Soham Desai SE IT 11 202003021

Date
Page 1

MPL Assignment -1

Q1) Implement the following SOP expression using only NAND gates
$y = \Sigma m(0,1,5)$

Sol:

By using K-maping.

A\BC	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$	1 $_0$	1 $_1$	0 $_3$	0 $_2$
A	0 $_4$	1 $_5$	0 $_7$	0 $_6$

1st pair = $\bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C \Rightarrow \bar{A}\bar{B}$

2nd pair = $\bar{A}\bar{B}C + A\bar{B}C \Rightarrow \bar{B}C$

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

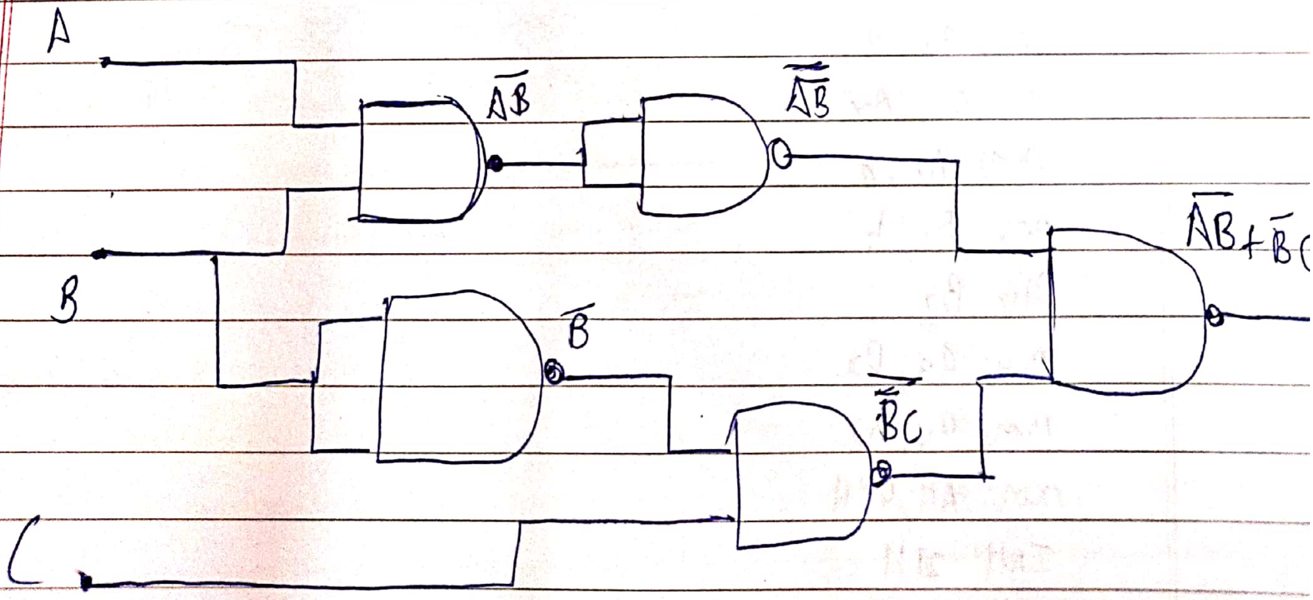
$y = \overline{\overline{\bar{A}\bar{B} + \bar{B}C}}$

Using Demorgan's theorem

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

10

The logic gate diagram,


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

Q.2) Write 8086 assembly language program to multiply and divide 4 digits number by 2-digit number. list and explain arithmetic instructions

Ans) :-

```
assume cs:code, ds:data
data segment
    a  dw  2048H
    b  dw  64H
    a1 dw  01 dup(?)
    a2 dw  01 dup (?)
    a3 dw  01 dup (?)
    a4 dw  01 dup (?)
data ends

code segment
start:
    mov Ax, data
    mov Ds, Ax
    mov Ax, a
    mov Bx, b
    Mul Bx
    mov a1, Dx
    mov a2, Ax
    mov Ax, a
    mov Bx, b
    Div Bx
    mov a3, Dx
    mov a4, Ax
    mov AH, 4CH
    INT 21H
    code ends
end start
```

There are two types of arithmetic instructions in the program:-

1) MUL (multiplication)

2) DIV (division)

1) MUL

MUL instruction in 8086 assembly Language is used for multiplying two 16-bit numbers. It deals with the multiplication of two unsigned numbers. It multiplies the contents of two general purpose registers and stores the result in a third general purpose register. The immediate operand is not allowed

Syntax: MUL multiplier

There are three types of multiplication depending on number of bits.

(i) Byte with Byte -

In this one operand resides in an Al register and the other one in source. Source can be a register or memory address

(ii) Word with word-

In this one operand is loaded in Ax register and the source should be a 16-bit register or memory address

(iii) Byte with word-

In this one operand is loaded in AH register and it is set to zero and Al is loaded with a byte operand. Result gets stored in Dx, Ax

10

2) DIV

DIV instruction in 8086 assembly language performs the division of two unsigned operands. The denominator resides in a source operand and it should not be immediate. But it can be register or a memory location.

Syntax: DIV dividend

There are four types of division depending on number of bits:-

1) Byte with byte:-
In this the numinator and denominator operands are bits. The numinator resides in Al and AH is set to zero

2) word with word:-
In this, the Ax register holds the number. After division, the quotient is stored in Ax register and remainder in Dx register

3) word with byte:-
In this, the numerator is 16-bit word stored in Ax which is divided with an 8-bit denominator. After division Al contains the quotient and AH contains reminder

4) Double word by word:-
In this case the numerator is a 32 bit number and denominator is 16 bit number, Ax and Dx stores numerator.

Q3) Write 8086 assembly language program to find the smallest/ largest number from a given set of numbers using LOOP condition. List and explain control instructions

Ans): I) For the smallest number:-

```
assume cs:code, ds:data
data segment
no db 67H, 55H, 12H, 21H, 72H, 89H, 93H
a db 01 dup(?)
data ends

code Segment
Start: mov Ax, data
       mov Ds, Ax
       mov Cx, 06H
       Lea Si, no
       mov Ah, [Si]
       INC Si
up: mov Al, [Si]
     CMP Ah, Al
     JC down
     mov Ah, Al
down: INC Si
       Loop up
       mov a, Ah
       mov AH, 4CH
       INT 21H
code ends
end start
```

2) For the largest number

Ans) :-

```
assume CS:code, ds:data
 data segment
 no db .91H, 57H, 11H, 05H, 90H, 85H, 71H
  b  db  01H dup (7)
data ends

Code segment
Start: mov Ax, data
          mov DS, Ax
          mov Cx, 06H
          Lea Si, no
          mov Ah, [Si]
          JNC Si
up: .mov Al, [Si]
        CMP Ah, Al
        JNC .down
        mov Ah, Al
down: INC Si
        Loop up
        mov  b, Ah
        mov  AH, 4CH
        INT 21H
Code ends
end start
```

The control instructions in the program where :

1) JC instruction:
JC stands for 'Jump if Carry'. It checks whether the carry flag is set or not. If yes then jump takes place, i.e if CF=1, jump

Syntax: JC me

2) JNC instruction:
JNC stands for 'Jump No Carry'. It checks whether the carry flag is set or not. If it is not set then jump takes place, i.e if CF=0, jump

Syntax: JNC up

3 CMP instruction:
The CMP instruction is used to perform comparison. Its identical to the sub instruction except it does not affect operands. It impacts the zero Flag and Carry Flag

Syntax: CMP destination, source

10