CMPE 12L Lab 3 - Winter 2015
Writing Assembly and Generating Machine Code

Prof. HP Dommel
Due: February 9, 2015 5pm
75 Points (15 Report, 60 Work)

Prerequisites

- Read through this **entire** lab assignment.
- Read the textbook chapter 5 and 6.
- Review the lecture notes on the LC3 architecture.

Overview

You are asked to make a program in LC3 assembly language that takes as input 2 single digit decimal numbers from the user. The program will then perform a subtraction, multiplication, and division of the two inputs. The results will then be stored in the following memory locations:

```
x3100 for subtraction result
x3101 for multiplication result
x3102 for division result
x3103 for the remainder of division
```

In this lab you will use the LC3 simulator to verify that your program works correctly. This is a processor simulator that implements a full instruction set architecture (ISA) and uses an ALU (with many more operations) like the one that you designed in lab 2.

Tutor/TA Review

Your lab tutor/TA will cover the following items in the first portion of the first lab:

- How to run LC3Edit and Simulate
- The GUI of Simulate
- How to step through a program in Simulate
- Explain how echo.asm works

Program Behavior

Your program should behave as follows:

- Display a greeting
- Prompt the user to input a single digit number
- Get user input using GETC. This will become your first operand.
- Prompt the user to input another single digit number
- Get user input using GETC. This will become your second operand.
- Perform subtraction, multiplication, and division, store the result to the correct memory locations, and output to the terminal. Assume that the user will only input valid values (0 to 9) and that the 2nd operand is less than the first.
- Go back to (1)

Additional Requirements

- Starting location of your code is to be 0x3000
- Subtraction, multiplication, and division is to be implemented as subroutines

The LC-3 Editor

- Start the LC-3 editor, LC3Edit.exe. You can find it on the PCServer in $F:\backslash ClassFolders\backslash ComputerEngineering\backslash CE12L\backslash LC3$.
- Download the echo.asm and open it using the LC-3 editor. Assemble the file by clicking on the 'asm' button.
- Check for errors. If there were problems converting, correct the errors now.
- Upon successful conversion, you should have generated the object file echo.obj. This can be run in the simulator.

The LC-3 Simulator

For the next two lab assignments, you will use Simulate to simulate an LC-3 processor.

- Start the LC-3 simulator, Simulate.exe. You can find it on the PCServer in $F:\backslash ClassFolders\backslash ComputerEngineering\backslash CE12L\backslash LC3$.
- In the simulator, open your object (.obj) file. You can open it with the File->Load Program dialog, or with the toolbar button as shown in Figure 1.

- Now, you are ready to run the program. You can use the Run Program button in the toolbar (see Figure 2) to run your program all the way through. The screenshot in Figure 3 shows another program having been run once through.
- You can rerun your program by selecting a new starting address. If you just hit Run without resetting the starting address, the simulator will yell at you! Reset the starting address by setting the "Jump to" field to x3000, and set the program counter (PC) to that address by pushing the button with the blue arrow. Recall that x3000 is the starting address of the program (the first two bytes in the object file are x3000). Also, you can use the step-into and step-over buttons (next to the Run button) your program to see each line of code being executed one at a time. See Figure 4.
- Every time you make changes to the machine code, you must re-convert it, re-load it in the simulator, and run it, verifying that it works as expected.

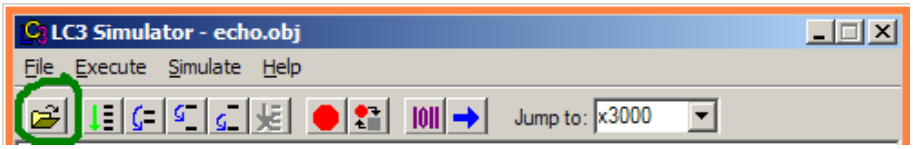

Figure 1: Use this toolbar button to open an object in the simulator

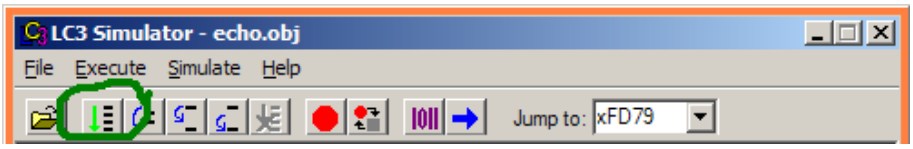

Figure 2: Use this toolbar button to run an object in the simulator.

Lab Submission

Your lab will be submitted via your eCommons account. Please log in to eCommons using your UCSC account and attach the following files to your "Lab3" assignment submission:

- lab3_[username].bin
- lab3_report_[username].pdf

Note that the final report must be submitted in PDF format. Make sure to confirm that your assignment is SAVED and SUBMITTED before the deadline. You may resubmit your assignment an unlimited number of times up until the due date.

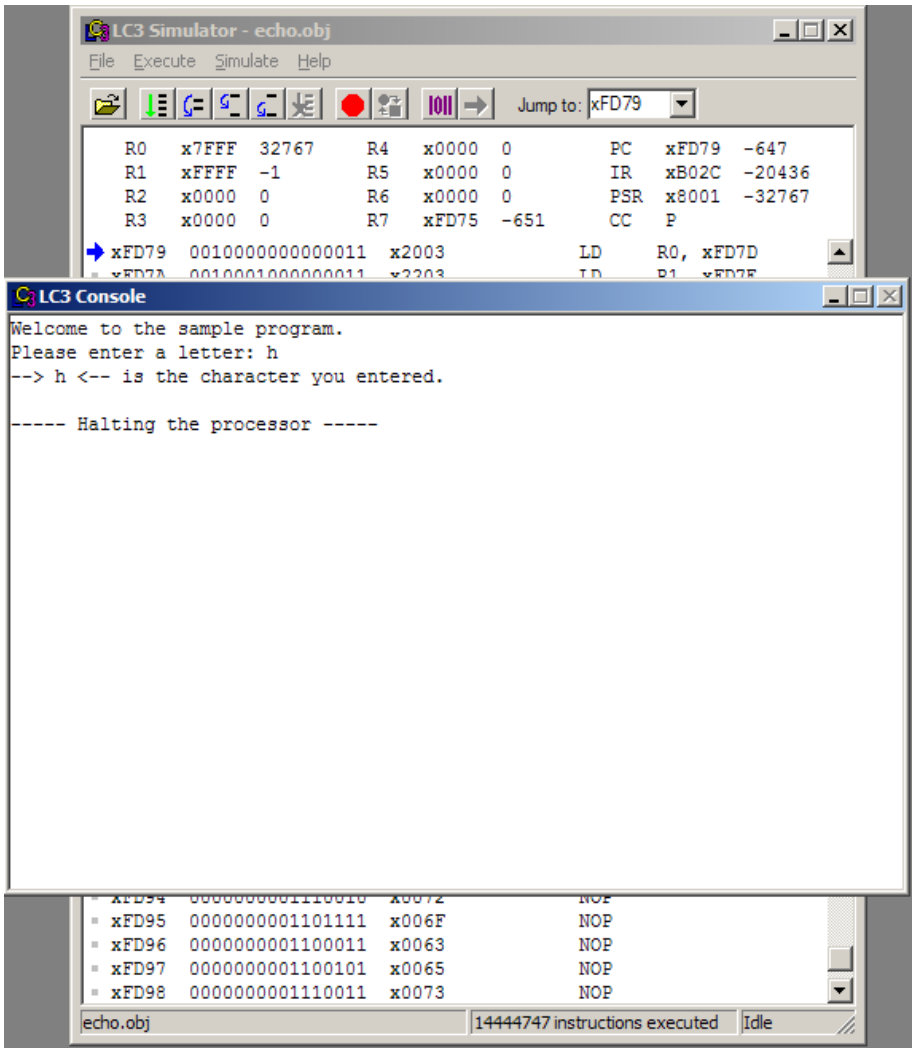

Figure 3: The sample file (echo.asm) has been run in the simulator.

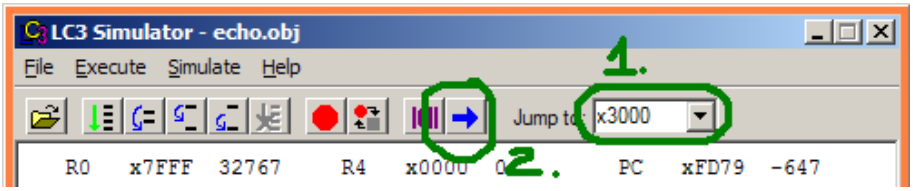

Figure 4: Jump to the starting address (defined by .orig 0x3000 in the program), and set the PC to that value, to run the program again.

Grading template

This is a suggested grading rubric. It is also a good general guideline before submitting your lab to check off these points.

☐ (3 pts) Does it prompt the user to enter 2 inputs?

☐ (2 pts) Does your program accept 2 1-digit decimal inputs?

☐ (5 pts) Character to decimal conversion is correct

☐ (5 pts) Subtraction is correct

☐ (10 pts) Multiplication is correct

☐ (15 pts) Division is correct

☐ (4 pts) Remainder is correct

☐ (5 pts) Outputs are written to the correct address

☐ (1 pts) Able to output the result to the terminal

☐ (10 pts) Comments, including:

- Block Comments
- Clear comments on register usage, and clear comments all along
- Code is indented and aligned nicely
- Does your code do what you think it does? (accurate comments)

Lab write-up requirements

In the lab write-up, we will be looking for the following things. The lab report is worth 15 points. We do not break down the point values; instead, we will assess the lab report as a whole while looking for the following content in the report.

- The algorithm you are using in your program in simple, layperson's terms (or in C-like pseudocode)
- What works and what does not.
- Sample output.

Additionally, please answer the following questions.

- How does a branch instruction in LC-3 work?
- Which store instruction (ST, STI, STR) did you use to store your result to the correct memory location? Why did you use that particular instruction and not the other two?
- What is an addressing mode? What are the fiv LC-3 addressing modes? Give an example of each one.
- Which register is used as a place to put return values for TRAP instructions?
- By stepping through your program, what does the PUTS trap do? Please explain the entire process of how PUTS is executed.