Miniproject 1: Making an AND gate using XSCHEM and Magic

Kristtiya Guerra

February 2021

1 Introduction

The purpose of this project is to get used to using digital software, such as Xschem and Magic VLSI, to create functional schematics and layouts. We will also learn how to make hierarchical schematics and layouts using the respective software. Our goal is to create an AND gate using both software. By the end of this project, the netlists from both the Xschem schematic and the Magic VLSI software should match when compared through Netgen's layout vs. schematic (LVS) function.

Link to view repository: https://github.com/Kristtiya/MAD_VLSI/tree/main/MiniProject1-AND

2 Layout of Gate

In order to begin making a layout, we need to first determine what components we will be using and how we will be wiring out circuit. An AND gate is comprised of two major components, a NAND gate followed by an inverter. Figure 1 shows the most well-known layout.

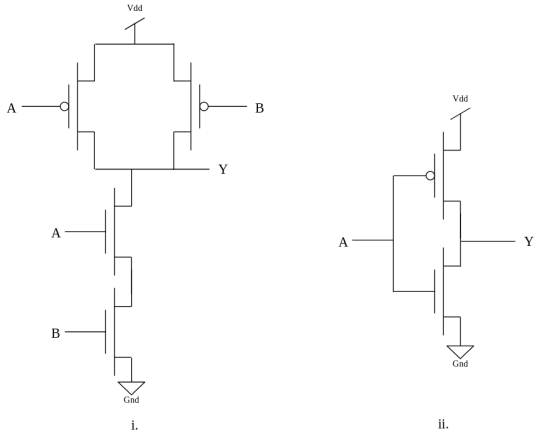

Figure 1: Basic schematics for a NAND (i) and an inverter (ii)

The issue with these two models is that they don't translate well onto a circuit when it comes to layout. More space is taken up by these individual components, and when they go into series, there is even more empty space being wasted. In order to combat this, we can approach drawing this circuit by putting the transistors in the general space they will be located in relation to other parts of the circuit. This is called a layout-based schematic.

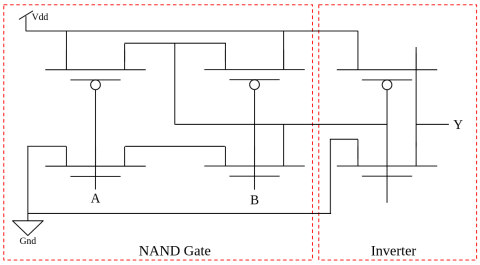

Figure 2: Layout-based schematic of a 2-input AND gate.

This layout is useful when creating layouts as it allows the VLSI designer to create their transistors and components without having to worry about channel placement since some of that step was completed during the schematic-designing process. I drew figure 2 to help determine the layout of my NAND, inverter, and AND gate in Magic.
* *Some recommended rules are putting the power rails on top and the ground rails on the bottom. It is also recommended to orient the transistors horizontally, or in parallel with the VDD and GND rails.*

3 Creating schematics in Xschem

Although I had schematics drawn out, as shown in the previous section. I made my Xschem schematics based on figure 1 rather than the layout-driven schematics. This was because I initially was initially having trouble using Xschem so I wanted to wire them in a way with zero overlap, which in the case of the NAND gate would take up more space layout-wise. I also initially mistook the line shortcut for the wire shortcut so I was putting lines down and was confused on why my circuit was not working. Once that problem was resolved, I was able to create working schematics for an inverter and a CMOS NAND gate.

3.1 Schematics

The schematics for Xschem were based off of the layouts shown in figure 1.

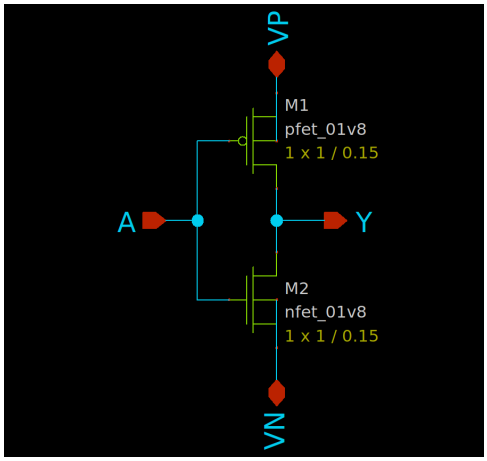

Figure 3: Schematic for a CMOS inverter

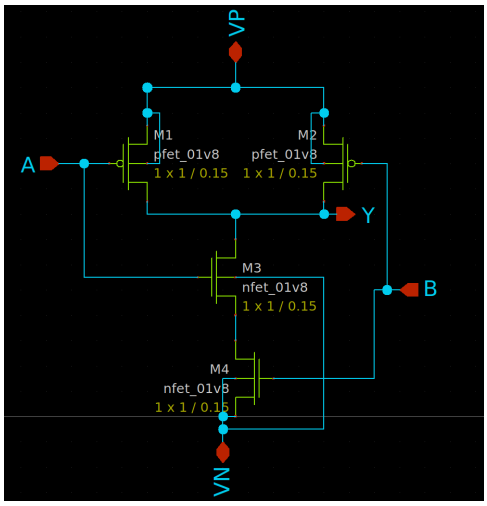

Figure 4: Schematic for a CMOS NAND gate

After I created the NAND and inverter schematics, I created their symbols, which can both be seen in action in figure 5, with the NAND gate on the left and the inverter in series with it on the right. Once the individual symbols were created and linked, I created a testbench file which I used to test the individual components.

The test involved running a square wave through the inputs. I created a voltage source that produced a pulse with a delay of 1ns, rise time and fall time of 1ns, high and low time of 4ns, and a period of 10ns. This pulse was used for both input A for the inverter, and inputs A and B for the NAND gate. Once I confirmed that both individual components worked, I started on the AND gate.

The AND gate, as shown in 5, features a NAND gate and an inverter in series.

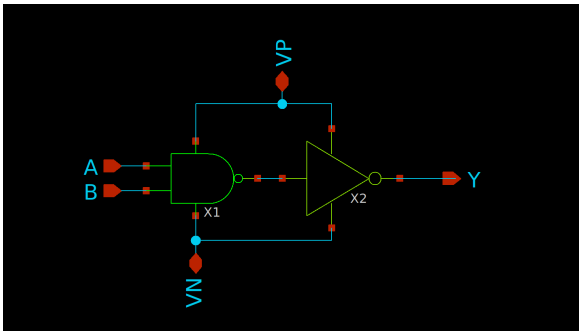

Figure 5: Hierarchical schematic for a CMOS AND gate

Once the schematic was set up with the proper ports, I created the symbol for it, which can be seen in action in figure 6. After the symbol was complete and linked to the schematic, I tested it out on the test harness shown.

3.2 Testing the Circuit

The harness features the AND gate, in symbol form, with a 200fF capacitor pulling down to ground. There are inputs A and B, and Vout is measure in between the capacitor and the output of the AND gate.

I used two different pulses as inputs A and B. Each had a delay of 1ns and a rise and fall time of 1ns. Where they differed was their high and low times and the period. One pulse had a high and low time of 8ns and a period of 18 ns, while the other had a high and low time of 16ns and a period of 36ns. The reasoning behind

this is it would allow all possible logical combinations of A and B to be tested.

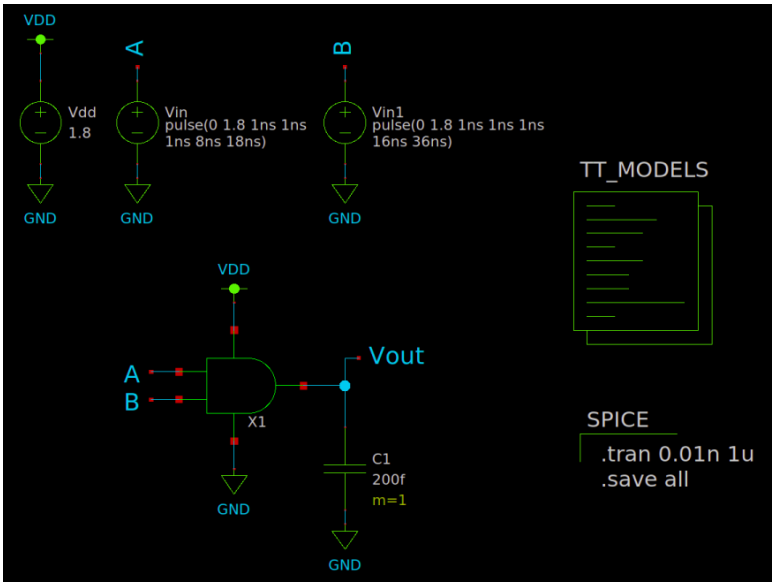

Figure 6: Layout of test bench for AND gate

3.2.1 Simulation Results

As shown in the results for the simulation, the AND gate works properly. The blue and red lines are the different inputs, A and B respectively. The green line is the output voltage measured from the Vout pin.

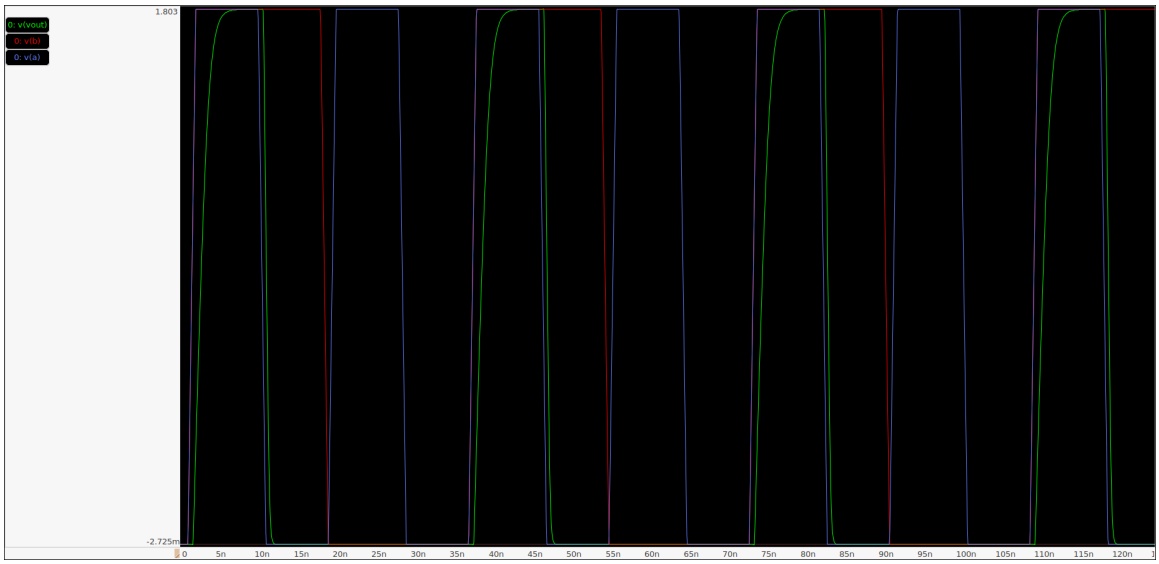

Figure 7: Waveform from simulated test

The capacitor makes the rise time of the circuit a negative exponential. When Both A and B are high, the output voltage rises sharply then levels at around 1.8 V. This only occurs when A and B are high- in any other case, the output voltage is low. This verifies that out AND gate is functioning properly.

4 Creating Layouts in Magic VLSI

Magic VLSI is a software that allows us to create VLSI layouts that can be fabricated. When designing transistors and components, we need to follow certain design rules and specifications. Luckily, Magic has a built-in feature that automatically checks if builds are within DRC constraints for Magic's VLSI design.

4.1 Inverter

First, I built the inverter. The in-class tutorial provided isntructions on how to build this component. As shown in figure 8, there is a pMOS transistor on top that is connected to an nMOS on the bottom. There is only one logical input, A, which is inputted through a local interconnect that then is connected to a poly fill that runs through both transistors. The pMOS transistors also takes in a positive voltage input (VP) while the nMOS transistor drains to ground (VN). The output of the inverter is a local interconnect that connects the pMOS to the cmos.

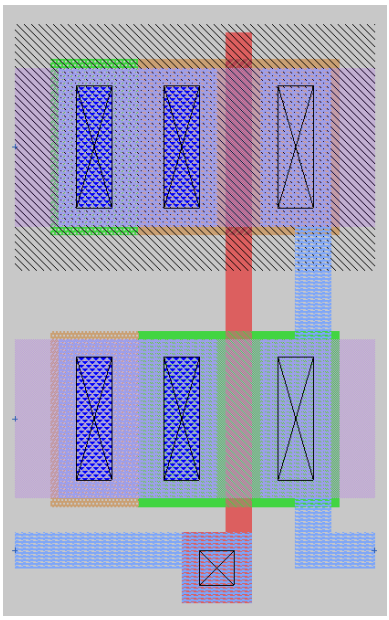

Figure 8: Top level image of layout for an inverter

4.2 NAND

The NAND gate has 2 logical inputs, A and B, which are structure the same as the inverter layout shown in figure 8. It also features two pMOS transistors that share a drain and a source but a different gate, making them parallel. The pMOS output flows into an nMOS transistor that is in series with another nMOS transistor.

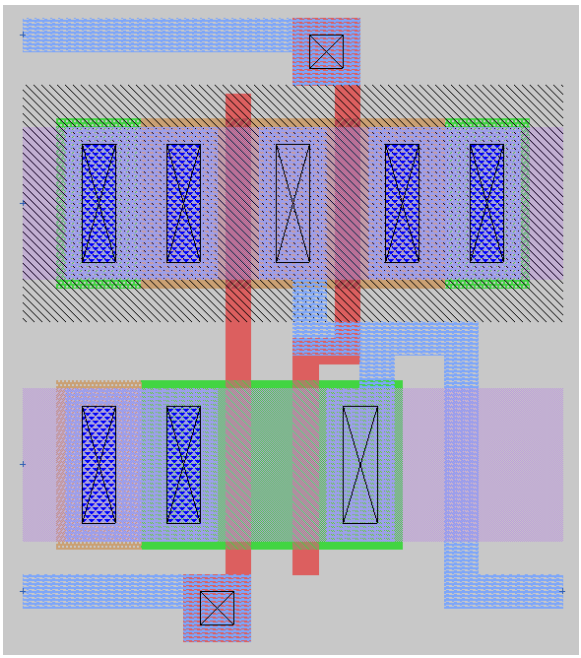

Figure 9: Top level image of layout for a NAND gate

4.3 AND

The AND gate makes use of the NAND and inverter cell made previously. The NAND and inverter are put in series, with the NAND's output leading into the input of the inverter. In addition to putting the two cells in series, we had to add two additional m1 plates at the VP and VN inputs of the pMOS and nMOS transistors, and local interconnects at inputs A and B, and output Y.

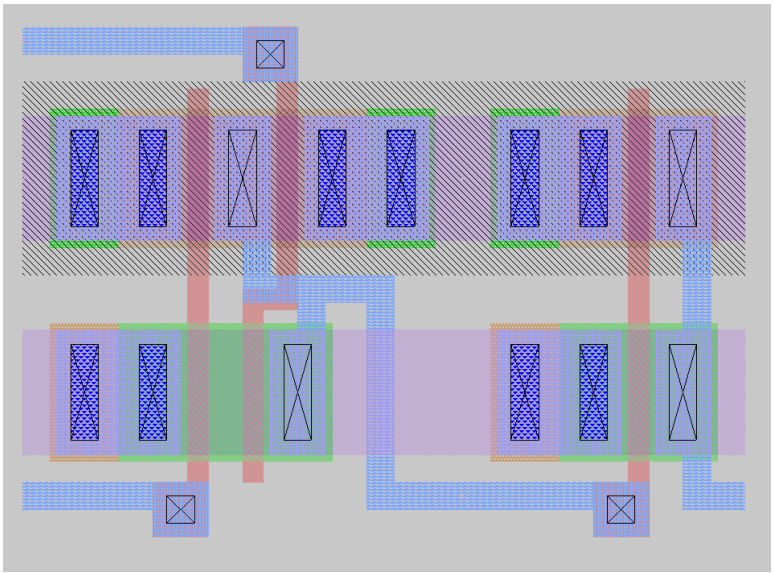

Figure 10: Top level image of layout for an AND gate

5 Netlist

After the main schematic was made in Xschem and layout was made in Magic VLSI, we could generate a netlist and compare between the two using Netgen's LVS function. Using "lvs", I compared the two netlists. Two successfully matched netlists will result in the terminal displaying the lines below each cell comparison:

Circuits match uniquely.
Netlists match uniquely.

My results can be viewed in my respository linked in the Introduction section in the file ResultsMatch.txt located in the SPICE folder.

6 Reflection

Throughout this assignment, I learned how to use Xschem and Magic. Because of a major mistake made due to forgetting a fundamental concept, I took a bit longer at completing this assignment as I overlooked my Xscheme and assumed that my Layout was incorrect rather than my schematic. The nMOS that took in the output of the parallel pMOS was wired such that its substrate was connected to the output of the transistor rather than to ground. Although the schematic successfully performed the simulation, its netlist featured the error that caused it to not match the layout's netlist.