Laboratory 3

The asm12 Assembler

Due Date: Beginning of Week 5 Lab

Introduction

In Lab 2, you wrote an assembly program on paper and converted it manually to machine code using the instruction set. Next, you had to type in the bytes of the machine code program by hand. These are tedious tasks that are easily automated using the asm12 assembler and Debug12's built-in loader program.

Assignment

Lab3prg1.asm is available under the "Labs" tab on Blackboard. Download it to your USB thumb drive and then open it in MiniIDE. This file contains a slightly modified version of the summation program from Lecture 11 using a one-byte length and unsigned numbers.

- Assemble the program by clicking on the "Build Current" button in the menu bar. The Output window should state that 0 error(s) were found.
- To download the machine code, type **LOAD** and hit return in the terminal window. The Debug12 program will not display the prompt since it is now awaiting a file transfer.
- Click on the "Download" button in the menu bar. In the dialog box, select the Lab3prg1.s19 file (which was created when you hit Build Current). The Debug12 program should now return the prompt.
- Open the Lab3prg1.lst file in MiniIDE editor (you will need to change the file extension filter at the bottom of the dialog window).
- Just to verify what happened, do a memory display at $2000 to display all of the program's machine code and compare it to the machine code shown in the listing file.
  Question 1: Based on the information in the listing file, where is the starting address of the array of numbers to be added supplied to the program?
  Question 2: Based on the information in the listing file, where is the length of the array of numbers to be added supplied to the program?
- Enter a starting address of $3000 and a length of $05 in the appropriate locations.
- Enter the values $3A, $5E, $FE, $8B, and $56 into the array starting at $3000. These should sound familiar. They are the same values added in Lab 1.

- Run the program.
  Question 3: What sum does the program give, and what address is it returned it?
  Question 4: Did overflow occur?
- Place a breakpoint at the BEQ Done instruction. Note that you will need to examine the listing file to get the memory address of BEQ.
- Complete a **breakpoint trace** by running the program from the beginning and proceeding from the breakpoint until the end of the program is reached. Include the registers affected and the four main CCR bits.
- Modify the program given so that it handles two-byte lengths and save it as Lab3prg2.asm.
- Test your program on the same data as before to verify that it works.
- Fill $3000 to $33FF with $80 and supply a length of $0400 bytes. Run the program again.
  Question 5: What sum does the program return, and did overflow occur?
- Write a program in assembly that sums an array of 4-byte numbers. Your program must meet the following requirements.
  a. The address of the array is supplied in location $1000.
  b. The one-byte length of the array is supplied in location $1004.
  c. The four-byte answer must be returned in addresses $1010-$1013.
  d. Address $1014 should be $FF if unsigned overflow occurs and $00 otherwise.
  e. Address $1015 should be $FF if signed overflow occurs and $00 otherwise.
- Enter your program into the Dragon12+ board and test your program. When it is working, demonstrate it to the instructor. **Important**: Only addresses $1000 to $35FF are available for use in lab for both program and data.

  Hints:

  - Since the answer is returned in a known location, you can use extended addressing as in the first program in Lecture 11 to access the locations.
  - The two numbers being added may be pointed to using X and Y, and the indexes may be used to access the different parts of the numbers, similar to the Big-Endian to Little-Endian conversion program in Lecture 10.
  - Use Lab3prg1.asm as a template, and be sure to update and add appropriate comments.
  - When in doubt, refer to the last three sample programs covered in class.

What to Demonstrate/Submit

- Demonstrate the program to add 4-byte numbers
- Typed answers to all questions, including the breakpoint trace
- Email the listing file and question answers to dfoster@kettering.edu.