EE 308 – LAB 3

Assembly Language Programming

Introduction and Objectives

The purpose of this lab is to write a few assembly language programs and test them on the simulator and on your EVBU.

Pre-Lab

Make sure you have the programs clearly thought out and written down before you come to lab. You should put all your code starting at memory location `0x0800`. You are encouraged to bring the programs in on disk.

The Lab

As in last week's lab you will write some programs in assembly language and run the programs on the HC12 simulator and the EVBU. To make sure your programs work you will sometimes use D-Bug 12 op codes as your input data. There is one problem when running programs on the simulator – D-Bug 12 code which is present in the EVBU is not on the simulator. We can fix this problem by loading D-Bug 12 into the simulator. To do this, start the `ZAP` simulator, and load the file `n:\ee308\dbug12.h12`

Write and run the following programs:

1. Write a program which writes a `0xff` to address `0x0002`, and then decrements address `0x0000` indefinitely. (Note: this is about a four-line program.)

   Test your program both on the simulator and the EVBU. Trace through the program on the simulator and observe what happens to the data at address `0x0000`.

   Note: This should cause the pins labeled PA0-7 on the HC12 to toggle. Use a logic probe to verify that this is happening.

2. Write a program to swap the last element of an array with the first element, the next-to-last element with the second element, etc. The array should have `0x20` eight-bit numbers and should start at `0x0900`. (This is similar to a problem from the homework due Feb. 4.)

   Check that your program works both on the simulator and the EVBU. Us the following data for your test:

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
001	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F

3. Write a program that puts the smallest one-byte signed number from memory locations `0xFCE0` through `0xFCFF` into memory location `0x0900`, and the largest one-byte signed number in that memory range into memory location `$0901`. (Note: The is similar to a problem on the homework due Feb. 4.) Test your program on the simulator and EVBU.

Loading Programs into EEPROM

You can load programs into the on-chip EEPROM. When loaded in EEPROM your program will remain on your HC12 even after turning off power. Before loading your program into EEPROM, you must re-assemble it after modifying your program to start your code at address `$0D00`. Also, modify your `LKF` link file to start the `.text` section at address `$0D00`

The details of how to download your program into EEPROM are given in the **Universal Evaluation Board User's Manual** in Appendix E, and are summarized here, with a description of how to make this work with our computers.

- Modify your program to put the code at address `$0D00`. Add this instruction as the first instruction in your program:

  ```
          clr   $16
  ```

  Leave the data at address `$0900`. Modify your `lkf` link file to put the `.text` section at address `$0D00`.
- Make sure you are talking to your EVBU with a Windows 3.11 Terminal. (The Windows 95/98/NT Hyperterminal will not work for this).
- Move jumpers `W3` and `W4` to their other position, then reset your board.
- On the Windows 3.11 Terminal menu bar, select `Settings`, `Text Transfer`, `Line at a Time`, `Wait for Prompt String`, and type in a "`*`" as your prompt string (without the quotes).
- From the HC12 command prompt, type `L` (for `LoadEE`). Go to the `Transfers` menu, choose `Send Text File`, and send your `S19` file which was assembled to load the program into EEPROM.
- After the HC12 responds `Programmed`, hit the `STOP` icon at the bottom of the Terminal window, then change your Terminal `Text Transfer` settings back to `Standard Flow Control`.
- Move your `W3` and `W4` jumpers back to their original position, and reset your EVBU.
- At the HC12 prompt, examine the contents of memory starting at location `$0D00` and verify that your program has been loaded into EEPROM.

After the code is in the EEPROM, you can run it in one of two ways – 1) From `D-Bug 12`, set the program counter to `$0D00` and `'g'` (or `'g 0D00'`); or 2) Move jumper W3 to its other position, and reset or power-cycle the EVBU. (Note that if you run your program using method 2, you cannot easily re-enter `D-Bug 12` to check the results of your program. We will use method 2 in later labs where we won't need to re-enter `D-Bug 12` after running a program.)

6. Load your last program from Part 3 (which finds the smallest and largest signed numbers) into EEPROM. Run it by giving the `G 0D00` command from `D-Bug 12`. Verify that the performance is the same as when you ran it from RAM.
7. Turn off power to your EVBU. Turn power back on, and rerun your program by giving the `g 0D00` command from `D-Bug 12`. Verify that your program stayed in EEPROM.