CS 310H **Computer Organization and Programming** **Spring 2010**
Professor Fussell **Due: Apr. 1, 2010**

Assignment #7

Instructions: The assignment is due on the date shown above. Tips to remember: give the assignments to your TA in section, remember your name, section number, TA name, and assignment number (5 points). Also, make sure your assignment is neat, stapled, and is entirely **your own work**.

1. Using the syntax presented in class, write the RTL sequence of steps for the LDI and STI instructions of the LC-3.

2. In this problem you will write a simple calculator program. The specifications are as follows:

   Prompt the user for a decimal number. You may assume that the number will be one or two digits and will be between 0 and 49.

   Prompt the user for a second number of the same type.

   Print the result (in decimal) of adding the two numbers together.

   Exit.

   While this looks like a simple program, implementing it in assembly code will require you to solve several problems, e.g.:

   - How to convert a two ASCII characters representing a single decimal number into a binary number.
   - How to convert a binary number into two ASCII characters representing a single decimal number.

   You may use any of the built-in TRAP functions of the LC-3. You may also need to figure out a creative way to do multiplies and divides to help in the conversion. You can exploit the fact that the multiplies and divides you might need are by a constant.

   (a) Turn in a printout of your code. Don't forget to fill in the header with your name and section number.

   (b) Your program should run on the UNIX version of the LC-3 simulator, and should be submitted using the turnin command, where the source file is named hw7-2.asm.

   (c) In the comments section after the program header, explain how your code works.

3. Using the template provided in hw7-3.asm (available on-line), write a program that takes an input value in R0, reverses the order of the bits, and provides the result in R1. For example, if the input contains 1101101100001111, the output should be 1111000011011011. Here are a couple of hints:

   - Recall that adding a number to itself has the effect of shifting the number up by one position.
   - You may decide that you need bit masks and use of the AND and NOT instructions to manipulate the bits.

We recommend designing the algorithm and planning the program before beginning to code - this general strategy will save you many headaches now and later as you begin to write more complex programs.

Make sure your program works for multiple different input bit patterns. Please turn in the following:

(a) A printout of your program, well documented, using the assembly code guidelines available on the course web page.

(b) Ensure that your program works on the Linux version of the lc3 simulator and submit using the turnin command. Make sure that your program is named hw7-3.asm.