EE 3755 Homework 1 Due: TBA

Important: When logging in for the first time select "Common Desktop Environment" after entering your user name and password.
(If you're reading this after selecting the wrong option, log out and then, from the log in dialog select Options, Session, Common Desktop Environment.)
Solve this problem by modifying a copy of http://www.ece.lsu.edu/alex/EE3755/2004/hw01.v
Html : http://www.ece.lsu.edu/alex/EE3755/2004/hw01.html

Estimated time to finish:

Knowing How to Use ncverilog	**: Assume you already know (from 1Min to….Inf..).**
Prob.0	**: 2Mins.**
Prob.1	**: 1Min.**
Prob.2	**: 5Mins.**
Prob.3	**: 10Mins.**
Prob.4	**: 5Mins.**
Prob.5	**: 30Mins.**
Total	**: 52 Mins..**

When you submit the Hw., Please write down how much time did you spend for each problem.
(No penalty for spending too little or too much time. Just want to know how long you spent.)
How to submit: Hard copy during the class.
Use "script " command to take a snap shot of your program.
After run "script "
Use "cat" command to display your program.
Use "ncverilog" to run your program.
then stop the script.
Prob.1 to Prob. 4 are easy to code. Each one takes less than 5 lines.

Script is a unix command : usage : >> script "filename"

Problem 0: Draw the figure for the following verilog code.//This code will help you to do prob.2.

The figure will be something like Fig.1.

```
 module slice_es(eo,a,b,ei);
input   a, b, ei;
output eo;
wire   aneb, aeqb;
xor x1(aneb,a,b);
not n1(aeqb,aneb);
and a1(eo,aeqb,ei);
endmodule
```

Problem 1: Copy the homework template into a subdirectory named hw in your class account.
Simulate the welcome module in the homework template.
The logic diagram below is a 1-bit slice of a circuit that is used to determine whether a=~b.
One bit of one integer is put on input a and one bit of the other integer is put on input b.
(If the number has ten bits then ten slices would be needed.)
Ports ei and eo work something like carry in and carry out in a binary full adder.
Input ei is logic 1 if the higher bits of the two numbers are not equal. (ahigh=~bhigh).
Output eo is logic 1 if ei=1 and a=~b.

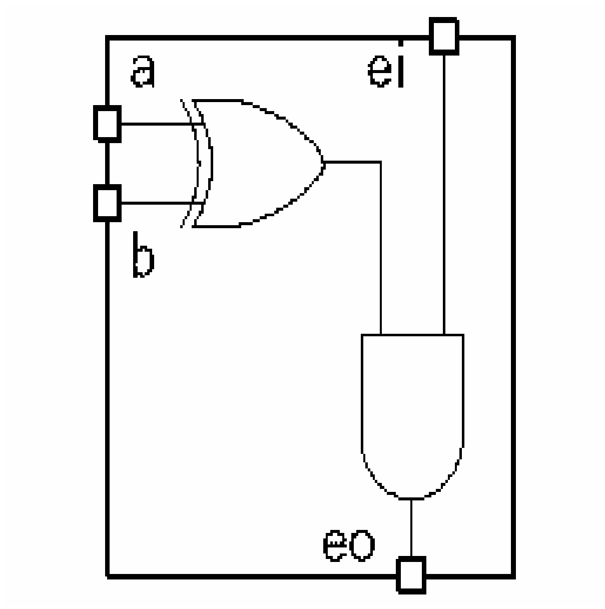

Fig.1) Slice of the unit.

The homework template contains module definitions for solutions to the problems below. It also contains a testbench, called test, that can be used to test the modules.
This should be the module that you simulate when testing the other modules. Do not modify testbench.

For example) if a=1, b=0 and ei=1, eo will be 1

Problem 2: Complete module neq_slice_es (in the homework template) so that it is an explicit structural description of the 1-bit slice hardware illustrated above. Remember that this module only handles one bit of a and b.

Problem 3: Complete module neq_slice so that it is an implicit structural description of the 1-bit slice hardware illustrated above. Remember that this module only handles one bit of a and b.

Problem 4: Module nequality is used to compare two four-bit numbers. It has two four-bit inputs, a and b, and a one-bit output eq.
Complete the module so that output eq is 1 iff a and ~b are equal. The module must instantiate four copies of neq_slice,
from the problem above.

For example) if 4 bit a=1110, b=0000 or a=1011, b=0101
eq will be 1.

These are just examples.

Hint)The solution should be something like this :

This is not the solution for the problem:
//intentional mistakes in it.

```
module not_eq(eq,a,b);
input [3:0] a, b;
output      eq;

wire        e1, e2, e3;

neq_slice es3(e2,a[3],b[3],1'b0);
neq_slice es2(e3,a[2],b[2],e3);
neq_slice es1(e1,a[1],b[1],e2);
neq_slice es0(eq,a[0],b[0],e1);
```

```
endmodule
```

Problem 5:

a) Draw figure of a slice just like the Fig.1, which will do magnitude comparator for 1 bit input a and b.
The slice has 5 input ports and 3 output ports.
.a,b : input ports:
.eq_in,big_in, small_in: input ports:
.eq_out, big_out, small_out: output ports:
General idea is : (if a==b) eq_out = 1. //(consider eq_in, small_in, big_in).
(if a < b) small_out = 1. //(consider eq_in, small_in, big_in).
(if a> b) big_out = 1. //(consider eq_in, small_in, big_in).
// you should have to consider input eq_in , big_in, small_in.(because we design one slice).

b) Draw figure of cascade of 3 slices to compare 3 bits number.