

Date: 13-01-2022 Exp.1 16 BIT arithmetic operations



**Aim:** To compute an assembly language code for addition, subtraction, multiplication and division for 16 bit numbers.

**Tool Used:**

Assembler - MASM 611

**Algorithm:**

First load the data into AX.

Load the data into BX.

ADD AX and BX.

The result of addition is stored in AX is moved to sumres.

Initialise the value of AX again

SUB AX and BX

The result of addition is stored in AX is moved to diffres1.

Erase the data in AX

Initialise the value of AX again

MUL BX

Store the value and initialise AX again

DIV BX and store the values of quotient in divquo1 and remainder in divrem1

Code ends

In DATA SEGMENT

Offset origin set at 1200

Values are stored as sumres1 as dw, diffres1 as dw ,mulres1 as dd , divquo1 as db ,divrem1 as db

**Program:**

code segment

assume cs:code, ds:data

start:

mov ax,data

mov ds,ax

xor ax,ax ; also use mov ax,0000h

mov ax, 2710h;input higher number

mov bx, 2002h; input lower number

add ax,bx

mov sumres1,ax

mov ax, 2710h

sub ax,bx

mov diffres1, ax

xor ax,ax

mov ax,2710h

mul bx ;16-bit multiplicand in AL & multiplier in any GP register

mov word ptr mulres1, ax

mov word ptr mulres1+2, dx

mov ax,2710h; ax-->2710h

div bx

mov divquo1, al;quotient of 16-bit division is stored in AL

```
    mov divrem1, ah;reminder of 16-bit division is stored in ah  
    hlt  
code ends
```

data segment

```
org 1200h ;defines the beginning offset for data segment
```

```
sumres1 dw ? ;used to store the result of addition
```

```
diffres1 dw ?
```

```
mulres1 dd ?
```

```
divquo1 db ?
```

```
divrem1 db ?
```

data ends

end start

### **Sample Input:**

16 bit number 1- 2710H

16 bit number 2- 2002H

### **Sample Output:**

Sum=4712

Difference=70E

Multiplication=4E24E20

Division=1 remainder-70E

**Manual Verification:**

Hex value:

$$2710 + 2002 = 4712$$

**Result**

Hex value:

$$2710 - 2002 = 70E$$

Hex value:

$$2710 \times 2002 = 4E24E20$$

Hex value:

$$2710 \div 2002 = 1 \text{ Remainder : } 70E$$

**Register/ Memory Contents for I/O:**

```

-u
0764:0021 A30412      MOV     [1204],AX
0764:0024 89160612     MOV     [1206],DX
0764:0028 B81027      MOV     AX,2710
0764:002B F7F3       DIV     BX
0764:002D A20812     MOV     [1208],AL
0764:0030 88260912     MOV     [1209],AH
0764:0034 F4         HLT
0764:0035 0000      ADD     [BX+SI],AL
0764:0037 0000      ADD     [BX+SI],AL
0764:0039 0000      ADD     [BX+SI],AL
0764:003B 0000      ADD     [BX+SI],AL
0764:003D 0000      ADD     [BX+SI],AL
0764:003F 0000      ADD     [BX+SI],AL

```

**Snapshot of the Output:**

```

-g 0034
AX=270E BX=2002 CX=124A DX=18F4 SP=0000 BP=0000 SI=0000 DI=0000
DS=0768 ES=0754 SS=0763 CS=0764 IP=0034  OV UP EI PL NZ NA PE CY
0764:0034 F4         HLT
-d 0768:1200 1207
0768:1200 12 47 0E 07 20 4E E2 04      .G.. N..

```

**Result:** Hence we have formulated the assembly code for operations on 16 bit numbers.