Assignment description – Project 3

Objective

Get a taste of low-level programming in assembly language, and get acquainted with the Hack computer platform. In the process of working on this project, you will also become familiar with the assembly process, and you will appreciate visually how the translated binary code executes on the target hardware.

Resources

In this project you will use two tools supplied with the book: An Assembler, designed to translate Hack assembly programs into binary code, and a CPU Emulator, designed to run binary programs on a simulated Hack platform.

Contract

Write and test the two assembly programs described in what follows.

Multiplication Program (Mult.asm): The inputs of this program are the current values stored in R0 and R1 (i.e., the two top RAM locations). The program computes the product R0*R1 and stores the result in R2. We assume (in this program) that R0>=0, R1>=0, and R0*R1<32768. Your program does not need to test these conditions, but rather assume that they hold. The supplied Mult.tst and Mult.cmp scripts will test your program on several representative data values.

I/O-Handling Program (Fill.asm): This program should run an infinite loop that listens to the keyboard input. While a key is pressed (any key), the program gradually blackens the screen (of the CPU emulator), namely, writes ‘‘black’’ in the pixels. When no key is pressed, the screen should be gradually cleared. Note, the screen should not change to black or to white immediately, but it should change gradually, as a key is pressed. You may choose to blacken and clear the pixels in any spatial order, as long as pressing a key continuously for long enough will result in a fully blackened screen and not pressing any key for long enough will result in a cleared screen. The program can be tested by inspecting the result on the screen.

Hints

- Read the recommended Steps, Debugging Tip, Assembler and Emulator information in the book, Chapter 4.4.
- Comment your programs, so that you can easily present your solution.
- Multiplication should be based on repeated addition.
- Use the Assembler and CPU emulator tools for deeper understanding of the course material. Use the Assembler to inspect the assembly and machine language codes. Observe how labels and variables appear in the tools.

Submission

Your project must be mailed to your group leader by the deadline given on the course web. Make sure that the subject field states *EP1200-groupN*, if you are in seminar group N. All projects should be done individually. Send one submission per student, in a zipped file named EP1200-seminarM-GroupN-firstname-familyname.zip including:
- The solutions (typically code).
- The list of the sub-projects you solved successfully and can present. Use the provided form.

Copying and other forms of plagiarism and cheating will be reported for disciplinary action!