EE345 Assignment: Combinatorial Logic in Verilog

Schedule

Assigned: 8/30/2017
Due: 9/06/2017, 8AM

Please work alone on this assignment; I'd like to get each student familiar with the mechanics of simulating Verilog.

Background

This assignment entails running some combinatorial SystemVerilog code on the ModelSim simulator. ModelSim is packaged with the Altera tools (Quartus), and should be pre-installed on Wilkes workstations. It's also available by itself, if you want a copy on your own computer (https://www.mentor.com/company/higher_ed/modelsim-student-edition).

Completion of this assignment will entail:

- Starting ModelSim
- Loading and building the supplied SystemVerilog code
- Modifying the code
- Running it
- Saving a waveform

Details

To start ModelSim:

- Click Start Menu
- Find Intel FPGA tools folder
- In there, click ModelSim to start it

In ModelSim, we want to create a project:

- Click File/New/Project
- Assign a project name, and location on your drive (leave other settings at default)
- You'll see a "Add items to the Project" dialog, click "Create New File"
    - File Name: top.sv
    - Add file as type: SystemVerilog
    - Click OK
- Close "Add items to the Project" dialog

Now we want to edit the file, so double-click the top.sv line. Add the following code:

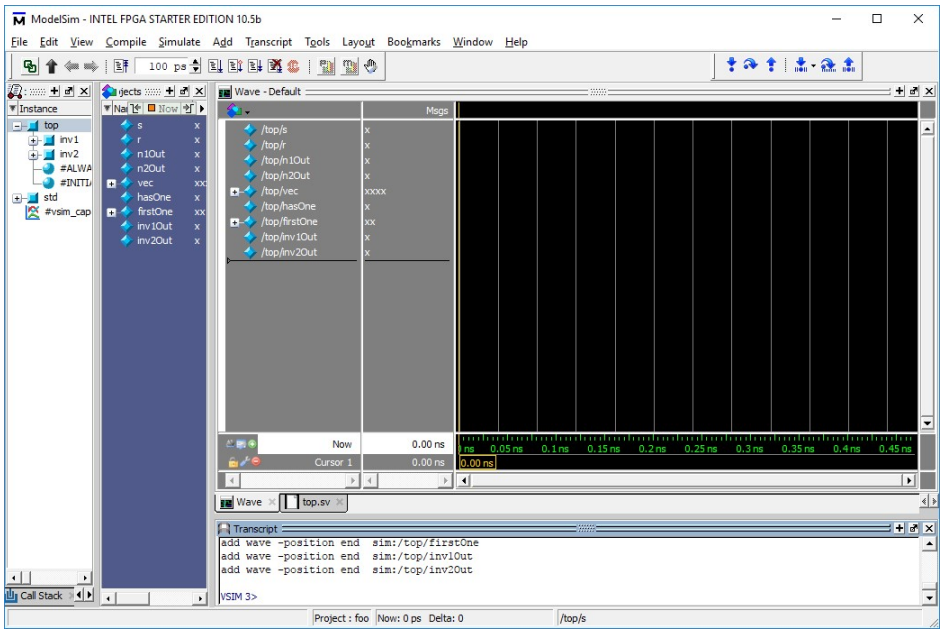
```
module Inverter(
                input logic  in,
                output logic out
                );
   assign out = ~in;
endmodule


module top();

   logic s, r;
   logic n1Out, n2Out;

   logic [3:0] vec;
   logic       hasOne;
   logic [1:0] firstOne;
   logic       inv1Out, inv2Out;


   Inverter inv1(
                 .in(n1Out),
                 .out(inv1Out)
                 );

   Inverter inv2(
                 .in(inv1Out),
                 .out(inv2Out)
                 );


   always_comb begin
      // Cross-coupled NAND gates
      n1Out = ~(s & n2Out);
      n2Out = ~(r & n1Out);

      // hasOne gets '1' if any bit is set in vec
      hasOne = (vec != 4'b0000);

      // Locate the first '1' bit in vec
      firstOne =
                vec[3] ? 2'h3 :
                vec[2] ? 2'h2 :
                vec[1] ? 2'h1 :
                0;
   end // always_comb

   // Run some test inputs through
   initial begin
      r = 0;
      s = 0;

      vec = 4'b0;

      #2;               // wait for 2 time units
```

```
        r = 0;
        s = 1;

        vec = 4'b0010;

        #2;

        r = 1;
        s = 0;

        vec = 4'b1111;

        #2;

        r = 1;
        s = 1;

        vec = 4'b0001;

        #2;
    end


endmodule
```

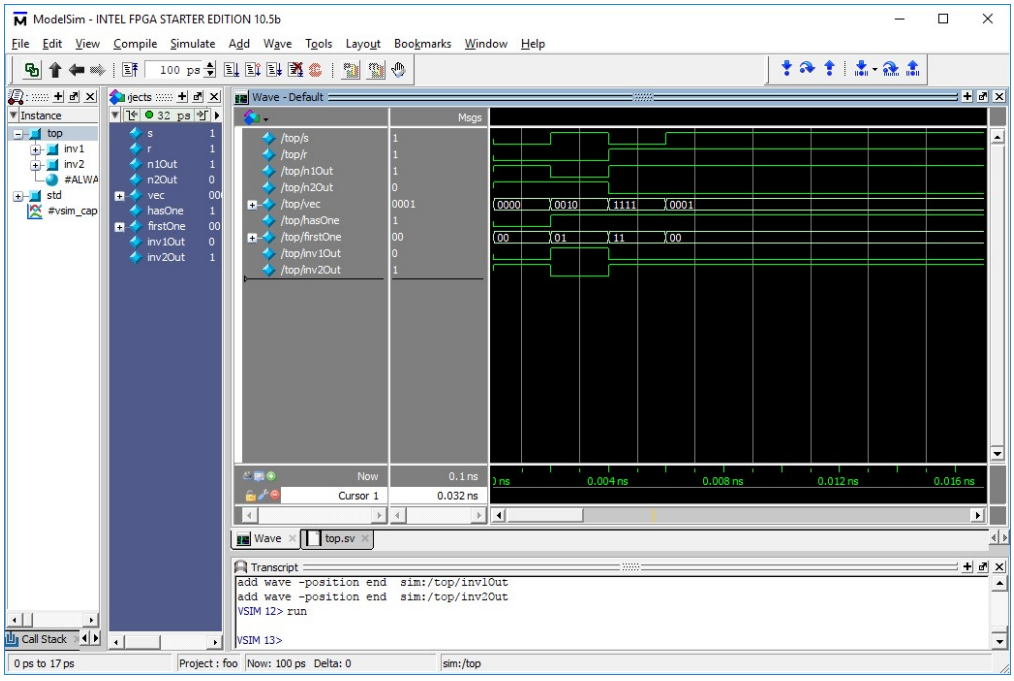
Save the file. Next we want to compile: click the menu item “Compile/Compile All”. You should see a message: “# Compile of top.sv was successful with warnings.”

Finally, we can simulate. Click the “Simulate/Start Simulation” menu item. The dialog will show a tree of modules. Click the cross next to “work”, choose “top”. Click OK. This should open a new window.

In the simulator window, click on the Wave tab to show waveforms. In the Objects pane, select the signals you want to show. I chose to display all, so my window looked like this:



Click the run button (right next to the “100 ps” box). You may need to zoom on the wave display. I ended up with this:



Now, you get to modify the code. Please do the following:

- Add an Inverter, call it “inv3”
- The input of inv3 should be the output of inv2
- The output of inv3 should drive a signal “inv3Out”
- Compile, simulate
- Capture a wave that shows inv1Out, inv2Out, inv3Out

Deliverables

Please save the following

- Listing of your Verilog code (just the first page, where you made changes)
- A PDF of the resulting waveform (showing the “inv*Out” signals)