CSc 256 Lab manual

(rev. 8/2016)

William Hsu, Department of Computer Science
Before we begin: the preliminaries

Welcome to the CSc 256 lab manual! This manual contains a number of tutorials, exercises and other useful information that will be helpful for you to successfully complete CSc 256.

This is a self-paced lab. Follow the schedule for each lesson, listed in your CSc 256 syllabus. Many of the lessons reinforce the material from lectures by tracing through code in detail, and working out supplementary examples. You work on the lab exercises on your own, according to the schedule laid out in the CSc 256 syllabus. For many of the exercises, you will have to email your results to the instructor; check the instructions in each exercise.

For this class, all the reference software will be on the server unixlab.sfsu.edu. Your SFSU account will enable you to log on to unixlab. If you already have an account on one of these machines, you are ready to continue on to the next section, Online Versions of this Manual. Otherwise, your account should be set up within a week. You may also install course software on your own machine; see instructions in the next section.
Online Versions of this Manual

Online versions of the CSc 256 lab manual can be found at

http://unixlab.sfsu.edu/~whsu/csc256/LABS/

Software for CSc 256

The main pieces of software we use in CSc 256 are 1) the GNU debugger for Unix, gdb, and 2) a CPU architecture simulator called spim (we'll explain more about what spim is later). A debugger such as gdb is an application that helps programmers to debug programs, by letting programmers examine carefully all activity in their programs, and trace through their programs step by step. gdb is available on most Unix systems. There are also graphics interfaces for gdb, some of which are available on systems on campus. Check with CSc lab staff for details.

spim is both an architecture simulator and a debugger. A version of spim (8.0) is available on unixlab, and you will be told how to access it. There are also versions of spim for the Mac and PC. In addition, there are versions of spim with graphical user interfaces: xspim for Macs and Unix systems, and PCspim for Windows. Spim/xspim software can be found on the CD that comes with the 4th edition of your textbook, Patterson and Hennessy’s Computer Organization. You may also download a copy of spim/xspim at

http://pages.cs.wisc.edu/~larus/spim.html

Hence, you have the choice of running the copy of spim/xspim on unixlab, or installing spim/xspim/PCspim on your own machine and doing your work locally.

Note that the current versions of spim (9.0+), QtSpim, are now based on the cross-platform Qt UI framework. These look a little different from spim/xspim, but the underlying spim engine should be pretty much identical. If you’d like to use QtSpim instead, you can download it at

http://spimsimulator.sourceforge.net/

There may be slight differences amongst the different versions. For the purposes of CSc 256, we will adopt the Unix version (provided on unixlab.sfsu.edu in ~whsu/bin/spim or ~whsu/bin/xspim) as the standard. All CSc 256 projects will be graded based on the standard Unix version, and CSc 256 exercises and exam questions will be based on the Unix version as well. If you use the PC version, be very careful when working with string and character data, as the byte-order is different (we'll explain this in CSc 256).
Yet another MIPS simulator that students in CSc 256 have had good experiences with is Mars:

http://courses.missouristate.edu/KenVollmar/MARS/

Mars is Java-based, thus is also cross-platform.

All the MIPS code examples used in CSc 256 should run pretty much identically on spim, xspim, QtSpim and Mars. (I’ve used all of these interchangeably at various times.)