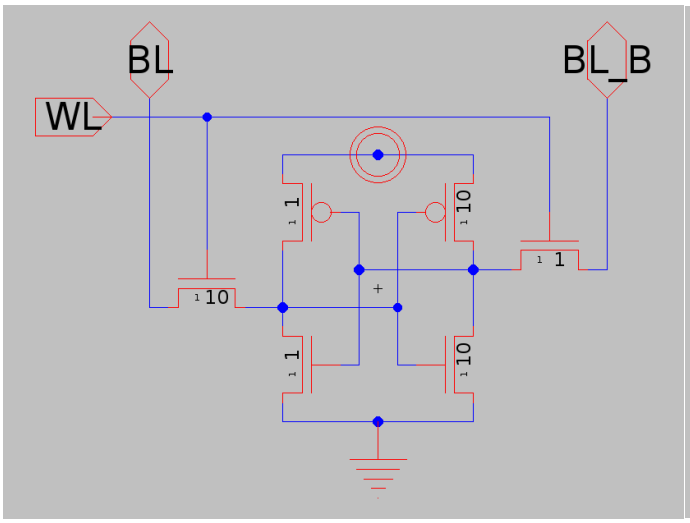
Neil Shah and Aasif Versi
ESE 370 – Project 2 Milestone
Design and Optimization of FIFO SRAM

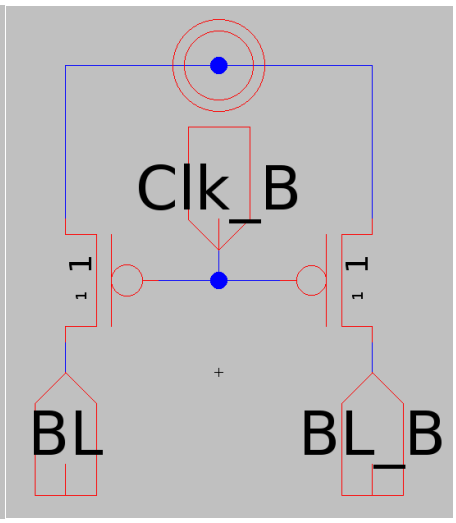
INTRODUCTION

Our goal is to build a 4 element long 4-bit FIFO SRAM using replicable 6 transistor SRAM (6TSRAM) cells to store individual bits of information in rows and columns in our overall memory layout. In this baseline implementation, we design, implement, and test the operation of a single 6TSRAM cell and column drivers for read and write operations. Most of our implementation is derived from minimal sized technology, with exceptions for the 6TSRAM and write drivers, which must be sized to ensure reads and writes will properly sample and overpower the latched bits, respectively. In our final implementation of the FIFO memory, we will be attempting to minimize the active energy when operated at as close to 500MHz as possible.

SCHEMATIC AND IMPLEMENTATION



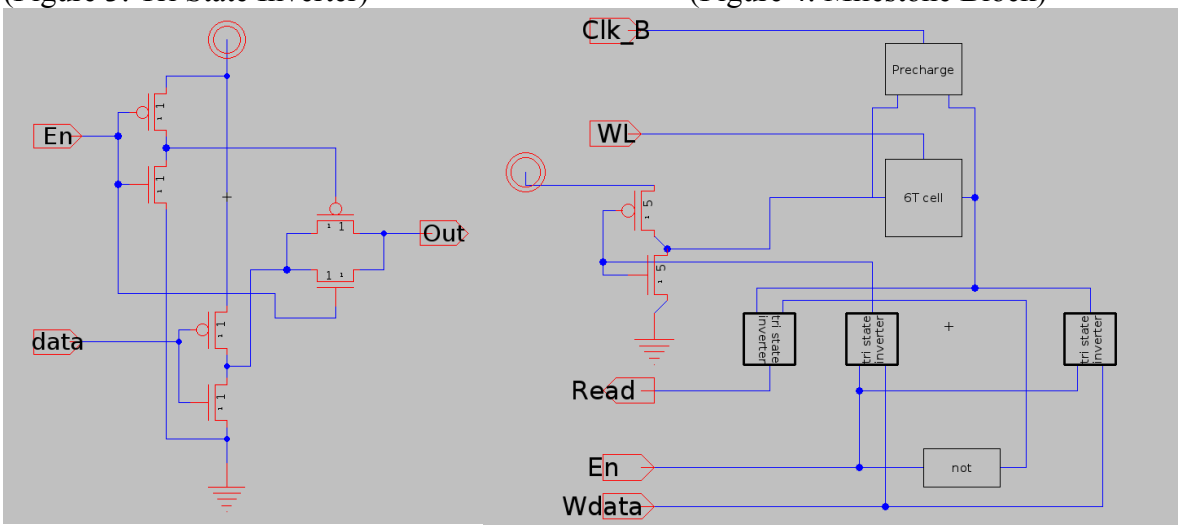
(Figure 1: 6TSRAM cell)



(Figure 2: Bit Line Precharging Circuit)

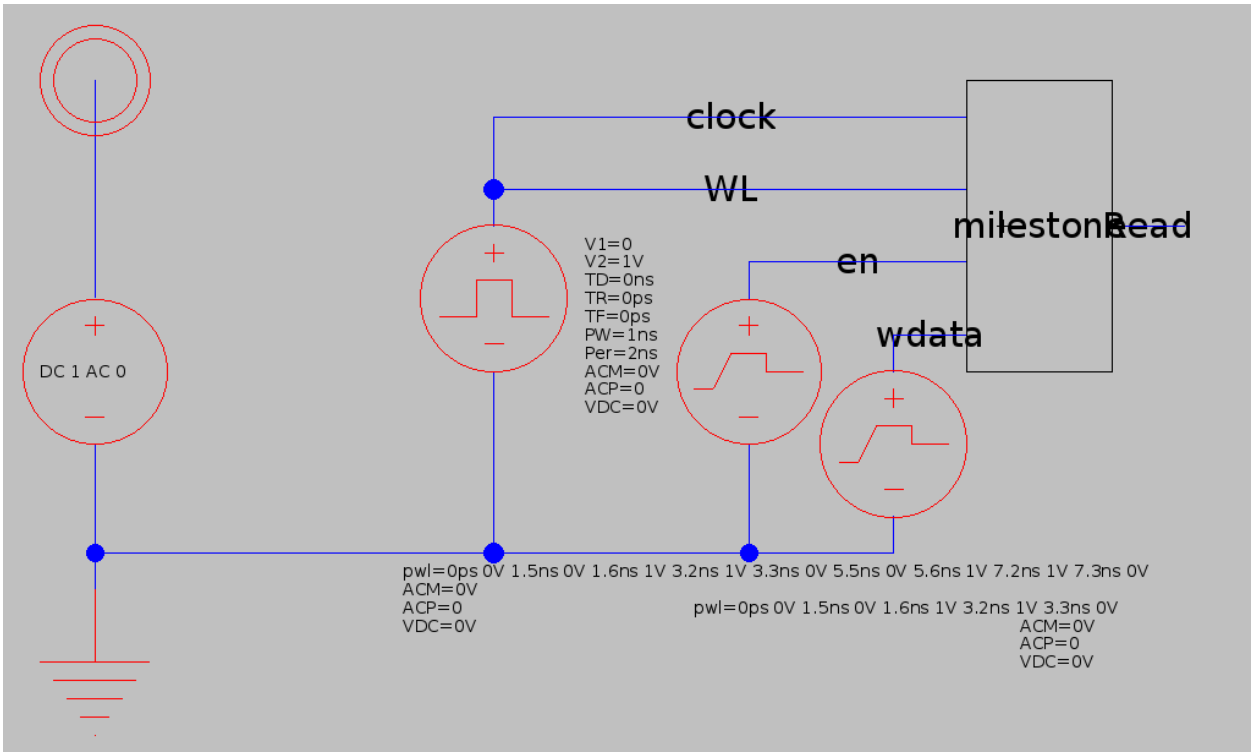
(Figure 3: Tri State Inverter)

(Figure 4: Milestone Block)



Our SRAM cell is level triggered, meaning read and write operations take place on the high clock level, and the pre-charging happens when the clock is low. We chose to use a tri state inverter in this configuration for the read and write drivers that, together with the Precharging circuit for Bit Line and Bit Line_B, make up the column drivers. We decided to implement the inverters with Pass Transistors in order to minimize the number of transistors that could switch. We sized the SRAM cell such that a read operation would not overpower the latched inverters to flip the stored value. The cell also had to be sized such that a write operation would overpower one of the latched inverters. Therefore, we decided to write on the Bit Line and read on the Bit Line_B. In order to make sure that a write operation would have sufficient drive strength, we added a large inverter to supply enough current to the cell that was being written. We also decided that since we were not using Bit Line to read the stored value, we did not need to connect a tri state inverter or use any sense amplification at this scale. Furthermore, since Bit Line_B was used primarily as a read line, we did not need to add a large inverter to power a write as we did for the Bit Line side.

EVALUATION OF CORRECT OPERATION

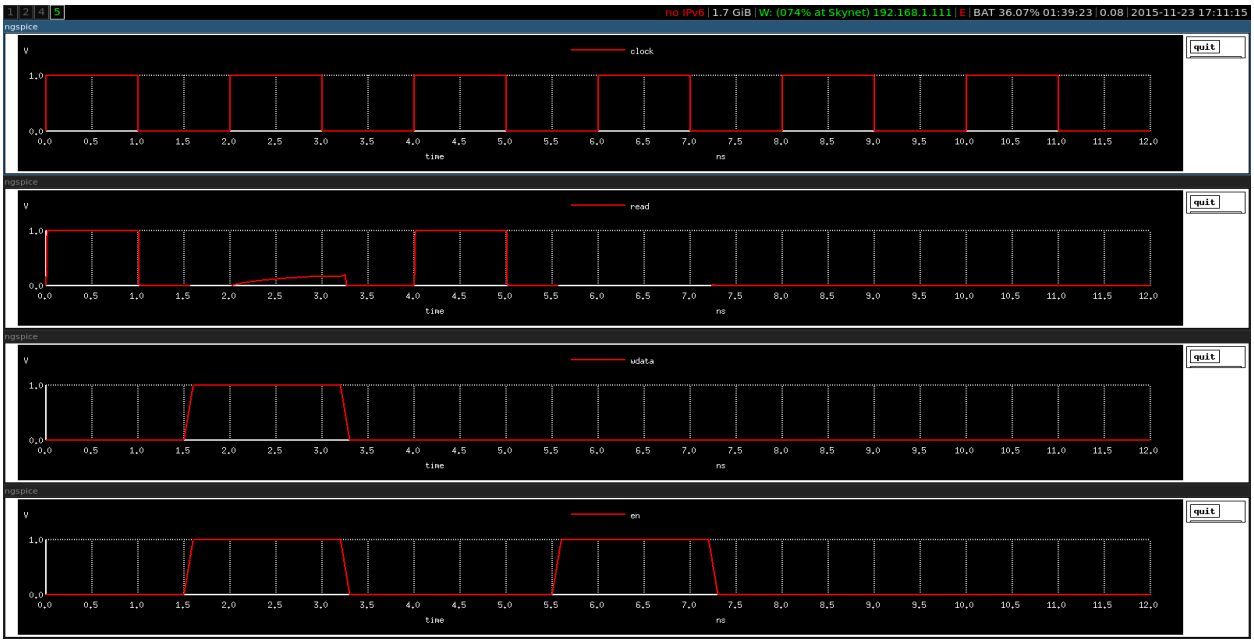


(Figure 5: Milestone Test Circuit)

In order to determine our 6TSRAM cell operates properly, we tried to write a 1 and 0 to the cell and read the values after each write in the simulation in Figure 6. We also set the Word Line of the cell to the clock signal, so on every clock cycle, we either read or write, depending on the value of En, which is the Write Enable signal. We can see in Figure 6 that after we write a 1 on the second clock cycle, we read a value of 1 on the third clock cycle, and after we write a 0 on the fourth clock cycle, we read a value of 0 on the fifth and sixth clock cycles. Since after this

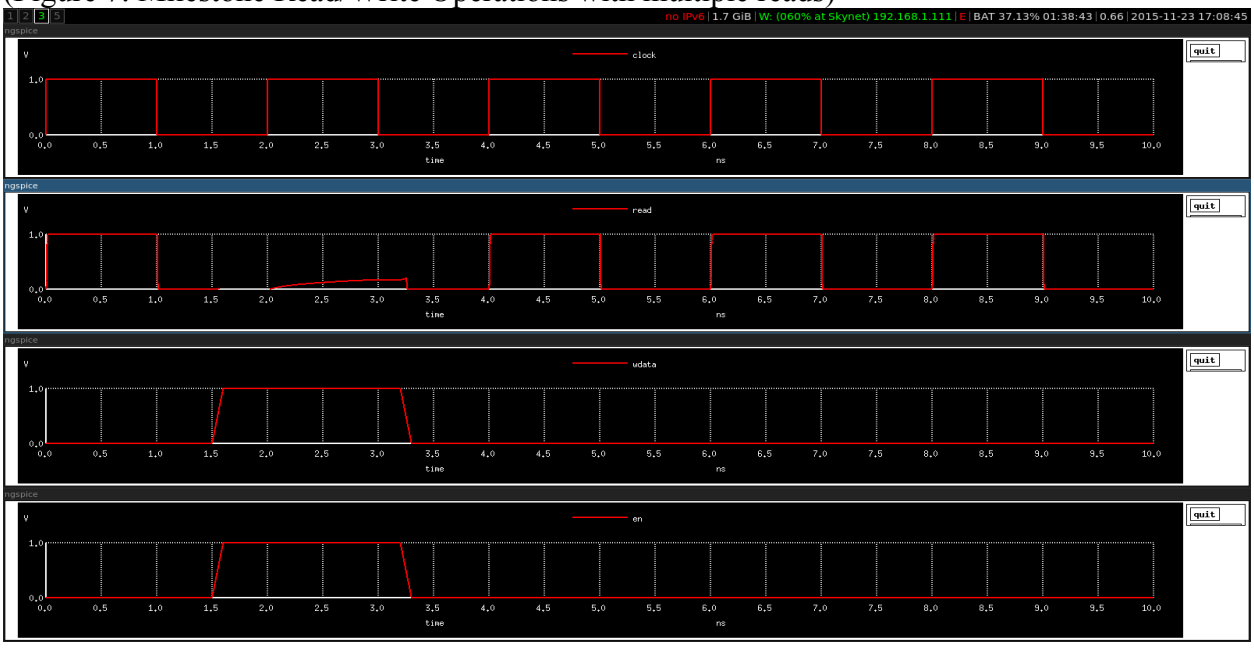
simulation we can see that reading a 0 does not flip the value of the cell, we needed to make sure that we can read a 1 from the cell without flipping its latched value. Therefore, we ran the simulation in Figure 7, where we wrote a 1 in the second clock cycle and read 1's in the third through fifth clock cycles afterwards, ensuring that we will never alter the stored value of the single bit memory cell by reading its value.

Regarding the "glitches" in the Read output on the first clock cycle of both Figures 6 and 7, we know that reading this value was due to the initial condition of the "Q" value of the 6TSRAM cell before all nodes of the entire milestone circuit have been set to their proper values.



(Figure 6: Milestone Read/Write Operations)

(Figure 7: Milestone Read/Write Operations with multiple reads)



DESIGN METRICS

6TSRAM CELL AREA:
Since the 6TSRAM cell is intended to be a highly replicable memory cell, we can calculate its area to determine a rough estimate of the area of the full memory block, so we can determine optimal memory placements when we layout our full CPU. Since SRAM cells likely would be used in several different subsystems, such as a register file, caches, buffers, and queues, being able to know how the memory cells would scale with the number of elements to store would allow us to be more effective in placing our memories. As such, when we look at the 6TSRAM schematic for the baseline, we can see that we would have approximately 33 area units based on the current sizing. Consequently, for the full implementation of the FIFO queue, we would expect to have roughly 500 area units for the 4x4 memory cells if we kept the same sizing. Since we will likely change the value of Vdd in the full implementation, we would expect to see changes in the sizing in order to minimize the active energy.

BIT LINE CAPACITANCE:
Bit Line and Bit Line_B are both connected to a full column of 6STRAM cells, so we can determine that the capacitance contributed from the repeatable memory cells is $80\gamma_0 C_0$ and $8\gamma_0 C_0$ for Bit Line, and Bit Line_B, respectively, from the diffusion capacitances from the pass transistors controlled by the Word Line. Both lines are connected to a precharging circuit, which contributes $1\gamma_0 C_0$ to both lines. Bit Line is connected to a diving write inverter that contributes $10\gamma_0 C_0$ from the size-5 inverter. Bit Line_B is the only line where the line is connected directly to the tri state inverters for the reading and writing circuits, since reading on the Bit Line is not required for this setup. Therefore, the two tri state inverters add $(2 + 2\gamma_0)C_0$ of capacitance to Bit Line_B. Therefore, the total capacitance of Bit Line will be $81\gamma_0 C_0$ and the total capacitance of Bit Line_B will be $(2 + 11\gamma_0)C_0$. Therefore, since we use the Bit Line as the primary write driver, and we use the Bit Line_B as the primary read line, we know that a read operation should be much faster than the write operation due to the significantly smaller Bit Line_B capacitance. (Note: these calculations are based on the assumption that $C_{0,n} = C_{0,p}$.)

TIMING CONSTRAINTS

In order to find the 'most limiting' timing constraint of the full memory, we need to find the critical path of a read or write operation. Since we know that a write operation should be much slower than a read operation, as noted in the Bit Line Capacitance analysis above, we need to find the critical path of a write operation. Therefore, the critical path will be from the control logic, which takes in the Enqueue command, through the logic that determines which Word Line to pull-up, and then the pass transistors to overwrite the previous stored value. Since we want this to be the limiting path, we want our column driver circuitry to be very fast, so we don't have to consider the delay on the Bit Line, since its capacitance is very large. Additionally, to make sure we can potentially read and write from a single cell in a single clock cycle, we need to make sure the write operation can take place within half of a clock period. Therefore, to evaluate the correct write operation, we need to try to enqueue and dequeue a value in one clock cycle when the queue is already full, and make sure that the dequeued value is what was expected, and that on a subsequent dequeue, we get the correct value out.

ACADEMIC INTEGRITY

We, Neil Shah and Aasif Versi, certify that we have complied with the University of Pennsylvania's Code of Academic Integrity in completing this project.