

## **Lab 1: Introduction to Schematic Entry**

### **Check Off by Friday Jan. 25th**

**Summary:** In Lab 1 students will be introduced to the Cadence VLSI EDA software, complete a tutorial on the use of schematic capture and analog simulation tools, and develop schematics for INV, NAND, NOR, and XOR logic gates.

#### **Learning Objectives:**

1. Setup and become familiar with the Cadence software package
2. Discover the lab resources posted on the 410 website
3. Practice schematic entry with Cadence Virtuoso and functional simulation using Cadence Affirma analog simulator.
4. Observe the functional characteristics of several basic CMOS logic gates
5. Generate gate-level and CMOS schematics for a general logic function and observe its functionality.

**Resources:** These documents on the ECE410 website support this lab:

- Cadence Setup Guide
- UNIX Tips
- Tutorial A
- Guide to Schematic Entry
- Guide to Writing Stimulus Files
- Guide to 410 Lab Reports
- Guide to Capturing Images
- Troubleshooting Guide

**Estimated Time to Complete:** 4-6 hours plus report preparation

#### **Procedure:**

1. Log in to a computer in a DECS PC lab and use a web browser to go to the ECE410 lab website ([egr.msu.edu/classes/ece410/mason/](http://egr.msu.edu/classes/ece410/mason/)). Become familiar with the information available on this website.
  - You can view all of the lab documents from a web browser while you do your assignments, but you may also print the PDF files for any documents you wish.
  - Note the **Troubleshooting Guide**. It is not directly referenced in any of the steps below but includes many useful tips for the Cadence tools that may be helpful in the future. If you do run into trouble, your first step should be to check this guide.
2. Although we are using PCs, Cadence runs on the UNIX platform and you will need to navigate through UNIX as you work on labs. If you are unfamiliar with UNIX, visit <http://www.egr.msu.edu/decs/support/unix/> to review basic commands.
3. Complete the instructions in the **Cadence Setup Guide**
  - Contact the instructor or a TA immediately if you have any problem with your class account. If you do not have a class account for some reason, you can also go to the DECS office for help.

- For information on tweaking your UNIX setup to simply using Cadence, review the [Unix Tips](#) guide.
4. Prepare to follow a tutorial on creating schematics using Cadence.
    - Be sure to do this assignment in your 410 class directory and not your personal home directory.
    - Review the [Guide to Schematic Entry](#) before completing the tutorial for some ideas on how to improve your schematics.
  5. Complete [Tutorial A](#) to create a CMOS INV gate.
  6. Follow the procedure of [Tutorial A](#) to construct the schematic of a 2-input NAND gate. Name the cell NAND2.
  7. Create a symbol for the NAND2 cell.

NOTE: Experiment with the Cadence tool to find out how construct symbols that look like the standard symbol for a NAND gate (and NOR and XOR gates, respectively).
  8. Simulate the functional characteristics of the NAND2 cell.

NOTE: Confirm the gate is functioning properly by construct input signals to test the logic output at all possible input combinations. Refer to a truth table if necessary and consult the *Testing Logic Truth Tables* section of the [Guide to Writing Stimulus Files](#).
  9. Repeat steps 6-8 to construct a schematic and symbol for a 2-input CMOS NOR gate and verify its logic function by simulation. Name the cell NOR2.
  10. Repeat steps 6-8 to construct a schematic and symbol for a 2-input static CMOS XOR gate (not the transmission gate version) and verify its logic function by simulation. Name the cell XOR2. Any inverted signals (e.g., A' and B') should be generated within your cell using inverters constructed at the transistor level.
  11. Generate the truth table for the function  $F = \overline{A \cdot (B + C)}$ .
  12. Construct the schematic for *transistor-level* CMOS circuit for the function F (don't use INV, NAND2, NOR2 gates). Simulate to verify that all possible input combinations produce the output expected from the truth table. Show the stimulus file to a TA during check off.
  13. Print the Lab 1 Grading Sheet and meet briefly (~5 min.) with a TA to check off your lab by 4pm on Friday. Lab check offs are worth 5 of the 25 points for this lab assignment.
  14. Construct a *Brief Report* of this assignment using the [Guide to 410 Lab Reports](#). Refer to the [Guide to Capturing Images](#) for instructions on gathering images from Cadence. Be sure to include responses to the Discussion Topics below. The report will be graded by a TA for 15 of the 20 points for this lab assignment. Brief Report are due by the beginning of class on Monday but can be turned in to a TA earlier.

**Deliverables:**

*The following items must be shown to a TA during the check off procedure and/or included in your brief report of this lab assignment.*

		check off	report
1	Schematic of a CMOS INV, NAND, NOR, XOR	X	X
2	Symbol for INV, NAND, NOR, XOR	X	X
3	Annotated simulation output for INV, NAND, NOR, XOR		X
4	Truth table for function 'F'		X
5	Transistor-level schematics for function 'F'	X	X
6	Stimulus file used to simulate function 'F'	X	
7	Annotated simulation output of function 'F'	X	X

**Discussion Topics:**

*Include type-written responses to the following discussion items in your brief report.*

- 1. What is the URL for the ECE 410 website?*
- 2. What is the UNIX command to display files in a directory?*
- 3. To which supply voltage is the bulk/body terminal of a pMOS device connected?*
- 4. What are the width (W) and length (L) of a minimum sized transistor in the AMI C5N CMOS process?*
- 5. What input signal rise and fall times were used in your simulations? Include the units of measure in your answer.*
- 6. What menu selection in the Waveform Window will split plotted signals into individual plots?*
- 7. Using spectre syntax, write the stimulus file line for a 100MHz, 3V square wave voltage source with a 50% duty cycle and 0.01nsec rise and fall time.*
- 8. Comment on how you would expect a simulation of a gate-level version of function F, built with logic gates rather than transistors, to compare with your results for the transistor-level simulation.*