.P = $\bar{A}B + A\bar{B}$

K-MAP FOR BORROW

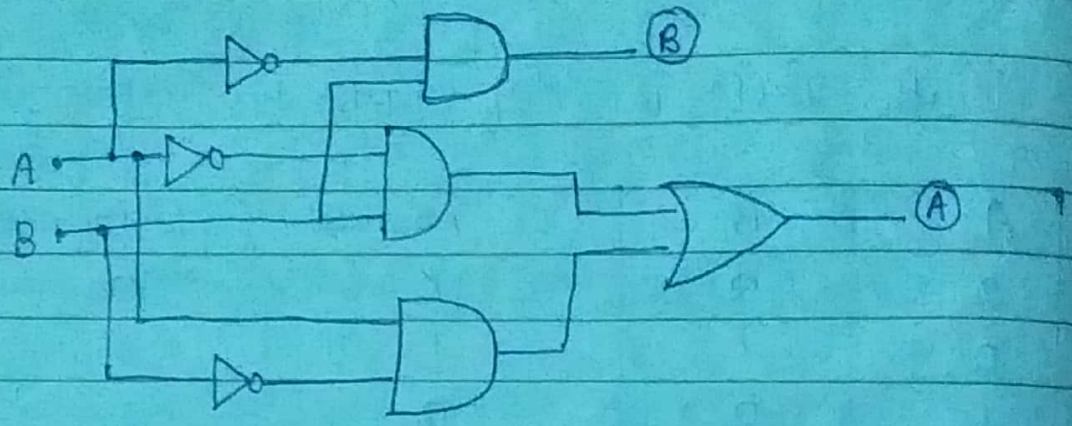
A \ B	$\bar{B}$	B
$\bar{A}$	0	1
A	1	0

S.O.P = $\bar{A}B$



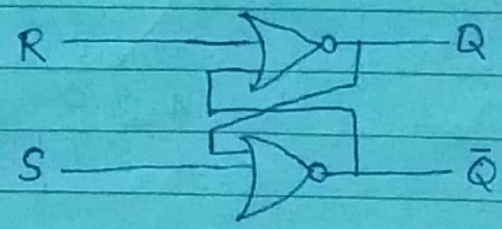
circuit diagram using XOR Gate

Circuit Diagram for Half Subtractor.



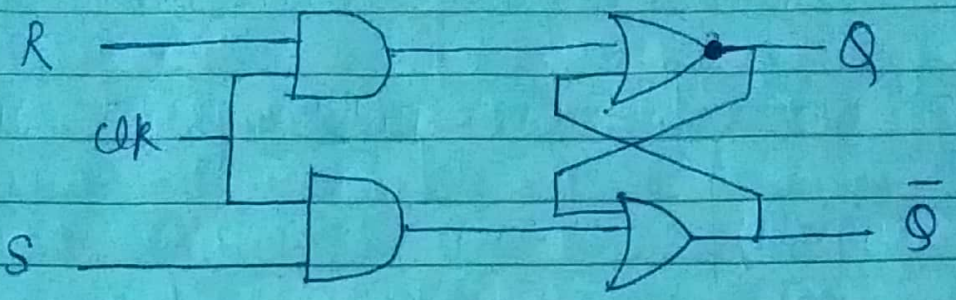
FLIP FLOP

SR LATCH



S	R	State
0	0	NO change
0	1	0 (Reset / clear)
1	0	1 (Set)
1	1	* (Undesirable) Race condition

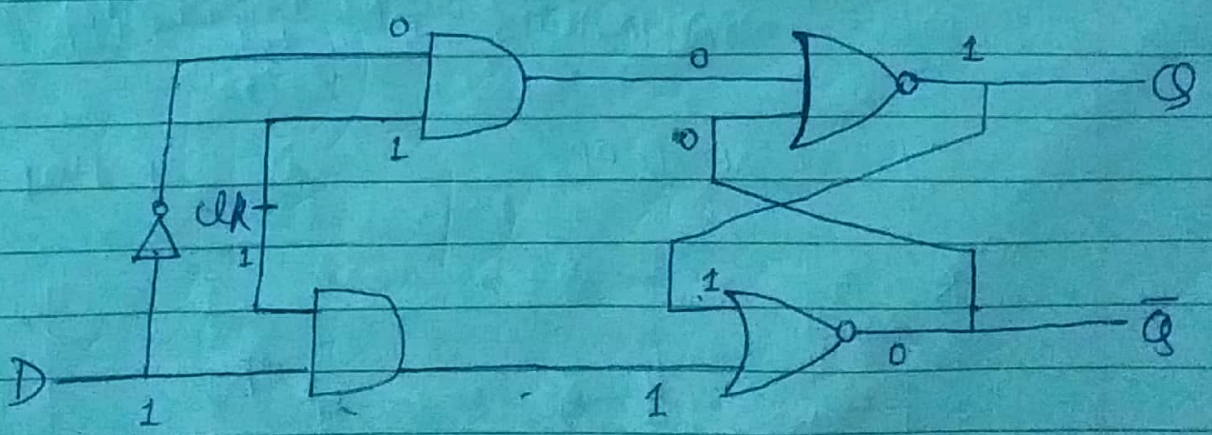
SR Flip Flop



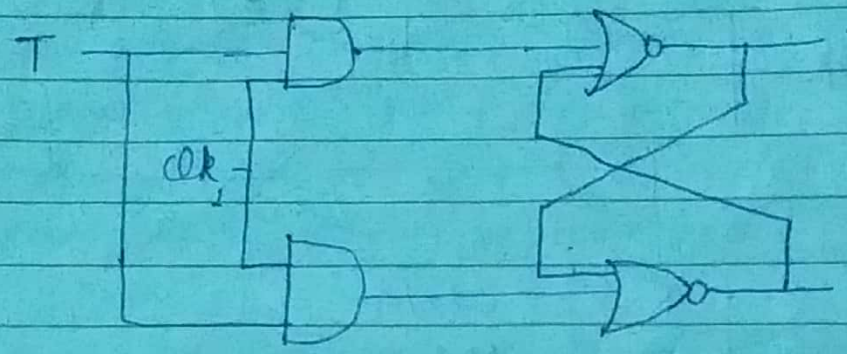
clk	S	R	state
↑	0	0	NC
↑	0	1	clear/Reset (0)
↑	1	0	set (1)
↑	1	1	* (Undesirable)

D - Flip Flop

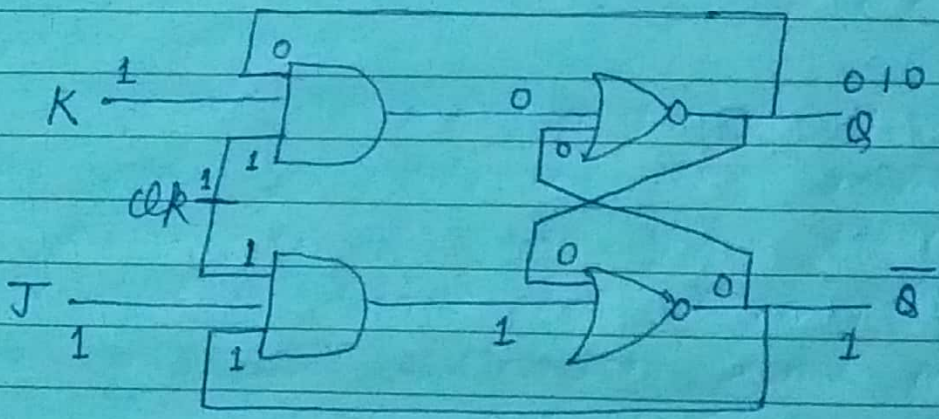
It is called data flip flop or delay flip flop
It store data which is given.



T Flip Flop (Toggle)

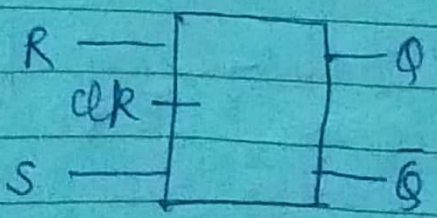


JK Flip Flop (Jump & Kick)

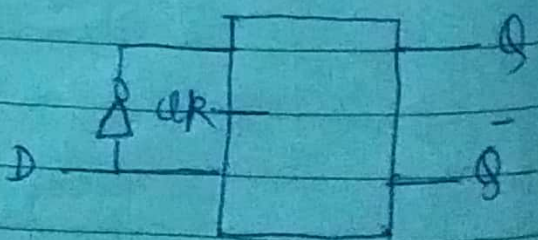


BLOCK DIAGRAMS OF All FLIP FLOPS

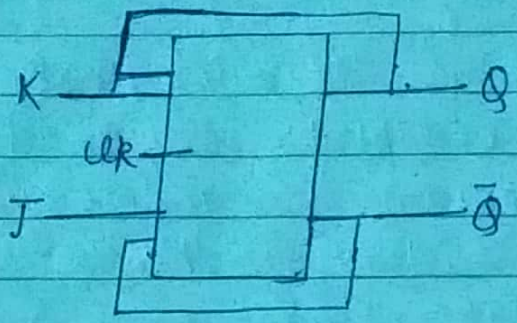
SR Flip Flop



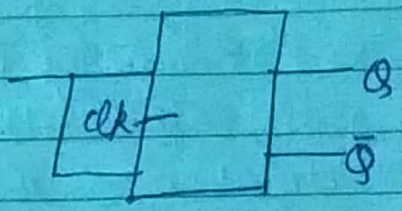
D Flip Flop



J. K Flip Flop



T Flip Flop



MASTER Flip Flop

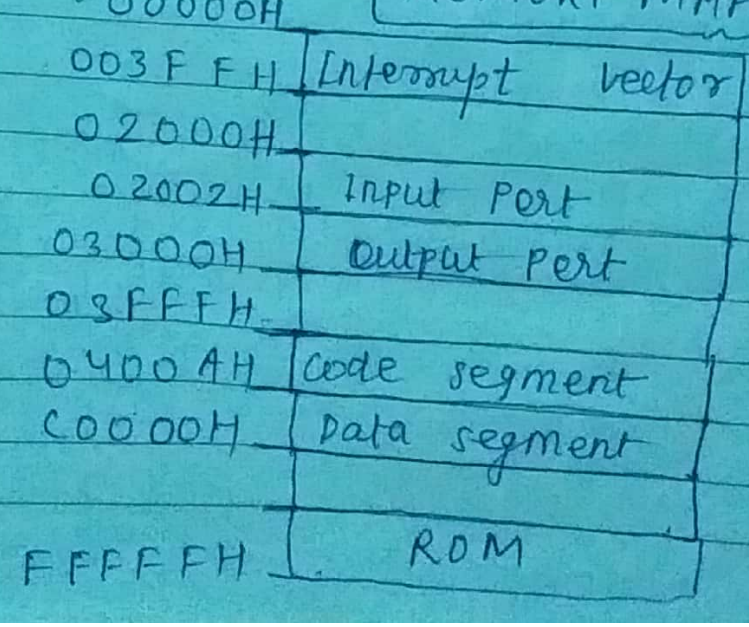
This Flip Flop is used to give output of a flip flop to another flip flop.

MEMORY MAPPING

A memory map is a diagram that represents all the occupied or read location in the memory. The memory map is used to identify the location and their purposes. The memory map shows its location or address usually as a hexadecimal number of ROM, RAM, port location, interrupt vectors, program segments. A method of treating a peripheral devices as a memory location is known as memory mapped input outputs.

The memory map is used as an aid in partitioning the available memory. It shows the location of all memory device and segments which is useful because some memory locations are dedicated for special purpose.

MEMORY MAPPING OF 8086



Date
Page

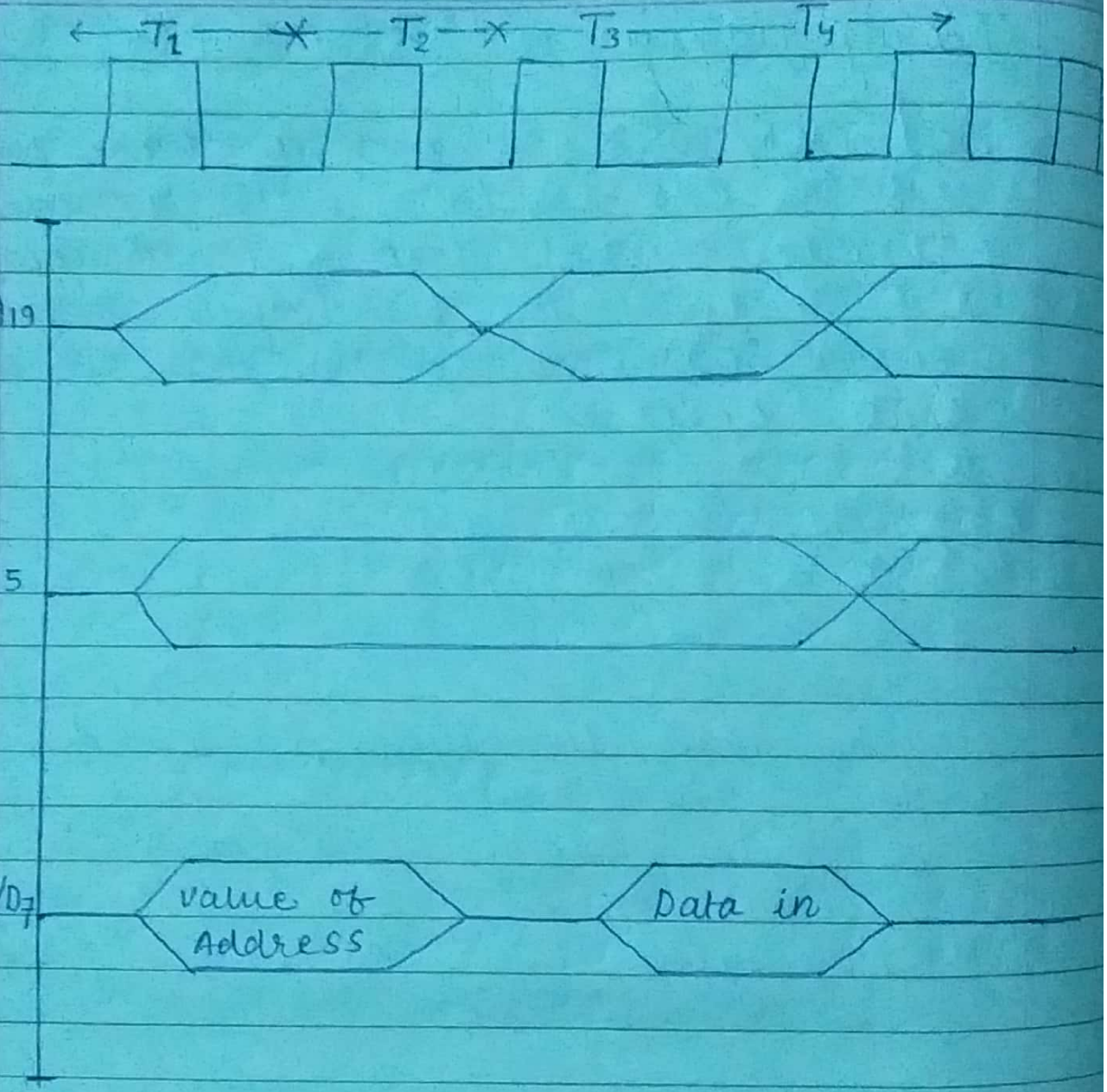
Timing Diagram of 8086

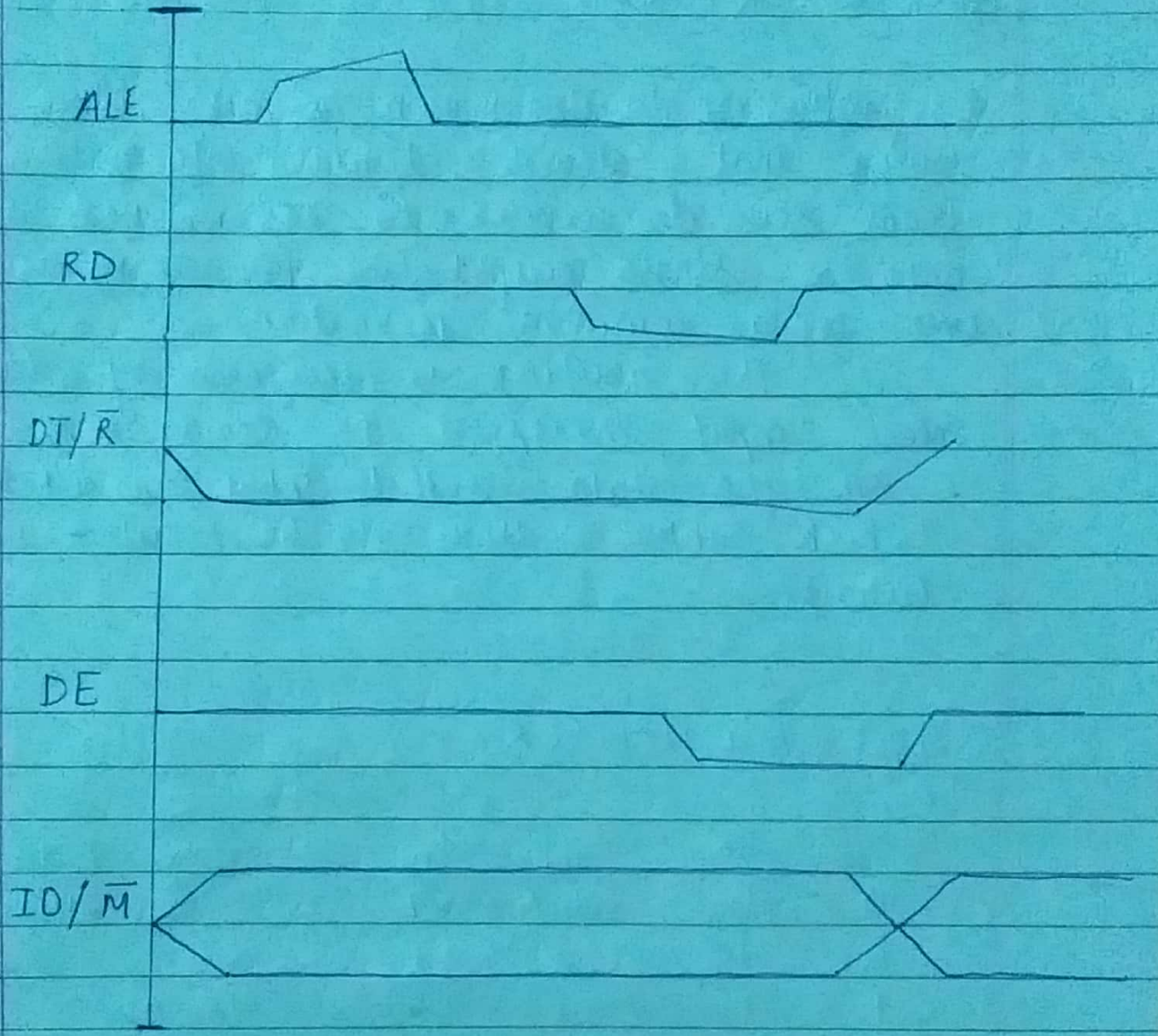
The 8086/8088 Operators in time periods called Bus cycle. Each bus cycle requires four clock cycle to complete. Therefore the bus cycle is completed energy 800 nanosecond. The two major bus cycle are :-

i) Read Bus cycle.
ii) Write Bus Cycle.

The read bus cycle is activated when the microprocessor is reading information from the memory or an input output device during the read bus cycle there are four clock cycle T_1, T_2, T_3, T_4.

T_1 : During the first clock cycle the address data bus is used to output the address of a memory or of input output processor also output during the first clock cycle are control signal. ALE, DT/$\overline{R}$, IO/$\overline{M}$.





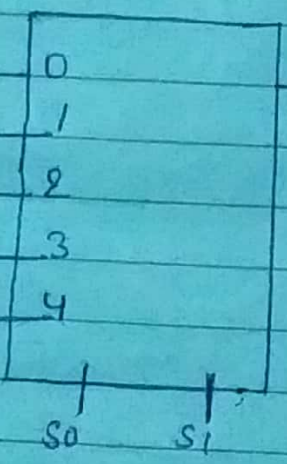
MULTIPLEXER (MUX)

A multiplexer is also known as MUX is a device that allows digital information from several sources to be routed onto a single output line for transmission over to a common destination

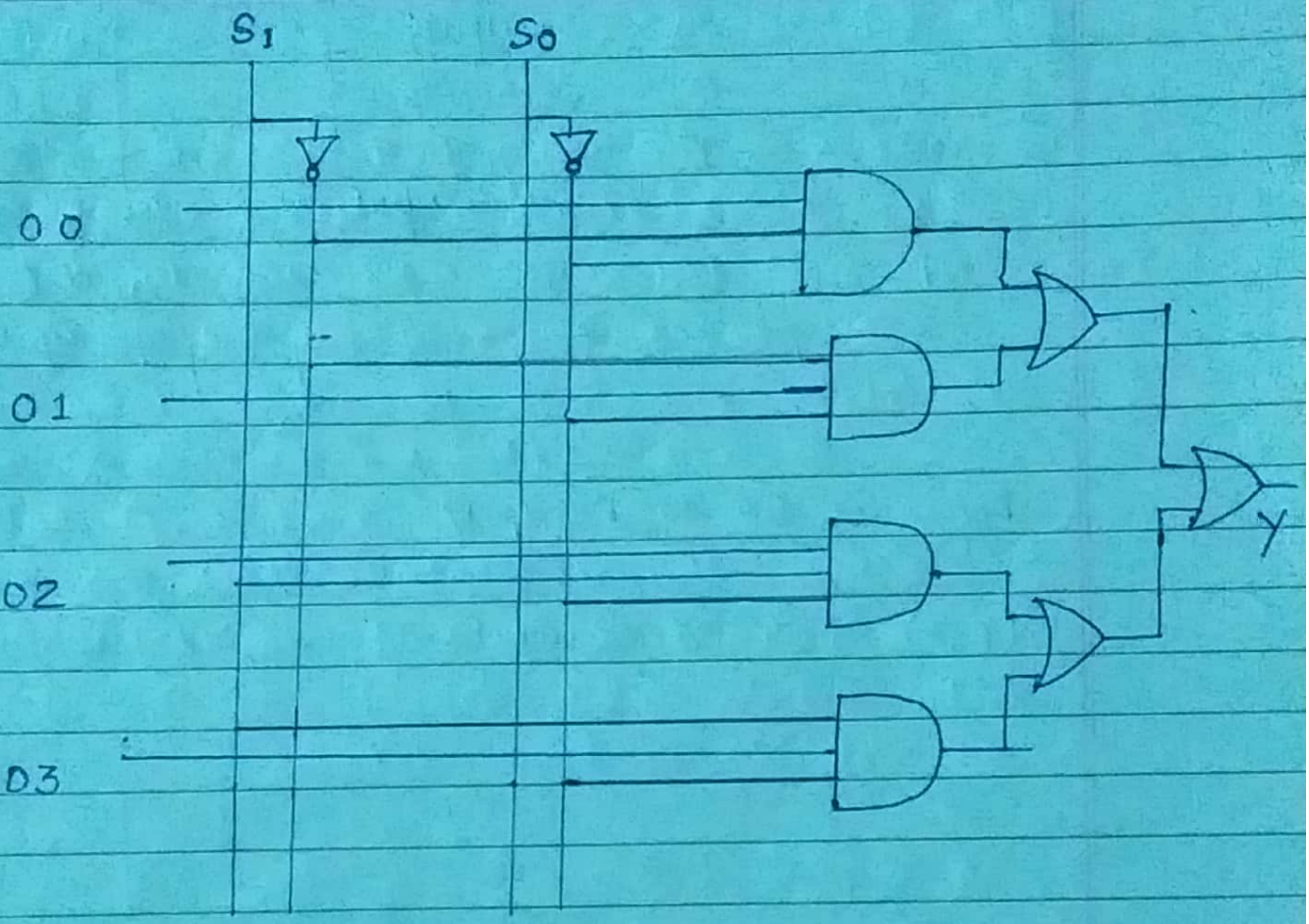
The routing is designed by data select input. Therefore the combination on the data select input decides which input line to be routed to the output.

4:1 MULTIPLEXER

S_1	S_0	input selection
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3



$$Y = \overline{S_1}\,\overline{S_0} D_0 + \overline{S_1} S_0 D_1 + S_1 \overline{S_0} D_2 + S_1 S_0 D_3$$

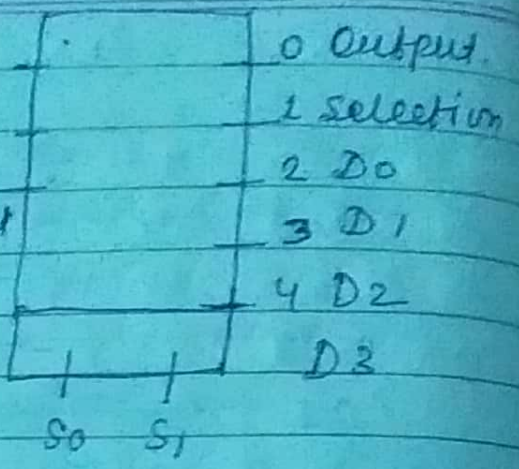


DE-MULTIPLEXER

Demultiplexer is also known as DEMUX it perform the reverse operation that of multiple function. It has one input line and many output line. The working of demultiplexer is to takes data from the input line and distributed it to output lines

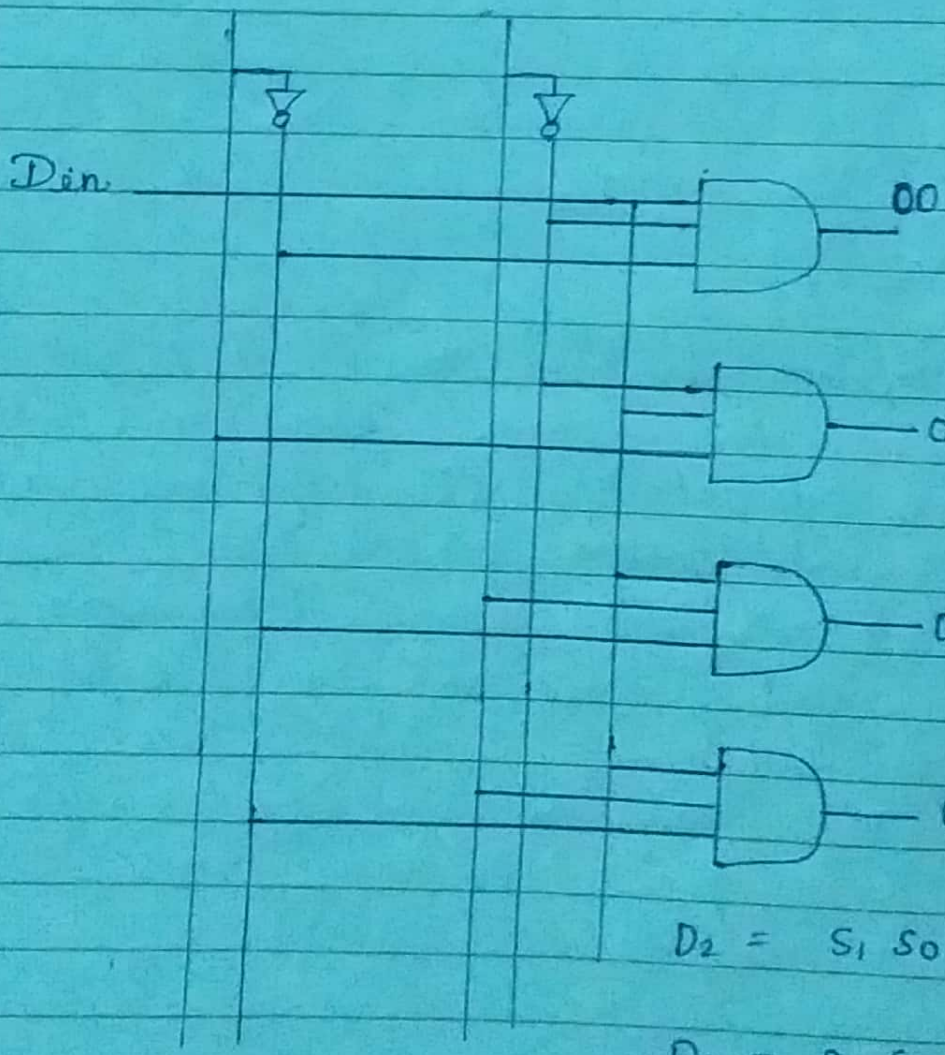
S_1	S_0	Output Selection
0	0	
0	1	
1	0	
1	1	

Data input



$D_0 = S_1 S_0 D_{input}$

$D_1 = \bar{S}_1 S_0 D_{input}$



$D_2 = S_1 \bar{S}_0 D_{in}$

$D_3 = S_1 S_0 D_{in}$

MICROPROCESSOR

Flag register of 8085.
Flag register of 8086.
Architecture of 8085.
Architecture of 8086.
Addressing mode of 8085.
Addressing mode of 8086.
Pinout diagram of 8085.
Pinout diagram of 8086.
Instruction set of 8086
memory structure of 8086.
memory mapping of 8086.

Date
Page

DECODER

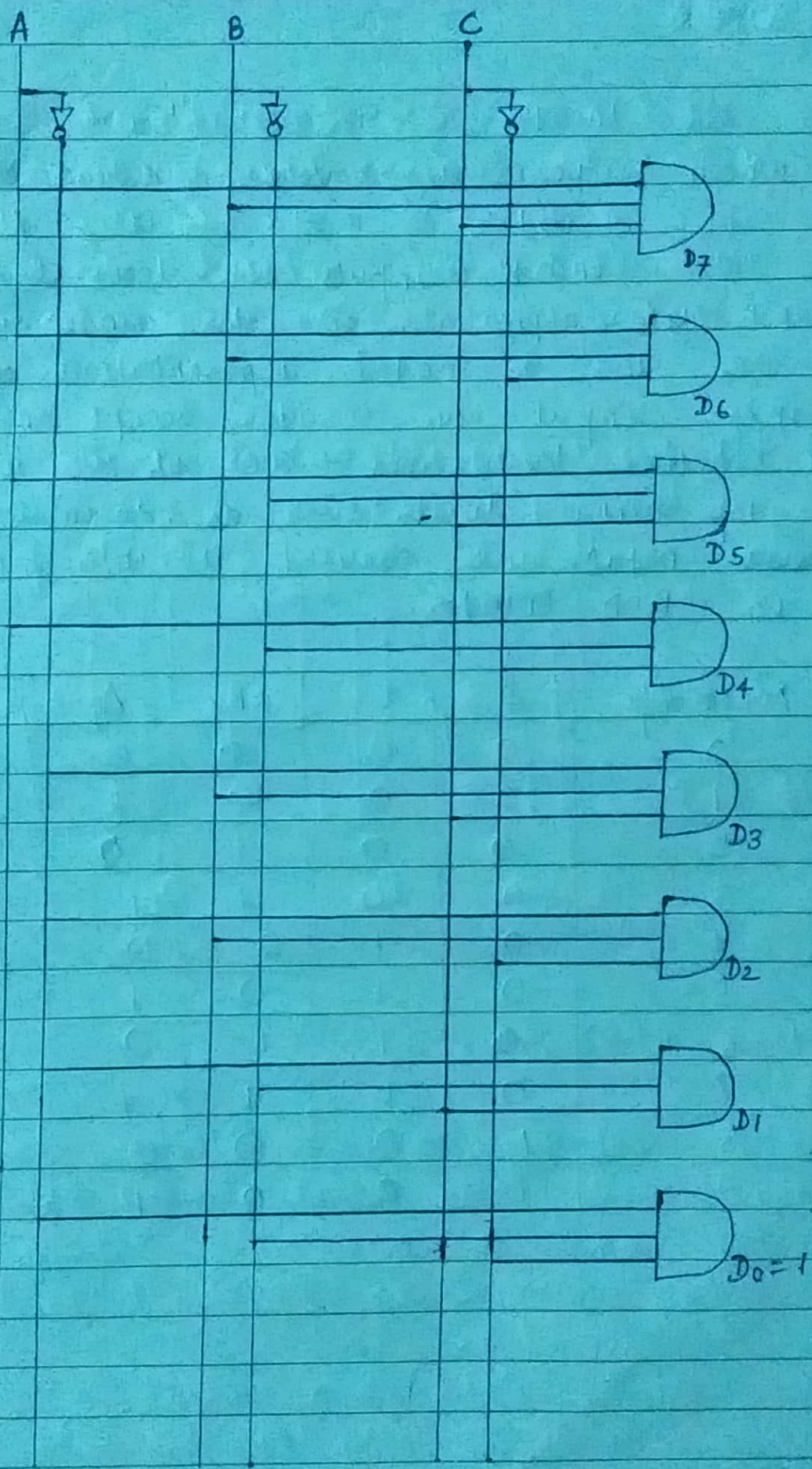
The Basic function of decoder is to detect the present of a Specified combination of bits on its input and to indicate that presence by a Specified output level.

A decoder has n input lines to handle n bits & from 1 to 2^n output line to indicate the presence of one or more n bits combination.

Ex: 2^3 = 8bits line

3 line to 8 line decoder

A



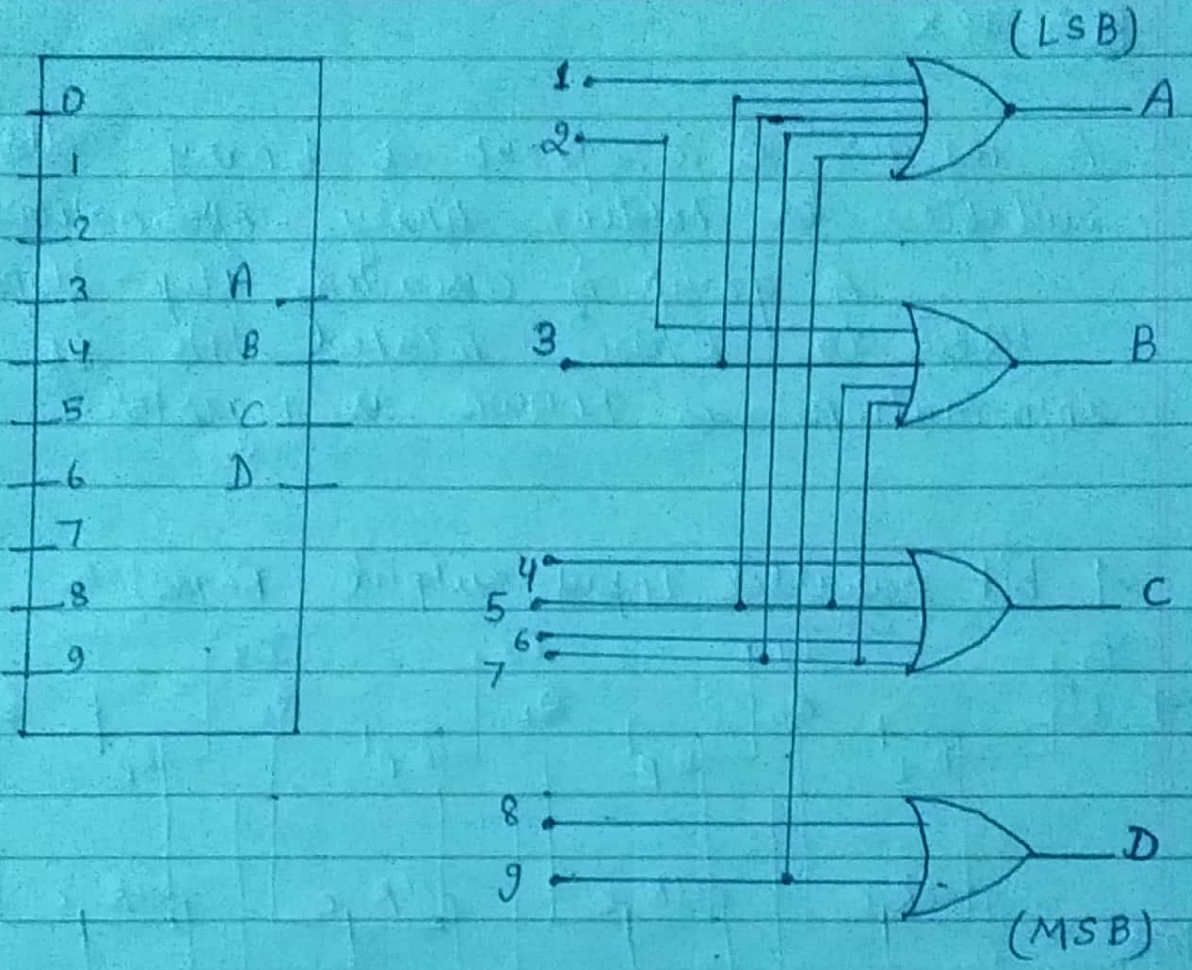
Date
Page

ENCODER

An Encoder is a combinational circuit which perform a reverse function that of decoder.

Encoder inputs are decimal digits and for alphabatic character and output are the coded representation of these input an encoder accept an active level and one of its input representing digits such as decimal or ~~oct~~ octal and convert it into binary or BCD encoder.

Decimal	D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1



P.K sir

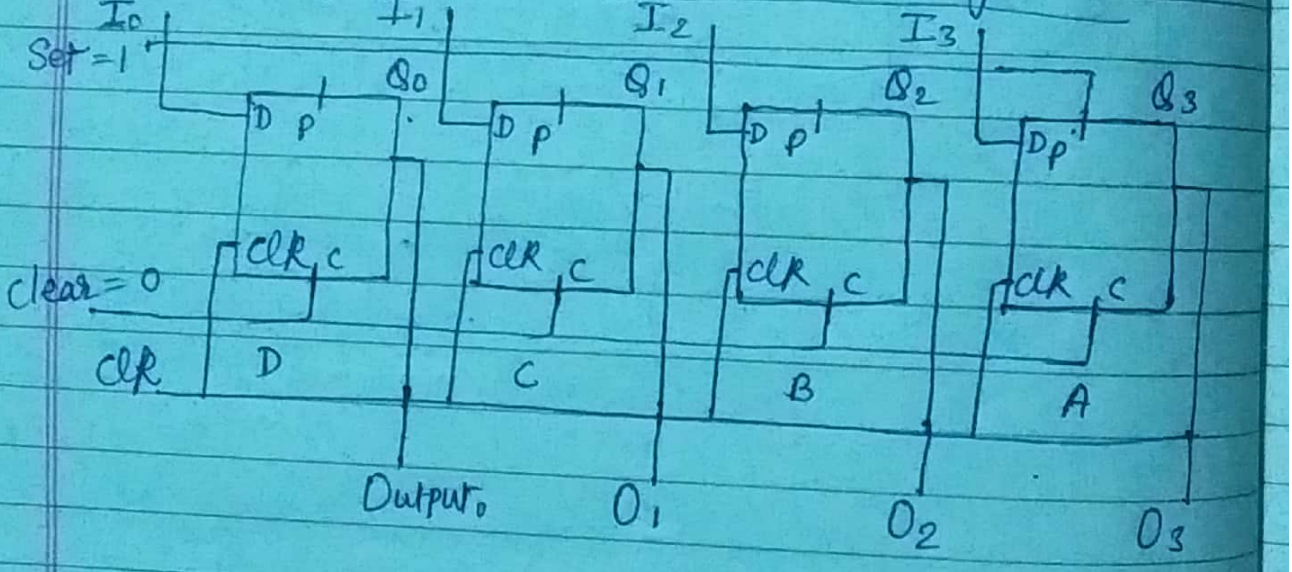
17/10/17

REGISTER

A register is a group of binary cells suitable for holding binary information

A group of cascaded flip-flops used to store related bits of information is known as register.

4 bit parallel Input output Register



SHIFT REGISTER

A shift register is a storage device that used to store binary data.

When a number of flip-flop are connected in series it is called a register.

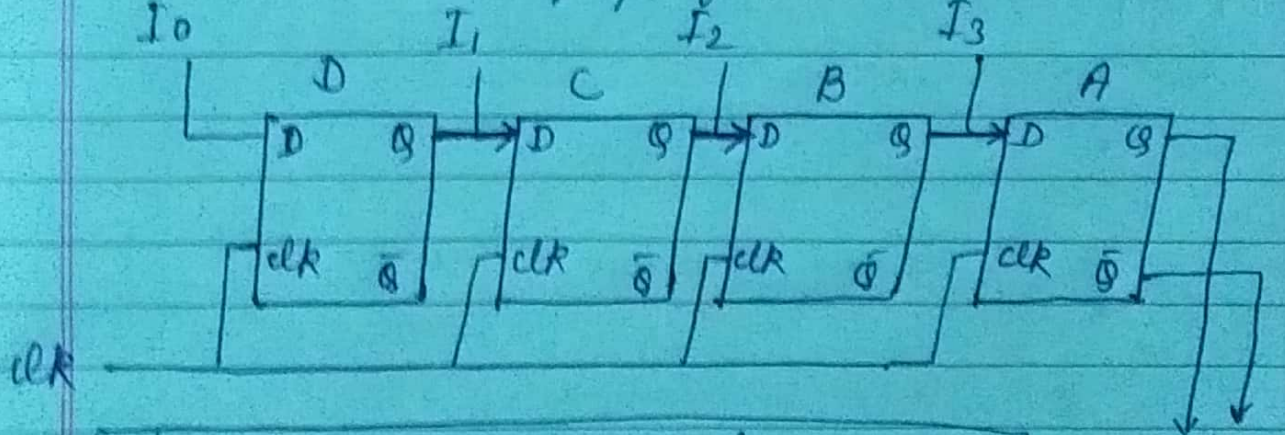
The data in a shift register is moved serially (one bit at a time)

Shift Register can be built using RS, JK or D flip flops various types of shift registers are available some of them are given as

(i) Left Shift Register
(ii) Right Shift Register
(iii) Round Shift Register
(iv) Bi-directional Shift Register.

(1) RIGHT SHIFT REGISTER

The right shift Register is built in D & JK flip flops.



SHIFT RIGHT OPERATION

Shift pluse	D	C	B	A
0	X	X	X	X
1	1	X	X	X
2	0	1	X	X
3	1	0	1	X
4	1	1	0	1