

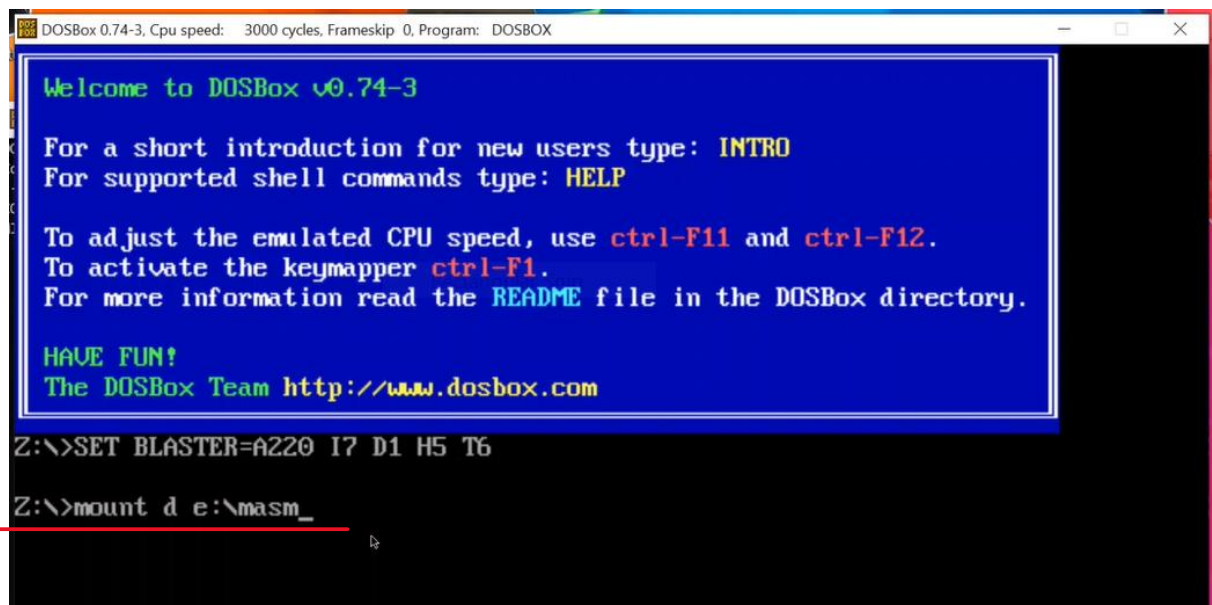
SSN College of Engineering
Department of Computer Science and Engineering
III year - UCS2512 – Microprocessors Lab

Procedure for executing 8086 programs in MASM assembler

1. Install the software DOSBox 0.74-3

(The setup file for windows can be downloaded from the **lab LMS page** or from the following link <https://www.dosbox.com/download.php?main=1>)

2. Download the masm.zip file from **lab LMS page** and unzip all files to the folder “masm” (eg: E:\masm\)
3. Run Dosbox and mount your masm folder to a drive in dosbox



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

Welcome to DOSBox v0.74-3

For a short introduction for new users type: INTRO
For supported shell commands type: HELP

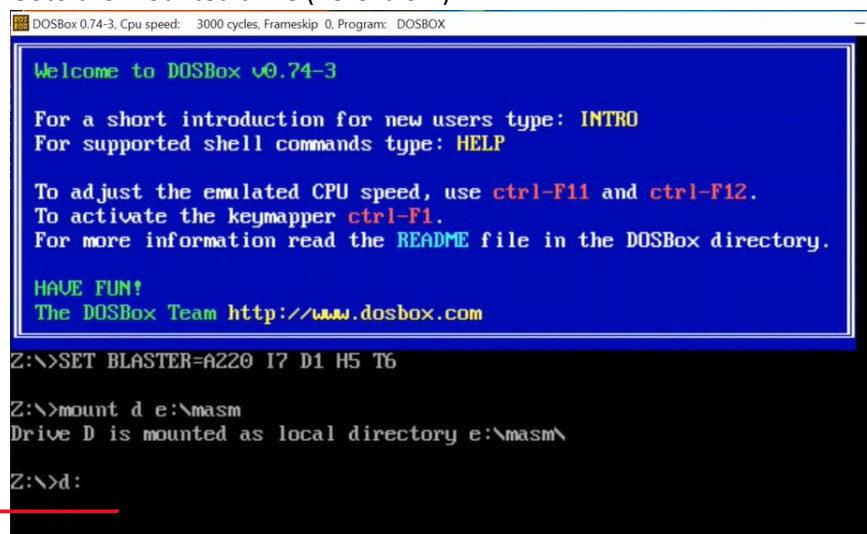
To adjust the emulated CPU speed, use ctrl-F11 and ctrl-F12.
To activate the keymapper ctrl-F1.
For more information read the README file in the DOSBox directory.

HAVE FUN!
The DOSBox Team http://www.dosbox.com

Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>mount d e:\masm_
```

4. Goto the mounted drive (here it is D)



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

Welcome to DOSBox v0.74-3

For a short introduction for new users type: INTRO
For supported shell commands type: HELP

To adjust the emulated CPU speed, use ctrl-F11 and ctrl-F12.
To activate the keymapper ctrl-F1.
For more information read the README file in the DOSBox directory.

HAVE FUN!
The DOSBox Team http://www.dosbox.com

Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>mount d e:\masm
Drive D is mounted as local directory e:\masm\

Z:\>d:
```

5. Save the 8086 program with extension .asm in the same folder using command “edit”.

```
EDIT      EXE           81,864 23-03-2001  1:20
LINK      EXE           42,330 08-07-2009 14:15
MASM      EXE           49,152 19-08-1993 18:50
ML        ERR           9,287 24-09-1993  8:21
ML        EXE          388,608 24-09-1993  8:25
      8 File(s)          986,247 Bytes.
      2 Dir(s)          262,111,744 Bytes free.

D:\>edit 8bitadd.asm
```

6. After creating the file, assemble it using the command “masm filename.asm”

```
D:\>masm 8BITADD.ASM
Microsoft (R) MASM Compatibility Driver
Copyright (C) Microsoft Corp 1993. All rights reserved.

Invoking: ML.EXE /I. /Zm /c /Ta 8BITADD.ASM

Microsoft (R) Macro Assembler Version 6.11
Copyright (C) Microsoft Corp 1981-1993. All rights reserved.

Assembling: 8BITADD.ASM

D:\>
```

7. Link the file using the command “link filename.obj;”

```
D:\>link 8bitadd.obj;

Microsoft Object Linker V2.01 (Large)
(C) Copyright 1982, 1983 by Microsoft Inc.

Warning: No STACK segment

There was 1 error detected.

D:\>
```

This warning and error can be neglected as the stack segment is not being used in this program.

8. Use debug command with filename.exe to execute and analyse the memory contents, "debug filename.exe".

```
MASM      EXE      49,152 19-08-1993 18:50
ML        ERR      9,287 24-09-1993  8:21
ML        EXE     388,608 24-09-1993  8:25
      12 File(s)      987,968 Bytes.
      2 Dir(s)      262,111,744 Bytes free.

D:\>debug 8bitadd.exe
```

9. In debug, command "u" will display the unassembled code

```
MASM      EXE      49,152 19-08-1993 18:50
ML        ERR      9,287 24-09-1993  8:21
ML        EXE     388,608 24-09-1993  8:25
      12 File(s)      987,968 Bytes.
      2 Dir(s)      262,111,744 Bytes free.

D:\>debug 8bitadd.exe
-u
076B:0100 B86A07      MOV     AX,076A
076B:0103 8ED8        MOV     DS,AX
076B:0105 8A260000     MOV     AH,[0000]
076B:0109 8A3E0100     MOV     BH,[0001]
076B:010D B500        MOV     CH,00
076B:010F 02E7        ADD     AH,BH
076B:0111 7302        JNB     0115
076B:0113 FEC5        INC     CH
076B:0115 88260200     MOV     [0002],AH
076B:0119 882E0300     MOV     [0003],CH
076B:011D B44C        MOV     AH,4C
076B:011F CD21        INT     21
```

8BITADD.ASM - Notepad

```
;Program for adding 2, 8 bit numbers
assume cs:code,ds:data
data segment
    opr1 db 11h
    opr2 db 99h
    result db 00H
    carry db 00H
data ends
code segment
    org 0100h
start: mov ax,data
       mov ds,ax

       mov ah,opr1
       mov bh,opr2
       mov ch,00h
       add ah,bh
       jnc here
       inc ch
here:  mov result,ah
```

- Use command “d segment:offset” to see the content of memory locations starting from segment:offset address

The screenshot shows a debugger window with two panes. The left pane displays assembly code with addresses from 076B:0100 to 076B:011F. The right pane shows a memory dump starting at 076A:0000. Red arrows indicate the mapping of instructions to memory addresses. A Notepad window titled '8BITADD.ASM' is overlaid on the right, showing the source code for a program that adds two 8-bit numbers.

```

-u 076b:0100
076B:0100 B86A07      MOV     AX,076A
076B:0103 8ED8        MOV     DS,AX
076B:0105 8A260000     MOV     AH,[0000]
076B:0109 8A3E0100     MOV     BH,[0001]
076B:010D B500        MOV     CH,00
076B:010F 02E7        ADD     AH,BH
076B:0111 7302        JNB     0115
076B:0113 FEC5        INC     CH
076B:0115 88260200     MOV     [0002],AH
076B:0119 882E0300     MOV     [0003],CH
076B:011D B44C        MOV     AH,4C
076B:011F CD21        INT     21

-d 076a:0000
076A:0000 11 99 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0030 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00

8BITADD.ASM - Notepad
File Edit Format View Help
;Program for adding 2, 8 bit numbers
assume cs:code,ds:data
data segment
    opr1 db 11h
    opr2 db 99h
    result db 00H
    carry db 00H
data ends
code segment
    org 0100h
start: mov ax,data
       mov ds,ax

       mov ah,opr1
       mov bh,opr2
       mov ch,00h
       add ah,bh
       jnc here
       inc ch
  
```

- To change the value in memory, use the command “e segment:offset”

The screenshot shows a debugger command window with the following text:

```

D:\>debug 8bitadd.exe
-e 076a:0000
076A:0000 11.80 99.80
  
```

Here “80s” are the new values given by the user . Press space if you want to edit the next location.
To stop editing, press enter.

- Verify the memory contents to ensure the updates (using command “d”).

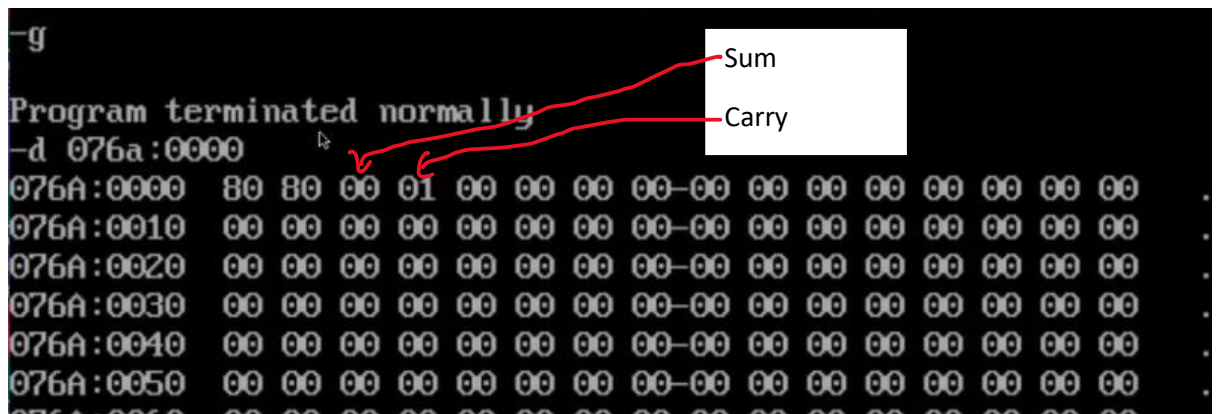
The screenshot shows a debugger window with a memory dump starting at 076A:0000. The values at 076A:0000 and 076A:0010 have been updated to 80 and 80 respectively.

```

-d 076a:0000
076A:0000 80 80 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0030 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0050 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
  
```


13. Execute using the command “g” and check the outputs.

```
-g
Program terminated normally
-d 076a:0000
076A:0000  80 80 00 01 00 00 00 00 00 00 00 00 00 00 00 00
076A:0010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
076A:0050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```



14. Command “q” to exit from debug and command “exit” from command prompt to close dosbox

```
-q
D:\>exit
```