Computer Science 237 Final Project 2014
The WArm Virtual Emulator (WAVE)
Final Deadline: Last day of classes, 5pm

Our final project is to host a virtual version of the WARM on a processor called the WIND. The WARM is, as you now know, an ARM-like RISC processor. The WIND is similar to an Intel x86-like processor. Your task is to load WARM programs into the WIND memory and simulate the the execution of the program by a WARM processor. As with any successful emulator, the WARM virtual emulator (or WAVE) will be a success if it (1) runs correctly, (2) runs quickly, and (3) uses as little memory as possible.

This is, needless to say, a complex project that will require considerable planning. Here are some steps that are likely to lead to a successful project:

1. You must work with a partner (I will allow an argument for a group of three). Each of you must send me an email message for me to recognize the partnership. You should also email Mary (`mary`) to arrange for unix group containing each of your team members. Only one member need apply.

2. Carefully read the documentation associated with the WARM and the WIND, and ask questions if you do not understand something.

3. Begin your work on your *design* over the next week. Launching into writing code immediately is not advisable, but you may want to write a couple of experimental programs to help you understand some of the subtleties involved.

4. After you form your group (and it is acknowledge), sign up for a appointment to meet with me before the end of next week (November 21) to discuss your approach. You will find the signup sheet outside my office, all appointments are 15 minutes.

5. Strive to get the emulator finished early in the last week of classes. This will allow you to meet with me for another mandatory appointment and to spend some of your time optimizing your code.

6. Turn in the final project on the last day of classes.

Grading of this project will be based on correctness, speed, and size, in that order. Understanding that there are basic information-theoretic reasons for the trading of time and space, your grade will depend on how close you get to the curve that represents that tradeoff. Optimization, is likely to play an important role.

Phase 1, wave1.s. Due Friday, November 14.

For Friday I would like everyone to **individually** write a WIND program that loads a WARM program into memory, starting at location WIND address 1000. That address will correspond to a WARM address of 0. Once loaded, your program should initialize a virtual WARM program counter (in a register or in memory), with `0x0` and, in a tight loop on this WARM `pc`, read each WARM instruction, checking to see if it corresponds to `swi #SysHalt` (an instruction we expect to see in most programs). When you reach the halt instruction, print out the value of the WARM program counter.

Phase 2, wave2.s. Due Tuesday, November 25.

By this time I expect that you will be able to execute (at least) unconditional arithmetic and logical instructions, and traps using (at least) registers and immediate data. You should also be able to (unconditionally) branch to other locations.

Phase 3, wave3.s. Due last day of classes.

This is a firm and unconditional deadline. You should submit the completed project, which includes testing and conditional instruction execution, all shifter modes, and load/store instructions. Your emulator should be able to execute any WARM programs.