Design and Optimization of Non-Volatile Latch using Resistive Memory Technology

Vladislav Miftakhov[1], Cody Del Prato[1], Søren Tornøe[1], Kwan Lim[1], Aliya Attaran[2], Hamid Mahmoodi[2]
[1]Canada College, Redwood City, CA
[2]San Francisco State University, Daly City, CA

Abstract

Spin Transfer Torque Random Access Memory (STTRAM) is a new area of memory technology that shows great potential in certain applications. STTRAM is a technology for information storage that has the advantage of non-volatility, complementary metal-oxide-semiconductor (CMOS) scalability, and added security of the hardware. In this technology, information storage is in the form of magnetic orientation, compared to the existing charge-based memories such as static random-access memory (SRAM), dynamic random-access memory (DRAM), and flash. In our non-volatile (NV) latch design, we employ two Magnetic Tunnel Junctions (MTJ) that store data in resistive form, a write driver that changes the state of the MTJ cells, and a sense amplifier that sets the state of the volatile cell (output) based on the state of the non-volatile cells. To increase sensing reliability, the design uses the two MTJs as differential resistive cells, which make the entire latch a one-bit storage cell. The write control and bit-lines in the design are shared. Although this latch provides the benefits of STTRAM technology, it is larger than SRAM, DRAM, and flash due to the combined area of the transistors needed in the circuit. Given that the MTJs are set at minimum size and the length of our transistors is set at the available 32nm technology, the only size parameter we can modify is the width of the transistors. However, as the transistors decrease in width, the delay of the circuit increases up to a point at which the latch fails to output proper values in the read cycle or the MTJs fail to switch states in the write cycle. The purpose of this research is to correlate delay with failure rate, and then optimize the circuit for minimal area while still retaining a failure rate of less than 0.1%. We found that delay has, on average, a relative inverse relationship to the area of the circuit. As area increases, delay decreases. The failure rate of the circuit decreases as delay decreases. The best transistor values for the write circuit resulted in a delay of 2.999 nanoseconds at a total area of 15.249 micrometers. For the read cycle, we had to account for the capacitive effect of the write operation transistors. We modified the original circuit to include these transistor values before starting the optimization of the read operation. The sense enable pmos and both of the output buffer transistors in the sense amplifier work at minimum size of 0.1 micrometer. The most sensitive transistors are the 4 which make up the two back-to-back inverters and the bottom sense enable nmos through which a relatively large current passes. The Read path transistors were optimized with the optimized write path transistors widths set as constants. The optimized Read path was much smaller than the Write path, despite having almost twice the transistors. The Read path was more sensitive to changes in area. After optimization, the latch was configured in hspice layout to generate the parasitics of the fully optimized circuit. The parasitics generated did not affect the reliability of the circuit.

1 Introduction

Hardware security is an ongoing major concern for both private and government organizations. The Integrated Circuit (IC) designs that these organizations develop are at risk of being probed, reverse engineered, and illegally reproduced. One of the leading solutions to protecting these

designs is replacing conventional charge-based logic and memory with reconfigurable logic based on Spin Transfer Torque (STT) technology. Spin Transfer Torque technology allows chips to store binary information in the form of magnetic resistance as opposed to using charge-based memory such as SRAM, DRAM, and flash. This allows information to be retained without power. STTRAM utilizes the advantages of SRAM, DRAM, and flash memory, which are fast read/write speeds, CMOS compatibility, and non-volatility, respectively.

A major potential application of this technology is replacing existing field-programmable gate arrays (FPGA). This technology has the potential to be much faster than FPGA. A non-volatile STTRAM latch with a Multiplexer can be used to build Look Up Tables (LUT) that have reconfigurable logic.

2 Background

In this project, we optimized a non volatile latch that is using resistive memory technology. Compared to traditional computer memory, where information is stored as charge, resistive memory technology has information stored as a resistance state. The resistance state of the non-volatile cell is altered by a write driver, while a sense amplifier interprets the resistance of the non-volatile cell and sets the state of a volatile cell based upon that interpretation. The first aspect we focused on within the project, was to optimize the size, area, and delay of the transistors in the read and write functions of the latch. This process was completed by exporting a schematic of the circuit from HSpice as a netlist, and then adding parameters into the netlist that could then be swept and the results plotted. These results were then used to establish a range for the optimization to focus on. After determining what size transistors were the most effective for this application, the circuit was subjected to a Monte Carlo simulation to test for failure.

2.1 Non-Volatile Technology

Traditional memory storage systems that are used in almost all computer systems today rely on charges and charge levels to store information. Over time this charge can deteriorate and under sudden power failure be lost resulting in the permanent loss of information. Common examples of this include power outages, computers crashing, and batteries becoming faulty. The traditional method of storing memory is considered volatile memory as it can be lost very easily. The project we worked on presents a potential solution to this now commonplace problem using resistive memory technology which is non-volatile by nature. While the technology could be used for any memory system type, we are specifically looking at its applications in Random Access Memory, RAM, for computers.

The resistive memory that we used for our project is Spin-Transfer Torque RAM, STTRAM, which uses a latch consisting of two magnetic tunnel junctions and seventeen transistors. Information is stored in the magnetic tunnel junction that can either be a binary one or zero bit of

data. The reason information cannot be lost with this design is that it takes current to change the value of the data bit, which is determined by the resistive value present at the junction. The bit is held in place by the being in either of the two aforementioned states, both of which are the only stable forms that the junction can take.

2.2 Magnetic Tunnel Junction: How it Works

Magnetic Tunnel Junction or MTJ is a circuit element that consists of a pinned, fixed, magnetic layer—an insulating layer, which is the center of the circuit element—and a free magnetic layer. The pinned layer, composed of a ferromagnetic substance such as Cobalt-Iron-Boron (CoFeB), generates a magnetic field that points only in one direction. The free layer, also made of CoFeB, can point either in the same direction as the pinned layer or in opposition to it; this is called parallel orientation and anti-parallel orientation respectively. The insulating layer is composed of crystallized magnesium-oxide, MgO, which acts as a natural resistive barrier to the flow of electrons. When this insulating barrier is small enough, just a few nanometers, then electrons can, as defined by quantum mechanics, tunnel their way through the barrier and to the other side. The free layer changes orientation based on Spin Transfer Torque of the electrons and the direction of the current.

Parallel orientation can be generated when electrons pass through the pinned layer first, represented in *Figure 1A* below. The electrons take on a specified orientation that matches the fixed layer's magnetic orientation. Due to the electron's spin polarization they will apply a torque onto the free layer causing the free layer's magnetic orientation to match the fixed layer's. During this process some electrons will be reflected back towards the pinned layer but since the pinned layer is fixed it will have a negligible effect. To generate perpendicular orientation, *Figure 1B*, the electrons must flow from the free layer to the pinned layer. Since, the electrons are flowing in the opposite direction as the previous explanation, then the torque will be applied in the opposite direction causing some of the electrons to bounce off of the pinned layer. If enough electrons bounce off the pinned layer, where they will then take up the magnetic orientation of the pinned layer, then the torque they exert will cause the free layer to become anti-parallel.

A **B**

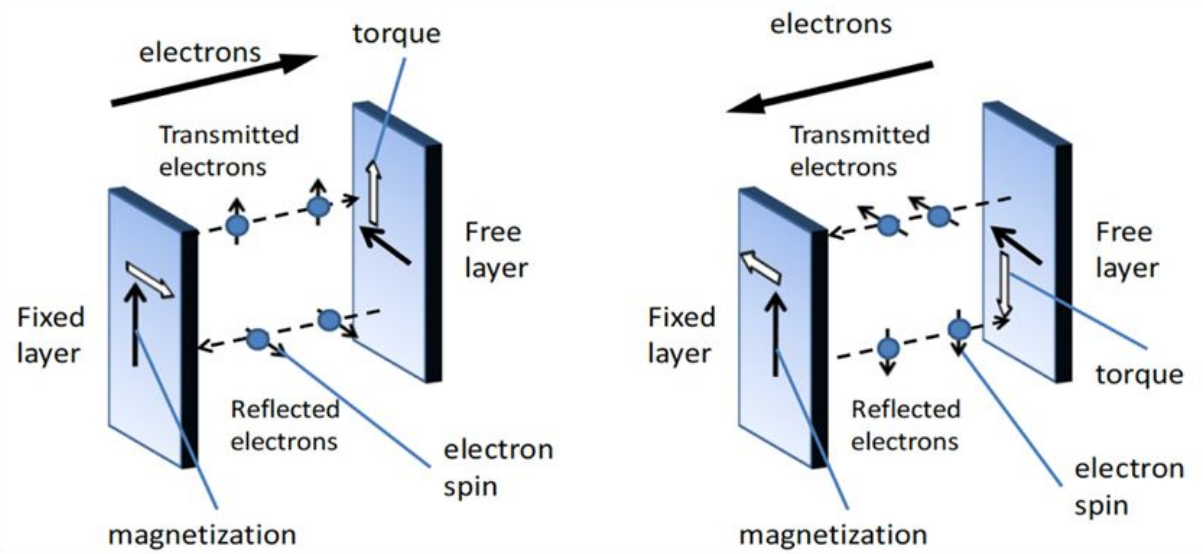

Figure 1. Physics of a MTJ from: Iong Ying Loh, Master Thesis, MIT, 2009

2.3 MTJ Operation Mechanism: Read & Write

The orientation of the free layer with respect to the pinned layer can be interpreted as a binary system. When the layers are parallel to each other, a configuration that has a low resistance, the state can be interpreted as being a zero. If the layers are antiparallel, a configuration with a high resistance, the signal can be interpreted as being a one. We can use this binary Magnetic Tunnel Junction system to represent one bit of memory data or a memory cell. For our purposes we will use two MTJs that will always have opposite orientations of each other. In order to read the bit value of the memory cell a current must be passed through the MTJ that is well below the critical write current, the current value that causes the MTJ free layer to change orientation. This lowered current is still large enough to trigger the transistors in the inverters in the latch above the MTJs. This latch has two outputs one for each MTJ. Since the two MTJs are always opposite each other, one will always have a lower resistance and thus drain to ground faster. The MTJ that drains to ground first will trigger a response from its related output in the latch. Depending on which output triggers we can determine if the bit of data being stored in the MTJ was a one or a zero.

The process of writing data values to the MTJ is the same as changing the free layer's orientation from parallel to antiparallel or antiparallel to parallel. In binary, the previous example would be changing from zero to a one or a one to a zero respectively. In order to write data to the MTJ a current must be supplied that meets or exceeds the critical write current. Since we have two equal and opposite MTJs, the actual amount of current needed is directly related to whichever of the two MTJs will need to be placed into an antiparallel orientation as it takes considerably more current to change to an antiparallel state. Once the MTJs have been overwritten and now represent their new bit of data, the supplied current drains to ground eliminating any unnecessary current left in the circuit.

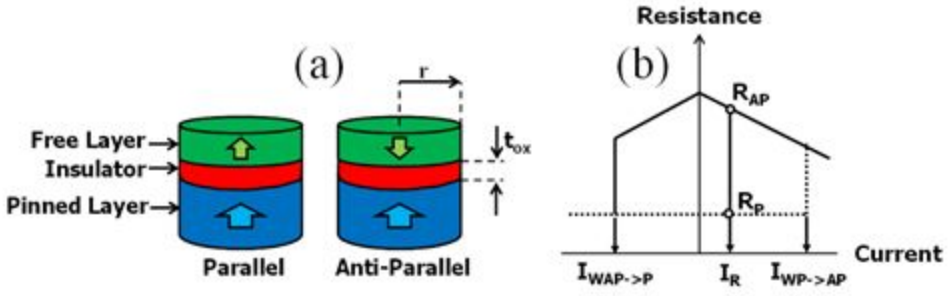

Figure 2. Perpendicular MTJ: (a) Parallel and Antiparallel states, (b) R-I characteristics

3 Proposed Design: Precharge Sensing Non-Volatile Latch

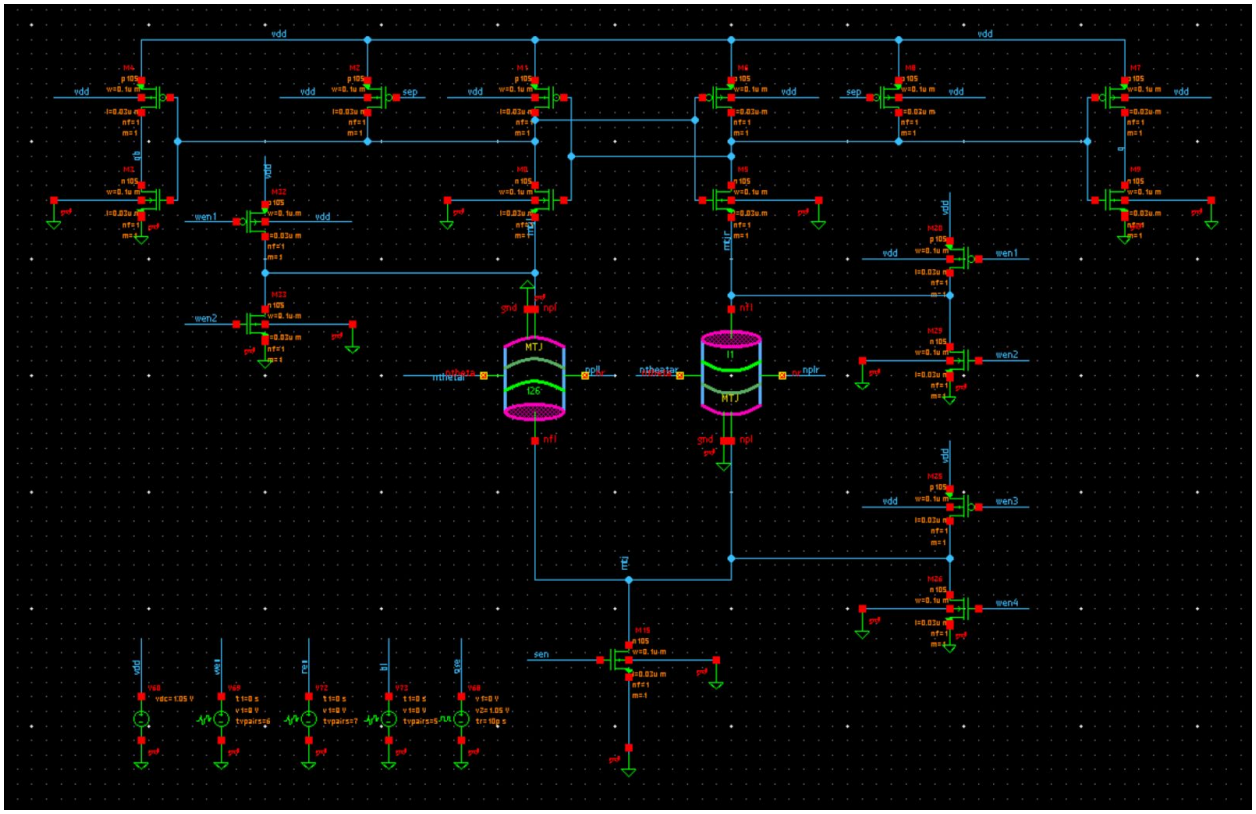

Figure 3. HSpice Circuit Diagram of Precharge Latch

3.1 Precharge Latch

A Precharge Latch is referred to as a resistive memory system and is non volatile by nature making it more stable than traditional RAM. In the Precharge Latch, as displayed above in *Figure 3*, we use the two MTJs as resistive memory cells to hold our data as binary one or zero. For read, there is the input, SEN, and two outputs for this circuit. The outputs are the voltage out referenced as Q for the left side of the circuit and Q' for the right side of the circuit. For write, there are four inputs; WEN1, WEN2, WEN3, and WEN4. There is also a constant VDD connected to all pmos in the circuit.

The three inverters that appear directly around the MTJs are the Write Enable, WEN, portion of the circuit. This portion of the circuit's job is to collect a signal from WE#, write enable input, where # ranges from one to four, and depending on the input signal determines whether VDD flows from the bottom up or from top down. The direction of VDD will determine which of the two MTJs will be parallel and antiparallel. Essentially, WEN controls the data written to the memory cell.

The bottommost transistor is the Sense Enable section, GSE, while the top most section comprising of ten transistors placed to be symmetric down the middle is the Sense Amplifier. Vdd enter and travels down the left and right side of the Sense Amplifier and travels through the transistors until it reaches the MTJs. At the MTJs it continues down through the MTJ with the least resistance, the one that will be in a parallel state. Once out of the MTJ the current will travel to ground through the GSE where the current will become zero. With the current becoming zero, the GSE side opposite to the MTJ that carried the initial current is shut closed preventing current flow. Finally, the zero current reaches the inverter at the top left/right and outputs VDD to the Q that is on the same side of symmetry as least resistive MTJ.

The main difficulty in using a latch (resistive) is that a small charge needs to be applied, but a dc current could create potential leakage. The fact that we will be continuously reading from this makes this risky, and that is where the Precharge Latch comes into play. In the Precharge Latch, a voltage accumulates in the read stage that ultimately is discharged. The inverters are flipped and then optimization comes into play for reliability. With a low delay, chances are that process variations will be absorbed better. Right after power-up, the sense enable signal (GSE) needs to switch from high to low, while write enable (WEN) is low.

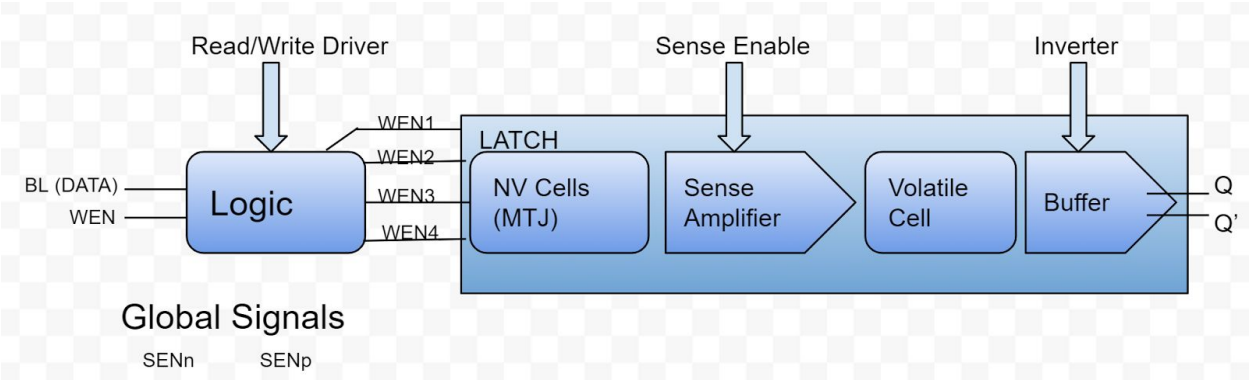

Figure 4. Block Diagram of Logic and Precharge Latch

3.2 Logic

The control logic is composed of two inverters and two nand gates, and it generates the signals to flip the MTJ. It generates the input signals to write and read, allowing for certain values in the truth table to be known. Depending on the direction of the current, values (either 0 or 1) are introduced for the Data and the Write Enable (1-4). From the truth table, we are able to get our "K-map" and get the equations that give us the results of the logic.

Bit-Line (Data)	**WEN**	WEN1	WEN2	WEN3	WEN4
0	**0**	1	0	1	0
0	**1**	1	1	0	0
1	**0**	1	0	1	0
1	**1**	0	0	1	1

Figure 5. Truth Table for Write Operation Logic

3.3 Write Operation

The write operation of the proposed circuit is an improved version of previous STTRAM circuits due to the use of logic to control gate voltages of the transistors. This circuit allows logic to control the gate voltages of WEN1 and WEN2, which then control the node voltage for the top portion of the circuit. Similarly, the gate voltages of WEN3 and WEN4 are also controlled by logic and those transistors determine the node voltage for the bottom portion of the circuit. The

voltage difference between the top and bottom portions of the circuit determines what direction the current will flow through both of the MTJs. This current flow will then dictate the orientation of the free layer of the MTJ and whether or not the MTJ will be in a parallel or antiparallel state. By using logic to control the gate voltages of the transistors, there is no need to have a bitline connected directly to the MTJ. When a bitline is connected directly to the MTJ current can only flow in one direction unless there are four inputs available to control the transistors. The design proposed by this paper only requires the use of two inputs, WEN and Bit-line (DATA), which enter the logic design and are converted into the four WEN signals that control the transistors.

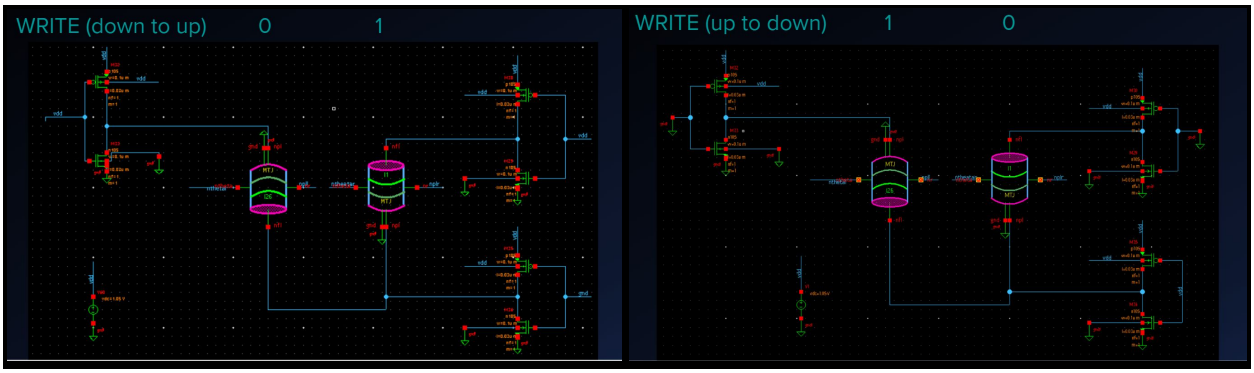

Figure 6. HSpice Circuit Diagram of the Two Isolated Write Operations

In order to write a 0 1 to the pair of MTJs, WEN1 and WEN2 are low while WEN3 is high. When WEN1 and WEN2 are low the top NMOS transistors are on and the top of the circuit is connected to ground while WEN3 allows the bottom PMOS to turn on and VDD is introduced at the bottom portion of the circuit. The potential introduced into the circuit allows current to flow from the bottom to the top, which is currently referred to as down to up. When writing 1 0 to the pair of MTJs, WEN1 and WEN2 are high, which allows VDD to be present at the top of the circuit. WEN3 is set to low which, allows the bottom of the circuit to be connected to ground and a potential created. This potential allows the current to flow from the top of the circuit to the bottom of the circuit and is referred to as up to down.

3.4 Read Operation

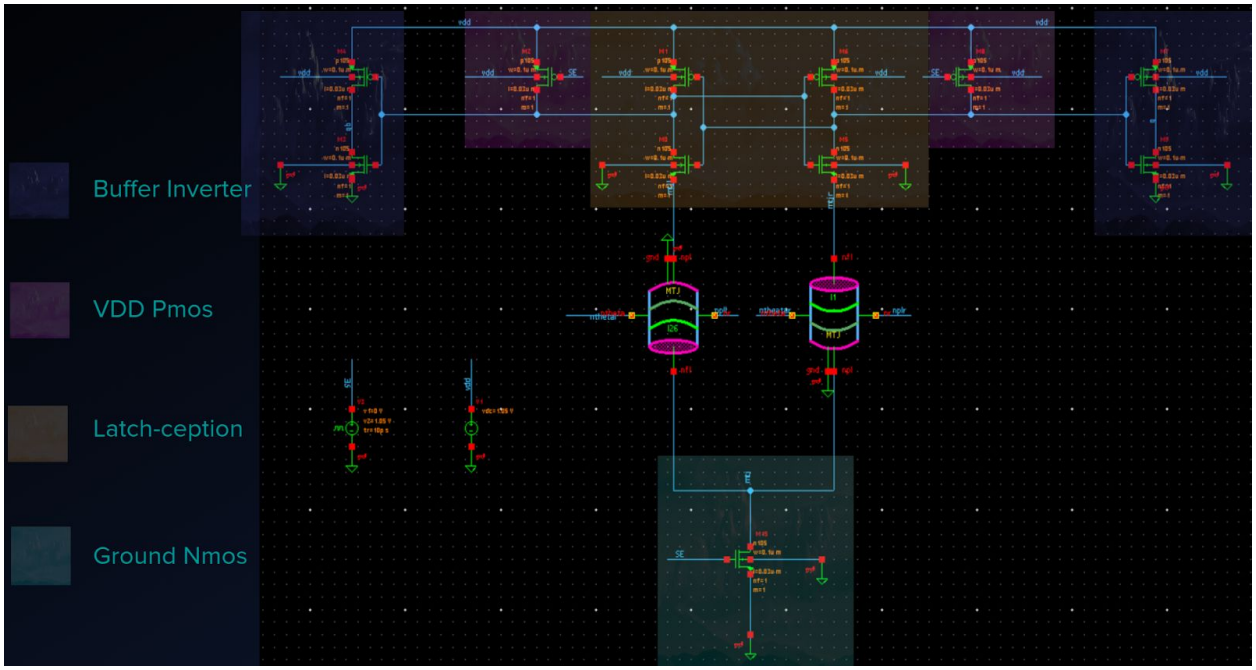

Figure 7. Hspice Circuit Diagram of Isolated Read Operation

The read operation consists of a Precharge phase and an evaluation phase. During the Precharge phase, sense enable is low, and the PMOS at the top of the circuit allows the circuit to be precharged with VDD while the NMOS that connects to ground is not active. The node voltage between both of the MTJs and their respective inverter will be VDD - VT. During the evaluation phase, sense enable is high and the VDD PMOS is inactive while the Grounding NMOS is active and allows the circuit to connect to ground. Once grounded the MTJ with the least amount of resistance will deplete the stored voltage before the other MTJ. This will allow the NMOS of the inverter to become connected to ground and will allow the inverter to invert the VDD that was stored during precharge in the portion of the circuit that contains the inverters. This VDD signal will be inverted to a 0, but will become a 1 after passing through the buffer Q. Since the circuit is differential, the side of the circuit that has the MTJ with the higher resistance will behave opposite to that of the side with the lower MTJ resistance. This behavior is how the read circuit will interpret the stored values of the MTJs.

4 Design Setup and Individual Transistor Simulations

4.1 Delay vs Width Sweeps

Because area and power consumption are key factors in designing integrated circuits and other devices, the non-volatile latch design used in this project needed to be optimized. To optimize the circuit, we needed to figure out how changing transistor widths affected the circuit and then

find the combination of transistors that produced the lowest total area with 100% reliability. The circuit was first created in the form of a schematic. This schematic served as a reference for the exported HSpice netlist of the circuit we made to alter features, add measurements, and run simulations.

Once the netlist was exported, we added lines of code to measure delay, area, MTJ resistance and MTJ orientation. Delay was measured differently for the Read and Write operations. For the Read operation, delay was measured as the time it took for Q and Q' to be updated after the Sense Enable signals were inputted into the circuit. For the Write operation, the delay was measured as the time it took for the MTJ to fully change orientation after the WEN# signals were inputted into the circuit from the logic. The time between inputting WEN and bit-line DATA and the logic generating the four WEN signals was not significant and not measured. For both the Write and Read operations, two delays were measured due to two possible MTJ orientations, which are 01 and 10. In simulations and testing, the maximum between these two was taken to serve as the delay for that run or simulation. From early testing runs of the circuit, we realized that one delay was higher than the other for the write operation measurements. This supports our theoretical knowledge of the MTJ needing more current to flip from a low resistance (0) to high resistance (1) than from high resistance to low resistance. Any delay over 15 nanoseconds was considered failure and was used throughout the project. Area was measured as the summation of the transistor widths in micrometers. MTJ resistance was measured in ohms and MTJ orientation was measured in radians. It was decided that power was not to be measured do to this design being already low power and this factor was not as significant in importance as the factors above.

Using the exported netlist, we ran simulations of the full circuit, not including the logic, with variable individual transistor widths. Our goal was to test how the adjustment of each transistor width value individually would affect the delay of the read and write operations. This testing allowed us to determine which transistors were most sensitive and which were not sensitive at all. Sensitivity was defined as the probability that a decrease in area will result in an increase in failure. To complete these simulations and get accurate data, we used the following procedure for each transistor in the circuit. We set all transistor values to the minimum size of 0.1 micrometers except for the one being tested. The transistor being tested was given a variable for the width and this width was "swept" so that the code would run thousands of simulations, one for each width generated within a specific range with a specific resolution. The range used was always from 0.1 micrometer minimum size to 10 micrometers. The resolution was usually .01 micrometers. From this data, we were able to determine exactly how delay related to each transistor's width and circuit failure. Linear delay vs width graphs represented our nonsensitive transistors. These graphs showed that delay increased linearly with increased area due to the capacitive effect of increasing the width of the transistor. From these graphs, it was clear that the lowest delay corresponded with the lowest area and thus these transistors were left at 0.1 micrometers for the rest of the project; they did not require optimization. The pmos Transistors located at input signal SE and transistors located at output buffers of Sense Amp were not sensitive. These were left at minimum size.

All of the remaining transistors in our simulations showed an exponentially decreasing graph of

delay vs width. This can be seen in Figure 8 below.

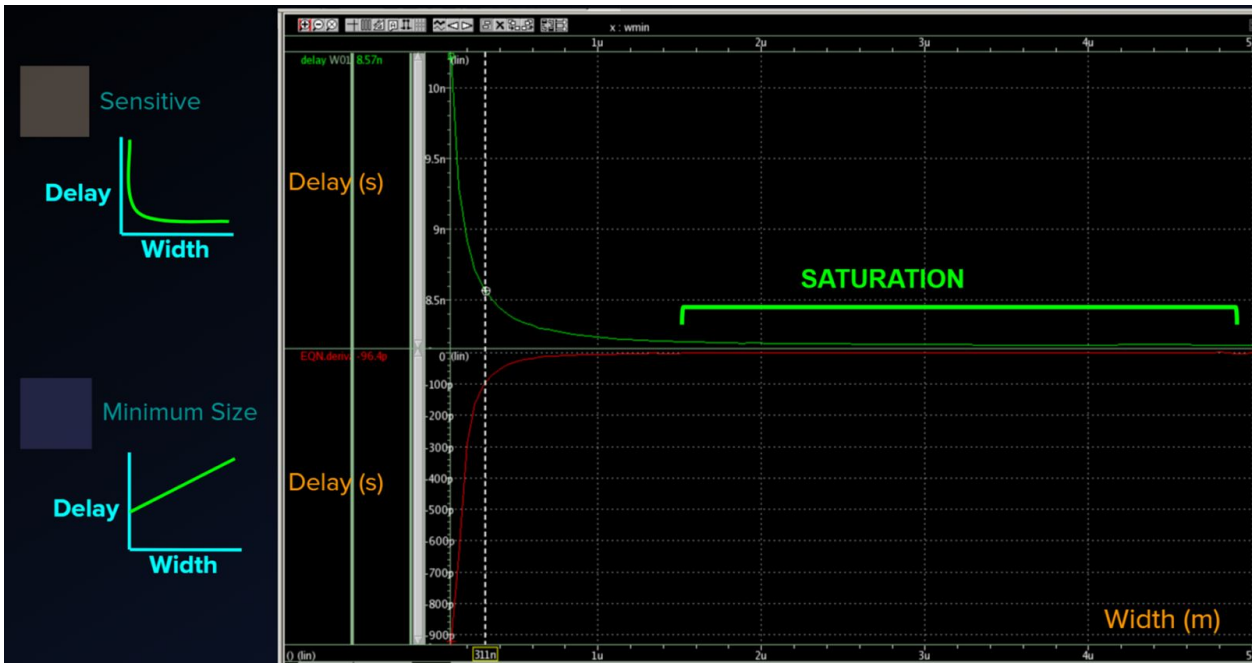

Figure 8. HSpice Delay vs Width Simulation Analysis Graph with Derivative Function

This graph represents the sensitive transistors in our circuit, which have a unique delay vs width relationship. For our sensitive transistors, delay was inversely related to width as an exponential function. At very low width, delay was very high and the circuit failed. As the width was increased, delay dropped at an exponentially decreasing rate. At a certain width that we determined, which varied for each transistor, the delay would no longer drop significantly, as less than 1 picosecond per nanometer. This width was named the saturation point of that transistor. Any value of width after the saturation point was an overdesign of the circuit; the increase in width did not decrease delay by a significant enough amount to be efficient or feasible from both design and manufacturing standpoints.

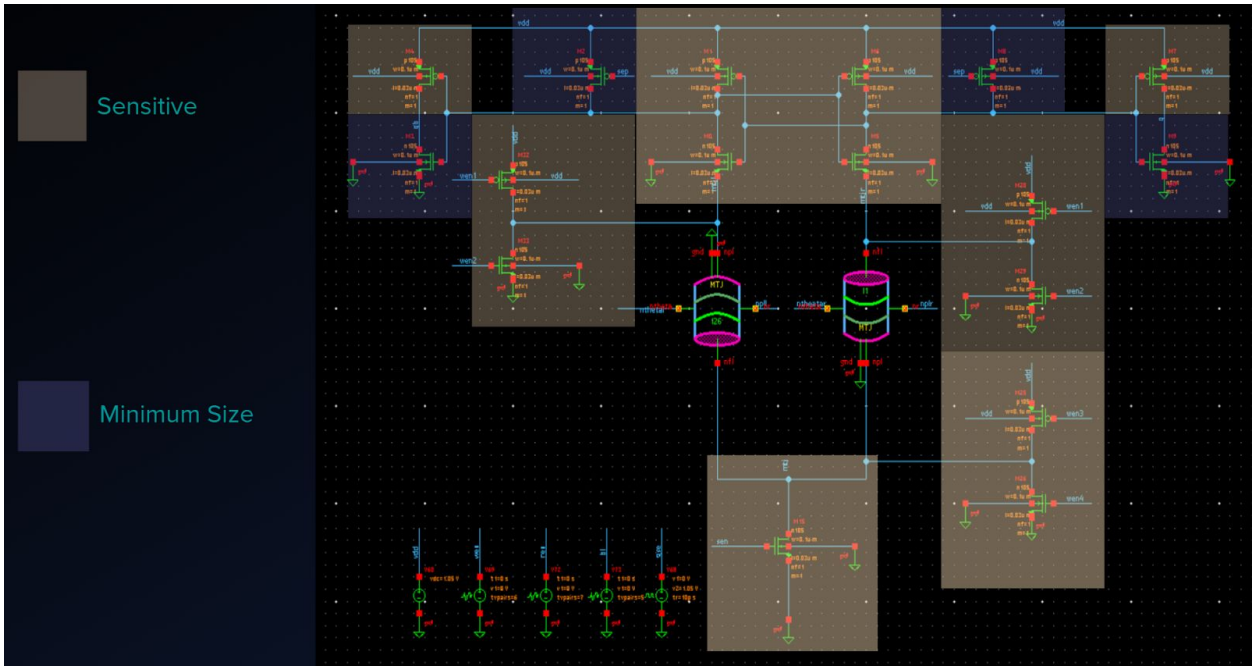

Figure 9. HSpice Circuit Diagram of Precharge Latch Sensitivity

Figure 9 is a visual representation of the full latch and the varying sensitivity of the transistors. The most sensitive transistors, and thus the most likely to be large, were the m0 and m5, m1 and m6, m15, m35, m36 transistors. The most sensitive transistors were the ones for which delay changed at the highest rate before reaching the saturation point. The data on the sensitive transistors gave us a more defined range of where the optimal width values of these transistors might be. The approximate range was the low width where the rate of decreasing delay matched the rate of increasing width to the high width value of the saturation point.

We also identified symmetry patterns. The symmetric transistors are those that make up the sides of the Sense Amp and sides of top Write cycle. These are symmetric because of the differential setup design of the entire latch.

5 Optimization of the Precharge Latch Write Cycle

5.1 Multivariable Simulations of Precharge Latch Write Cycle

The individual transistor simulations gave us a rough idea of which transistors to optimize and the starting point of the width ranges for each of them. Our next step was to run multivariable simulations to see how varying transistor values affected each other and to determine much more accurate optimal width value ranges. The multivariable simulations required much more processing power and time, so we decided to simplify the testing circuit in the netlist code as

much as possible. We were able to split up the circuit into two distinct parts; the Read and the Write operation. These two operations run separately and only slightly affect each other due to capacitive effects. The write cycle optimization was chosen to be done first because it was much larger and thus less sensitive to reliability. The Read operation was found to be more sensitive from our preliminary individual sweeping results and thus was chosen to be done last. Because of the lower sensitivity and higher area, the capacitive effects of the large Write cycle would affect the more sensitive Read cycle and thus we wanted to set the values for Write cycle before optimizing the Read cycle.

The multivariable simulation netlist we set up for the Write cycle has four variables. We only needed four variables to represent the six Write cycle transistors because the two Write inverters above the MTJs were symmetric. Once the netlist was complete, we ran simulations with the four transistor width variables TopNmos, TopPmos, SinkNmos, and SinkPmos. We quickly began to notice that the bottom transistors and top transistors of the Write cycle were dependent on each other. The m35 and m36 transistors were found to be more important/more sensitive than (above MTJ) transistors m32-m30 and m33-m29. When set at minimum size, the m35 and m365 transistors could not conduct enough current to change the orientation of the MTJs, which resulted in increasing delay in width sweeps of the (above MTJ "on") transistors. The minimum size of the bottom transistors correlated with failed delay. Failed is defined here as a state at which the the MTJs did not flip.

To further decrease the amount of simulations we had to run, we redesignated and redefined the variables giving us two width values and two ratios. The dependency we recorded earlier allowed us to use ratios instead of values and decreased the overall amount of combinations we would need to test for the Write cycle significantly. The new variable names were TopNmos, SinkNmos, Ratio01, and Ratio10. The two ratios are defined in the formulas below. They were used to determine the values of sinkPmos and sinkNmos.

Equation 1:

$$\textbf{Ratio10 * TopNmos = SinkPmos}$$

Equation 2:

$$\textbf{Ratio01 * SinkNmos = TopPmos}$$

Figure 10 below shows the thousands of combinations simulated to identify the relationships between the Write cycle transistors and determine the optimal width value combinations.

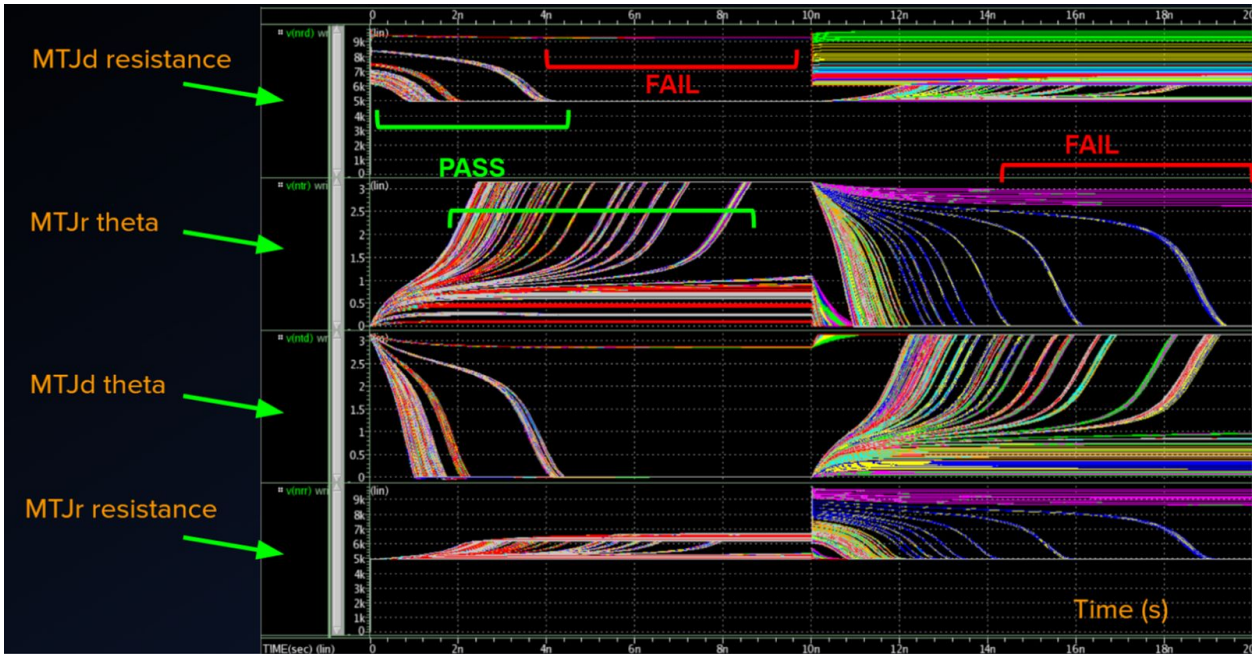

Figure 10. HSpice Graph of Write Operation Simulations, MTJ resistance and orientation

The sweeps outputted ~10,000 combinations, which were then sorted in MS Excel for further data analysis. The data analysis showed lowest delay to be no less than 2n. This also correlated with largest area; 30+ micrometers. The minimum area without failure was 10µ and all four of the transistors had minimum values well above .1µ. Our data showed failures for many values and combinations; failure was determined by either of the MTJs not flipping after 15n seconds. Data analysis also showed that in most good runs, topNmos was consistently the smallest value width (under 2µ), followed by topP (around 2.5µ), sinkN (around 3µ), and sinkP (around 4µ). Once the ranges were narrowed for each width, more sweeps were run with higher resolution. In addition, we noted that less than 3 nanosecond delay did not fail for any combinations. We made this our benchmark for analysis of good combinations. Our next task was to find the combination of transistors that would give us the least area while still operating at less than 3n delay in the write cycle. The optimal combinations from all of our recorded data was saved to be used in our next step as initial optimization parameters.

5.2 Delay and Area Optimization

To get the absolute best combination of width values for the four Write transistors, we used Star-HSpice optimization. This optimization method is based on incremental optimization that works to solve DC, AC, and transient parameters in sequential order. To set the optimization up, we took our Write cycle input netlist file and added a .MODEL statement in which we specified minimum and maximum parameter and component limits, variable parameters and components,

initial values, and the circuit performance goals. The optimization netlist automatically generated model parameters and component values from the initial values and electrical specifications given to it. The closer the initial values, the faster and more accurate the optimization.

Close width values from the multivariable simulations were inputted into the optimization netlist as the initial parameters and run with two specific performance goals to complete. The initial goals were less than 3n delay and less than 18µ area. Both initial goals were given the same "weight" of importance in the netlist to begin optimization. However, as we ran the program for our sets of combinations, we discovered that the program was optimizing for delay more than area. We adjusted the weight until the optimization would keep the delay just under 3 nanoseconds and not much less. With the weight on the optimization of area set at five times the weight of the delay goal, we proceeded to decrease the area goal while keeping the delay goal constant. This achieved the best results. We recorded the values we obtained from the optimization runs until we reached failure in optimization from decreasing the area goal to an impossibly small size. This is where we stopped the optimization. All of the values outputted by the optimization program were recorded for the final step; failure rate simulations.

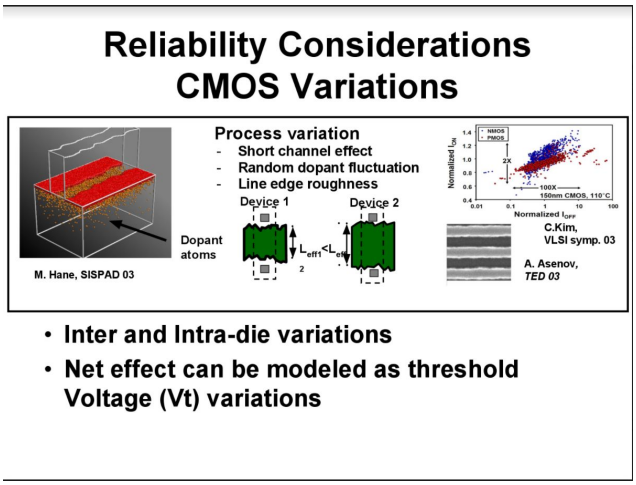

Figure 11. Reliability Considerations and CMOS Process Variations

The reason for the need for all of these optimization test and failure rate simulations comes from the reliability considerations we have made for possible variations in the cmos. On scales as small as we are working on small mistakes in manufacturing do occur and our tests try to account for these variation errors. The reliability considerations are outlined below in Figure 11.

5.3 Monte Carlo Failure Rate Simulations

A Monte Carlo failure rate netlist program was used to test each set of values recorded in the optimization process. This Monte Carlo program is a statistical simulation that utilizes Gaussian distribution to test the process variations that arise during manufacturing.

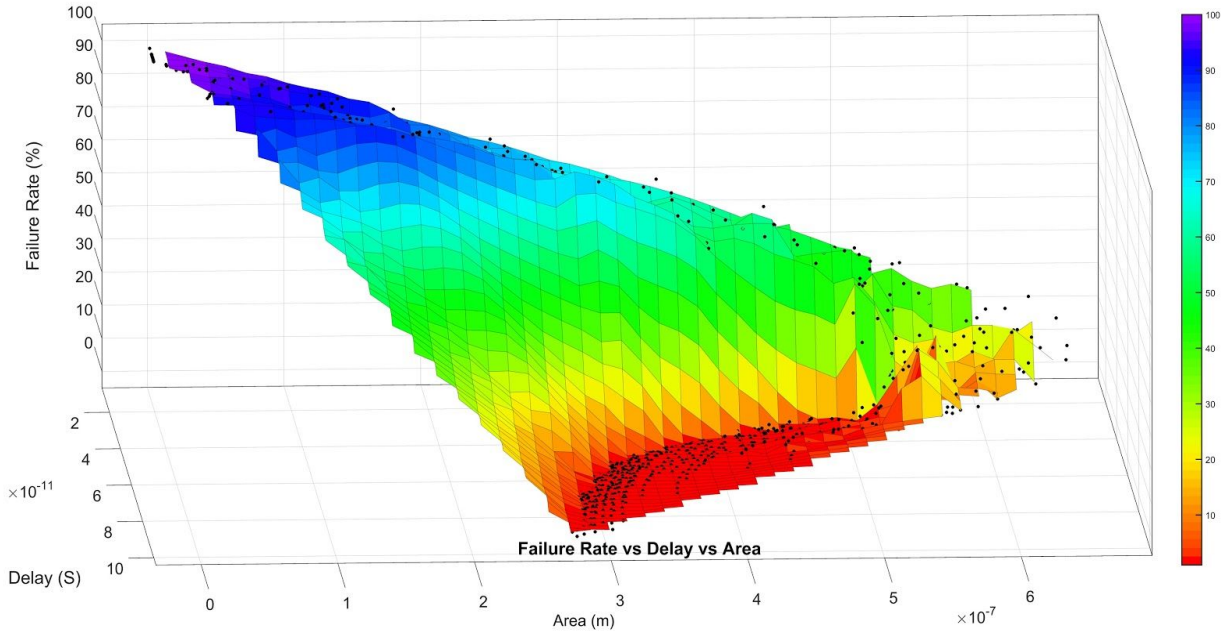

Figure 12. MATLAB Graph of Failure Rate vs Delay vs Area 3D View

The Monte Carlo can be set to run anywhere from 1 to 10000 iterations. Each set of our values was run at 1000 iterations in the Monte Carlo simulation to ensure reliability of the circuit to .1% accuracy. We looked for values that passed simulations with 0% failure rate, which meant at least 99.9% reliability at 1000 iteration runs. Above in Figure 12 is the three dimensional plot of our results from our Monte Carlo simulations. The top left purple on the graph represents the lowest area and lowest delay but also represents the highest percentage failure rate. Our desired result, represented in red in Figure 12, had a varied level of delay and area but had a consistent success rate of one hundred percent.

Table 1. Optimal Monte Carlo Results for 1000 and 100 iteration runs

1000 run	100 run
.alter	.alter
.param topN = 1.871	.param topN = 1.693
.param topP = 2.402	.param topP = 2.103
.param sinkP = 3.702	.param sinkP = 3.506
.param sinkN = 3.001	.param sinkN = 2.601
*area: 15.249	*area: 13.699
*delay: 2.9999	*delay: 3.2203

Our final results for the Write Operation are shown in Figure 13. The optimization for 100 run Monte Carlo yielded best values that had higher delay, lower reliability, and lower area than the optimization for 1000 run Monte Carlo. This means that for higher reliability of the circuit, delay would have to be made less and area would have to increase.

6 Optimization of the Precharge Latch Read Cycle

6.1 Multivariable Simulations of Precharge Latch Read Cycle

The read cycle consisted of 11 transistors and it was necessary to optimize each of them in order to obtain the smallest possible area for the latch. The previous testing of each individual transistor revealed that the VDD PMOS transistors in the sense amplifier did not need to be optimized and thus were left at the minimum size of 0.1 micrometers. Because of this and the symmetry of the left and right side of the sense amplifier, the total number of variables that were left to simulate dropped to four. We constructed the multivariable netlist for the Read operation simulations in the same way we constructed the Write operation netlist, reusing much of the code. We then ran a few hundred multivariable simulations and discovered that the buffer pmos transistors were also able to operate at the minimal size value of 0.1 micrometers. This fact allowed the optimization to be further simplified to three variables. In the next series of sweep simulations, we left the buffer pmos transistors, m3 and m9, at minimum size and only varied the widths of the remaining three transistors. This allowed us to increase our sweeping resolution to less than .1 micrometers. The four variables were named WN1, WN2, WP1, and WP2. WN1 represented the ground nmos m15 transistor, WN2 represented the symmetric inner latch nmos m0 and m5 transistors, WP1 represented the symmetric inner latch pmos m1 and m6 transistors, and WP2 represented the m4 and m7 output buffer transistors.

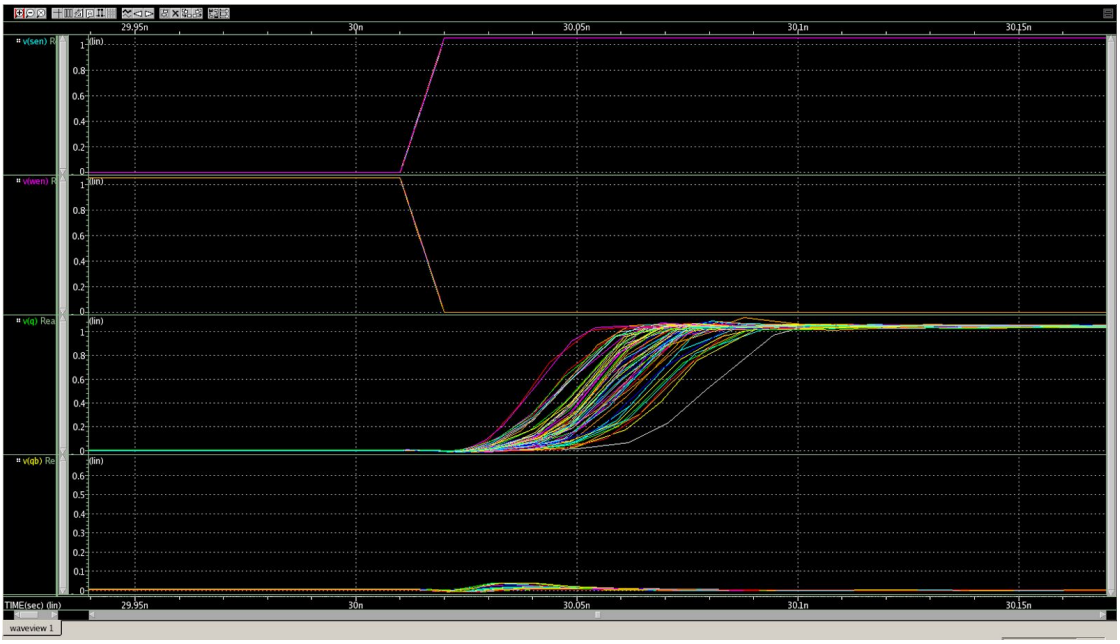

Figure 14. HSpice Graph of Delay in Read Operation Simulations

6.2 Delay and Area Optimization

Optimization of the read cycle was performed by using HSPICE and a netlist that allowed delay and area to be used as weighted goals. The netlist contained a parameter which allowed an area goal to be set so that HSpice would determine the best possible transistor sizes based on that area and weight. The initial weights used were 100 picoseconds for the delay constraint and 10 micrometers for the area for the area was set at around 50 and the resulting areas tended to be around 22µm with the largest area measured as 34.05 µm. Once the weight was changed to 100 the areas decreased sharply to around 4um with the smallest area measured as 2.52µm . In total there were about 70 sets of transistor values generated.

6.3 Monte Carlo Failure Rate Simulations

The Monte Carlo simulations were initially run with 100 simulations on each set of transistor values that were generated from the optimization netlist. If any of the transistor sets did not achieve 100% success, the data was recorded but the values were determined to be unusable for further simulations. The transistor values which reached 100% success were then run through the simulation again but with 1000 simulations. The majority of the transistor values failed to reach 100% success, but all of the remaining values achieved above 99.7% success rate, which meant our initial multivariable sweeps were accurate.

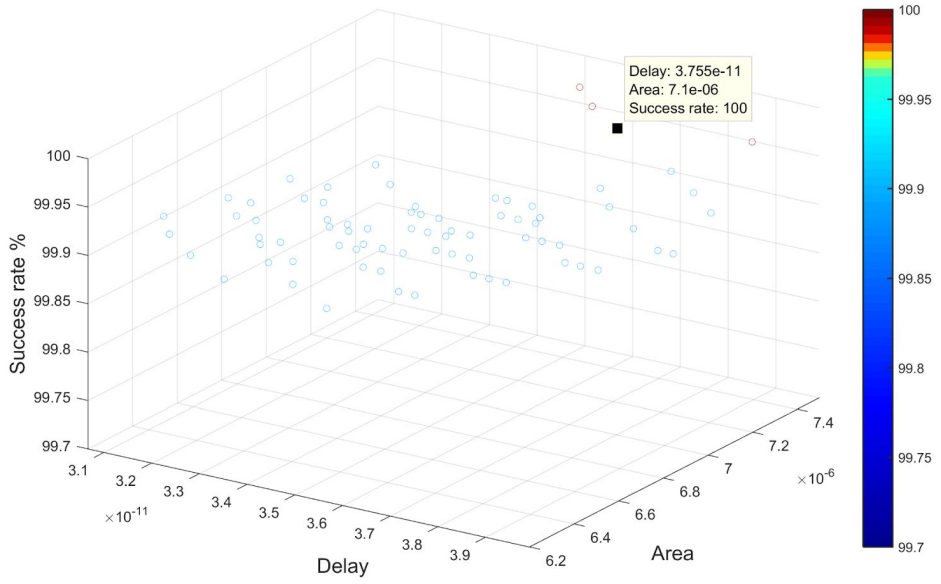

Figure 15. MATLAB Graph of Success Rate vs Delay vs Area 3D View for Read Operation

The smallest area for the 100 run was 4x0.03um2. We repeated this process for 1000 Monte Carlo and the smallest area was 7.1x0.03um2. this area passed 100% for 10000 Monte Carlo as well. The transistor widths in this set of values were 1.3 um, 1.5um, 1.5um, and 0.1 for WN1, WN2, WP1, and WP2 respectively.

7 Layout

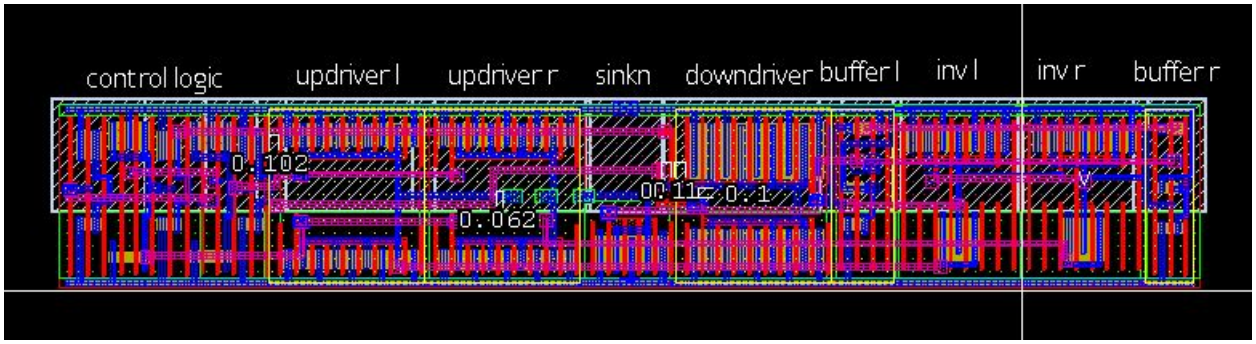

Figure 16. HSpice Full Latch Layout at Optimal Area

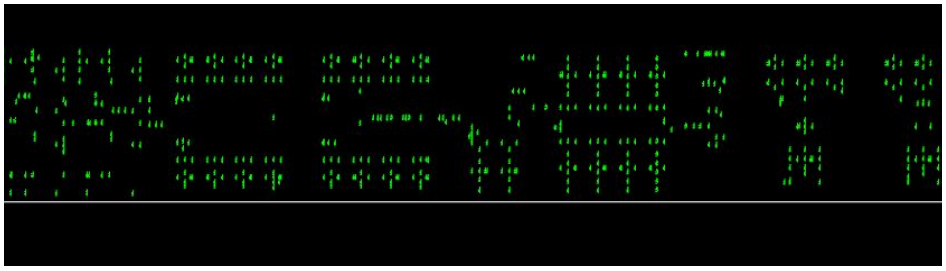

Figure 17. HSpice Full Latch Layout Parasitics

Once all of the transistors were optimized we were able to work on the physical layout of the circuit as it would be when printed into a chip, viewable above as Figure 16. The goal of the layout was to minimize the area the circuit would take up while still being able to perform function as desired. The layout was broken down into parts for ease of construction with each part comprising of, with exception of the *sinkn*, of a minimum of two transistors; at least one pmos and one nmos. In the above diagram *sinkn, bufferl, invl, invr,* and *bufferr* comprise the read section of the circuit while *updriverl, updriverr,* and *downdriver* are the write section of the circuit. The layout itself is comprised of several layers of materials. The base layers being the NWELL, white box with lines running through it, and the PIMP, red box with translucent red filling, for the pmos and NIMP, green box with translucent green filling, for the nmos. The top is the VDD, electrical power source, which is surrounded by NIMP. Conversely, at the bottom is the VSS or electrical drain that is surrounded by PIMP. The yellow is the diffusion layer that represent the body of the transistor. In the above design many of the transistors have multiple fingers, red poly layers, running through them which effectively treats them as multiple transistors. The reason we divided them was to decrease the overall width, represented here as the height, of the transistors. Connecting the different parts of the circuit are the wires

represented by the blue, pink, and green lines, meta one, meta two, and meta three respectively. Contacts are placed within the circuit, represented as white boxes with and x in it, that are the locations in the circuit for input and output. Higher level metas use VIA, which look the same but a different colour. All these different colours are placed to represent 3D layers due to the fact that wires can't be allowed to make unwanted connections with each other. With all of these materials we constructed the layout in portions.

The first test we run once our a circuit portion was constructed in the layout is the Design Rules Checking, DRC, which checks to make sure our design follows the most up to date rules and conventions used in manufacturing. This is where a lot of the troubleshooting for the layout will occur as this test will pick up on all design failures made during the construction of the layout. The next test run is the Layout Versus Schematic, LVS, which checks to make sure that the fabricated layout matches the original circuit that was being modeled. During this test it is important to make sure that there is nothing mislabeled; this is important because the label name of a node or wire can change when calling an instance of a previously constructed circuit element. Note, an 'instance' is when a prefabricated layout is called and used in a new layout. The instance counts as one single piece and cannot be edited on the surface level. To edit the instance would be editing the original file, which in turn would cause all called instance of that one layout piece to be altered as well. The final test run is the Layout Parameter Extraction, LPE, which calculates the electrical parasitics present in the circuit. The parasitics are important for getting final, and most accurate, values of the effectiveness and function of the circuit.

Finally, the entire circuit is assembled using instances of all of the previously generated pieces. For this, all of the nmos and pmos layers must overlap each other and the VDD and VSS must all be formed into one bar across all instances respectively. Finally, everything must be connected together by meta one and meta two layers to create one continuous circuit. Along the lines of this, new labels must be generated across the layout to match the final circuit that the layout is supposed to represent. To make certain that everything is running correctly DRC, LVS, and LPE are run again. The final constructed size of our circuit ended up being 1.672μm by 10.988μm, which resulted in a total area of 18.37um2. Figure 17 is the parasitics output by the LPE.

8 Post-Layout Results

With the layout complete and layout parasitics generated by the LPE, we inputted all of this data back into the hspice netlist of the full circuit to run further tests on it. We ran the write path of the full circuit with parasitics to measure our new write delay. Our unideal write delay was 3.5 nanoseconds. We then ran the read path of the circuit with parasitics to measure sensing delay and sensing power. Our sensing delay, measured as the time from when Sense Enable goes high to when Q and QB are updated, was 168 picoseconds. Our sensing power was 42.4009 microWatts. The SE frequency at which this power was measured was 250 megahertz (MHz). The full circuit was also run in standby/idle mode, which means no input signals were generated. This was done to measure the leakage power over one cycle. The leakage power over one cycle

was 1.5594 microWatts.

After the initial tests with the parasitics, we also ran a 1000 iteration monte carlo run to test the reliability of the circuit with parasitics. The newly introduced capacitance and resistance values of circuit components were not represented in the ideal simulations done previous to the layout and thus could have altered the reliability of our design. Our results for the 1000 iteration monte carlo runs were 100% pass rates for both the Write and Read paths.

9 Conclusion

Spin Transfer Torque Random Access Memory (STTRAM) has the potential of replacing existing memory technologies such as SRAM, DRAM, FRAM, and Flash. The purpose of this research was to optimize the design of a non-volatile (NV) Precharge Latch and to use the data from the optimization to show the correlation between delay and failure rate. To do this, each individual transistor in the latch was run with variable width values in HSpice to simulate the relationships between transistor width, delay, and failure rate. High area and high power consumption corresponded with low failure rate. Low delay corresponded with low failure rate and thus high reliability. The write operation was much larger than the Read operation and less sensitive, requiring a large write current. Optimal values from the individual simulations were taken to be used as initial conditions in optimization runs to determine the most accurate and reliable transistor widths possible. We subsequently took these width values and ran Monte Carlo failure rate simulations on them to make sure that the transistors functioned at different degrees of reliability. The entire write cycle was done first. The most sensitive transistors were the those located at the sink of the latch, below the MTJs, and had to be made larger. The top 4 transistors located above the MTJs were dependent upon the 2 sink transistors and specific ratios were established between them. The best transistor values for the write circuit were 1.871, 2.402, 3.702, and 3.001 micrometers for topN, topP, sinkP, and sinkN respectively. The highest delay for the circuit at these values was 2.9999 nanoseconds. These values were subsequently inputted into the circuit netlist as constants to now optimize the Read operation. The methodology for optimizing the Read operation was similar to the Write operation. Variable names were changed and the full circuit netlist was used to include the write transistors. The optimal transistor widths for the Read operation were 1.3, 1.5, 1.5, and 0.1 micrometers for WN1, WN2, WP1, and WP2 respectively. The delay for the Read operations was 3.755 picoseconds. The optimized values of the read path transistors were much smaller than those of the write path transistors. Small increases in the read area drastically improved reliability whereas larger increases in width were needed in the write path for similar improvements in reliability. This is also evidenced by the extremely low delay of the optimized Read operation compared to the optimized delay of the Write operation. Due to the delay in the read cycle being in the picosecond range, changes in delay did not result in significant changes in failure rate. This explains why the reliability of the circuit remained at 100% despite the multifold increase in Read path delay. Power consumption was decided to not be an important metric of optimization for us because the device was already

theoretically extremely power efficient. Furthermore, in our final tests of the optimized circuit with parasitics, we recorded a power leakage of only 1.5594 microWatts and a read path power consumption of only 42.4009 microWatts. This power leakage is less than leading active power consumption memory technologies. The focus of future research on this subject will most likely be on the improvement of the write path in terms of area, power consumption, and delay.

Acknowledgments

Thank you to Dr. Hamid Mahmoodi and Ali Attaran for mentoring us in this project with technical, aesthetic, and presentation guidance. Thank you to Synopsys for providing EDA licenses to NECRL and to DARPA for continued funding for this research. We are very grateful for this opportunity brought to us by Dr. Amelito Enriquez and the ASPIRES program.

Bibliography

[1] H. Mahmoodi, A. Attaran, T. Sheaves, “Design of a Non-Volatile Latch using Resistive Memory Technology”
[2] H. Mahmoodi, S. Srinivasan Lakshmipuram□, M. Aora, Y. Asgarieh, H. Homayoun, B. Lin and D. M. Tullsen “Resisitive Computation: A Critique.” IEEE COMPUTER ARCHITECTURE LETTERS, VOL. 13 NO.2, JULY-DECEMBER 2014
[3] W. Zhao, E. Belhaire and C. Chappert “Spin-MTJ based Non-Volatile Flip-Flop.” Proceedings of the 7th IEEE International Conference on Nanotechnology August 2-5 2007, Hong Kong
[4]Wicht, Bernhard, Thomas Nirschl, and Doris Schmitt-Landsiedel. "Yield and Speed Optimization of a Latch-Type Voltage Sense Amplifier." IEEE JOURNAL OF SOLID-STATE CIRCUITS , VOL.39 NO.07, JULY 2004