This handout shows how to:

- Login to `timberlake.cse.buffalo.edu`
- Use the `iverilog` and `vvp` command
- Use `gtkwave` to visualize circuits
- Use `vim` editor to create your own verilog program
- Submit your assignment

The instructions here are for students who use Linux and Mac systems. Although the screenshots are taken on a Ubuntu distro, they should be easily applied on other variants (Fedora, Mint, etc.) and Mac OS.

As a Linux/Mac user, we assume you at least know how to launch a terminal and several basic Linux commands, such as `ls`, `cd`. If this is not the case, check out this web page.

`http://mally.stanford.edu/~sr/computing/basic-unix.html`

Login to `timberlake`

Your user name on `timberlake` should be your UBIT name. If you haven't logged in any department server before, please refer to this web page on how to determine your initial password and/or how to change it.

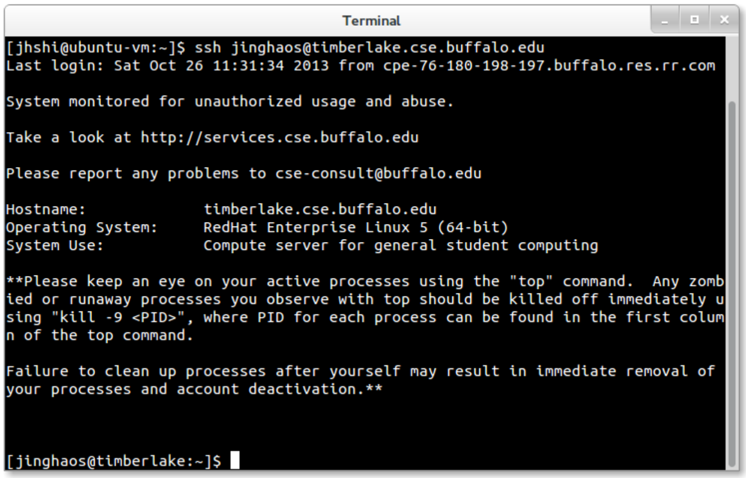
`https://wiki.cse.buffalo.edu/services/content/cse-unixlinux-accounts`

Once you determined your password, you can login `timberlake` using `ssh` command.

$ `ssh` *`YOUR_UBIT_NAME`*`@timberlake.cse.buffalo.edu`

Please note that

- You need to substitute *`YOUR_UBIT_NAME`* with your actual UBIT name
- You'll probably need to enter password if prompted



```
[jhshi@ubuntu-vm:~]$ ssh jinghaos@timberlake.cse.buffalo.edu
Last login: Sat Oct 26 11:31:34 2013 from cpe-76-180-198-197.buffalo.res.rr.com

System monitored for unauthorized usage and abuse.

Take a look at http://services.cse.buffalo.edu

Please report any problems to cse-consult@buffalo.edu

Hostname:              timberlake.cse.buffalo.edu
Operating System:      RedHat Enterprise Linux 5 (64-bit)
System Use:            Compute server for general student computing

**Please keep an eye on your active processes using the "top" command.  Any zomb
ied or runaway processes you observe with top should be killed off immediately u
sing "kill -9 <PID>", where PID for each process can be found in the first colum
n of the top command.

Failure to clean up processes after yourself may result in immediate removal of
your processes and account deactivation.**



[jinghaos@timberlake:~]$ 
```

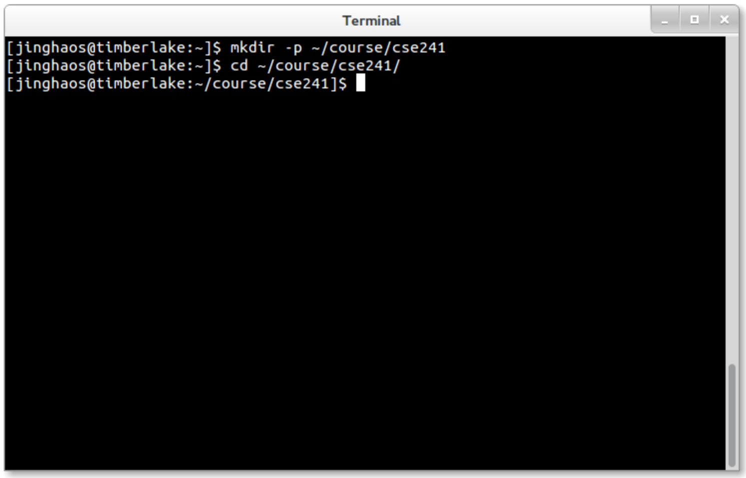
Figure 1: `ssh` into `timberlake`

Use `iverilog` to Compile HDL Source

Once you're in `timberlake` , you probably want to create a directory just for this course's files, so your home directory won't get polluted. In this tutorial, we use `~/course/cse241` as working directory.

```
$ mkdir -p ~/course/cse241
$ cd ~/course/cse241
```

In which, `mkdir` means *make directory*, `-p` means create any parent directory if needed, and `cd` means *change directory*.



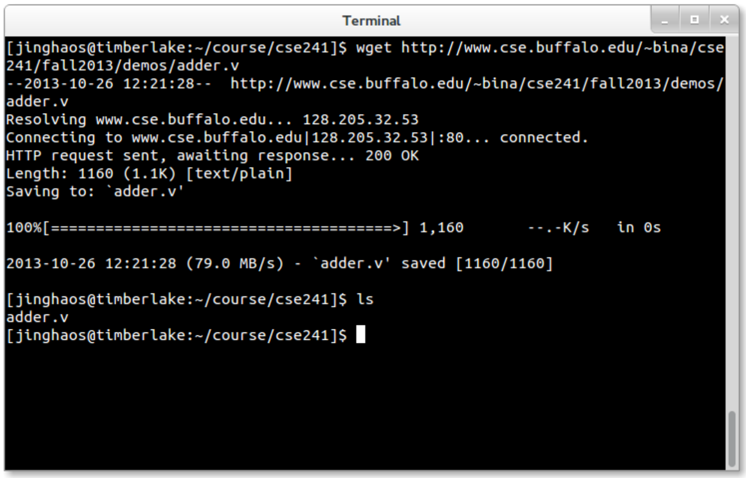
```
[jinghaos@timberlake:~]$ mkdir -p ~/course/cse241
[jinghaos@timberlake:~]$ cd ~/course/cse241/
[jinghaos@timberlake:~/course/cse241]$ 
```

Figure 2: Create a work directory

Here we use the `adder.v` example provided by the professor to demonstrate the usage of `iverilog` command.

First, download the file to our work directory.

```
$ wget http://www.cse.buffalo.edu/~bina/cse241/fall2013/demos/adder.v
```



```
[jinghaos@timberlake:~/course/cse241]$ wget http://www.cse.buffalo.edu/~bina/cse
241/fall2013/demos/adder.v
--2013-10-26 12:21:28--  http://www.cse.buffalo.edu/~bina/cse241/fall2013/demos/
adder.v
Resolving www.cse.buffalo.edu... 128.205.32.53
Connecting to www.cse.buffalo.edu|128.205.32.53|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1160 (1.1K) [text/plain]
Saving to: `adder.v'

100%[======================================>] 1,160       --.-K/s   in 0s

2013-10-26 12:21:28 (79.0 MB/s) - `adder.v' saved [1160/1160]

[jinghaos@timberlake:~/course/cse241]$ ls
adder.v
[jinghaos@timberlake:~/course/cse241]$ 
```

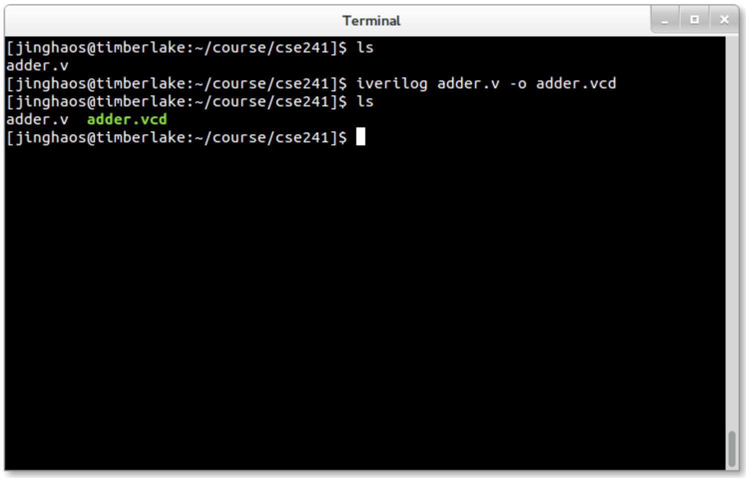
Figure 3: Download `adder.v` example

Then use `iverilog` command to compile the source file.

```
$ iverilog adder.v -o adder.vcd
```

Similar to `gcc`, `iverilog` takes `adder.v` as input source file, `-o adder.vcd` specifies the output dump file name.

If everything is OK (it should be), you should see a generated dump file named `adder.vcd` file.



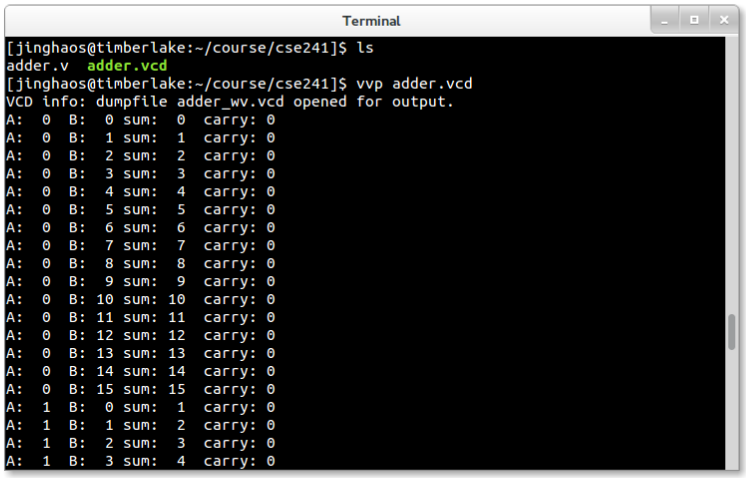
```
[jinghaos@timberlake:~/course/cse241]$ ls
adder.v
[jinghaos@timberlake:~/course/cse241]$ iverilog adder.v -o adder.vcd
[jinghaos@timberlake:~/course/cse241]$ ls
adder.v  adder.vcd
[jinghaos@timberlake:~/course/cse241]$ 
```

Figure 4: `iverilog` usage

Use `vvp` to Simulate the Circuit

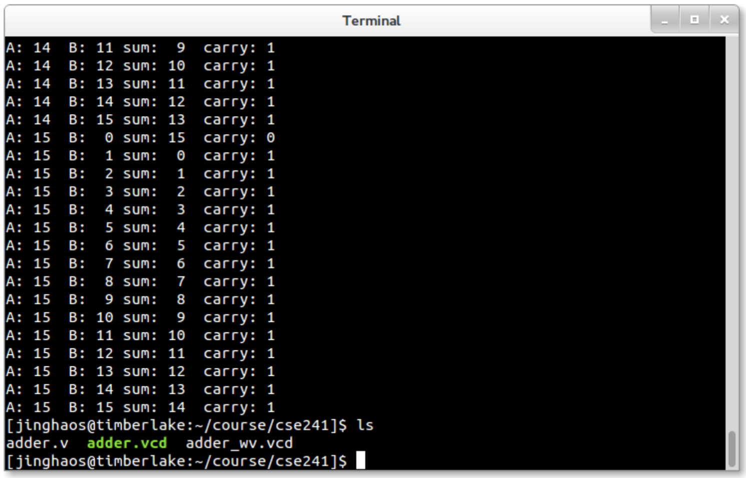
Once you have the compiled `vcd` file, you may want to use `vvp` command to simulate the circuit. This will test the adder module defined and will output the adder output for various test bench inputs. This will also generate the `adder_wv.vcd` file that we'll use for next step.

```
$ vvp adder.vcd
```



```
[jinghaos@timberlake:~/course/cse241]$ ls
adder.v  adder.vcd
[jinghaos@timberlake:~/course/cse241]$ vvp adder.vcd
VCD info: dumpfile adder_wv.vcd opened for output.
A:  0  B:  0 sum:  0  carry: 0
A:  0  B:  1 sum:  1  carry: 0
A:  0  B:  2 sum:  2  carry: 0
A:  0  B:  3 sum:  3  carry: 0
A:  0  B:  4 sum:  4  carry: 0
A:  0  B:  5 sum:  5  carry: 0
A:  0  B:  6 sum:  6  carry: 0
A:  0  B:  7 sum:  7  carry: 0
A:  0  B:  8 sum:  8  carry: 0
A:  0  B:  9 sum:  9  carry: 0
A:  0  B: 10 sum: 10  carry: 0
A:  0  B: 11 sum: 11  carry: 0
A:  0  B: 12 sum: 12  carry: 0
A:  0  B: 13 sum: 13  carry: 0
A:  0  B: 14 sum: 14  carry: 0
A:  0  B: 15 sum: 15  carry: 0
A:  1  B:  0 sum:  1  carry: 0
A:  1  B:  1 sum:  2  carry: 0
A:  1  B:  2 sum:  3  carry: 0
A:  1  B:  3 sum:  4  carry: 0
```

Figure 5: `vvp` usage



```
A: 14  B: 11 sum:  9  carry: 1
A: 14  B: 12 sum: 10  carry: 1
A: 14  B: 13 sum: 11  carry: 1
A: 14  B: 14 sum: 12  carry: 1
A: 14  B: 15 sum: 13  carry: 1
A: 15  B:  0 sum: 15  carry: 0
A: 15  B:  1 sum:  0  carry: 1
A: 15  B:  2 sum:  1  carry: 1
A: 15  B:  3 sum:  2  carry: 1
A: 15  B:  4 sum:  3  carry: 1
A: 15  B:  5 sum:  4  carry: 1
A: 15  B:  6 sum:  5  carry: 1
A: 15  B:  7 sum:  6  carry: 1
A: 15  B:  8 sum:  7  carry: 1
A: 15  B:  9 sum:  8  carry: 1
A: 15  B: 10 sum:  9  carry: 1
A: 15  B: 11 sum: 10  carry: 1
A: 15  B: 12 sum: 11  carry: 1
A: 15  B: 13 sum: 12  carry: 1
A: 15  B: 14 sum: 13  carry: 1
A: 15  B: 15 sum: 14  carry: 1
[jinghaos@timberlake:~/course/cse241]$ ls
adder.v  adder.vcd  adder_wv.vcd
[jinghaos@timberlake:~/course/cse241]$ 
```

Figure 6: `vvp` usage (cont.)

Note that the above outputs, and the wave file name (`adder_wv.vcd`) are generated or specified by the `adder.v` itself, not by `vvp`. Please take a look at the content of `adder.v` and make sure you understand what each line does.

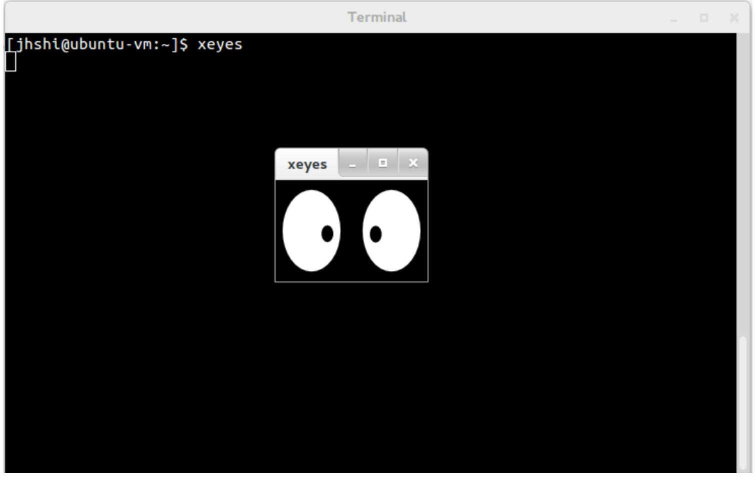
Use `gtkwave` to Visualize the Wave

For this part, you'll need to access the `timberlake` graphically to see the `gtkwave` window.

First of all, make sure your local X-Windows work. Type `xeyes` in your *local* terminal.

$ `xeyes`

You should be able see a window with a pair of eyes.



```
[jhshi@ubuntu-vm:~]$ xeyes
```

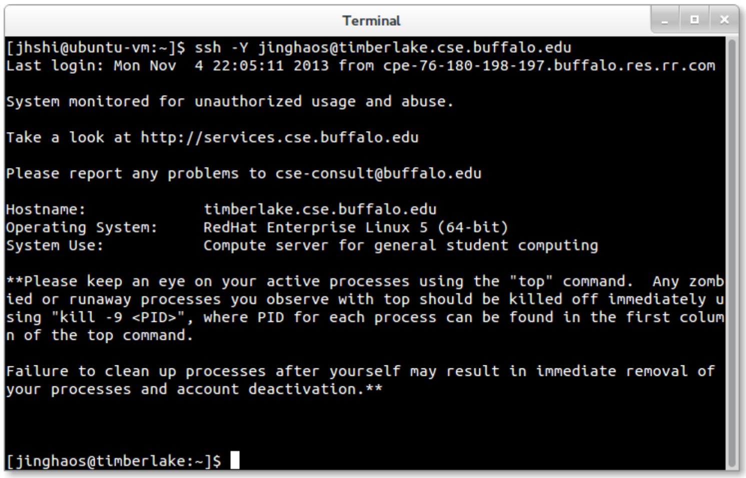
Figure 7: xeyes

For Mac users, if you can not execute `xeyes` command, please install X-Window system from here.

`http://xquartz.macosforge.org/landing/`

Then, login `timberlake` using `-Y` option when you `ssh`.

$ `ssh -Y jinghaos@timberlake.cse.buffalo.edu`

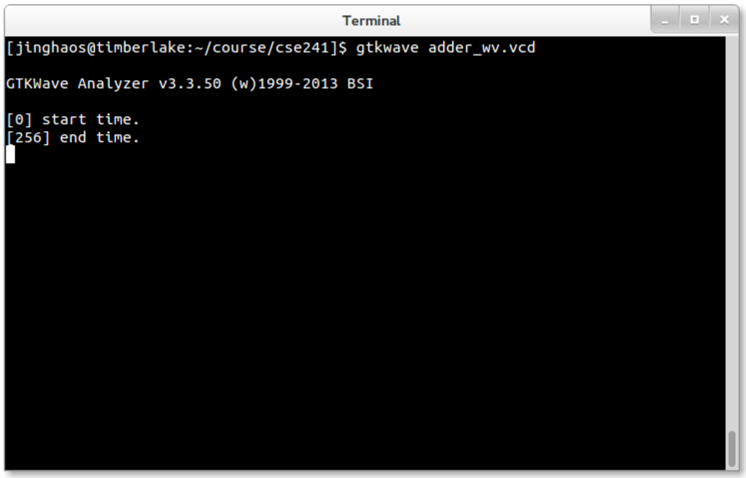


```
[jhshi@ubuntu-vm:~]$ ssh -Y jinghaos@timberlake.cse.buffalo.edu
Last login: Mon Nov  4 22:05:11 2013 from cpe-76-180-198-197.buffalo.res.rr.com

System monitored for unauthorized usage and abuse.

Take a look at http://services.cse.buffalo.edu

Please report any problems to cse-consult@buffalo.edu

Hostname:              timberlake.cse.buffalo.edu
Operating System:      RedHat Enterprise Linux 5 (64-bit)
System Use:            Compute server for general student computing

**Please keep an eye on your active processes using the "top" command.  Any zomb
ied or runaway processes you observe with top should be killed off immediately u
sing "kill -9 <PID>", where PID for each process can be found in the first colum
n of the top command.

Failure to clean up processes after yourself may result in immediate removal of
your processes and account deactivation.**


[jinghaos@timberlake:~]$ 
```

Figure 8: `gtkwave` usage

Fire up `gtkwave` using this command.

$ `gtkwave adder_wv.vcd`



```
[jinghaos@timberlake:~/course/cse241]$ gtkwave adder_wv.vcd

GTKWave Analyzer v3.3.50 (w)1999-2013 BSI

[0] start time.
[256] end time.
```

Figure 9: `gtkwave` usage

Click on the “ex2_1” in the SST panel, and drag the signals (`inA, inB, sum, carry`) to the “Signals” panel, you should see the waves!

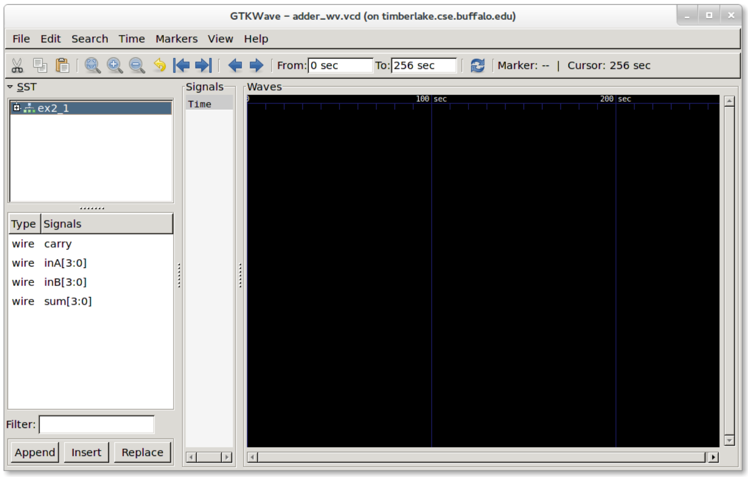


Figure 10: `gtkwave` screenshot

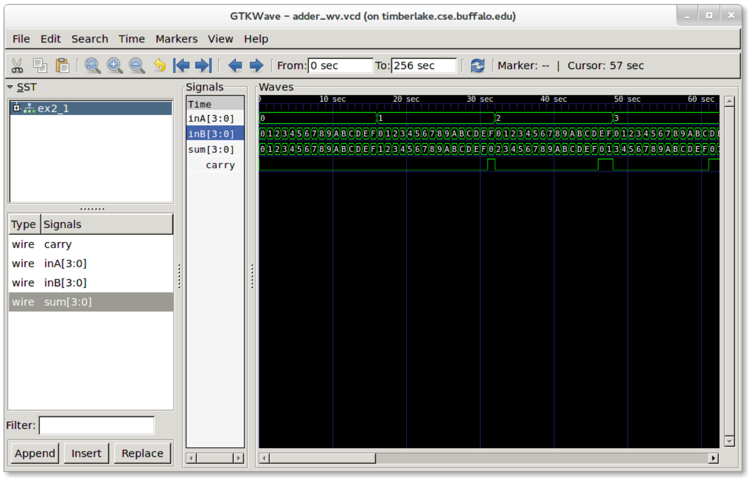
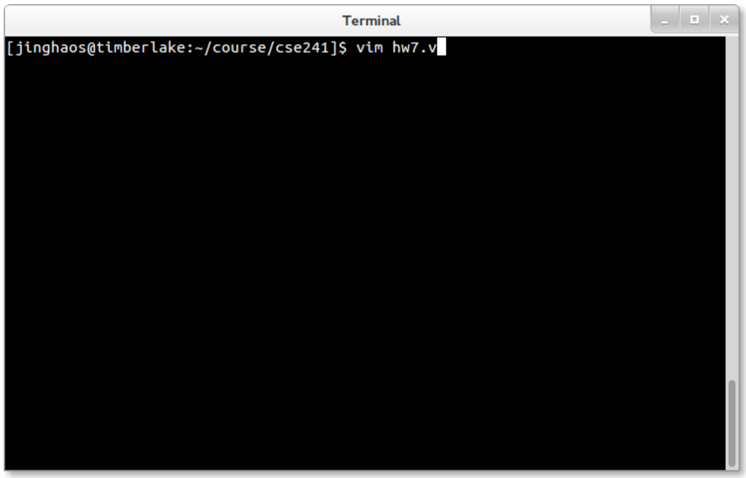


Figure 11: Waveform in `gtkwave`

Use `vim` to Create Your Verilog Program

At this moment, you should be familiar with the work flow of compiling and visualize verilog program. Now let's create your own verilog program, say, for homework 7. We will use `vim`, a powerful editor, to create the file.

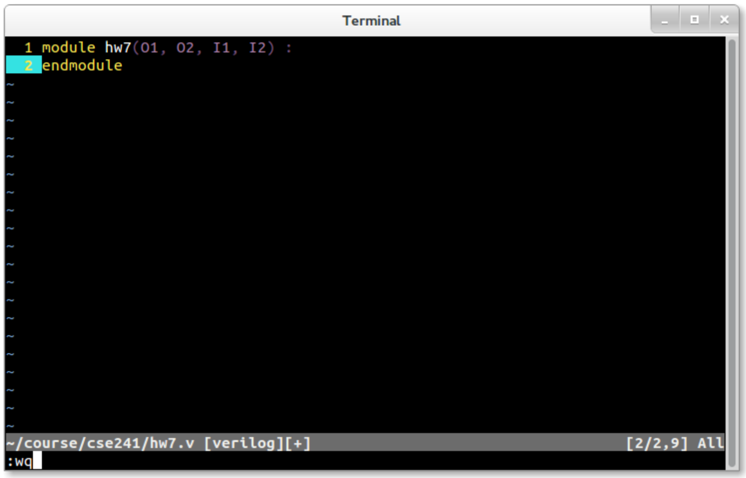
```
$ vim hw7.v
```



```
[jinghaos@timberlake:~/course/cse241]$ vim hw7.v
```

Figure 12: Use `vim` to create vile

Inside `vim`, type `a` to *append* content. After you done editing, type `Esc` to *escape*, then type `:wq` to save and quit `vim`.



```
module hw7(O1, O2, I1, I2) :
endmodule
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~/course/cse241/hw7.v [verilog][+]                      [2/2,9] All
:wq
```

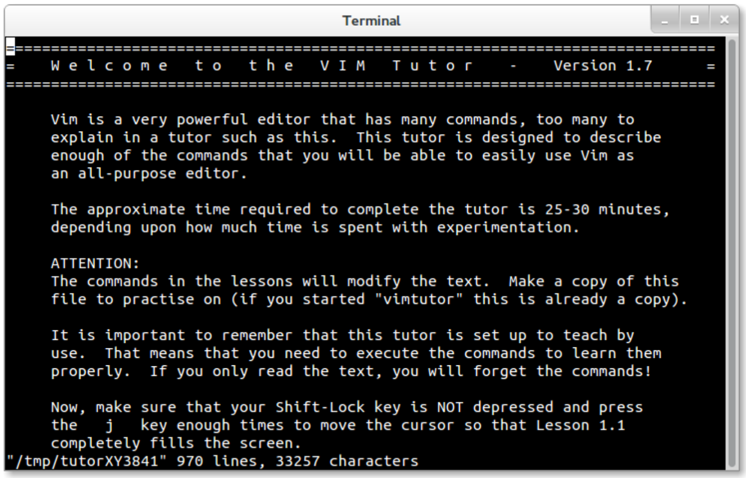
Figure 13: Save and quit `vim`

For more `vim` usage, please refer to this tutorial.

`http://vim.wikia.com/wiki/Tutorial`

Also, there is a nice command line tool called `vimtutor`, it should be good enough to get your started on basic `vim` usage.

$ `vimtutor`



```
===============================================================================
=    W e l c o m e   t o   t h e   V I M   T u t o r    -    Version 1.7      =
===============================================================================

     Vim is a very powerful editor that has many commands, too many to
     explain in a tutor such as this.  This tutor is designed to describe
     enough of the commands that you will be able to easily use Vim as
     an all-purpose editor.

     The approximate time required to complete the tutor is 25-30 minutes,
     depending upon how much time is spent with experimentation.

     ATTENTION:
     The commands in the lessons will modify the text.  Make a copy of this
     file to practise on (if you started "vimtutor" this is already a copy).

     It is important to remember that this tutor is set up to teach by
     use.  That means that you need to execute the commands to learn them
     properly.  If you only read the text, you will forget the commands!

     Now, make sure that your Shift-Lock key is NOT depressed and press
     the   j   key enough times to move the cursor so that Lesson 1.1
     completely fills the screen.
"/tmp/tutorXY3841" 970 lines, 33257 characters
```

Figure 14: `vimtutor`

How to Submit

Say you completed your HDL assignment and are pretty confident about it, you can use the `submit_cse241` command to submit your work. Suppose your HDL file is named as `my_awesome_sol.v`.

```
$ submit_cse241 my_awesome_sol.v
```

Please refer to this link about more information about the submit command.

`https://wiki.cse.buffalo.edu/services/content/submit-script`

Note that

- You can submit multiple times before the deadline
- If you submit files with the same file name multiple times, the previous one will be overwritten.
- We'll only grade the latest submission before deadline.

Resources

We just cover the basic usage of `iverilog`, `vvp` and `gtkwave` here, for more information, please refer to these resources.

- Iverilog wiki, including user guide, and examples.
  `http://iverilog.wikia.com/wiki/Main_Page`
- GtkWave manual.
  `http://gtkwave.sourceforge.net/gtkwave.pdf`