Final Report

A High Speed 4-bit Perceptron with 256-bit SRAM

Table of Contents

1. Introduction

This report is over the design of a critical component of an integrated circuit for building infrastructure for a “classroom of the future.” The component is a perceptron, the fundamental element of the neural network to help transcribe the classroom scene. The transcription will be able to recognize the instructor's handwriting, gestures, and student reactions. The perceptron requires smaller components including SRAM, a multiplier, and an adder that will be outlined in detail in the report. Specifically, the SRAM block for storing the learned weights of the neural network will be discussed in Section 2 of this report, while the classifier computational blocks will be discussed in Section 3.

The input signal to the perceptron will be 8 bits. It will be modified by synaptic weights which are powers of two ranging from 2^{-4} to 2^{0}. The output will be 9 bits. A high-level diagram of the system is shown in Figure 1.

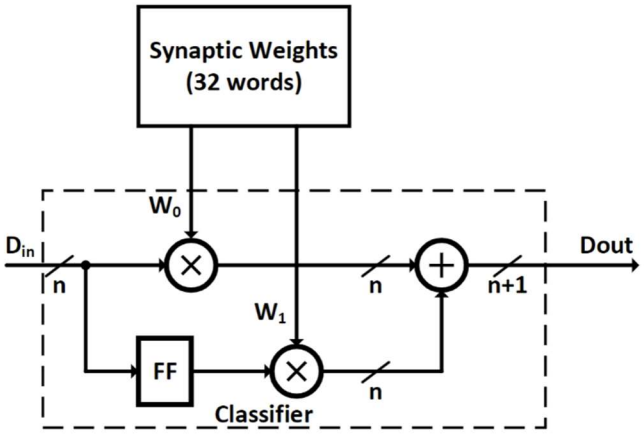

Figure 1: System Diagram

The company also specified that the clock frequency must be able to be at least 1 GHz. This means the total propagation delay of the critical path of the perceptron must be less than 1 nanosecond. Section 4 will discuss this requirement. Section 5 will discuss the power consumption of the solution. It should be noted that the solution will make use of a 1V supply and GDPK45 technology (a 45 nm technology).

Lastly, Section 6 will conclude the findings and developments, and section 7 is the acknowledgments.

2. Synaptic Weights

For weight storage, a 32-word SRAM array where each word is 1 byte was Implemented. The SRAM consists of 5 major subparts. They are a 32-word memory array, a 4 to 16-row decoder, a 1-2 column decoder, and 2 drivers for data write and read driving. The detailed structure of the SRAM system is shown in Figure 2 below.

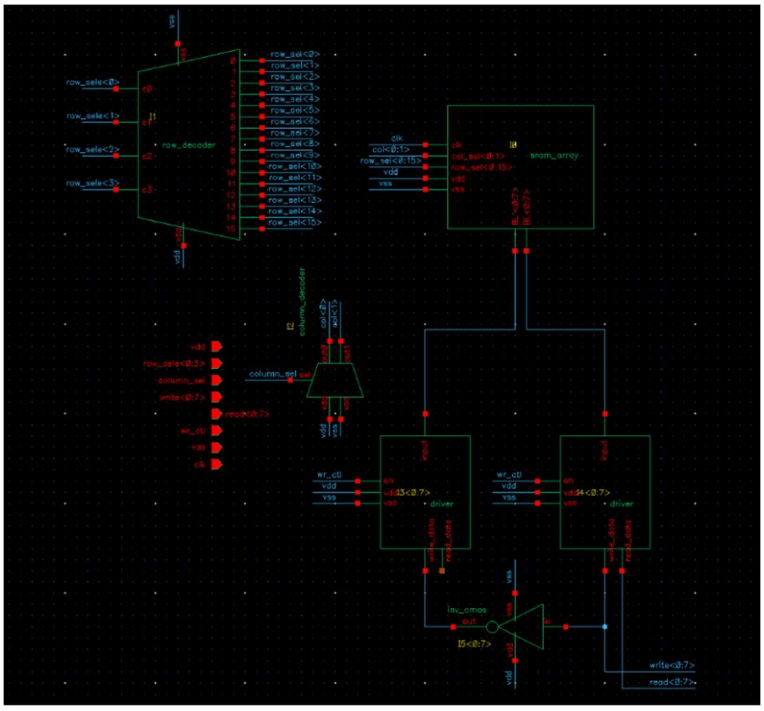

Figure 2: Detailed Structure of SRAM

2.1 SRAM Cell

As shown in Figure 3, a 6-transistor SRAM cell structure was applied in this project.

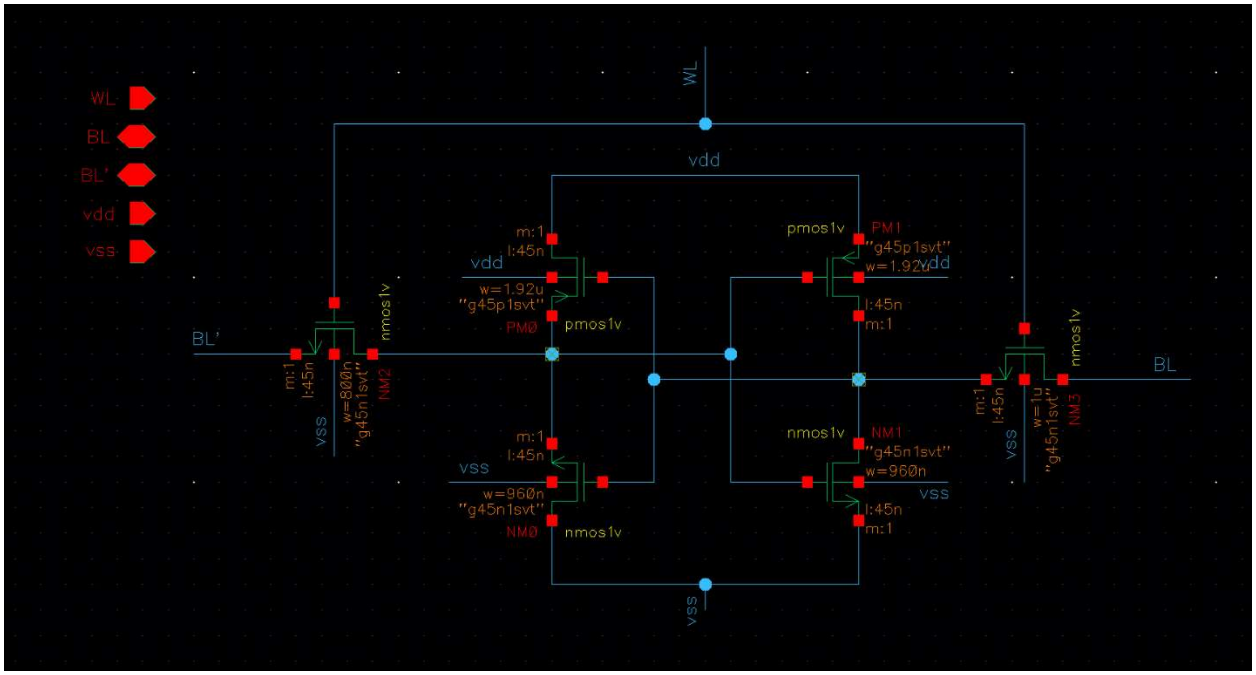

Figure 3: 6-transistor SRAM cell

2.2 SRAM Array

The schematic of the SRAM array is shown below in Figure 4. It is divided into 2 columns and 16 rows. As a result, 32 cells appear in the schematic. Each cell shown in the schematic indicates 8-bits by using bus notation. Therefore, in total, there are 256 bits, which is 32 words. For each cell, there is a WL pin used to control if the cell is activated. When $WL = 1$, the cell is activated. The write or read operations can proceed on BL and $\overline{BL}$. The WL is controlled by combinational logic WL=*row_sel* AND *col_sel* AND *clk*. This makes sure on clk high, only one cell unit (8 bits), detail shown in Figure 4 is selected. Also, there are 32 PMOS used as switches for Vdd connection. When *clk* is low, the PMOS is on, and all bit line buses (all BL and $\overline{BL}$) are pre-charged to high.

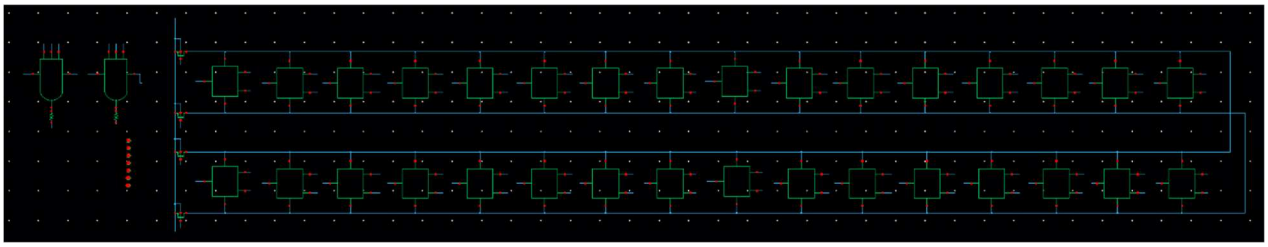

Figure 4: Schematic of SRAM Array

2.3 Driver and Sense Amplifier

The driver was created to enable both reading and writing functionality. When writing to the SRAM cell, the driver on writing direction needs to be strong enough to overwrite the previous value latched inside the SRAM cell. So, buffers are added. Besides, two transmission gates in opposite directions and opposition clock triggering are placed to make sure during reading, the writing circuit does not interfere with the latched signal on the bit line, and vice versa for writing. The schematic is shown in Figure 5.

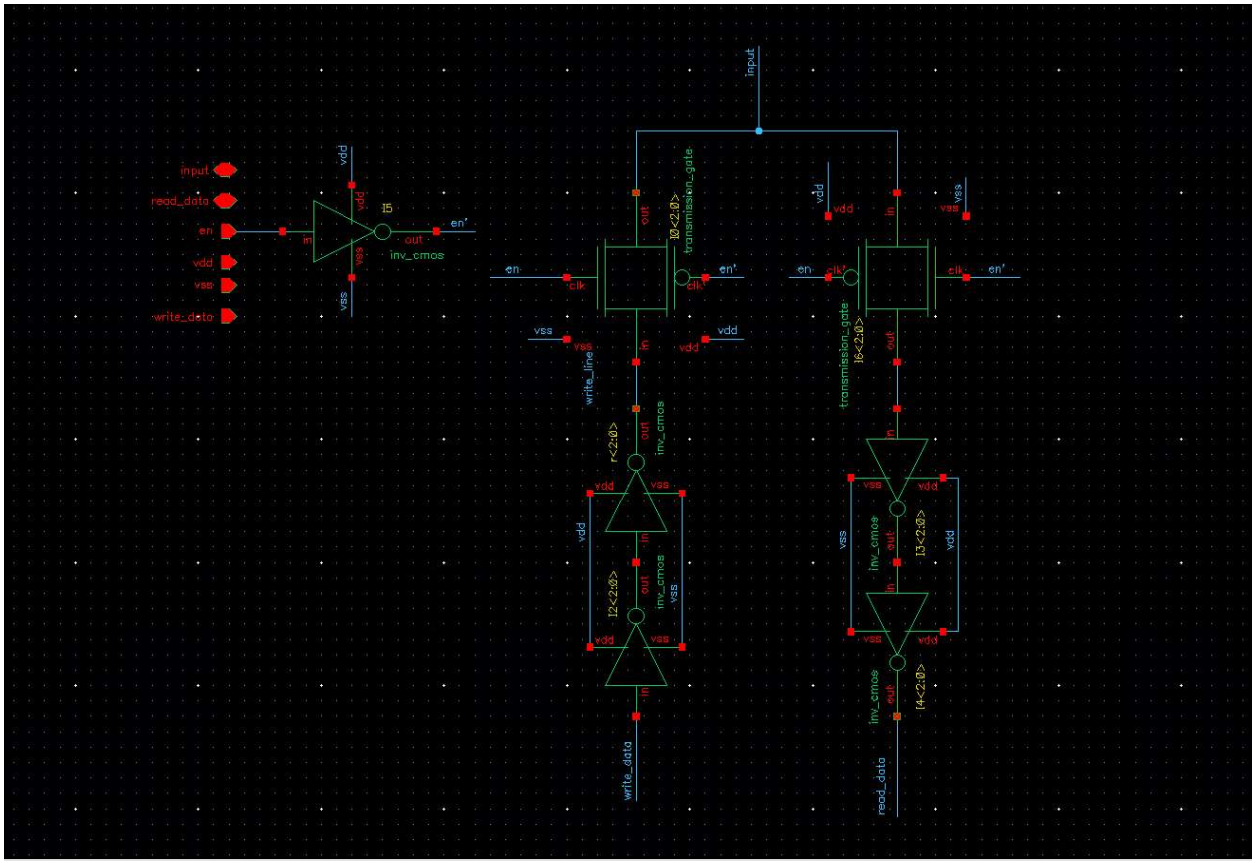

Figure 5: Schematic of Driver and Sense Amplifier

2.4 Row Decoder

Row decoder is a 4 to 16 decoder. The schematic is shown in Figure 6. It is used to generate a row selection signal. Only one bit of output is asserted at a time.

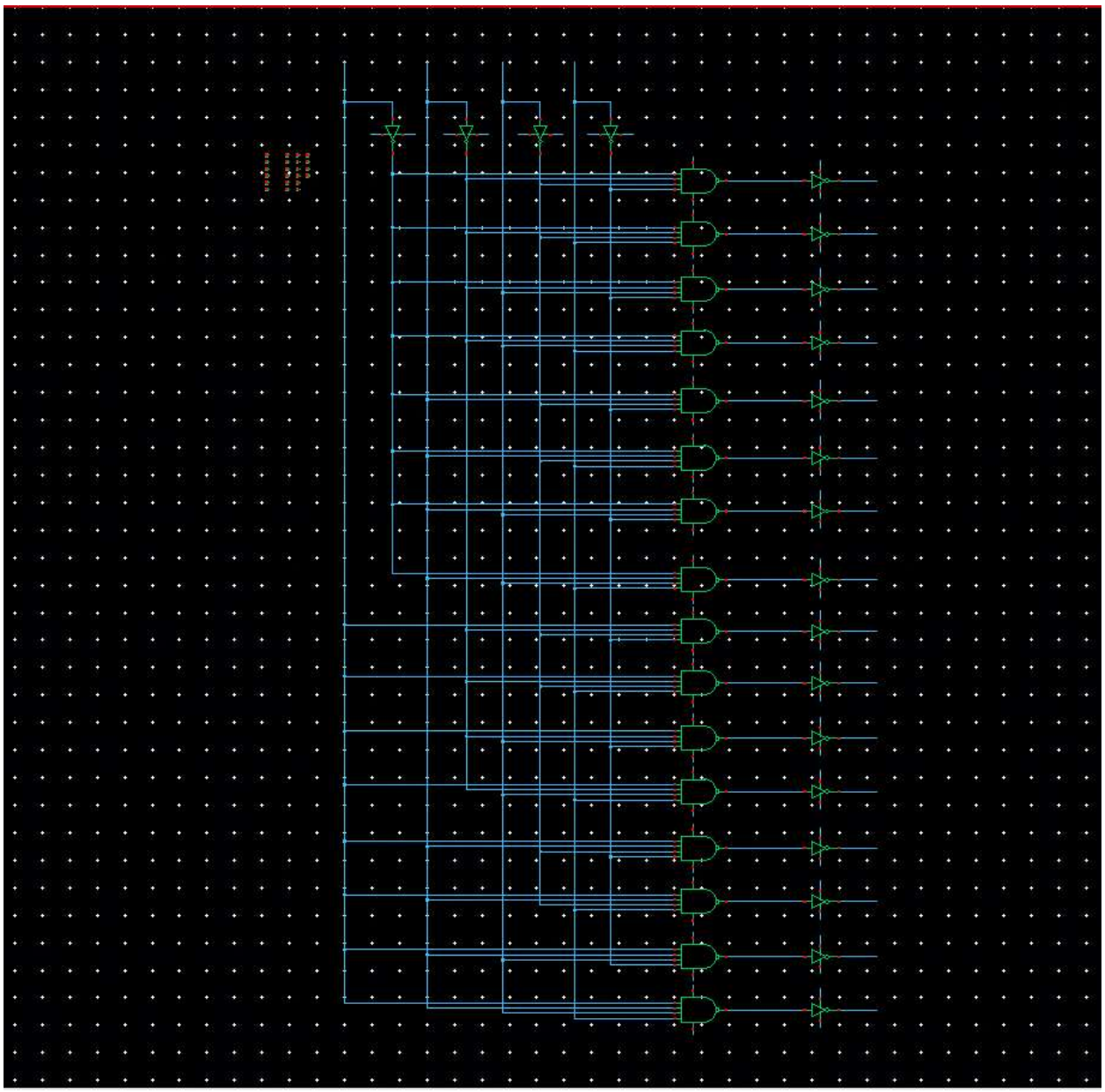

Figure 6: Schematic of Row Decoder

2.5 Column Decoder

A column decoder is a 1 to 2 decoder. The schematic is shown in Figure 7. It is used to generate a column selection signal. Only one-bit output is asserted at a time.

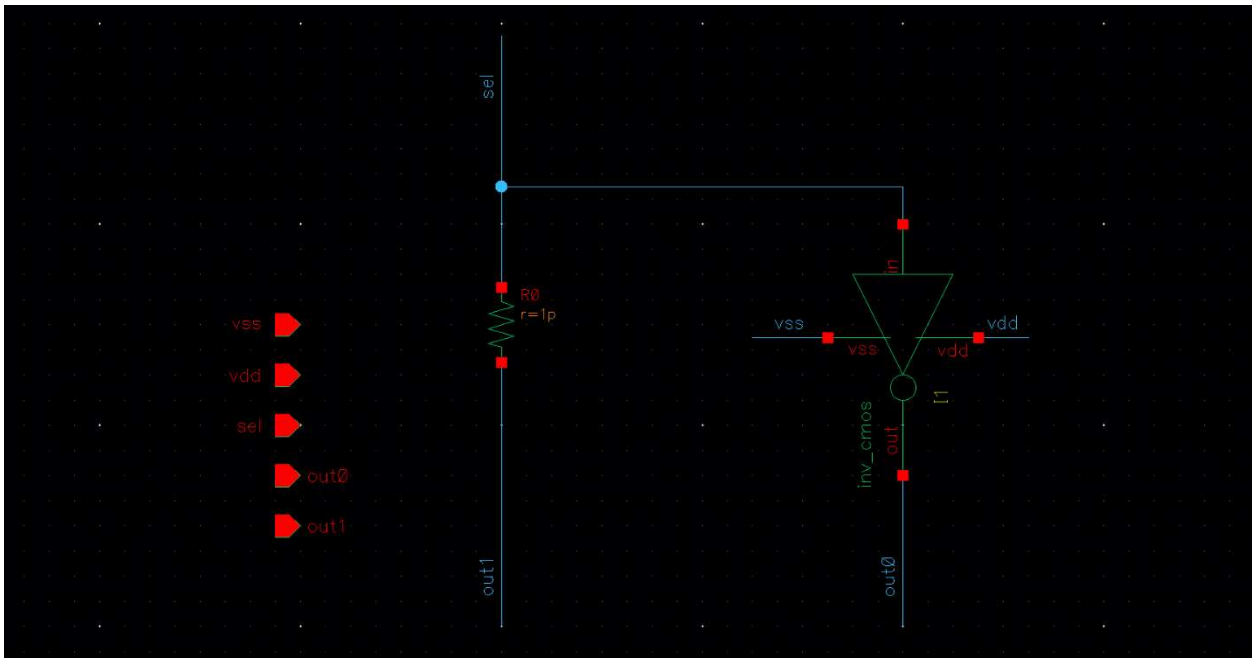

Figure 7: Schematic of Decoder

2.6 SRAM System Verification

The schematic of the testbench of the SRAM system is shown in Figure 8.

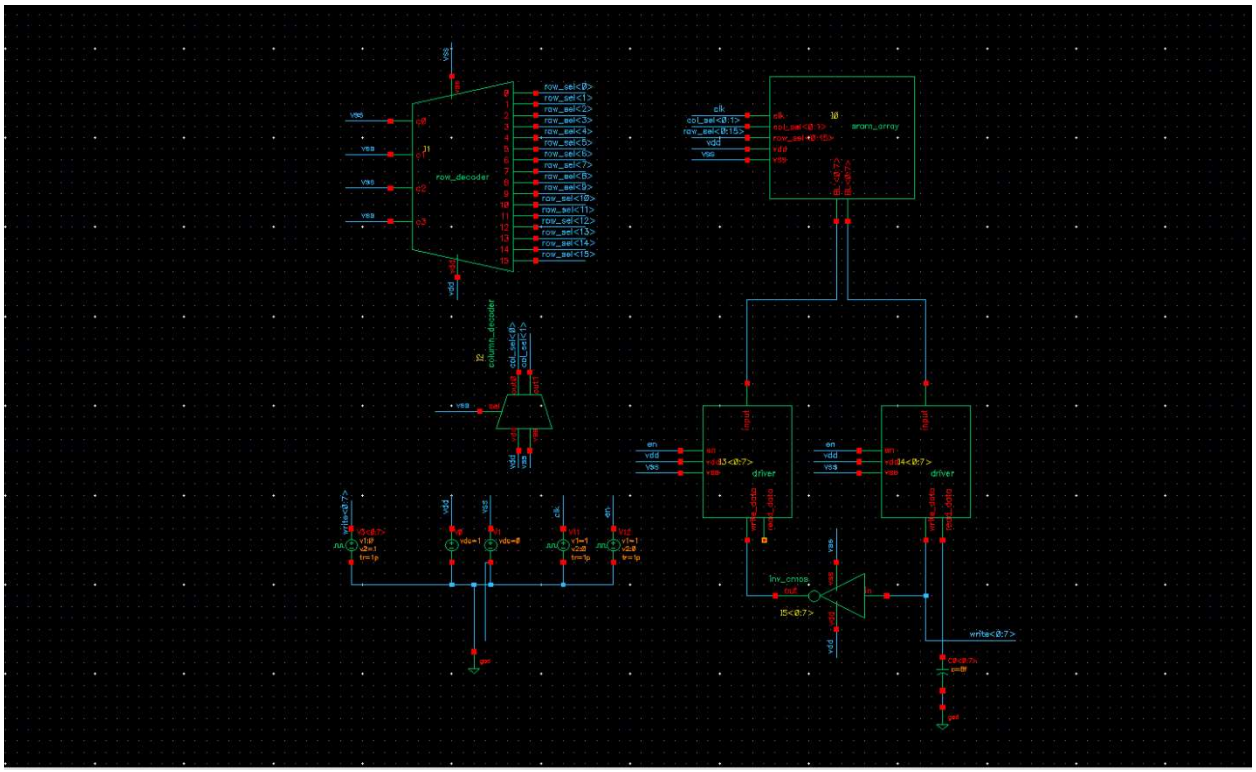

Figure 8: SRAM Testbench

The simulation result is shown in Figure 9 and Figure 10. When *clk*=1 and *en*=1, the write data is written to the selected SRAM cell. When *clk*=1 and *en*=0, the read data is read from the selected SRAM cell. When *clk* = 0, the SRAM is inactivated, no operation is done. From the simulation result, the read operation delay is (59.04ps+60.11ps) / 2 = 59.575ps.

In this test bench, the 8-bit parallel-connected cell at row = 0 and column = 0 is activated. /net11<0> indicates the read result. Write<0> indicates external write signal. /I010<0>/net7 indicates the node inside the SRAM cell connected to bit line *BL* that is separated by an NMOS. /I010<0>/net1 indicates the node inside the SRAM cell connected to the bit line $\overline{BL}$. So, the value at /I010<0>/net1 and /I010<0>net7 should always be opposite.

Before 30ns, 0 was stored in SRAM cell written in the previous clock cycle. /net11<0> is 1, but read in the previous clock cycle, so the value here is considered do not care. At 30ns, *clk* is high,

en is pulled down. Read operation is enabled. The /net11<0>, or the reading value, is updated to 0. At 35ns, *clk* is high, *en* is pulled to up, which enables write operation. Signal Write<0> at the moment is 1. So, 1 is written to SRAM. The result of /I010<0>/net7 = 1, /I010<0>/net1 = 0 verify this. However, at 39ns, the writing signal falls back to 0. Therefore, the SRAM cell value was updated to 0. The result of /I010<0>/net7 = 0, /I010<0>/net1 = 1 verify this. At 45ns, *en* is pulled to high.

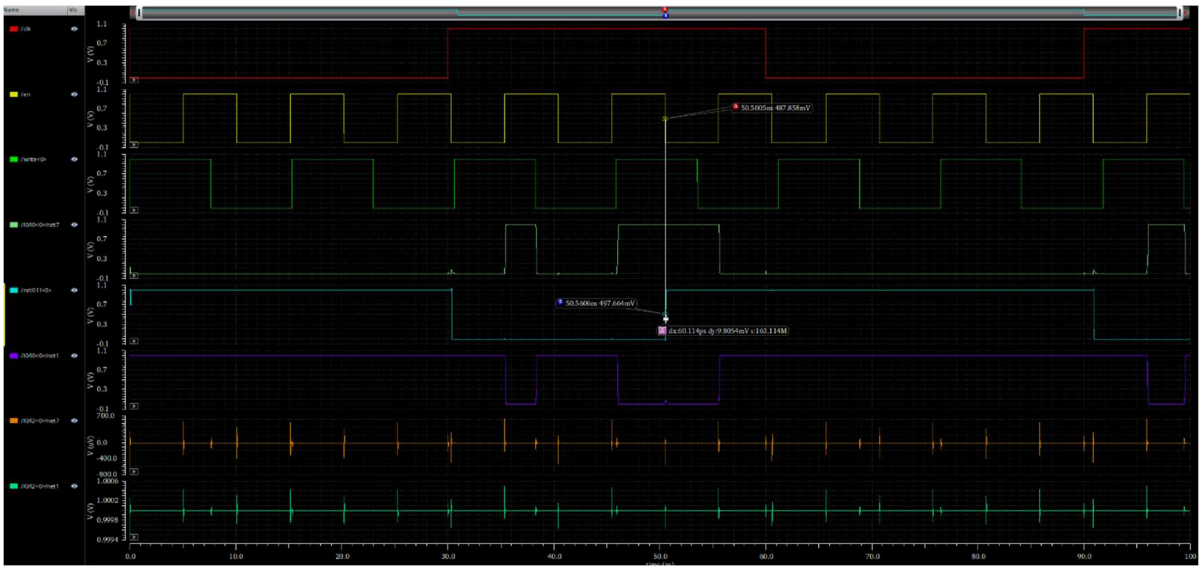

Figure 9: SRAM Simulation Result (rising edge delay)

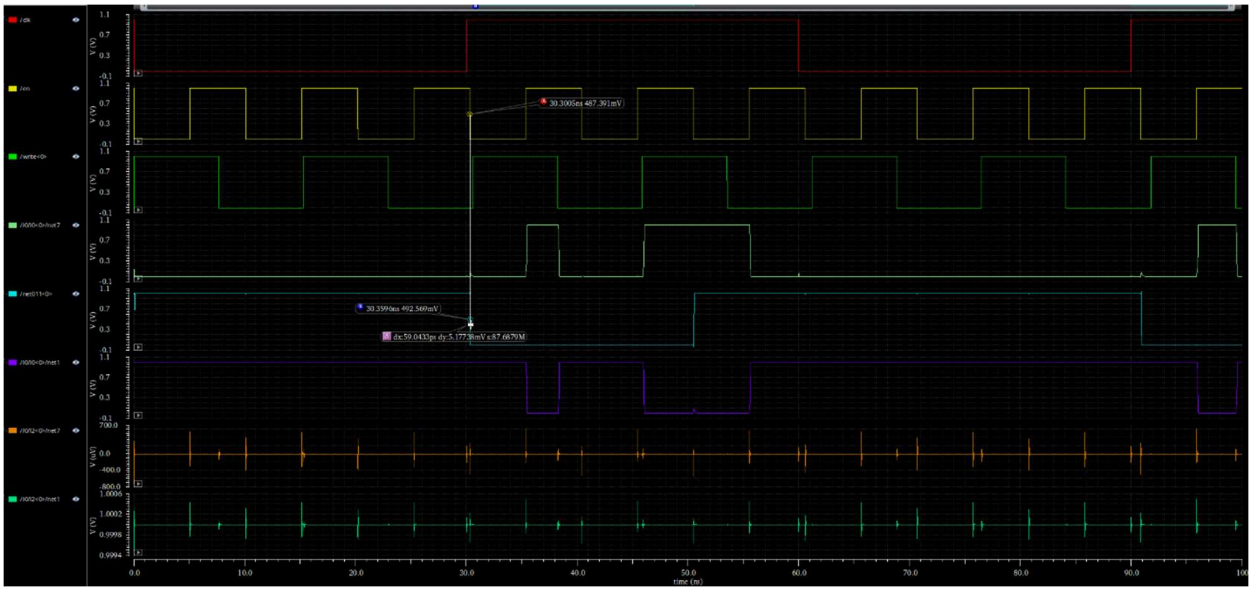

Figure 10: SRAM Simulation Result (falling edge delay)

3. Classifier

A typical classifier consisted of 1 flip-flop, 2 shifters, and an adder. To maximize clock frequency, pipelining is introduced. Specific implementations of individual parts and how pipelining is achieved will be discussed in the following pages.

Original Design

The original design does not introduce pipelining. After resizing components, the schematic is shown below in Figure 11.

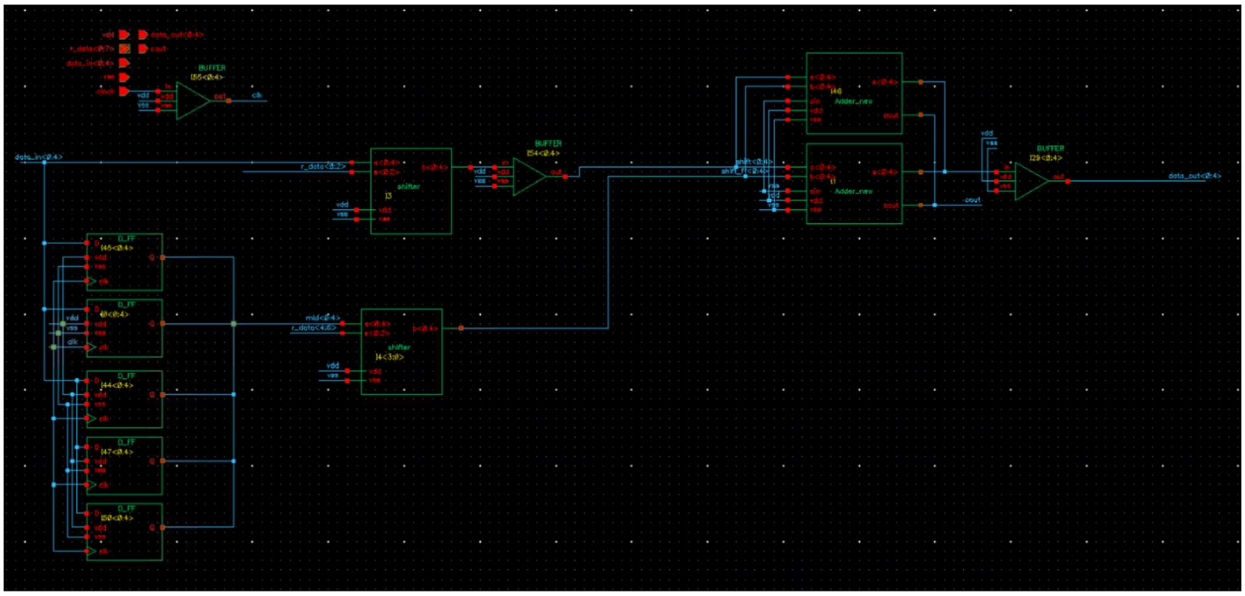

Figure 11: Schematic of Classifier (Original)

The testbench for the classifier is shown in Figure 12 below.

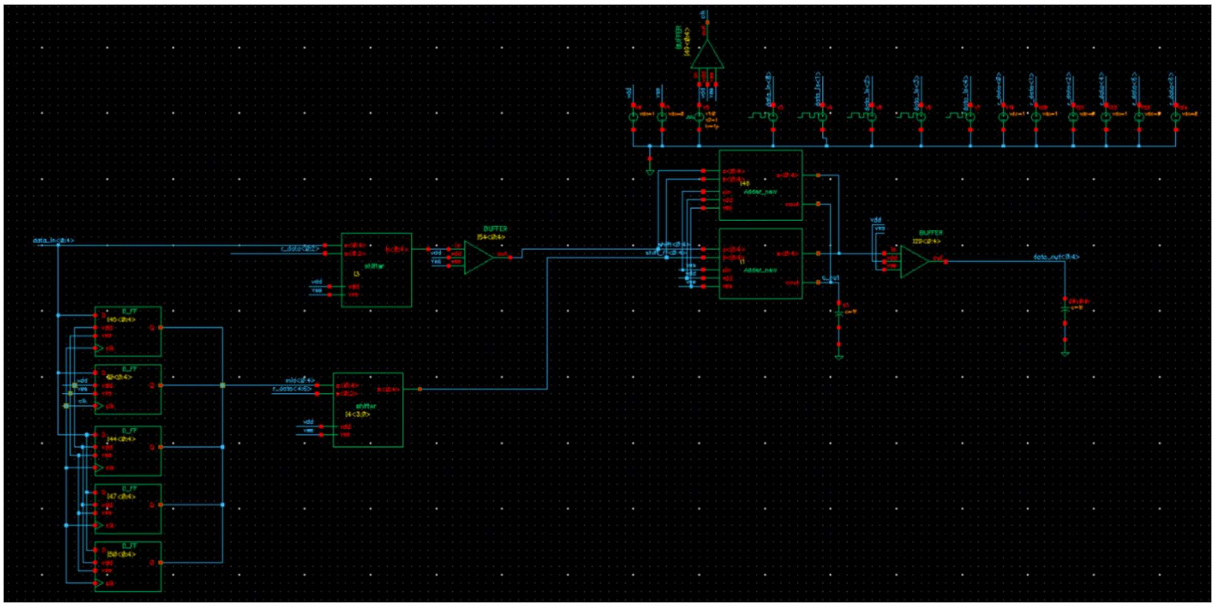

Figure 12: Classifier Testbench

As shown in the figure below, the total falling edge delay for the critical path, the 5th bit, is 302.74ps. The total rising edge delay for the critical path, the 5th bit, is 422.29ps. This is mainly caused by the cascading design of the carry select adder. So, the total delay is (422.29ps+302.74ps)/2=362.515ps. The maximum clock frequency can be set to 1s/422.29ps=2.36GHz.

In the simulation below, at clock =1.5ns, the arithmetic result is settled. The current data input is 10111. As shown in the previous figure, shift control bits for the first shifter are set to 011, which means the incoming data will be divided by 8. As the clock switched to high, the data input propagates through the flip flop. So, the data input at the second shifter is also 10111. The shift control bits for the second shifter are set to 001 as shown in the previous figure. This means the data input will be divided by 2. 10111/8=11110, and 10111/2=11011. 11011(-5) +11110(-2) =11001 (-7). In the Figure X below. At time 1.5ns, data_out<4:0> is 11001, which is the same as the hand calculated result.

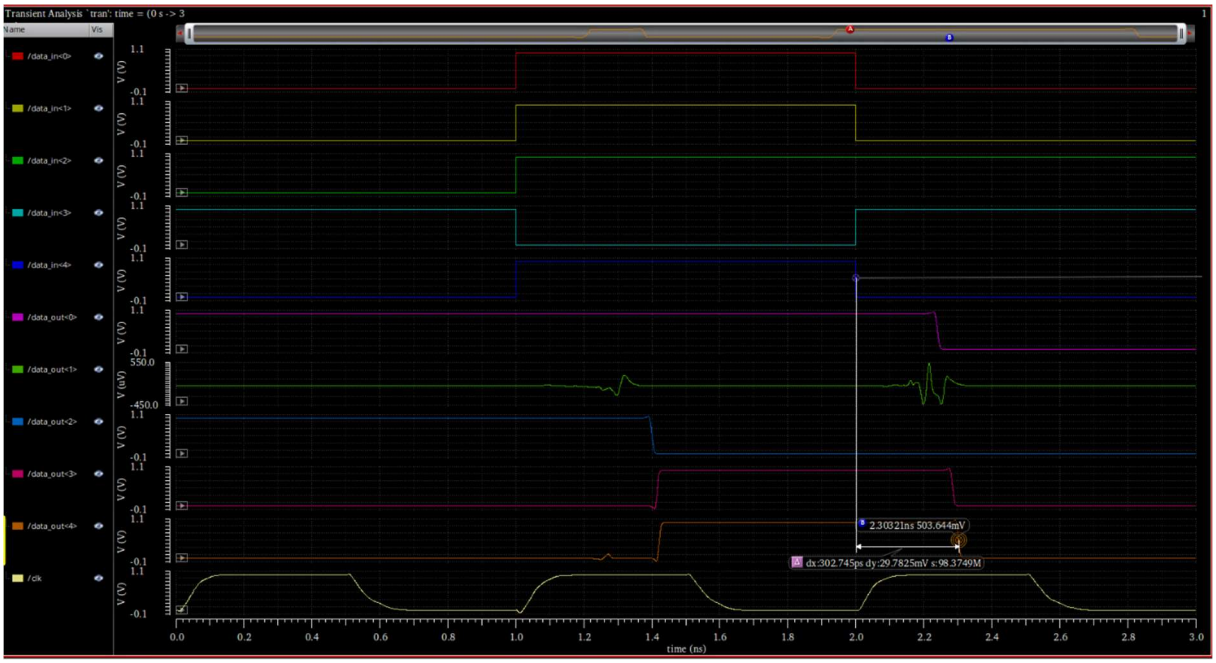

Figure 13: Classifier Simulation Result (rising edge delay)

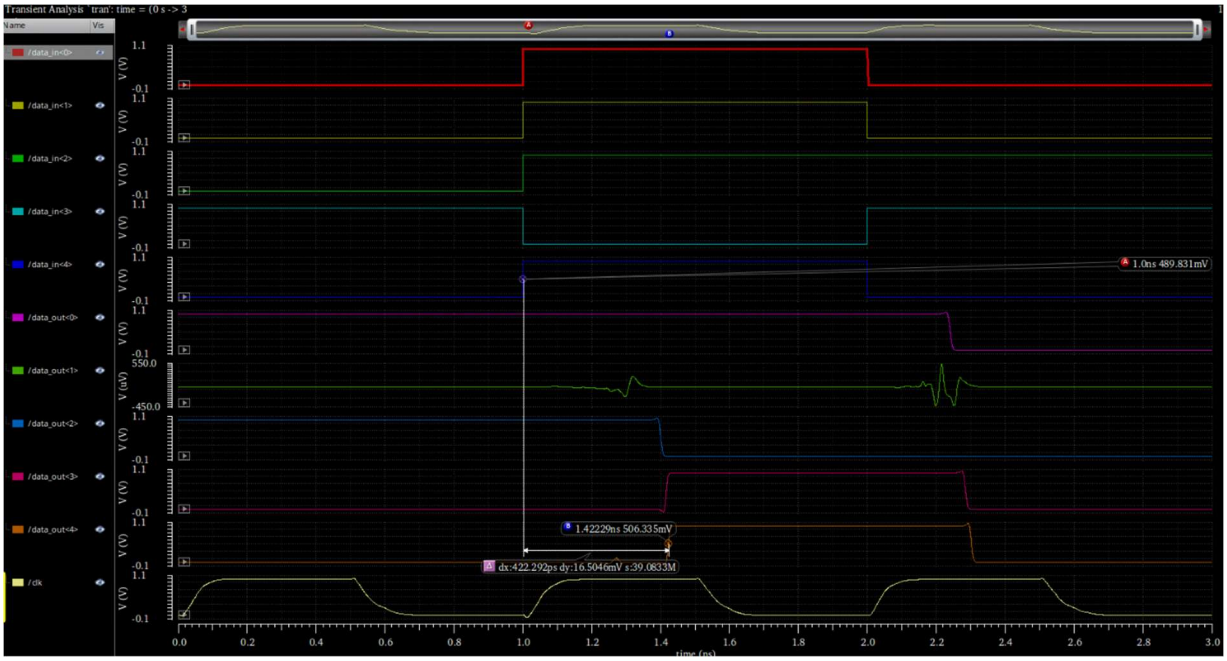

Figure 14: Classifier Simulation Result (falling edge delay)

3.1 Barrel Shifter

For this project, to realize that the input data can be divided by 2, 4, 8, 16, and a barrel shifter consisting of 15 mux is used. There are three select signals to determine how many bits the input date will be shifted (as shown in Table.1).

S0	S1	S2	Weight
0	0	0	1
1	0	0	1/2
0	1	0	1/4
1	1	0	1/8
0	0	1	1/16

To test the barrel shifter, and the test bench is completed as shown in Figure 15, For the first 3 test cases, the input is set as 01000, when it is divided by 2,4,8, the consequence should be 00100, 00010, 00001, respectively. Figure 16, 17, 18 show the consequence and they prove that the shifter can work as expected.

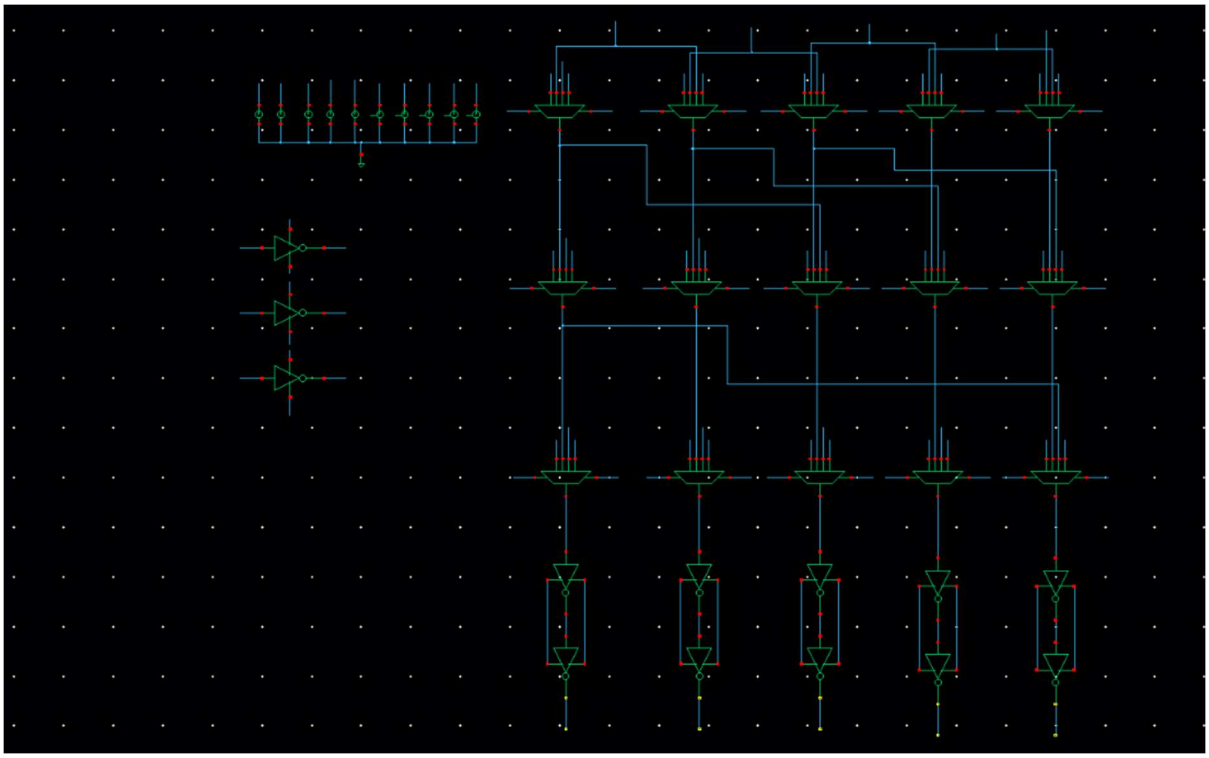

Figure 15: Barrel Shifter Testbench

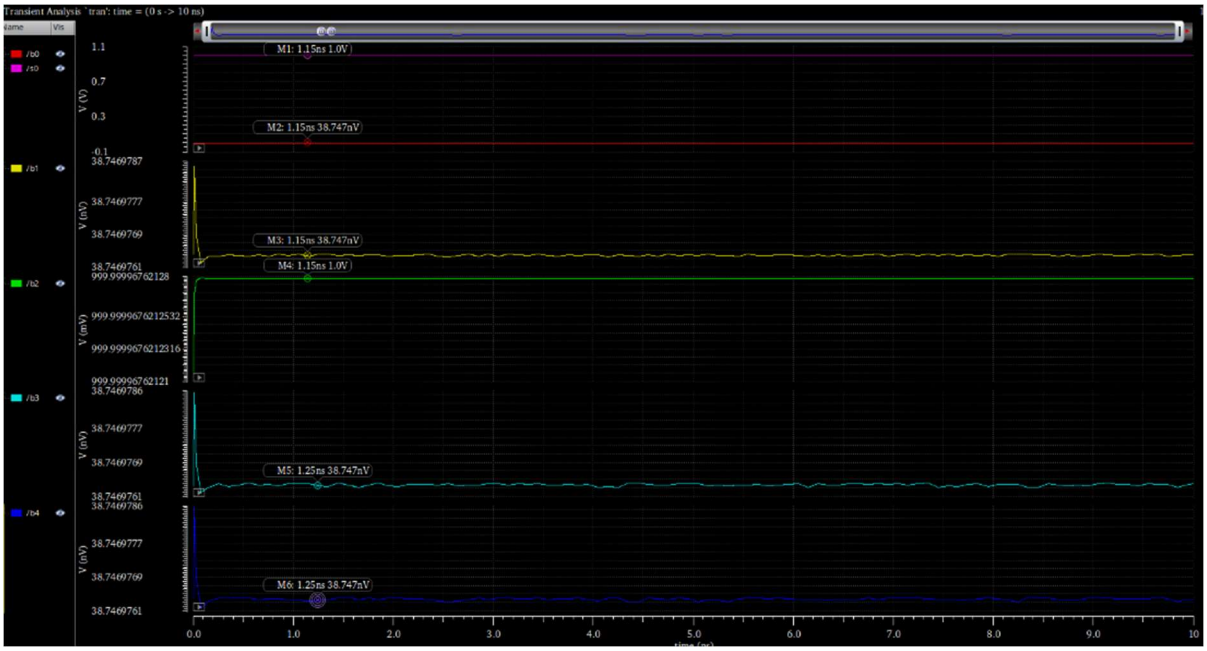

Figure 16: Test 1 (01000 to 00100)

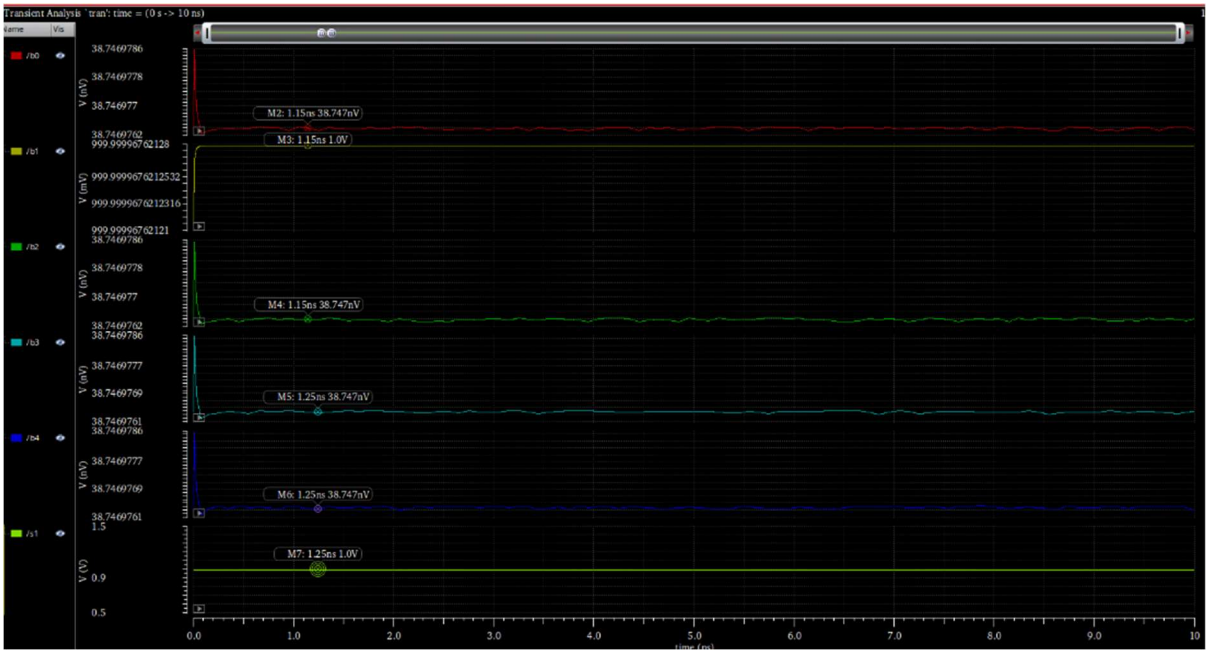

Figure 17: Test 2 (01000 to 00010)

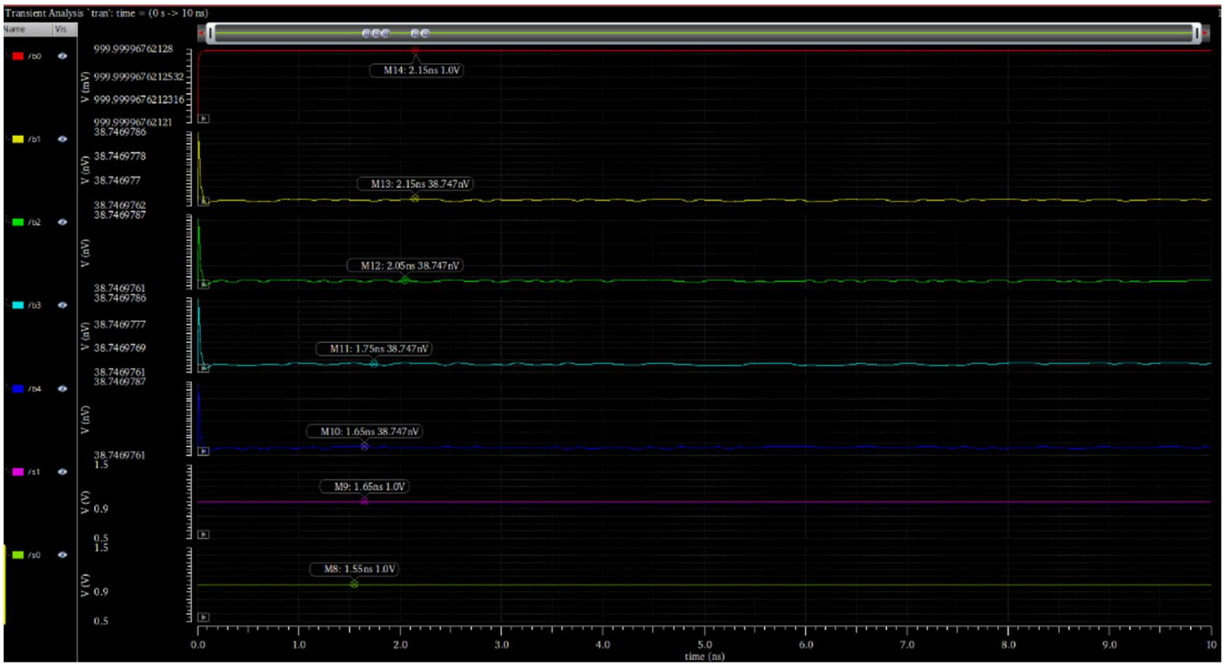

Figure 18: Test 3 (01000 to 00001)

For the last test case, the input is set to 01110, which is 15 in decimal, when it is divided by 16, the consequence should be 00000 since 15/16 is smaller than 1. Figure 19 shows the consequence, and it proves that the shifter can work as expected.

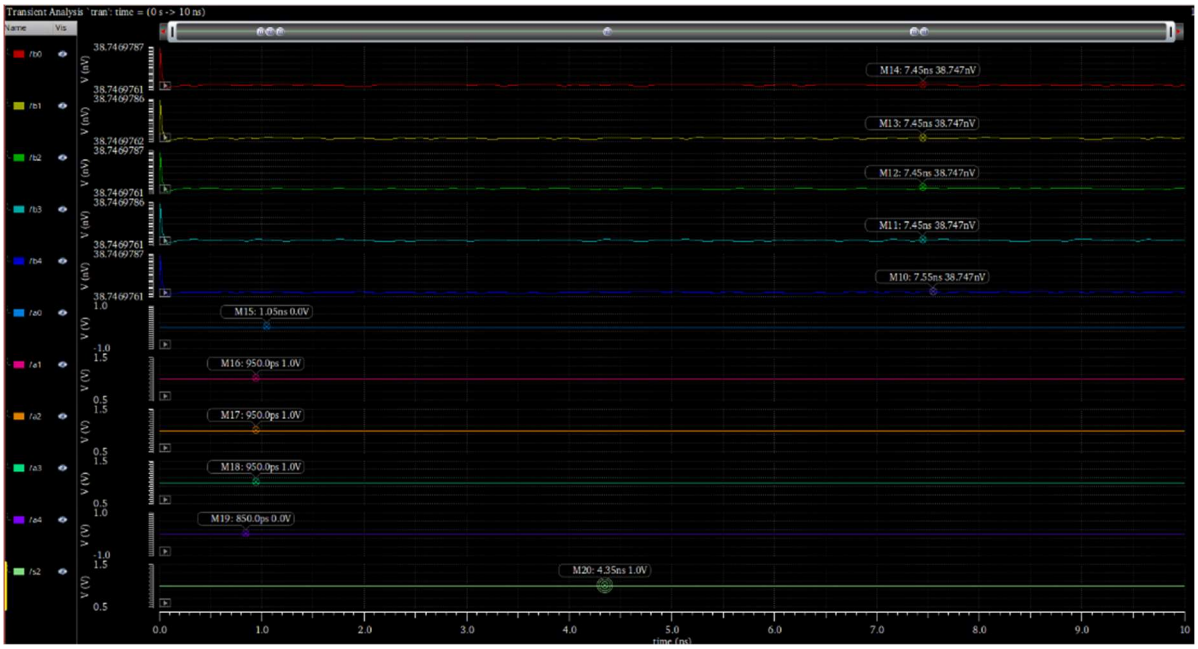

Figure 19: Test 4 (01110 to 00000)

For this barrel shifter, the delay of the critical path should be the delay of 3 mux. In the test shown in Figure 20, a critical path from a0 to b1 is selected, by calculation, the delay of this barrel shifter is 73.14ps.

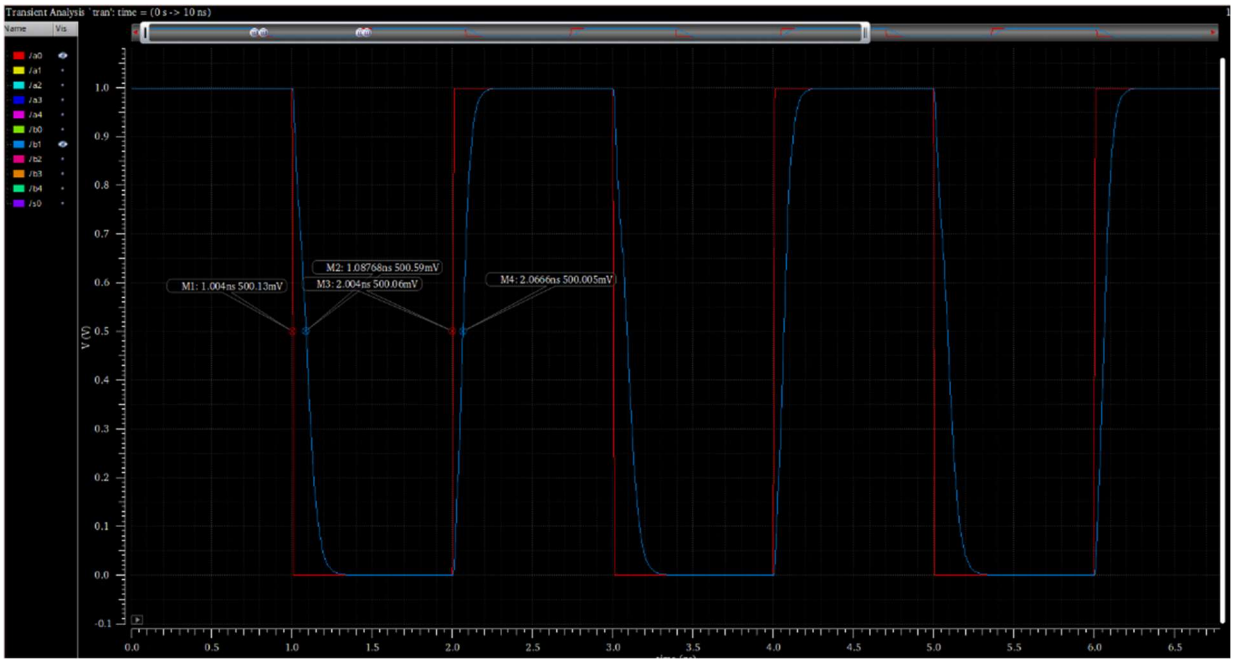

Figure 20: Delay of the Barrel Shifter

3.2 Carry Select Adder

For this project, a 5-bit carry select adder will be implemented. The carry selected adder consists of 5 full adders and 5 MUX (as shown in Figure 21). The carry select adder will calculate the two results with the default carry of 0 and 1 respectively. When the carry of the previous bit comes, the output result can be selected through a mux.

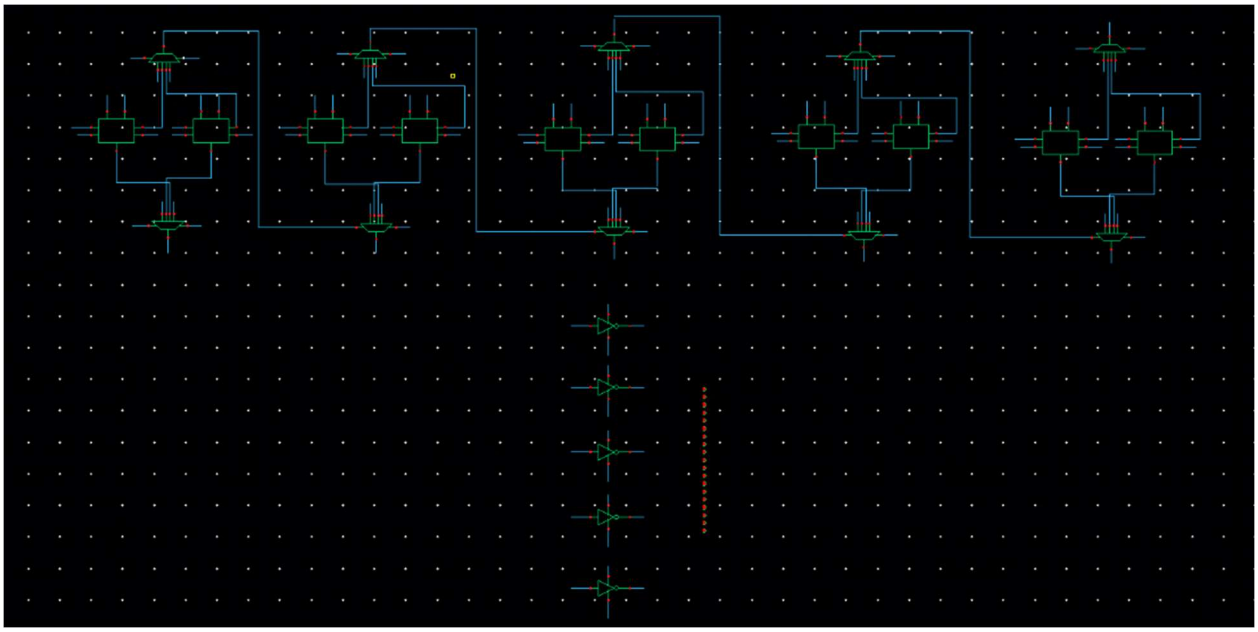

Figure 21: Schematic of 5-bit Carry Select Adder

To test the adder, a buffer is connected to the output of the adder so that the output signal can be strongly pulled up to 1 or pulled down to 0. The test bench is shown in Figure 22.

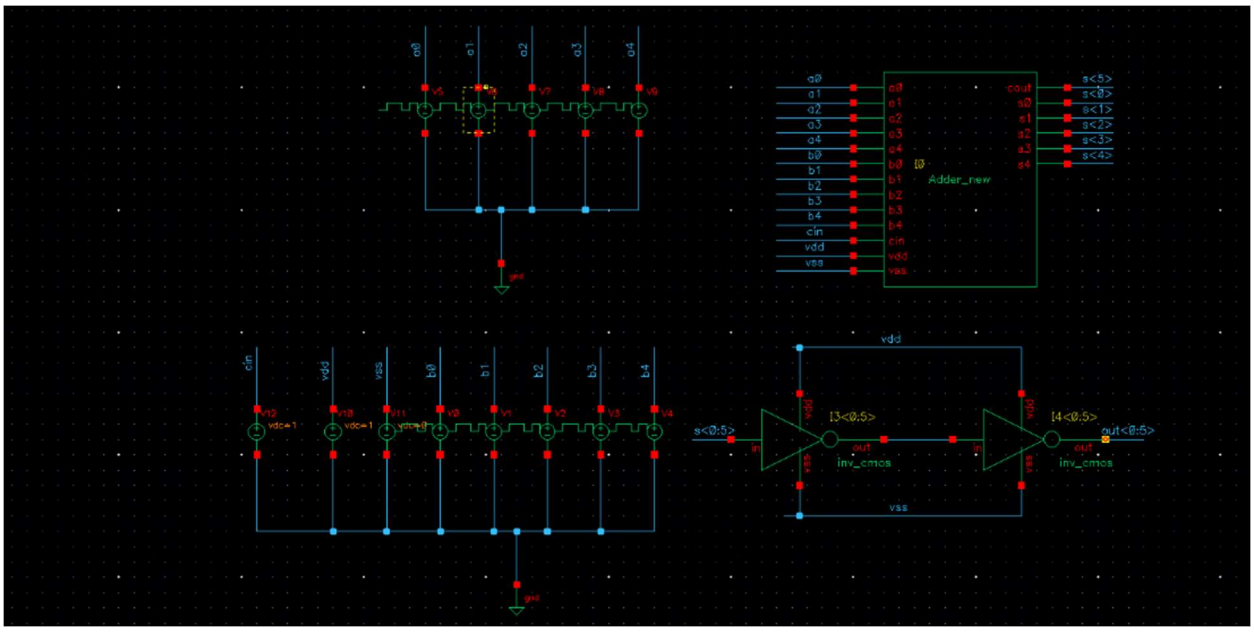

Figure 22: Carry Select Adder Testbench

For the first test case, we set carry-in to 0, and set both two inputs to 11000, the expected result is 10000 with 1 carry-out. The test result is shown in Figure 23, and the adder works as expected.

For the second test case, we set carry-in to 1 and set both two inputs to 11000, the expected result is 10001 with 1 carry-out. The test result is shown in Figure 24, and the adder works as expected.

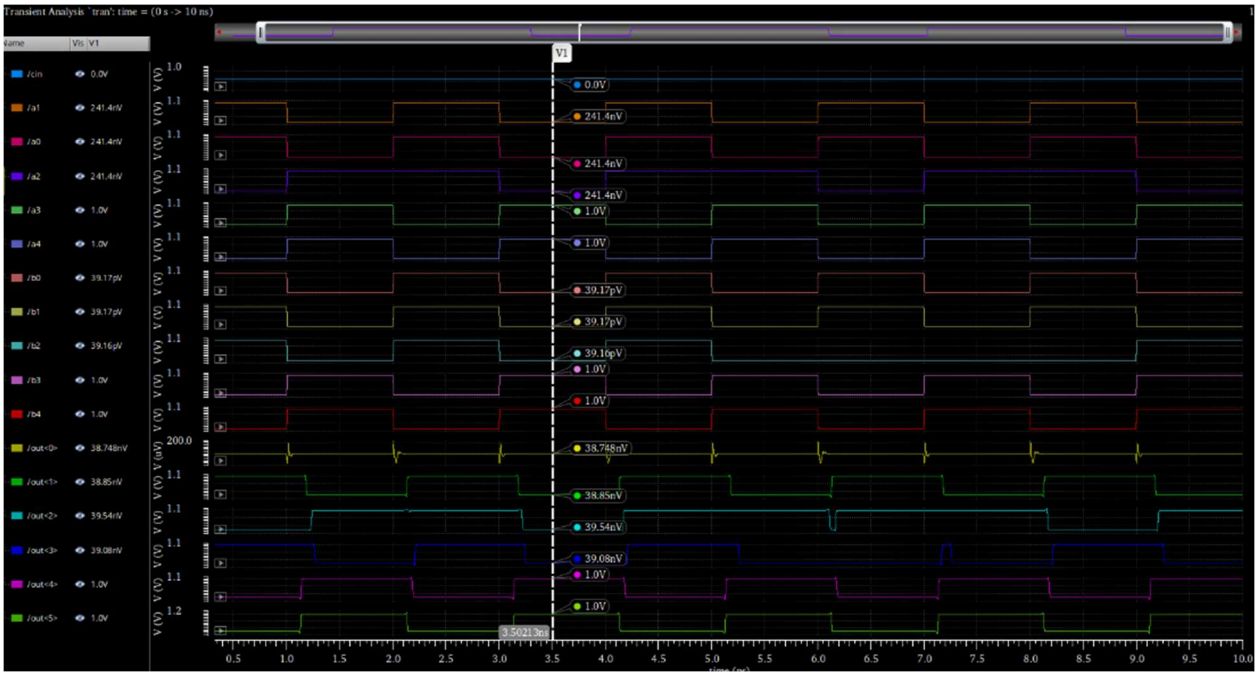

Figure 23: Test 1 (11000+11000 with 0 carry-in = 10000)

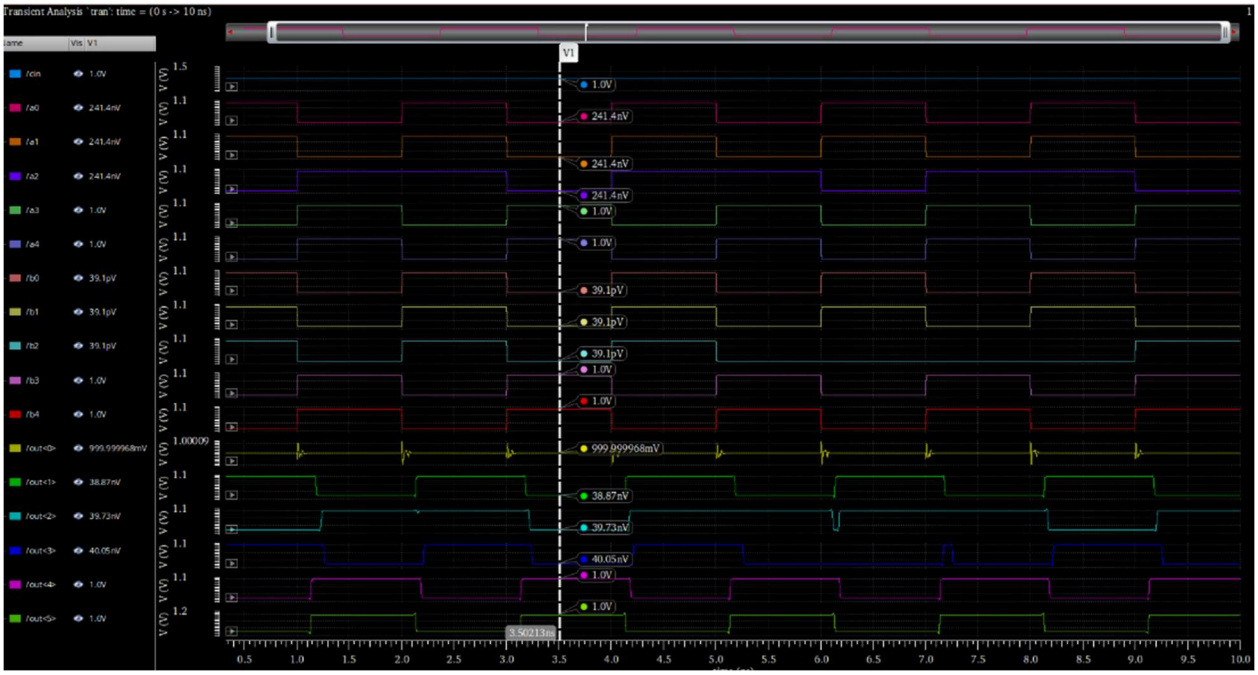

Figure 24: Test 1 (11000+11000 with 1 carry-in = 10001)

To test the delay of the 5-bit carry select adder, the critical path is found as from a0 to Cout, Vprbs are used to generate random input, and carry-in is set to 0. The test result is shown in Figure 25, by calculating, the delay of this adder is 130.5ps which is acceptable.

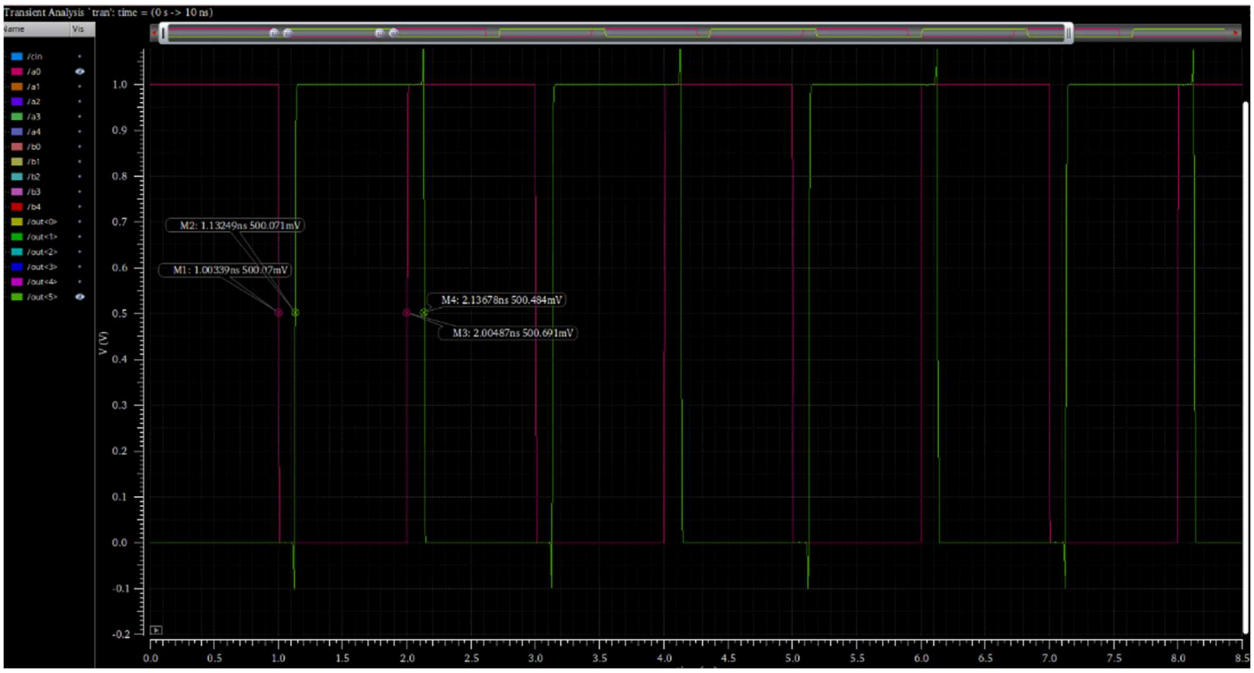

Figure 25: Delay of the Carry Select Adder

3.3 Maximizing clock frequency

To maximize clock frequency, pipelining is introduced. This is achieved by adding one more stage between shifters and adder with one more flip flop inserted. The detailed design is shown below in Figure 26.

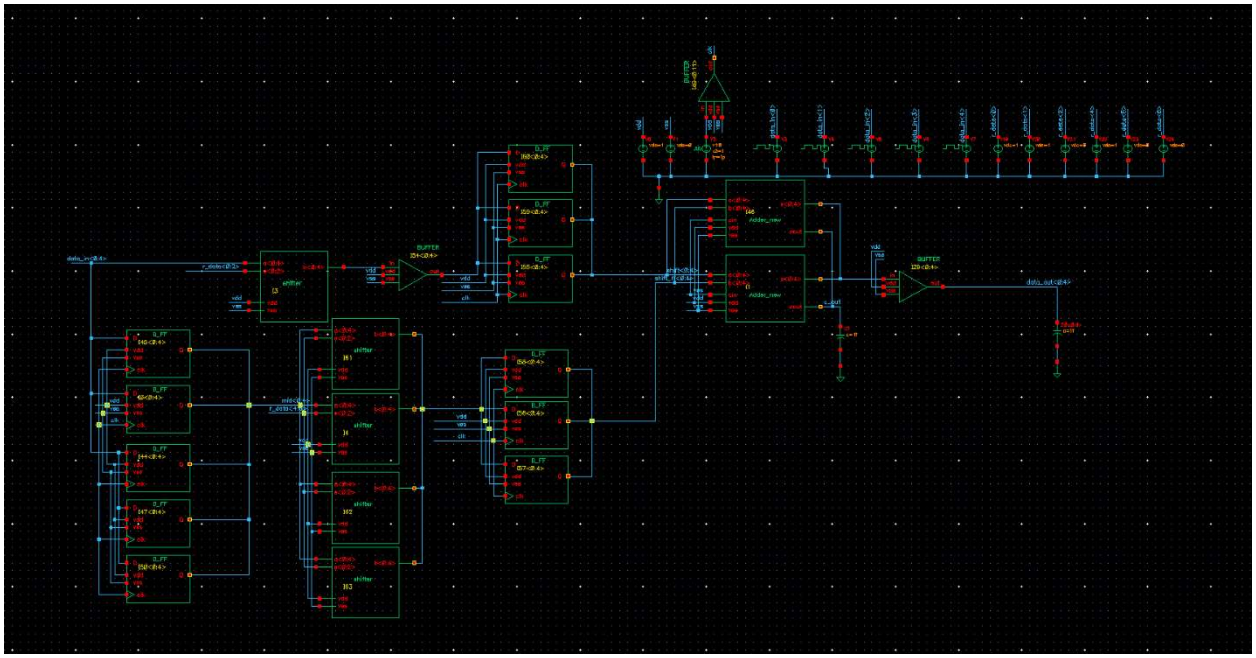

Figure 26: Schematic of Pipelined Classifier

In the testbench, the clock frequency is set to 3GHz. The weight is fixed to 011 and 001, which means shifter 3 bits and 1 bit, respectively. The result is shown below. At time=260ps, the input is set to 01000 (+8), so the output of two shifters is 00100(+4) and 00001(+1), then the final result is 00101 (+5). As shown in Figure 27, data_out<0:4> is 00101 (+5), which is the same as the hand-calculated result. The classifier works as expected.

From Figure 28 and Figure 29, the delay of rising edge and falling edge is 299.19ps and 281.05ps. The average delay is (299.19ps+281.05ps)/2=290.12ps. The theoretical maximum clock frequency calculated with rising edge delay is 1s/299.19ps=3.34GHz.

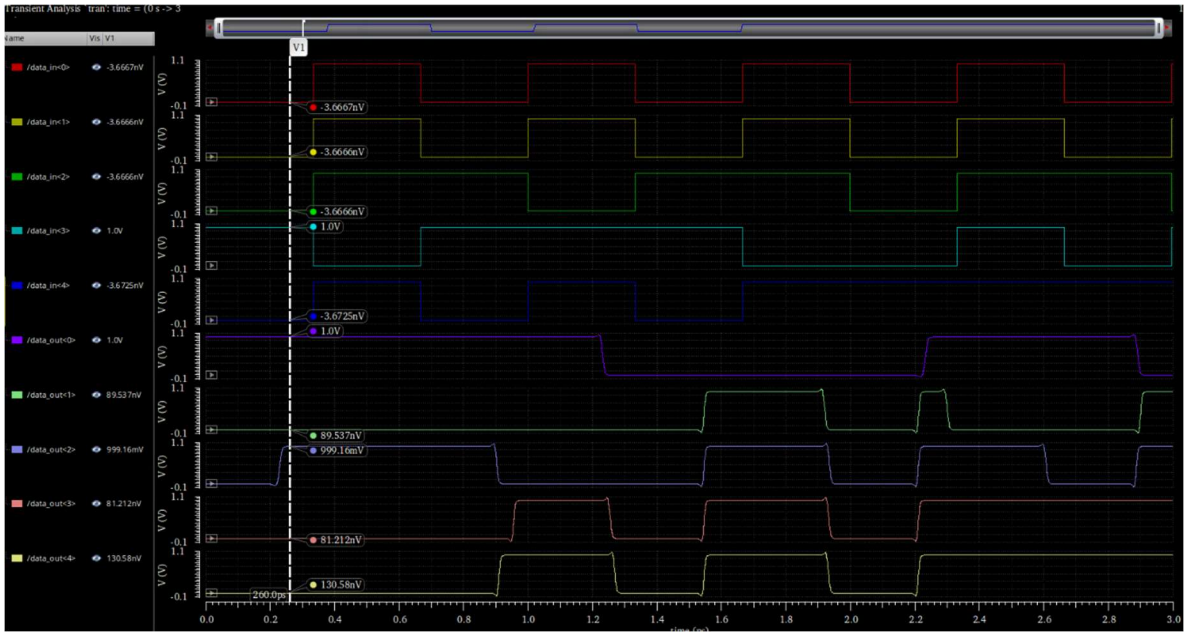

Figure 27: Test Result of Pipelined Classifier

Figure 28: Delay of Pipelined Classifier (rising edge)

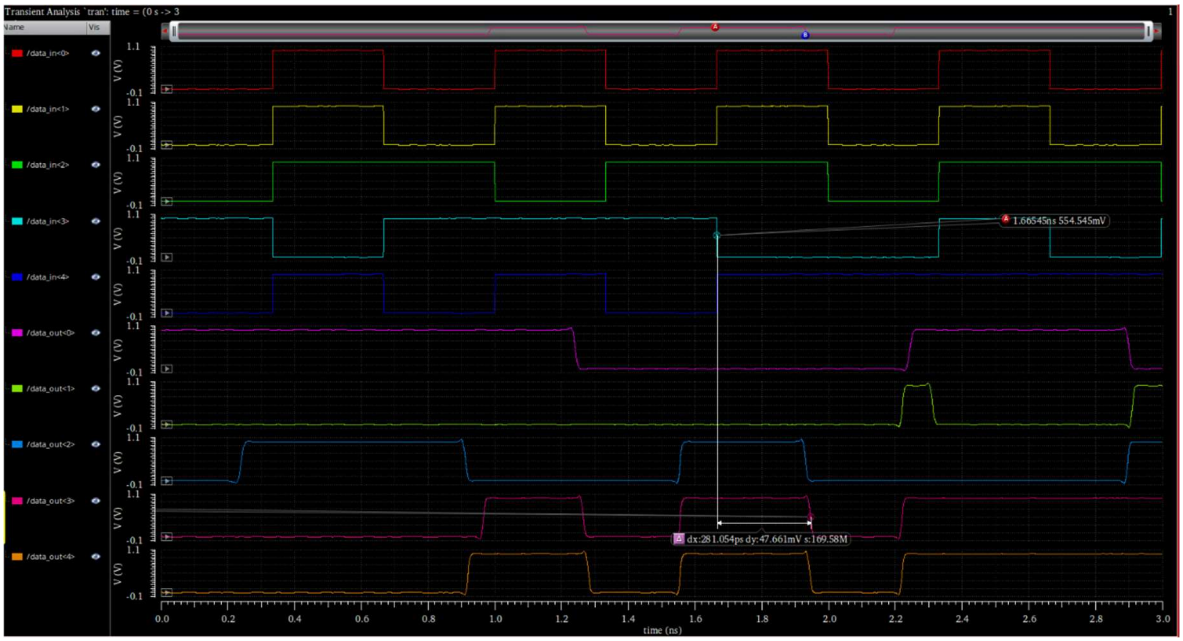

Figure 29: Delay of Pipelined Classifier (falling edge)

4. Top-level Testbench

At the top-level testbench, both pipelined classifier and SRAM are packaged into symbols. The schematic is shown below in Figure 30.

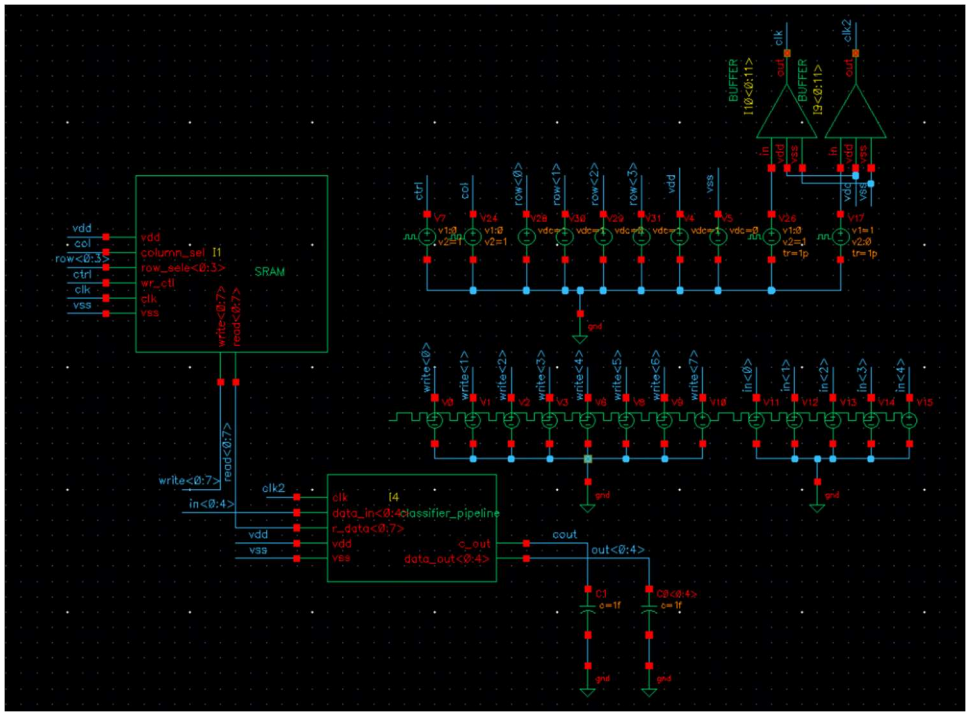

Figure 30: Schematic of Top Level Testbench

The Simulation result is shown in the Figure 31 and Figure 32 below.

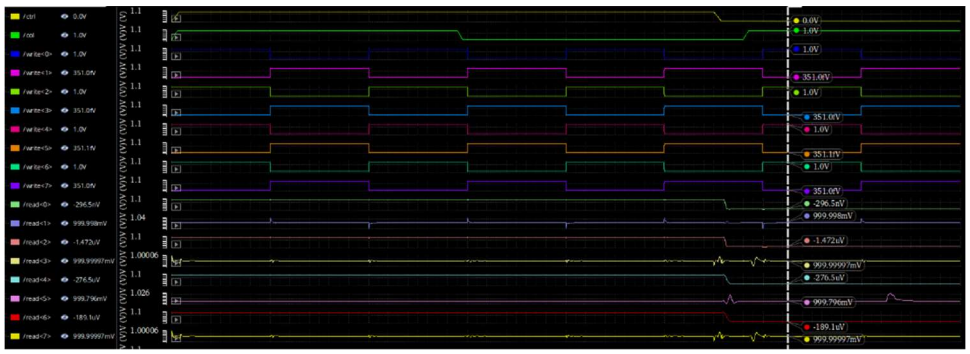

Figure 31: Top Level Simulation Test Result 1

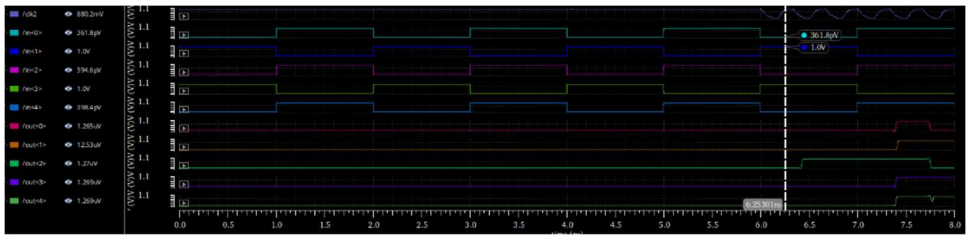

Figure 32: Top Level Simulation Test Result 2

As shown in Figure 31, weights are written to SRAM. Initially, the write/read control is set to 1 to enable writing. The 8-bit memory cell at column 1 and row 11 is selected. The data written into the SRAM is 10101010. Then, column 0 is selected for later verifying if the

memory at column 1 is written successfully. At time = 5.5ns, the write/read control is set to 0 to enable reading. And column 1 is re-selected. This time from read bus, the previous written data 10101010 is successfully read to the classifier. Bit <2:0> will be passed to the first shifter and bit <4:6> will be passed to the second shifter connected by the flip flop. At time = 6ns, the clock for the classifier starts to work. The incoming data at time = 6.25 ns is 01010 (+10). As a rising clock edge, the data is passed through the flip flop, the incoming data for both shifters are 01010. As bit <2:0> is 010, and bit <4:6> is also 010 or divided by 4. So, the result for both shifters is 00010 (+2). After adding up two results from shifters, the output should be 00100 (+4). And At time = 6.5ns, the result out<4:0> is already settled to 00100. The arithmetic result is correct.

5. Power Consumption

5.1 SRAM Power Consumption

For SRAM power consumption, the same testbench as shown in Figure 26 was simulated in four operation cycles (two read and two write). The simulation result is shown in Figure X. As seen in this waveform, the instantaneous power is calculated by equation Vdd*Idd, which is 42mW. By using the average function, the average power consumption of this 32-word SRAM is 1.72mW. Between time = 5ns to10ns, 15ns to 20ns, there is a constant power consumption because the Vdd keeps pre-charging all the SRAM cells as the clk is 0.

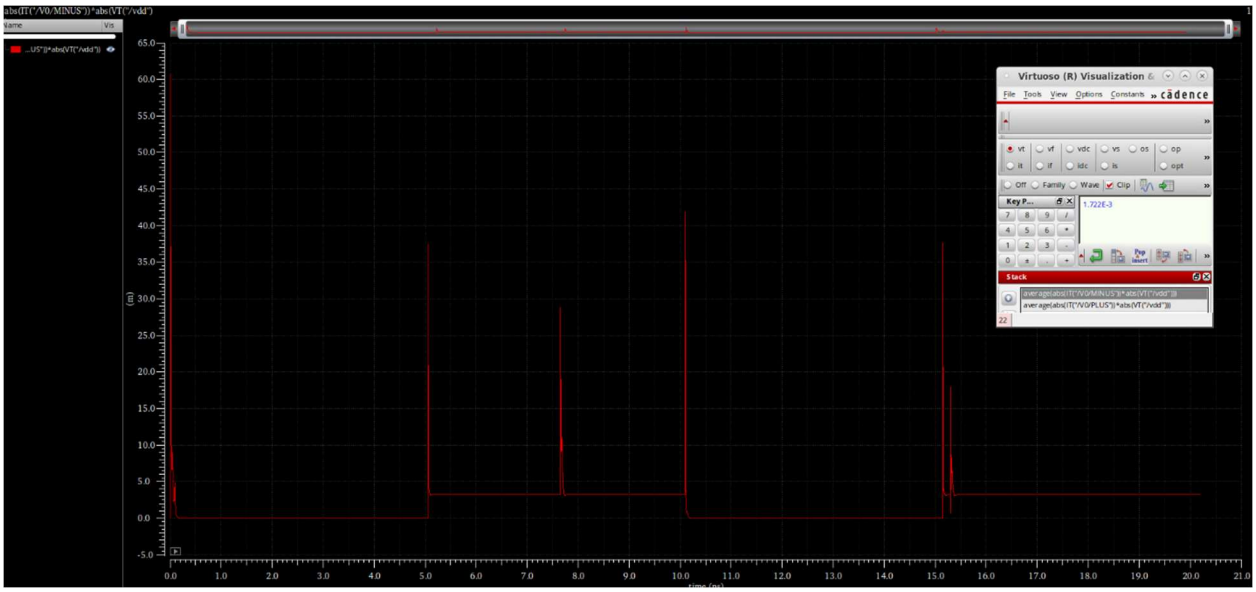

Figure 33: SRAM Power Consumption

5.2 Classifier Power Consumption

For pipelined classifier power consumption, the same testbench as shown in Figure 8 was simulated in ten operation cycles. The simulation result is shown in Figure 34. As seen in this waveform, the instantaneous power is calculated by equation Vdd*Idd, which is 28mW. By using the average function, the average power consumption of this pipelined classifier is 5.349mW.

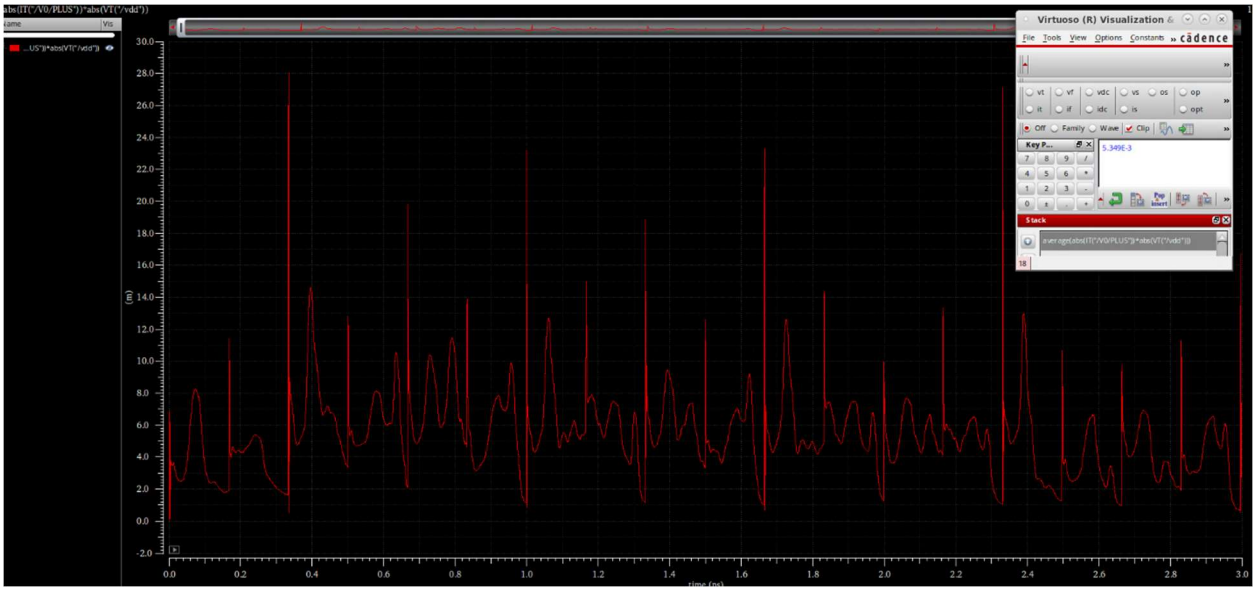

Figure 34: Pipelined Classifier Power Consumption

5.3 Total Power Consumption

For the entire design, the power consumption is tested in the same testbench as shown in Figure 30. The simulation result is shown in Figure 35. As seen in this waveform, the instantaneous power is calculated by equation Vdd*Idd, which is 29mW. By using the average function, the average power consumption of this pipelined classifier is 3.562mW. The total power consumption is dependent on the intensity of the arithmetic operation and time elapsed.

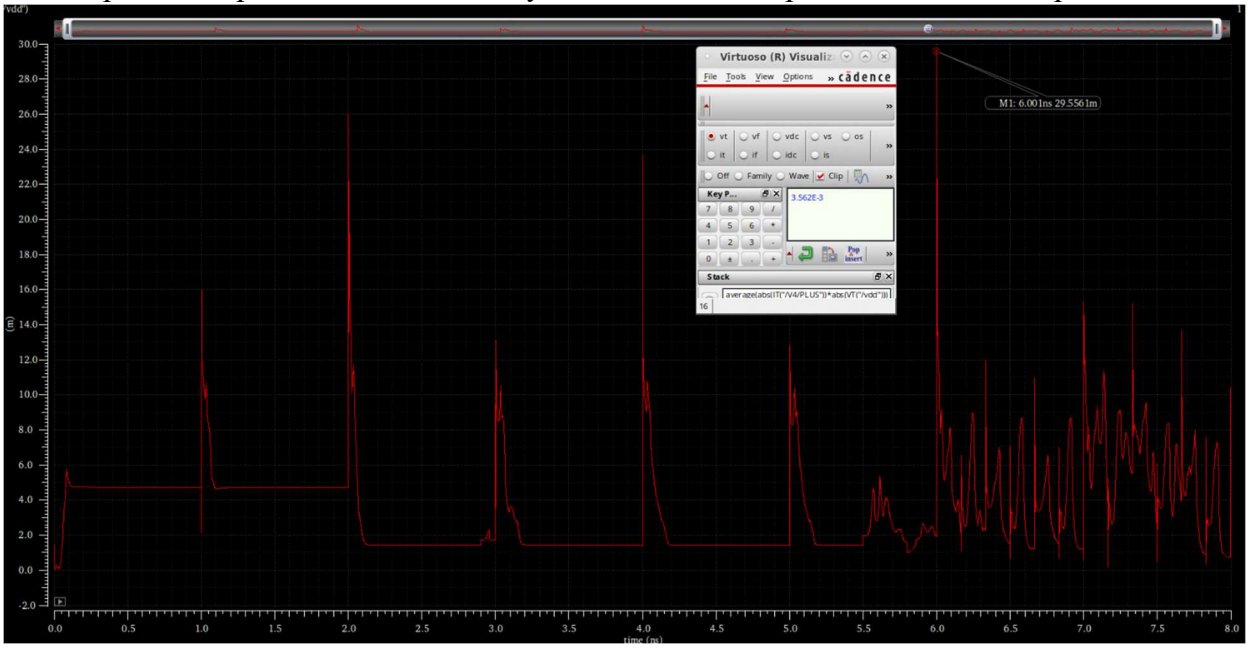

Figure 35: Total Power Consumption

6. Conclusion

By designing and combining all the elements, both SRAM and classifier work as expected. To maximize clock frequency, the classifier is resized by controlling fan-out and adding buffers, and the maximum clock frequency is improved to 2.36GHz. In order to further maximize the clock frequency, pipelining is achieved by adding one more stage between shifters and adder with one more flip flop inserted. And the result is verified at 3GHz. However, the theoretical maximum clock frequency is 3.34GHz.