Spring 2010 ECE 190

Machine Problem 1: ToUpper Function
Due: Wednesday 2/17 at 5:00pm

Description:
This MP will take in an address and length for a source string in memory and a destination address. The string is to be copied to the destination, except all lowercase letters must be replaced with their uppercase equivalents. The following labels contain the input source address and length, and output destination address. Assume each character uses 16 bits (the upper 8 bits are 0).

Provided addresses:

0x4000 - Address of start of source string
0x4001 - Size of string
0x4002 - Address of start of destination string

Specifics:
- Your program must be named mp1.bin.
- Your program must use a loop (i.e. there must be a conditional branch).
- Your code must be written in LC-3 machine language.
- Your code must begin at memory location 0x3000.
- Your code must be well commented and you must provide an introductory paragraph and a register table at the top of your file (Style Requirements).

Example:
Suppose the memory looks like the following to begin with;

Address	Data
x4000	x5000
x4001	3
x4002	x6000
...	...
x5000	x74 (‘t’)
x5001	x68 (‘h’)
x5002	x45 (‘E’)

After your program completes, the following should be stored at memory x6000;

Address	Data
x6000	x54 (‘T’)
x6001	x48 (‘H’)
x6002	x45 (‘E’)

Note that we are using a full 16 bits for each character, so every character in the string has its own address. As another example, consider the following string stored at a memory location pointed to by x4000:

The 2 quIck bROWN foxes jumped OveR the lazy Dog.

After the program is run, the address stored pointed to by x4002 should contain:

THE 2 QUICK BROWN FOXES JUMPED OVER THE LAZY DOG.

Ensure that letters already capitalized and non-alphabetic characters are not modified (you can’t make a period or number uppercase). You must modify only those letters which are lowercase.

Hint:
The difference in value between an uppercase and lowercase letter is 0x20. Also, the only letters to be modified lie in the range [a-z].

Remember, you can store any data you want at addresses after the halt instruction We recommend you make liberal use of this ability, particularly for precomputed negative values.

Make sure you take a close look at an ASCII table, to figure out what values to compare against. An ASCII table is available at http://www.asciitable.com/.

Style Requirements:
MP1 must be written in LC-3 machine language for any credit to be received (i.e. each instruction is a 16-bit binary word). Your code must contain less than 50 instructions (this does not include data or comment lines, just instructions) and you must use a loop in your solution to receive credit. Your code must be well-commented (every line for this MP because it is in binary). Comments begin with a semicolon. They should explain why something is done instead of restating what the ASM/RTL already says. In general, you

should use the commenting style provided in the Student Manual (see Page 35 for an example). For readability reasons, you must not have more than 80 characters on a single line. If you are writing a long comment and exceed 80 characters, continue your comment on the next line. In addition, you must use spaces to separate instruction fields in your binary encoding (e.g. 0010 001 000000100, not 0010001000000100). This will improve the readability of your code. Include a paragraph at the top of your code explaining the purpose of your program and your approach. Below that paragraph, create a table explaining the role of each register used by your program. Be sure to make the paragraph and the table a comment using semicolons.

Tools: You will use the “lc3convert” command to convert your machine code to an “obj” file, and the LC-3 simulator in order to execute and test the program that you write for this MP. You should use a text editor on a Linux machine (vi, emacs, pico, etc.) in order to code your program. Your code must work in the command line LC-3 simulator on the Linux EWS machines in Everitt lab to receive credit. Important: The LC-3 tools have a bug that prevents them from loading the last line of machine program correctly. To bypass this, create a blank line at the end of your mp1.bin file.

Hand In: Turning in your program is done electronically via the ECE 190 handin script. You must name your file `mp1.bin'; we will NOT grade files with any other name. To hand in the code, enter the ECE 190 work area by typing `ece190' from a Linux EWS machine. Next change to the directory containing your code and type: “handin --MP 1 mp1.bin”.
You may hand in as many times as you like, but only the last submission will be graded.

Grading:
The grade script will only inspect the address stored at 0x4002. If you do not store the output to the correct address, you will not get any credit for your answers!

Comments and style - 40%
Correctness - 60%