



Microprocessor and Assembly Language Lab

Lab Material 4 for CSE 312 (M&AL Lab)

Dr. Shah Murtaza Rashid Al Masud

Associate Prof.

Dept. of CSE, UAP

Arithmetic and Logic Operations

Arithmetic Operations

Introduction to Arithmetic

Instructions (INC, DEC, ADD, SUB
and NEG)

Arithmetic Operations

INC and DEC Instructions

The INC (increment) and DEC (decrement) instructions, respectively, add 1 and subtract 1 from a single operand . The syntax is:

INC *reg/mem*

DEC *reg/mem*

Example:

Following are some examples :

MOV AX, 1000H

INC AX

INC AL

INC AH

DEC AX

```
org 100h
```

```
MOV AX, 1000H  
INC AH
```

```
ret
```

emulator: noname.com_

file math debug view external virtual devices virtual drive help

Load reload step back single step run step delay ms: 0

registers

| | H | L |
|----|----|----|
| AX | 11 | 00 |
| BX | 00 | 00 |
| CX | 00 | 06 |
| DX | 00 | 00 |

0700:0105

| | | | |
|--------|----|-----|------|
| 07100: | B8 | 184 | 7 |
| 07101: | 00 | 000 | NULL |
| 07102: | 10 | 016 | ▶ |
| 07103: | FE | 254 | ! |
| 07104: | C4 | 196 | - |
| 07105: | C3 | 195 | |

0700:0105

| |
|----------------|
| MOV AX, 01000h |
| INC AH |
| RET |
| NOP |
| NOP |
| NOP |

original source co...

```
01  
02 ; You may customize this  
03 ; The location of this t  
04  
05 org 100h  
06  
07  
08 MOV AX, 1000H  
09 INC AH  
10  
11  
12  
13 ret  
14  
15
```

ADD Instruction

The ADD instruction adds a source operand to a destination operand of the same size. The syntax is:

ADD dest,source

Source is unchanged by the operation, and the sum is stored in the destination operand . The set of possible operands is the same as for the MOV instruction .

Example:

Here is a short code example that adds two 16-bit integers:

```
ORG 100h
```

```
MOV AX, var1
```

```
MOV BX, var2
```

```
ADD AX,BX
```

```
RET ; stops the program.
```

```
VAR1 DW 7
```

```
var2 DW 1234h
```












org 100h? **ORG** (abbr. for ORiGin) is an **assembly** directive (not an instruction). It defines where the machine **code** (translated **assembly** program) is to place in memory.

ADD Instruction

 emu8086 - assembler and microprocessor emulator 4.08

—

[file](#) [edit](#) [bookmarks](#) [assembler](#) [emulator](#) [math](#) [ascii codes](#) [help](#)

 new
  open
  examples
  save
  compile
  emulate
  calculator
  converter
  options
  help
  about

```
01 ORG 100h
02
03 MOV AX, var1
04 MOV BX, var2
05 ADD AX, BX
06
07 RET ; stops the program.
08
09 VAR1 DW 7
10 var2 DW 1234h
```

The screenshot shows the emulor emulator window titled "emulator: noname.com_". The menu bar includes file, math, debug, view, external, virtual devices, virtual drive, and help. Below the menu are several control buttons: Load, reload, step back, single step, run, and a slider for step delay ms: 0.

The main area displays registers on the left and two panels on the right. The registers panel lists AX through DS with their current values. The top-right panel shows the selected register's address and value. The bottom-right panel displays the assembly code being executed.

| Register | H | L |
|----------|------|----|
| AX | 12 | 3B |
| BX | 12 | 34 |
| CX | 00 | 0E |
| DX | 00 | 00 |
| CS | 0700 | |
| IP | 0109 | |
| SS | 0700 | |
| SP | FFFE | |
| BP | 0000 | |
| SI | 0000 | |
| DI | 0000 | |
| DS | 0700 | |

| Address | Hex | Comment |
|---------|-------------|---------|
| 07100: | A1 161 i | |
| 07101: | 0A 010 NEWL | |
| 07102: | 01 001 @ | |
| 07103: | 8B 139 i | |
| 07104: | 1E 030 ^ | |
| 07105: | 0C 012 % | |
| 07106: | 01 001 @ | |
| 07107: | 03 003 v | |
| 07108: | C3 195 | |
| 07109: | C3 195 | |
| 0710A: | 07 007 BEEP | |
| 0710B: | 00 000 NULL | |
| 0710C: | 34 052 4 | |
| 0710D: | 12 018 t | |
| 0710E: | 90 144 é | |
| 0710F: | 90 144 é | |
| 07110: | 90 144 é | |
| 07111: | 90 144 é | |
| 07112: | 90 144 é | |
| 07113: | 90 144 é | |
| 07114: | 90 144 é | |
| 07115: | 90 144 é | |


```

MOV AX, [0010Ah]
MOV BX, [0010Ch]
ADD AX, BX
RET
POP ES
ADD [SI], DH
ADC DL, [BX + SI] + 0090
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...

```

 original source co...

```

01  ORG 100h
02
03  MOV AX, var1
04  MOV BX, var2
05  ADD AX, BX
06
07  RET ; stops the program
08
09  VAR1 DW 7
10  var2 DW 1234h
11
12

```


ADD Instruction

```
.MODEL SMALL
.STACK 100H

.DATA SEGMENT
    A DW 2222H
    B DW 5555H

.CODE SEGMENT
MAIN PROC

START:
    MOV AX, @DATA
    MOV DS, AX

    MOV AX, A
    ADD AX, B

    MOV AX, 4C00H
    INT 21H

    MAIN ENDP
END MAIN
```

emulator: sum2_variable.exe_

file math debug view external virtual devices virtual drive help

Load reload **step back** single step run step delay ms: 0

registers

| | H | L |
|----|------|----|
| AX | 77 | 77 |
| BX | 00 | 00 |
| CX | 01 | 21 |
| DX | 00 | 00 |
| CS | 0721 | |
| IP | 000C | |
| SS | 0710 | |
| SP | 0100 | |
| BP | 0000 | |
| SI | 0000 | |
| DI | 0000 | |
| DS | 0720 | |

0721:000C

| | | |
|--------|--------|------|
| 07210: | B8 184 | ↓ |
| 07211: | 20 032 | SPA |
| 07212: | 07 007 | BEEP |
| 07213: | 8E 142 | ↕ |
| 07214: | D8 216 | ↕ |
| 07215: | A1 161 | ↓ |
| 07216: | 00 000 | NULL |
| 07217: | 00 000 | NULL |
| 07218: | 03 003 | ♥ |
| 07219: | 06 006 | ↑ |
| 0721A: | 02 002 | 0 |
| 0721B: | 00 000 | NULL |
| 0721C: | B8 184 | ↓ |
| 0721D: | 00 000 | NULL |
| 0721E: | 4C 076 | L |
| 0721F: | CD 205 | = |
| 07220: | 21 033 | ↑ |
| 07221: | 90 144 | É |
| 07222: | 90 144 | É |
| 07223: | 90 144 | É |
| 07224: | 90 144 | É |

0721:000C

| |
|------------------|
| MOV AX, 00720h |
| MOV DS, AX |
| MOV AX, [00000h] |
| ADD AX, [00002h] |
| MOV AX, 04C00h |
| INT 021h |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |

original source co...

```
.CODE SEGMENT
MAIN PROC

START:
    MOV AX, @DATA
    MOV DS, AX

    MOV AX, A
    ADD AX, B

    MOV AX, 4C00H
    INT 21H
```

ADD Instruction

```
.MODEL SMALL
.STACK 100H

.DATA SEGMENT
    A DB 09H
    B DB 09H

.CODE SEGMENT
MAIN PROC

START:
    MOV AX,@DATA
    MOV DS,AX

    MOV AL,A
    MOV BL,B
    ADD AL,BL

    MOV AX,4C00H
    INT 21H

MAIN ENDP
END MAIN
```

emulator: sum3_variable.exe_

file math debug view external virtual devices virtual drive help

Load reload step back single step run step delay ms: 0

| registers | H | L | |
|-----------|------|----|--|
| AX | 07 | 12 | |
| BX | 00 | 09 | |
| CX | 01 | 23 | |
| DX | 00 | 00 | |
| CS | 0721 | | |
| IP | 000E | | |
| SS | 0710 | | |
| SP | 0100 | | |
| BP | 0000 | | |
| SI | 0000 | | |
| DI | 0000 | | |

| 0721:000E | | | 0721:000E | | |
|-----------|----|-----|------------------|--|--|
| 07210: | B8 | 184 | MOV AX, 00720h | | |
| 07211: | 20 | 032 | MOV DS, AX | | |
| 07212: | 07 | 007 | MOV AL, [00000h] | | |
| 07213: | 8E | 142 | MOV BL, [00001h] | | |
| 07214: | D8 | 216 | ADD AL, BL | | |
| 07215: | A0 | 160 | MOV AX, 04C00h | | |
| 07216: | 00 | 000 | INT 021h | | |
| 07217: | 00 | 000 | NOP | | |
| 07218: | 8A | 138 | NOP | | |
| 07219: | 1E | 030 | NOP | | |
| 0721A: | 01 | 001 | NOP | | |
| 0721B: | 00 | 000 | NOP | | |
| 0721C: | 02 | 002 | NOP | | |
| 0721D: | C3 | 195 | NOP | | |
| 0721E: | B8 | 184 | NOP | | |
| 0721F: | 00 | 000 | NOP | | |
| 07220: | 4C | 076 | NOP | | |
| 07221: | CD | 205 | NOP | | |
| 07222: | 21 | 033 | NOP | | |
| 07223: | 90 | 144 | NOP | | |

original source co...

```
.CODE SEGMENT
MAIN PROC

START:
    MOV AX,@DATA
    MOV DS,AX

    MOV AL,A
    MOV BL,B
    ADD AL,BL

    MOV AX,4C00H
    INT 21H
```

ADD Instruction

```

.MODEL SMALL
.STACK 100H

.DATA SEGMENT
    A DW 1111H
    B DW 9999H

.CODE SEGMENT

MAIN PROC

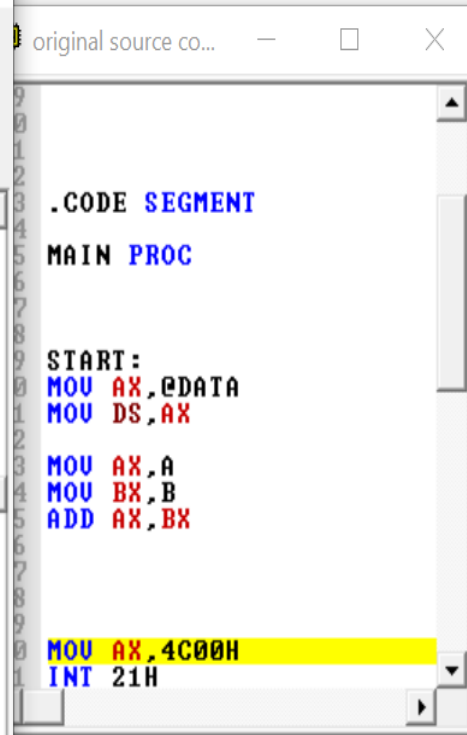
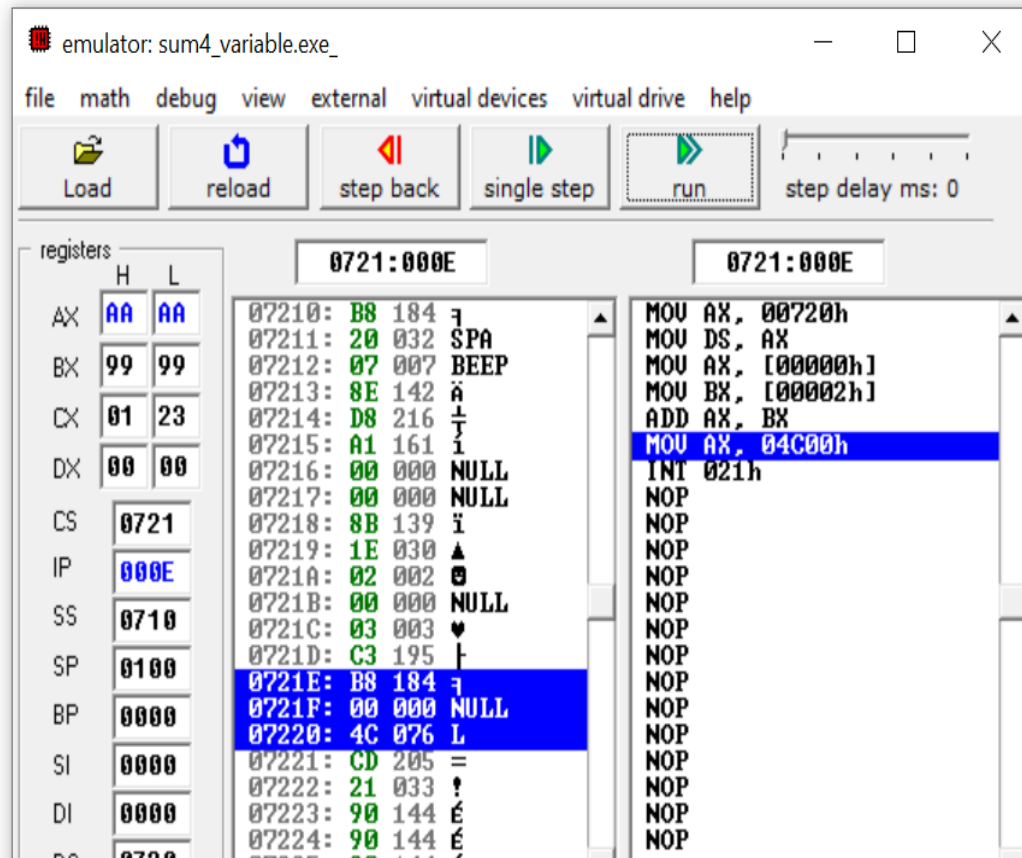
START:
    MOV AX, @DATA
    MOV DS, AX

    MOV AX, A
    MOV BX, B
    ADD AX, BX

    MOV AX, 4C00H
    INT 21H

MAIN ENDP
END MAIN

```



ADD Instruction

```
.MODEL SMALL
.STACK 100H
.DATA SEGMENT
    A DW 1111H
    B DB 99H

.CODE SEGMENT
MAIN PROC

START:
    MOV AX,@DATA
    MOV DS,AX

    MOV AX,A
    MOV BL,B
    ADD AX,BL

    MOV AX,4C00H
    INT 21H

    MAIN ENDP
END MAIN
```

assembler status

external view

there are errors. |

(25) wrong parameters: ADD AX,BL

(25) operands do not match: 16 bit and 8 bit register

SUB Instruction

The SUB instruction subtracts a source operand from a destination operand. The set of possible operands is the same as for the ADD and MOV instructions. The syntax is:

SUB *dest,source*

Example:

Example:

Here is a short code example that adds two 16-bit integers:

```
ORG 100h
```

```
MOV AX, var1
```

```
SUB AX,var2
```

```
RET ; stops the program.
```

```
VAR1 DW 2345H
```

```
var2 DW 1234h
```

SUB Instruction

```
ORG 100h
MOV AX, var1
SUB AX, var2
RET ; stops the program.
VAR1 DW 2345h
var2 DW 1234h
```

emulator: noname.com_

file math debug view external virtual devices virtual drive help

Load reload step back single step run step delay ms: 0

registers

| | H | L |
|----|------|----|
| AX | 11 | 11 |
| BX | 00 | 00 |
| CX | 00 | 0C |
| DX | 00 | 00 |
| CS | 0700 | |
| IP | 0107 | |
| SS | 0700 | |

0700:0107

| | | | |
|--------|----|-----|------|
| 07100: | A1 | 161 | i |
| 07101: | 08 | 008 | BACK |
| 07102: | 01 | 001 | ⊙ |
| 07103: | 2B | 043 | + |
| 07104: | 06 | 006 | ⬆ |
| 07105: | 0A | 010 | NEWL |
| 07106: | 01 | 001 | ⊙ |
| 07107: | C3 | 195 | ⌞ |
| 07108: | 45 | 069 | E |
| 07109: | 23 | 035 | # |
| 0710A: | 34 | 052 | 4 |
| 0710B: | 12 | 018 | ‡ |
| 0710C: | 90 | 144 | € |

0700:0107

| |
|---------------------------|
| MOV AX, [00108h] |
| SUB AX, [0010Ah] |
| RET |
| INC BP |
| AND SI, [SI] |
| ADC DL, [BX + SI] + 09090 |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |
| NOP |

original source co...

```
01 ORG 100h
02 MOV AX, var1
03 SUB AX, var2
04 RET ; stops the progr
05 VAR1 DW 2345h
06 var2 DW 1234h
07
08
```

NEG Instruction

The NEG (negate) instruction reverse s the sign of a number by converting the number to its two's complement. The following operands are permitted:

NEG *reg*

NEG *mem*

Recall that the two's complement of a number can be found by reversing all the bits in the destination operand and adding 1.

```
org 100h
```

```
MOV AX, 7DE0H
ADD AL, 10H      ; AL=0F0H, CF=0, SF=1, ZF=0, OF=0
SUB AH, 3        ; AH=7AH, CF=1, SF=0, ZF=0, OF=0      4
```

```
ret
```

emulator: noname.com_

file math debug view external virtual devices virtual drive help

Load reload step back single step run step delay ms: 0

registers

| | H | L |
|----|------|----|
| AX | 7A | F0 |
| BX | 00 | 00 |
| CX | 00 | 09 |
| DX | 00 | 00 |
| CS | 0700 | |
| IP | 0108 | |
| SS | 0700 | |

0700:0108

| Address | Hex | Dec | Symbol |
|---------|-----|-----|--------|
| 07100: | B8 | 184 | 7 |
| 07101: | E0 | 224 | α |
| 07102: | 7D | 125 | > |
| 07103: | 04 | 004 | ♦ |
| 07104: | 10 | 016 | ► |
| 07105: | 80 | 128 | ⌂ |
| 07106: | EC | 236 | se |
| 07107: | 03 | 003 | ♥ |
| 07108: | C3 | 195 | |
| 07109: | 90 | 144 | É |
| 0710A: | 90 | 144 | É |
| 0710B: | 90 | 144 | É |
| 0710C: | 90 | 144 | É |

MOV AX, 07DE0h
ADD AL, 010h
SUB AH, 03h
RET
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP

original source co...

```
01  
02 ; You may customize this  
03 ; The location of this t  
04  
05 org 100h  
06  
07 MOV AX, 7DE0H  
08 ADD AL, 10H      ; AL=0F  
09 SUB AH, 3        ; AH=7  
10  
11  
12 ret  
13  
14  
15  
16  
17  
18  
19
```


Experiment 01:

Addition of two Hexadecimal numbers in 8086 Microprocessor.

Assembly code :

```
MOV AX,1234H
```

```
MOV BX,5678H
```

```
ADD AX,BX
```

Experiment 02:

Logical operations (AND, OR, NOT, XOR, TEST) in 8086 Microprocessor.

Assembly code :

```
MOV AX,2053H ;----0010000001010011B
```

```
MOV BX,3167H ;----0011000101100111B
```

```
AND AX,BX ;
```

```
OR AX,BX ;
```

```
NOT AX ;
```

```
NOT BX ;
```

```
XOR AX,BX ;
```

```
TEST AX,BX
```

Experiment 03 :

Perform the logical operation of the following function: $(A+(B\oplus C)).D$

Also find the 2's complement of the result.

Where, $A=21H$; $B=11H$; $C=35H$; $D=57H$

Assembly code:

```
MOV AX,21H
```

```
MOV BX,11H
```

```
MOV CX,35H
```

```
MOV DX,57H
```

```
XOR BX,CX
```

```
OR AX,BX
```

```
AND AX,DX
```

```
NEG AX
```

Experiment 03 :

Perform the logical operation of the following function: $(A+(B\oplus C)).D$

Also find the 2's complement of the result.

Where, $A=21H$; $B=11H$; $C=35H$; $D=57H$

Assembly code:

```
MOV AX,21H
```

```
MOV BX,11H
```

```
MOV CX,35H
```

```
MOV DX,57H
```

```
XOR BX,CX
```

```
OR AX,BX
```

```
AND AX,DX
```

```
NEG AX
```