Lab 1: Introduction to VerySimpleCPU

Objective

Introduce the concepts and general practices for programming VerySimpleCPU using our Instruction Set Simulator (ISS).

Background

B1: VerySimpleCPU ISS

VerySimpleCPU.exe is an ISS, that is, simulation model.

- Takes a program file, coded in VerySimpleCPU assembly language, as input,
- Reads VerySimpleCPU instructions,
- Executes the instructions, and
- Writes results on to the command window as well as text files.

Provides a debugging environment for VerySimpleCPU programmers.

B2: VerySimpleCPU Instruction Set

Please look in "VSCPU/InstructionSet" folder. Make sure you go through the README file, which contains the detailed explanations of the instructions. Do not forget to look at the pictures in the same folder.

B3: Operating System Requirement

For those of you who are using another operating system other than windows, instead of using VerySimpleCPU.exe use http://verysimplecpu.org/

What To Do

Make sure you **watch the video before coming to the lab session**: https://youtu.be/vZBMbAt4rPM

Part 1

In Part 1 of this lab, you will run your first VerySimpleCPU assembly program that finds the **minimum of the two numbers** in addresses **101, 102** and writes the **result into address 200.** In Part 2, you modify the program so that it does something else and then you run it again.

Step 1

1. Create a workspace for CS240 Lab
    a. Create a new folder and name it "CS240workspace"
    b. Download LAB01 folder from LMS
2. Open and analyze the two C files with a text editor. (I suggest you download notepad++.)
3. Open "lab01_part1_min.asm" then analyze and compare it with "lab01_part1_min_low.c"
4. Open the command prompt
5. Type "cd "with file path which is "..\CS240workspace\VSCPU"
   (i.e. cd C:\Users\student\Desktop\CS240workspace\VSCPU)

6. Type “VerySimpleCPU.exe lab01_part1_min.asm r > log” then press ENTER
7. Type “exit” then press ENTER again
8. Open the log file and look at it. The ISS displays memory location before and after every instruction.
9. Look at “memoutd.txt”. It displays the final memory contents. Look at location 200 and make sure it is the smallest among *101 and *102.

Step 2

1. Rewrite the two C codes and the .asm in files lab01_part1_min_hi.c, lab01_part1_min_low.c, and lab01_part1_min.asm so that the program finds the maximum not the minimum. Name them lab01_part1_max_hi.c, lab01_part1_max_low.c, and lab01_part1_max.asm respectively.
2. Run your program and check the result through memoutd.txt.
3. If it does not work correctly, debug through the “log” file.

Part 2

1. Rewrite the two C codes and the .asm in part1, lab01_part1_max_hi.c, lab01_part1_max_low.c, and lab01_part1_max.asm so that the program finds the maximum of 3 numbers at *101, *102, *103 and writes to *200. Name those new files lab01_part2_max3_hi.c, lab01_part2_max3_low.c and lab01_part2_max3.asm.

2. Run your program and check the result through memoutd.txt.

3. If it does not work correctly, debug through the “log” file.

Submission

- Submit the following files in LMS under the assignment LAB01. Do not zip your files, upload them directly on LMS!
    - lab01_part1_max_hi.c
    - lab01_part1_max_low.c
    - lab01_part1_max.asm
    - lab01_part2_max3_hi.c
    - lab01_part2_max3_low.c
    - lab01_part2_max3.asm