Lab 2: Basic Boolean Circuits

Brittany Duffy
EE 330- Integrated Electronics
Lab Section B
Professor Randy Geiger
1/31/13

Brittany Duffy

Introduction

The main goal of this lab was to become familiarized with the methods for evaluating Boolean circuits. Basic CMOS applications were the focus of this lab. However, these concepts are very relatable to much larger circuit applications. The figures below show digital inverters:

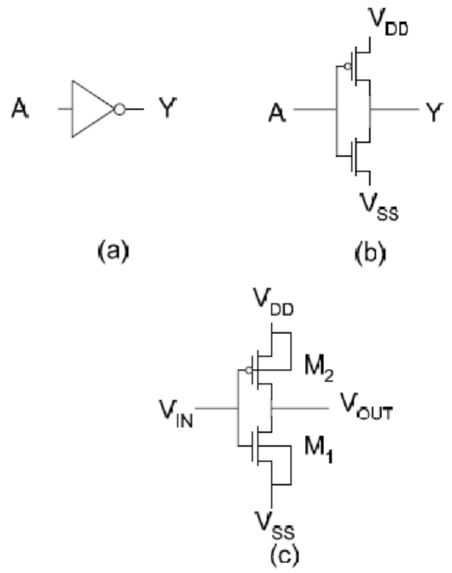


Figure 1: Digital Inverter a) Gate Symbol b) Transistor Implementation c) Transistor Implementation with Bulk Connections

Figure 1c was used most heavily in this lab due to its detailed description of connectivity in the CMOS inverter. Note: The input and output variables are labeled as voltage variables rather than Boolean variables as seen in Fig. 1b. The time domain behavior of the inverter was also considered during this entire lab. This report will demonstrate my experiment and observations throughout lab 2.

Procedure and Results

Part 2: Simulation of a CMOS Inverter

In the first step, I opened the Virtusuo software via the terminal.

Part 2.1: Attaching technology information

Next, I took steps to implement an inverter using transistors. I created a new cell called 'inverter' in schematic view. I attached a library to a technology by right clicking on my library 'lablib' and chose *Attach Tech library*. I then chose the technology library name to be *AMI .6u C5N*.

Part 2.2: Creation of a Schematic

This section of the lab involved creating the implementation of the CMOS inverter of Fig. 1c. The size of the devices M_1 and M_2 with the drawn dimensions given in Table 1.

	W	L
M_1	1.5u	0.6u
M_2	4.5u	0.6u

Table 1: Inverter Device Sizes

I created a schematic view of the inverter in a new cell in my lablib library. I called it 'inverter'. Using nmos4 and pmos4 from the *NCSU_Analog_Parts* library, I created the NMOS and PMOS transistors. I designated the pins Vss and Vdd by going to Add → pin. I made sure they were both bidirectional (inputOutput) pins. Vin is an input pin. Vout is an output pin.

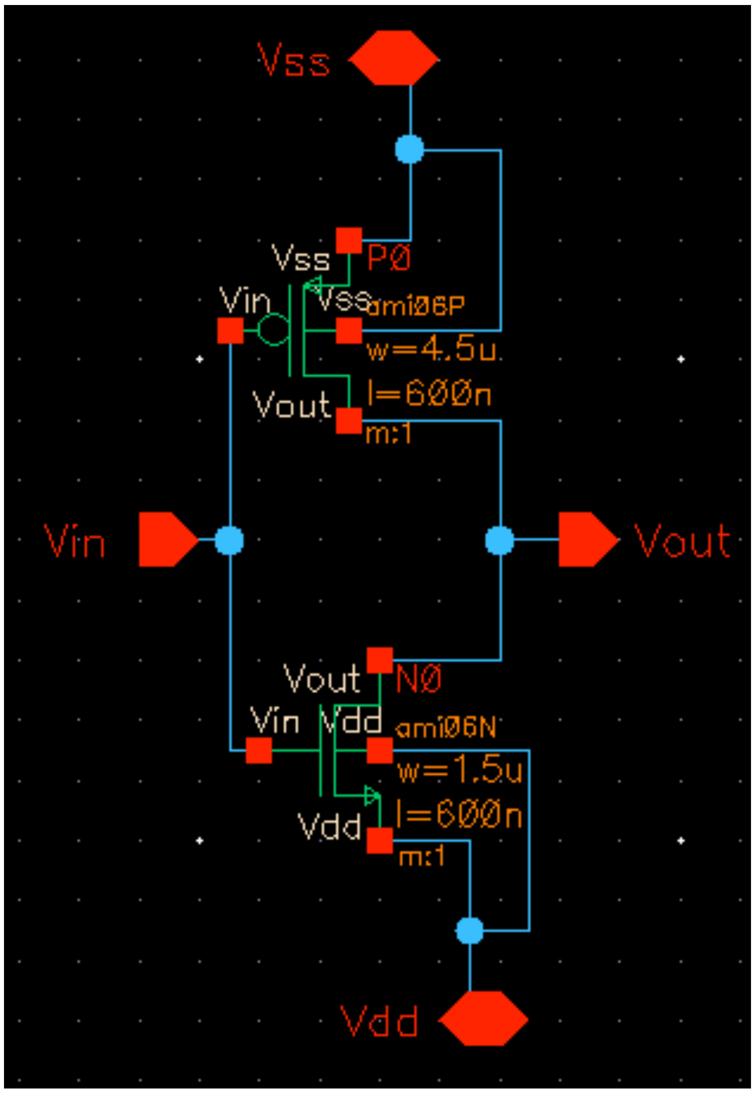


Figure 2: Inverter Schematic

Part 2.3 Symbol Creation

After the schematic was finalized, it was time to create a symbol for the design. In order to do this, I went to **Design→Create Cellview→From Cellview**. Using various shapes, I created a symbol very similar to Fig1a. You can see the symbol below in Figure 3.

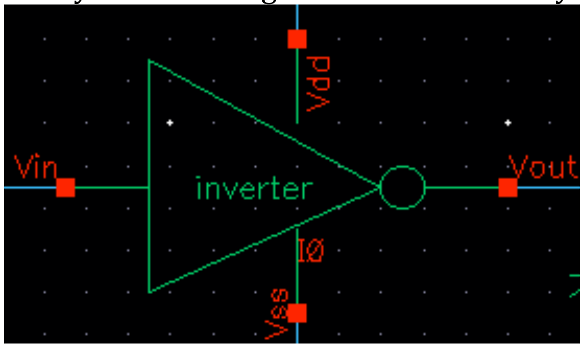


Figure 3: Inverter symbol after alterations

Part 2.4 Inverter transient response simulation

Setting up the test bench was the next step after making the inverter symbol. I created another schematic called 'test_inverter'. Figure 4 shows the finalized Inverter simulation test bench. All voltage sources are from analogLib. vpulse was used with a frequency of 1MHz. Therefore, the period was set at 1/1MHz = 1u secs. Voltage 1 had a magnitude of 5V with rise and fall times of 1 pico seconds. In order to instantiate my inverter symbol, I chose it from my lablib library. The dc power was set to 5V and the load capacitor to 1pF. Lastly, I included labels for significant wires. A Check and Save was done when finalized.

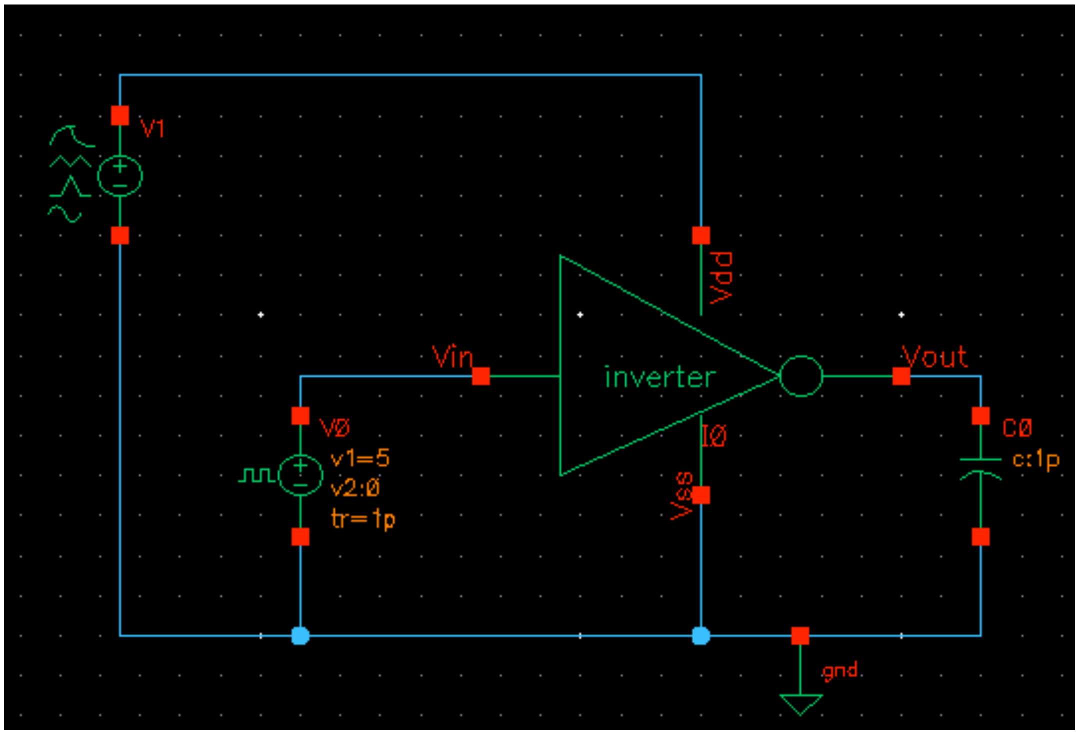


Figure 4: Test Bench Schematic

Analog Design Environment (ADE) was then setup by clicking on **Setup** → **Simulator** → **Directory** → **Host** and checking the Project Directory. I set the simulation results to be stored to /local/baduffy/cadence/simulation.

The transient analysis was set up to run for 5 complete input cycles. The input and output voltages were selected for plotting the simulation and then the simulation began. Figure 5 shows the results.

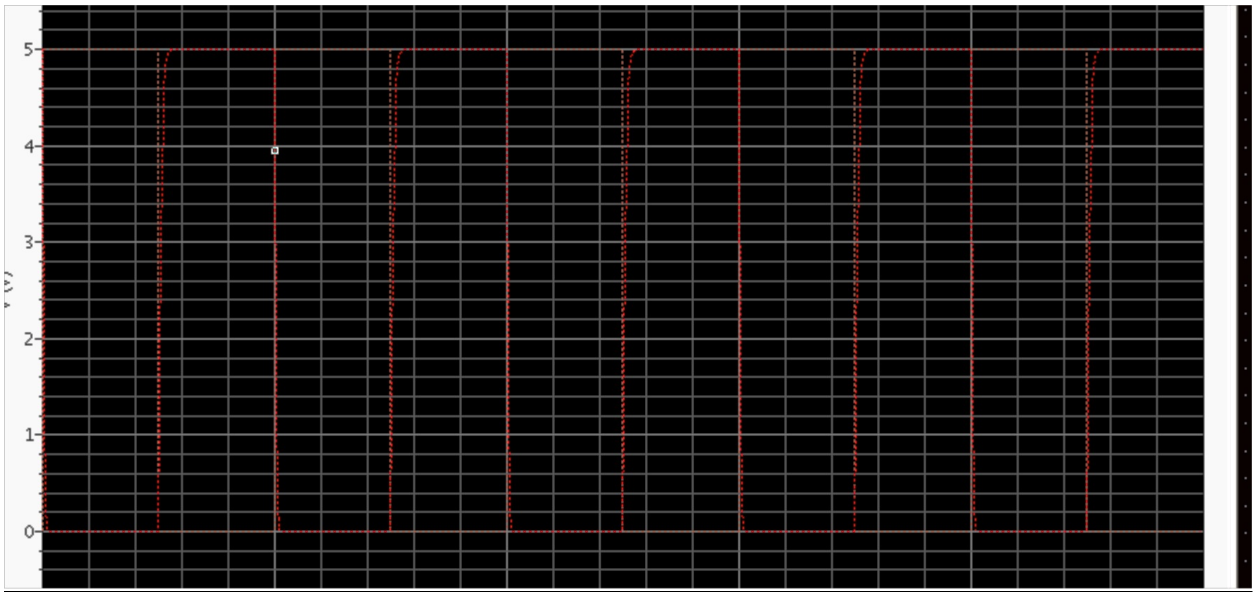


Figure 5: Transient analysis of input and output voltages

Part 2.5: Inverter transfer characteristics simulation

For this part, I modified the source and simulation environment to obtain the dc transfer characteristics of the inverter for input voltages between 0V and 5V. At the value 2.9 Vin, the input and output are equal. The max value of Vin that can be applied and still keep the output near 0V is around 4 V. The minimum value of Vin that can be applied and still keep the output near Vdd is around 1V. You can see the dc transfer characteristics on the next page in Figure 6.

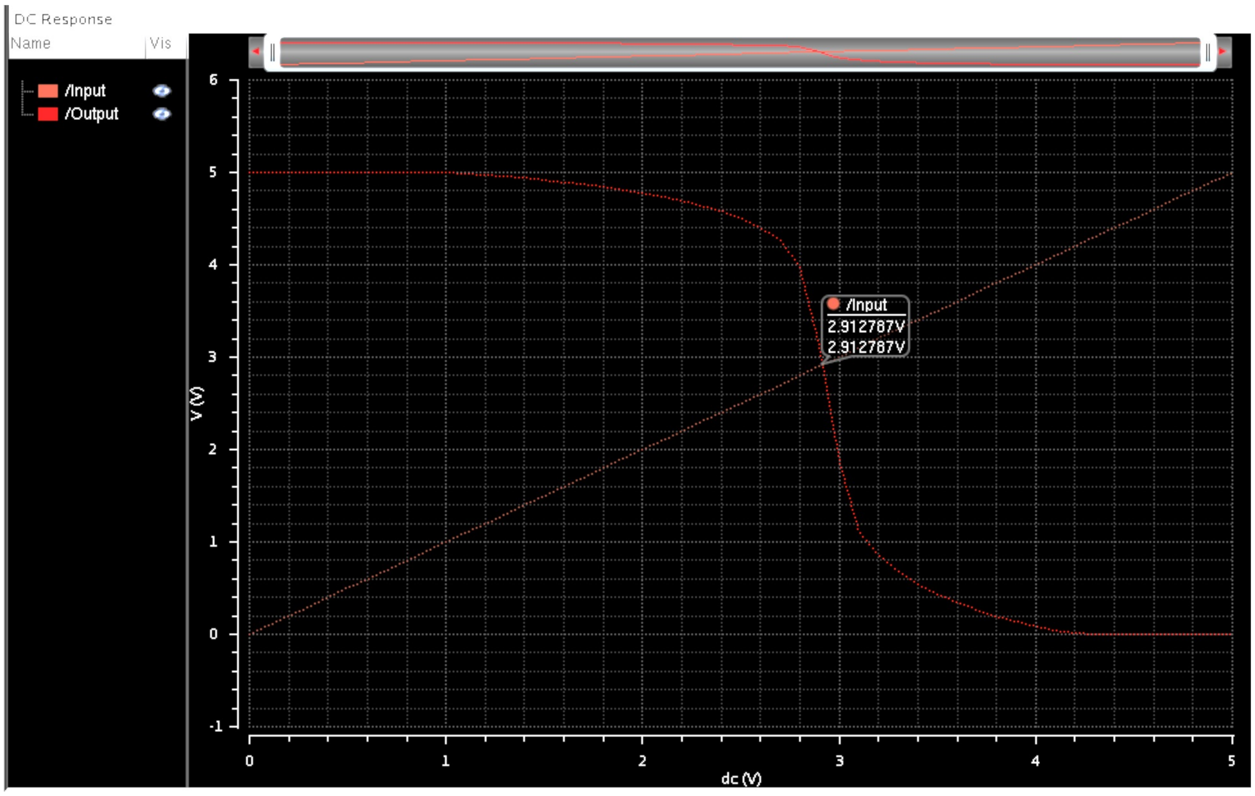


Figure 6: DC Analysis

Part 2.6: Inverter driving a load

In this section, I increased the load to 10 pF. From Figure 7, you can see the delay time is longer than using a 1 pF capacitor. Since $\tau = RC$, increasing C increases τ.

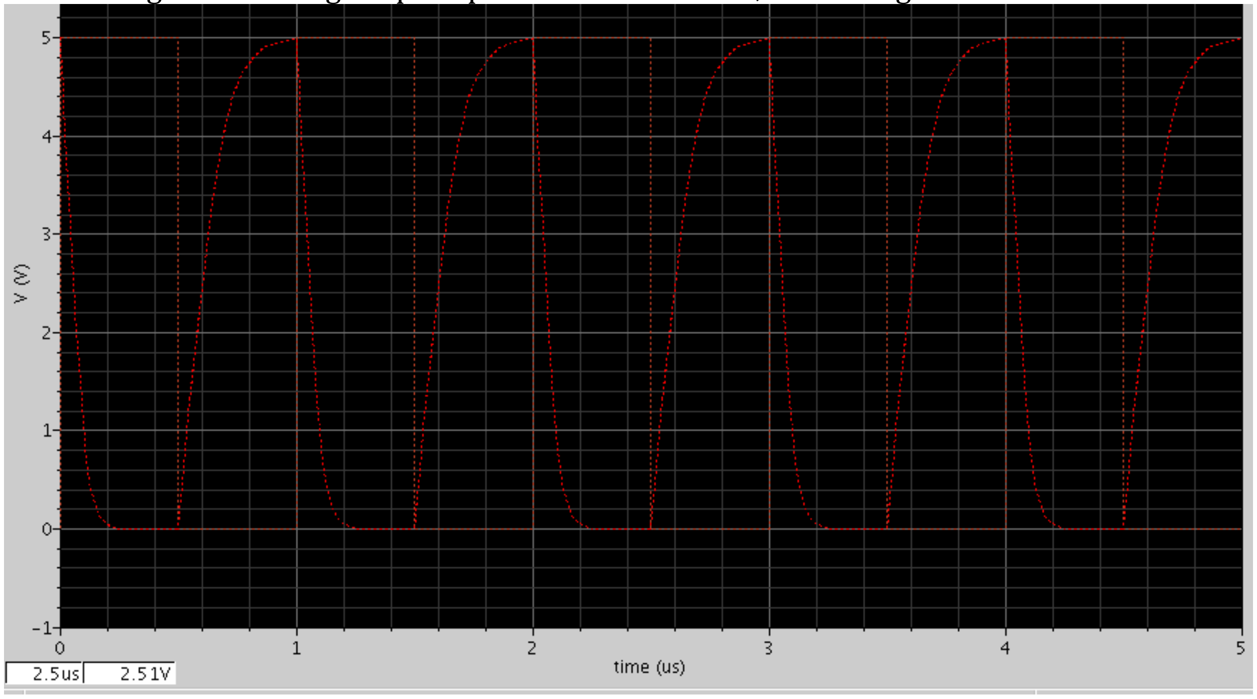


Figure 7: Transient Response with 10pF capacitor

Figure 8 shows the transient response to the increased 100 pF capacitor. As you can see, the time delay becomes even larger. Consequently, the capacitor does not have time to get fully charged and discharged.

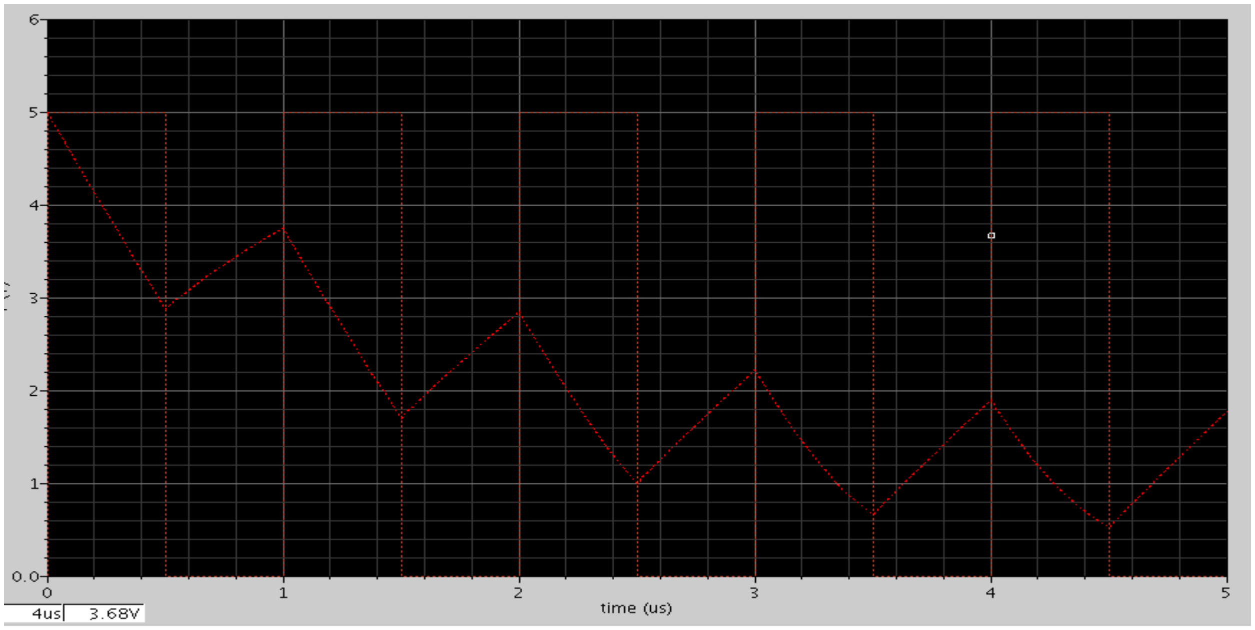


Figure 8: Transient response with 100pF capacitor

Increasing the capacitance to 1uF gives the result in Figure 9.

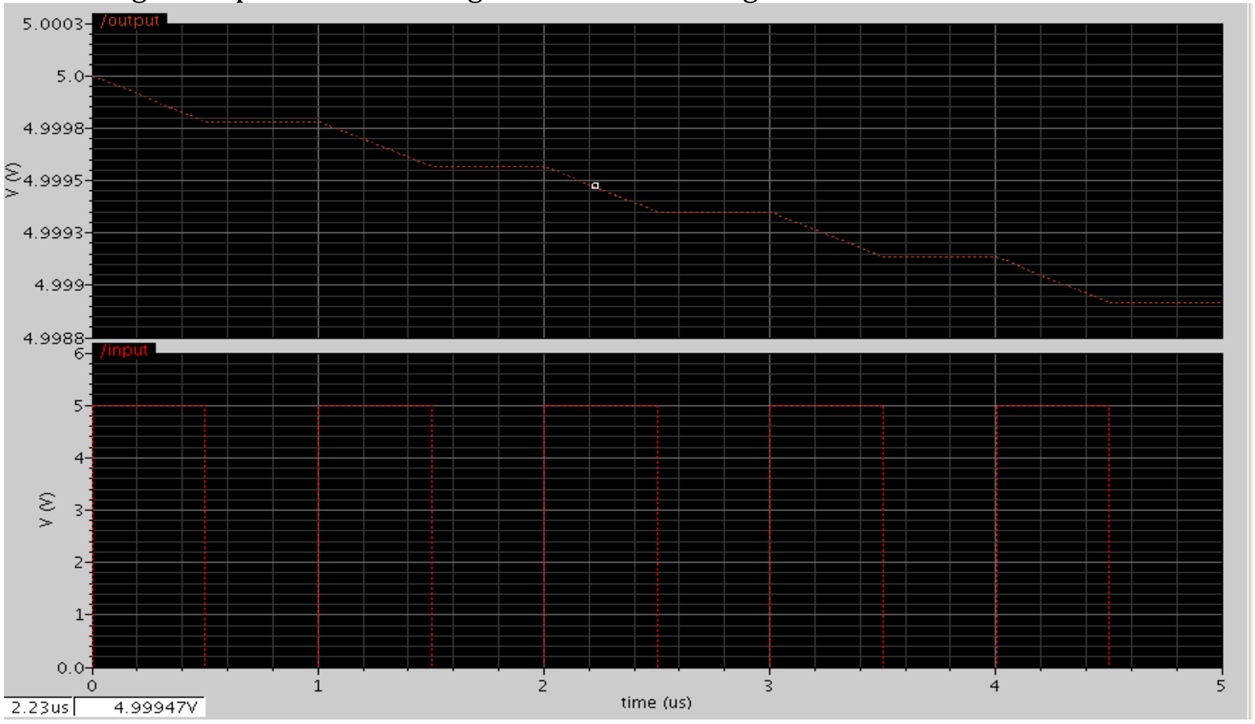


Figure 9: Transient Response with 1uF capacitor

Next, I reverted the load back to 10pF and increased the width of the transistors in the inverter by a factor of ten. $\tau = \frac{\rho LC}{WH}$, so increasing W by a factor of 10 makes τ become smaller.

Part 3: Cascaded Inverters

In this next section, I cascaded two inverters and analyzed how the signal propagates from one stage to the other. The following figure is the finalized cascaded inverter simulation.

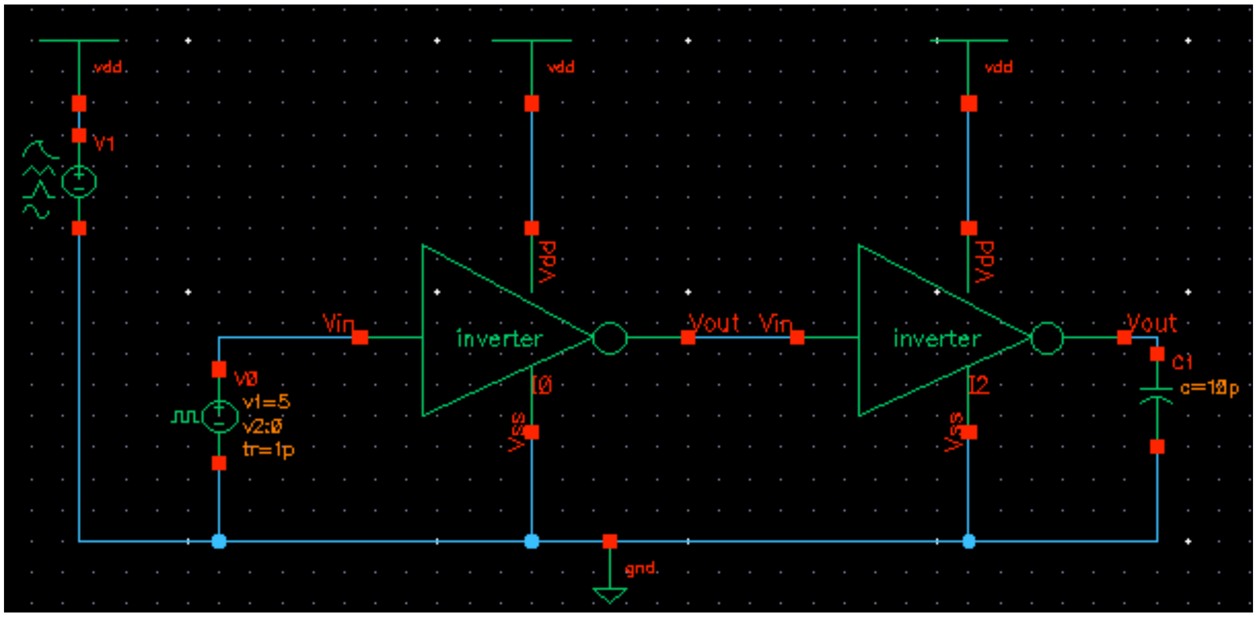


Figure 10: Cascaded Inverter Schematic

I then ran a simulation plotting the three voltages of the circuit at the input of inverter 1, the output of inverter 1, and the output of inverter 2. Figure 11 (on the next page) shows Vout1 and Vout2 being very similar. However, a small loss of signal integrity in Vout2 can be found with a delay.

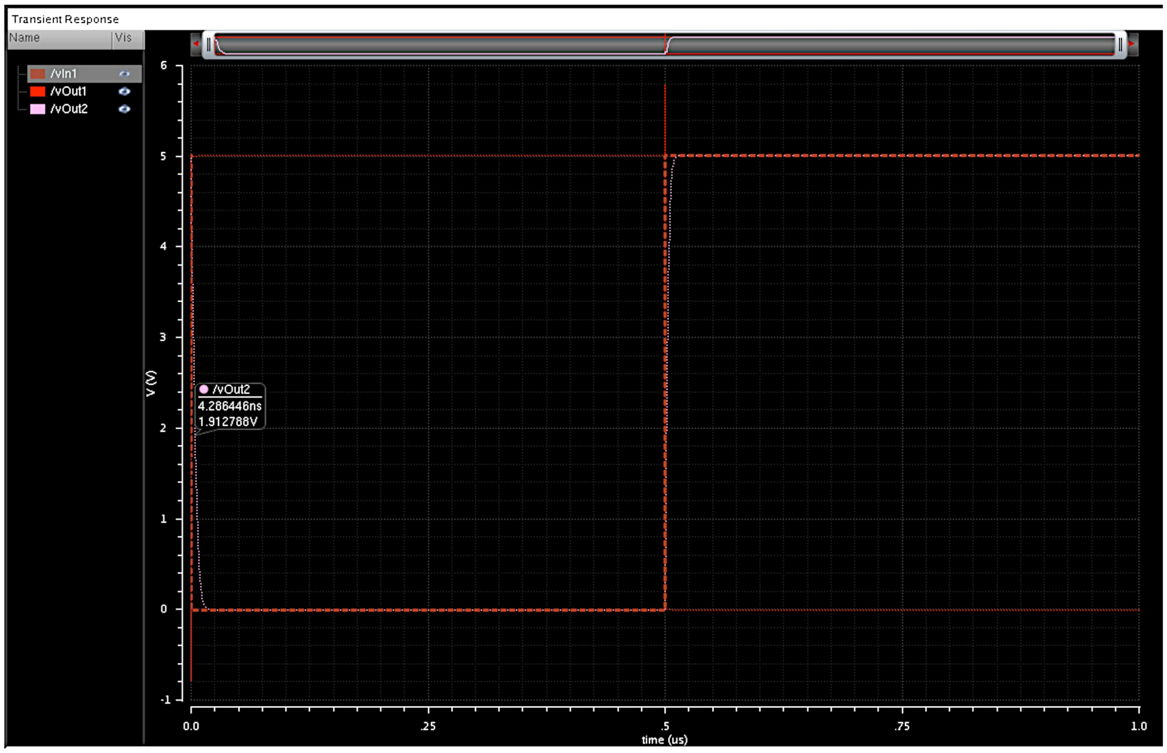


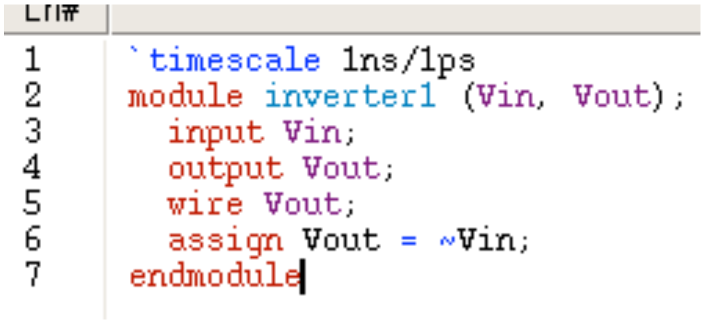
Figure 11: Transient Response of Cascaded Inverters

Part 4: Inverter Simulation with Verilog HDL inside ModelSim

In future labs, we will be using Verilog to simulate and design digital circuits and systems. To have some practice, I went through a tutorial on how to simulate the digital inverter using Verilog HDL in ModelSim software. During this tutorial, I was able to run ModelSim, create an inverter behavioral file in Verilog HDL, create a testbench, and run a simple simulation.

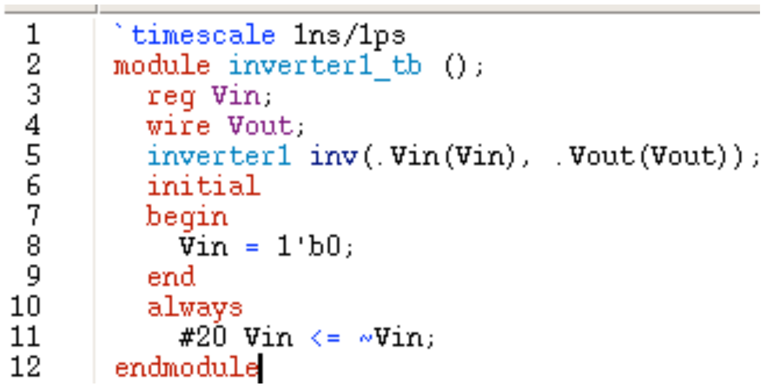
1. First, I downloaded ModelSim_env.txt from the Lab page. I then opened it using the terminal.
2. I created a new project and called it 'inverter'.
3. Next, I added a new file to the project.
4. Once the new file 'inverter1.v' was made, I double-clicked on the file to show its contents. Figure 12 shows the content of the file.
5. Adding a testbench was the next step. To add 'inverter1_tb.v" the same steps were taken from before in 'inverter1.v'. The contents were then added as you can see in Figure 13.
6. After saving both 'inverter1.v' and 'inverter1_tb.v', I checked the syntax of both files. In order to do so, I clicked on the *Compile* Menu and selected *Compile All*.
7. After both files compiled correctly, I simulated the design by clicking on the simulate menu and chose *Start Simulation*. I selected 'inveter1_tb' and unchecked the 'optimize design' box.
8. After clicking OK, I created a simulation waveform window. I clicked on the *View* menu, chose *New Window*, and then chose *Wave.* In order to view the signals, I dragged the signal from the middle pane to the waveform window.

9. After clicking the Run icon, the simulation was complete. Figure 14 shows the results.



```
`timescale 1ns/1ps
module inverter1 (Vin, Vout);
  input Vin;
  output Vout;
  wire Vout;
  assign Vout = ~Vin;
endmodule
```

Figure 12: Inverter1.v



```
`timescale 1ns/1ps
module inverter1_tb ();
  reg Vin;
  wire Vout;
  inverter1 inv(.Vin(Vin), .Vout(Vout));
  initial
  begin
    Vin = 1'b0;
  end
  always
    #20 Vin <= ~Vin;
endmodule
```

Figure 13: Inverter1_tb.v

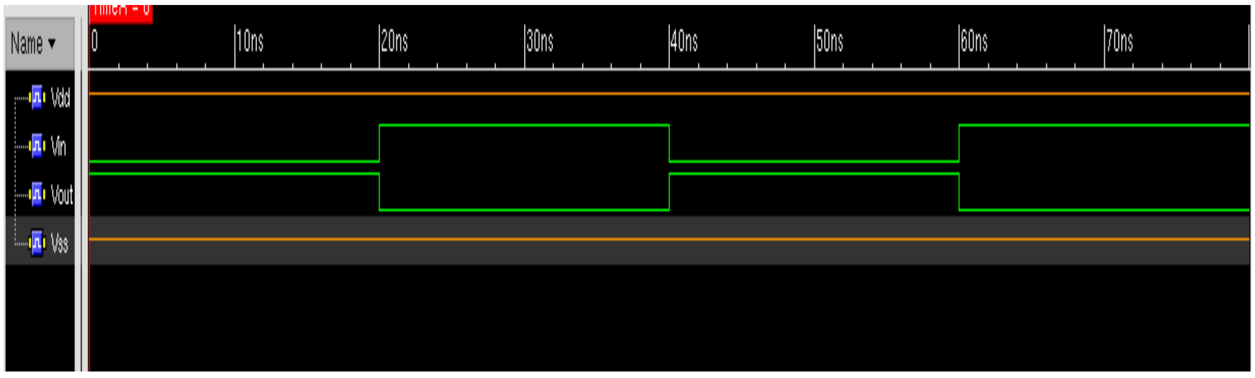


Figure 14: Inverter1_tb Simulation

Conclusion

After this laboratory, I feel much more comfortable in working with the Virtuoso software as well as ModelSim. In this lab, I investigated methods for evaluating the performance of Boolean circuits. I created a basic CMOS inverter through both Schematic and Verilog to simulate a Boolean circuit. I foresee this lab being extremely resourceful in future labs dealing with complex components.