**Computer Organization and Assembly Language**

|  |  |
| --- | --- |
| **Lab 06** | |
| **Topic** | 1. Extended addition, subtraction, shifting 2. Multiply, Divide using shift instructions 3. MUL, IMUL, DIV and IDIV instructions |

**PART 1**

**Example 1(Extended addition)**

MOV AX, Num1 ;loads two bytes into AX register, AX=FFFF

MOV BX, [Num1+2] ;loads Next two bytes into AX register, AX=0001

ADD AX, Num2 ; adds into AX; AX=AX+0002;

ADC BX, [Num2+2]; Now higher bits are added into register

MOV SUM,AX ; Move the lower bits into Sum variable

MOV [SUM+2],BX ; Move the higher bits into Sum variable higher bits

ret

Num1 dd 0x0001FFFF

Num2 dd 0x00010002

SUM dd 0

**Example 2(Extended subtraction)**

MOV AX, Num2 ;loads two bytes into AX register, AX=0002

MOV BX, [Num2+2] ;loads Next two bytes into AX register, AX=0001

SUB AX, Num1 ; sub into AX; AX=AX-FFFF;

SBB BX, [Num1+2]; Now higher bits are subtracted into register

MOV SUM,AX ; Move the lower bits into Sum variable

MOV [SUM+2],BX ; Move the higher bits into Sum variable higher bits

ret

Num1 dd 0x0001FFFF

Num2 dd 0x00010002

SUM dd 0

**Example 3 (Mul and IMUL)**

1. MOV Al, 0x9 ;

MOV Bl, 0x7 ;

MUL Bl ; The product will be in AX in this case

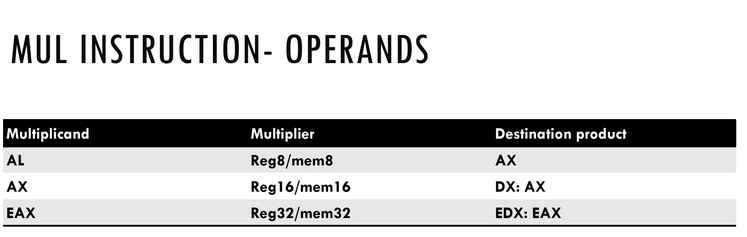
1. MOV AX, 0x90AB ; DECIMAL 37035

MOV BX, 0x332 ; DECIMAL 818

MUL BX ; NOTE IN THIS CASE THE RESULTANT VALUE IS DOUBLE SO THE HIGHER BITS ARE STORED IN DX REGISTER

1. MOV AX, 0x90AB ; DECIMAL -28501

MOV BX, 0x332 ; DECIMAL 818

IMUL BX ; NOTE IN THIS CASE THE RESULTANT VALUE IS DOUBLE SO THE HIGHER BITS ARE STORED IN DX REGISTER 

**Example 4 (Div)**

a)

MOV AL, 9 ;

MOV BL, 4 ;

DIV BL ;Divides AX/BL, gives AL=02 (quotient) and AH=01 (remainder);

In this example, it is assumed that value in AH=0. If value in AH is not zero, same code will produce different results.

b)

MOV AH,01

MOV AL,9 ; AX=0109

MOV BL, 4 ; BL=04

DIV BL ; 265/4 gives quotient AL=42 hex (66 decimal) and remainder AH=01.

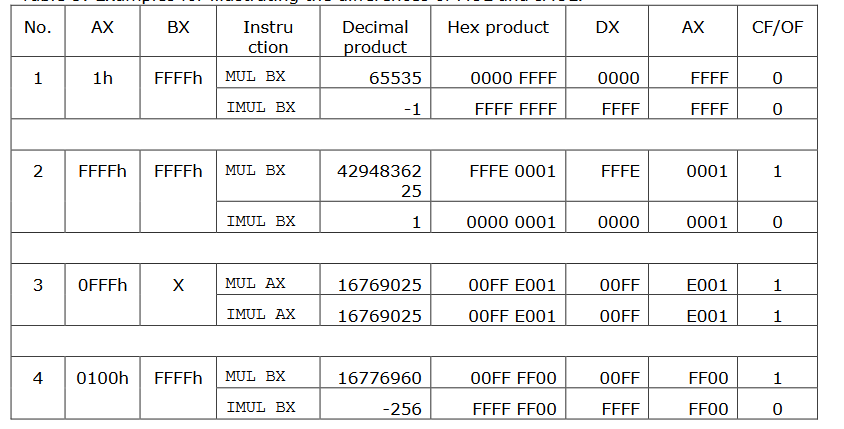
c)

MOV AX, 9 ; AX=0009

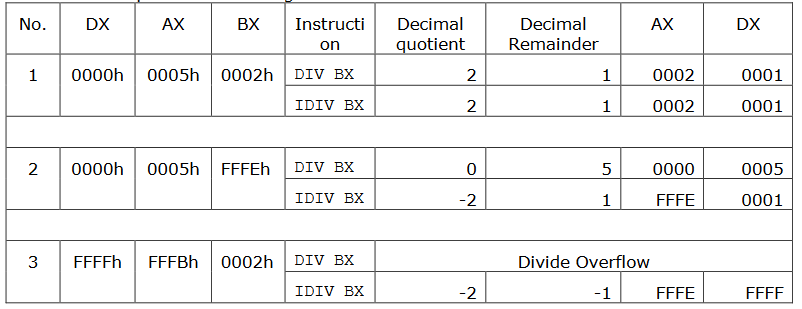
MOV BX, 4; BX=0004

DIV BX ;Divides [DX:AX]/BX, gives AX=0002 (quotient) and DX=01 (remainder);

Examples for illustrating the differences of MUL and IMUL



Examples for illustrating the differences of DIV and IDIV



**PART 2**

**Problem #1:**

**Write an assembly language program to perform extended addition on the values of array1 and extended subtraction on array2.**

**Let the array1:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Index | 0 | 1 | 2 | 3 | 4 |
| VALUE | 0XAACBEF11 | 0X32CF4523 | 0X2C94B44 | 0X9D112233 | ?RESULT? |

**Array[4]=Array[0]+Array[1]+Array[2]+ Array[3]**

**Ans:**

jmp start

array1 dd 0XAACBEF11,0X32CF4523,0X2C94B44,0X9D112233

RESULT dd 0

start:

MOV AX, [array1]

MOV BX, [array1+2]

ADD AX,[array1+4]

ADC BX,[array1+6]

ADC AX,[array1+8]

ADC BX,[array1+10]

ADC AX,[array1+12]

ADC BX,[array1+14]

MOV RESULT,AX

MOV [RESULT+2],BX

ret

**Let the array2:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| INDEX | 0 | 1 | 2 | 3 | 4 |
| VALUE | 0XFE3B5672 | 0XADABDD32 | 0X3911219F | 0X24191111 | ?RESULT? |

**Store the subtraction of array on last element.**

**Array[4]=Array[0]-Array[1]-Array[2]- Array[3]**

**Ans:**

jmp start

array2 dd 0XFE3B5672,0XADABDD32,0X3911219F,0X24191111

RESULT1 dd 0

START:

MOV AX, [array2]

MOV BX, [array2+2]

SUB AX,[array2+4]

SBB BX,[array2+6]

SBB AX,[array2+8]

SBB BX,[array2+10]

SBB AX,[array2+12]

SBB BX,[array2+14]

MOV RESULT1,AX

MOV [RESULT1+2],BX

ret

**Problem #2**

**Write an assembly language program to find the area of the TRIANGLE?**

**(use mul and div for multiplication and division)**

**Height=20**

**BASE=15**

**Ans:**

**jmp start**

**Height dw 20**

**Base dw 15**

**Area dw 0**

**start:**

**mov ax,Height**

**mov bx,Base**

**mul bx**

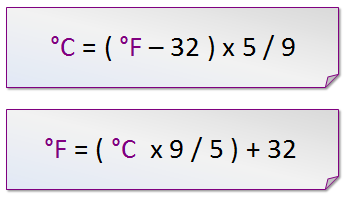
**mov bx,2**

**div bx ;Area in Ax**

**mov Area,ax**

**Problem #3**

**Write an assembly language program to convert the temperature units.**



**a)**

**b)**

*For example:-* 35℃⬄95℉

Ans:1) Fahrenheight To Celsius.

jmp start

Farenhieght db 0x95h

Celsius db 0x0h

start:

mov al,Farenhieght

sub al,0x32h

mov bl,0x5h

mul bl

mov bl,0x9h

div bl

mov Celsius,al

2) Celsius To Fahrenheight.

jmp start

Celsius db 0x37h

Farenhieght db 0x0h

start:

mov al,Celsius

mov bl,0x9h

mul bl

mov bl,0x5h

div bl

add al,0x32h

mov Farenhieght,al

**Problem #4**

**Write an assembly language program to check whether any two word sized numbers are multiples of each other or not? Take values of your choice.**

**If they are found multiple then clear the accumulator else set the accumulator from maximum possible signed number.**

*Hint: For example, 20 is divisible by 4, so 20 is a multiple of 4*

*Ans:*

*jmp start*

*val dw 30*

*val2 dw 10*

*start:*

*mov ax,val*

*mov bx,val2*

*div bx*

*cmp dx,0*

*jz label1*

*jnz label2*

*label1:*

*mov ax,0x0000*

*ret*

*label2:*

*mov ax,0x7FFF*

*ret*

**Problem # 5**

**Write an assembly language program to find the prime numbers in an array.**

**If a number is prime replace it with 1 else if it is not replace with 0.**

Let values are (byte size)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Value | 0x0C | 0xCF | 0x1E | 0x13 | 0x3 | 0x0D | 0x84 | 0x14 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Value | 1 | ? | ? | ? | ? | ? | ? | 0 |