***Practical Example of Assembly Language***

***By:***

***M.Mustafa Hayat\_Amarkh***

First Program:

;================================================================

# # 01) Hello, World Program;

; ===============================================================

org 100h ; mean start my program for 100h address

; required field

include "emu8086.inc" ; include the library for the print commond

; required field when to print text

print "Hello, World" ; simple print a text

ret ; mean return the program

# # 02) Adding break between two line:

include emu8086.inc

org 100h

print "This is Mohammad Mustafa HayatAmarkhil"

print 10 ; add line break

print 13 ; clear the register

int 13h ; intrupt in the operation

print "Welcom to assembly Language"

int 21h ; used to stop the screen

; the statment after that is not executed.

Ret

# # 03) Addition, Subtraction and Multiplication;

org 100h

include "emu8086.inc" ;this is the library used for

; the string value output and print.

mov ax, 3

mov bx, 5

add ax, bx

; adding two number

; subtraction of two number

sub ax, 2

; ax, bx, cx register are used for the input operation

; and dx register for both input and output operation

; addition and subtraction could perform on all register

; but the multiplication operation is performed and the result is

; stored only in the ax register

print "operation is performed, addition and subtraction"

int 10h

; int 10h is used to clear the screen and start from the beginning

int 13h

; interrupt the operation

print "the Multiplaction will start now..."

mul bx

ret

# # 04) Print value Using DX register

;, this register is used for both

; input and output

; the ax, bx, cx registers are used only for input.

include emu8086.inc

org 100h ; mean that start the program for the 100h address

; print data using the dx register

mov ax, 12

mov dx, ax ; to print the data we must assign it to the dx register

; the dx will print the data in the character form

mov ah, 2 ; this is used to print the value into the screen working

; same as cout in the C++

int 21h ; stop the screen like getch() in the c++

;===================================================================

; if you want to print the data in the Decimal form we use the

; following methode:

; in the next example

;===================================================================

;==================================================================

# # 05) print a value in the decimal form

;==================================================================

include emu8086.inc

org 100h

define\_print\_num ; is used to print the negative number

define\_print\_num\_uns ; is used to print unsigned (positive) number.

; this is necessary for the printing a decimal value

; to print it we use the following commonds:

; call print\_num / this is use for print a signed value(-)

; call print\_num\_uns / this s used for printting a unsigned (+)

; example

; write an example to insert two value, add these two values and

; after that show the addition result in the decimal form in the

; screen

; solution:

mov bx, 3

mov cx, 4

add bx, cx ; added

; one main point that befor that you want to print the value you

; should first assign it in the as register, otherwise the default

; value of the ax that is 0 will printed.

; first without assign to ax:

print "value without assigning to ax: "

call print\_num\_uns ; this is positive number

; this will show the 0 in the screen

print 10 ; line break

print 13 ; clear the register

; with assigning to ax it will show the correct result:

mov ax, bx

print "value after assigning a value to as: "

call print\_num\_uns

ret ; return

# # 06) Procedure in Assembly Language:

; write a program in the assembly using procedure

; the first procedure, to add two number

; the second procedure, to subtract two number and show

; the third procedure, to multiply two number and show

org 100h

include emu8086.inc

define\_print\_num\_uns

define\_print\_num

define\_scan\_num

call pr1

call pr2

call pr3

ret

proc pr1

mov ax, 3

mov bx, 4

add ax, bx

print "Addition: "

call print\_num\_uns

printn

endp pr1

ret

proc pr2

mov ax, 3

mov bx, 4

sub ax, bx

print "Subtraction: "

call print\_num

printn

endp pr2

ret

proc pr3

mov ax, 3

mov bx, 4

Mul bx

print "Multiplication: "

call print\_num\_uns

endp pr3

ret

# # 07) loop in the assemble:

org 100h

include "emu8086.inc"

printn "Hello, World"

mov cx, 3 ; we use the cx register for th loop so, it's the

; the counter register

Lable: ; define a lable

printn "one" ; print simple massege

; the printn mean that print the message and bring the

; the cursor to the new line

; it is the usage of the print 10 and print 13

; we use the printn instead of that.

loop Lable; ; loop the lable till the spicified numbers

; write a program to print the number in 0, 1, 2, 3, 4,

mov cx, 4

lab:

call print\_num\_us

print ", "

inc ax

loop lab;

# # 08) Dynamically insert a value to the Screen

org 100h

include "emu8086.inc"

define\_print\_num\_uns

define\_print\_num

define\_scan\_num ; it's used to take dynamically value from screen

; we are define this statement for it

printn "Hello, World"

call scan\_num; ; if we want input a value frome screen

; we use this for input

; and the value will be stored in the

; cx register by defualt

mov ax, cx ; if you want show the value dynamically we have

; to move the value to ax and then to print it.

print "input is:"

call print\_num\_uns

; write a program to start the loop dynamically

printn "Start..."

printn

print "Value For Loop: "

call scan\_num

printn

lab:

call print\_num\_uns

print ", "

inc ax

loop lab;

# # 09) decrement the value and print it in the descending order:

org 100h

include emu8086.inc

define\_print\_num\_uns

define\_print\_num

define\_scan\_num

printn "Hello, World"

printn

print "Enter a value you want to start: "

call scan\_num

printn

mov ax, cx

printn "The decrement is ready to start..."

Lable:

call print\_num\_uns

printn

; we can use the print 10 and print 13 as well

dec ax

loop Lable;

**Midterm Examinations**

# # 10) Procedure in the Assembly Language"

org 100h

include emu8086.inc

printn "Hello, World"

call pr1

call pr2

call pr3

ret

proc pr1

printn "This is the pre1"

endp pr1

ret

proc pr2

printn "This is the pre2"

endp pr2

ret

proc pr3

printn "This is the pre3

endp pr3

ret

# # 11) loop in Assembly

org 100h

include emu8086.inc

define\_print\_num\_uns

define\_print\_num

define\_scan\_num

printn "Hello, World"

; the program using loop

mov cx, 4 ; this register is use for counter variable

mov ax, cx

lbl:

call print\_num\_uns

printn

dec ax

loop lbl; ; this loop will operate according to cx value

; this loop will continue fro 4 till one.

# # 12) Do while loop in Assembly

; Do while loop program.

org 100h

include emu8086.inc

define\_print\_num\_uns

define\_print\_num

mov ax, 1

mov bx, 5

dwloop:

call print\_num\_uns

printn

cmp ax, bx

je whilloop ;if equal jump to the lable named whilloop.

inc ax ;increment ax.

jmp dwloop ; call again the lable named dwloop

whilloop:

printn "While loop is ended."

ret

# #13) Declaration of the variable in the Assembly

org 100h

include emu8086.inc

define\_print\_num\_uns

printn "hello, World"

; to declar we use the following symbols:

; db : decalare in byte = 8 bits

; dw : declare in word = 16bits (2byts)

; ddw : declare double word = 32bits (4byts).

; dqw : declare quad wor = 64bits (8byts)

; syntax of declaring the variable

; name space size space value

; example x db 3

x dw 13

; Note:

; we have to assign the variable value to that register

; that has same size.

; like value from 0-7 to al, ah......

; value from 0-15 to ax, bx, cx, dx

; value from 0-32 eax, ebx......

mov ax, x

call print\_num\_uns

# # 14) Find the Greater Number b/w Three Number:

; program for finding the

; greatest Number between

; three Numbers.

org 100h

include emu8086.inc

define\_print\_num\_uns

define\_print\_num

define\_scan\_num

mov ax, 20

mov bx, 30

mov cx, 4

cmp ax, bx

je equal

jg axgreater

jl axlittle

axgreater:

cmp ax, cx

je equal

jg axgreatest

jl cxgreater

axlittle:

cmp bx, cx

je equal

jg bxgreater

jl cxgreater

equal:

print "both Are equl"

jmp exit;

bxgreater:

print "BX is greater."

jmp exit;

cxgreater:

print "CX greater."

jmp exit;

axgreatest:

print "AX greatest."

exit:

ret

# # 15) add and subtract three users define variable

org 100h

include emu8086.inc

define\_print\_num\_uns

define\_print\_num

x dw 10

y dw 20

z dw 2

call pro\_sum

call pro\_sub

proc pro\_sum

mov ax, x

add ax, y

add ax, z

call print\_num\_uns

printn

ret

endp pro\_sum

proc pro\_sub

mov ax, x

sub ax, y

sub ax, z ; in subtract not correct result

call print\_num

ret

endp pro\_sub

# # 16) Find the smallest number between three number

org 100h

include emu8086.inc

mov ax, 10

mov bx, 3

mov cx, 0

cmp ax, bx

js axsmall

jg axlarge

axsmall:

cmp ax, cx

js axsmaller

jg cxsmaller

axsmaller:

print "ax is the smallest"

ret

cxsmaller:

print "cx is the smallest."

ret

axlarge:

cmp bx, cx

js bxsmaller

jg cxsmaller

bxsmaller:

print "bx is the smallest"

ret

org 100h

include emu8086.inc

define\_print\_num\_uns

define\_print\_num

define\_scan\_num

define\_clear\_screen; will clear the screen

define\_pthis; to display text on screen using

; variable of type db

call pthis

var db 'ahmad is my Brother', 0; we give 0 because

; it prevent from other

;garbage values.

printn

print "This is another."

call clear\_screen ; it will clear all screen.

PUTC 65; it will show the character of the A

PRINTN

Putc 'A'; it will show A as well.

# # 17) ALL LIBRARIES

INCLUDE EMU8086.INC

ORG 100H

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_PRINT\_NUM

DEFINE\_SCAN\_NUM

DEFINE\_PTHIS

DEFINE\_CLEAR\_SCREEN

PRINTN "HELLO Mustafa Hayat"

COMMENT!

PRINT "VALUE: "

CALL SCAN\_NUM

PRINTN

MOV AX, CX

PRINT "VALUE IS: "

CALL PRINT\_NUM

PRINTN

CALL PTHIS

VAR DB "SALAM ALIKOM",0

PRINTN

PRINT "CLEAR SCREEN"

CALL CLEAR\_SCREEN !

PUTC 97

PUTC 'A'

RET

INCLUDE EMU8086.INC

ORG 100H

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_PRINT\_NUM

DEFINE\_SCAN\_NUM

DEFINE\_PTHIS

DEFINE\_CLEAR\_SCREEN

MOV AX, 0

MOV BX, 3

MOV CX, 1

CMP AX, BX

JL AXSMALL

JG BXSMALL

AXSMALL:

CMP AX, CX

JL AXSMALLER

JG CXSMALLER

BXSMALL:

CMP BX, CX

JL BXSMALLER

JG CXSMALLER

AXSMALLER:

PRINT "A IS SMALL: "

CALL PRINT\_NUM

JMP EXIT;

BXSMALLER:

PRINT "B IS SMALL: "

MOV AX, BX

CALL PRINT\_NUM

JMP EXIT;

CXSMALLER:

PRINT "C IS SMALL: "

MOV AX, CX

CALL PRINT\_NUM

EXIT:

RET

***THE END***

**After an exam extra work:**

# # 18) Multiplication Table

include emu8086.inc

org 100h

define\_print\_num ; is used to print the negative number

define\_print\_num\_uns; is used to print unsigned (positive) number.

mov bx, 1 ; start of the loop

mov cx, 10 ; the number of loop

Lable: ; position called again and agian

print 10 ; line break

print 13 ; clear the register

mov ax, 4 ; assign number (4) beacuse we want to print this table

call print\_num\_uns ; simply print 4, in this case

print " \* " ; simply print \* sign

mov ax, bx ; move the bx value to ax, to print to the screen

call print\_num\_uns ; print to screen

print " = " ; print = sign

mov ax, 4 ; move the number 4 to ax, for printing to scr

; do the multiplication

mul bx

call print\_num\_uns ; print to screen

inc bx ; increment the bx value by one

loop Lable; ; carry the control back to the 'Lable' part

ret ; return

# Note:

If you have any question, feel free and ask me about, I will proudly answer you as soon as possible.

# Credit goes to:

Mohammad Mustafa Hayat, Data analyst & Developer

# Email: [hayatzaimustafa@gmail.com](mailto:hayatzaimustafa@gmail.com)