Building Vesp in a Nutshell

ENEE350, University of Maryland

Fall 2010, Timothy Creech

This document contains brief instructions for building Prof. Oruç's VESP computer simulator, which is implemented in C++.

Requirements

C++ Compiler. Vesp is likely to compile fine under other compilers, but for the sake of brevity we will suggest GCC. We are interested in the C++ compiler included in GCC, called `g++`. This is free and widely-available software, however it is most easily (in the author's opinion) used on Unix and Unix-like systems.

For OSX: Install Apple's Developer Tools. Among these tools is GCC.

For Linux: Install (through your distribution's package manager) development tools. For example, in Fedora, try running in a terminal:

`sudo yum groupinstall "Development Tools" "Legacy Software Development"`

In Debian/Ubuntu, try:

`sudo aptitude install build-essential`

For Windows: I suspect you can install MinGW (www.mingw.org) which will provide GCC. I have no Windows machine on which I can test this.

If all else fails: Use an SSH client (PuTTY is good and free for Windows) to connect to terpconnect.umd.edu. Use your University common login to authenticate. These are Solaris 10 systems with lots of software including GCC installed already.

Instructions

1. Save the `cpp` file from the course website to a convenient location: right-click on the link to the cpp source file, and then use a command like

   `wget "http://www.ece.umd.edu/the/path/to/main.cpp"`

   or,

   `curl -O "http://www.ece.umd.edu/the/path/to/main.cpp"`

2. Run the command: `g++ main.cpp -o main`

3. The above should produce no output on screen, but a new file, "`main`," as specified above, will be created. This is an executable which you can run: `./main`

NOTE: `g++` as installed on each operating system will produce a binary ("`main`") which is only likely to run on that same machine type and OS. So, for example, don't expect that a "`main`" built on Terpconnect will run in OSX or Linux.