Laboratory 2: Basic Boolean Circuits

Objective: The objective of this experiment is to develop methods for evaluating the performance of Boolean circuits. Emphasis will be placed on the basic CMOS inverter but the concepts are applicable to larger circuits. Time domain behavior of the inverter will be considered.

Part 1.1: Introduction

A digital inverter with Boolean input A and Boolean output Y is shown in Fig. 1a

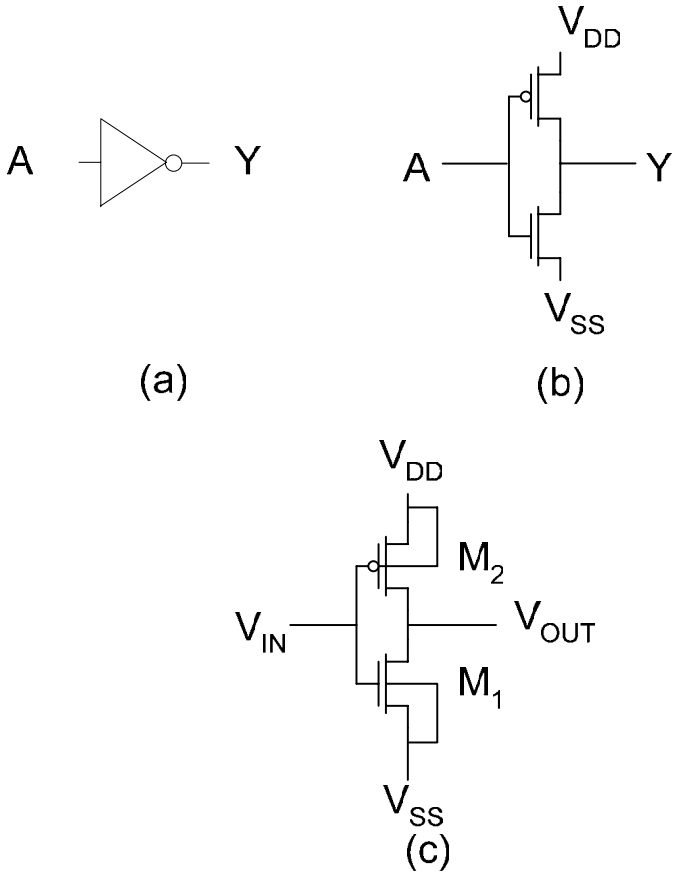

Fig.1 Digital Inverter a) Gate Representation b) Simple Transistor Implementation c) Transistor Implementation Showing Bulk Connections

This gate level representation provides no detail about the underlying circuit used to realize the gate. Fig. 1b shows transistor level detail but does not show how the bulk is connected in the circuit. The representation in Fig. 1c shows all connectivity of the inverter. In the latter view, the input and output variables are labeled as voltage variables rather than as Boolean variables.

Part 2 Simulation of a CMOS Inverter

In this section we will focus on the creation of a schematic of a CMOS inverter and on the simulation of this inverter in Spectre.

Part 2.1 Attaching technology information

In the last lab, we simulated a simple circuit consisting only resistors and capacitors. In this lab, we will implement an inverter using transistors. To use the correct models for the transistors, we need to first choose which "Technology" we would like to use. By choosing a technology, we decide which semiconductor manufacturer will manufacture our design if we choose to fabricate it as an IC. For the purposes of this course, we will use the 0.5µ process from AMI semiconductor. When the attach process is complete, all the AMI 0.5µ process information will automatically be used in our designs. The process information includes critical parameters such as the minimum device size, the available layer masks, and the supply voltage level.

To attach a library to a specific technology, right click on the desired library name in the Library Manager and choose **Attach Tech lib…** In the form that pops up, choose the technology library to be **AMI 0.60u C5N** and click on ok. Check CIW to see if the technology file was attached successfully.

Part 2.2 Creation of a Schematic

Consider an implementation of the CMOS inverter of Fig. 1c in the AMI 0.5µ CMOS process. Use a 5V power supply and size the devices M_1 and M_2 with the drawn dimensions given in Table 1.

Table 1: Device Sizes for Inverter

	W	L
M1	1.5µ	0.6µ
M2	4.5µ	0.6µ

Create a schematic view of the inverter in a new cell in your lablib library (for convenience use the name inverter for this cell). For the NMOS and PMOS transistors, use the cells nmos4 and pmos4 from the **NCSU_Analog_Parts** library, respectively. When editing the transistor properties, the lengths and widths need to be in the fields labeled "Width" and "Length", not in the "Width (grid units)" or "Length (grid units)" fields. When entering numbers in the Cadence design and simulation environment, you can use engineering notation suck as k (10^3), M (10^6), m (10^{-3}), u (10^{-6}), n (10^{-9}), etc. The suffix should be written right next to the number without spaces. For example, $10*10^{-9}$ can be entered as 10n or 10e-9 but not as 10 n.

The labeled voltages on the schematic of Fig.1c include the voltage V_{SS} which is often set at ground but for generality we have designated it as V_{SS}. These labeled voltages must be designated with pins on the schematic you enter in Cadence. Pins can be of different types depending upon the intended use of the pin. The voltages V_{DD} and V_{SS} should be bidirectional pins. The voltage designated as V_{IN} should be in input pin and the voltage designated as V_{OUT} should be designated as an output pin.

Part 2.2 Symbol Creation

To re-use the inverter we just created efficiently, we need to create a symbol for our schematic design. Go to **Design→Create Cellview→From Cellview**. If you think the information automatically filled is correct, press OK. A window with the symbol will be opened. Change the symbol view to make it look like an inverter. Make sure you keep the pins consistent.

Part 2.3: Inverter simulation

Now that we have a symbol for our inverter, let us set up the test bench. Create another one called **test_inverter** to test the design. The test bench is shown below.

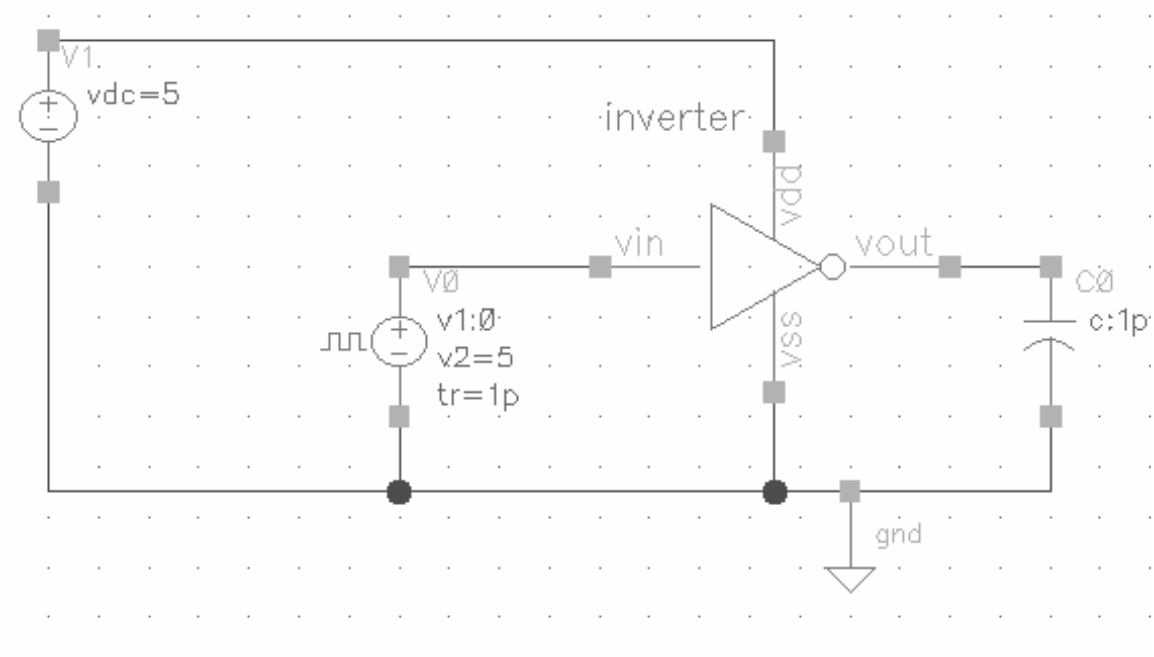

Figure: Inverter simulation test bench

Use the voltage sources from analogLib. For the pulse input source, you can use either the vpulse or the vsource component from the analogLib library. You will need to apply an input square wave of frequency 1MHz with magnitude of 5V and rise and fall times of 1 pico seconds each. To instantiate your own inverter, use the instantiation procedure as you would for any other component from analogLib but choose the inverter cell instead from your own lablib library. Set the dc power supply voltage to 5V and the load capacitor to 1pF. **Check and Save** your work and correct all errors or warnings.

Open the Analog Design Environment and check the Setup parameters first. Click on **Setup → Simulator → Directory → Host** and check the Project Directory. This is the directory where simulation results are stored. It is recommended that you always store your simulation data on the local hard drive. To ensure this, set your project directory to */local/username*. If you see */home/username* then you should be aware that some simulations might run unusually slow and could take your entire ENGR disk quota.

You can make this choice the default for all of your future Cadence sessions by creating a new file called `.cdsenv` in your home directory and adding the following line to it (with your username).

```
asimenv.startup  projectDir  string  "/local/your_username"
```

Set up the transient analysis to run long enough for 5 complete input cycles. Select the input and output voltages for plotting and start the simulation. Did you create labels for those nodes of interest (recall how we used the labels vIn, vMid, and vOut in lab1)? Debugging and evaluation of results become easier with labels.

- Print a copy of the inverter schematic, of the test bench schematic, and a plot of your simulation output.

Part 3: Cascaded Inverters

We will now cascade two inverters and analyze how the signal propagates from one stage to the other. Modify your inverter_test cell so it looks like the following figure:

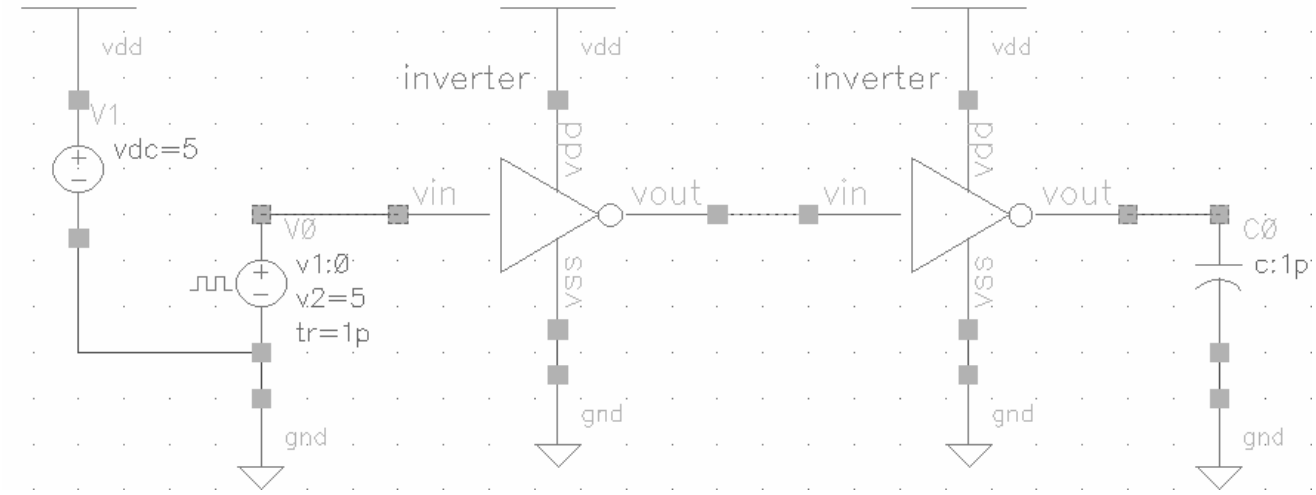

Figure: Cascaded inverter simulation

The "antenna" on top of the vdc source and the power supply pin of the inverters is a visual aid and a method of connecting multiple power supply nets together. The antenna shaped cell is called vdd and is available from analogLib.

Run a simulation and plot the three voltages of the circuit: input of inverter 1, output of inverter 1, and output of inverter 2. Label these nodes before running the simulation

- Compare the output of inverter 2 with the input of inverter 1. Do they look the same? Do you see any loss of *signal integrity*? Is there any noticeable delay?

Part 3.1: Inverter driving a load

Let's go back to the test bench at the beginning of this lab: one inverter driving a capacitive load. Increase the load to 10 pF.

- What do you notice at the output? What if you have a 100 pF load? Or a 1uF load? Explain what you think happens.

Part 3.2: Inverter driving a load (revisited)

Increase the width of the transistors of the inverter by a factor of ten (i.e. Wp=45u, and Wn=15u). Let's revert back to a 10pF and simulate

- How does the output look compared to that of Part 2.1 with a same load? Explain.