ECE112 - Lab 6

Purpose

- Make sure everyone can access a Linux host
- Get further acquainted with Linux
- Create a spice netlist, run a simulation
- Create a lab report using Latex

Parts/tools needed:

- Your laptop loaded with:
  (if PC): *PuTTY and Xming*
  (if Mac): *XQuartz*

Make sure we have a properly working environment

This lab works differently than previous ones. Proceed through the instructions the best you can. If you get stuck, try to figure it out, but if you're stuck for more than a few minutes, ask a TA for help. Most of the problems that come up in this lab are from not following directions or reading closely.

Mac Users

Install software:

Go to: *http://www.xquartz.org/*. Install XQuartz-2.7.8.dmg. After installing, log out and back into your Mac. This ensures that XQuartz will be your default X-server. Also, from the applications folder, drag the icon for *Terminal* into your Doc so it will be readily available from the desktop.

To log into one of the college of engineering (COE) Linux compute servers, type:
`ssh -Y -l <user_name> access.engr.orst.edu <enter>`
in the terminal window. The name `access.engr.orst.edu` is a pseudo-name for a group of COE Linux compute servers.

You may get a question from your laptop the first time you login to the Linux server. It may say that the authenticity of the host cannot be established. When it asks if you want continue to connect, type `yes` or `<enter>`.

You supply your user name between the greater and less than signs. Don't include either sign either. Also, that's a lower case "L", not a numeral one. The -Y switch on the `ssh` command indicates that you intend to display any graphical windows on your laptop, not on the remote computer. Your laptop will merely be acting as a display device. The switches `-Y` and `-l` to the `ssh` command are case sensitive as is the rest of Linux.

You may get a query from the remote Linux machine like:
`Terminal type? [xterm-color]` or
`Terminal type? [xterm-256color]`
You can simply ignore these questions and hit `<enter>`.

After that you will be sitting at a Linux prompt in the Terminal window, ready to do work. When you are ready to disconnect from the remote Linux host, type `exit` followed by `<return>`. Do the same steps to close the Terminal window on the laptop.

If you find your keyboard does not interpret the backspace correctly, in the Terminal app under *Terminal* > *Preferences*, select the Settings (Profiles in later versions of OSX) sub pane, and *Advanced*. Click the box that says *Delete sends Control-H*. See figure 1.

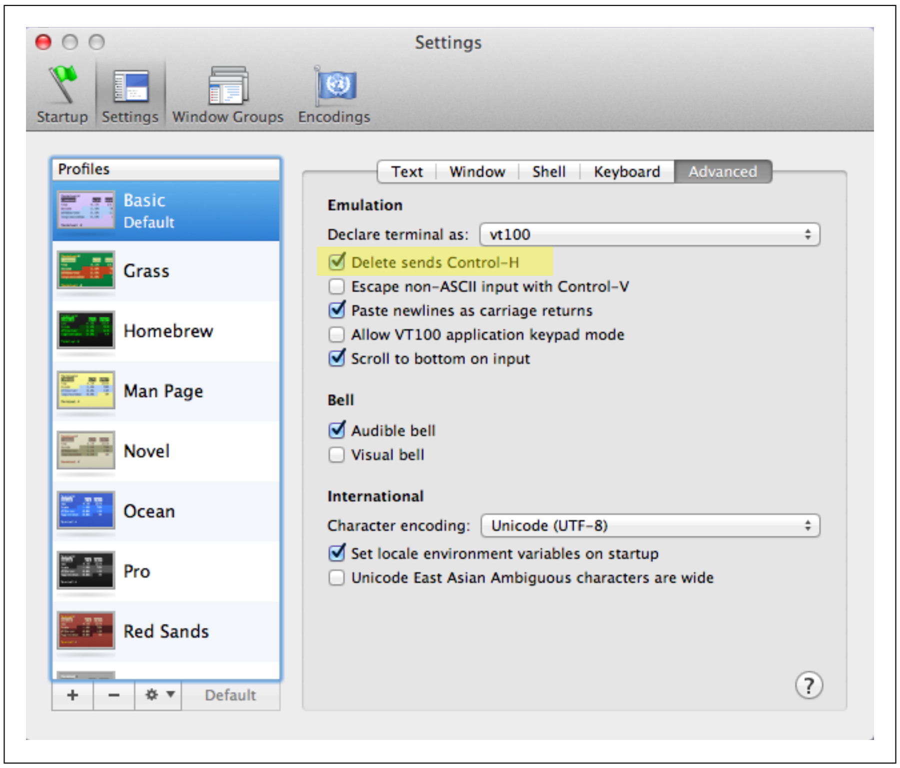


Figure 1: Setting Delete key Preferences

To have OS X automatically close the terminal window when you exit, under the Terminal preferences, select the *Shell* tab and select *Close the window* under the selection for when the shell exits.

When you run Terminal and tell it (via ssh -l .....) to connect to a Linux machine, anything you type into that window is sent straight to the Linux machine, and everything the Linux machine sends back is displayed in the window. So you can do work on the Linux machine while sitting somewhere else.

Windows Users

Do the necessary installations:

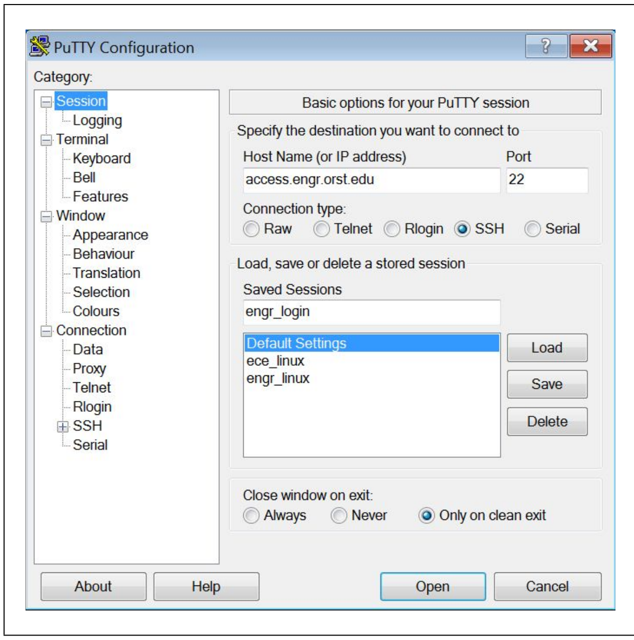
PuTTy can be downloaded from:
`http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html`
Select the file: `putty-0.66-installer.exe`

PuTTY is an application for creating secure shell connections from a Windows OS. PuTTY provides you a LINUX shell to access your files and email stored on the engineering servers. PuTTY can be used from Windows computers on or off campus. PuTTY is installed on all engineering computers running Windows.

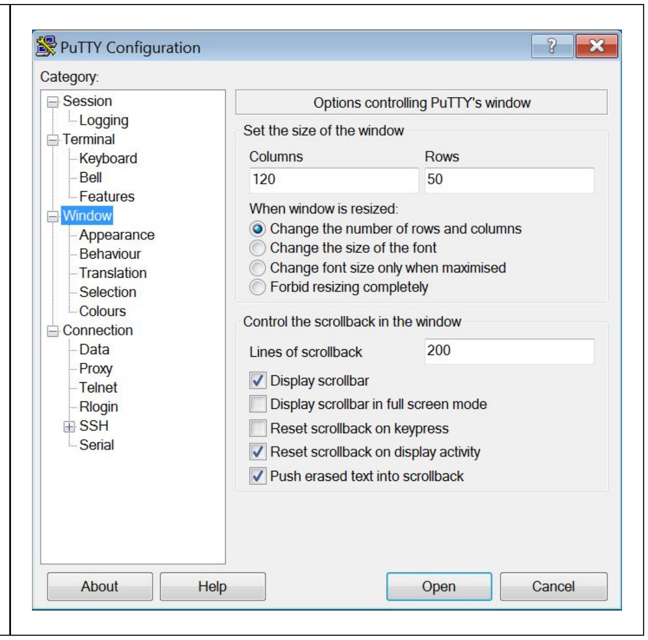
Xming can be downloaded from:
`http://sourceforge.net/projects/xming/files/Xming/6.9.0.31`
Select the file: `Xming-6-9-0-31-setup.exe`. Xming is a free X window server for Microsoft Windows. It allows one to use Linux graphical applications remotely. A more detailed description may be found at the software's home page.

When you run PuTTY on a Windows machine, and tell it to connect to a Linux machine, PuTTY opens a window locally and a shell remotely. Then, anything you type into that window is sent straight to the Linux machine, and everything the Linux machine sends back is displayed in the window. So you can work on the Linux machine while actually sitting somewhere else.

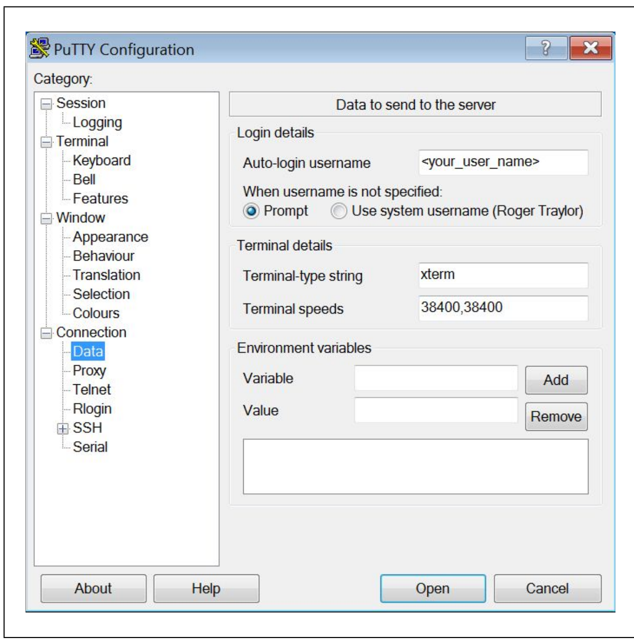
Set up PuTTY and Xming as the following dialog boxes indicate in figures 2 and 4. making sure that you go back to the session screen and click save to save the new settings for PuTTY as shown in 3a. After doing that, first start Xming. Xming must be started first. When Xming is started, no windows open. To check that Xming is running the logo should be visible in the lower right hand corner of your laptop screen. All editing for Xming must be done in the partner application, Xlaunch.



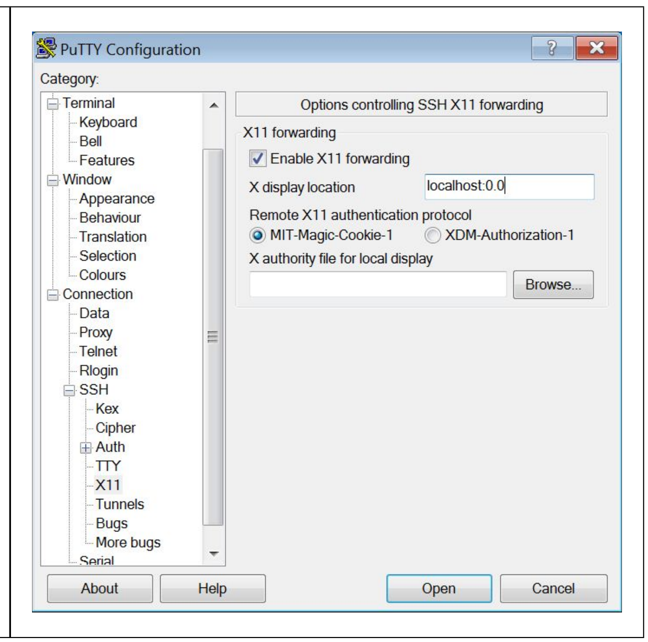
(a) PuTTY 1



(b) PuTTY 2



(c) PuTTY 3



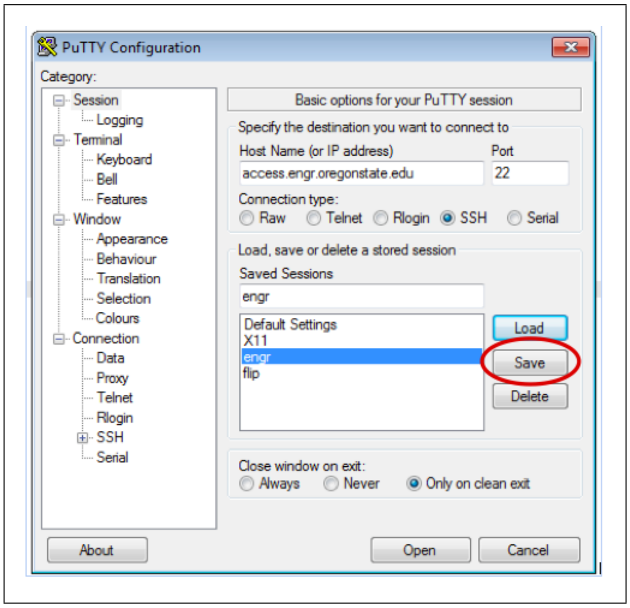
(d) PuTTY 4

Figure 2: PuTTY Setup Dialog Boxes

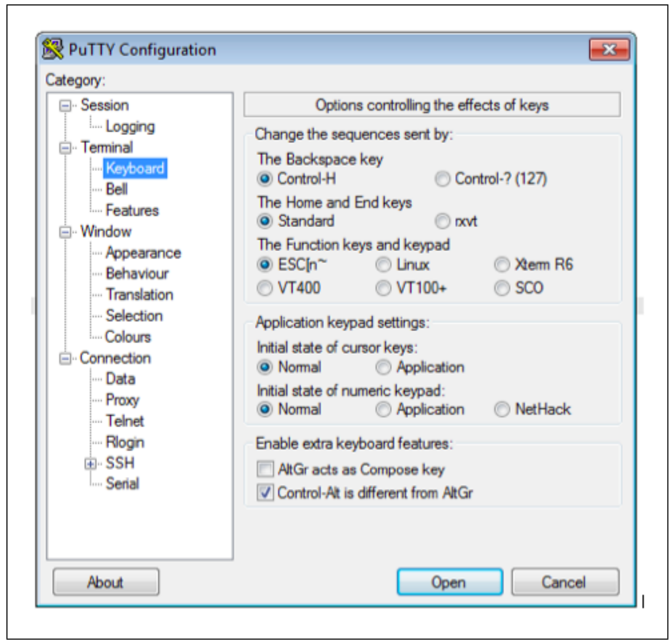
Start PuTTY and choose the session name you entered initially and click OPEN. You will then be presented with a Linux prompt in the window PuTTY created. Login with your password. You will then be ready to do work. Should you get a question from your laptop the first time you login to the Linux server concerning the authenticity of the host, just continue to connect, by typing `yes` or `<enter>`.

If your backspace key does not function properly, go back to your PuTTY setup and set the backspace sends control-H functionality. See figure 3b

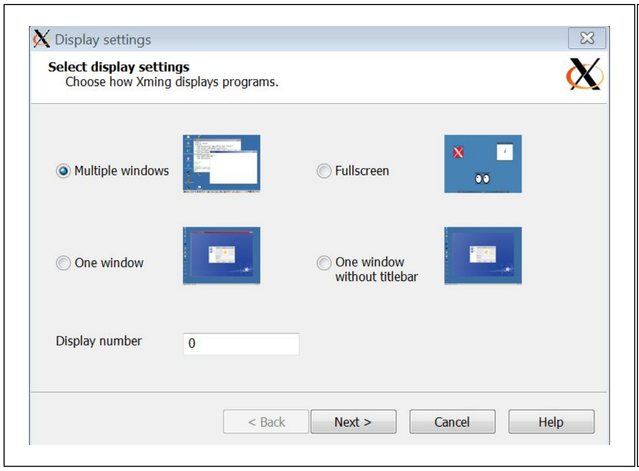
When you are ready to disconnect from the remote Linux host, type `exit` followed by `<return>`. You can then close PuTTY.



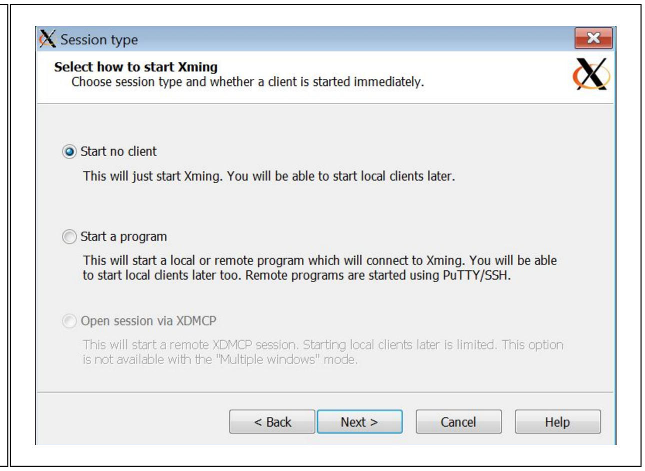
(a) Saving the Configuration of PuTTY



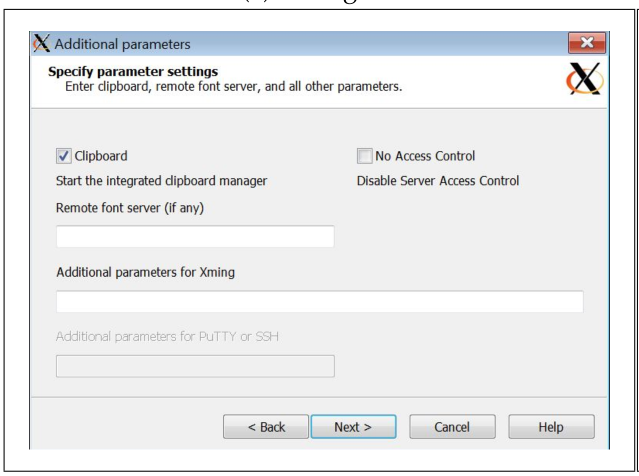
(b) Setting PuTTY's backspace Key Functionality



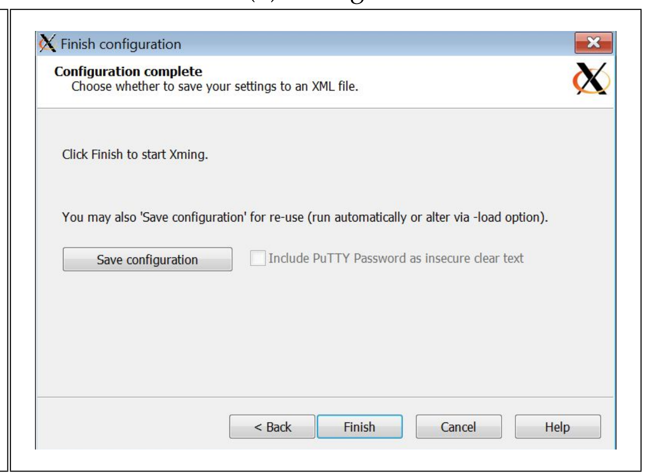
(a) Xming 1



(b) Xming 2



(c) Xming 3



(d) Xming 4

Figure 4: Xming Setup Dialog Boxes

Play with some Linux commands

Try out some simple Linux commands. Describe what each do.

`ls`

`ls -al`

`pwd`

`cd`

`whoami`

`who`

Run a spice simulation

Create a new directory in your home directory called `ece112` using `mkdir`. When using Linux, do not embed spaces in your directory names or file names. It makes life hard. You can get to your home directory by simply typing `cd` or `cd $HOME`. Create another directory in the `ece112` directory called `spice` in the same fashion. Descend into the `spice` directory. Now copy the spice file
`http://web.engr.orst.edu/~traylor/ece112/labs/src/lab6/rc.sp`
to the `spice` directory using `curl` as in:
`curl http://web.engr.orst.edu/~traylor/ece112/labs/src/lab6/rc.sp > rc.sp`

See if you were successful in downloading the file by checking how big in bytes it is using `ls -al`. If you are not sure what you are seeing, use the manual for ls: `man ls`. What is the file size of rc.sp

The size of `rc.sp` is: ____________________

Display the text of the spice file to your screen using either `cat`, `less` or `view` command and draw the schematic diagram from the netlist portion of the spice file. Be sure to include all the node names and show how the ground symbol is connected. Note: "*" means a line is commented out.

Run a spice simulation with the spice file `rc.sp` using `ngspice`. This is done by giving the shell the program to run followed by the input spice file. e.g.: `ngspice rc.sp`

When ngspice runs the simulation, you will see two plots, one created by ngspice and another created by gnuplot. The one ngspice makes is for interactively working on your circuit while in ngspice. We are going to take the prettier one created by gnuplot and put it in a *Latex* document.

At the ngspice prompt, quit the program (with `quit`) and close the simulation windows that were created. Now look to see what files have been created. The file `sim_output.plt` holds the commands that ngspice produced for gnuplot to create the print-worthy plot. The file `sim_output.data` holds the actual data from the simulation as a time and voltage listing. This could be input to excel if you wanted. Finally, `sim_output.eps` is an Adobe encapsulated postscript file that can be used in document preparation.

Make a lab report using Latex

It can be efficient to use a programming language, such as *Latex* to create documents. For documents that have a fixed format, such as a lab report, journal paper or a master's thesis, a programming description of the layout can be used and reused for many documents. With each new document, you only need to enter the text and pictures.

One very big advantage is that the format will always be exactly correct, since it's been programmed into the document. Secondly, you never have to bother with pagination, creating a table of contents or figures. Being freed from any formatting chores, you never get distracted by playing with fonts, colors, margins or other non-essentials. Finally, if changes are to be made, the entire document, including the table of contents, lists of tables or figures, can be completely recreated and updated without any other efforts.

In this process, you never open a word processor, import a new picture, re-size it, save, quit, etc. Typically, a one word command will recreate the document again with all the formatting intact and any changes included. However, you will not see what things look like until you compile your document so the *feel* of typical document preparation is very different. Latex is not the best tool for all document preparation. It is cumbersome to use if you want to create unstructured, or highly *artistic* documents.

Download the file:
`http://web.engr.orst.edu/~traylor/ece112/labs/src/lab6/lab_rpt_skel.tex`
to the `spice` directory (probably where you are now). Think about how you downloaded the spice file and use a similar pattern. Open the file with `vim` and change the author to your name. Look around in the file. This `.tex` file contains both the formatting and the text of your lab report. Below, write the Latex command used to include the graphics file of your simulation. The command will be `includegraphics.......` Use the search facilities of `vim` to do this.

We need to first convert the output of the ngspice simulation, `sim_output.eps` from .eps to .pdf format. This is easily done with the command:
`convert sim_output.eps sim_output.pdf`

Now, if we run `pdflatex` it will create a formatted .pdf file from the .tex source file that includes the plot we made as well as our spice source file. At the prompt, type:
`/usr/local/apps/tex_live/current/bin/x86_64-linux/pdflatex lab_rpt_skel.tex`

At the conclusion of the latex compile operation, you should see a line that reports that a .pdf file was created, such as:
`Output written on lab_rpt_skel.pdf (1 page, 74967 bytes)`

You can view your lab report by launching one of several .pdf viewers:
`okular lab_rpt_skel.pdf`
OR
`evince lab_rpt_skel.pdf`

After closing the pdf viewer, let's change the spice file, rerun the simulation, and recreate the lab report. Using `vim`, edit the spice file `rc.sp` by commenting out the line with the 10pf capacitor by placing a `*` at the beginning of the line. The "`*`" in column one of a spice file indicates that this is a comment line. Copy this line (`yy`) and paste the copy (`p`) just below the original line. Then change the value of the capacitor on the uncommented out line to be 1pf instead of 10pf.

Rerun the simulation and then quit. Again, convert the sim_output.eps file to a .pdf format with:
`convert sim_output.eps sim_output.pdf`

Then recreate the lab report by again typing:
`pdflatex lab_rpt_skel.tex`
Make sure your .pdf version of the lab report has been changed both in the plot shown and in the source code you changed. You may notice that the referece to figure 1 has changed from "??" to

the correct figure number 1. The pdflatex program takes two passes to resolve figure numbers.

You can view and print your lab report for your TA by launching one of the .pdf viewers again.

Sign your lab report and have your TA check off your work.