EE457: Digital IC Design

Project #2 FALL SEMESTER 2023

Report Cover Sheet*

Due 10/28/2023 by 8PM (-2 pts per hour up to 5 points/day)

PROJECT TITLE: CMOS 8-to-1 MUX

Student's Name: Dylan Kirdahy

Topics (Do not change orders of section)	GRADES
Section1: Executive Summary (1/2 page)	Required
Section 2: Introduction and Background	/5
Section 3: Electric Circuit Schematics (TG and Conventional CMOS designs)	/10
Section 4: LTSPICE Simulation for Schematic (TG and Conventional CMOS designs)	/10
Section 5: IRSIM for Schematic (TG and Conventional CMOS designs)	/10
Section 6: Electric Layouts (landscape mode and must legible to see gates) (TG and Conventional CMOS designs)	/25
Section 7: LTSPICE Simulation for Layouts (TG and Conventional CMOS designs)	/10
Section 8: IRSIM for Layout (TG and Conventional CMOS designs)	/10
Section 9: Compare LTSPICE Measurements for Schematic and Layout (must provide comparisons between the two in table format) Provide delays, rise and fall times.	/10
Section 10: Measurements of chip area, number of transistors for the layout (provide a table to compare) and write summary for TG and CMOS designs.	/10
Section 11: Conclusions and References	Required
Late Penalty	
TOTAL	**/100**

*Penalty rules: A) -5pts per day for late submission. After five days, a score of zero will be given, but you are still required to complete the report. B) -2pts for any violations and delays submitted after 8PM.

Contents

1 Executive Summary

The goal of this project was to build an 8-to-1 multiplexer twice, once using transmission gates and once using ordinary CMOS logic. A multiplexer is a combinational logic circuit that allows for one of many inputs to be passed through to a single output. In this case there are 8 inputs, D0 to D7, and the state of output Y reflects the state of the input selected with the select lines S0, S1, and S2. Multiplexers have a variety of uses in digital electronics, most notably in the design of CPUs.

In this project, the multiplexer is realized using the software Electric, which allows us to design a schematic, layout, and icon for the multiplexer. We are also able to export the design to LTSpice in order to run in-depth simulations as well as simulate the designs with the built-in IRSIM.

I designed the transmission gate multiplexer and the CMOS multiplexer with two different philosophies. For the transmission gate design, I used the method that we learned about in class, where each input has its own set of tranmission gates that connect to a single node for the output. For the CMOS design, I chose to create a 2-to-1 multiplexer and then chain them together to build the 8-to-1 multiplexer. Both designs were an interesting challenge.

The simulation component of the project involved generating input signals and then confirming that the designs generates the expected output. In addition, various measurements can be taken within LTSpice with the `.meas` command. These measurements include propagation delay, rise and fall times, and average current draw.

2 Introduction and Background

The multiplexer is a combinational logic circuit that is crucial to many digital designs. Multiplexers allow us to choose which input out of a number of inputs reaches the output using select lines. They can be designed to take any number of inputs, but typically a power of two. The simplest multiplexer design is a 2-to-1 multiplexer: there are two inputs, D0 and D1, and a single output, Y, and the select line, S, determines which of the inputs reach the output. For instance, when a logic 0 is applied to S, whatever digital signal is applied to D0 is output on Y, and when a logic 1 is applied to S, the signal applied to D1 is output on Y. When one input is selected, all other inputs are ignored.

This logic extends to multiplexers with more inputs. A 4-bit multiplexer has four inputs and one output, and two select lines (S1 and S0) are used to select which of the four inputs is output. 00 selects D0, 01 selects D1, 10 selects D2, and 11 selects D3. Multiplexers can be designed with 8 inputs, 16 inputs, 32 inputs, etc., however in the scope of this project we are building an 8-bit multiplexer which has eight inputs and uses three select lines (S2, S1, S0) to determine which input reaches the output. An interesting side note is that multiplexers can pass an analog signal or a digital signal based on how they are designed; the CMOS design can only pass a digital signal, but the transmission gate design can pass an analog signal as well due to the transmission gates acting as closed or open switches. However, passing analog signals is outside of the scope of this project.

In order to create the design for our 8-to-1 multiplexer, we first need to create a function and a truth table to get a better understanding of the logic that powers the multiplexer. Since our CMOS design is based on a 2-to-1 multiplexer, we will start there. The function for the 2-to-1 multiplexer is as follows:

$$Y = \overline{S} \cdot D_0 + S \cdot D_1$$

The working principle is that when $S = 0$ the output is $Y = D_0$ and when $S = 1$ the output is $Y = D_1$. The truth table for this function is provided below in Table 1.

S	**Y**
0	D_0
1	D_1

Table 1: Truth Table for the 2-to-1 multiplexer.

This working principle can be applied to the 8-to-1 multiplexer as well. The function for the 8-to-1 multiplexer is as follows:

$$Y = \bar{S_2}\cdot\bar{S_1}\cdot\bar{S_0}\cdot D_0 + \bar{S_2}\cdot\bar{S_1}\cdot S_0\cdot D_1 + \bar{S_2}\cdot S_1\cdot\bar{S_0}\cdot D_2 + \bar{S_2}\cdot S_1\cdot S_0\cdot D_3 + S_2\cdot\bar{S_1}\cdot\bar{S_0}\cdot D_4 + S_2\cdot\bar{S_1}\cdot S_0\cdot D_5 + S_2\cdot S_1\cdot\bar{S_0}\cdot D_6 + S_2\cdot S_1\cdot S_0\cdot D_7$$

The truth table for the 8-to-1 multiplexer is shown below in Table 2.

$\mathbf{S_2}$	$\mathbf{S_1}$	$\mathbf{S_0}$	**Y**
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	0	0	D_4
1	0	1	D_5
1	1	0	D_6
1	1	1	D_7

Table 2: Truth Table for the 8-to-1 multiplexer.

The symbol for the 8-to-1 multiplexer, designed in Electric to represent the schematic, is shown in Figure 1.

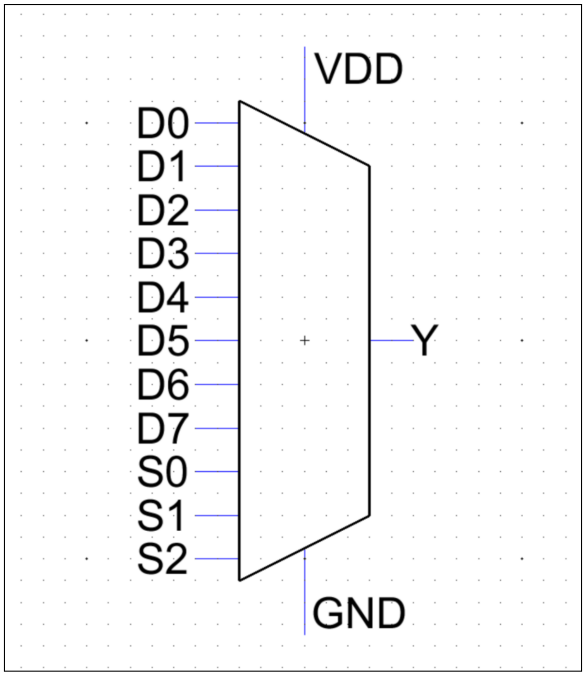

Figure 1: The symbol for the 8-to-1 multiplexer designed in Electric.

As mentioned before, the transmission gate design was made with 8 separate sets of three transmission gates (for S2, S1, and S0). However, the CMOS design was built by chaining together seven 2-to-1 multiplexers. The configuration of these 2-to-1 multiplexers in order to make the 8-to-1 multiplexer is shown in Figure 2.

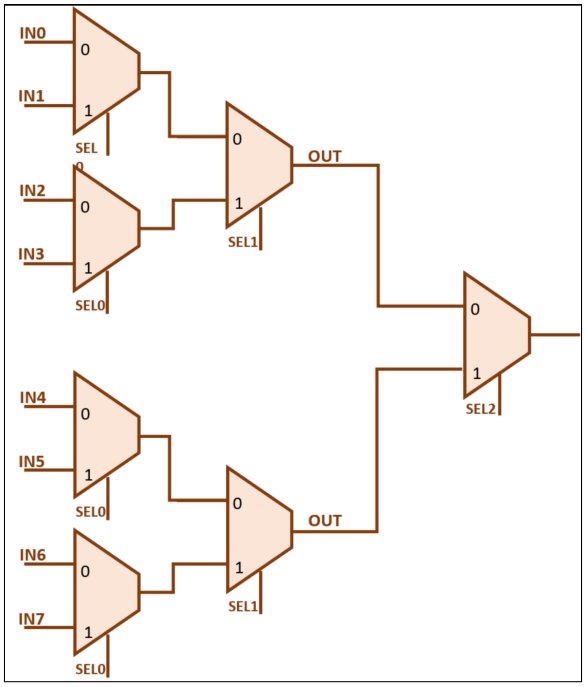

Figure 2: The configuration of seven 2-to-1 multiplexers to create a single 8-to-1 multiplexer.

In order to implement this, we first need to create the 2-to-1 multiplexer. I started with the function $Y = \overline{S} \cdot D_0 + S \cdot D_1$ and implemented it with CMOS logic. Since CMOS logic outputs an inverted signal, we can do one of two things: either invert the function and then implement that, or add an inverter at the end. I chose to add an inverter at the end, since inverting the whole function would involve inverting both D0 and D1 in addition to S. Adding an inverter at the end means we only need two inverters instead of three. The sketch I drew of the schematic is shown in Figure 3.

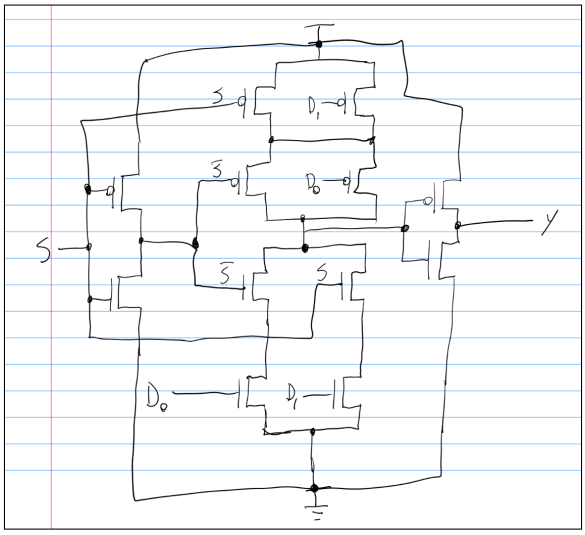

Figure 3: A sketch of the schematic for a single 2-to-1 multiplexer.

The next step was to determine an Euler's Path from the sketch. The Euler's Paths that I determined are shown in Figure 4.

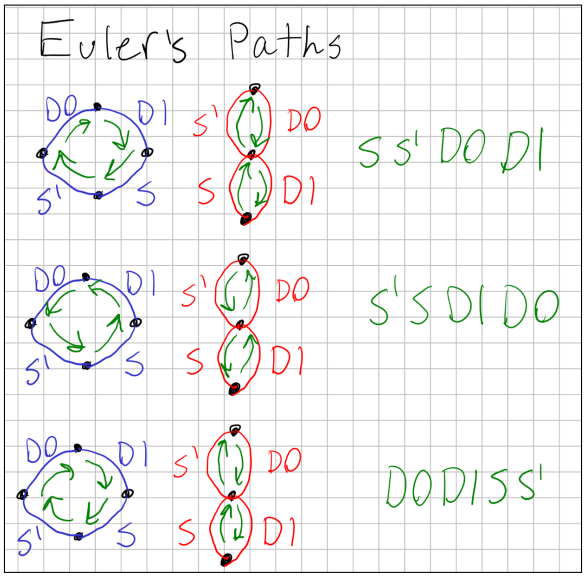

Figure 4: A few Euler's Paths for the 2-to-1 multiplexer.

I decided to use the path $\overline{S}, S, D_1, D_0$ since it seemed to be the most space efficient to me. The stick diagram I sketched based on the Euler's Path chosen is shown in Figure 5.

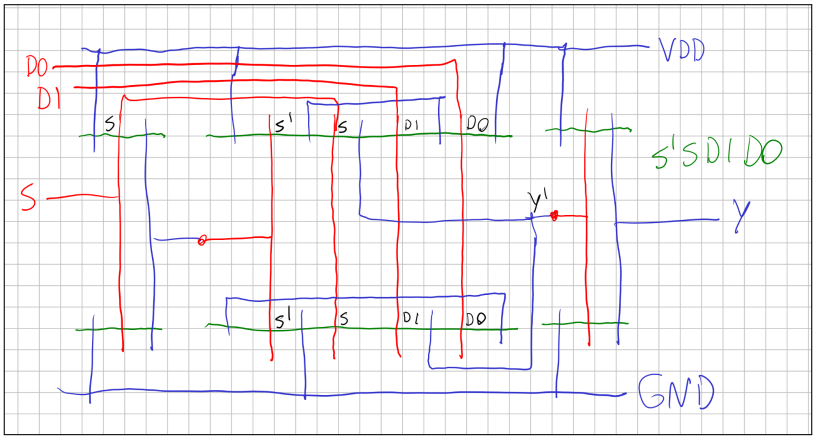

Figure 5: The stick diagram for the 2-to-1 multiplexer.

We now have all of the information we need to begin the design of both the transmission gate and CMOS multiplexers.

3 Electric Circuit Schematics

This section details the design of the schematics for the transmission gate and CMOS designs of the multiplexer.

3.1 Transmission Gate Schematic

The schematic for the transmission gate design of the 8-to-1 multiplexer is shown in Figure 6. The design works by giving each input (D0 through D7) its own set of three transmission gates. Each transmission gate corresponds to one of the select pins, S2, S1, or S0. As shown in table 2, each input corresponds to a specific combination of inputs to the select line. When that combination of the inputs is applied to the select line, the corresponding D input (and only that input) is connected to the output. Since there are eight inputs, there are eight sets of three transmission gates. A closer look at the sets of transmission gates can be seen in Figure 7. Since each transmission gate requires an inverted signal as well as a non-inverted signal to operate, each select pin has its own inverter, which can be seen more clearly in Figure 8.

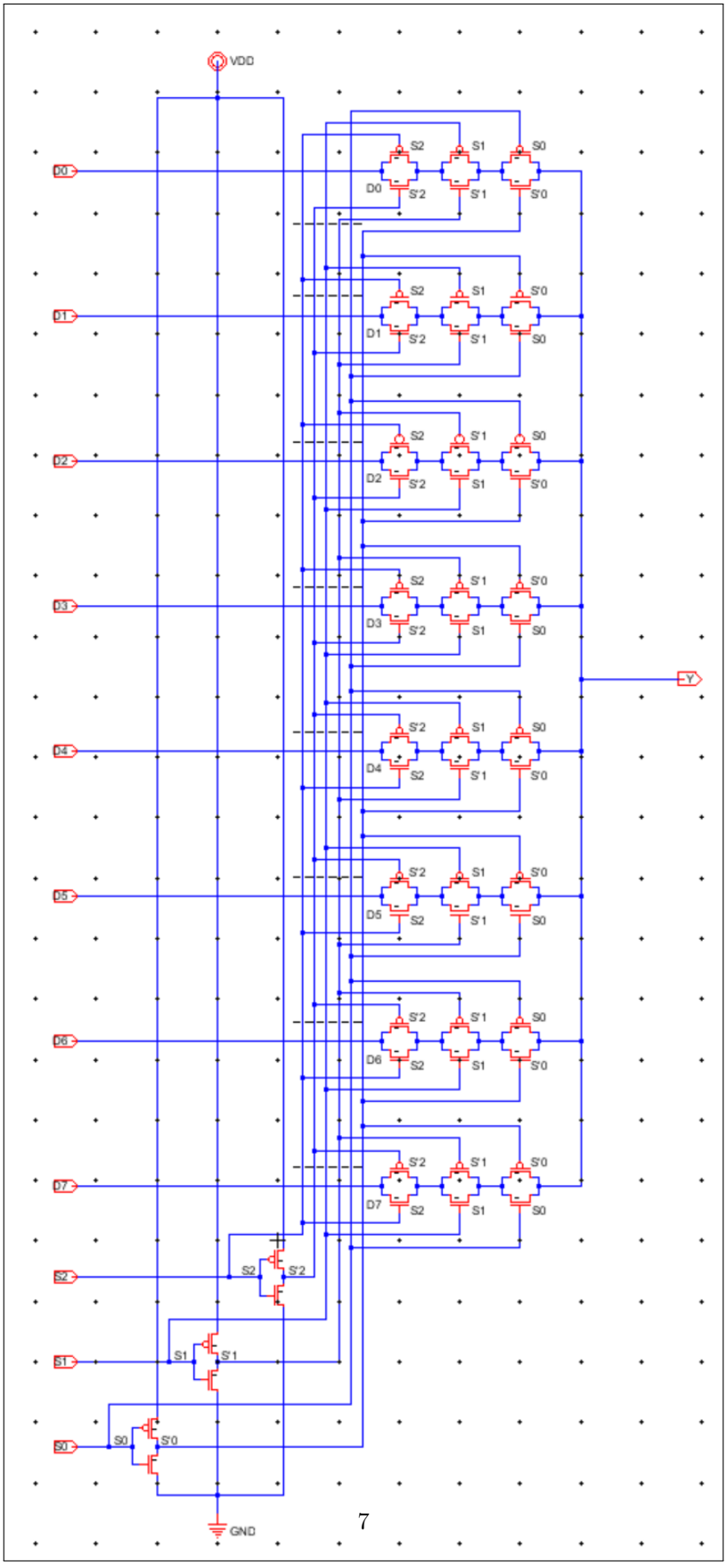

Figure 6: The schematic in Electric for the transmission gate design.

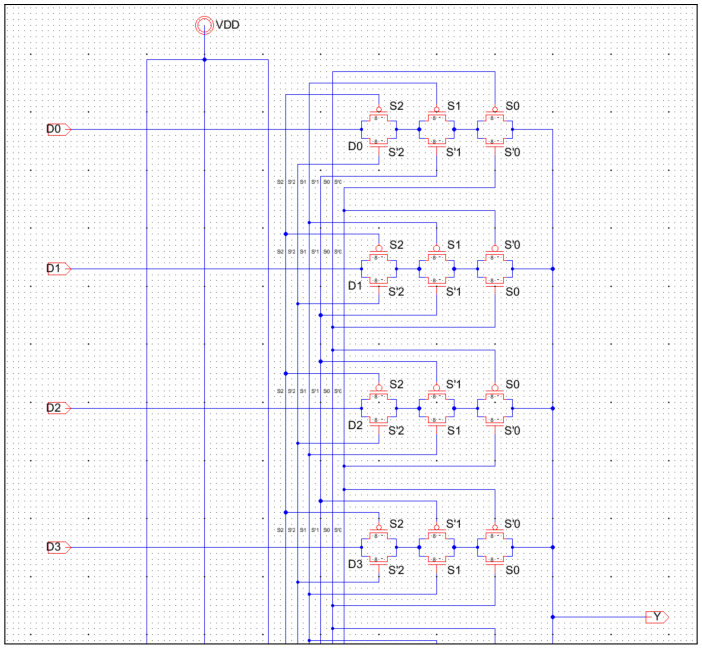

Figure 7: A close up of the schematic showing the sets of transmission gates.

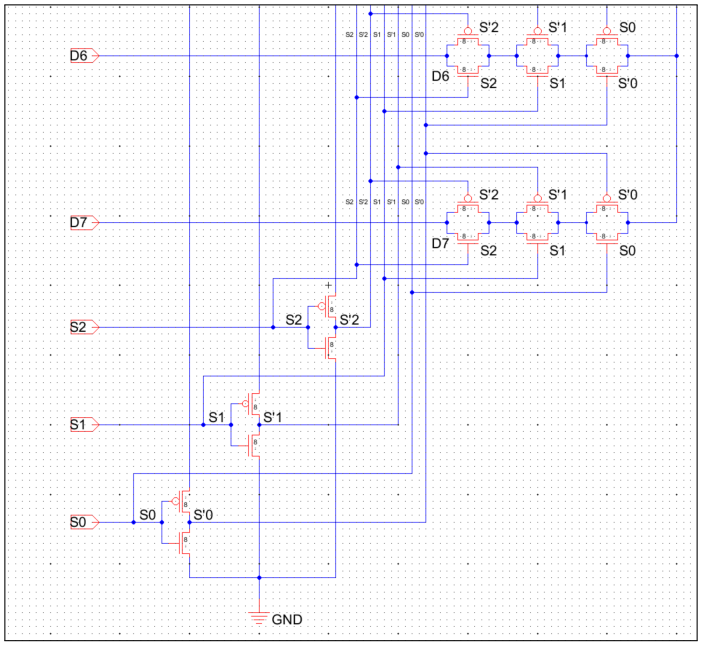

Figure 8: A close up of the schematic showing the inverters which invert the select pins.

Figure 9 shows that the schematic passes the DRC check.

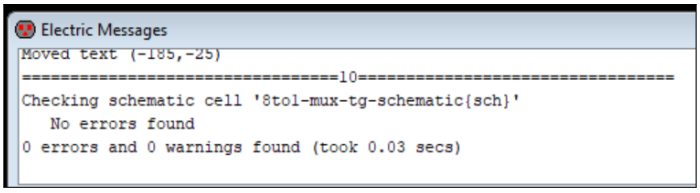

Figure 9: The DRC check for the transmission gate schematic.

3.2 CMOS Schematic

The schematic for the CMOS design of the 8-to-1 multiplexer is simply the 2-to-1 multiplexer schematic sketched in Figure 3 repeated seven times in the configuration shown in Figure 2. However, there is one change. I realized that having an inverter on the output of each of the 2-to-1 multiplexer was redundant. Only the last stage needs an inverter. This is because while the first stage inverts the output (due to CMOS logic outputting an inverted signal), the second stage will invert it again, cancelling out the first inversion. This saves 12 transistors in the final design. The full schematic is shown in Figure 10. A close up of the first two stages of the multiplexer is shown in Figure 11. A close up of the last stage is shown in Figure 12.

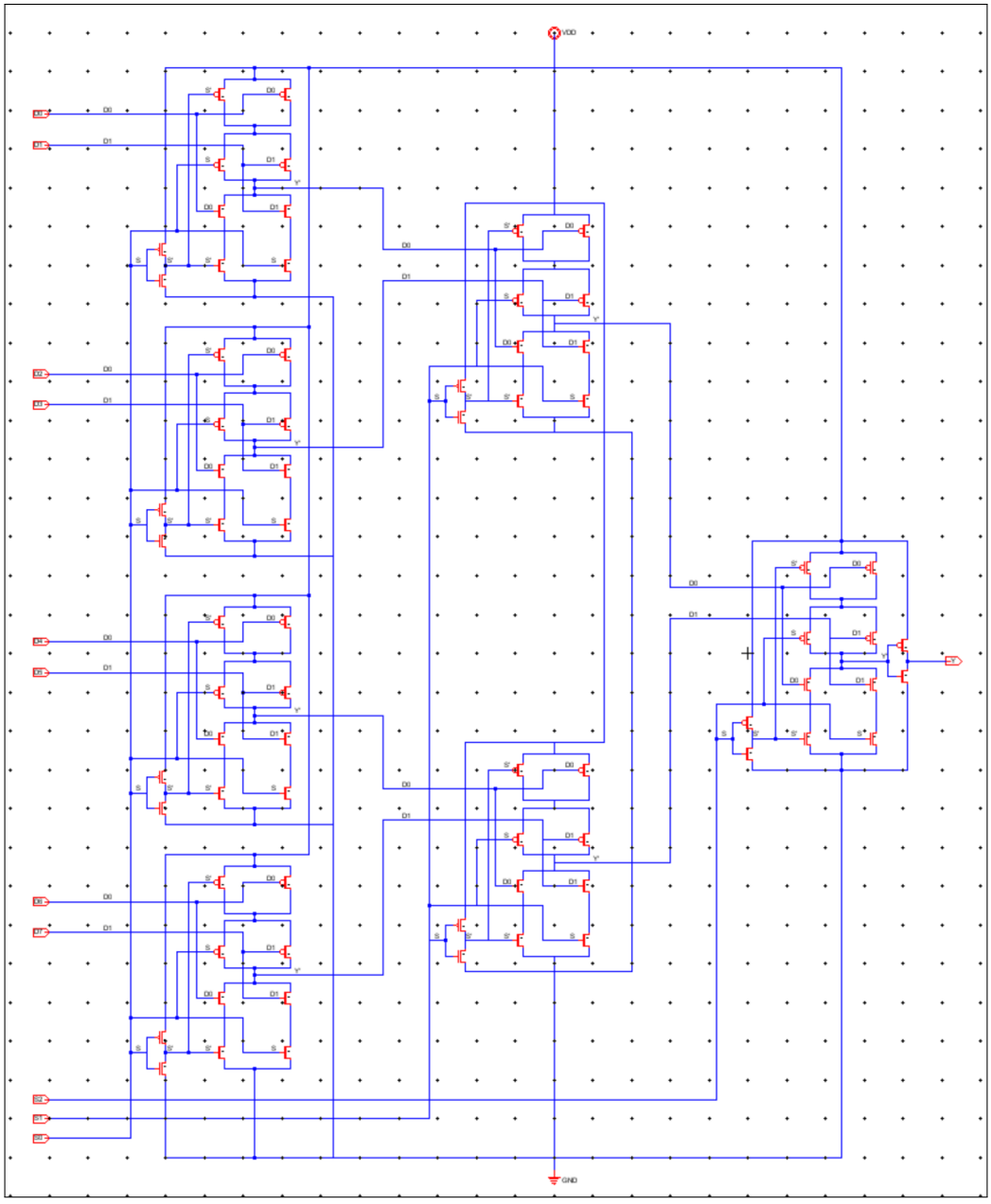

Figure 10: The schematic in Electric for the CMOS design.

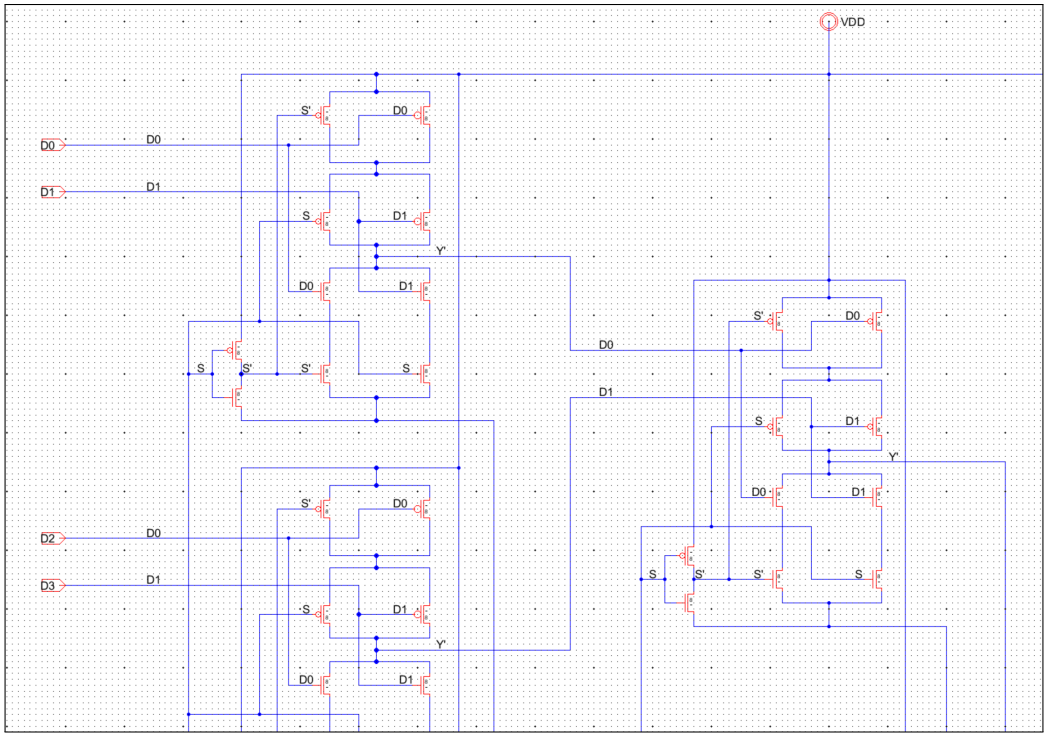

Figure 11: A close up of the schematic showing the first two stages.

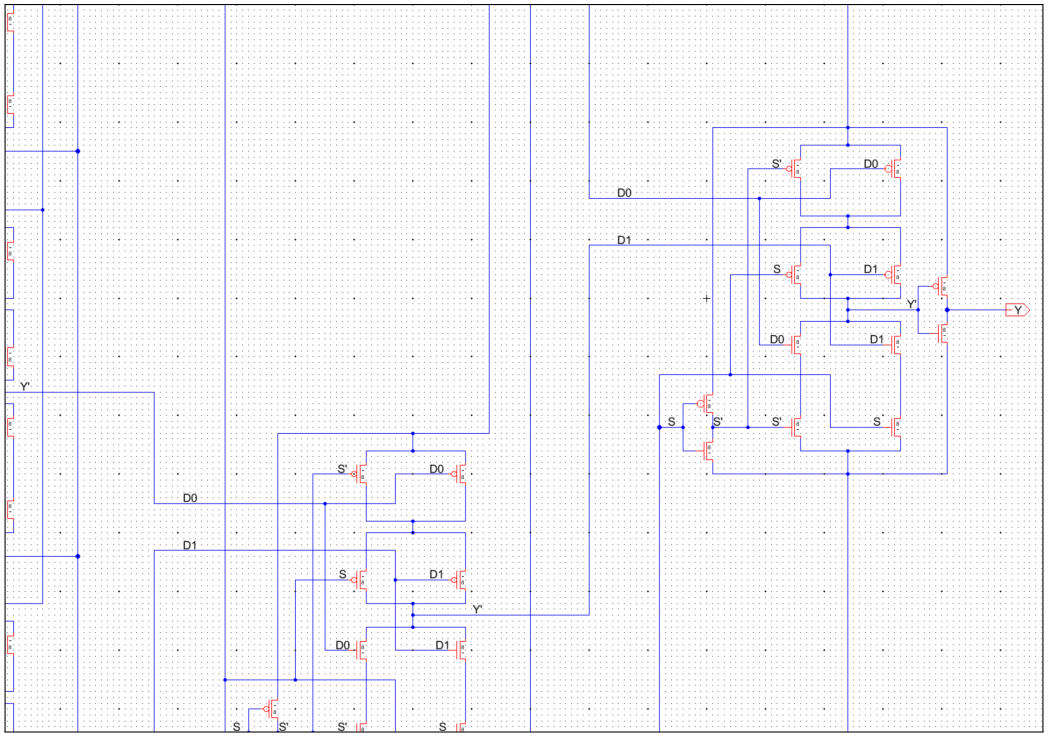

Figure 12: A close up of the schematic showing the last stage.

Figure 13 shows that the schematic passes the DRC check.

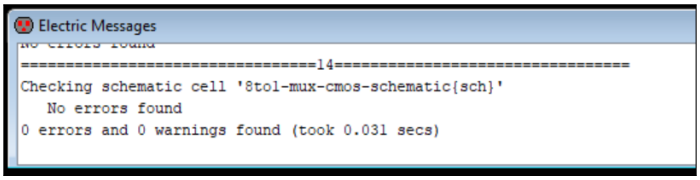

Figure 13: The DRC check for the CMOS schematic.

4 LTSpice Simulation for Schematic

This section will detail the LTSpice simulations for both the transmission gate schematic and the CMOS schematic. The spice code used to generate the graphs for this simulation are shown in Figure 14. LTSpice was also used to take measurements that will be detailed later in the report. The code used to generate those measurements is shown in Figure 15.

```
VDD VDD 0 DC 3.3
VGND GND 0 DC 0
*
*For logic simulation:
*Select lines:
VS2 S2 0 PULSE(3.3 0 0 1n 1n 1000n 2000n)
VS1 S1 0 PULSE(3.3 0 0 1n 1n 500n 1000n)
VS0 S0 0 PULSE(3.3 0 0 1n 1n 250n 500n)
*
*Inputs:
VD0 D0 0 PULSE(3.3 0 0 1n 1n 100n 200n)
VD1 D1 0 PULSE(3.3 0 0 1n 1n 50n 100n)
VD2 D2 0 PULSE(3.3 0 0 1n 1n 25n 50n)
VD3 D3 0 PULSE(3.3 0 0 1n 1n 12.5n 25n)
VD4 D4 0 PULSE(3.3 0 0 1n 1n 150n 200n)
VD5 D5 0 PULSE(3.3 0 0 1n 1n 75n 100n)
VD6 D6 0 PULSE(3.3 0 0 1n 1n 37.5n 50n)
VD7 D7 0 PULSE(3.3 0 0 1n 1n 18.75n 25n)
*
.TRAN 0 2000n
.include Z:\MOS_model.txt
```

Figure 14: The LTSpice code used to generate the graphs in the following section.

```
VDD VDD 0 DC 3.3
VGND GND 0 DC 0
*
*For measurements:
*Select lines:
VS2 S2 0 DC 0
VS1 S1 0 DC 0
VS0 S0 0 DC 0
*
*Inputs:
VD0 D0 0 PULSE(3.3 0 0 1n 1n 500n 1000n)
VD1 D1 0 DC 0
VD2 D2 0 DC 0
VD3 D3 0 DC 0
VD4 D4 0 DC 0
VD5 D5 0 DC 0
VD6 D6 0 DC 0
VD7 D7 0 DC 0
*
.TRAN 0 2000n
.meas risetime TRIG v(Y)=0.33 TD=0 rise=1 TARG v(Y)=2.97 TD=0 rise=1
.meas falltime TRIG v(Y)=2.97 TD=0 fall=1 TARG v(Y)=0.33 TD=0 fall=1
.meas tpHL TRIG v(D0)=1.65 TD=0 fall=1 TARG v(Y)=1.65 TD=0 fall=1
.meas tpLH TRIG v(D0)=1.65 TD=0 rise=1 TARG v(Y)=1.65 TD=0 rise=1
.meas propagationdelay param=(tpHL+tpLH)/2
.meas chip_current I_vdd AVG I(VDD)
.include Z:\MOS_model.txt
```

Figure 15: The LTSpice code used to generate the measurements which will be detailed in a later section.

4.1 Transmission Gate Schematic

The spice simulation for the transmission gate schematic is shown in Figure 16. The inputs D0 through D7 are shown first, each with their own uniquely identifying waveform. The select line is stepped through all eight values from 000 to 111, each corresponding to the inputs D0 through D7 as shown in the figure. At the output Y, the correct waveform can be seen corresponding to the selected input. The waveforms have been highlighted with a colored dotted rectangle to show the location of the input and the output at Y. This figure illustrates the expected behavior.

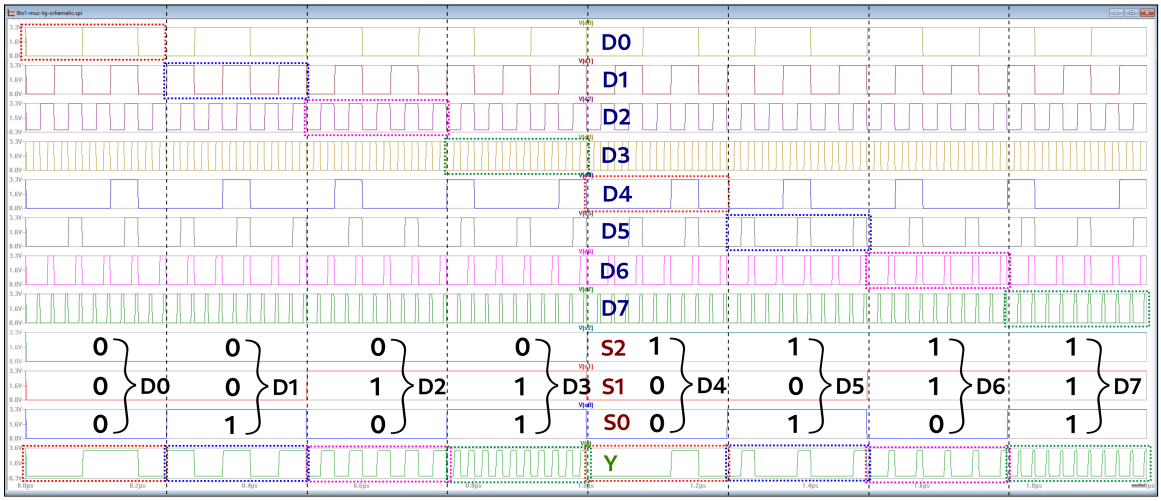

Figure 16: The LTSpice simulation of the transmission gate schematic showing that each combination of inputs to the select line passes through the corresponding inputs D0 through D7.

The measurements obtained with the spice code shown in Figure 15 for the transmission gate schematic are shown in Figure 17.

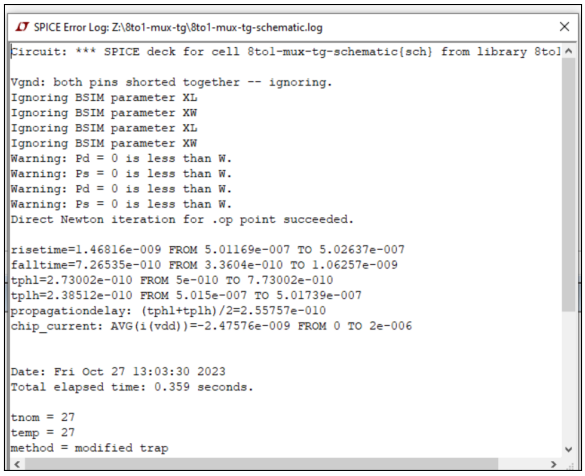

Figure 17: The measurements obtained through LTSpice simulation of the transmission gate schematic.

4.2 CMOS Schematic Spice

The spice simulation for the CMOS schematic is shown in Figure 18. As described in the last subsection, the behavior in the graph is as expected. Each input is passed through to the output when selected by the select line.

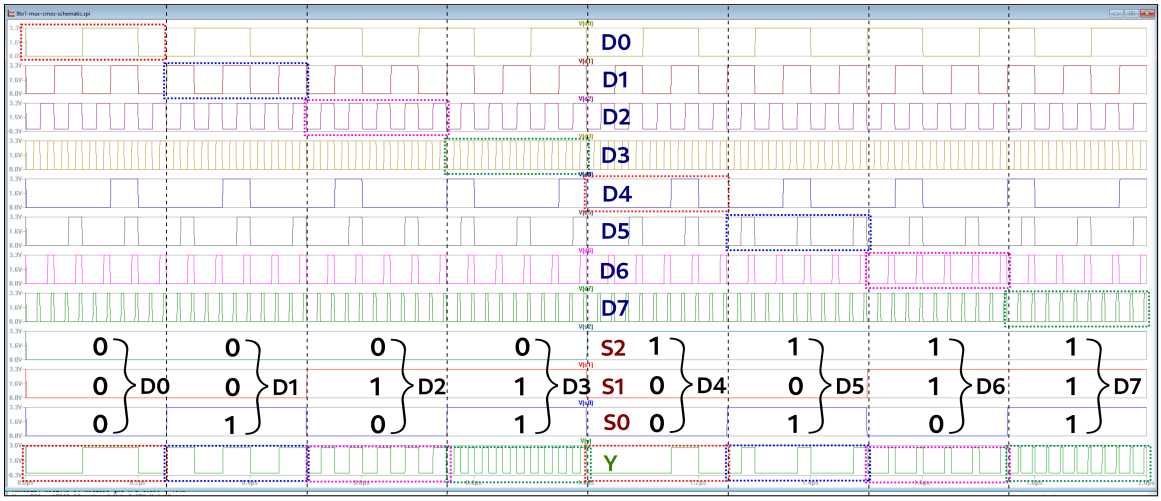

Figure 18: The LTSpice simulation of the CMOS schematic showing that each combination of inputs to the select line passes through the corresponding inputs D0 through D7.

The measurements obtained with the spice code shown in Figure 15 for the CMOS schematic are shown in Figure 19.

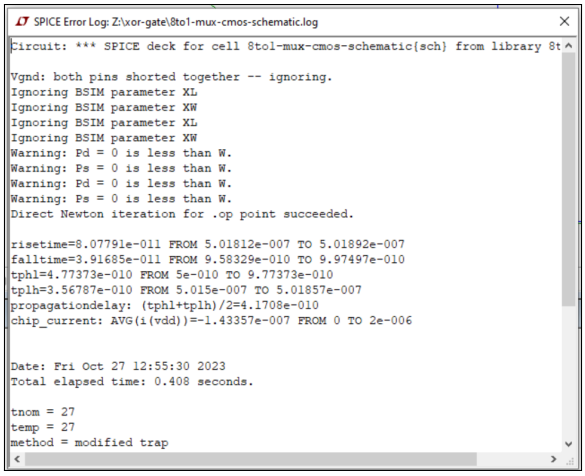

Figure 19: The measurements obtained through LTSpice simulation of the CMOS schematic.

5 IRSIM for Schematic

This section details the IRSIM simulations done on both the transmission gate schematic and the CMOS schematic. I chose to simulate only one D input per screenshot and two per design. This is because IRSIM can be buggy and has a hard time with too many inputs changing at once.

5.1 Transmission Gate Schematic IRSIM

The IRSIM simulation for the transmission gate schematic is shown in Figure 20 and 21. As shown in the screenshots, the schematic behaves as expected.

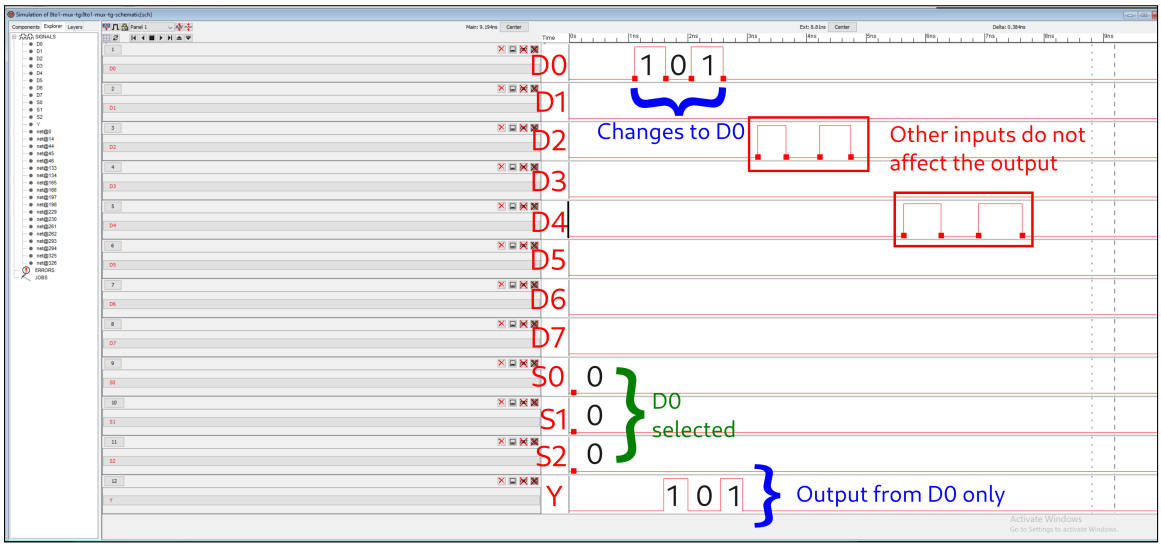

Figure 20: One IRSIM simulation for the transmission gate schematic showing that the design works as expected.

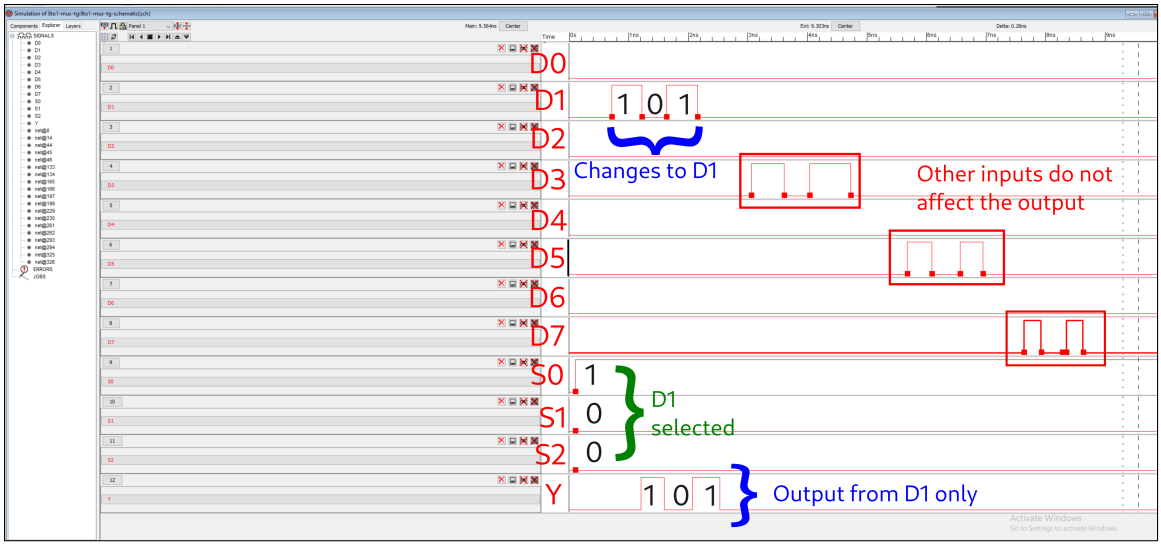

Figure 21: Another IRSIM simulation for the transmission gate schematic showing that the design works as expected.

5.2 CMOS Schematic IRSIM

The IRSIM simulation for the CMOS schematic is shown in Figure 22 and 23. As shown in the screenshots, the schematic behaves as expected.

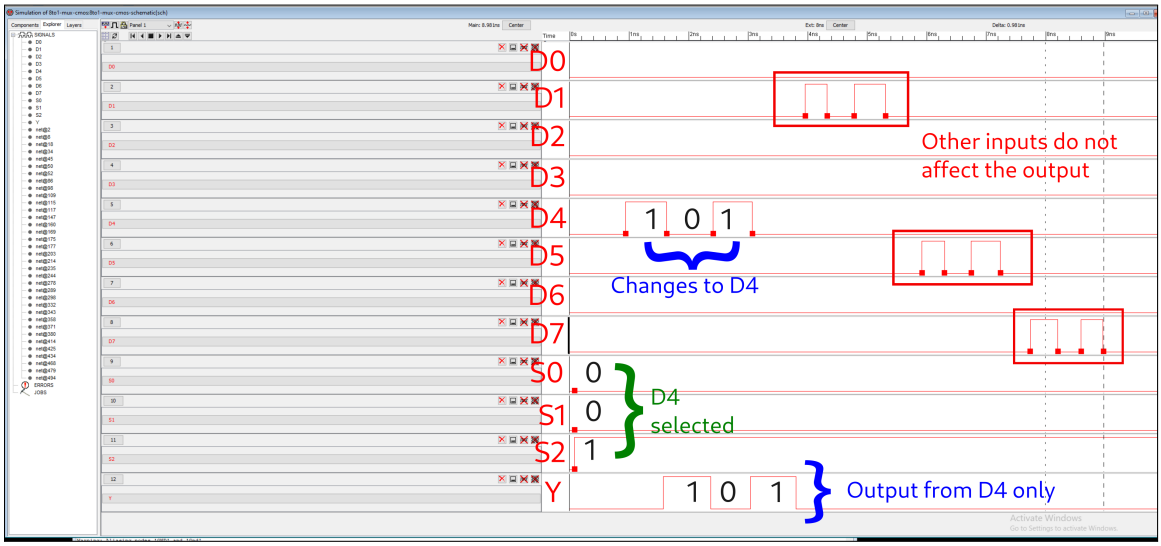

Figure 22: One IRSIM simulation for the CMOS schematic showing that the design works as expected.

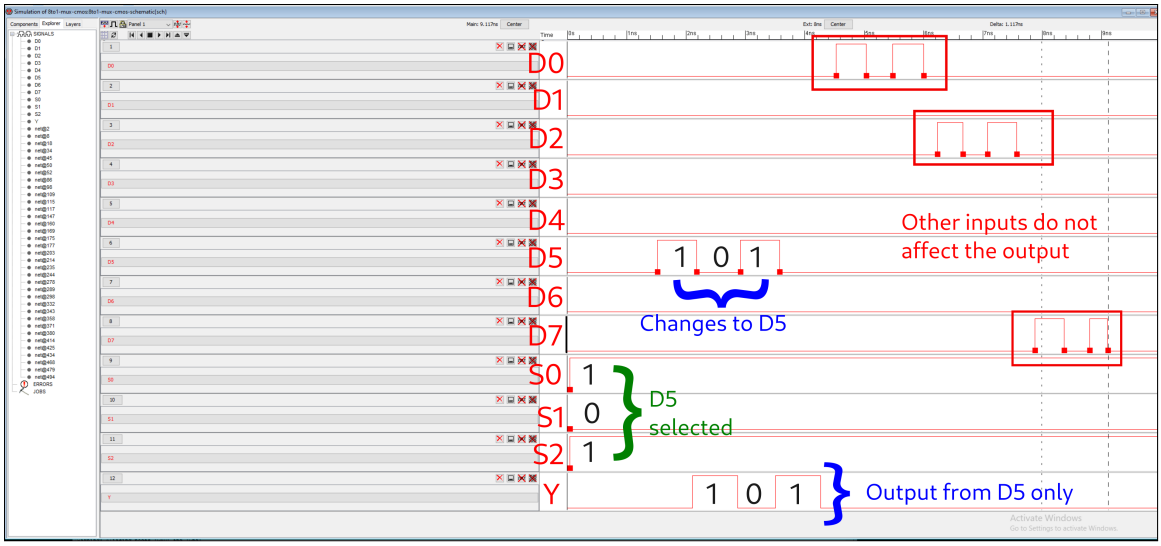

Figure 23: Another IRSIM simulation for the CMOS schematic showing that the design works as expected.

6 Electric Layouts

This section details the design of the layouts in Electric for both the transmission gate multiplexer and the CMOS multiplexer.

6.1 Transmission Gate Layout

The transmission gate layout of the 8-to-1 multiplexer is shown in Figure 24. I first designed one set of three transmission gates and then stacked them, running the inputs to the sets along the left and right. I had to pay careful attention to the layout of each set of gates to make sure that I could run the proper signals to each. A close up of the individual sets of transmission gates is shown in Figure 25. I placed the inverters for the select line at the bottom as shown in Figure 26. On the left of the transmission gates are metal and polysilicon lines for S_2, $\overline{S_2}$, and S_1, and to the right are lines for $\overline{S_1}$, $\overline{S_0}$, and S_0. I also had to route VDD to the right as well.

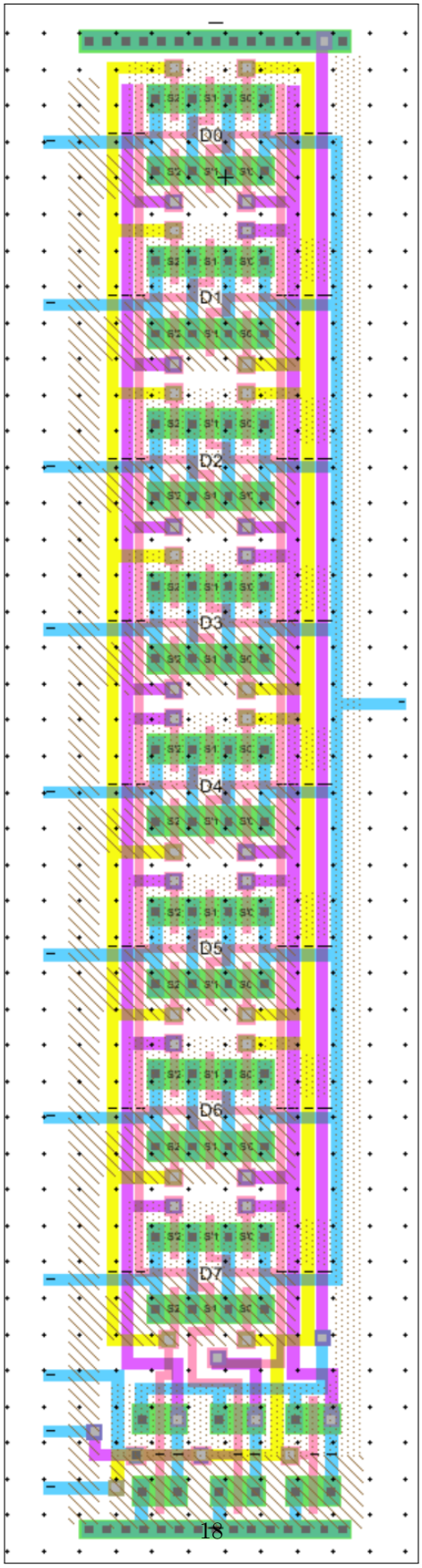

Figure 24: The layout for the transmission gate design of the 8-to-1 multiplexer.

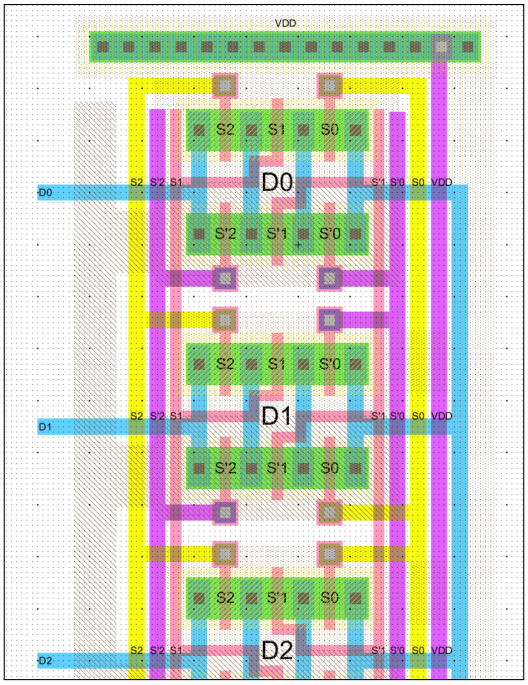

Figure 25: A close up of the sets of transmission gates for the layout.

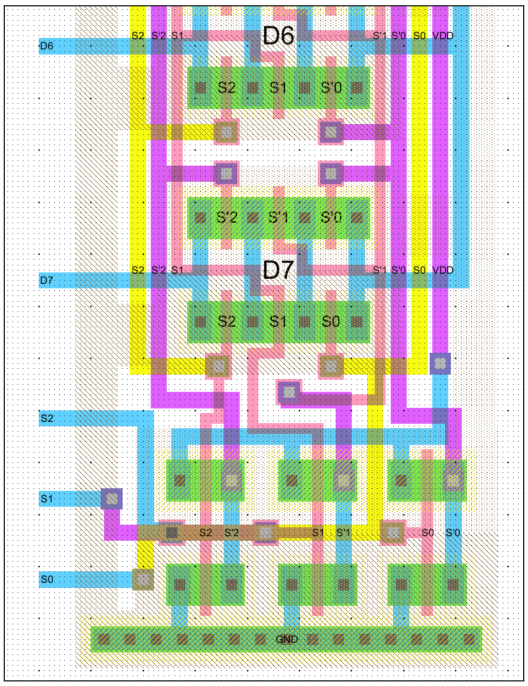

Figure 26: A close up of the inverters at the bottom of the transmission gate layout.

The DRC results are shown in Figure 27 and the well check results are shown in Figure 28, both showing no errors.

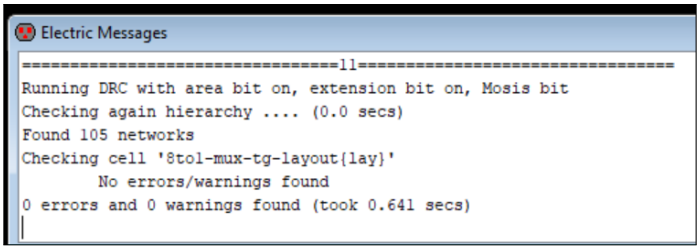

Figure 27: The DRC results for the transmission gate layout showing no errors.

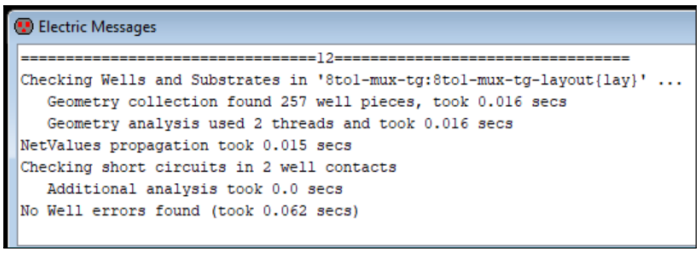

Figure 28: The well check results for the transmission gate layout showing no errors.

6.2 CMOS Layout

The CMOS layout of the 8-to-1 multiplexer is shown in Figure 29. I first designed one 2-to-1 inverter based off of the stick diagram shown in Figure 5. I then planned out how I would arrange the seven 2-to-1 multiplexers. I decided the most space-efficient approach would be to make the cell two multiplexers high. The first four multiplexers take all 8 of the inputs. I used metal 2 and metal 3 to bring the inputs over the multiplexers to their inputs. I then connected the output of those four multiplexers to another two multiplexers, and then the output of those two to another singular multiplexer. I then managed to fit the inverter above the final multiplexer. I ran the select lines at the bottom of the design. Figure 30 shows a close up of the first stage, consisting of four 2-to-1 multiplexers. Figure 31 shows a close up of the last two stages and the inverter.

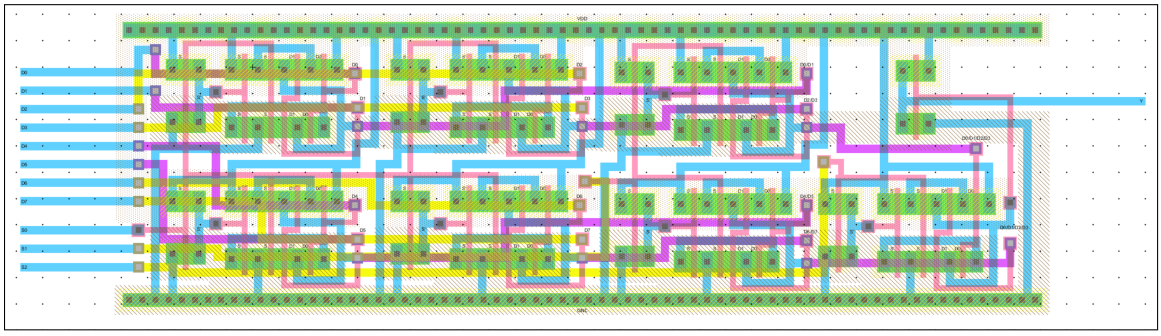

Figure 29: The layout for the CMOS design of the 8-to-1 multiplexer.

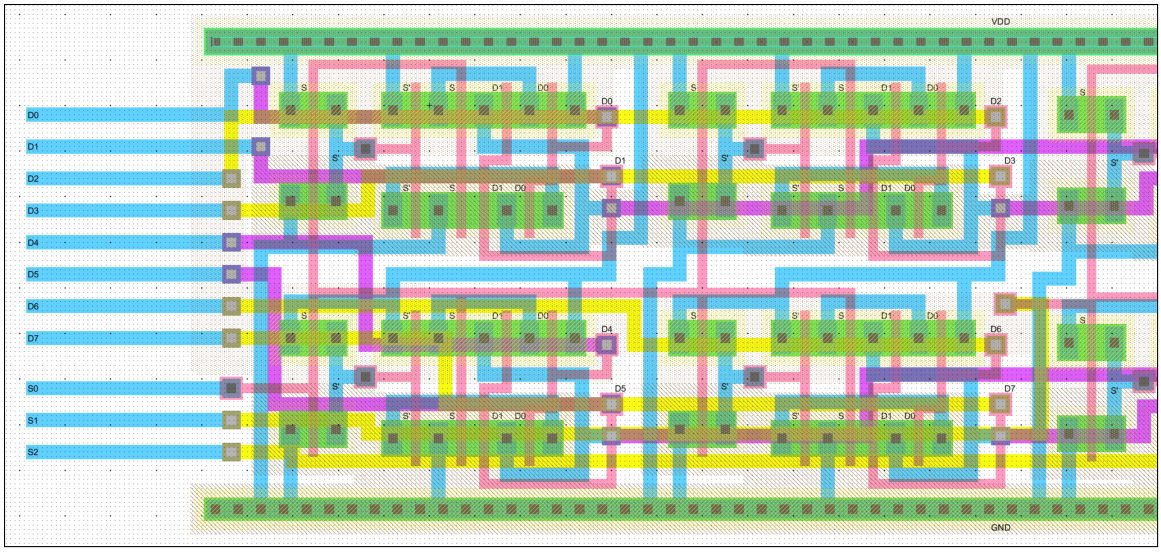

Figure 30: A close up of the first stage of the multiplexer.

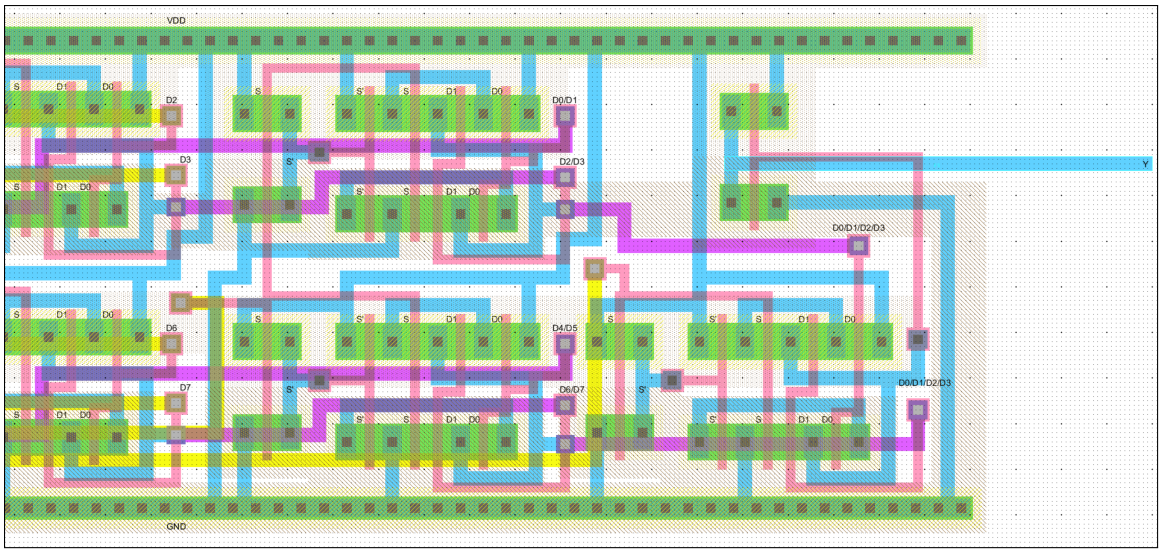

Figure 31: A close up of the last two stages and final inverter of the multiplexer.

The DRC results are shown in Figure 32 and the well check results are shown in Figure 33, both showing no errors.

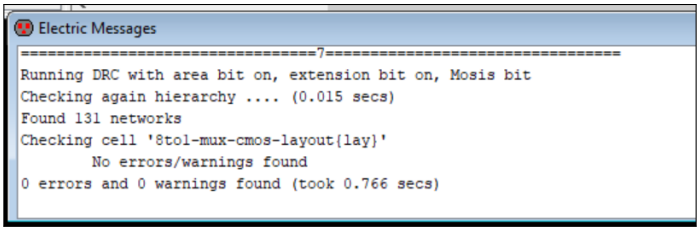

Figure 32: The DRC results for the CMOS layout showing no errors.

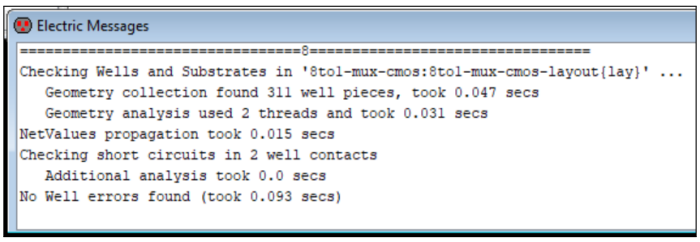

Figure 33: The well check results for the CMOS layout showing no errors.

7 LTSpice Simulation for Layout

This section will detail the LTSpice simulations for both the transmission gate schematic and the CMOS layouts. As previously mentioned, the spice code used to generate the graphs for this simulation are shown in Figure 14 and the code used to generate the measurements is shown in Figure 15.

7.1 Transmission Gate Layout LTSpice

The spice simulation for the transmission gate layout is shown in Figure 34. As described in previous sections, the behavior in the graph is as expected. Each input is passed through to the output when selected by the select line.

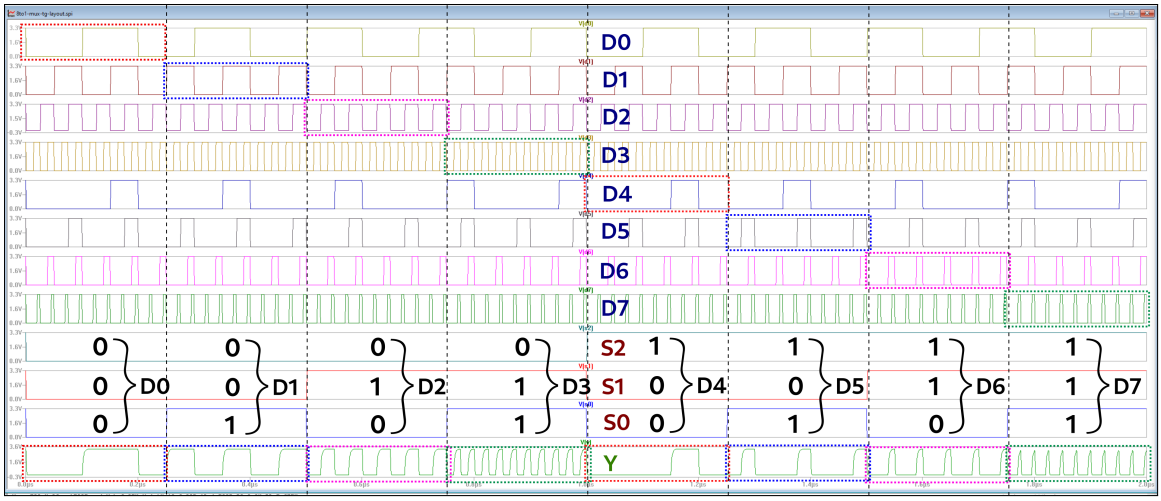

Figure 34: The LTSpice simulation of the transmission gate layout showing that each combination of inputs to the select line passes through the corresponding inputs D0 through D7.

The measurements obtained with the spice code shown in Figure 15 for the transmission gate layout are shown in Figure 35.

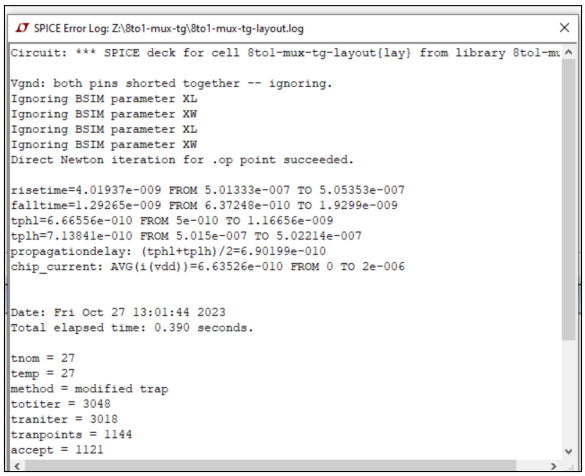

Figure 35: The measurements obtained through LTSpice simulation of the transmission gate layout.

7.2 CMOS Layout LTSpice

The spice simulation for the CMOS layout is shown in Figure 36. As described in previous sections, the behavior in the graph is as expected. Each input is passed through to the output when selected by the select line.

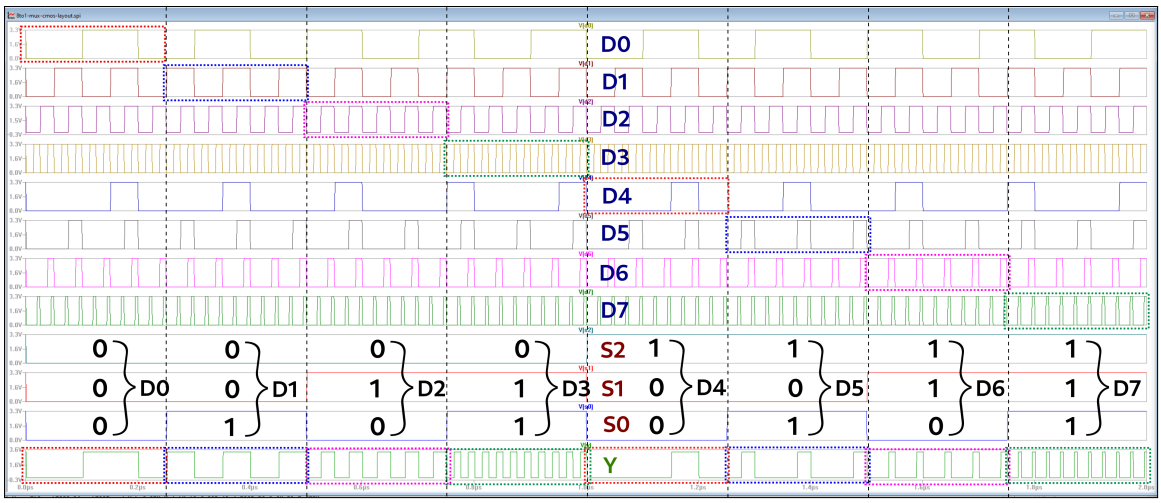

Figure 36: The LTSpice simulation of the CMOS layout showing that each combination of inputs to the select line passes through the corresponding inputs D0 through D7.

The measurements obtained with the spice code shown in Figure 15 for the CMOS layout are shown in Figure 37.

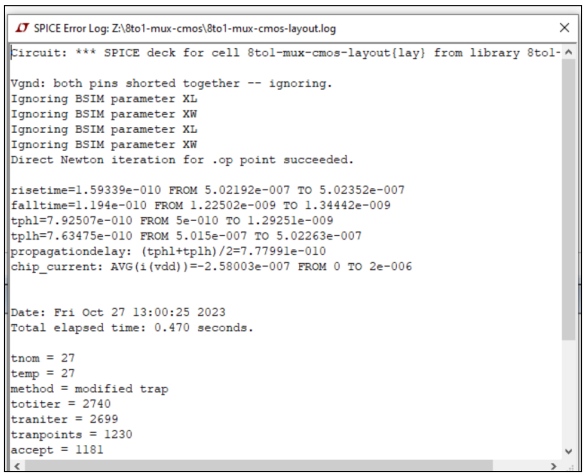

Figure 37: The measurements obtained through LTSpice simulation of the CMOS layout.

Parameter	TG Schematic	TG Layout	CMOS Schematic	CMOS Layout
Rise Time	1.47ns	4.02ns	80.8ps	159ps
Fall Time	7.27ns	1.29ns	39.2ps	119ps
t_{pHL}	273ps	667ps	477ps	793ps
t_{pLH}	239ps	714ps	357ps	763ps
t_p	256ps	690ps	417ps	778ps
Power Draw	8.17nW	2.19nW	473nW	851nW
Chip Dimensions (λ)	N/A	87λ x 429.5λ	N/A	362λ x 120λ
Chip Dimensions (μm)	N/A	15.2μm x 75.2μm	N/A	63.4μm x 21μm
Chip Area(λ^2)	N/A	$37.4*10^3\lambda^2$	N/A	$43.4*10^3\lambda^2$
Chip Area (μm^2)	N/A	$1.14*10^3\mu m^2$	N/A	$1.33*10^3\mu m^2$
Transistor Count	54	54	72	72

Table 3: The measurements results for the XOR gate schematic and layout in LTSpice.

8 Conclusion

In this project, I designed the schematic for an XOR gate using CMOS logic, turned that schematic into a layout, simulated both in LTSpice and IRSIM, and then took measurements on both in LTSpice. I learned a lot in the process about digital IC design, Electric, LTSpice, and the various parameters that make a design good or bad. I enjoyed the process of turning a simple logic gate into a schematic and then into a completed layout far more than I expected, and it was interesting to be able to analyze the final designs with LTSpice and IRSIM.

9 References

[1] EE 457 Lecture 1, 2, and 3
[2] https://cmosedu.com/videos/electric/tutorial3/electric_tutorial_3.htm
[3] https://cmosedu.com/videos/electric/tutorial4/electric_tutorial_4.htm
[4] https://en.wikipedia.org/wiki/XOR_gate