The City University of New York

EE457: Digital IC Design

Fall Semester 2017

Final Project Report Cover Sheet

Due 12/18/2017, 3:30PM

PROJECT TITLE: **128 bit SRAM: 16 rows, 8 columns**
Group Name: **Pepe & Ramos**
Student Names: **Ankit Shrestha and Chhewang Sherpa**

Check for completion	Topics	GRADES
	Section1: Executive Summary	/5
	Section 2: Introduction and Background	/5
	Section 3: Circuit Schematics	/10
	Section 4: Detailed Electric Layouts	/20
	Section 5: IRSIM Logic Simulations and Measurements for Layout and Schematic (must provide comparisons between the two)	/10
	Section 6: LTSPICE code and parasitic extractions	/10
	Section 7: Measurements in LTSPICE for delays for Layout and Schematic (must provide comparisons between the two)	/15
	Section 8: Measurements of power, delay, chip area, timing, number of transistors for the layout. (If you are using TG, static or dynamic CMOS, compare here.)	/10
	Section 9: Conclusion and References	/5
	Section 10: Required Presentation (submit soft copy of your PPT file and this report in a separate CD or USB in class with all of your Electric and LTSpice files.)	/10
	TOTAL	**/100**

We want the chip fabricated and test it through Independent Study. YES____, NO_____

I. Executive Summary

Static Random-Access memory or simply SRAM are very important part of many applications and they are required in data storage embedded applications, cache memories, microprocessors. Today, large SRAM arrays that are widely used as cache memory in microprocessors and application-specific integrated circuits occupy a significant portion of the die area [1]. For high-speed memory applications such as cache, a SRAM is often used because of its access time, speed, and power consumption. These are the three key parameters for an SRAM memory design(SRAM). In this project, an effort is made to design 16X8 SRAM memory. The integrated SRAM is operated with analog input voltage of 3.3v. The 16x8 SRAM memory has been designed, implemented & analyzed in Electric VLSI design system and LTspice. Our first step of this project was to design 6T memory cell, which consists of 6 transistors and it is most commonly used as an SRAM cell. Following the memory cell, the next step was to design a write driver circuit to write data in the memory cell and after that we designed sense amplifier to read data from the cell. After testing a single cell, we had to make 16 rows and 8 columns of the same cell and use row decoder to choose the specific row and write 8 bit words of data since one memory cell holds 1 bit of data.

II. Introduction and Background

The SRAM is a major component which occupy larger area of the chip die and for SOC designs, the technology selection and system design choices are mainly driven by digital circuit requirements. The demand for static random-access memory (SRAM) is increasing with large use of SRAM in mobile products, System On-Chip (SoC) and high-performance VLSI circuits. 70% of the area in System On-Chip (SoC) is consumed by SRAM memory [2]. We need memory cell to hold a data and one memory cell can hold a single bit of data. SRAM can retain a data until the power is on. Unlike dynamic RAM or DRAM, SRAM does not require periodic refreshing. In this paper, we are using 6T memory cell to design our SRAM. Besides 6T, there are other memory cells used in SRAM such as 4T to 12T. However, for this project we are using 6T memory cell SRAM due to its low power consumption and compactness. If we want to use 6T memory cell SRAM then we need external circuitry to perform read and write operation in our memory. Basic SRAM's block diagram can be seen in the block figure below.

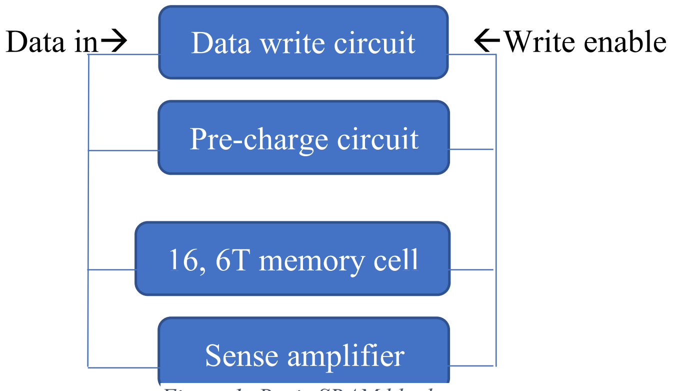


Figure 1: Basic SRAM block

The same block is made 8 times in horizontal direction to form a 16X8 SRAM array. One 4:16 row decoder is used to access a specific row cell in the array. This project has 9 sections including the executive summary, and introduction. After executive summary and introduction, section 3 covers the schematic design of SRAM and all the peripheral circuits. Section 4 covers the layout design of the schematic. Section 5 is the simulation part where we compare the IRSIM simulation of both schematic and layout. Section 6 included the LTspice code that we used for our simulation and measurement and also the parasitic extraction of our design. Section 7 compares result of LTspice simulation of schematic and layout. Section 8 is the measurement of our design and how efficient it is in terms of power consumption, time delay, chip area, timing, and the number of transistors. Lastly, section 10 includes the conclusion and references.

III. <u>Circuit Schematic</u>

The basic Static RAM cell is the six transistor (6T) cell as shown in Fig. 2. The advantage of 6T cell is that static power dissipation is very less; essentially, it is limited by the PMOS transistors. It has high noise immunity [3]. In existing SRAM topologies of 8T, 9T and higher transistor count, the read static. noise margin (SNM) is increased but size of the cell and power consumption increases relatively [4]. The basic 6T SRAM cell consist of two cross coupled CMOS inverters along with two pass or access transistors. The access transistors are two NMOS whose drain/source is connected to the outputs of the inverters. Whereas, its gate is connected to the word line, which is represented as WL in the figure below. The access transistors are turned on when word line is logic 1. We need word line to be high or logic 1 to perform read and write operation on that specific row. When word line is logic 0 or low, then the NMOS transistors are turned off from the bit lines, which are connected to the drain/source of the access transistors, and keeps it in standby mode.

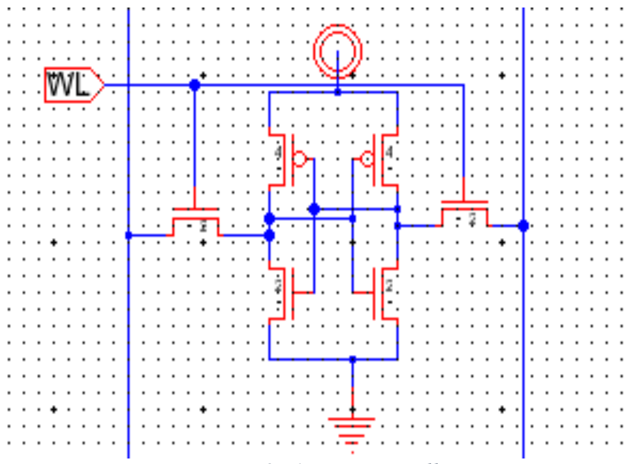


Figure 2: 6T memory cell

In figure 3, the read operation is performed. Assuming that 0 is stored on the left side of the cell and 1 on the right side. M1 is on and has a path to the ground. The BL and $\overline{BL}$ are pre-charged to Vdd. Whenever the word line or WL is activated to logic high or 1, access transistors M5 and M6 are turned on. Then current begins to flow through M5 to M1 to ground. Therefore, the memory cell discharges the capacitance Cbit. On the right side, the voltage on M6 remains high since there is no path to ground. The difference between BL and $\overline{BL}$ is fed to the sense amplifier and it senses and gives valid output, in this case logic low or 0.

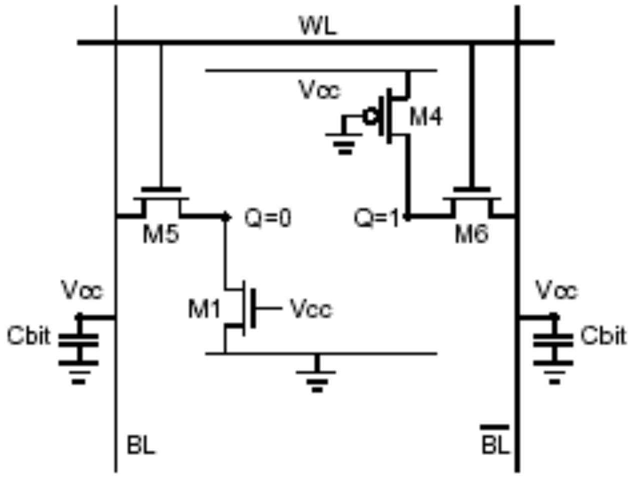


Figure 3: Read Operation

In order to write to the memory cell, the wordl both BL and $\overline{BL}$ should be giving input to the cell. For example, in BL an input of 1 is placed on the BL and 0 on $\overline{BL}$. By doing so, we can overwrite the previous value by the new value.

The next step was to design a pre-charge circuit to charge the bit line, BL, and bit line bar, $\overline{BL}$ to Vdd prior to our read and write operation. The circuit's figure can be seen in figure 4 below. Pre-charge circuit is nothing but two PMOS in parallel, whose gates are connected to clock and clock is 0 and the bit lines are represented by BL and BL' in the figure below.

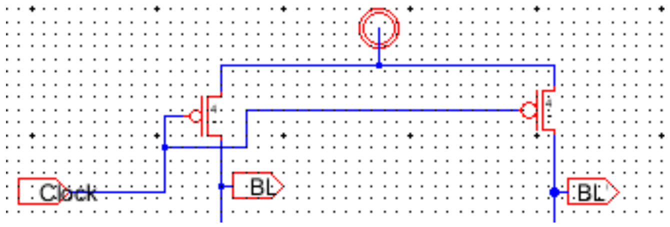


Figure 4: Pre-charge Circuit

After making the pre-charge circuit, we had to make a write driver circuit to write data to our memory cell. The function of the SRAM write driver is to discharge either bit line or bit line bar from the pre-charge level or Vdd to below the write margin of the SRAM cell. The write driver circuit can be seen in figure 5 below.

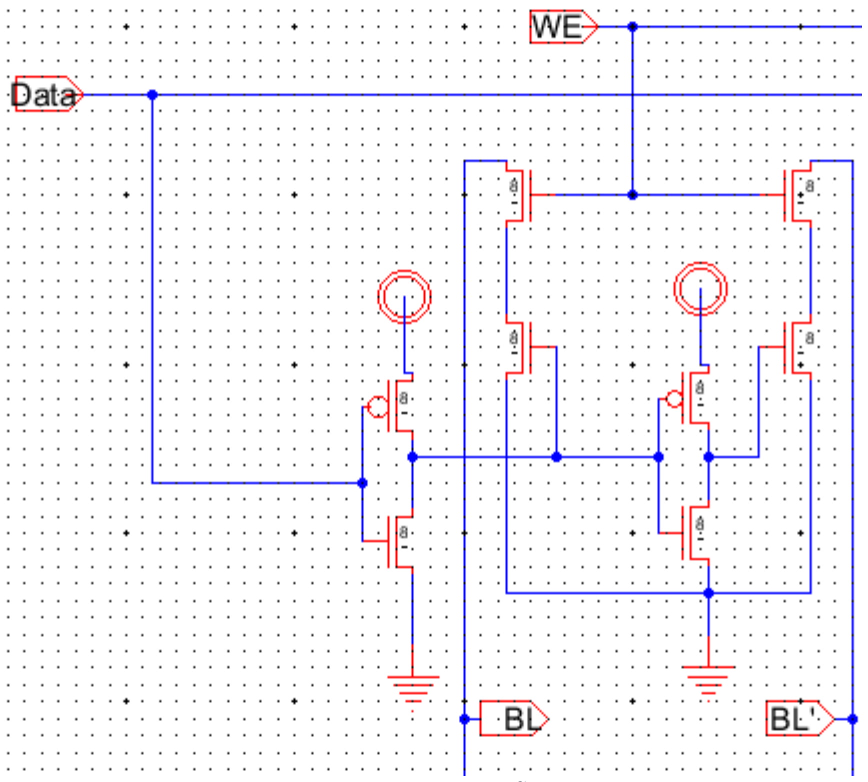


Figure 5: Write Driver Circuit

In figure 5, we can see that the gate of two NMOS transistor is connected to the write enable, which is labeled as WE, to turn on or activate the write drive circuit. The write driver circuit uses two stacked NMOS transistors to form pass transistors AND gates. The data is fed to an inverter and the output of the inverter is connected to the gate of the second stacked NMOS and to the input of the second inverter whose output is connected to the gate of the other NMOS transistor. The bit line and bit line bar are coming out of the drain of the two first stacked NMOS on either side and the source of the second stacked NMOS has a path to the ground from its source.

The last peripheral circuit that we had to design was a sense amplifier which is connected at the end of bit line and bit line bar. Sense amplifier are very important in memory design. The right design of sense amplifier defines the read speed and robustness of bit line sensing. The primary function of sense amplifier in SRAMs is to amplify a small analog differential voltage developed on the bit lines to reduce the time required for a read operation. The sense amplifier's figure can be seen in figure 6 below.

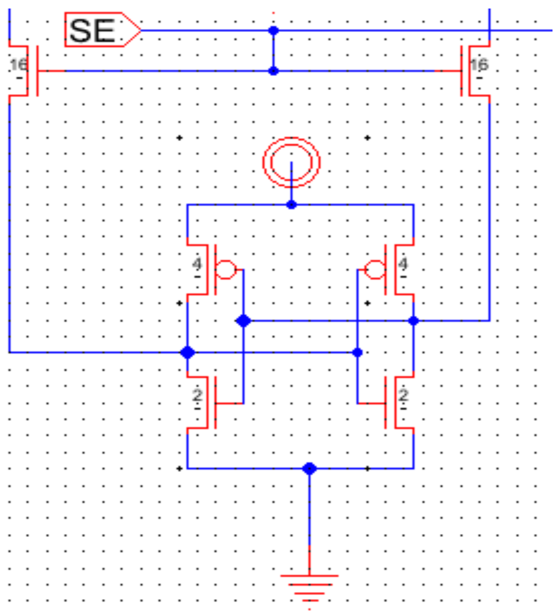


Figure 6: Sense amplifier

The above sense amplifier is called as the differential sense amplifier. It is formed by cross coupling two inverters similar to the 6T memory cell. The sense amplifier operates when sense enable, which is represented by SE, is logic high or 1. Thus, it activates the two NMOS whose gates are connected to SE and the drain/source to bit line and bit line bar.

After designing all the peripheral circuits required for one column, we had to design one last peripheral circuit called row decoder to choose the specific row and activate the cells from that row. The output of the row decoder goes to the word line of SRAM and activates that row. We designed 4-16 row decoder since our project is 16X8 SRAM and it has 16 rows. The gate figure of row decoder can be seen in figure 7 below.

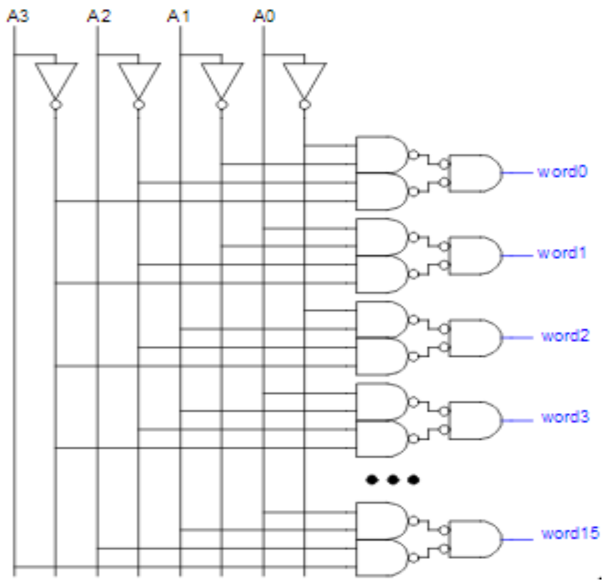


Figure 7: Row Decoder

Here in the figure, we see that two of the two input NAND gates are used and their output is fed to the output of two input AND gate. We use this scheme instead of 4 inputs row decoder because our total row of SRAM is 16 and row decoder must have 4 inputs and when our number of input is equal to or greater than 4 then NAND gates become slow. Therefore, we break the gates into multiple smaller gates. The schematic of the single row decoder can be seen in figure below in figure 8. In the figure A and A' are the input and the Vout is the output which is connected to the word line of SRAM.

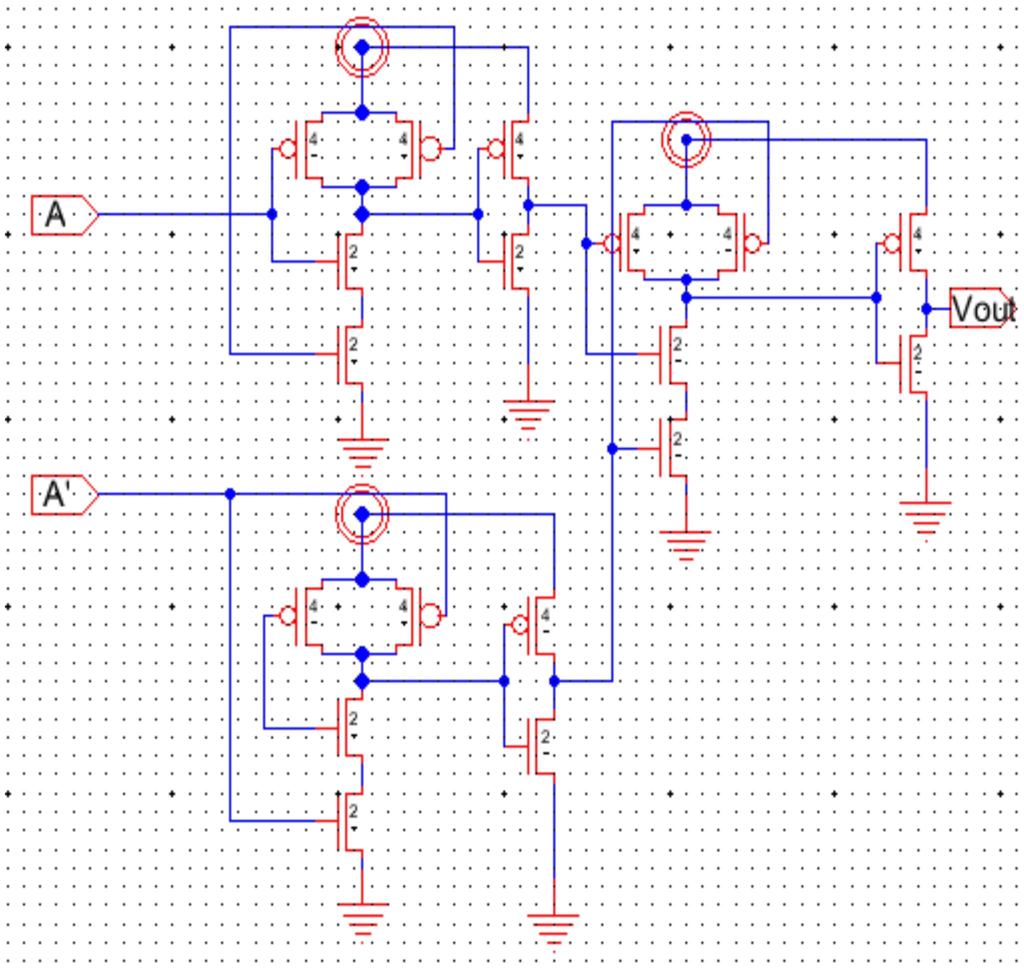


Figure 8: Single Row Decoder Circuit

The complete schematic of 4-16 row decoder can be seen in figure 9 below.

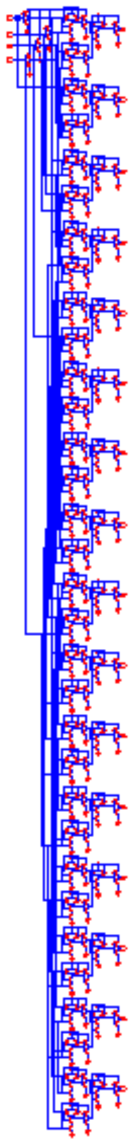


Figure 9: Schematic of Row Decoder

After making 16x8 memory cell, and connecting the peripheral circuits, the overall schematic was designed. The figure of overall 128 bits: 16 rows, 8 columns, 8-bit words SRAM can be seen in figure 10 & 11 below.

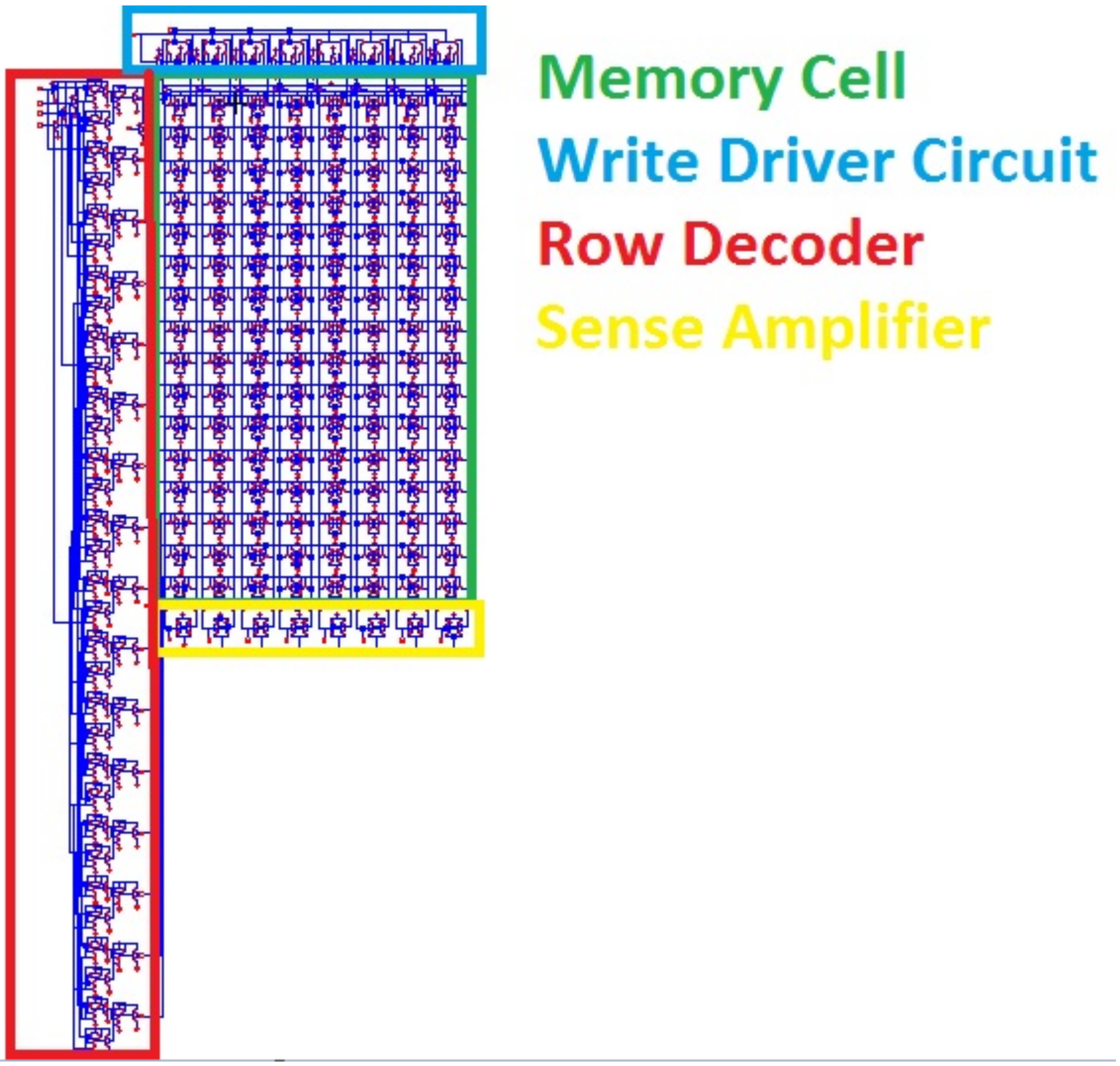


Figure 10: Labeled Schematic of SRAM

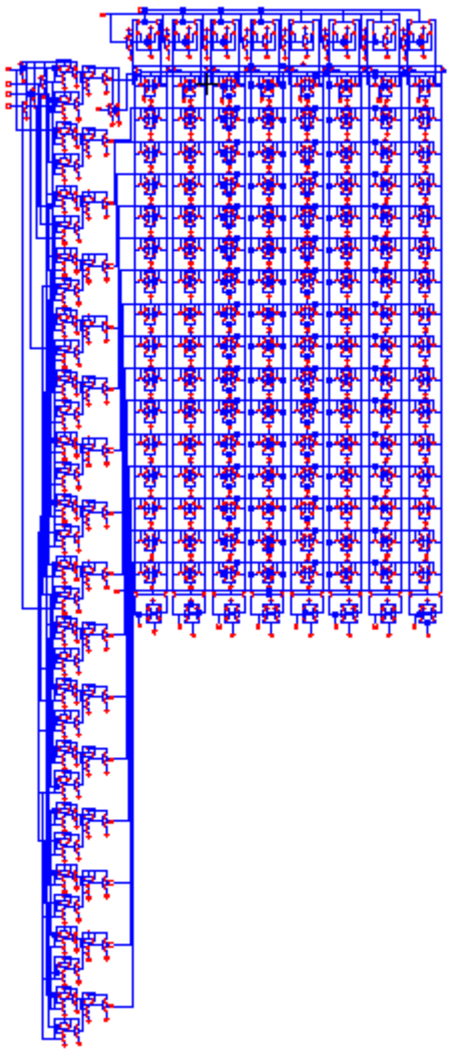
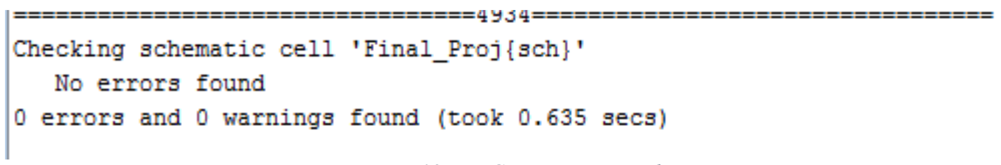


Figure 11: Complete Electric Schematic of SRAM

After completing the design of SRAM, we had to check for any design rule errors. Therefore, we performed DRC error check for the schematic. The DRC error message box can be seen in figure 12 below.



```
=================================4934=================================
Checking schematic cell 'Final_Proj{sch}'
   No errors found
0 errors and 0 warnings found (took 0.635 secs)
```

Figure 12: DRC error message box

IV. Detailed Electric Layout

Next step of this project after designing the SRAM schematic was to successfully design SRAM's layout. We designed all the peripheral circuits like we did in schematic. First, we designed the 6T memory cell. The stick diagram of 6T memory cell can be seen in figure 13 below.

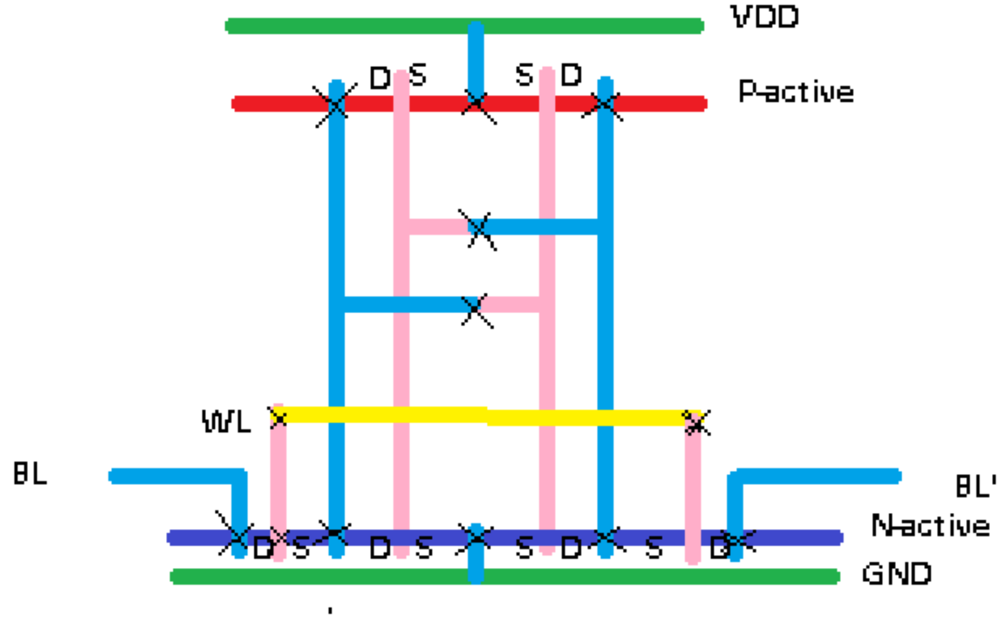


Figure 13: 6T memory cell Stick Diagram

From the stick diagram's figure above, we have labeled VDD in green color, p-active in red, N-active in blue, and GND in green. The polysilicon metal is pink in color, where as the metal that connects the drain and the source of the inverter is blue in color. We are using cross coupled inverters and we have used the merge technique since both inverters share the same node to VDD and same Source to GND. The output of the let inverter is connected to the input of the right inverter and vice versa. The pass transistors are the small polysilicon to the far left and right, whose gates are connected by the yellow colored metal and the drains go to bit line (BL) and bit line bar (BL'). The sources of pass transistors are merged with the NMOS part of the inverter since they are connected and share the same node. The layout of 6T memory cell designed in electric can be seen in figure 14 below.

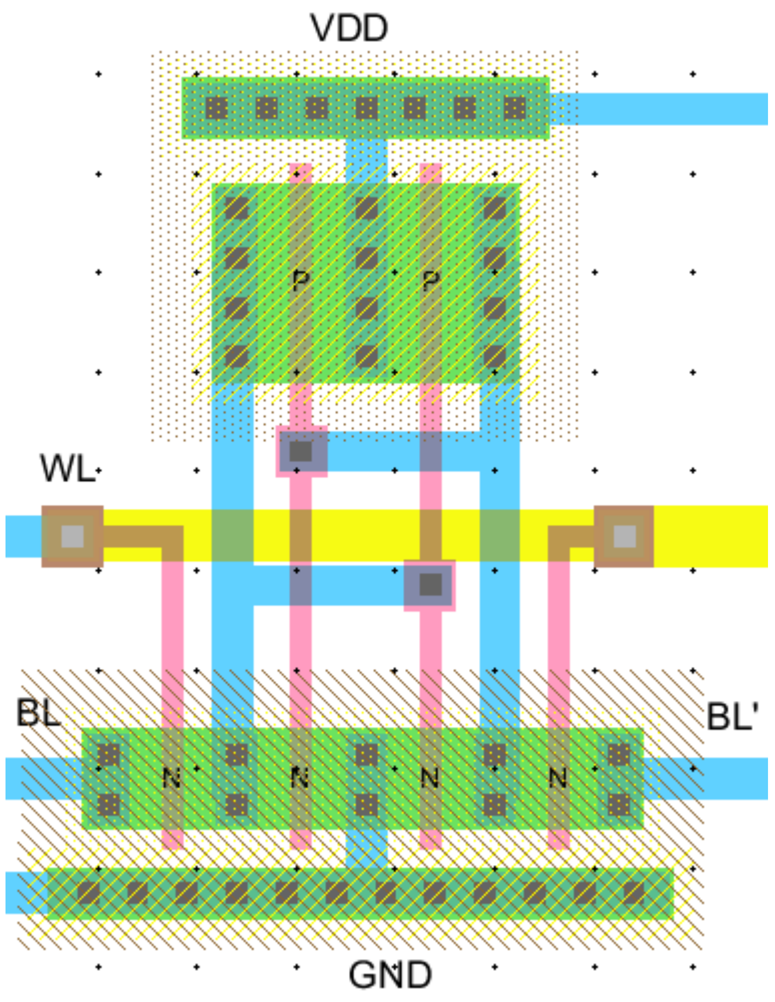


Figure 14: 6T memory cell Electric Layout

After designing 6T memory cell layout the next step was to design a pre-charge circuit to charge the bit line, BL, and bit line bar, $\overline{BL}$ to Vdd prior to our read and write operation. The circuit's stick diagram figure can be seen in figure 15 below.

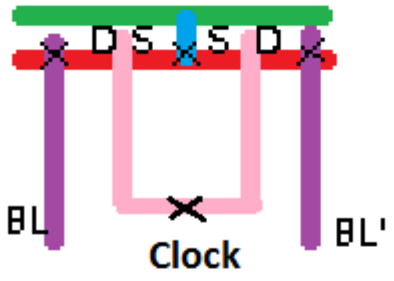


Figure 15: Precharge circuit Stick diagram

Pre-charge circuit is nothing but two PMOS in parallel, whose gates are connected to clock. In pre-charge circuit, we have used the merge technique since the source of both PMOS are connected to VDD. The bit lines are represented by BL and BL' in the figure below.

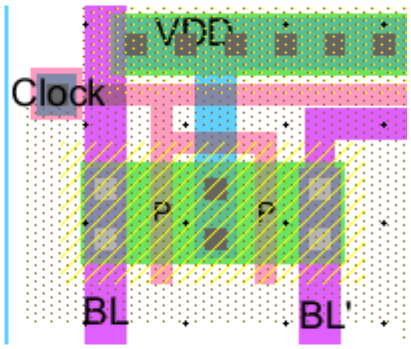


Figure 16: Pre-charge circuit's layout

After making the pre-charge circuit, we had to make a write driver circuit to write data to our memory cell. The function of the SRAM write driver is to discharge either bit line or bit line bar from the pre-charge level or Vdd to below the write margin of the SRAM cell. The write driver stick diagram can be seen in figure 17 below.

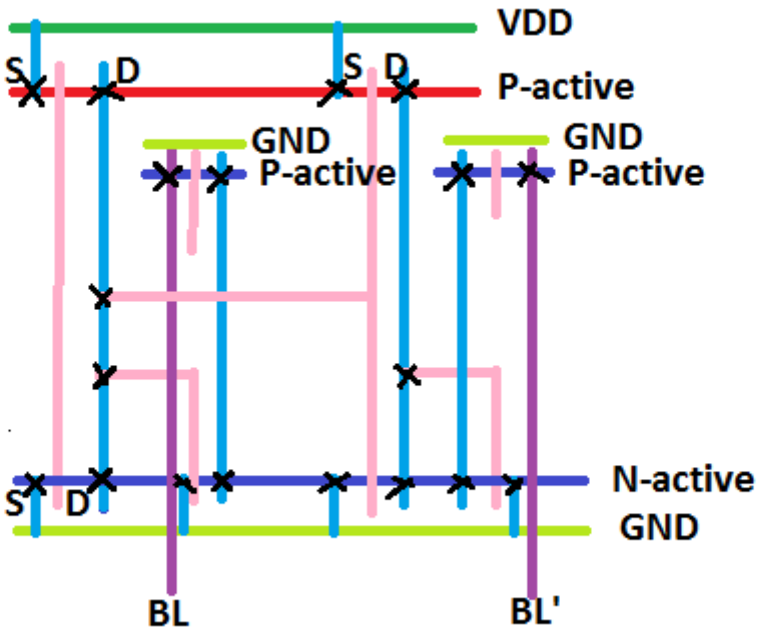


Figure 17: Write Driver stick diagram

From the figure above, we can see that VDD is represented by the green line, p-active by red, N-active by blue, and GND by lime green. The polysilicon metals are pink in color and the contacts are marked as X close to its respective polysilicon metals and wires. For the write driver circuit, the data is fed into the inverter and the inverted output goes to the gate of NMOS and to the input of other inverter whose output goes to the gate of the right most NMOS. The NMOS above the NMOS is required for us to connect word enable and turn the NMOS on and its source is connected to the source of the lower NMOS and the drains are the bit lines which are represented by BL and BL'. The electric layout of the write driver circuit can be seen in figure 18 below.

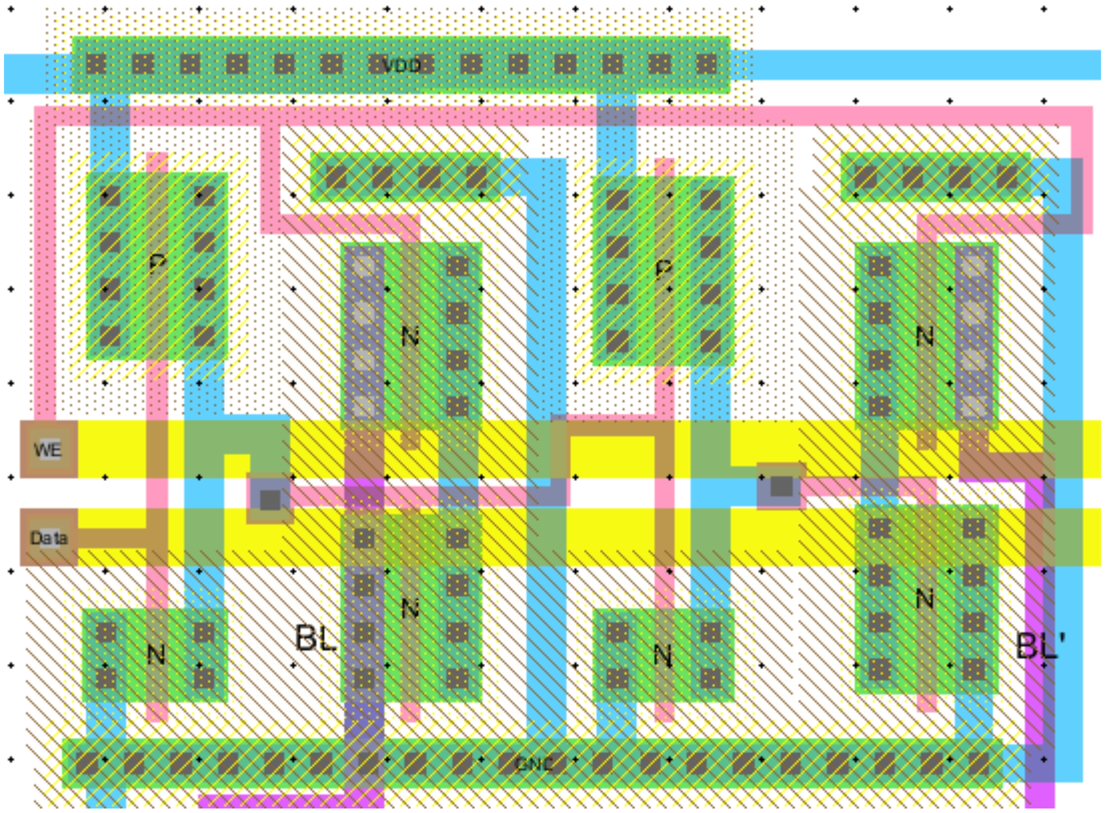


Figure 18: Write Driver Circuit Layout

The last peripheral circuit that we had to design was a sense amplifier which is connected at the end of bit line and bit line bar. The primary function of sense amplifier in SRAMs is to amplify a small analog differential voltage developed on the bit lines to reduce the time required for a read operation. The sense amplifier's stick diagram can be seen in figure 19 below.

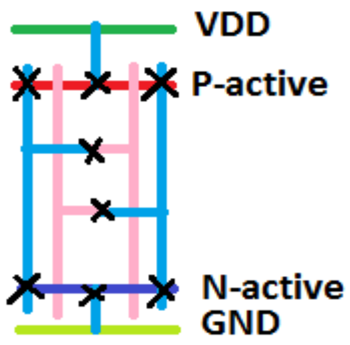


Figure 19: Sense amplifier Stick Diagram

The sense amplifier is nothing but two cross coupled inverter whose outputs are connected to their inputs like latch. The complete layout of the sense amplifier can be seen in figure 20. We added two NMOS above the sense amplifier and controlled its gate by sense enable, which is labeled as SE in figure below.

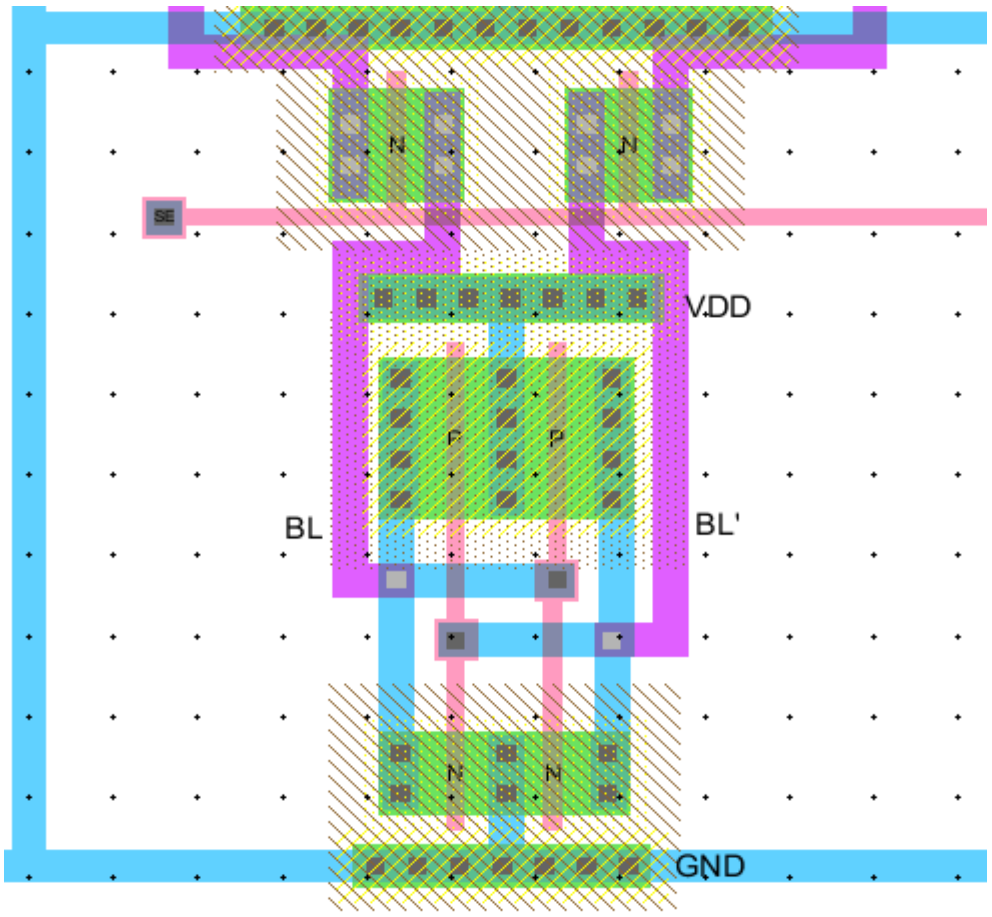


Figure 20: Sense amplifier Layout

After designing all the peripheral circuit for one column, now we had to design to choose a specific row out of 16 rows. For that we designed 2 inputs row decoder or essentially two input nand gate. To design its layout, we just had to design 2 input CMOS nand gate and the Euler's path, and stick diagrams are in figure 21 & 22 below.

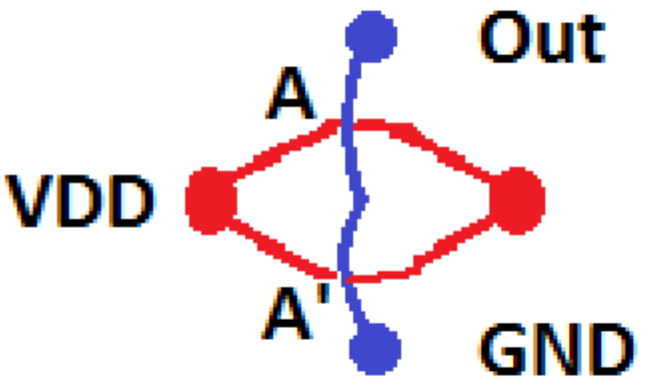


Figure 21: Euler's path of two input nand

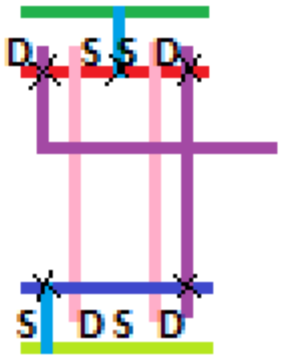


Figure 22: 2 inputs Decoder stick diagram

The output of the row decoder goes to the word line of SRAM and activates that row. We designed 4-16 row decoder since our project is 16X8 SRAM and it has 16 rows. We need 4 inputs to get 16 outputs. For that, two of the two input NAND gates are used and their output is fed to the output of two input AND gate. So, in total we have 3 gates. The layout of the single row decoder can be seen in figure 23 below.

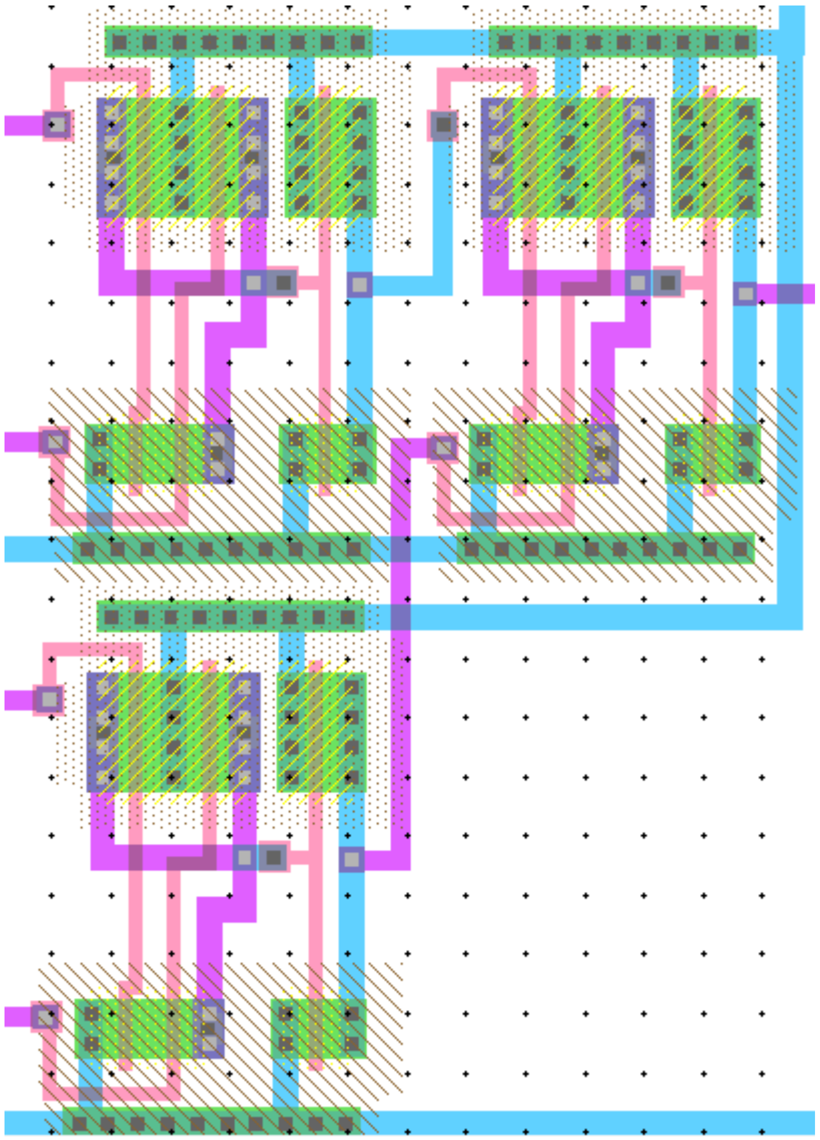


Figure 23: Single Row Decoder

The complete layout of row decoder can be seen in figure 24.

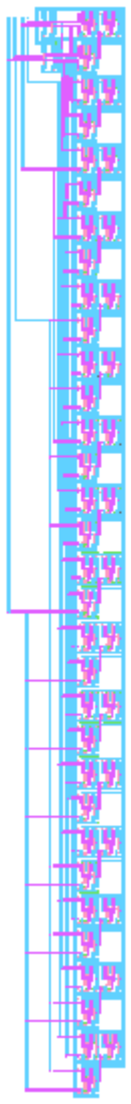


Figure 24: Complete Row Decoder Layout

After making 16x8 memory cell, and connecting the peripheral circuits, the overall layout was designed. The layout figure of overall 128 bits: 16 rows, 8 columns, 8-bit words SRAM can be seen in figure 25 & 26 below.

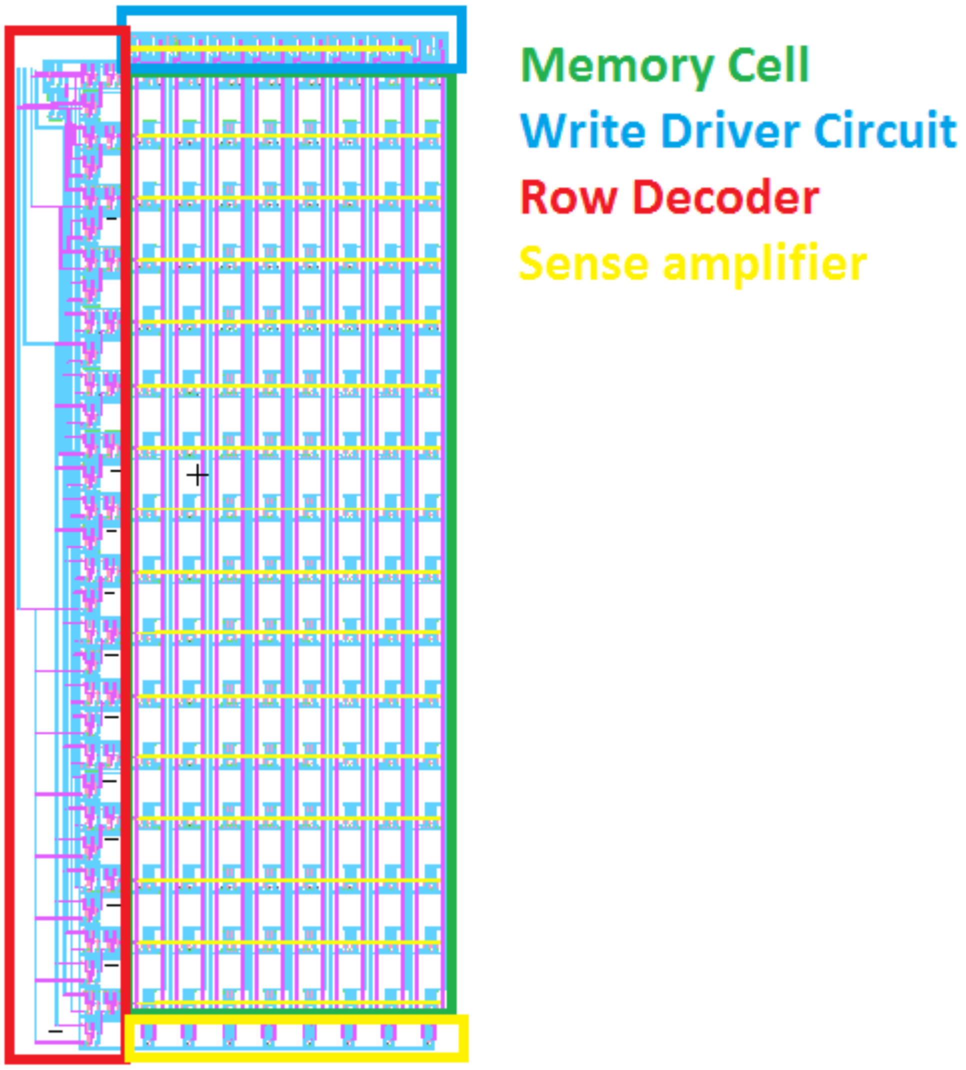


Figure 25: Labeled Layout

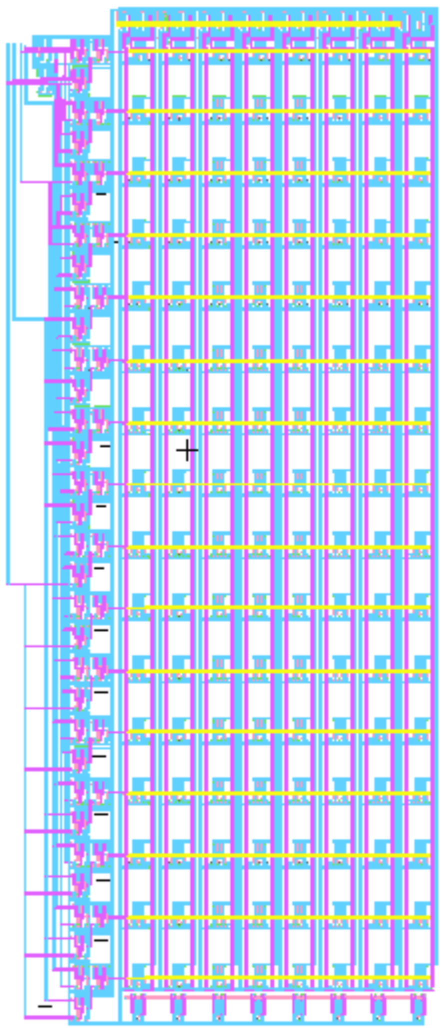
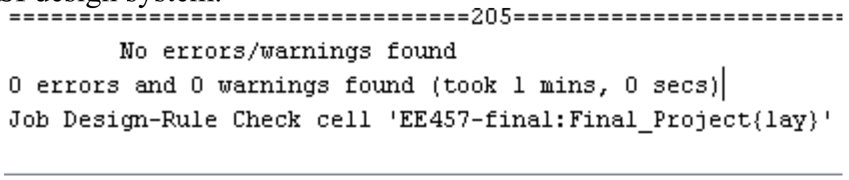


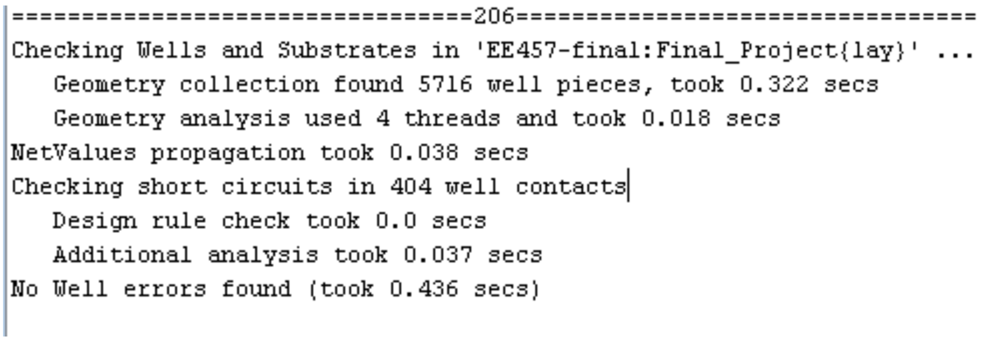
Figure 26: Complete Layout

After completing the layout design of SRAM, we had to check if there were any design rule errors or any well errors. Therefore, we performed DRC and ERC error check from Electric VLSI design system.



```
================================205=========================
        No errors/warnings found
0 errors and 0 warnings found (took 1 mins, 0 secs)
Job Design-Rule Check cell 'EE457-final:Final_Project{lay}'
```

Snapshot 1: DRC error message box



```
=================================206=================================
Checking Wells and Substrates in 'EE457-final:Final_Project{lay}' ...
   Geometry collection found 5716 well pieces, took 0.322 secs
   Geometry analysis used 4 threads and took 0.018 secs
NetValues propagation took 0.038 secs
Checking short circuits in 404 well contacts
   Design rule check took 0.0 secs
   Additional analysis took 0.037 secs
No Well errors found (took 0.436 secs)
```

Snapshot 2: ERC error message box

V. IRSIM simulation

The part of this project required us to simulate our schematic in IRSIM which is a plugin for Electric VLSI design system. The IRSIM tells us the logic behind our design. In figure 27, the IRSIM simulation of read operation is performed.

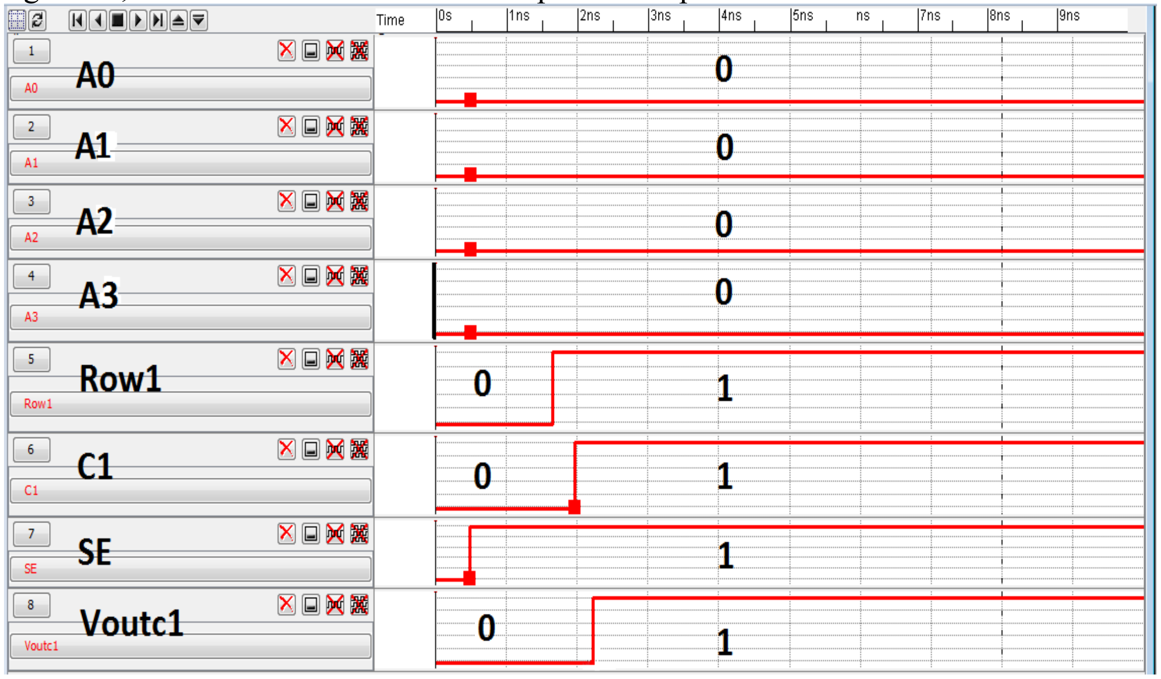


Figure 27: IRSIM simulation of Read Operation of Schematic

In the figure above, A0, A1, A2, and A3 are the input to the row decoder. We can see that we input 0 0 0 0 in our row decoder which sets Row1 to logic high or 1. C1 is the data we write in our memory cell and SE is the sense enable which allows us to read from the memory cell when SE is high. We can see that Voutc1 is logic low because when the sense enable is high at 0.5ns, the data that is written in the memory is 0, as can be seen in the figure. When the written data, in C1, changes to 1 at 2ns, Voutc1 reads 1. Thus, it tells that our read operation of SRAM circuit is working.

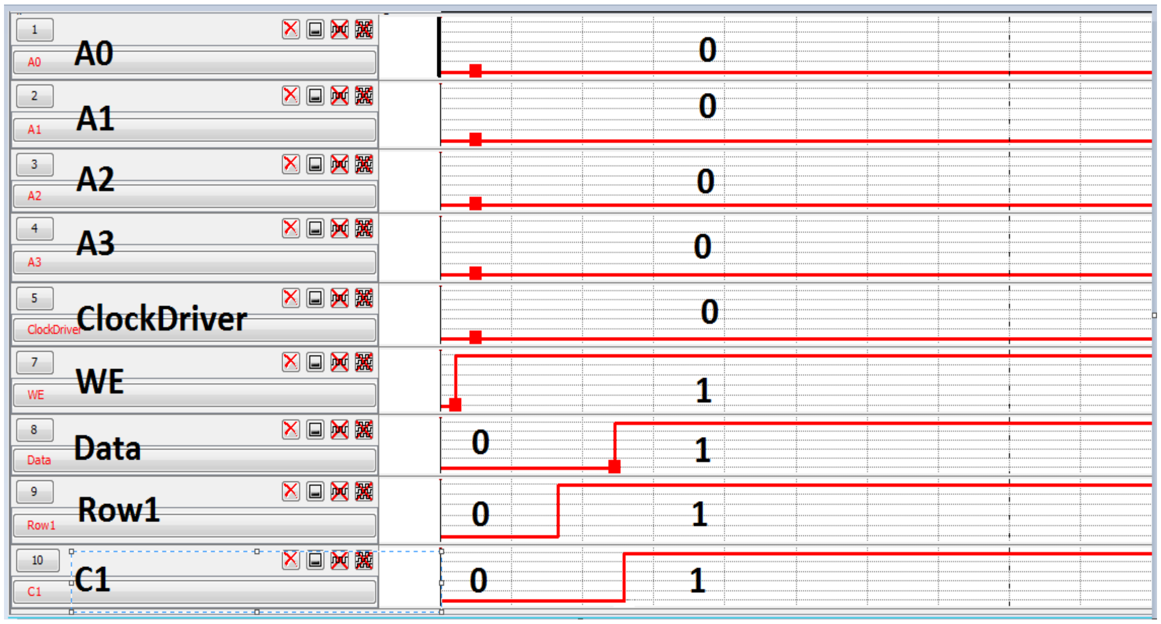


Figure 28: IRSIM simulation of write operation of Schematic

Figure 28 above is the write operation of schematic performed in the IRSIM simulatioin. We cannot show read and write operation in the same figure because for SRAM to perform read and write operation, there must be a difference of at least 10ns. Since IRSIM's domain is only from 0 9ns, performing read and write operation in the same figure was not possible. Like the read operation A0, A1, A2, and A3 are the input to the row decoder for the write operations as well. We can see that we input 0 0 0 0 in our row decoder which sets Row1 to logic high or 1. Clock driver must be set to logic 0 for the access transistors to be on and write enable must be turned on for us to be able to write in the memory cell. We see that the write enable is turned on at around 0.2 ns which is labeled as WE. Our data input was 0 and after the row decoder is set to high the we store 0 in the memory cell labeled as C1. When the data changes to 1, then the we overwrite the memory cell and it changes to 1 as well, as can be seen in figure above. Thus, we can conclude that our write operation of SRAM is also working properly without any errors.

After we finished the IRSIM simulation of schematic, I performed the similar procedure for the layout as well. The IRSIM simulation of the layout can be seen in figure 29 & 30 below.

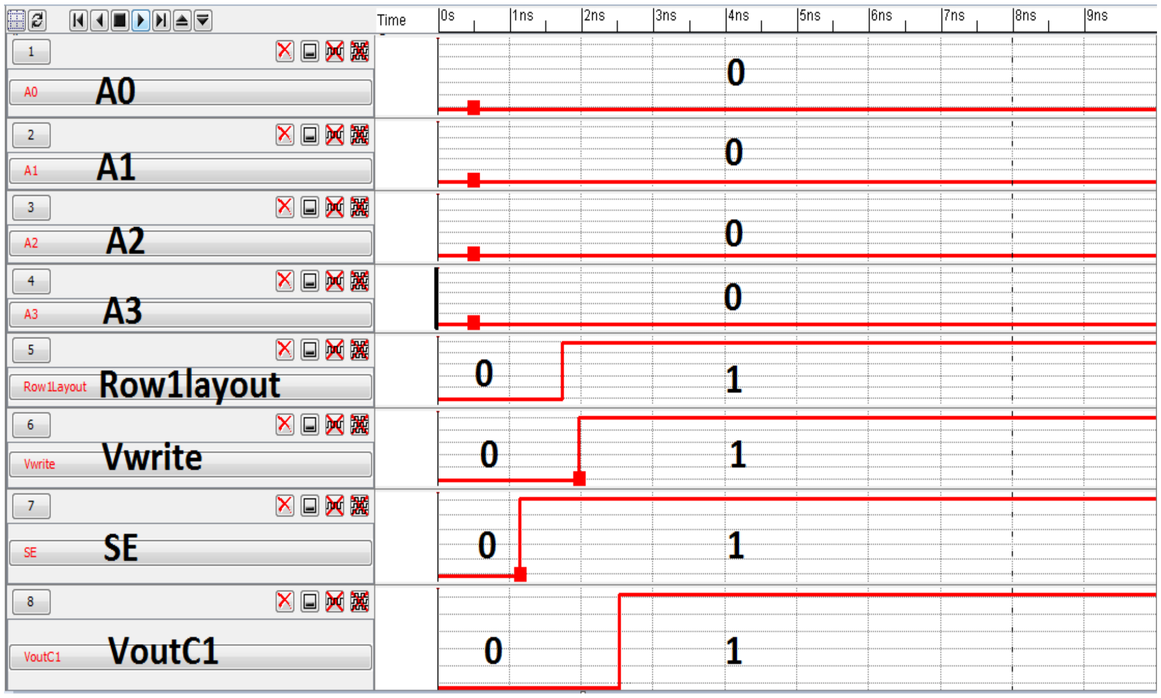


Figure 29: IRSIM simlation of read operatin of Layout

The IRSIM simulation of write operation of layout can be seen below.

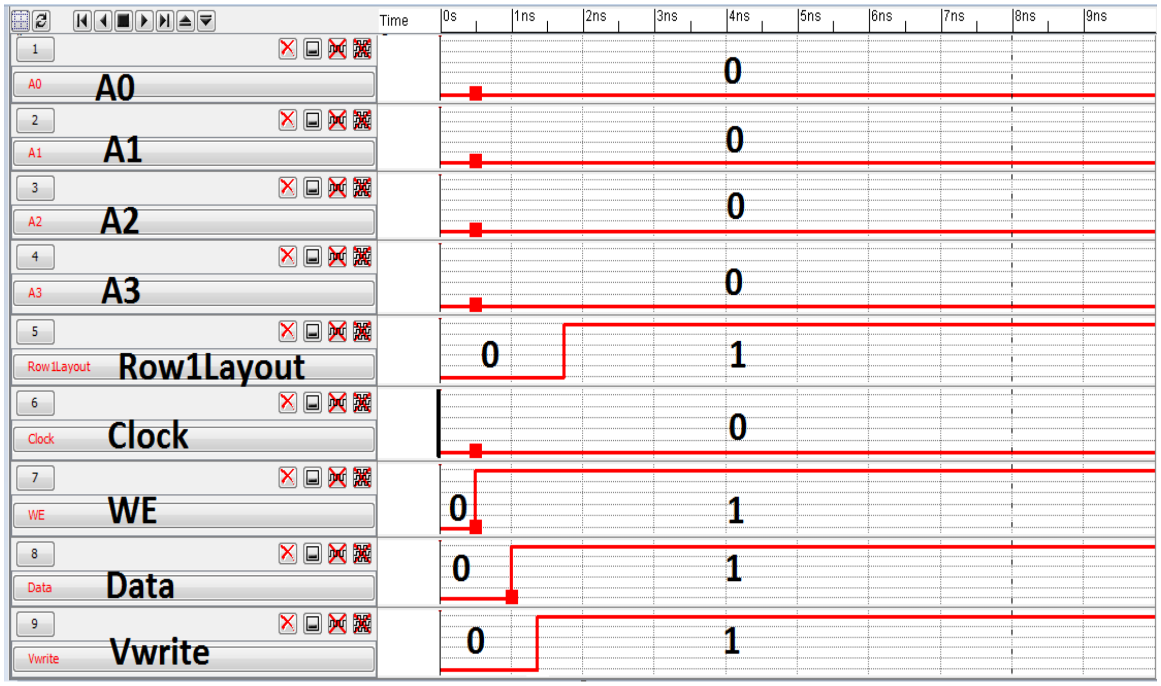
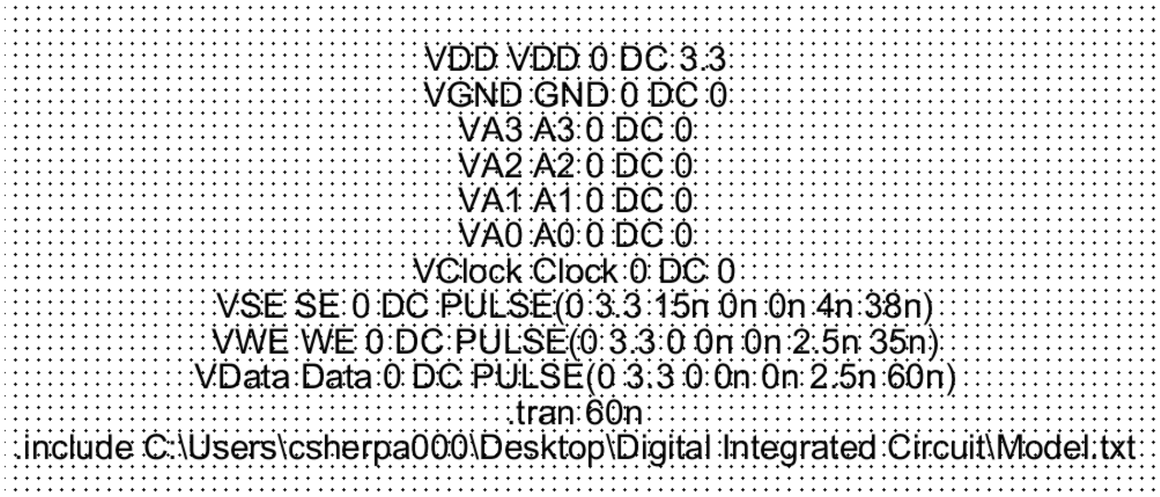


Figure 30: IRSIM simulation of write opeartion of layout

Comparing the IRSIM simulation of both schematic and the layout, we don't see much of a difference between them. For both the read and write operation, the layout's output response was slightly slower by some picoseconds. Overall, the read and write operation of SRAM cell was working for both schematic and the layout.

VI. LTspice code and Parasitic extraction

The spice code we used for our SRAM design is in figure 31 below.



```
VDD VDD 0 DC 3.3
VGND GND 0 DC 0
VA3 A3 0 DC 0
VA2 A2 0 DC 0
VA1 A1 0 DC 0
VA0 A0 0 DC 0
VClock Clock 0 DC 0
VSE SE 0 DC PULSE(0 3.3 15n 0n 0n 4n 38n)
VWE WE 0 DC PULSE(0 3.3 0 0n 0n 2.5n 35n)
VData Data 0 DC PULSE(0 3.3 0 0n 0n 2.5n 60n)
.tran 60n
.include C:\Users\csherpa000\Desktop\Digital Integrated Circuit\Model.txt
```

Figure 31: Spice code

We got the parasitic from LTspice and for the 6T memory cell, the total R= 37,838.08 Ohm, C = 839.68 fF since we have 16x8 = 128 6T memory cells. Similarly, for all the other peripheral circuits, the total R = 26095.856, and C = 659.15 fF.

VII. Measurements in LTSPICE for delays for Layout and Schematic

Once the circuit's schematic and layout design was completed, we had to use LTspice to simulate the circuit to see the behavior of the circuit. In figure 38 below, we are showing that the data, write enable, data written the row 1 column 1's cell represented as r1c1, sense enable, and the output read from the memory.

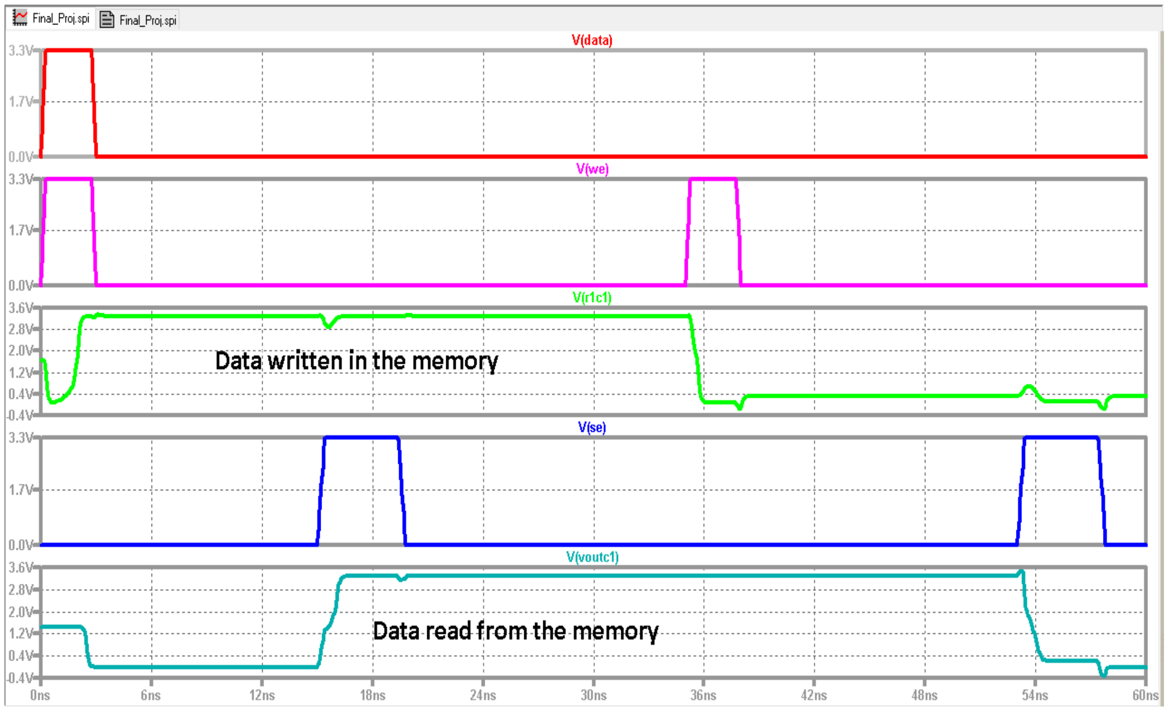


Figure 32: Read and Write operation of Schematic

To make sure every other rows and columns are performing read and write operation, we are showing the operation from different cells. Similarly, for row 2 column2, the operation of read and write operation is shown below in figure 33 and so on.

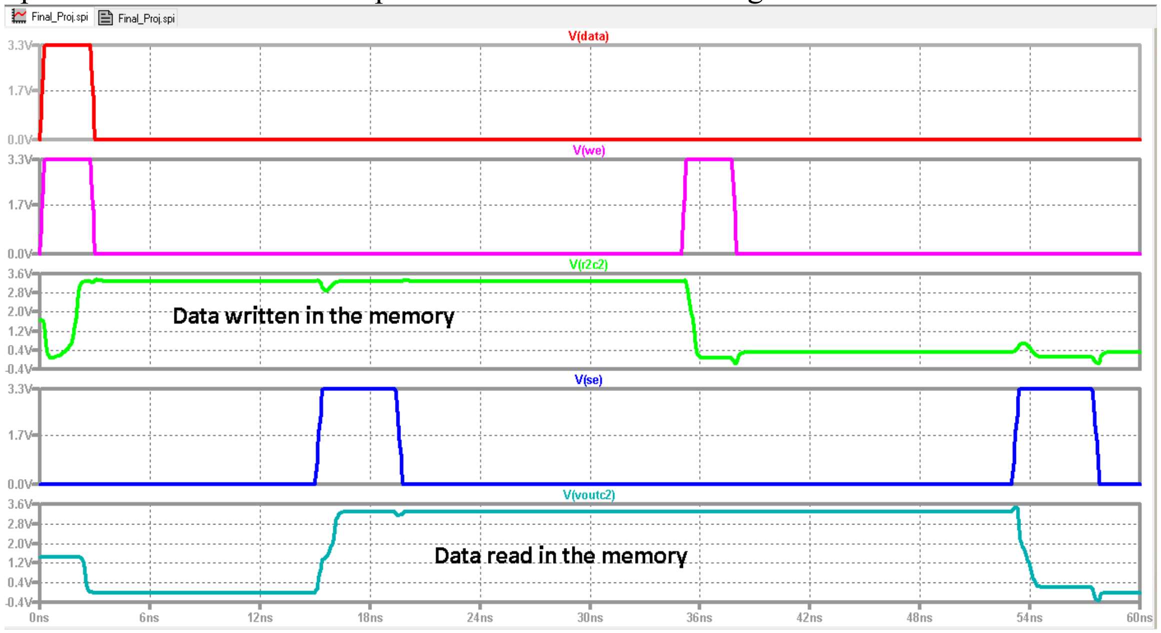


Figure 33: Read and Write operation of Schematic

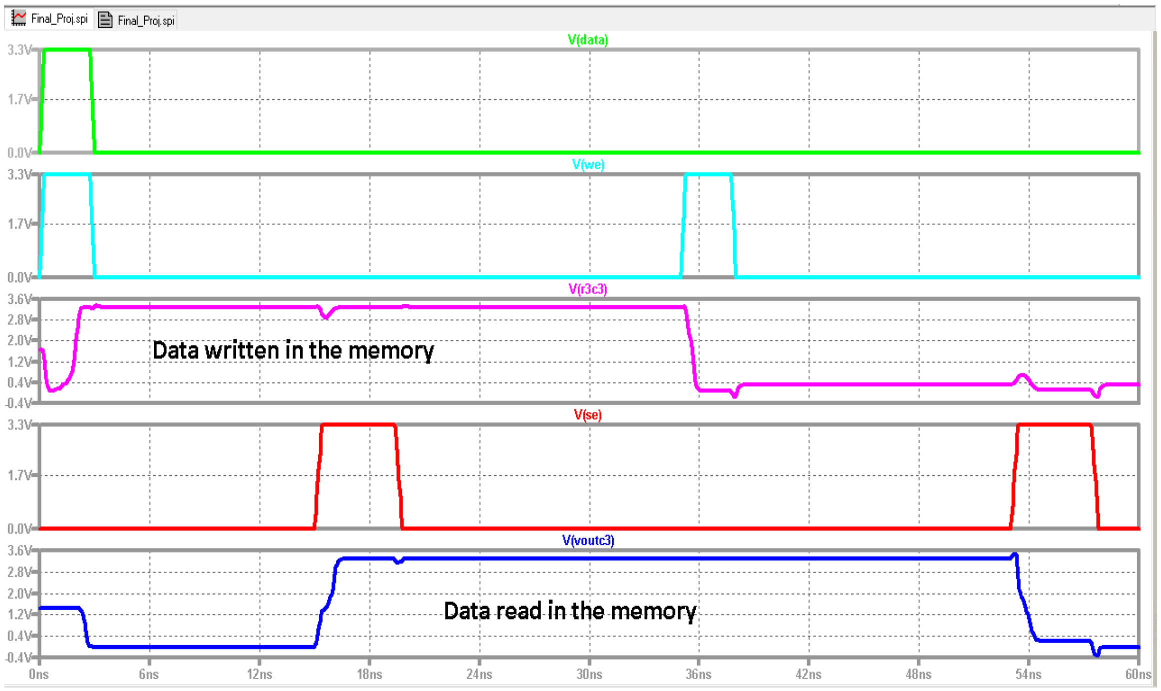


Figure 34: Read and Write operation of Schematic

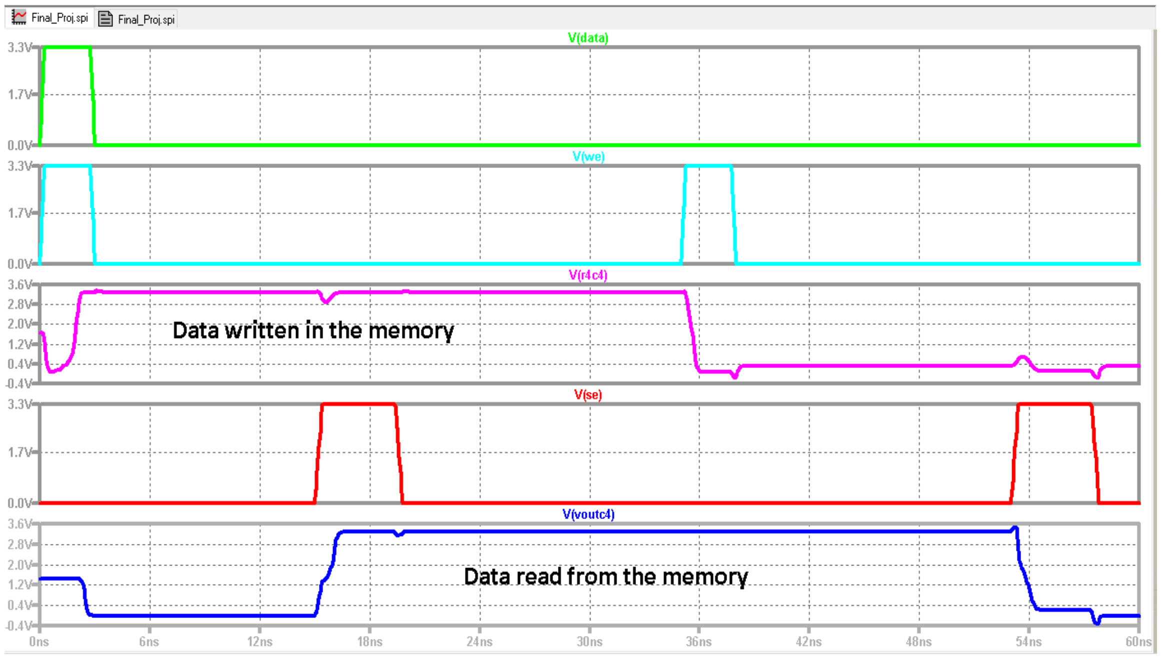


Figure 35: Read and Write operation of Schematic

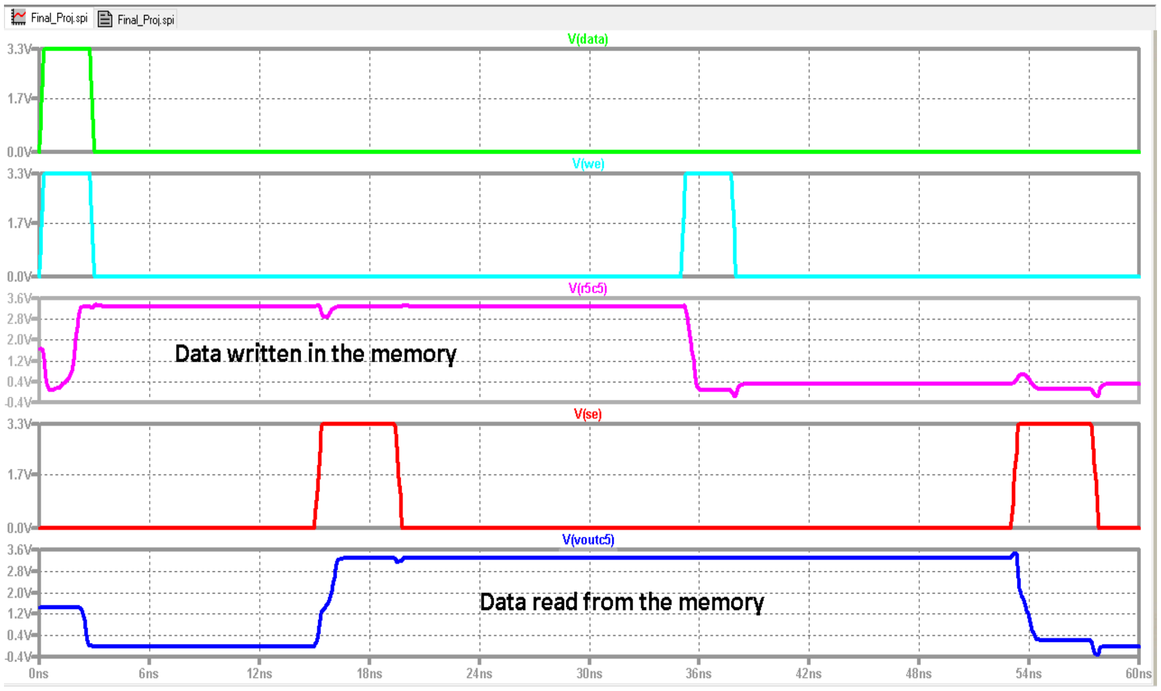


Figure 36: Read and Write operation of Schematic

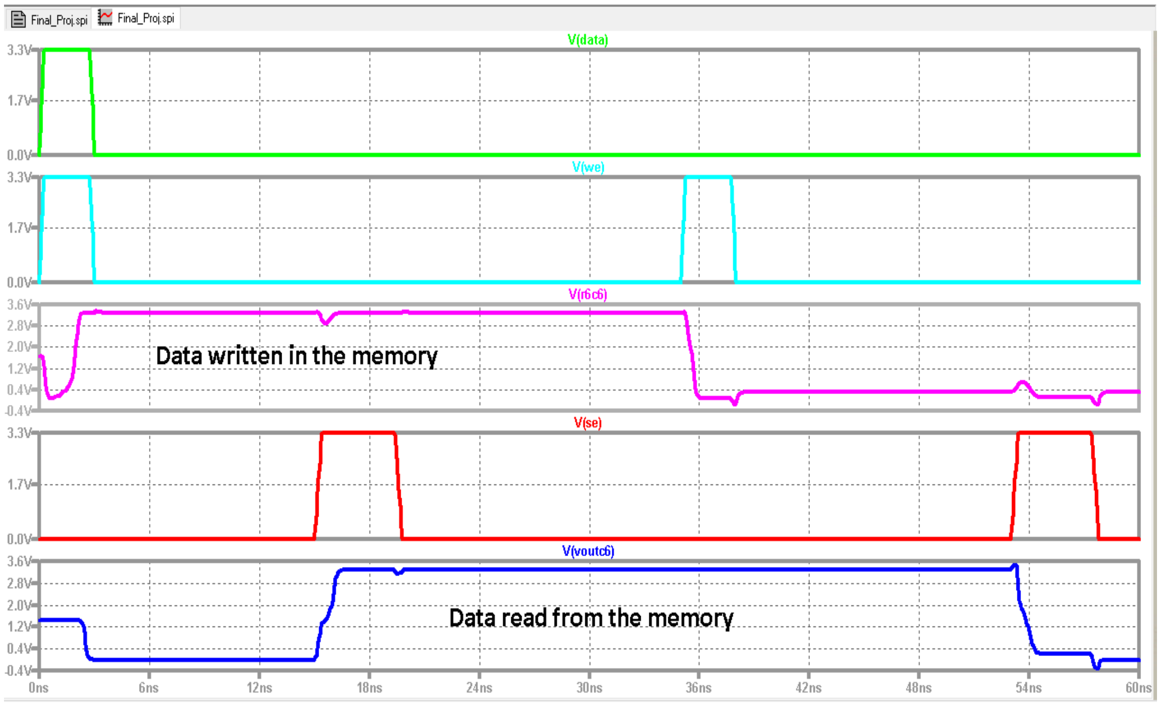


Figure 37: Read and write operation of Schematic

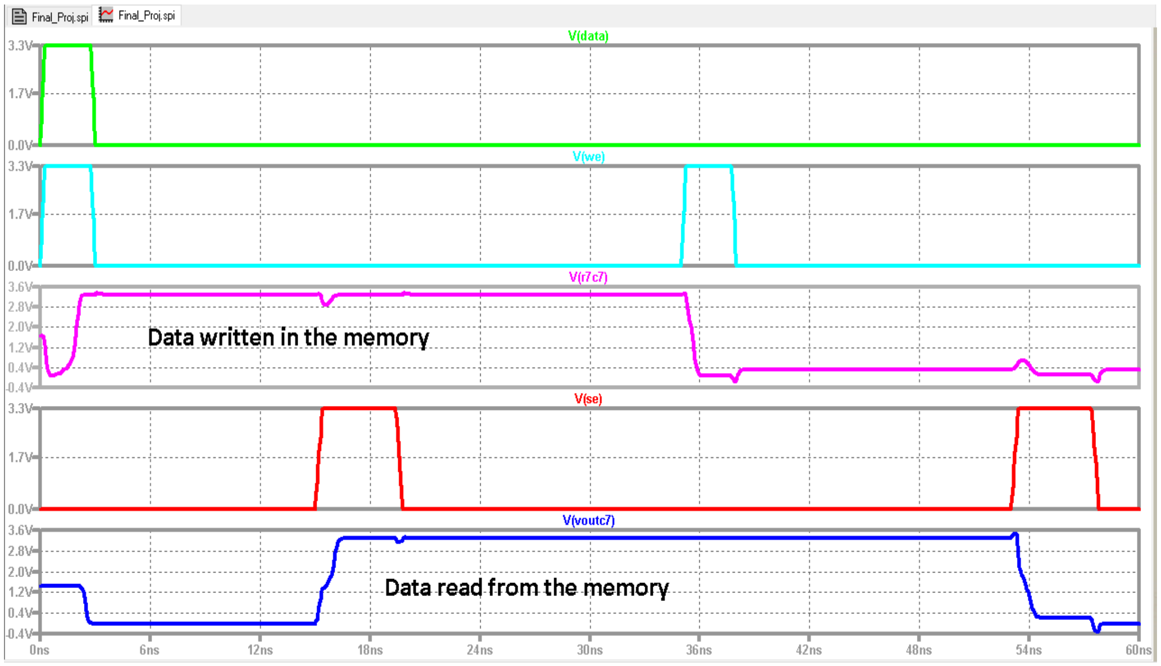


Figure 38: Read and write operation of Schematic

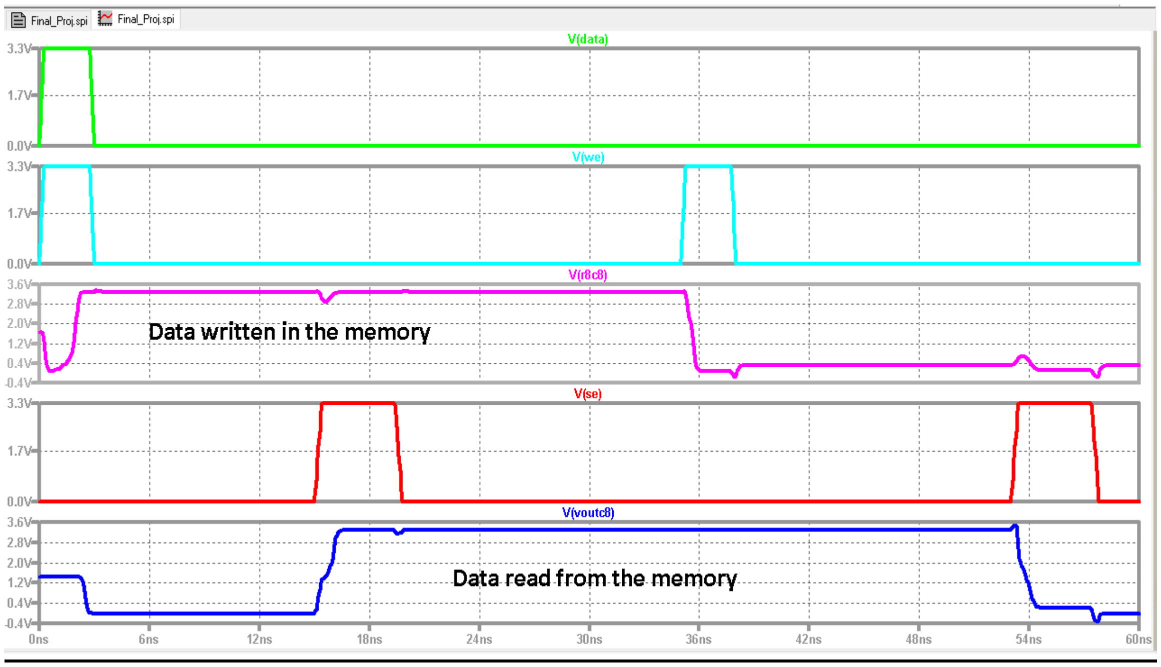


Figure 39: Read and write operation of Schematic

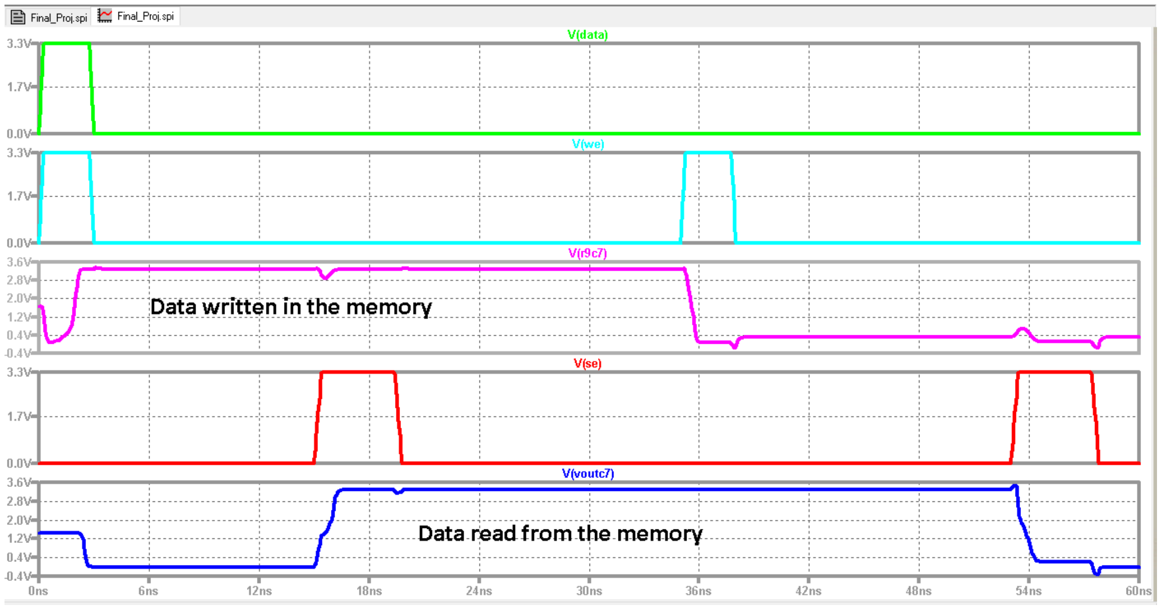


Figure 40: Read and write operation of Schematic

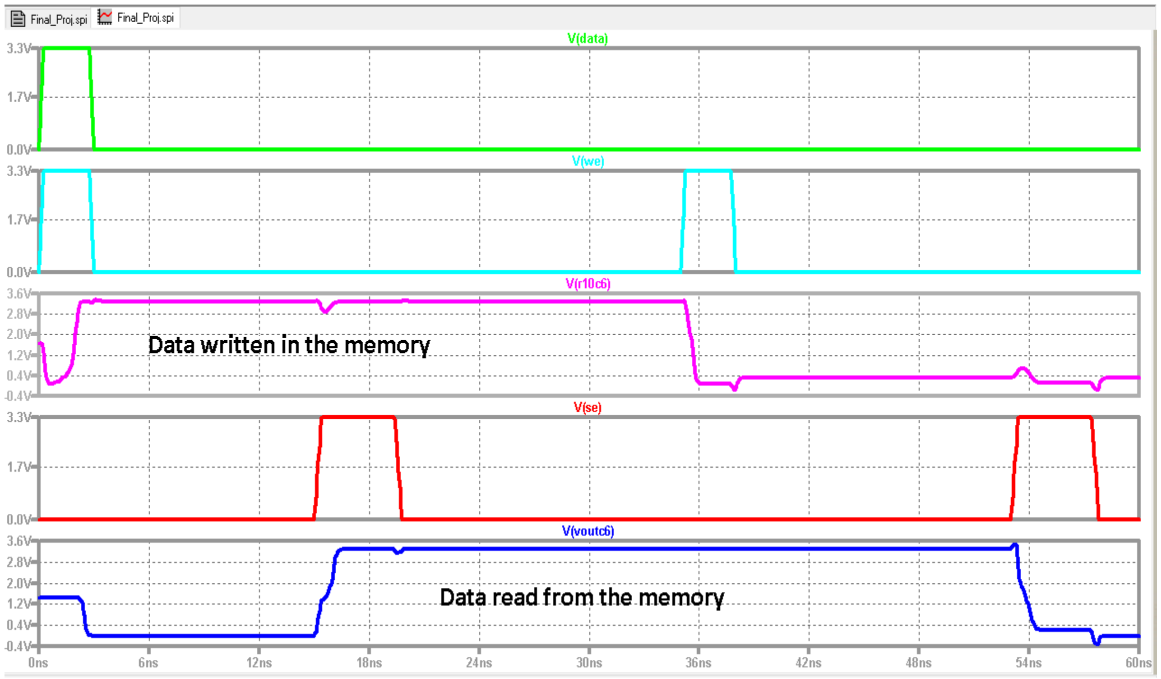


Figure 41: Read and write operation of Schematic

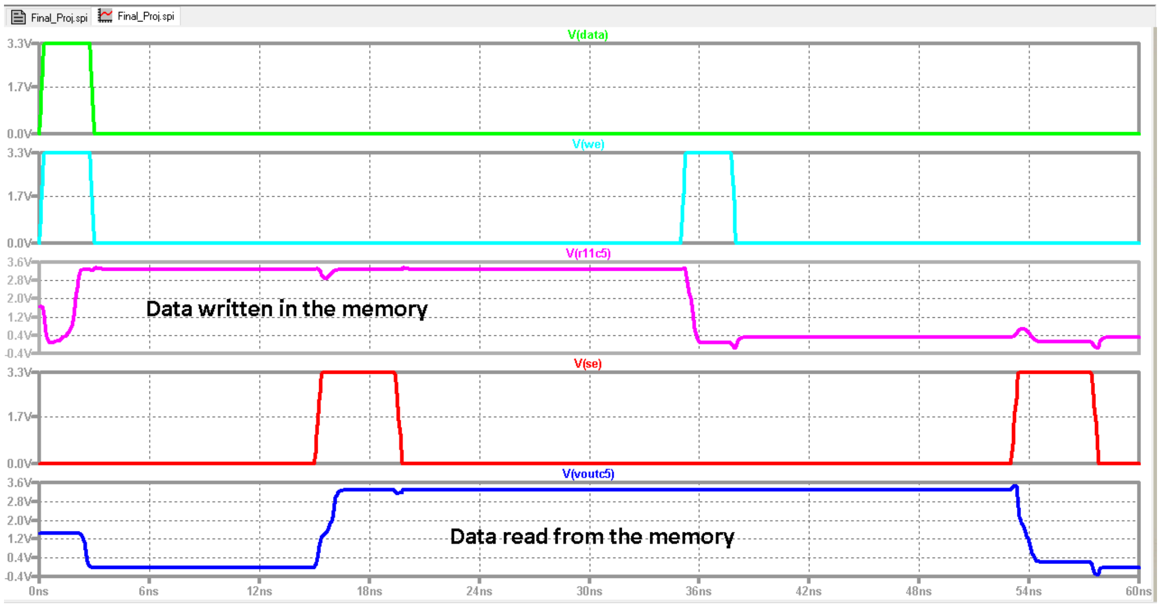


Figure 42: Read and write operation of Schematic

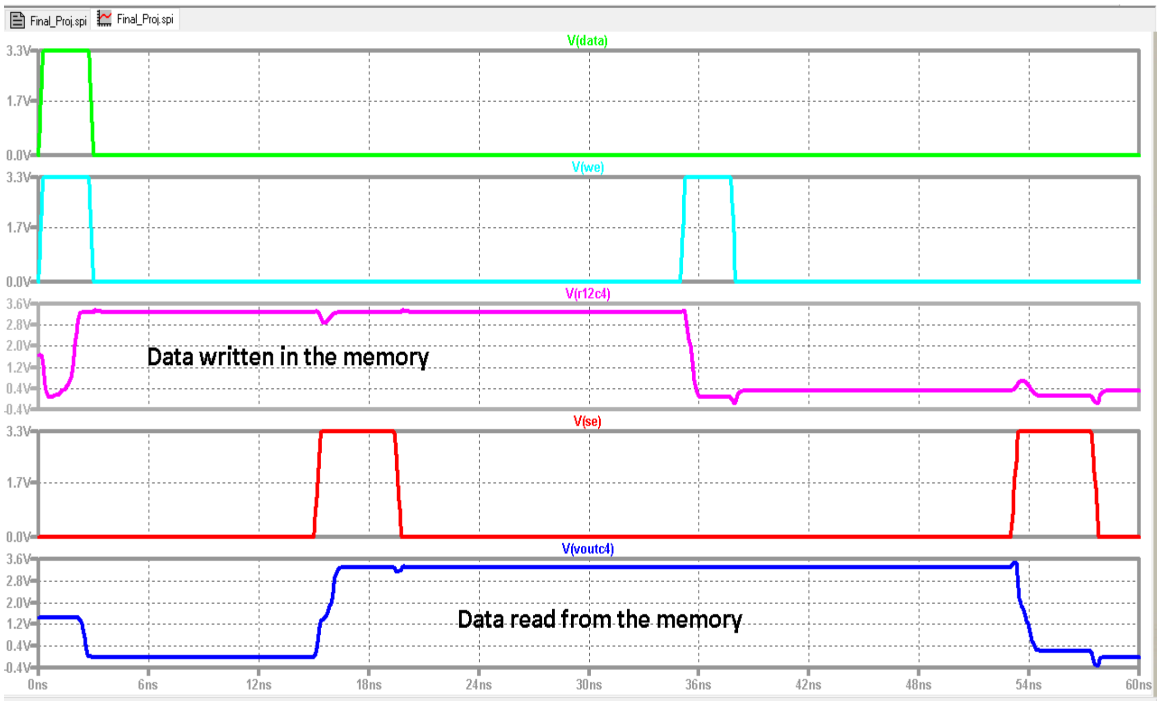


Figure 43: Read and write operation of Schematic

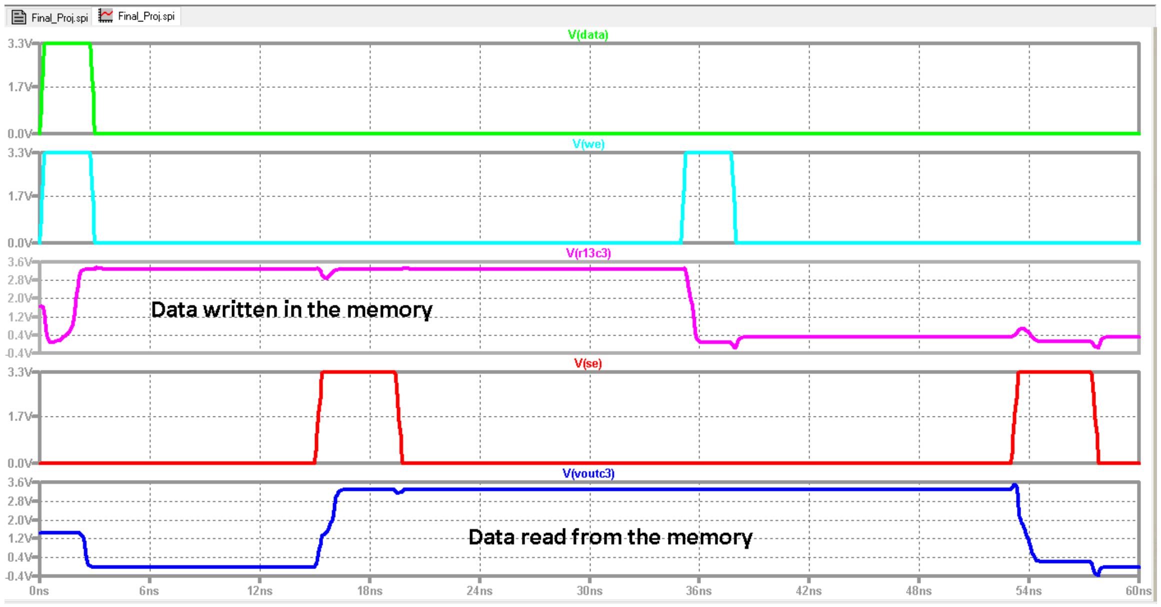


Figure 44: Read and write operation of Schematic

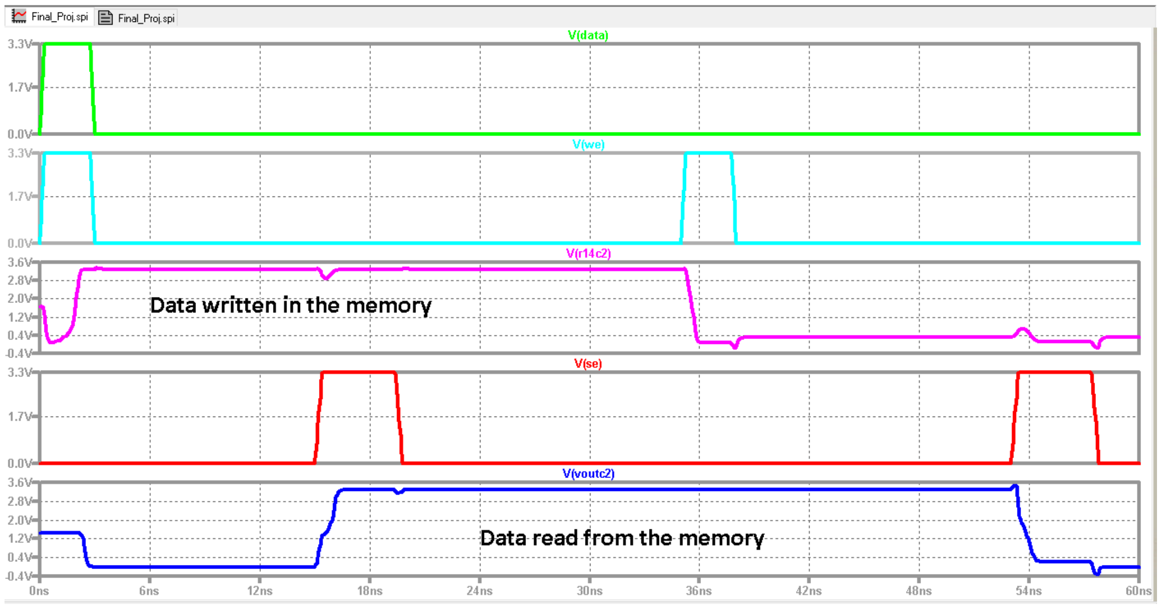


Figure 45: Read and write operation of Schematic

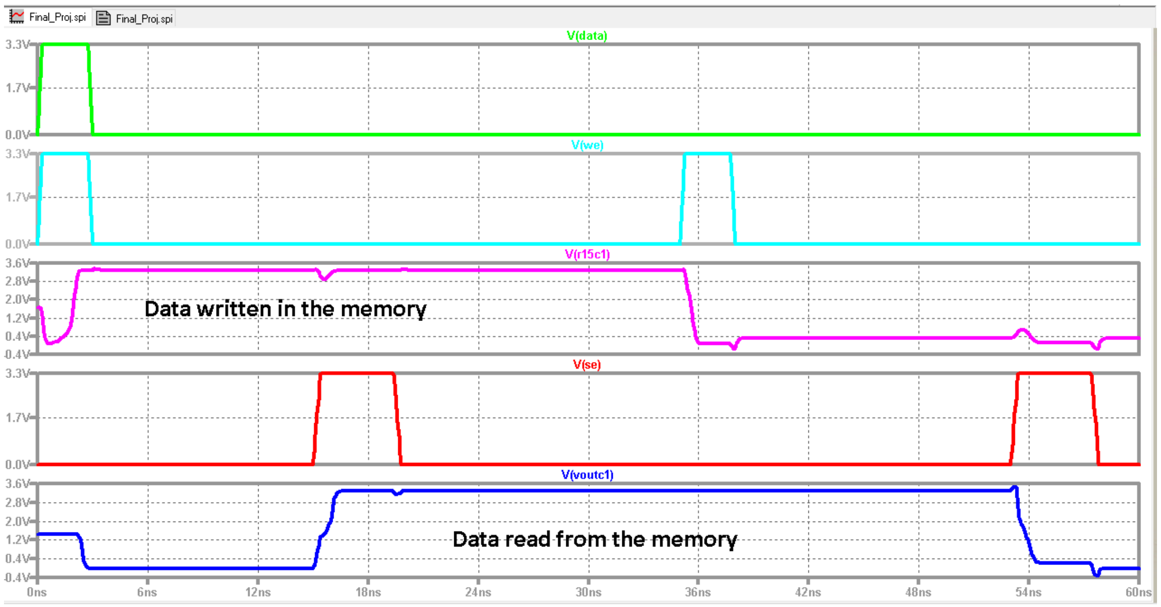


Figure 46: Read and write operation of Schematic

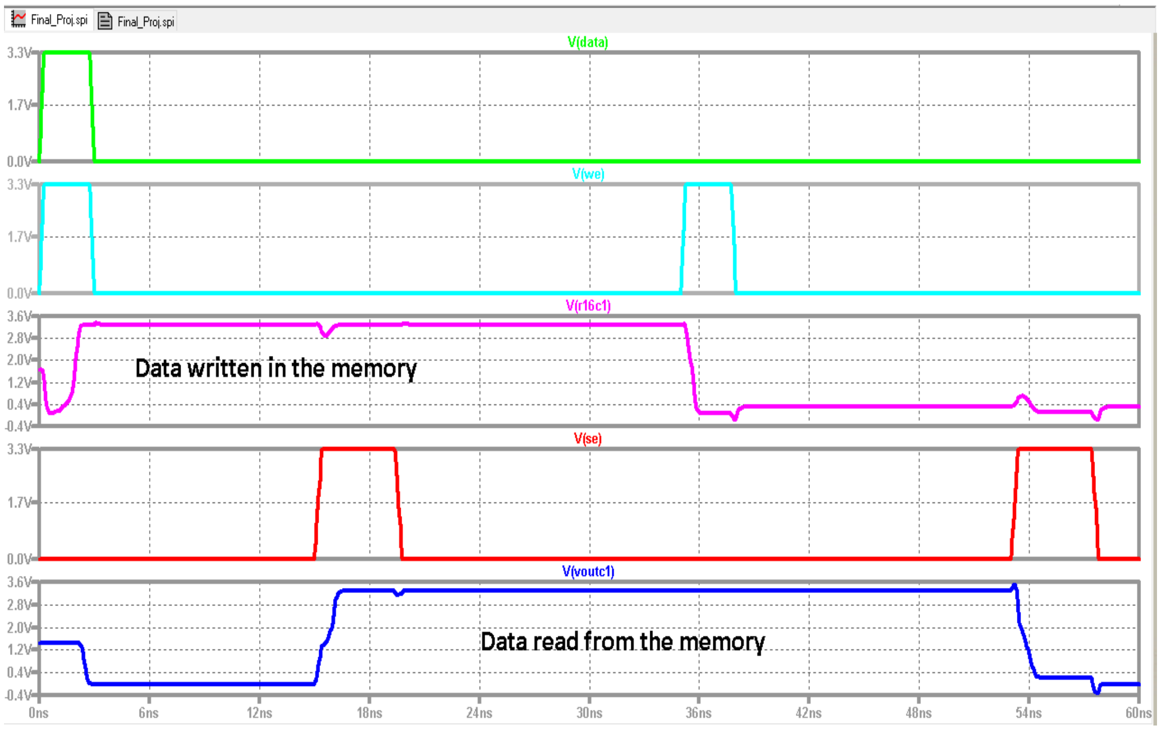


Figure 47: Read and write operation of Schematic

All the 16 figures above prove that all the rows are successfully activated and the data are successfully written and read from memory cells of schematic.

After simulating the schematic and using a lot of pictures, for the layout we decided to activate only first row and write data to it and read data from whatever is stored in that memory cell. Instead of activating 16 individual rows and 8 individual columns, and having 32 pictures, we decided to put the all the columns in the same graph. The write operation in the layout can be seen in figure 48 below.

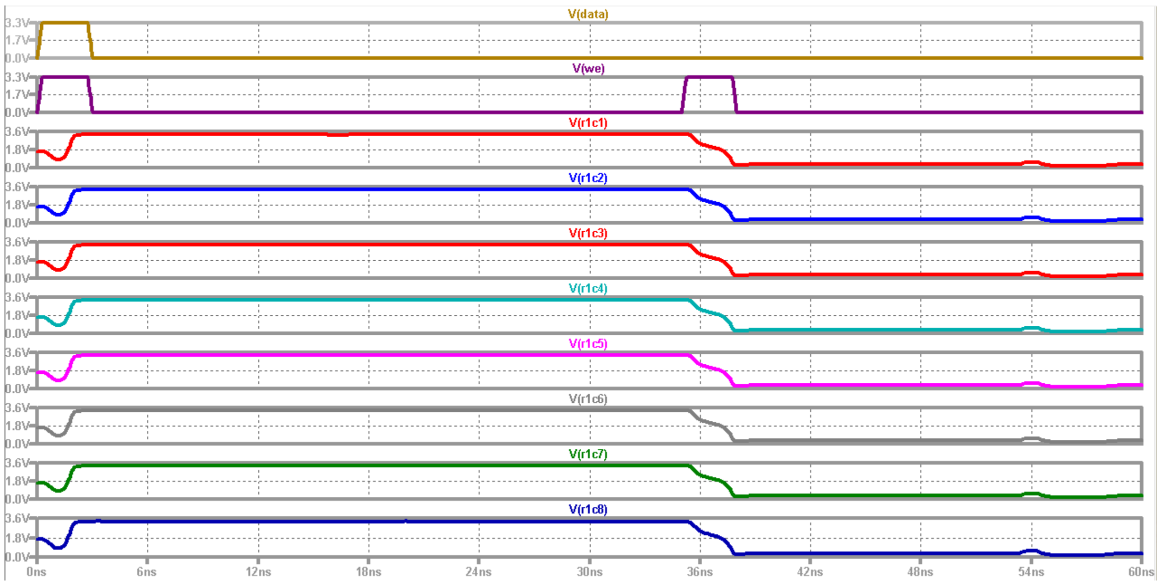


Figure 48: Write operation to all the 8 cells in the first row

Comparing figure 48 and figures from 32-47, the write operation of the schematic and layout looks similar. However, the starting point are different. For the schematic, we can see that the data starts to read logic 1 from around 1.25V and goes to logic high completely at 3ns. Whereas, for the schematic, the data starts to read logic 1 from around 1.7 V and goes to logic high at around 2.5ns. Another interesting thing to observe is that for the layout, the data written in the memory is 1 without any spikes, whereas for the schematic, we can see that when writing 1 into the memory, we see little spikes at around 15ns. Similarly, read operation's figure of the layout can be seen in figure 49 below.

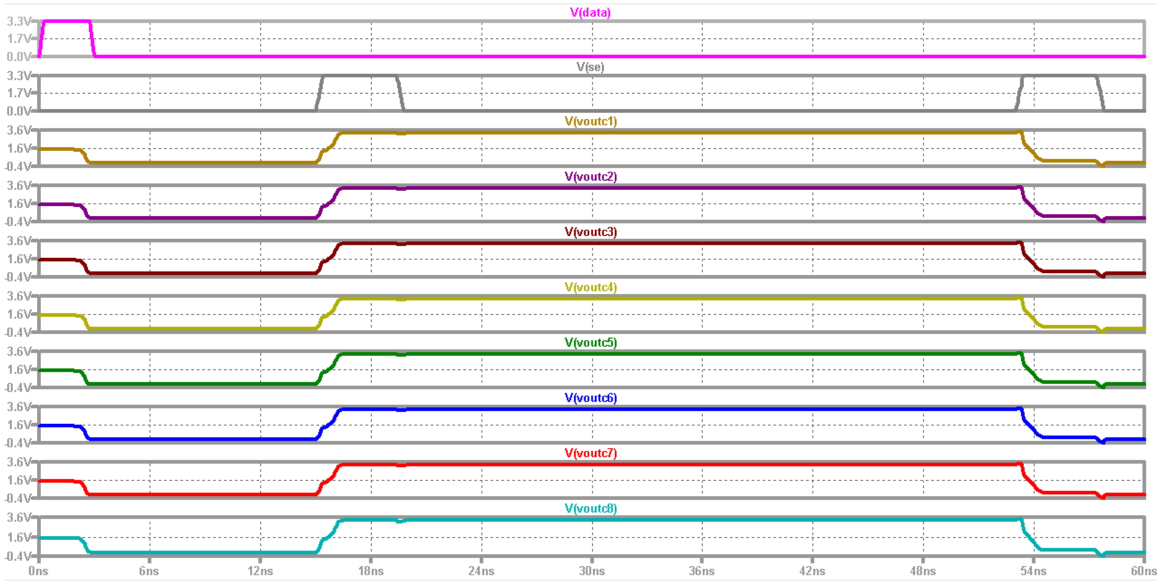
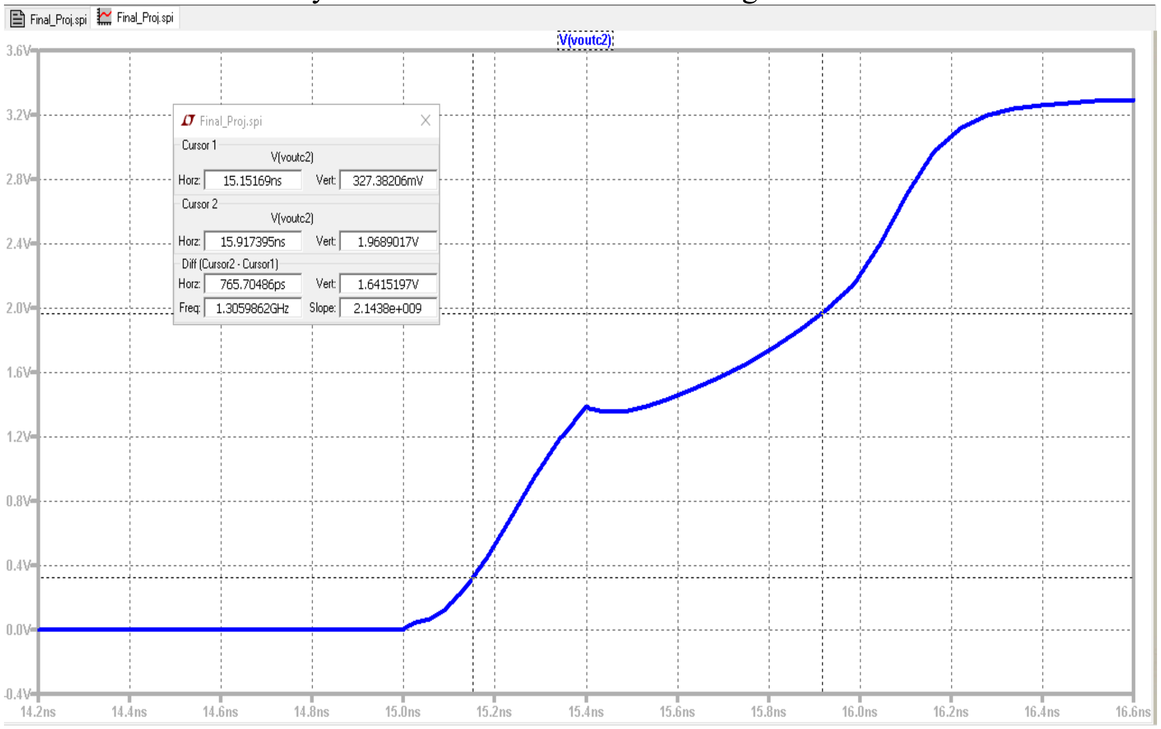


Figure 49: Read operation of layout of row 1 and 8 columns

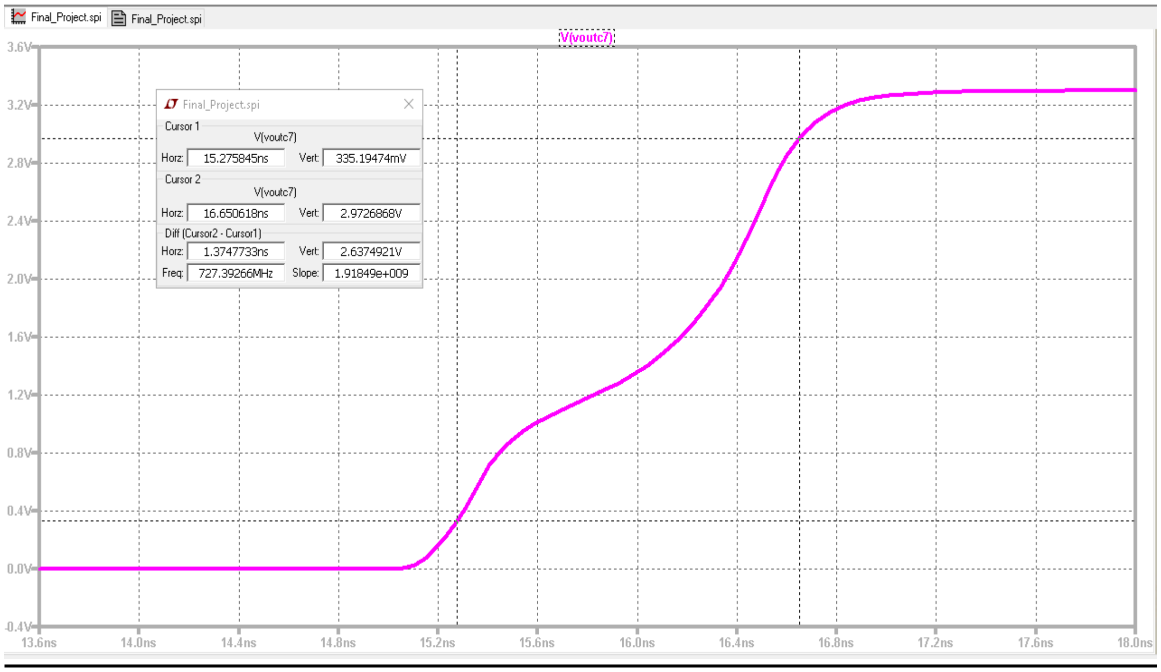
From figure 49, we can compare the read operation of layout with the schematic, which can be seen from figure 32-47. We can clearly see that the starting point of read operation for the layout is at around 1.6V, whereas for the schematic is at around 1.2V. The rise time can be clearly seen and we can conclude from the figure that the rise time of the layout's read operation is greater than that of schematics.

The rise time, fall time, and propagation delay are calculated through LTspice for both the schematic and the layout. The results can be seen in figures below.



```
Final_Proj.spi
Cursor 1
                V(voutc2)
Horz: 15.15169ns      Vert: 327.38206mV
Cursor 2
                V(voutc2)
Horz: 15.917395ns     Vert: 1.9689017V
Diff (Cursor2 - Cursor1)
Horz: 765.70486ps     Vert: 1.6415197V
Freq: 1.3059862GHz    Slope: 2.1438e+009
```

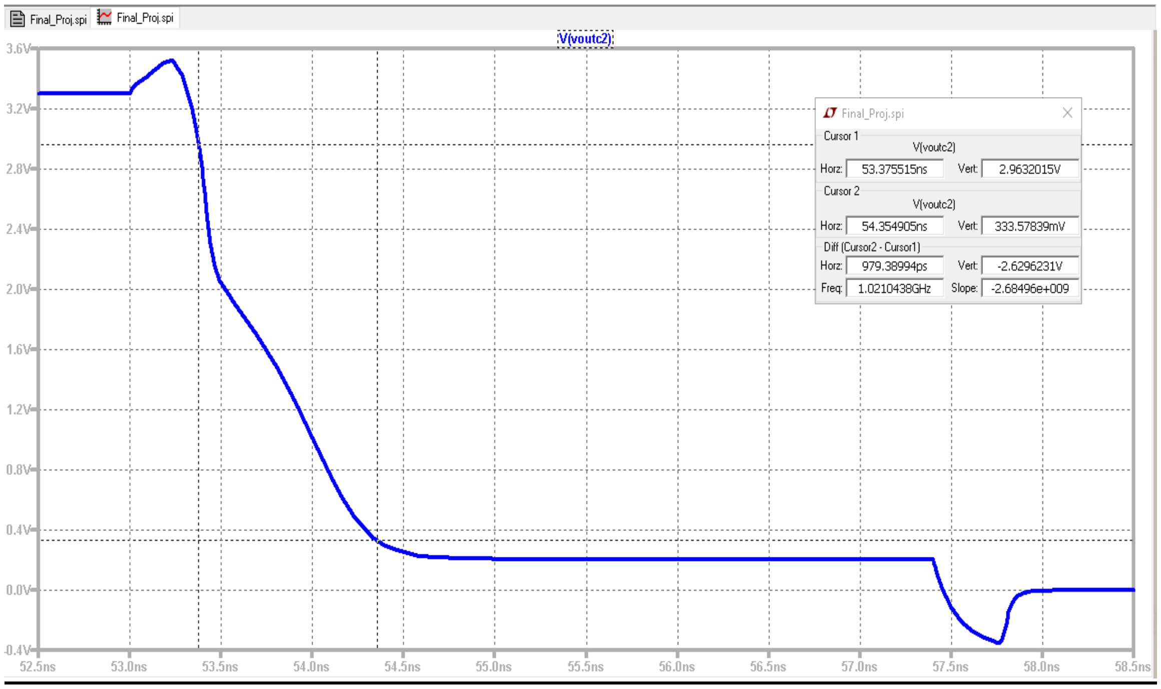
Figure 50: Rise time of read operation of schematic



```
Final_Project.spi
Cursor 1
                V(voutc7)
Horz: 15.275845ns     Vert: 335.19474mV
Cursor 2
                V(voutc7)
Horz: 16.650618ns     Vert: 2.9726868V
Diff (Cursor2 - Cursor1)
Horz: 1.3747733ns     Vert: 2.6374921V
Freq: 727.39266MHz    Slope: 1.91849e+009
```

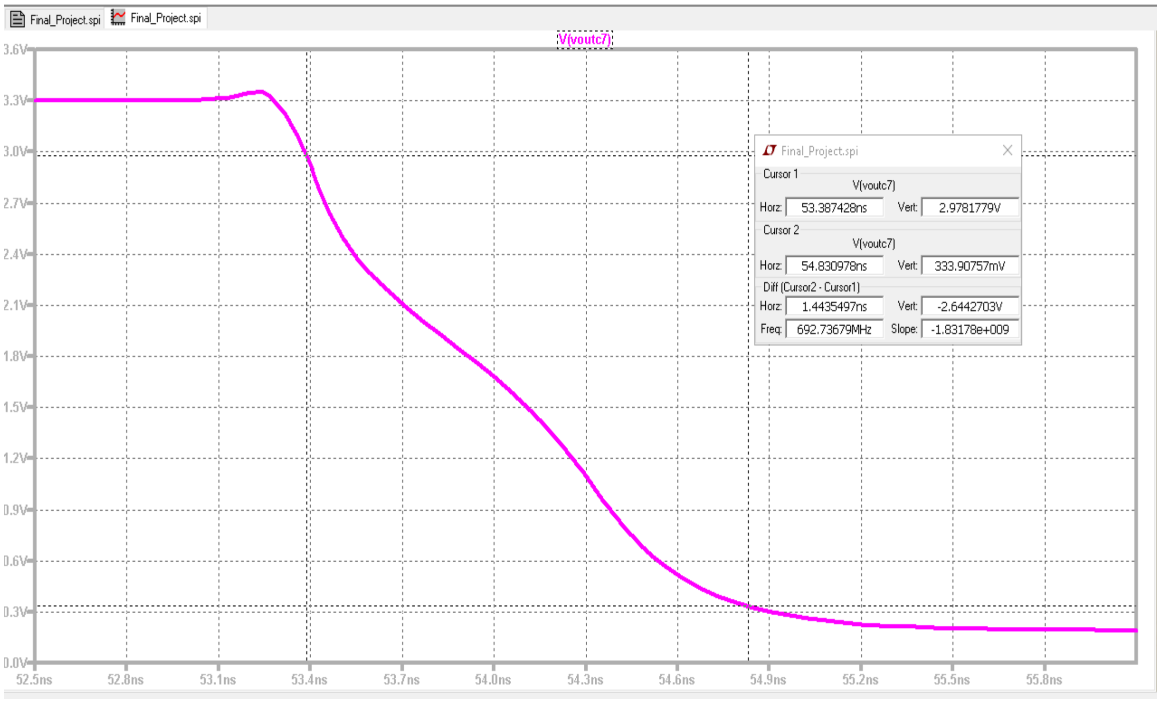
Figure 51: Rise time of read operation of Layout:

The rise time of the schematic's read operation is 765.91ps and the rise time of the layout's read operation is 1.374 ns.



```
Final_Proj.spi
Cursor 1
                V(voutc2)
Horz: 53.375515ns      Vert: 2.9632015V
Cursor 2
                V(voutc2)
Horz: 54.354905ns      Vert: 333.57839mV
Diff (Cursor2 - Cursor1)
Horz: 979.38994ps      Vert: -2.6296231V
Freq: 1.0210438GHz     Slope: -2.68496e+009
```

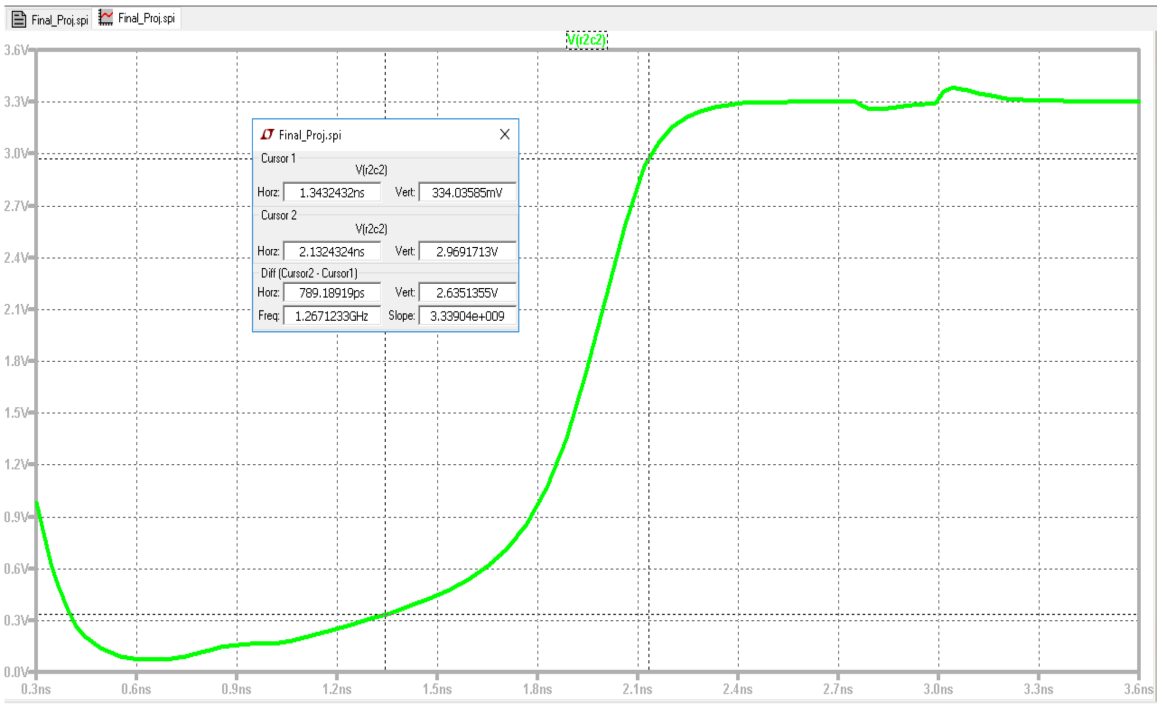
Figure 52: Fall time of read operation of schematic



```
Final_Project.spi
Cursor 1
                V(voutc7)
Horz: 53.387428ns      Vert: 2.9781779V
Cursor 2
                V(voutc7)
Horz: 54.830978ns      Vert: 333.90757mV
Diff (Cursor2 - Cursor1)
Horz: 1.4435497ns      Vert: -2.6442703V
Freq: 692.73679MHz     Slope: -1.83178e+009
```

Figure 53: Fall time of read operation of Layout

The fall time of the schematic's read operation is 979.38ps and the fall time of the layout's read operation is 1.443 ns.



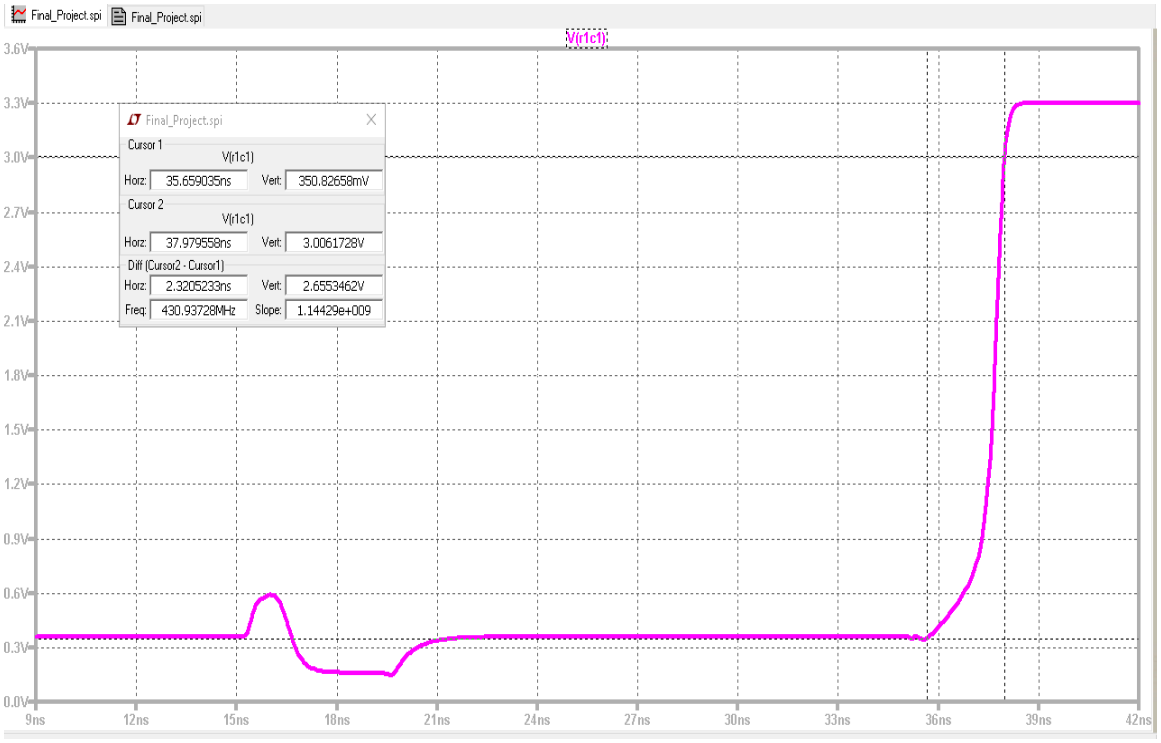
Final_Proj.spi

Cursor 1
V(r2c2)
Horz: 1.3432432ns Vert: 334.03585mV

Cursor 2
V(r2c2)
Horz: 2.1324324ns Vert: 2.9691713V

Diff (Cursor2 - Cursor1)
Horz: 789.18919ps Vert: 2.6351355V
Freq: 1.2671233GHz Slope: 3.33904e+009

Figure 54: Rise time of write operation of schematic



Final_Project.spi

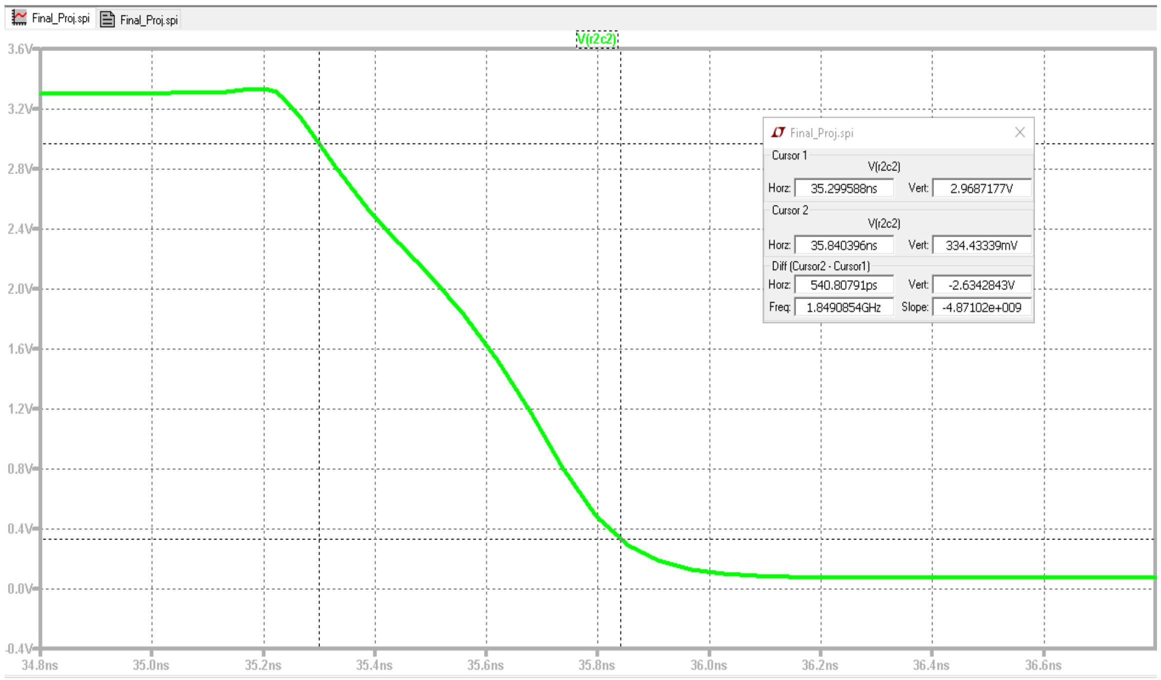
Cursor 1
V(r1c1)
Horz: 35.659035ns Vert: 350.82658mV

Cursor 2
V(r1c1)
Horz: 37.979558ns Vert: 3.0061728V

Diff (Cursor2 - Cursor1)
Horz: 2.3205233ns Vert: 2.6553462V
Freq: 430.93728MHz Slope: 1.14429e+009

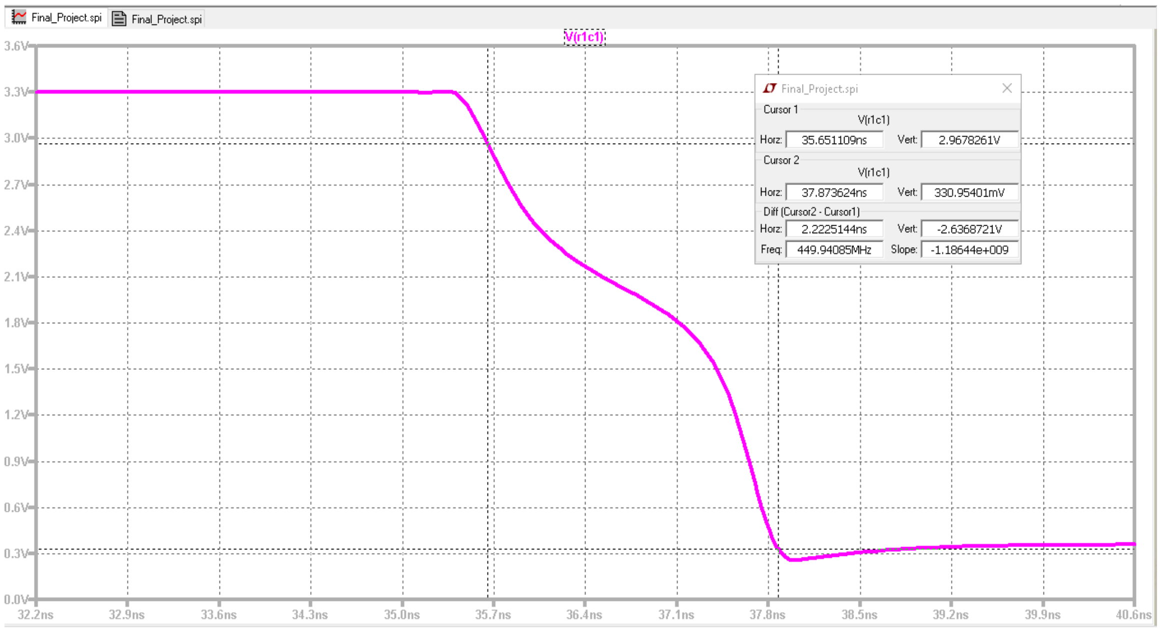
Figure 55: Rise time of write operation of Layout

The rise time of the schematic’s write operation is 789.18ps and the rise time of the layout’s write operation is 2.320 ns.



```
Final_Proj.spi
Cursor 1
                 V(r2c2)
Horz: 35.299588ns     Vert: 2.9687177V
Cursor 2
                 V(r2c2)
Horz: 35.840396ns     Vert: 334.43339mV
Diff (Cursor2 - Cursor1)
Horz: 540.80791ps     Vert: -2.6342843V
Freq: 1.8490854GHz    Slope: -4.87102e+009
```

Figure 56: Fall time of write operation of schematic

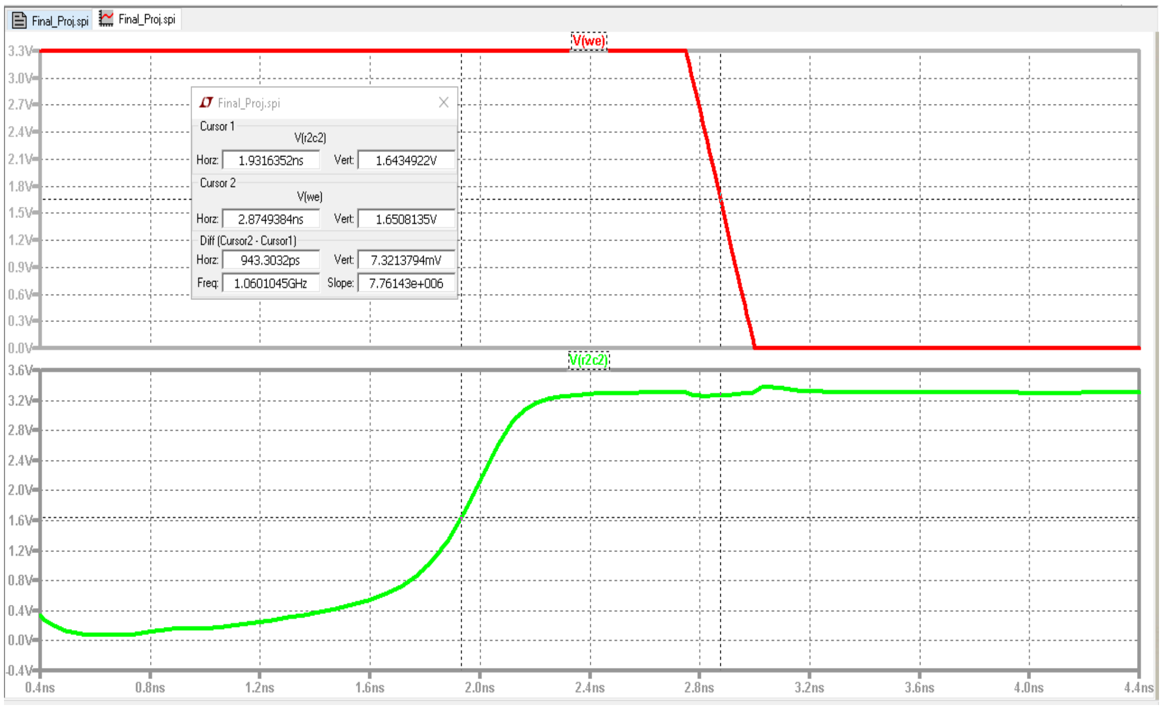


```
Final_Project.spi
Cursor 1
                 V(r1c1)
Horz: 35.651109ns     Vert: 2.9678261V
Cursor 2
                 V(r1c1)
Horz: 37.873624ns     Vert: 330.95401mV
Diff (Cursor2 - Cursor1)
Horz: 2.2225144ns     Vert: -2.6368721V
Freq: 449.94085MHz    Slope: -1.18644e+009
```

Figure 57: Fall time of write operation of schematic

The fall time of the schematic’s write operation is 540.807ps and the fall time of the layout’s write operation is 2.222 ns.

After finding the rise time and fall time for both schematic and layout, our next step was to find the propagation delay of both of them. In order to save pages and reduce the number of pictures, we only compared the propagation delay of the write operation of both schematic and layout. The figures can be seen below.



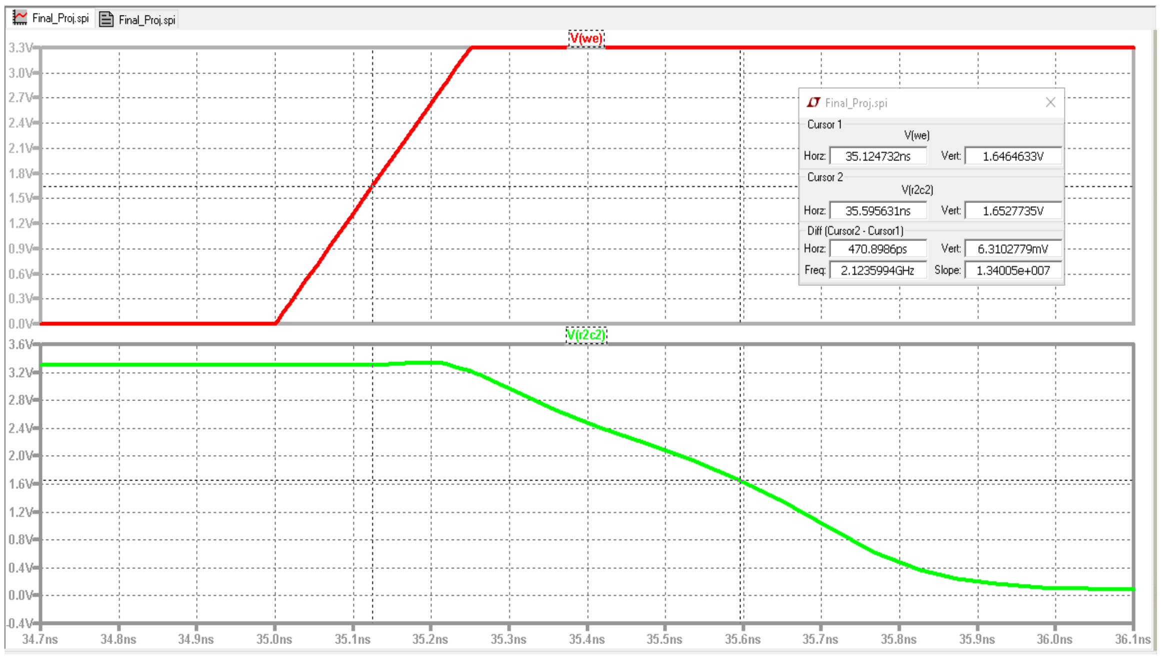
Final_Proj.spi

Cursor 1 V(r2c2)
Horz: 1.9316352ns Vert: 1.6434922V

Cursor 2 V(we)
Horz: 2.8749384ns Vert: 1.6508135V

Diff (Cursor2 - Cursor1)
Horz: 943.3032ps Vert: 7.3213794mV
Freq: 1.0601045GHz Slope: 7.76143e+006

Figure 58: Time from low to high(TPLH) of schematic



```
Final_Proj.spi
Cursor 1
                V(we)
Horz: 35.124732ns     Vert: 1.6464633V
Cursor 2
                V(r2c2)
Horz: 35.595631ns     Vert: 1.6527735V
Diff (Cursor2 - Cursor1)
Horz: 470.8986ps      Vert: 6.3102779mV
Freq: 2.1235994GHz    Slope: 1.34005e+007
```

Figure 59: Time from high to low (TPHL) schematic

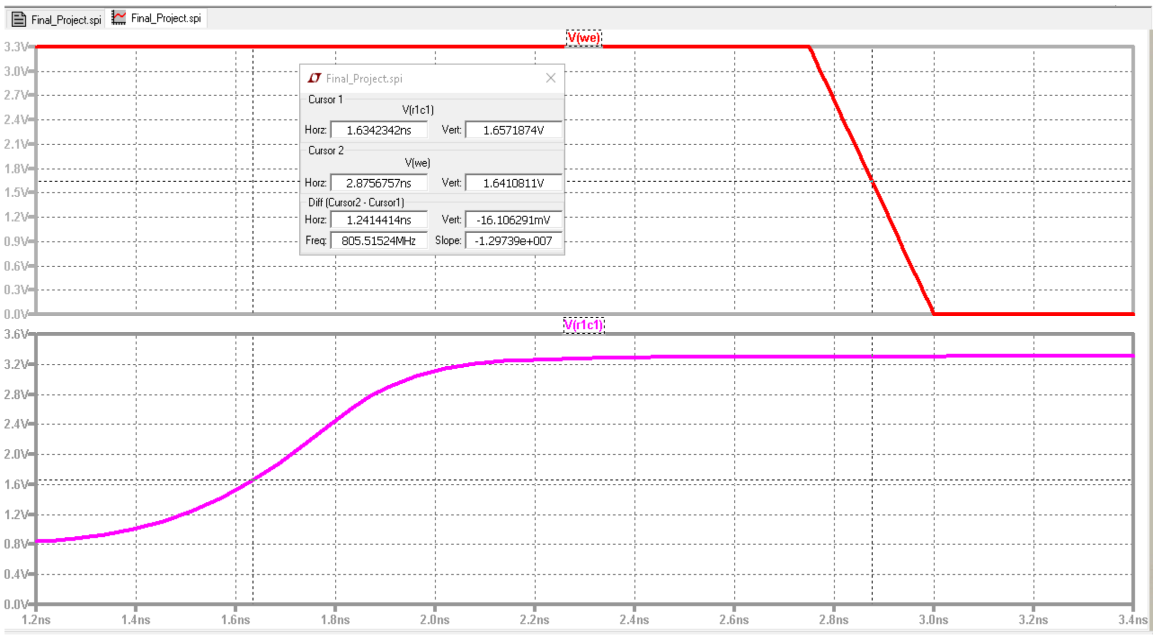
The formula to find the average propagation delay is given as $t_p = \frac{t_{PHL}+t_{PLH}}{2}$. and the result is:

$$t_p = \frac{t_{PHL}+t_{PLH}}{2}$$

$$t_p = \frac{943.30ps + 470.89ps}{2}$$

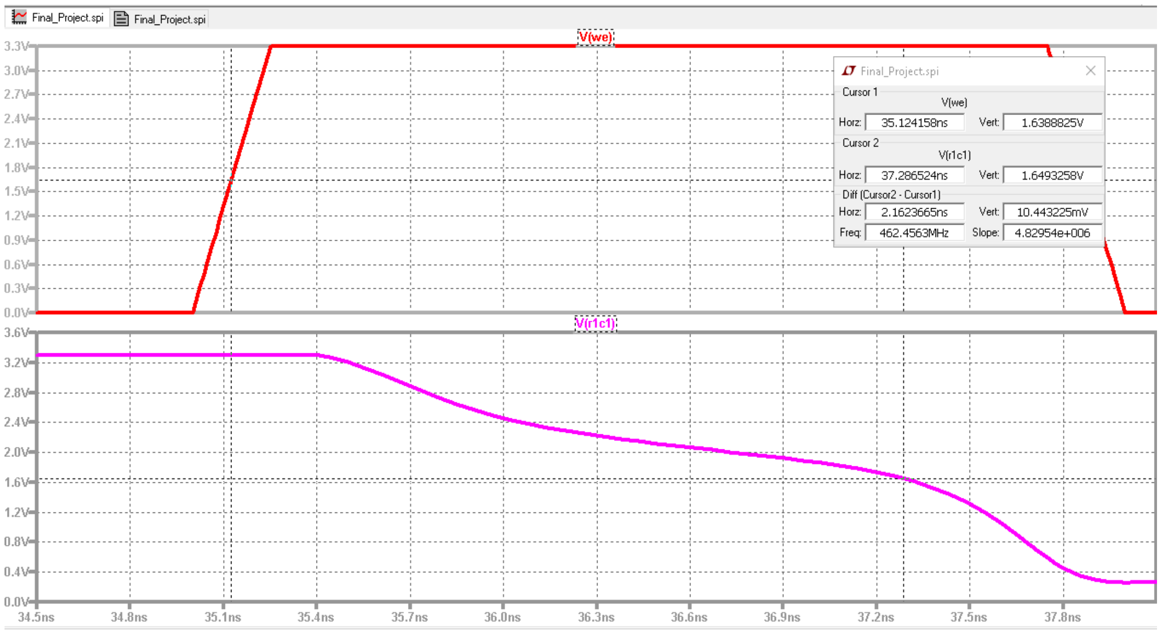
$$t_p = 707.095\ ps$$

Now, for the layout the figures are below in figure 60 & 61.



```
Final_Project.spi
Cursor 1
                V(r1c1)
Horz: 1.6342342ns     Vert: 1.6571874V
Cursor 2
                V(we)
Horz: 2.8756757ns     Vert: 1.6410811V
Diff (Cursor2 - Cursor1)
Horz: 1.2414414ns     Vert: -16.106291mV
Freq: 805.51524MHz    Slope: -1.29739e+007
```

Figure 60: Time from low to high(TPLH) of Layout



```
Final_Project.spi
Cursor 1
                V(we)
Horz: 35.124158ns      Vert: 1.6388825V
Cursor 2
                V(r1c1)
Horz: 37.286524ns      Vert: 1.6493258V
Diff (Cursor2 - Cursor1)
Horz: 2.1623665ns      Vert: 10.443225mV
Freq: 462.4563MHz      Slope: 4.82954e+006
```

Figure 61: Time from high to low (TPHL) layout.

$$t_p = \frac{t_{PHL} + t_{PLH}}{2}$$

$$t_p = \frac{2.16ps + 1.24ns}{2}$$

$$t_p = 2.7\ ns$$

Table 1: Comparison of rise time and fall time of schematic and layout

Operation	**Schematic**	**Layout**
Read	Rise Time: 765.91 ps Fall Time: 979.38 ps	Rise Time: 1.374ns Fall Time: 1.442 ns
Write	Rise Time: 789.18 ps Fall Time: 540.807 ps	Rise Time: 2.320 ns Fall Time: 2.222 ns
Delay	707.095 ps	1.7ns

VIII. Measurements of power, delay, chip area, timing, number of transistors for the layout.

Below is the graph for the power dissipation.

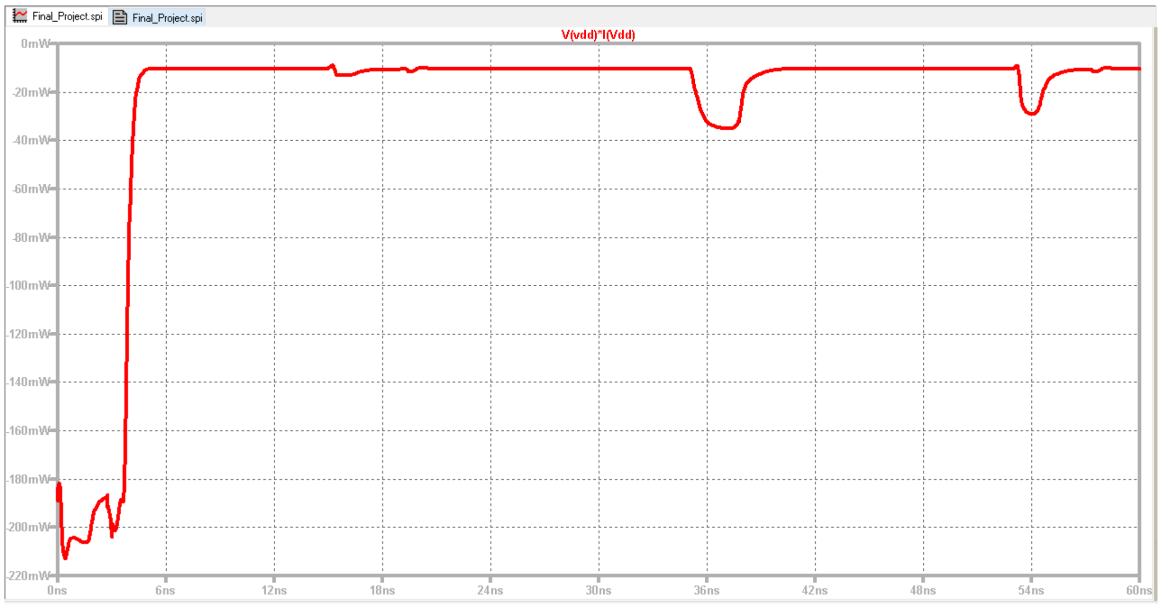
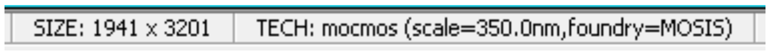


Figure 62. Power dissipation of Layout

From Figure 62. We can see that the power dissipation is 15mW. The layout of SRAM is dissipating the power of 15mW.

It can be seen from the table.1 that the delay is 707.095ps for schematic and 1.7ns for layout. It can be concluded that the delay for layout is higher that the delay of schematic. Furthermore, the Rise time and fall time for read and write operation of both schematic and layout are shown in Table.1 as well. From the data, it is concluded that the rise and fall time is higher in layout in compared to the schematic design.

The chip area of the layout is calculated as follows:



SIZE: 1941 x 3201 | TECH: mocmos (scale=350.0nm,foundry=MOSIS)

Figure 63. Area of the layout design.

We have used the lambda value of 350nm. That's why the total chip area is

$=1941*3201*350*350 \text{ nm}^2$

$=761{,}109.772 \ \mu\text{m}^2$

The number of transistors used by the layout are shown below:

Table 2: Number and size of transistors

Circuit	**Number**	**Transitor(W/L) Schematic**	**Transitor(W/L) Layout**
SRAM Cell	768	PMOS:4/2 NMOS:2/2	PMOS:20/2 NMOS:10/2
Row Decoder	192	PMOS:4/2 NMOS:2/2	PMOS:20/2 NMOS:10/2

Pre-Charge Circuit	16	PMOS:2/2	PMOS:10/2
Data Driver Circuit	64	PMOS:8/4 NMOS:4/4	PMOS:20/2 NMOS:10/2
Sense Amplifier	48	PMOS:4/2 NMOS:2/2	PMOS:20/2 NMOS:10/2
Total Number of transistors	1088		

IX. Conclusion and References

This project reflected on all the previous projects we did for this class. At the end, everything added up to what we learned from this class which made this project not as hard. Designing the SRAM was not hard however, learning how SRAM functions and the peripheral circuits and what role do they play in SRAM was the hardest and most time-consuming part of this project. Once we knew about the peripheral circuits and how SRAM operates, then design was done correctly in Electric VLSI design system with no errors. This project taught us how to design a SRAM, how it operates, and where is it used. We never thought that just by using cross coupled inverters we can design a memory cell. Overall, this project helped me to realize that sizing of the component matters and the right design of peripheral circuits can determine how good your overall chip works.

References:

[1] R. K. Shah, I. Hussain, M. Kumar, “Performance Comparison for Different Configurations of SRAM Cells,” International Journal of Innovative Research in Science, Engineering and Technology, vol.4, pp.18543-18545, Jan. 2015.

[2] Neil H. E. Weste, David Harris, Ayan Bannerjee (2009), CMOS VLSI Design –A Circuits and Systems Perspective, Pearson Education, 3rdEdition.

[3] D. Clein. CMOS IC Layout: Concepts, Methodologies, and Tools.

[4] Serge Arnaud Viraazel Luigi Dilillo, Patrick Girard. “Analysis and test of resistive-open defects in sram pre-charge circuits” JETTA special issue, 2015

[5] Sajith Ahamed Palatham-Veedu “Design and analysis of sense amplifier circuits used in high-performance and low-power Srams”