M.I.T. LAB Assignment – 10

U19CS012

1. Program to add two numbers.
(16 bit addition, 32 bit addition, 32 bit addition using DD directives)

TASM Code [16 Bit Addition]:

```
model small ; code & Data Segment fits in 64 KB
.8086

.data   ; DATA SEGMENT
; Intialise the First Number [QW - (Define Word)]
num1 DW 1234H
; Intialise the Second Number
num2 DW 5140H
; Variable ('ans') to Store the Answer
ans DW ?
; Declare carry as an 8-bit variable with a value 0
carry DB 00H

.code   ; CODE SEGMENT
MOV AX, @DATA
MOV DS, AX

; Move the First Number (num1 data-word) in 'ax' reg
    MOV AX, num1
; Move the Second Number (num2 data-word) in 'bx' reg
    MOV BX, num2
; 16 Bit Addition
    ADD AX, BX
    JNC skip
    MOV carry, 01H
; Store the Result in 'ans' data-word
skip: MOV ans, AX

; HLT
; 4CH is the parameter for the terminate function
MOV AH, 4CH
; If AL is 00h then the program terminated without an error.
MOV AL, 00H
INT 21h
END
```

Input:

Number1: (1234)H
Number2: (5140)H
Answer: (6374)H

Output:

```
-g

Program terminated normally
-d 076B:0000
076B:0000  C6 06 14 00 01 A3 12 00-B4 4C B0 00 CD 21 34 12
076B:0010  40 51 74 63 00 10 00 02-00 82 0E 00 00 00 00 83
076B:0020  0E FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0030  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0040  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0050  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0060  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0070  FF FF FF 26 00 FF FF FF-FF FF FF FF FF FF FF FF
```

[Due to **Little Endian** Rule: Higher Address -> Higher Byte & Lower Address Lower Byte]

TASM Code [32 Bit Addition]:

```
model small
.8086

.data   ; DATA SEGMENT
; First 32 Bit Number
num1high dw 1254H
num1low dw 25A0H
; Second 32 Bit Number
num2high dw 5835H
num2low dw 3627H
; 'ans' to Store Result of 32 Bit Addition
anshigh dw ?
anslow dw ?
; Carry
carry db 00H
```

```
.code    ; CODE SEGMENT
mov ax,@data
mov ds,ax

; Add Lower 16 Bit Part
mov ax,num1low
mov bx,num2low
add ax,bx
; Store Answer in 'anslow'
mov anslow,ax
; Add Higher 16 Bit Part
mov ax,num1high
mov bx,num2high
; Add with Carry [Generated from Addn of Lower Bit]
adc ax,bx
; Store Answer in 'anshigh'
mov anshigh,ax

; If Carry was Generated
jnc skip
mov carry, 01H
skip: mov anshigh,ax

;HLT
mov ax,4C00h
int 21h
end
```

Input:

Number1: (1254 25A0)H
Number2: (5835 3627)H
Answer: (6A89 5BC7)H

Output:

```
-g

Program terminated normally
-d 076C:0000
076C:0000  06 18 00 01 A3 14 00 B8-00 4C CD 21 54 12 A0 25
076C:0010  35 58 27 36 89 6A C7 5B-00 FF FF FF FF FF FF FF
076C:0020  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076C:0030  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076C:0040  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076C:0050  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076C:0060  FF FF FF 26 00 FF FF FF-FF FF FF FF FF FF FF FF
076C:0070  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
```

TASM Code [32 Bit Addition using DD Derivatives]:

```
model small
.8086

.data   ; DATA SEGMENT
; Double Word [DD]
num1 dd 123403A0H
num2 dd 0ABC0FFFH
; Answer & Carry
ans dd ?
carry db ?

.code   ; CODE SEGMENT
mov ax,@data
mov ds,ax
mov dl, 00H

; Intialize the Word Pointers
mov ax,word ptr num1
mov bx,word ptr num2
add ax,bx

mov word ptr ans,ax
mov ax,word ptr num1+2
mov bx,word ptr num2+2

; Add the Carry of Previous Lower 16 Bit Part
adc ax,bx
mov word ptr ans+2,ax

jnc skip
inc dl  ; Increment the Carry
skip: mov carry, dl

; HLT
mov ax,4C00h
int 21h
end
```

Input:

Number1: (1234 03A0)H
Number2: (OABC 0FFF)H
Answer: (1CF0 139F)H

Output:

```
-g

Program terminated normally
-d 076C:0000
076C:0000  02 FE C2 88 16 18 00 B8-00 4C CD 21 A0 03 34 12
076C:0010  FF 0F BC 0A 9F 13 F0 1C-00 FF FF FF FF FF FF FF
076C:0020  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076C:0030  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076C:0040  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076C:0050  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076C:0060  FF FF FF 26 00 FF FF FF-FF FF FF FF FF FF FF FF
076C:0070  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
```

2. Program to subtract two numbers. (16 bit subtraction, 32 bit subtraction)

TASM Code [16 Bit Subtraction]:

```
model small
.8086

.data   ; DATA SEGMENT
num1 dw 9876H
num2 dw 1234H
ans dw ?
borrow db 00H

.code   ; CODE SEGMENT
mov ax,@data
mov ds,ax

mov ax,num1
mov bx,num2

sub ax,bx

jnc skip
mov borrow, 01H
skip: mov ans,ax

; HLT
mov ax,4C00H
int 21h
end
```

Input:

Number1: (9876)H
Number2: (1234)H
Answer: (8642)H

Output:

```
-g

Program terminated normally
-d 076B:0000
076B:0000  C6 06 14 00 01 A3 12 00-B8 00 4C CD 21 00 76 98
076B:0010  34 12 42 86 00 10 00 02-00 82 0E 00 00 00 00 83
076B:0020  0E FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0030  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0040  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0050  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0060  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0070  FF FF FF 26 00 FF FF FF-FF FF FF FF FF FF FF FF
```

TASM Code [32 Bit Subtraction]:

```
model small
.8086

.data   ; DATA SEGMENT
; Double Word [DD]
num1 dd 97DE207AH
num2 dd 46AC1313H
ans dd ?
borrow db ?

.code   ; CODE SEGMENT
mov ax,@data
mov ds,ax

mov dl,00h  ; Borrow [Intial]

mov ax,word ptr num1
mov bx,word ptr num2

sub ax,bx
```

```
mov word ptr ans,ax

mov ax,word ptr num1+2
mov bx,word ptr num2+2
; Subract with the Borrow from Previous Subtraction
sbb ax,bx
mov word ptr ans+2,ax

jnc skip
inc dl  ; Increment the Borrow
skip: mov borrow,dl

; HLT
mov ax,4C00h
int 21h
end
```

Input:

Number1: (97DE 207A)H
Number2: (46AC 1313)H
Answer: (5132 0D67)H

Output:

```
-g

Program terminated normally
-d 076C:0000
076C:0000  02 FE C2 88 16 18 00 B8-00 4C CD 21 7A 20 DE 97
076C:0010  13 13 AC 46 67 0D 32 51-00 FF FF FF FF FF FF FF
076C:0020  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076C:0030  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076C:0040  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076C:0050  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076C:0060  FF FF FF 26 00 FF FF FF-FF FF FF FF FF FF FF FF
076C:0070  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
```

3. Program to multiply signed 16-bit numbers

TASM Code:

```
model small
.8086

.data   ; DATA SEGMENT
num1 dw 0421H
num2 dw 003EH
ans dw ?

.code   ; CODE SEGMENT
mov ax,@data
mov ds,ax

mov ax,num1
imul num2

mov ans, ax

mov ax,4C00h
int 21h
end
```

Input:

Num1: (0421) H
Num2: (003E) H
Answer: (FFFE) H

Hex value:
421 × 3E = **FFFE**

Decimal value:
1057 × 62 = **65534**

Output:

```
-g

Program terminated normally
-d 076B:0000
076B:0000  00 4C CD 21 21 04 3E 00-FE FF 00 00 00 82 0E 00
076B:0010  00 00 00 82 0E 10 00 02-00 82 0E 00 00 00 00 83
076B:0020  0E FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0030  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0040  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0050  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0060  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0070  FF FF FF 26 00 FF FF FF-FF FF FF FF FF FF FF FF
```

4. Program to multiply unsigned 16-bit numbers

TASM Code:

```
model small
.8086

.data
num1 dw 023AH
num2 dw 0072H
ans dw ?

.code
mov ax, @data
mov ds, ax

mov ax, num1
mul num2

mov ans,ax

mov ax,4C00h
int 21h
end
```

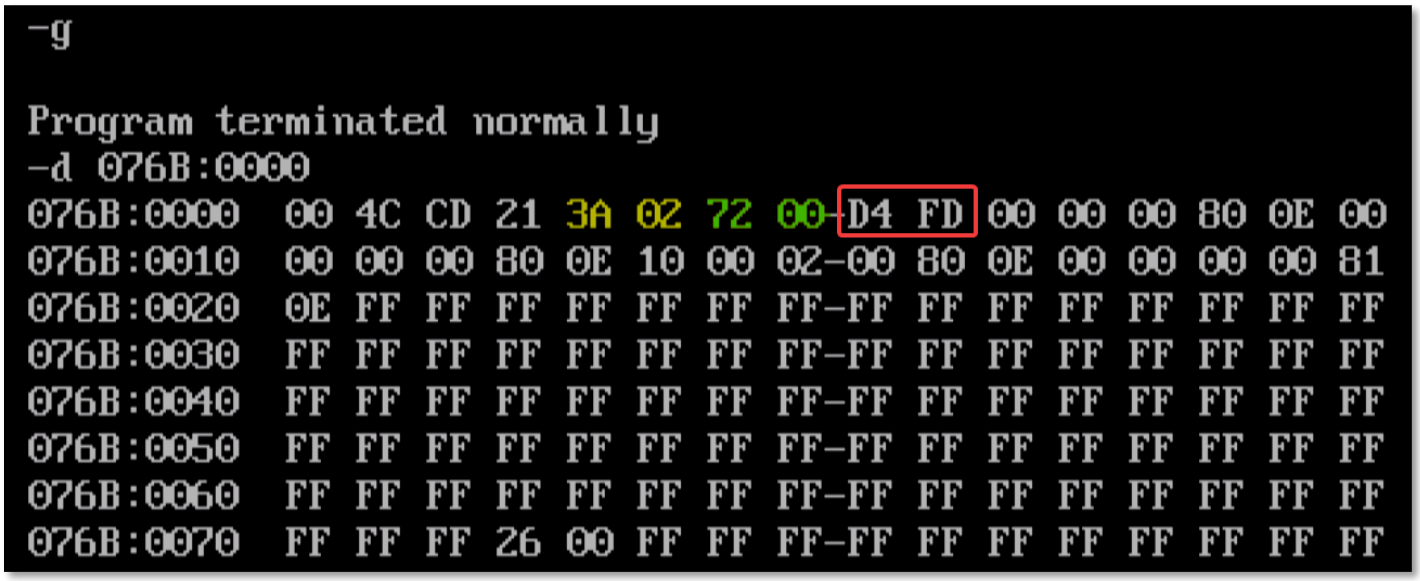
Input:

Num1: (023A) H
Num2: (0072) H
Answer: (FDD4) H

Hex value:
23A × 72 = **FDD4**

Decimal value:
570 × 114 = **64980**

Output:



5. Program for division of unsigned 8-bit numbers

TASM Code:

```
model small
.stack 100
.8086

.data
num1 db 9DH
; This Should not be Zero [Divide by Zero Error]
num2 db 1CH
quotient db ?
remainder db ?

.code
mov ax,@data
mov ds,ax

; Making it 16 Bit [00 num1] [(16 Bit)/(8 Bit)]
```

```
mov ah, 00H
mov al, num1

div num2

mov quotient, al
mov remainder, ah

mov ax,4C00h
int 21h
end
```

Input:

Num1: (9D) H
Num2: (1C) H
Answer: Quotient = (05) H & Remainder = (11)H

Hex value:
9D ÷ 1C = **5 Remainder : 11**

Decimal value:
157 ÷ 28 = **5 Remainder : 17**

Output:

```
-g

Program terminated normally
-d 076B:0000
076B:0000  00 88 26 0D 00 B8 00 4C-CD 21 9D 1C 05 11 0E 00
076B:0010  00 00 00 81 0E 10 00 02-00 81 0E 00 00 00 00 82
076B:0020  0E FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0030  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0040  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0050  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0060  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF 1A 00
076B:0070  6A 07 02 72 00 FF FF FF-FF FF FF FF FF FF FF FF
```

6. Program for division of unsigned 16-bit numbers

TASM Code:

```
model small
.stack 100
.8086

.data
num1 dw 2A81H
num2 dw 003CH
quotient dw ?
remainder dw ?

.code
mov ax,@data
mov ds,ax

mov ax,num1
div num2

mov quotient,ax
mov remainder,dx

mov ax,4c00h
int 21h
end
```

Input:

Num1: (2A81) H

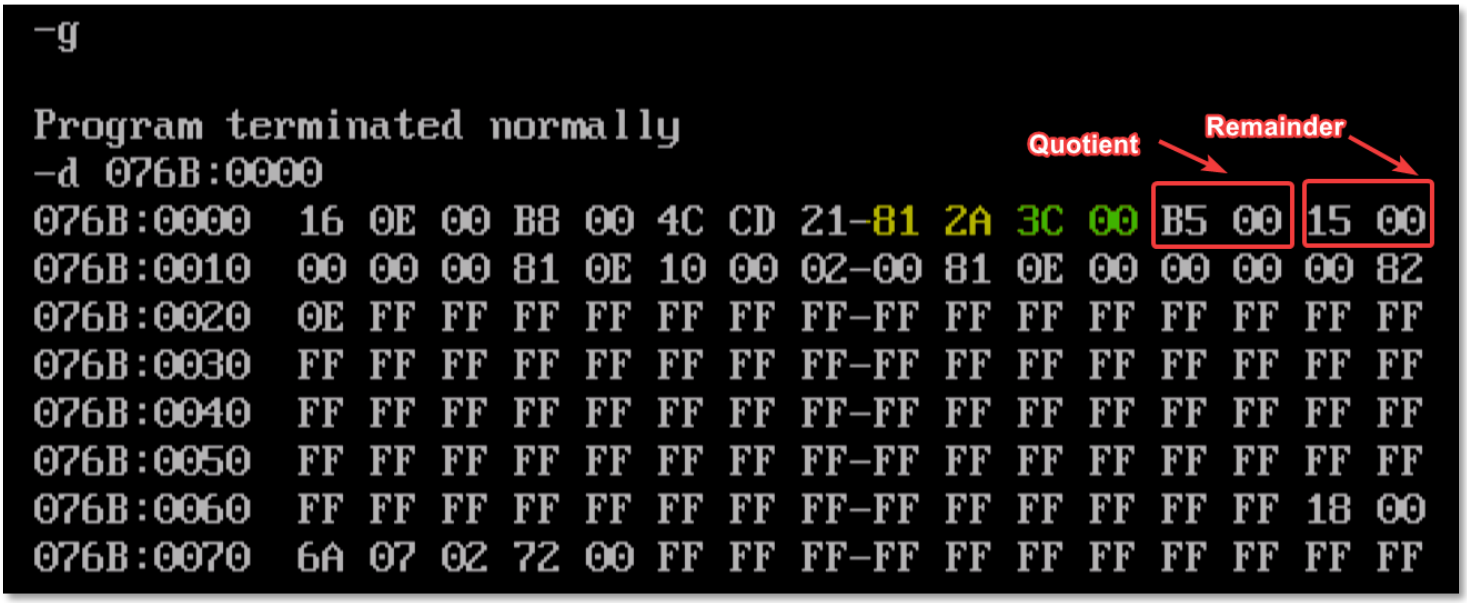
Num2: (003C) H

Answer: Quotient = (B5) H & Remainder = (15)H

Hex value:
2A81 ÷ 3C = **B5 Remainder : 15**

Decimal value:
10881 ÷ 60 = **181 Remainder : 21**

Output:



7. Program for division of signed 8-bit numbers

TASM Code:

```
.model small
.stack 100
.8086

.data
num1 db -82H
num2 db 05H
quotient dw ?
remainder dw ?

.code
mov ax, @data
mov ds, ax

mov ah, 00H
mov al, num1

; CBW (convert byte to word) [Sign Preserving]
cbw
idiv num2

mov quotient, ax
mov remainder, dx

mov ax, 4C00H
int 21h
end
```

- The CBW (convert byte to word) instruction extend s the sign bit of AL into the AH register. This preserves the number 's sign:

```
.DATA
byte_val SBYTE -101
.CODE
mov al, byte_val ; AL = 9Bh
cbw              ; AX = FF9Bh
```

Note that both **9Bh** and **FF9Bh** both equal decimal **-101**, the only difference is the storage size.

Input:

Hex value:
-82 ÷ 05 = **-1A**

Decimal value:
-130 ÷ 5 = **-26**

Output:

```
-g

Program terminated normally
-d 076B:0000
076B:0000  0E 00 89 16 10 00 B8 00-4C CD 21 00 7E 05 19 01
076B:0010  00 00 00 81 0E 10 00 02-00 81 0E 00 00 00 00 82
076B:0020  0E FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0030  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0040  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0050  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0060  FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
076B:0070  FF FF FF 2C 00 FF FF FF-FF FF FF FF FF FF 1B 00
```

8. Program for division of signed 16-bit numbers

TASM Code:

```
.model small
.stack 100
.8086

.data
num1 dw -4FF1H
num2 dw 0512H

quotient dw ?
remainder dw ?

.code
mov ax, @data
mov ds, ax

mov ax, num1

; CWD (convert word to doubleword)
cwd
idiv num2

mov quotient, ax
mov remainder, dx

mov ax, 4C00H
int 21h
end
```

- The CWD (convert word to doubleword) instruction extends the sign bit of AX into the DX register:

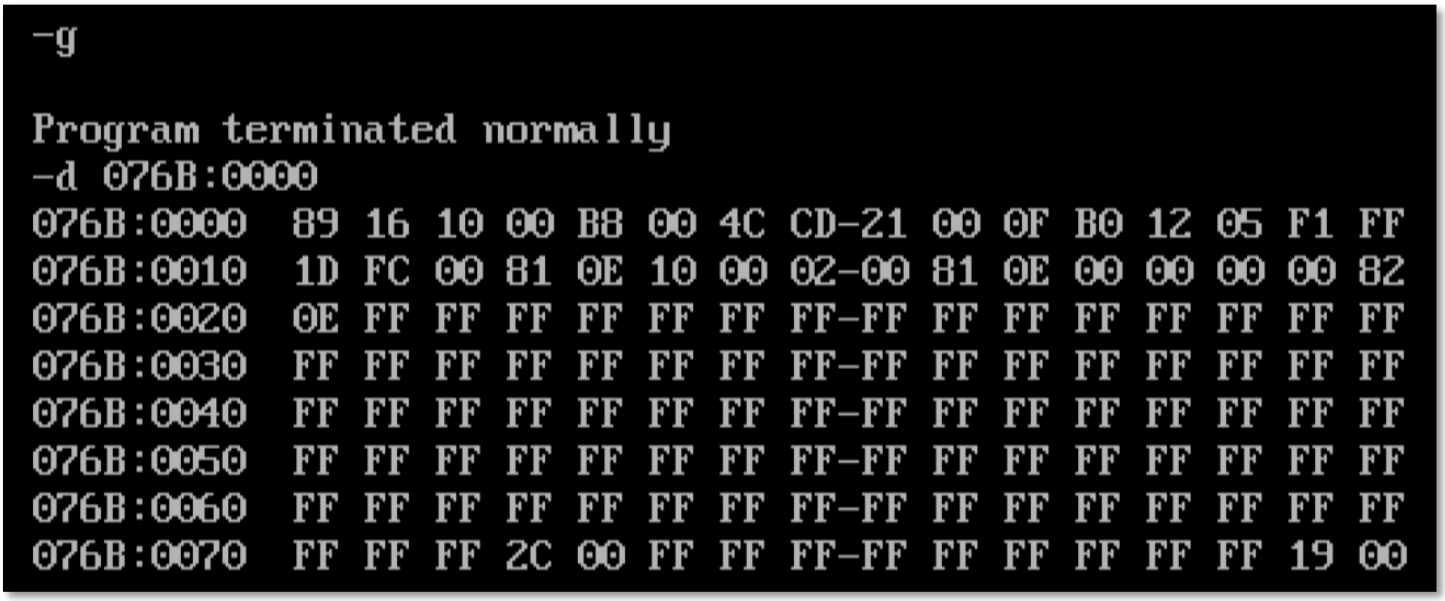
```
.DATA
word_val SWORD -101 ; FF9Bh
.CODE
mov ax, word_val    ; AX = FF9Bh
cwd                 ; DX:AX = FFFFh:FF9Bh
```

Input:

Hex value:
-8251 ÷ 32 = **-29B Remainder : -B**

Decimal value:
-33361 ÷ 50 = **-667 Remainder : -11**

Output:



SUBMITTED BY:

BHAGYA VINOD RANA

[***U19CS012***]