

# CS/EE 5710/6710 -- CAD Assignment #1

Due Friday, September 4th, 5:00pm

Put assignments in the slot outside the SoC office

For this assignment you'll use *Digital VLSI Chip Design with Cadence and Synopsys Tools*. You'll need to look at Chapters 1-4 for this assignment. The chapters are tutorial in nature so you should be able to follow along as you're trying out the tools. This assignment looks like a lot to do, but most of it is just following along with the tutorial in the book.

## Assignment

**Step 1:** Look at Chapters 1 and 2 for information about configuring and starting the tools. Note also that the tool paths in Chapter 2 of the CAD book are close, but not exactly the right paths to use. In particular:

- Use the **cad-ncsu** script from **/uusoc/facility/cad\_common/local/bin/F09/cad-ncsu**
- Set your **LOCAL\_CADSETUP** variable to **/uusoc/facility/cad\_common/local/class/6710/F09**  
This is different than what the book says!
- Make a new directory in which to start up the tools. I recommend **IC\_CAD** in your home directory
- Make a new directory for Cadence under that **IC\_CAD** directory. I recommend calling it **cadence-f09** so that you can keep it isolated from other classes in the future that might use different versions of Cadence.
- Make sure to remember to link the **.cdsinit** file from **/uusoc/facility/cad\_common/NCSU/CDK-F09** to your **~/IC\_CAD/cadence-f09** directory

Here is the recommended sequence of steps (read the material in Chapter 2 also!). First you will need to log in to a CADE machine. Cadence tools on all the linux machines in the CADE lab.

- `cd`
- `mkdir IC_CAD`
- `cd IC_CAD`
- `mkdir cadence-f09`
- `ln -s /uusoc/facility/cad_common/NCSU/CDK-F09/.cdsinit cadence-f09`

Now edit your **.cshrc** (or **.tcshrc** if you use that) to add the following things (if you use bash as your shell, the syntax will be slightly different and you will have to figure that out because I don't use bash...):

```
set path = ($path /uusoc/facility/cad_common/local/bin/F09)
setenv LOCAL_CADSETUP /uusoc/facility/cad_common/local/class/6710/F09
```

Now that these lines are in your .cshrc (or other setup file), the next time you login or start a new shell you will be able to connect to your IC\_CAD/cadence-f09 directory and start up the Cadence tools with cad-ncsu.

- cd IC\_CAD/cadence-f09
- cad-ncsu

**Step 2:** Complete the first part of the Cadence Composer tutorial in Chapter 3 by making a new library (call your library CAD1) and designing a full adder using the standard cells in the **NCSU\_Digital\_Parts** library. Note that this is a different parts library than used in the CAD book tutorial! Pay attention to where your gates are coming from! You are welcome to copy the schematic in the CAD book, or make a different gate-level circuit that also implements a Full Adder. You should turn in the schematic for the full adder.

**Step 3:** Test your full adder using NC\_Verilog or Verilog-XL. You'll need the information from Chapter 4 for simulation, but you should be able to get away with just the first few sections of that chapter (Section 4.1.1 or 4.1.2 in particular). I know that you may not be a Verilog expert already, but you should be able to understand enough by looking at the examples to get things simulating. You should turn in the Verilog testbench for your full adder, the output from that simulation, and the timing waveform that results from the simulation.

**Step 3:** Create a symbol for your full adder and use it in building a 3-bit adder in a new schematic. You should turn in the schematic for your 3-bit adder.

**Step 4:** Simulate the 3-bit Adder using VerilogXL or NC\_Verilog and observe the timing diagrams. You should turn in the Verilog testbench, the output from the simulation, and the timing waveform that results from simulation.

**Step 5:** Complete the next part of the tutorial in Chapter 3 by designing a 2-input NAND gate using **nmos** and **pmos** transistors from the **UofU\_Analog\_Parts** library. Use **vdd** and **gnd** symbols for power and ground connections. Again, pay attention to where your gates are coming from! Create a symbol for the NAND that looks something like a NAND symbol. That is, modify the rectangle provided by Cadence to something that looks more like a standard NAND symbol. Simulate your NAND using VerilogXL or NC\_Verilog and observe the timing diagrams. Turn in the schematic for the NAND, the Verilog testbench, the simulation output, and the timing waveform that results from simulation.

**Step 6:** Using instances of your own 2 input NAND gate, build a circuit that implements the following Boolean function (don't minimize or manipulate the function, and don't use any gates except for your own NAND gate). Turn in the schematic for this circuit.

$$F = \overline{A}\overline{B} + \overline{A}C + \overline{B}C$$

**Step 7:** Simulate this circuit using Verilog XL or NC\_Verilog and observe the timing diagrams. Turn in the Verilog testbench, the simulation output, and the timing waveform that results from simulation.

## Instructions

For all of these simulation tasks, make sure that your Verilog test fixture uses "if" and "\$display" statements to check for the correct results in the simulation. You should be able to tell from running your test fixture whether the circuit is working correctly before you look at the timing waveforms (that is, if the circuit produces an incorrect output, an error message should be printed!). For these simulations, and for subsequent simulations in this class, you should either test things exhaustively (i.e. test for all possible input combinations), or describe on a separate sheet what tests you did run and justify why that is a good set of tests. It's unlikely that you'll be able to test larger circuits in future labs exhaustively so you'll have to put some thought into what to test and why that is a good set of tests. However, exhaustive testing will work fine for this assignment.

Also, make sure to use an **Asheet** frame from the **UofU\_Sheets** library on every schematic. Spend the time to make your schematics neat and orderly. Straighten out the wires, space out the components appropriately, don't over crowd, and generally make things look nice. Neatness counts when grading schematics.

Note that this assignment is to be done individually. We'll form teams later.

## Things to Turn In

Turn in hardcopies of:

1. Gate-level schematics of the Full Adder, 3-bit Adder, their Verilog test fixtures, simulation logs and timing waveforms
2. Transistor-level schematic of the 2 input NAND gate, Verilog test fixture, simulation log and timing diagram
3. Schematic of the Boolean function using your new NAND gate, the Verilog test fixture, simulation log and timing diagram

Information about the schematic capture tool, Verilog simulation, printing the simulation logs, printing the timing waveforms, etc. can all be found in the CAD book.

Make sure your name is easily visible on all the pages you turn in. Turn in assignments to the CS/ECE 5710/6710 box outside the SoC (School of Computing) front office.