CSE 331L: Microprocessor Interfacing & Embedded System Lab

Faculty: DR. DIHAN MD. NURUDDIN HASAN

Instructor: Moin Shahriyar

EEE 332/ CSE 331
Lab 2

Topic: Variables, Basic Arithmetic Operations, Array

Topics to be covered in class today:

- Creating Variables
- Basic Arithmetic Operations (addition, subtraction, multiplication, division)
- Creating Arrays
- Create Constants
- Introduction to INC, DEC, LEA instruction
- Learn how to access Memory.

Creating Variable:

Syntax for a variable declaration:

"name" DB "value"

e.g. a DB 9

name DW value

e.g. a DW 9

DB - stands for Define Byte. (8 bit) define byte

DW - stands for Define Word. (16 bit) define word

name - can be any letter or digit combination, though it should start with a letter. It's possible to declare unnamed variables by not specifying the name (this variable will have an address but no name).

value - can be any numeric value in any supported numbering system (hexadecimal, binary, or decimal), or "?" symbol for variables that are not initialized.

**Variable must declare after return.

Creating Constants:

Constants are just like variables, but they exist only until your program is compiled (assembled). After definition of a constant its value cannot be changed. To define constants EQU directive is used: (always 16 bit)

name EQU < any expression >

For example:

- k EQU 5
- MOV AX, k

Creating Arrays:

Arrays can be seen as chains of variables. A text string is an example of a byte array; each character is presented as an ASCII code value (0-255).

Here are some array definition examples:

- a DB 48h, 65h, 6Ch, 6Ch, 6Fh, 00h
- b DB 'Hello', 0

You can access the value of any element in array using square brackets, for example:

- MOV AL, a [3]

You can also use any of the memory index registers BX, SI, DI, BP, for example:

- MOV SI, 3
- MOV AL, a[SI]

If you need to declare a large array you can use DUP operator.

The syntax for DUP:

number DUP (value(s))

number - number of duplicate to make (any constant value).

value - expression that DUP will duplicate.

for example:

- c DB 5 DUP (9)

an alternative way of declaring is:

- c DB 9, 9, 9, 9, 9

one more example:

- d DB 5 DUP (1, 2)

an alternative way of declaring is:

- d DB 1, 2, 1, 2, 1, 2, 1, 2, 1, 2

Memory Access:

To access memory we can use these four registers: BX, SI, DI, BP. Combining these registers inside [ ] symbols, we can get different memory locations.

[BX + SI] [BX + DI] [BP + SI] [BP + DI]	[SI] [DI] d16 (variable offset only) [BX]	[BX + SI + d8] [BX + DI + d8] [BP + SI + d8] [BP + DI + d8]
[SI + d8] [DI + d8] [BP + d8] [BX + d8]	[BX + SI + d16] [BX + DI + d16] [BP + SI + d16] [BP + DI + d16]	[SI + d16] [DI + d16] [BP + d16] [BX + d16]

Displacement can be an immediate value or offset of a variable, or even both. if there are several values, assembler evaluates all values and calculates a single immediate value.

Displacement can be inside or outside of the [] symbols, assembler generates the same machine code for both ways.

Displacement is a signed value, so it can be both positive or negative.

Instructions

Instruction	Operands	Description
INC	REG MEM	Increment. Algorithm: operand = operand + 1 Example: MOV AL, 4 INC AL; AL = 5 RET
DEC	REG MEM	Decrement. Algorithm: operand = operand - 1 Example: MOV AL,86 DEC AL; AL=85 RET

LEA	REG,MEM	Load Effective Address.
		
		Algorithm:
		
		REG = address of memory (offset)
		
		Example:
		
		MOV BX, 35h
		MOV DI, 12h
		LEA SI, [BX+DI]

ADD	REG, memory memory, REG REG, REG memory, immediate REG, immediate	Add. Algorithm:
		operand1 = operand1 + operand2
		
		Example:
		
		MOV AL, 5; [AL = 5]
		ADD AL, -3; [AL = 2]
		RET

SUB	REG, memory memory, REG REG, REG memory, immediate REG, immediate	Subtract. Algorithm:
		operand1 = operand1 - operand2
		
		Example:
		
		MOV AL, 5
		SUB AL, 1 ; AL = 4
		RET

MUL	REG memory	Unsigned multiply. Algorithm: when operand is a byte: AX = AL * operand.
		when operand is a word:
		(DX AX) = AX * operand.
		
		Example:
		
		MOV AL, 200 ; [AL = 0C8h]
		MOV BL, 4;
		MUL BL ; [AX = 0320h (800)]
		RET

DIV	REG memory	Unsigned divide. Algorithm:
		when operand is a byte: AL = AX / operand AH = remainder (modulus)
		
		when operand is a word: AX = (DX AX) / operand DX = remainder (modulus)
		
		Example:
		MOV AX, 203; [AX = 00CBh]
		MOV BL, 4;
		DIV BL; [AL = 50 (32h), AH = 3]
		RET