Lab2- Introduction to MASM

2.0 Introduction

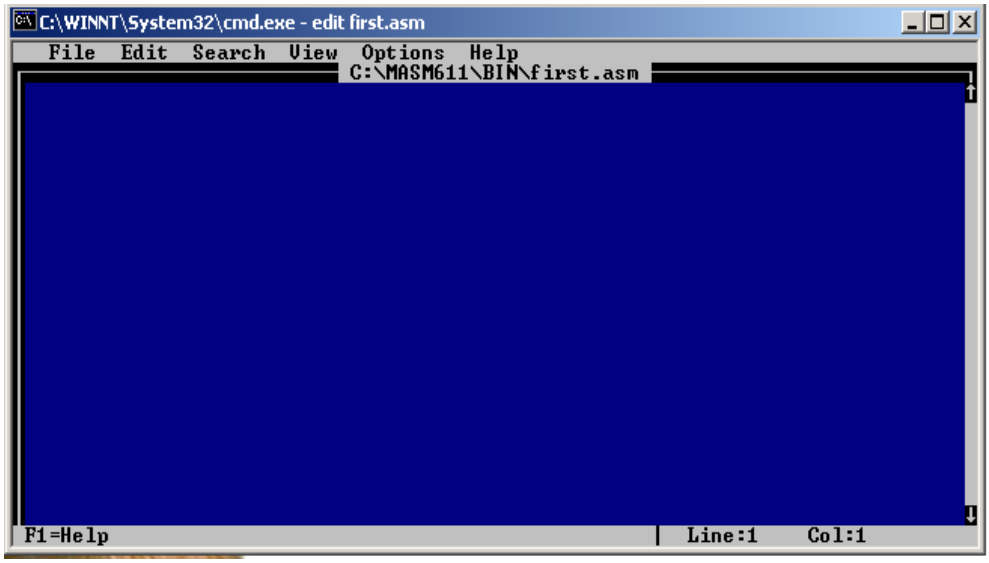
Assembly language unlocks the secrets of computers hardware and software. An assembler converts source code to machine own language. Microsoft Macro Assembler (MASM), product of Microsoft Corporation includes several features which makes programming efficient and productive. The following chapter will give an overview how to use MASM for assembling the 80x86 program you code.

Editor

We will use DOS text editor to type our program. Go to directory MASM611\BIN type edit filename.asm

(You can also use any other editor such as Notepad/Notepad++)

Then a blue screen will appear



Use "Alt" Key to start the "Top Menus", then use "Arrow Keys" to select entries in the "Menu".

- Arrow keys: move cursor up, down, left and right
- Insert key: enter insert mode, insert a character at the cursor
- Backspace key: delete the character before the cursor
- Home key: go to the beginning of a line
- End key: go to the end of a line
- Delete key: delete the character after the cursor.

2.0 Program Template

2.1 Using Model Tiny

All the data and code fit in one segment Tiny programs when compiled give .COM executable - the program must be originated at location 100_H

```
.Model Tiny
.386                                        ;Assembler by default accepts only 8086/8088 instructions, unless a program is preceded by .386/.486 directive to select the microprocessor
.data                                       ; Data Segment
COUNT       EQU     32H
VAL1        EQU     0030H
DAT1        DB      45H,67H,100,'A'
WRD         DW      10H, 3500H,0910H
DAT2        DD      0902H
DAT3        DW      2 DUP(0)
DAT4        DB      56H
RES         DB      10 DUP(?)
DWRD        DD      01020304H
.CODE
.STARTUP
            MOV     SI,DAT3
            MOV     AL, DAT1 + 1
            MOV     BX,WORD PTR DAT1+4
            ADD     BX,20H
            MOV     Al,[BX]
            LEA     BX,DAT4
            MOV     AL,[BX]
            MOV     BX,VAL1
            MOV     AL,FS:[BX]
            MOV     EBX, DWRD
.EXIT
END
```

2.1 Assembling

There are two methods of assembling

Method 1:

Type MASM filename.asm <enter>

If no error in code there is .OBJ file is generated

Now type

LINK filename.obj <enter>

Check the files created at each step and examine the content of .lst and .map file

Method 2:

Typ'e ML filename.asm <enter>

To create list and map file command format is 'ml /Fl /Fm Filename.asm

Check the files created at each step and examine the content of .lst and .map file

What is the difference between Method 1 and Method 2?

To check the working of the program – execute DEBUGX ***Filename.com*** and then trace or go in DEBUGX.

2.1 Using Model Small

Questions

(1)What are the errors if you just change Tiny to Small in the .Model Statement?

(2) Is there a PSP in .Model Small?

(3) Remove .386 statement. When you assemble what is the error? Why is there an error?

To check the working of the program – execute **DEBUGX *Filename.exe*** and then trace or go in DEBUGX.

Note:

1. **All your files .asm, .com/.exe must be present in MASM611/BIN**
2. **Make sure that you copy debugx.exe into MASM611/BIN folder**

Tasks:

1. Write an ALP that finds the maximum number from a set of 32-bit numbers
2. Write an ALP to add 2 16-byte nos. using them
    a. as 16-bit data
    b. as 32-bit data
3. Write an ALP that will examine the contents of set of 10 bytes starting from location ***'ARRAY1'*** for the presence of data '$0A_H$' and replace it with the ASCII character 'E'.
4. Write an ALP that will count the number of negative numbers in an array of 16-bit signed data stored from location 'ARRAY1'. The number of elements in the array is present in location 'COUNT'. The count of negative numbers must be stored in location 'NEG1'
5. Write an ALP that will transfer data from ***'ARRAY1'*** to ***'ARRAY2'.*** The number of elements in the array is 10. The array is a double word array. The starting address of ***ARRAY2*** = starting address of ***ARRAY1*** + 20_d.