# 16.317 Microprocessor I, Spring 2007
# Lab 2: Assembling and executing instructions with DEBUG software
# Due on 03/02/2007, 12:30pm EST

**Reference:**
Walter A. Triebel and Avtar Singh, Lab Manual to Accompany the 8088 and 8086 Microprocessors, Prentice Hall, ISBN 0-13-012843-0.

## Objective

Learn how to:

1. Assemble instructions into the memory of PC.
2. Execute an instruction to determine the operation it performs.
3. Verify the operation of data transfer and arithmetic instructions.

## Part 1: Practice 8086 Emulator

You need to practice a 8086 Emulator, emu8086, by reproducing the examples in the text book. It is helpful to read carefully the explanations along with the examples. Specifically, you need to practice the following examples:

**Loading, verifying and saving machine code**
Example 4.24 till page 130 (page 127 - 130)

**Executing instructions and tracing programs**
Example 4.26 (page 136)

To learn how to use emu8086 tool, please refer to the following web site.

**http://www.emu8086.com/assembly_language_tutorial_assembler_reference/reference.html**

## Part 2: Assignment

Note: You need to use emu8086 to do the following assignments.

**You need to show your screen print-out to the TA to check off. Make sure you're your print-out captures all the steps.**
**Answer ALL the questions in your report.**

### *Data transfer instructions I*

| | |
|---|---|
| Step 1 | Using emu8086 to assemble the instructions<br>`(a) MOV AX,BX`<br>`(b) MOV AX,AAAA`<br>`(c) MOV AX,[BX]`<br>`(d) MOV AX,[4]`<br>`(e) MOV AX,[BX+SI]`<br>`(f) MOV AX,[SI+4]` |

| | |
|---|---|
| | `(g) MOV AX,[BX+SI+4]` |
| Step 2 | Initializing the internal registers of the 80x86 as follows:<br>`(AX) = 0000H`<br>`(BX) = 0001H`<br>`(CX) = 0002H`<br>`(DX) = 0003H`<br>`(SI) = 0010H`<br>`(DI) = 0020H`<br>`(BP) = 0030H`<br>Verify the initialization by displaying the new content of registers |
| Step 3 | Fill all memory locations in the range DS:00 through DS:1F with 00H and the initialize the word storage location that follow:<br>`(DS:0001H) = BBBBH`<br>`(DS:0004H) = CCCCH`<br>`(DS:0011H) = DDDDH`<br>`(DS:0014H) = EEEEH`<br>`(DS:0016H) = FFFFH` |
| Step 4 | Trace the execution of the instructions (a) through (f). Explain the execution of **each** instruction, including addressing mode, physical address for memory addressing mode, value in AX. |

### *Data transfer instructions II*

| | |
|---|---|
| Step 1 | Assemble the instruction MOV SI, [0ABC] to memory at address CS:100 and verify loading of the instruction. How many bytes does the instruction take up? |
| Step 2 | Initialize the word of memory starting at DS:0ABC with the value FFFFH |
| Step 3 | Clear the SI register, verify by displaying its content. |
| Step 4 | Trace the execution. Describe the operation performed by the instruction |
| Step 5 | Assemble the instruction MOV WORD PTR [SI], ABCD into memory at address CS:100 and then verify loading of the instruction. How many bytes does it take? |
| Step 6 | Initialize SI register with the value 0ABCH |
| Step 7 | Clear the word of memory starting at DS:0ABC |
| Step 8 | Trace the execution. Describe the operation performed by the instruction |

### *Arithmetic instructions*

| | |
|---|---|
| Step 1 | Assemble the instruction ADC AX, [0ABC] to memory at address CS:100 and verify loading of the instruction. How many bytes does the instruction take up? |
| Step 2 | Initialize the word of memory starting at DS:0ABC with the value FFFFH |
| Step 3 | Initialize AX with the value 0001H. Verify by displaying register contents. |
| Step 4 | Clear the carry flag |
| Step 5 | Trace the execution. Describe the operation performed by the instruction |
| Step 6 | Does a carry occur? |

### *Flag-control instructions*

| | |
|---|---|
| Step 1 | Assemble the instruction sequence<br>`LAHF`<br>`MOV BH, AH`<br>`AND BH, 1FH`<br>`AND AH, 0E0H`<br>`MOV [200H], BH`<br>`SAHF`<br>Into memory at address CS:100 and then verify loading of the instruction. How many bytes do they take? |
| Step 2 | Initialize the byte of memory starting at DS:200 with the value 00H |
| Step 3 | Clear register AX and BX |
| Step 4 | Display the current state of flags, make sure the status flags equal NG, ZR, AC, PE and CY |
| Step 5 | Trace the execution. Describe the operation performed by each instruction.<br>What value is read out of the flags register?<br>What value is saved in memory?<br>What value is reloaded into flags register? |

### *Compare instructions*

| | |
|---|---|
| Step 1 | Assemble the instruction sequence<br>`MOV BX, 1111H`<br>`AND AX, 0BBBBH`<br>`CMP BX, AX`<br>Into memory at address CS:100 and then verify loading of the instruction. How many bytes do they take? |
| Step 2 | Clear register AX and BX. |
| Step 3 | Display the current state of flags |
| Step 4 | Trace the execution. Describe the operation performed by each instruction.<br>What value is read out of the flags register?<br>What value is saved in memory?<br>What value is reloaded into flags register?<br>What are the status flags before and after the compare instruction was executed? |

### *Jump instructions*

You need to download two files (L5P3.LST and L5P3.EXE) from the course webpage at
http://faculty.uml.edu/yluo/Teaching/MicroprocessorI/**umlrocks/L5P3.LST**
http://faculty.uml.edu/yluo/Teaching/MicroprocessorI/**umlrocks/L5P3.EXE**

| | |
|---|---|
| Step 1 | Download the files in one of your file folders |
| Step 2 | open the source listing in file L5P3.LST.<br>What is the starting address offset from CS: for the first instruction (PUSH DS) and the last instruction (RET)? |
| Step 3 | Load the run module L5P3.EXE with emu8086 |
| Step 4 | Verify loading of the program by unassembling the contents of the current code segment for the offset range found in Step 2 |

| | |
|---|---|
| Step 5 | Execute the program according to the instructions that follow<br>a. GO from address CS:00 to CS:5<br>b. Load the number whose factorial is to be calculated (N=3) into register DX<br>c. Clear the memory storage location DS:0000 for the value of factorial (FACT)<br>d. GO from address CS:5 to CS:10, what is the state of the zero flag?<br>e. Execute from the JZ instruction with a TRACE command, Was the jump taken?<br>f. GO from address CS:12 to CS:16, What is the current value of AL?<br>g. Execute the JMP instruction with a TRACE command<br>Was the jump taken? What is the address of the next instruction to be executed?<br>h. GO from address CS:E to CS:10. What is the state of the zero flag?<br>i. Execute the JZ instruction with a TRACE command. Was the jump taken?<br>j. GO from address CS:12 to CS:16, What is the current value of AL?<br>k. Execute the JMP instruction with a TRACE command<br>Was the jump taken? What is the address of the next instruction to be executed?<br>l. GO from address CS:E to CS:10. What is the state of the zero flag?<br>m. Execute the JZ instruction with a TRACE command. Was the jump taken?<br>n. GO from address CS:12 to CS:16, What is the current value of AL?<br>o. Execute the JMP instruction with a TRACE command<br>Was the jump taken? What is the address of the next instruction to be executed?<br>p. GO from address CS:E to CS:10. What is the state of the zero flag?<br>q. Execute the JZ instruction with a TRACE command. Was the jump taken? What is the instruction to be executed next?<br>r. GO to CS:1B. What is the final value in AL? At what address is the value in AL stored in memory as FACT?<br>s. Display the value stored for FACT in memory |
| Step 6 | Quit from emu8086. |

***Subroutine instructions***

You need to download one file (L5P4.EXE) from the course webpage at http://faculty.uml.edu/yluo/Teaching/MicroprocessorI/**umlrocks/L5P4.EXE**

| | |
|---|---|
| Step 1 | Load the file L5P4.EXE with emu8086 |
| Step 2 | Verify the loading of the program by unassembling the contents of the current code segment. |
| Step 3 | Execute the instructions as follows.<br>a. Single step from address CS:00 to CS:5. What instruction is to be executed next? |

| | |
|---|---|
| | b. Load the numbers that follow for use by the arithmetic subroutine.<br>`(AX) = -32 = FFE0H`<br>`(BX) = 27  = 001BH`<br>`(CX) = 10  = 000AH`<br>`(DX) = 200 = 00C8H`<br>c. Execute the call instruction with a TRACE command. What instruction is to be executed next?<br>d. Single step to address CS:10. What is the sum in DX?<br>e. Check the value of the last word pushed to the stack<br>f. Run the program to completion. What is the final value in DX? How did the contents of DX become this value? |
| Step 4 | Exit emu8086. |

**Check off**

Demonstrate the files you saved (in example 4.24) to the TA.
Get TA's signature on the screen print-out you obtained.
Attach your screen print-out to your report.

**Report format**

Your report needs to follow the format below.

**Lab # and title:**
**Student Name:**
**Partner's Name:**

**Lab Purpose:**
<It is usually the objective of the lab.>

**Lab Content:**
< Answer the questions in lab specification. Describe what you do in the lab, e.g. what commands you practiced. It has to be at least one page with 11pt font size. Try to organize and summarize the lab in itemized lists.>

**Difficulties:**
< state what difficulties you encountered in the lab and how you managed to solve it. If not, what have you tried? >

**Conclusion and Suggestions:**

**Grading**

Student Name ___________________ Student ID _____________________

## Lab Two Rubrics

***Partners can share screen capture or program print-out, however, each student* MUST *submit his/her own report. Duplicated reports (including both parties) are considered cheating, which result in a ZERO in the lab, reduction in letter grade for the course, grade of F for the course, and/or university administrative penalties*.**

| Item | Criteria | Full Points | Actual Points |
|---|---|---|---|
| Check off | Check off with TA | 40 | |
| Data transfer I/II | Print-out or screen capture of the instruction, Answer questions correctly | 10 | |
| Arithmetic | Print-out or screen capture of the instruction, Answer questions correctly | 10 | |
| Flag control | Print-out or screen capture of the instruction, Answer questions correctly | 10 | |
| Compare | Print-out or screen capture of the instruction, Answer questions correctly | 10 | |
| Jump | Print-out or screen capture of the instruction, Answer questions correctly | 10 | |
| Subroutine | Print-out or screen capture of the instruction, Answer questions correctly | 10 | |
| TOTAL | | 100 | |