ENGR xD52: HW b001

Due September 15th Midnight EST

Expected time is one hour of actual work, plus an unknown amount of time installing ModelSim.

The purpose of this assignment is to familiarize you with one of the main tools we will be using this semester. As a side effect, you will exhaustively prove DeMorgan's Law.

Install ModelSim

We will be using ModelSim 10.3c Student Edition. Installation may take longer than you expect. I suggest doing it in the background while you are working on other things.

The installer is located at Public/+Courses/ComputerArchitecture/2014/Installers

Copy it to your local disk first, this will greatly speed the process.

Registration seems to work better using a .edu email address. You may need to try more than once. Unfortunately, trying again requires you to uninstall and reinstall ModelSim.

Best case scenario, this process can take as little as 20 minutes. Plan for longer.

Create Your Workspace and Do File

Create a folder to host your workspace. I suggest something close to root, without spaces, and relatively short, such as "C:\CA14HW1\"

In ModelSim, switch to this folder: "cd c:/CA14HW1/" (Forward slashes)

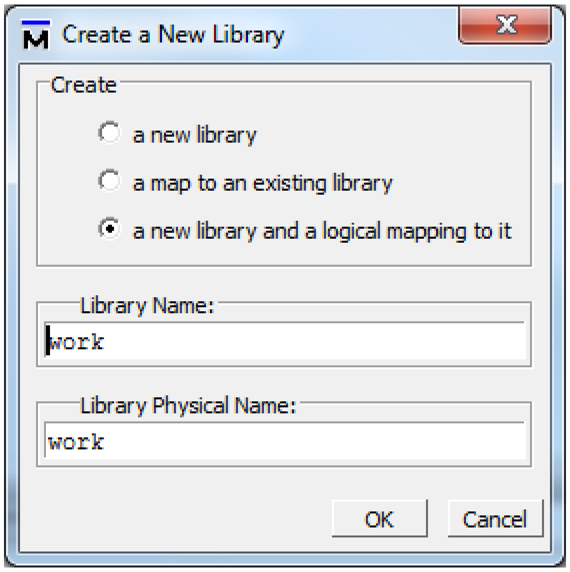
Create a new verilog file: "File -> New -> Source -> Verilog"

Save it with a useful name in that directory: "HW1.v"

Copy the following into it:

```
module hw1test;
initial begin
$display("Hello World");
end
endmodule
```

Create the working design Library: "File -> New -> Library"


Create a New Library

Create
- ( ) a new library
- ( ) a map to an existing library
- [✓ a new library and a logical mapping to it]

Library Name: work

Library Physical Name: work

Create a "do" file. This file will automate repetitive tasks for you. Anytime you are doing something manually often, consider creating a do file by seeing what commands clicking the buttons echo to the terminal.

"File -> New -> Source -> Do"

Copy the following into that file:

```
vlog -reportprogress 300 -work work hw1.v
vsim -voptargs="+acc" hw1test
run 100
```

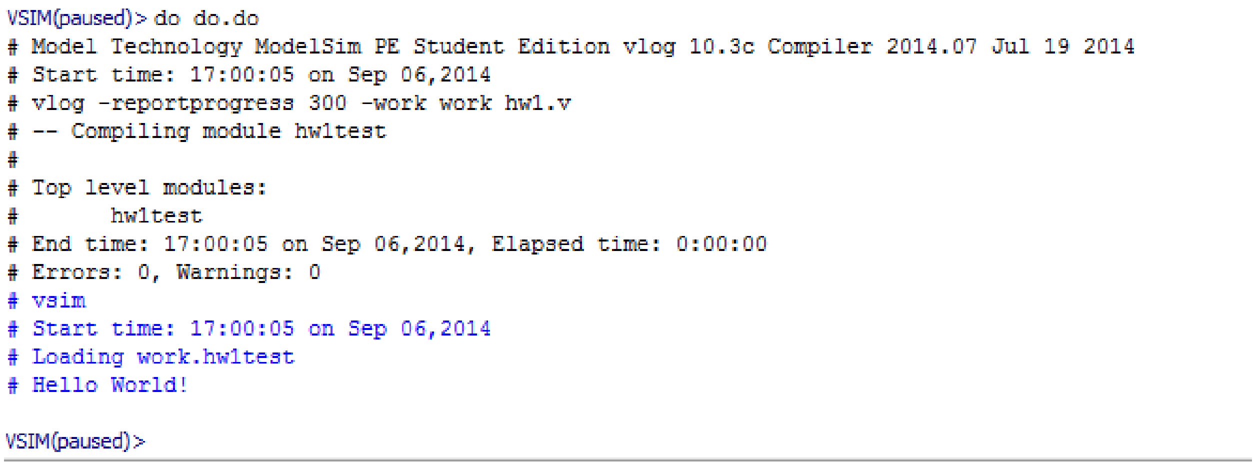
These commands mean the following:

1) Compile the file "hw1.v" into the workspace "work". If you use a different file name or use multiple source files, you will need to change this line.
2) Load the "hw1test" into the simulator. If you use a different top level module, you will need to change this line.
3) Run the simulator for 100 units of time. If your simulation takes more time, change this line.

The "-reportprogress 300" provides debugging information during compilation if something goes wrong. The "-voptargs="+acc"" makes sure the compiler allows you to see all the signals in your design. Otherwise it may optimize some of them away from you!

Save the .do file. "do.do"

Now you can run it from the command line: "do do.do". You should see the following:



```
VSIM(paused)> do do.do
# Model Technology ModelSim PE Student Edition vlog 10.3c Compiler 2014.07 Jul 19 2014
# Start time: 17:00:05 on Sep 06,2014
# vlog -reportprogress 300 -work work hw1.v
# -- Compiling module hw1test
#
# Top level modules:
#       hw1test
# End time: 17:00:05 on Sep 06,2014, Elapsed time: 0:00:00
# Errors: 0, Warnings: 0
# vsim
# Start time: 17:00:05 on Sep 06,2014
# Loading work.hw1test
# Hello World!

VSIM(paused)>
```

Your First Verilog

Prove Demorgan's Law using the exhaustive proof method. To do so, you will use ModelSim to create truth tables for the following 4 Equations:

$$\overline{(AB)}$$

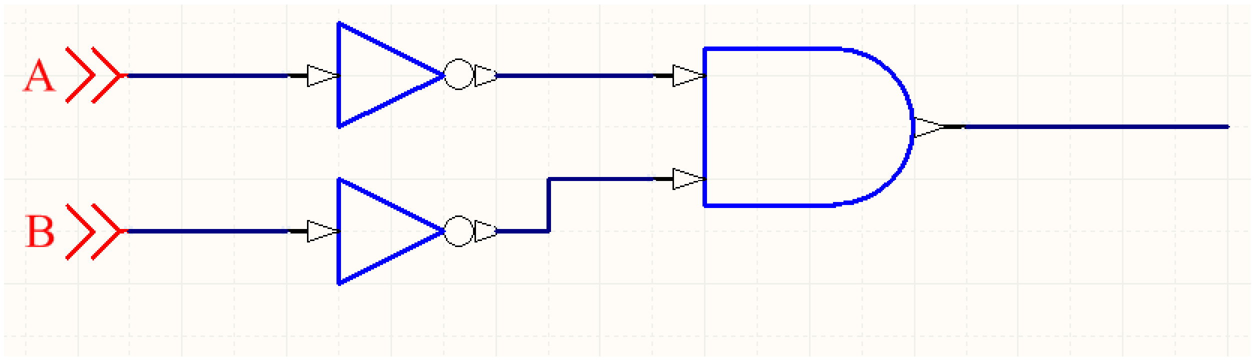
$$(\bar{A} + \bar{B})$$

$$\overline{(A + B)}$$

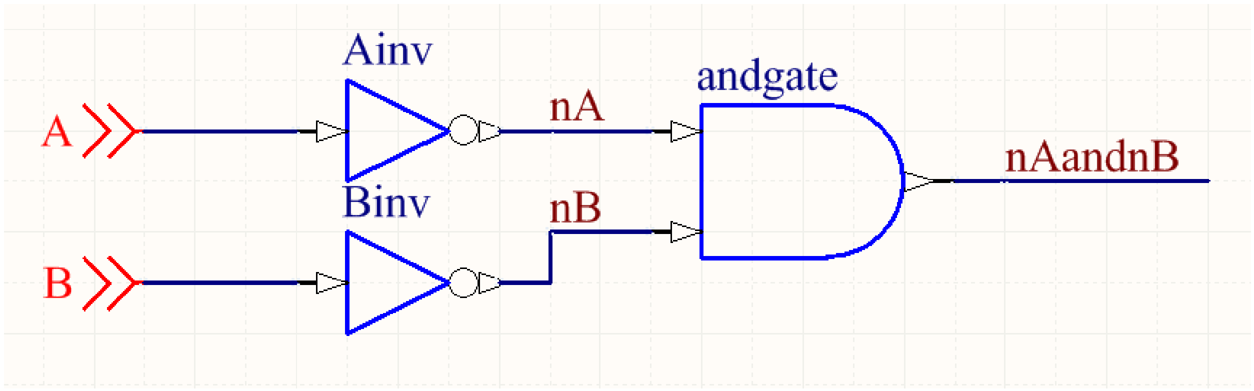
$$(\bar{A}\bar{B})$$

Schematic Capture

First, draw a schematic of each of the four equations. To help you, here is one for $(\bar{A}\bar{B})$



Then, give each gate and each wire a name. "A" and "B" are already named.



Translate to Verilog

Each element becomes a line of Verilog. Here is the schematic above, captured:

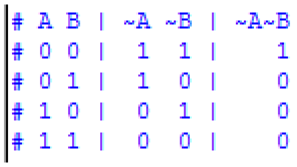
```
module hw1test;
reg A;                  // The input A
reg B;                  // The input B
wire nA;
wire nB;
wire nAandnB;
not Ainv(nA, A);        // top inverter produces signal nA and takes signal A, and is named Ainv
not Binv(nB, B);        // inverter produces signal nB and takes signal B, and is named Binv
and andgate(nAandnB, nA, nB);        // and  gate produces nAandnB from nA nB
                        // Place initial begin block here!
endmodule;
```

Add Test Signals and display outputs

You can now drive your module by adding test code within an "initial" block. The statements above show the structure of the module. The statements below poke at it. Place this before the "endmodule;" statement.

```
initial begin
$display("A B | ~A ~B | ~A~B ");                    // Prints header for truth table
A=0;B=0; #1                                         // Set A and B, wait for update (#1)
$display("%b %b |  %b  %b |    %b  ", A,B, nA, nB, nAandnB);
A=0;B=1; #1                                         // Set A and B, wait for new update
$display("%b %b |  %b  %b |    %b  ", A,B, nA, nB, nAandnB);
A=1;B=0; #1
$display("%b %b |  %b  %b |    %b  ", A,B, nA, nB, nAandnB);
A=1;B=1; #1
$display("%b %b |  %b  %b |    %b  ", A,B, nA, nB, nAandnB);
end
```

When you do do.do, you should see this:

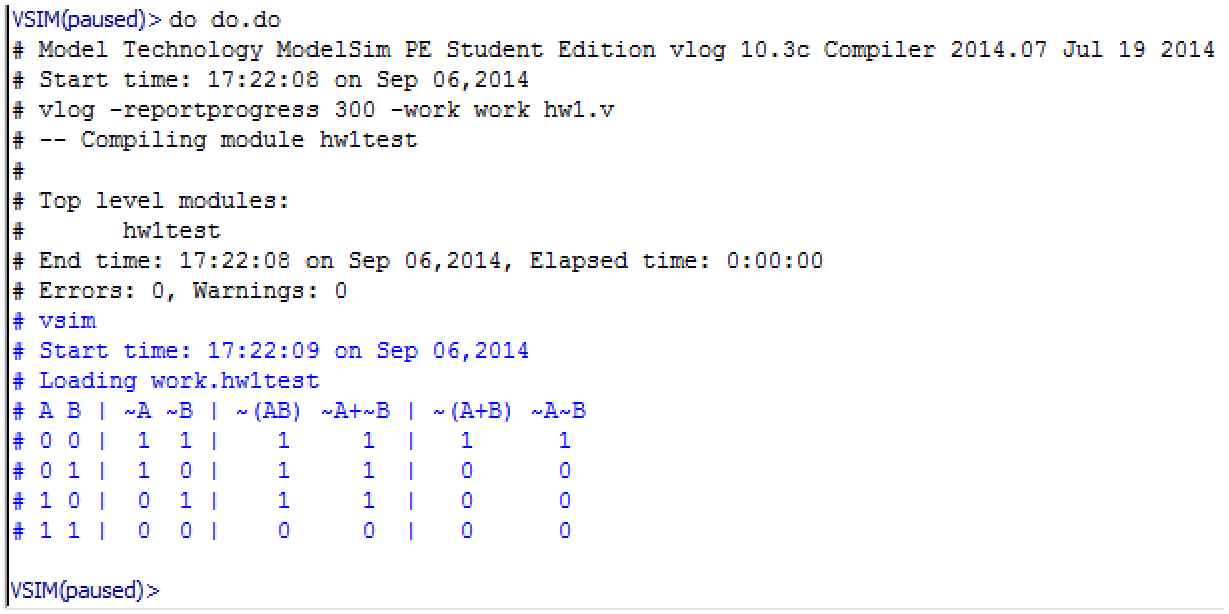


```
# A B | ~A ~B | ~A~B
# 0 0 |  1  1 |    1
# 0 1 |  1  0 |    0
# 1 0 |  0  1 |    0
# 1 1 |  0  0 |    0
```

Add the other 3 Equations

Continue in the manner described above to create the remaining three equations.

Your final result should look something like this:



```
VSIM(paused)> do do.do
# Model Technology ModelSim PE Student Edition vlog 10.3c Compiler 2014.07 Jul 19 2014
# Start time: 17:22:08 on Sep 06,2014
# vlog -reportprogress 300 -work work hw1.v
# -- Compiling module hw1test
#
# Top level modules:
#       hw1test
# End time: 17:22:08 on Sep 06,2014, Elapsed time: 0:00:00
# Errors: 0, Warnings: 0
# vsim
# Start time: 17:22:09 on Sep 06,2014
# Loading work.hw1test
# A B | ~A ~B | ~(AB) ~A+~B | ~(A+B) ~A~B
# 0 0 |  1  1 |    1     1  |   1      1
# 0 1 |  1  0 |    1     1  |   0      0
# 1 0 |  0  1 |    1     1  |   0      0
# 1 1 |  0  0 |    0     0  |   0      0

VSIM(paused)>
```

Submission

This homework will be visually checked off by the NINJAs. See Lyra, Derek, Jamie or Emily and have them check you off. There is no write-up or other deliverable.

Tips and Tricks

1) Save your Verilog file before you do your do file. Otherwise it will work on the stale version.
2) You can access your most recently run command by pressing the up arrow.
3) If a signal shows up as “z”, it means it is not being driven. Are you sure you wired it?
4) The display syntax is similar to what you might find in a printf style args function. The first input is a format string. %b means “interpret the next argument as a bit and display it here”.
5) “begin” and “end” are the equivalent of “{“ and “}” in C.