High Speed Cache SRAM

Alex Clunan, Korban Thepsoumane
ECE 4332
University of Virginia
<wat5ba xwu9um>@virgnia.edu
April 29, 2025

I- ABSTRACT

This paper demonstrates our design and verification of a 64kb high-speed cache SRAM design using the FreePDK 45 nm process. Our design is optimized for the metric shown in (1).

$$(ActiveEnergyPerAccess) * Delay^2 * Area * IdlePower \quad (1)$$

To achieve a successful working design that produces the best metric, we simulated different widths for the bitcell, compared sense amplifiers, tested different decoders, found the best block array sizing, and evaluated different wordline driver configurations. Using these simulations, we developed a fully functioning SRAM that produces $Metric = 6.403 * 10^{-31}$. To verify the robustness of our design, we conducted PVT simulations across all process corners, voltage levels of $VDD \pm 0.1V$, and temperatures of 0 C - 50 C.

II- INTRODUCTION

This document showcases our group's final design for a high-speed 64 kb cache SRAM. We scaled up our prototype model, and built a working 64 kb 32-bit word cache. We accomplished many of our design goals and simulated many different aspects of our design, optimizing for the metric shown in Eq. 1. The following section will discuss our SRAM design architecture, followed by simulation verification, and finally the metric used to evaluate the performance of our final design.

III- DESIGN ARCHITECTURE

III-A Inverter/Gates

All logic/non-driving gates are sized for a delay equivalent to one inverter. The inverter sizing is based on when Vm = VDD/2 on the VTC of the CMOS inverter. To find this ideal sizing, the NMOS was set to the minimum width (90 nm), and the width of the PMOS was swept between 90 nm and 280 nm using 10nm steps as shown in Fig. 2.

The ideal switching threshold voltage (Vm) occurs when the PMOS width is between 140 and 150 nm. After finding the suitable range of widths, we swept with narrower 1 nm steps to obtain the exact value of Vm. Fig. 3 shows that a PMOS width of 144 nm results in a Vm of 550.4 mV which is the closest value to VDD/2 = 550 mV. Since Cadence only permits transistor width changes in steps of 2.5 nm, the PMOS transistor width is set to 145 nm. As a result, final sizing is Wp = 145 nm and Wn = 90 nm.

The sizing of the NAND and NOR gates are based on this inverter sizing. The width is scaled by the number of transistors in series, and nothing changes if the transistors are in parallel. For example, a 2 input NAND gate has 2 PMOSs in parallel sized at Wp = 145 nm and 2 series NMOSs sized at Wn = 180 nm.

III-B 6T Bitcell

The Bitcell is sized based on comparisons between many combinations of access transistor widths (Wa) and pull-down transistor widths (Wn). The pull-up transistor width (Wp) is set at 90 nm due to the pull up ratio needing to be minimized and desire to minimize area/leakage. Fig, 5 shows the model used to simulate bitcell sizing.

Each combination of Wa and Wn was tested to find the lowest period until a read failure occurred. This gave the minimum period that each specific size could operate, providing a delay metric. The leakage current was also measured pull up transistors, with leakage squared used to represent Idle Power. Using these values, assuming that Active energy per access remains constant, and using the summation of bitcell widths as an analogue to area, all the values for the design metric were collected. The metric used to evaluate the bitcell is shown in Eq. 2.

$$Metric = I_{leakage}^2 * Delay^2 * W_{total} \quad (2)$$

Fig. III-B shows the metric (bubble size) vs. Wa and Wn.

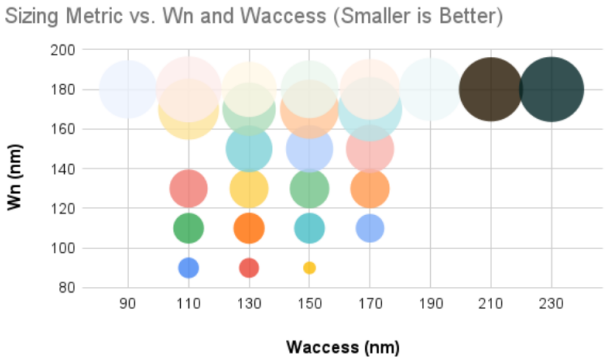

Fig. III-B Bitcell Sizing Metric

This shows that setting Wn to 90 nm provides the best metric performance. Based on only the metric, widths of Wa = 150 nm and Wn = 90nm provide the best performance. However, all Wa values from 112.5 nm and above generate read butterfly plots that are not bistable across all process corners. This leads to the final bitcell sizing as:

$$W_p = 90nm \quad (3)$$

$$W_n = 90nm \tag{4}$$

$$W_{access} = 110nm \tag{5}$$

Using this sizing, Read and hold butterfly plots of the bitcell were generated across all process corners. Fig. 7 shows the read butterfly plots and Fig. 6 shows the hold butterfly plots.

The hold plots show expected bistable behavior across all process corners with a minimum SNM of about 250 mV.

The read plots show bistable behavior across all process corners with a minimum SNM of about 50 mV. This represents the best case delay as there is minimal RC delay from the WL and BL due to the bitcell's proximity to the decoder and write NMOSs. shows the same scenario, but for the worst case delay bitcell. This bitcell is located furthest from the decoder and write NMOSs. This shows the correct read and write operation and verifies that our design works.

III-C Sense Amplifier

Our group decided to choose between two sense amplifiers: the 9T strong arm sense amplifier, and a 13T sense amplifier [1]. Fig. 16 shows the model used to evaluate both sense amplifiers. Using this model, we compared the delay of the two sense amps. and Fig. 17 and Fig. 18 show the maximum read speed, respectively, for each sense amp.

These plots show that the minimum delay of the strong arm sense amp is 24 ps while the minimum delay of the 13T sense amp is 668 ps when using the same setup. The strong arm sense amp provides a 28x boost in speed compared to the 13T sense amp. Using only this delay as a metric, we choose to use the strong arm sense amp in our design.

III-D Decoders

We tested 3 different types of 6-64 decoders: a NAND pre-decoded decoder, a NOR pre-decoded decoder, and AND Gate decoder. We compared them using the model shown in Fig. 14 to test the worst-case delays and calculated an area analogue by summing up all the widths in each decoder. A transient simulation showing the worst-case delays in the decoders model is shown in Fig. 15.

This shows that all the decoders have roughly the same delay, with the NOR pre-decoded decoder slightly faster than the other decoders. Summing up the widths results in a NOR pre-decoded decoder width of 211.645 um, NAND pre-decoded decoder width of 246.065 um, and AND Gate decoder width of 385.945 um. Based on the area and delay metric, we choose to use the 6-64 NOR pre-decoded decoder.

III-E Wordline Driver

To drive the wordlines, we used a NOR gate based driver architecture shown in Fig. 9 that has the capability to toggle between VDD (1.1V) and an overvolted VDDA (1.25V). We sized the driver transistors by sweeping the wordline driver widths according to the following metric and found the minimum period of correct read/write operations for our SRAM model.

$$Metric = Width^2 * MinimumPeriod^2 \tag{6}$$

Fig. 11 shows the results of our simulations, to which the smaller circle indicated a more effective width. From the simulations we determined that the most optimal widths were 360nM for the NMOS and 1160nM for the PMOS.

III-F Overvolted Wordline

We also used an overvolted wordline to assert the wordline faster during write and read operations. Preliminary testing done with a single higher VDDA during array sizing yielded a potential significant speed boost to each cycle. To find the most optimal overvolted wordline voltage we swept the VDDA voltage from 1.0V to 2.25V and found the minimum period for the model. Then we found the energy consumption of the setup tested and evaluated the efficacy using a metric based on the following metric.

$$Metric = MinimumPeriod^2 * PowerConsumption \tag{7}$$

Our results from the wordline overvolting simulations can be found in Fig. 7 We found that 1.25V was the most optimal VDDA for the wordline and noticed that any voltage above that resulted in short circuit current. Additionally we decided to only overvolt the last 2 columns of the SRAM array at 1.25V while keeping the rest at the normal VDD voltage because we were able to keep the same delay while reducing the power consumption (as if all of the wordlines were overvolted).

III-G SRAM Array Sizing

To find the best size of the SRAM array, we found the minimum period that each combination of rows and columns could support. To calculate the size of the array, we assumed that the number of SRAM blocks = 4. Given that the capacity = 64 kb, the number of rows and columns per block should multiply to 16384. Fig. 19 shows the model used to test each combination of rows and columns.

Using the model, we swept the array sizes and found the minimum allowable period with wordlines driven to VDD and to $VDD + 0.2V$. shows the delay vs. the number of rows. Fig. 8 shows the comparison of delay vs. number of rows/columns.

Based on this analysis, we set the number of rows to 64, and the number of columns to 256.

IV- SRAM Full Schematic

IV-A Top Level Layout

The top level layout of our SRAM includes all controls for determining the state of the SRAM. It contains a decoder for selecting the what block to access, NAND gates for controlling the read, write, and wordline enables, a mux for selecting between the outputs of the the blocks, and a negative edge register that holds the Dout<31:0> output. There are also three different clocks used in the design. One is the original clock which is connected to the wordline logic, another is a delayed clock, and the third is a clocked read signal that uses an inverted clock as the input. All are generated from the same clock input. The clocking is set up this way to allow the the sense amplifiers to read before the negative edge of the delayed clock. The output register is triggered at the negative edge of the delayed clock, so the read data needs to be ready before the negative edge of the delayed clock. A typical read follows: the main

clock triggers the wordline of the selected block -> the wordline powers up to VDD -> a bitline differential starts building up -> the clocked read signal triggers the sense amplifiers -> the sense amplifiers output to the output register -> the delayed clock activates the output register at its negative edge, and holds the output of the sense amplifiers. The precharge is also activated when the delayed clock is low.

The delayed clock is generated by 5 inverters in series that have $Wn = Wp = 90nm$, and a length of 175 nm. These output to a sized inverter which is the output of the delayed clock. This setup allows just enough delay to work at all process corners while minimizing read time.

IV-B Block Layout

One SRAM block takes in 6 different signals: ADDR<8:0>, Din<31:0>, nRen, nWLen, nWen, and precharge. ADDR<5:0> are the row addresses, and ADDR<8:6> are the column addresses. The row addresses are deciphered with a decoder which is enabled by the nWLen signal. Wordline overvolting is enabled when both ADDR<7> and ADDR<8> are high. This ensures that overvolting only happens when the accessed bitcells are the farthest two words from the decoder. There are 64 rows and 256 columns per block. 32 columns make one word, and each column of words has its own precharge and write circuitry. nRen and nWen signals are processed by a 3-8 decoder which determines the correct column address. These enable read and writes on a per word basis. There are 256 sense amplifiers per block, and all sense amplifiers in the same word are triggered by the same read enable signal. The sense amplifiers output to a series of muxes that select the correct column address, and output Dout. Each word has its own set of write drivers with 64 write drivers per word. The precharging is controlled by the delayed clock signal which passes through a series of inverters before reaching the precharge PMOSs. One whole block contains 16384 bitcells which are controlled by the circuitry stated above.

V- MODEL SIMULATION

V-A Block Diagram

This section shows the top level and block diagram of our SRAM, as well as the current status of each section. Fig 12 shows the block diagram of our current SRAM block.

The full SRAM schematic was generated by creating a top level module that generates a read enable, write enable, decoder enable, and precharge signal for each block and passes in the address and Data In signals to each block. Our model is a stripped down version of this, with all bit widths set to 1 instead of 32, all non-edge bitcells removed, and 3/4 SRAM blocks eliminated. Our SRAM block model is shown in Fig 13, and the top-level figure is the same as in Fig 12 with the two middle blocks removed and all data bit widths set to 1.

Using this model, we simulated accessing the best case and worst case delay bitcells. Fig. 20 shows a write of 1 followed by a read, followed by a write of 0, followed by another read. The precharge is activated when the clock is low, and read/write actions occur when the clock is high. This shows that the SRAM model correctly writes and reads a logic value of 1 during the first half of the simulation, and it also correctly writes and reads a logic value of 0 during the second half of the simulation.

V-B Pi Model

We used the following values to represent the wire resistances and capacitance for the entire length of the wordline and bitlines. These values were calculated using the width of our SRAM array and the 45nmPDK "metal1" values provided by the teaching assistant FAQs sheet.

TABLE I
PI MODEL PARAMETER VALUES

Parameter	Value
Bitline Capacitance	34.27 fF
Wordline Capacitance	137.1 fF
Bitline Resistance	104.2 Ω
Wordline Resistance	416.9 Ω

V-C PVT Variation

The 1b model was tested across all process corners at $T = 0C$ and $T = 50C$ and $VDD = 1.1 \pm 0.1V$ at its max frequency of 625 MHz. Fig. 20 shows the 32b SRAM model operating at max frequency at the TT process corner with $VDD = 1.1V$ and $T = 27C$. The model was also tested with a clock period of 2 us to check if the model worked at lower speeds.

VI- METRIC EVALUATION

PiCo evaluated the effectiveness of our design using the following metric formula:

$$(EnergyPerAccess) * Delay^2 * TotalArea * IdlePower \quad (8)$$

TABLE II
METRIC SUMMARY

Delivery Item	Description
Metric	6.403×10^{-31} J · s^2 · mm · W
Total "area"	27.5307 mm
Read energy	1.384×10^{-12} J
Write energy	1.40×10^{-12} J
Total energy	1.386×10^{-12} J
Read delay	0.54 ns
Write delay	0.36 ns
Total delay	1.6 ns
Idle Power	0.00655 W

VI-A Total Area

We were unable to obtain the exact area because we did not create a layout with the process node used in the schematics. Instead, the total area was calculated by taking the summation of all the transistor widths across the entire array.

VI-B Delay

The worst case delay for the model was found by documenting the minimum period it took for the entire model to produce outputs from simulations that matched our expected outputs. This was done for the read/write operations and for also holding data value(s) in the appropriate bitcells. This minimum period was checked across all PVT corners.

VI-C Energy Consumption

VI-C Total Energy Consumption

The total energy consumption was calculated by taking the average energy of 1 write and 5 reads as outlined in the project assignment document.

VI-C Read/Write Energy Consumption

The read and write energy consumption was calculated by taking the average energy of reading/writing a "1" and a "0". This was conducted with both overvolted and non-overvolted wordlines. Since overvolting happens for only 2 out of 8 columns, the non-overvolted energy was weighted 75% and the overvolted energy was weighted 25% in this calculation.

VI-D Idle Power

The idle power of the SRAM Model was calculated in two steps: finding current from both voltage sources and multiplying by VDD to find the power when all inputs are 0, and measuring individual gates when all inputs are 0. For the first step, we simulated the 1b full SRAM model and found the idle power by multiplying by 1.25 V (this is an overestimation). We then multiplied this number by 4 to compensate for the number of blocks and additional top level circuitry. The second step involved sampling the idle current from all the bitcell and block peripherals. We sampled idle current from a 3in NAND, 2in NAND, mux inverter, write driver, precharge PMOS, bitcells, sense amplifiers, and precharge inverters. We then multiplied the sampled currents by the number of actual instances of the object. Then we multiplied this number by $VDD = 1.1$ to find the power. The voltage source that connects to the bitline through the precharge PMOSs accounted for 71% of the power, and the bitcell's VDD accounted for 26% of the power.

VII- CONCLUSION

We have successfully designed a working high speed cache SRAM in accordance with PiCo's requests. We believe we have produced a high quality and optimized SRAM design that is the best option for PICo.

References

[1] V. M. Tripathi, S. Mishra, J. Saikia and A. Dandapat, "A Low-Voltage 13T Latch-Type Sense Amplifier with Regenerative Feedback for Ultra Speed Memory Access," 2017 30th International Conference on VLSI Design and 2017 16th International Conference on Embedded Systems (VLSID), Hyderabad, India, 2017, pp. 341-346, doi: 10.1109/VLSID.2017.15.

VIII- APPENDIX

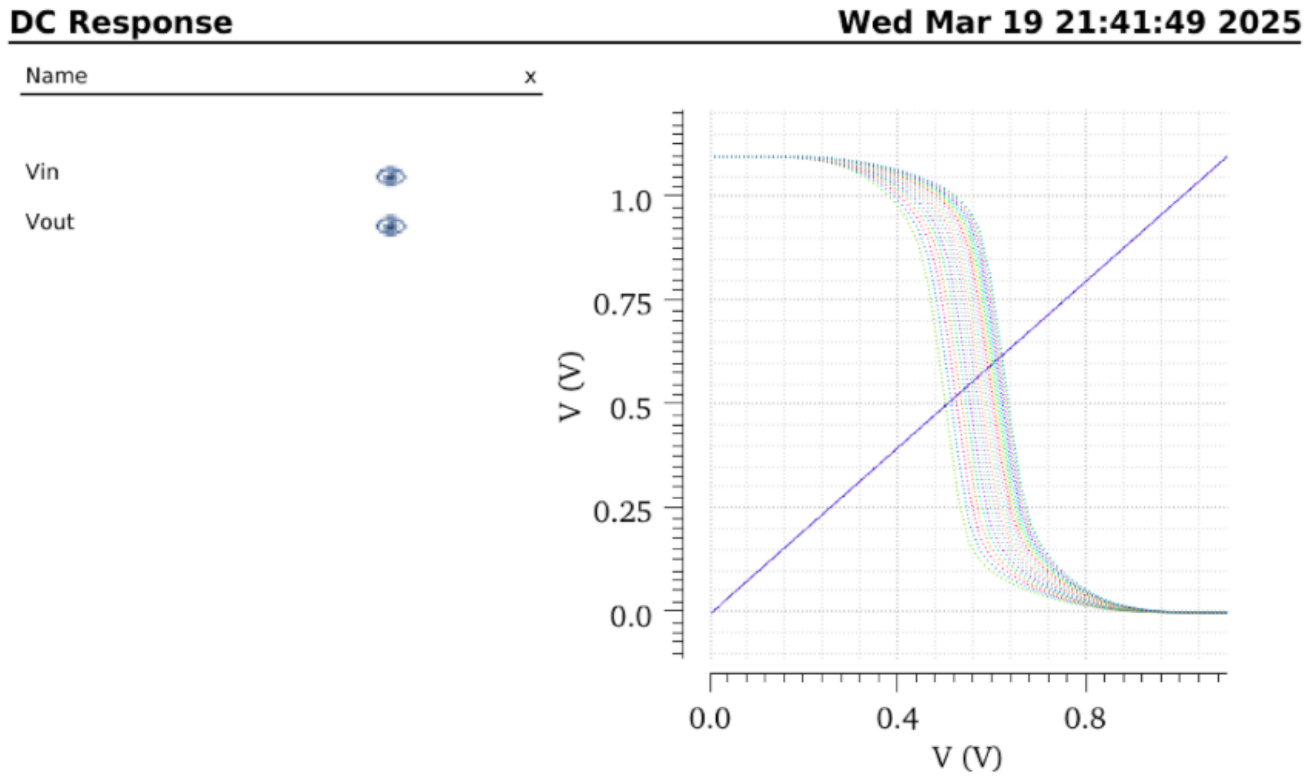

Fig. 1. Inverter Voltage Transfer Characteristic 90-280nM

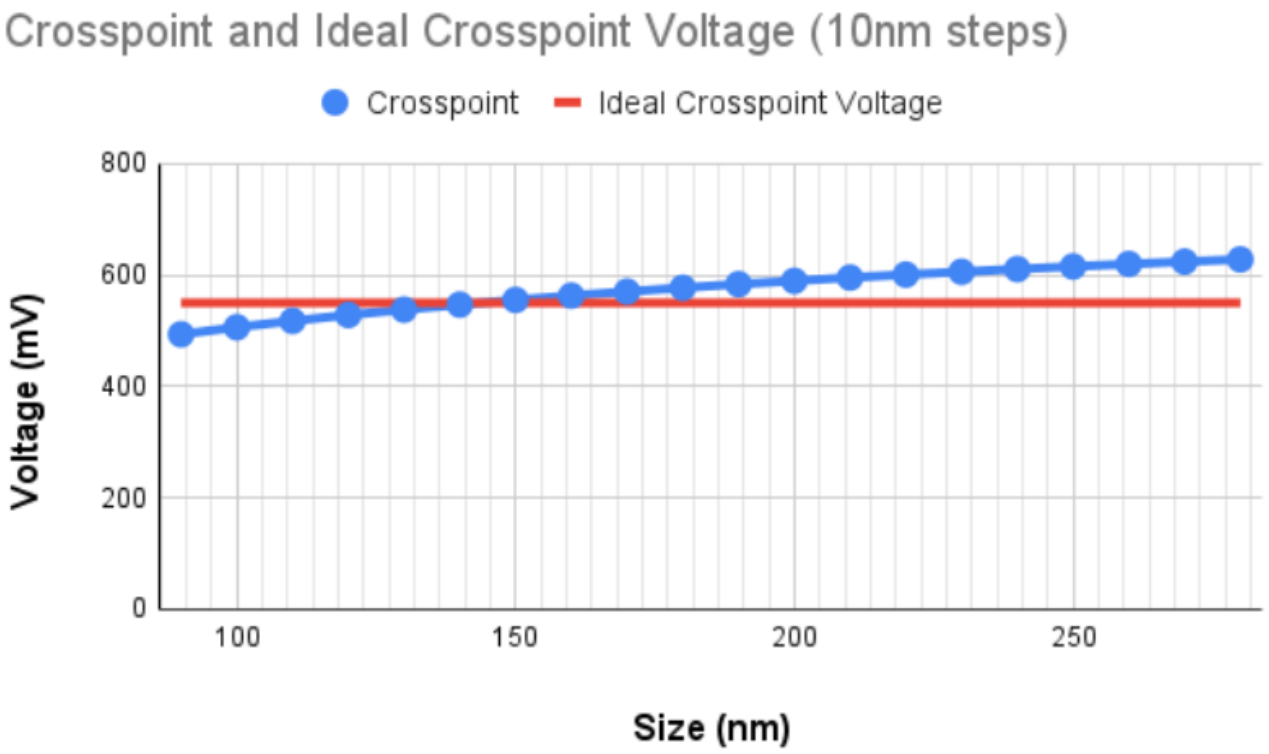

Fig. 2. Crosspoint Voltage 10nm Steps

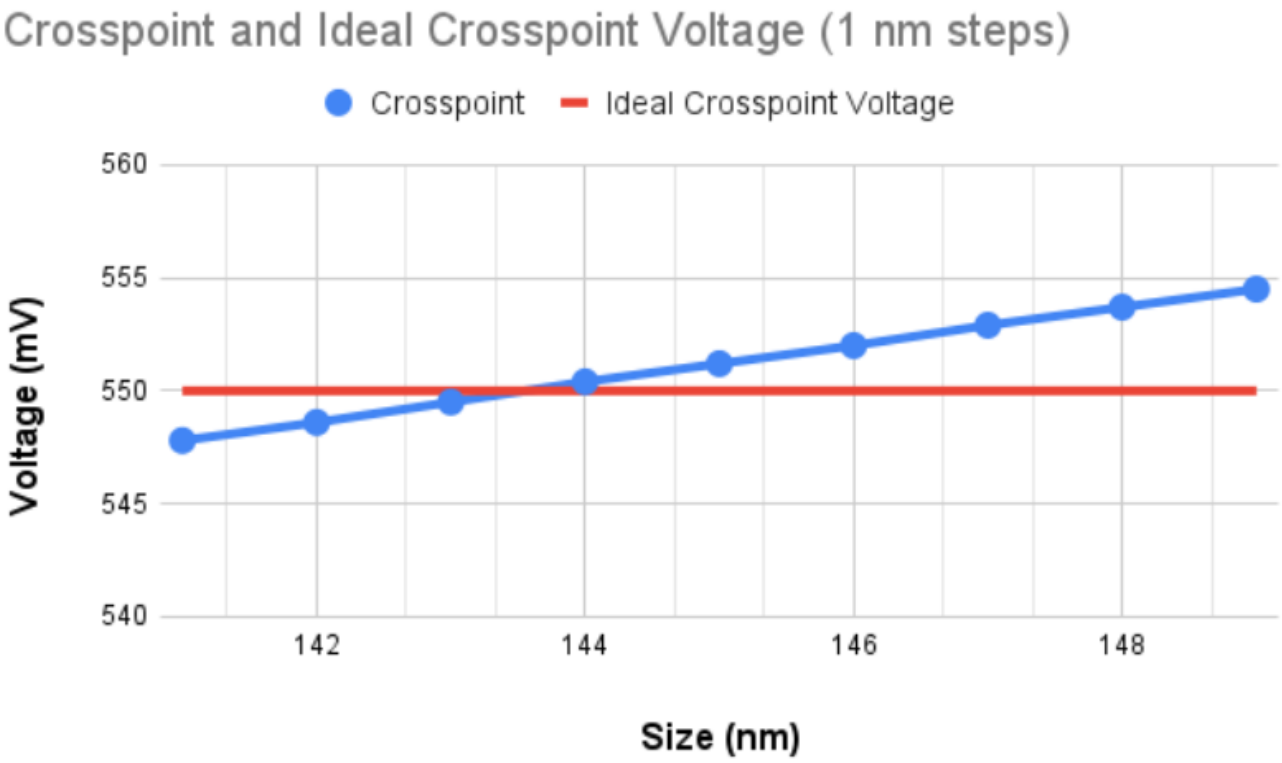

Fig. 3. Crosspoint Voltage 1nm Steps

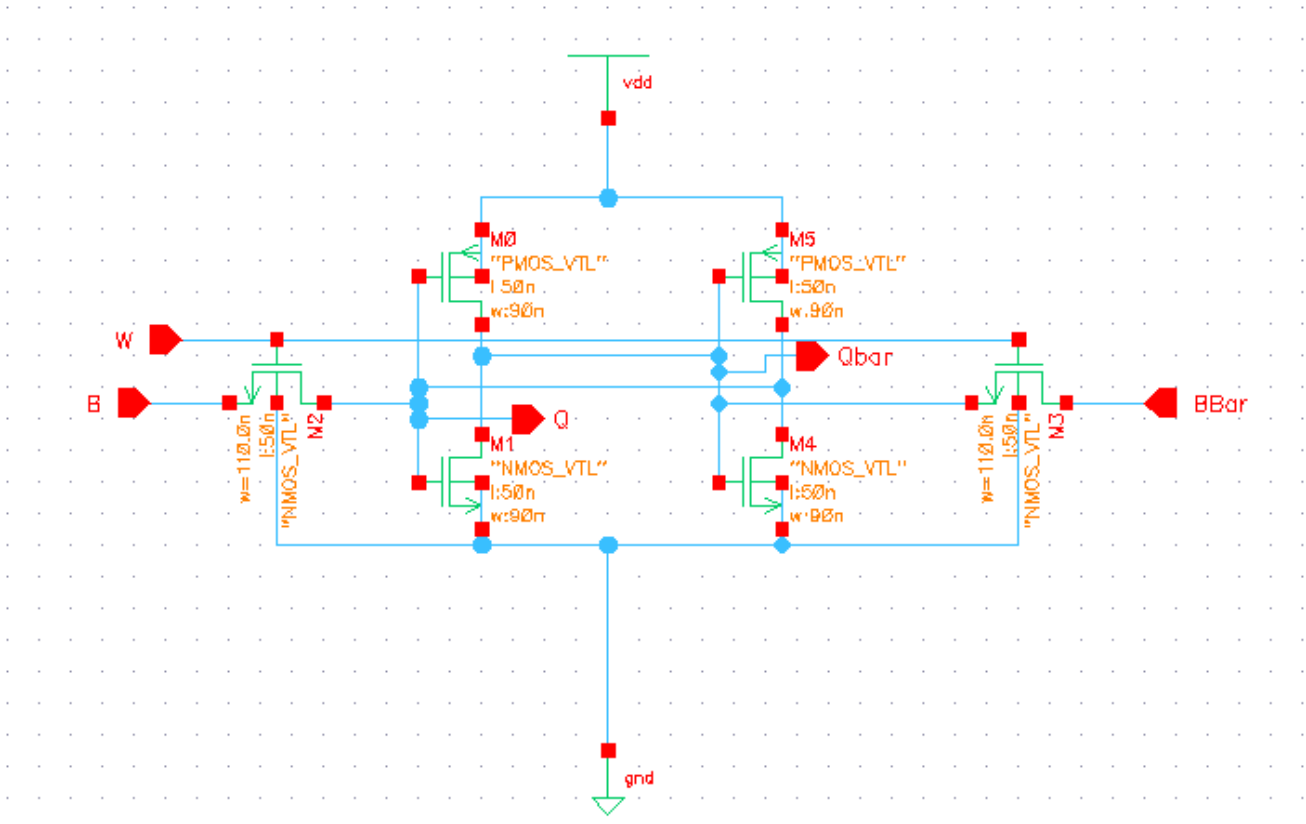

Fig. 4. Bitcell Schematic

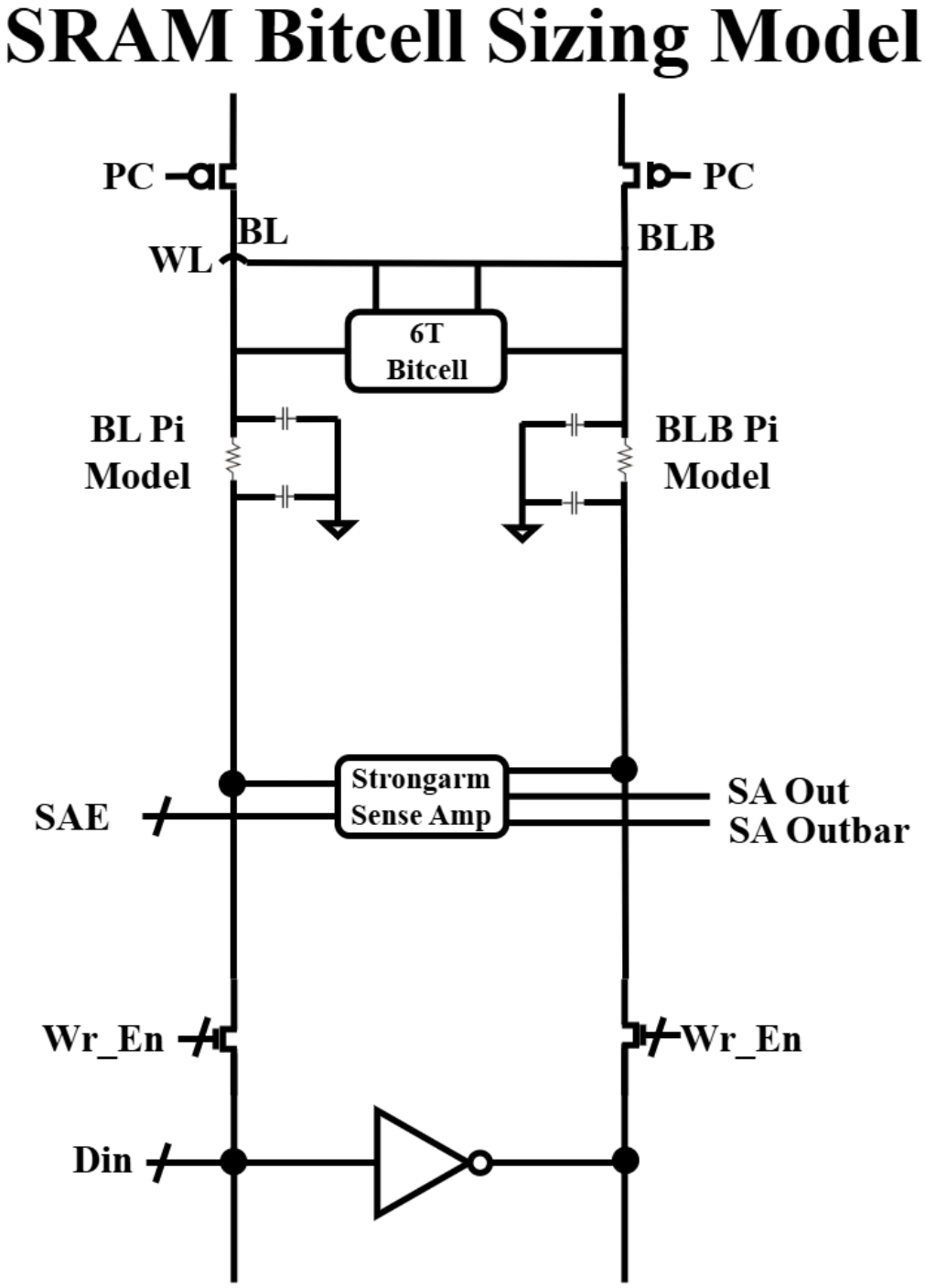

Fig. 5. Bitcell Sizing Testbench

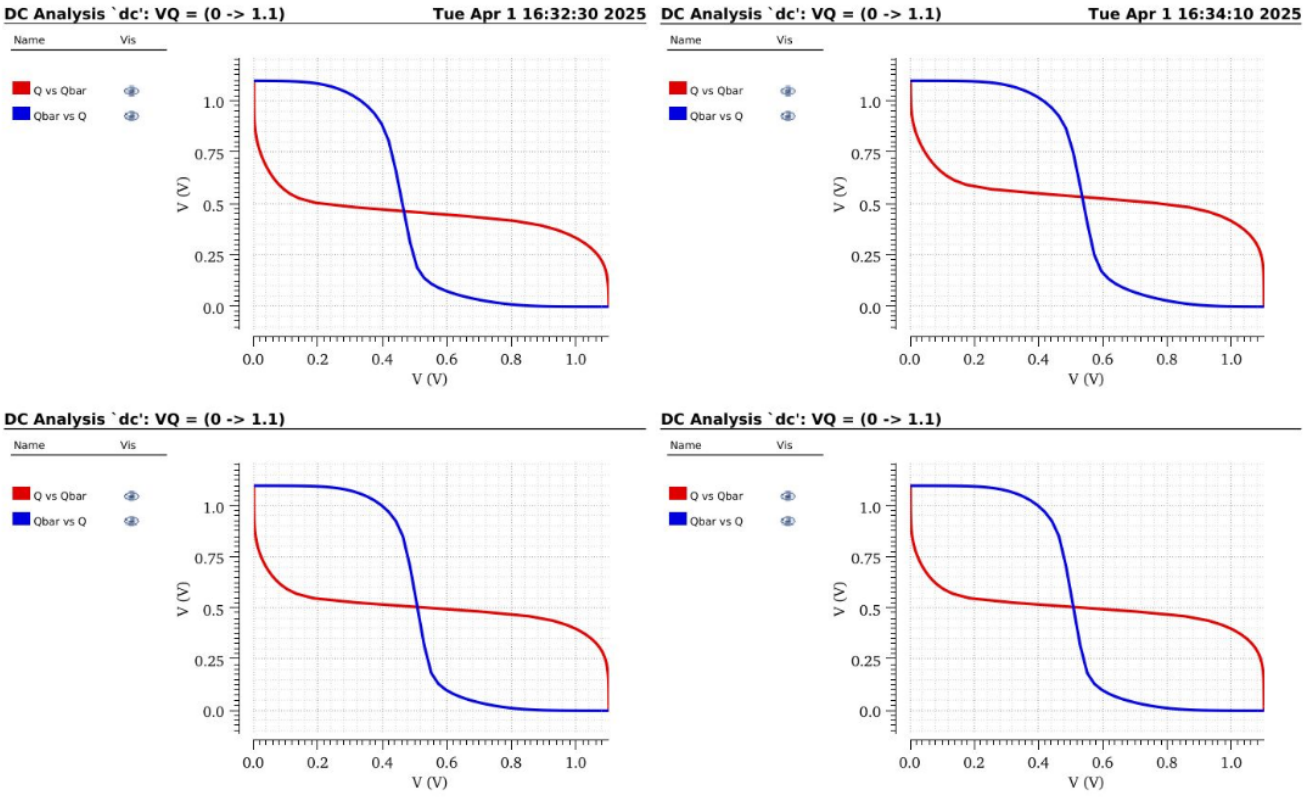

Fig. 6. Bitcell Hold PVT Plots

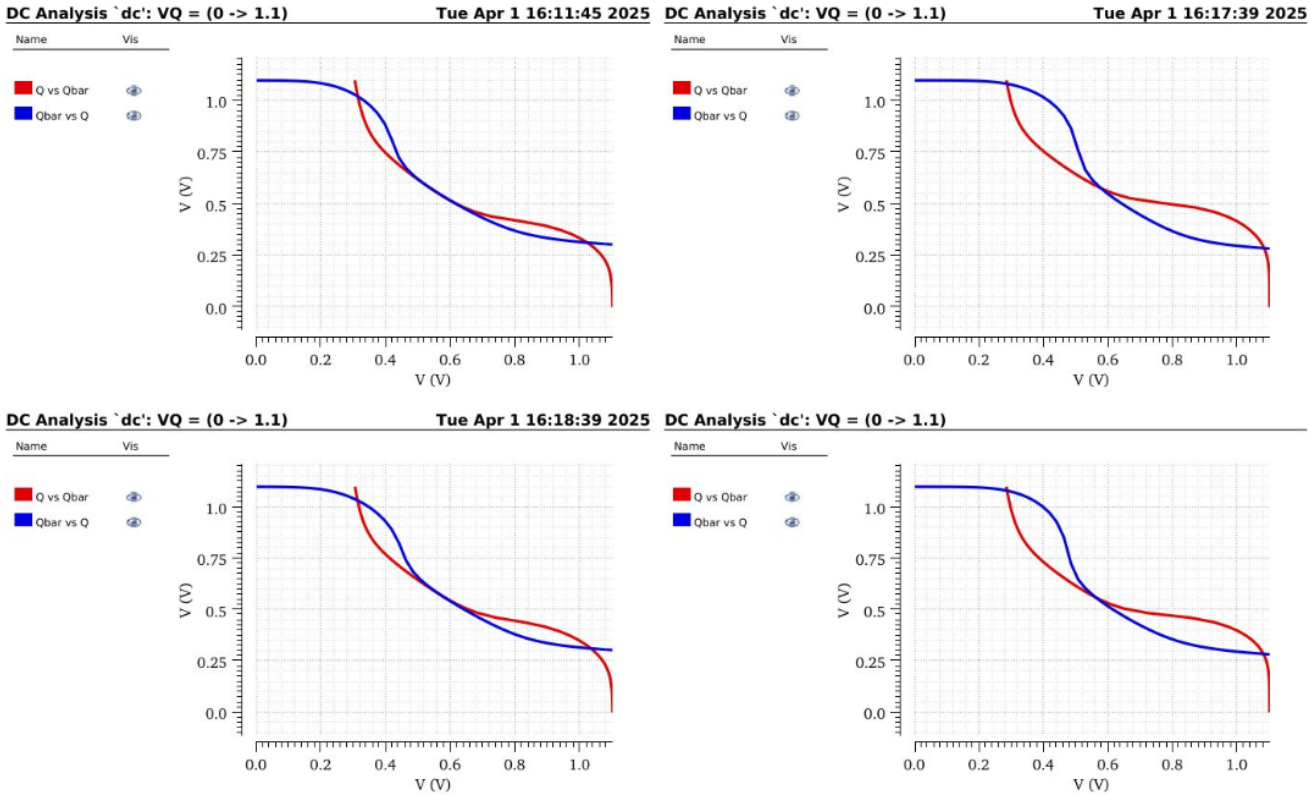

Fig. 7. Bitcell Read PVT Plots

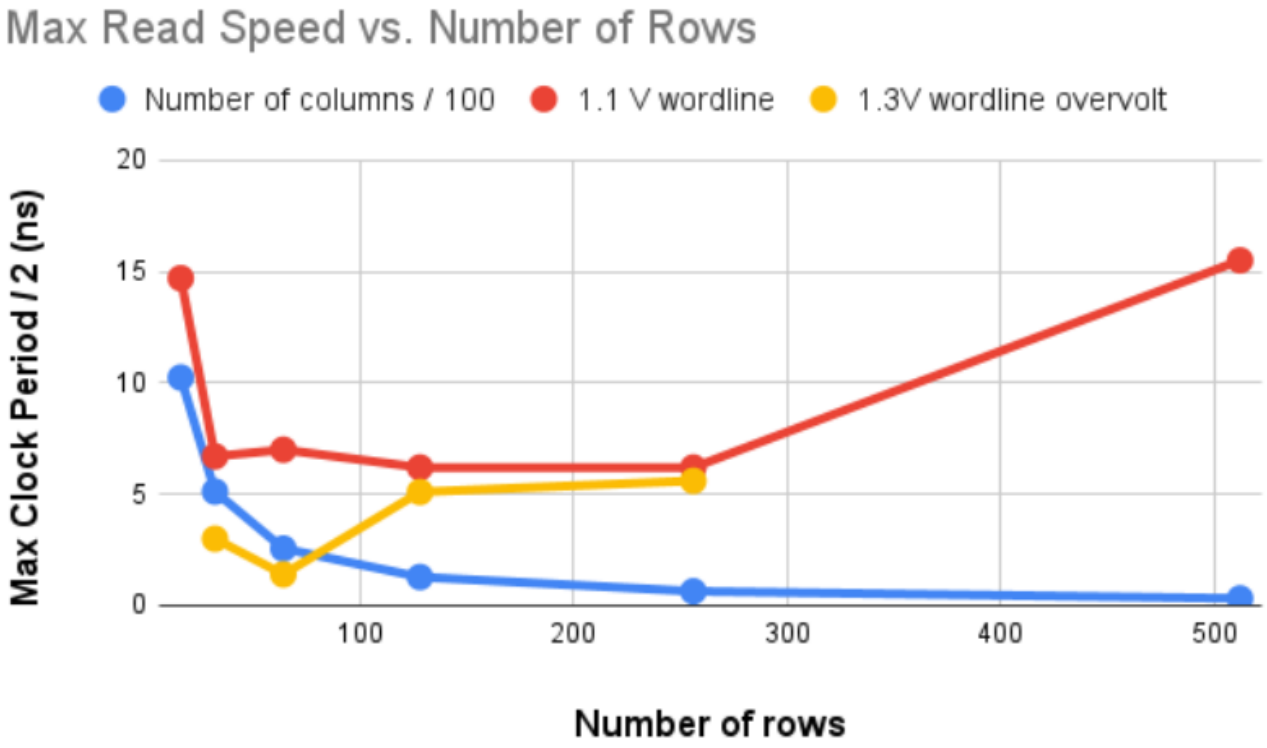

Fig. 8. Array Sizing

Wordline Driver Schematic

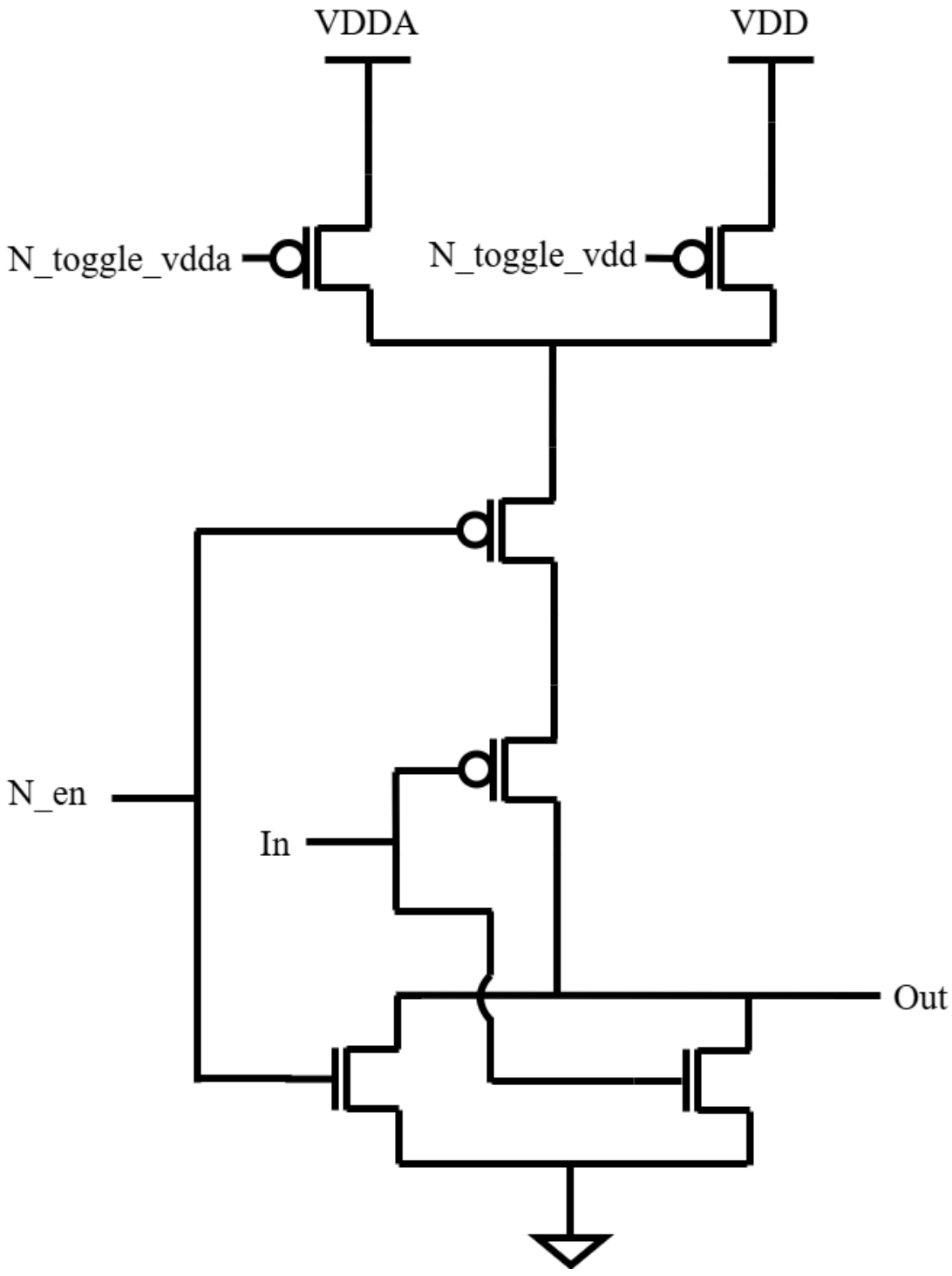

Fig. 9. Wordline Driver Schematic

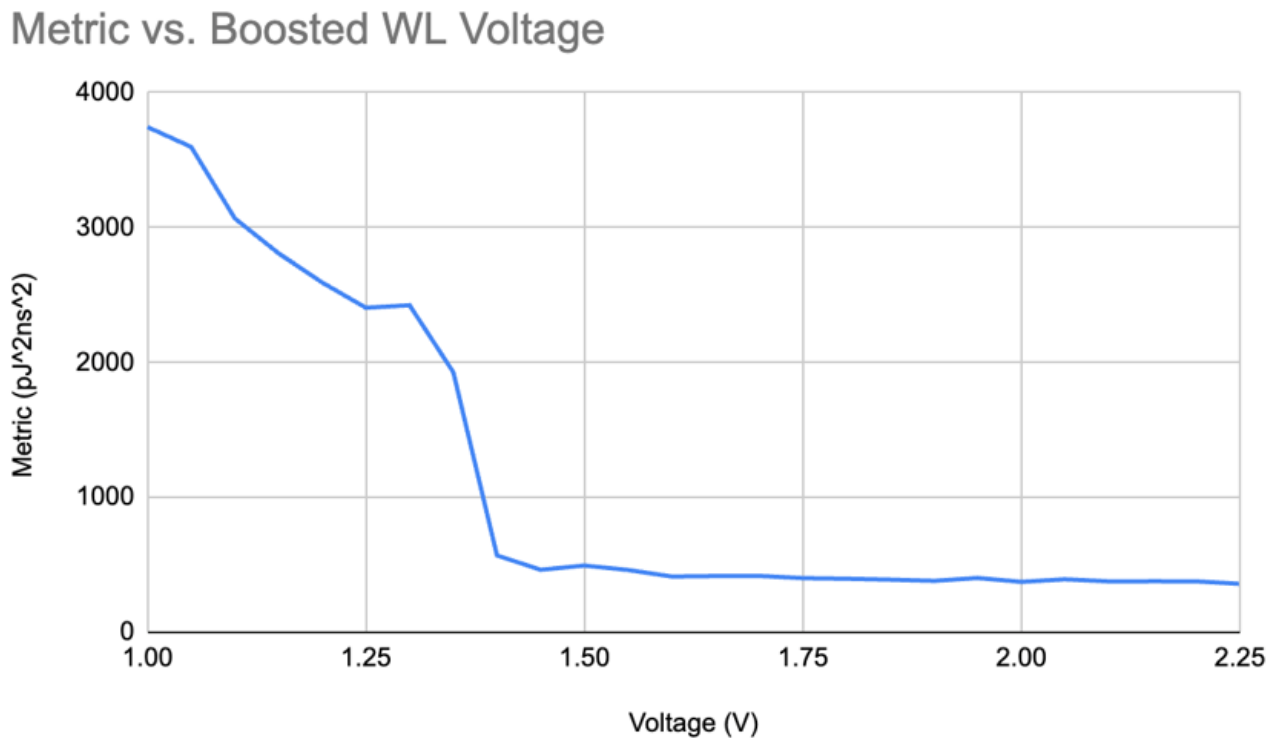

Fig. 10. Wordline Overvolting Results

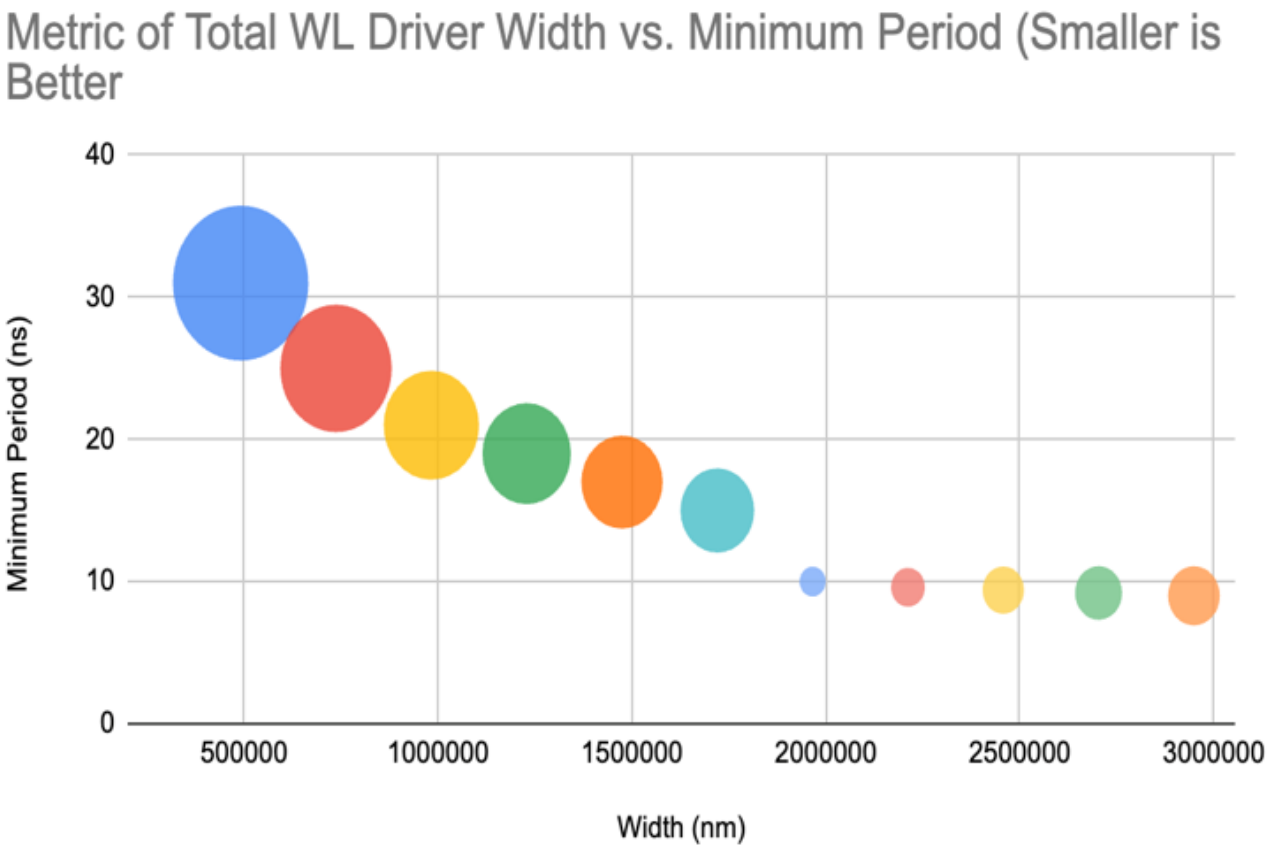

Fig. 11. Wordline Driver Width Metric

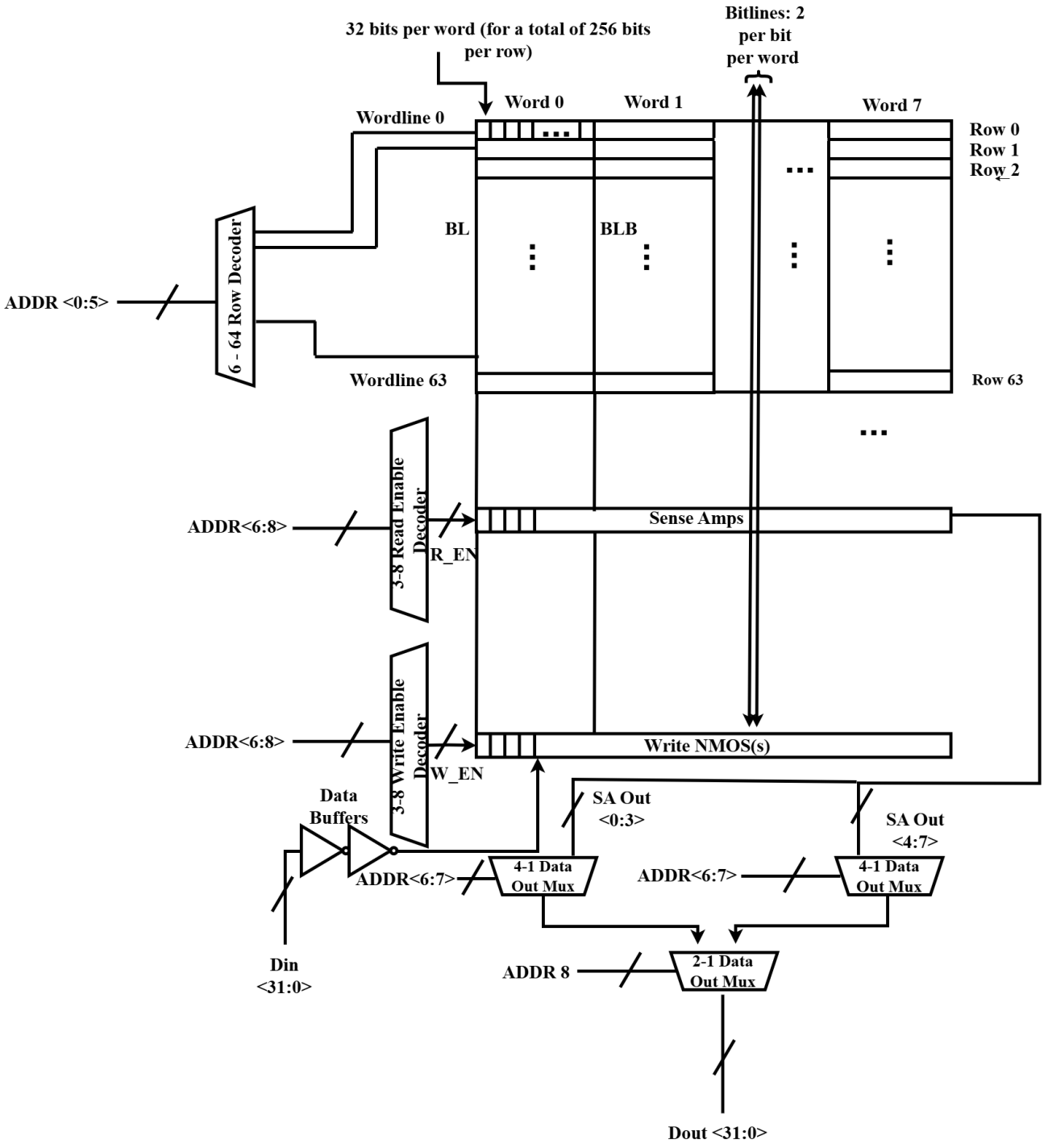

Fig. 12. SRAM Block Schematic

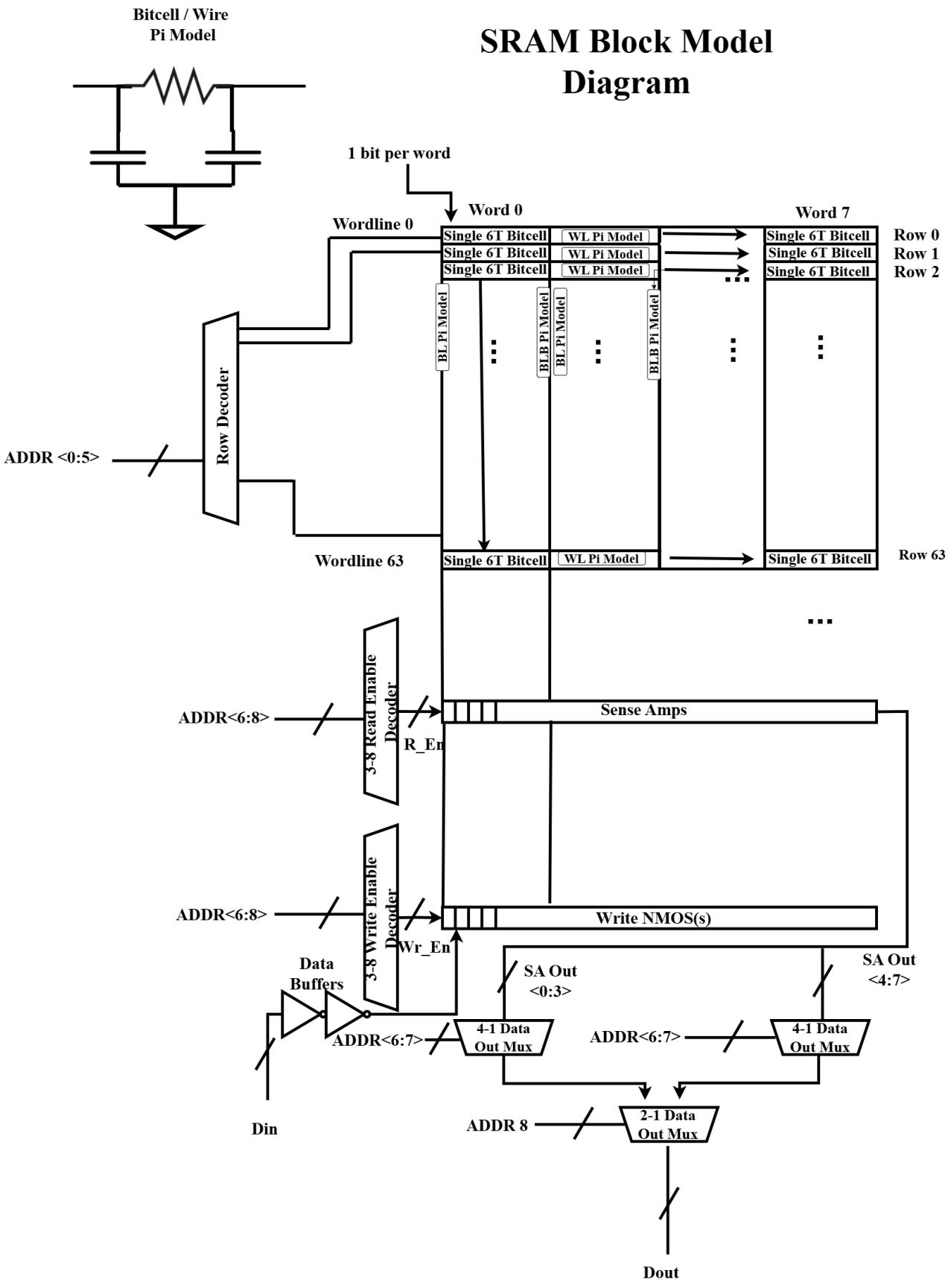

Fig. 13. SRAM Block Model Schematic

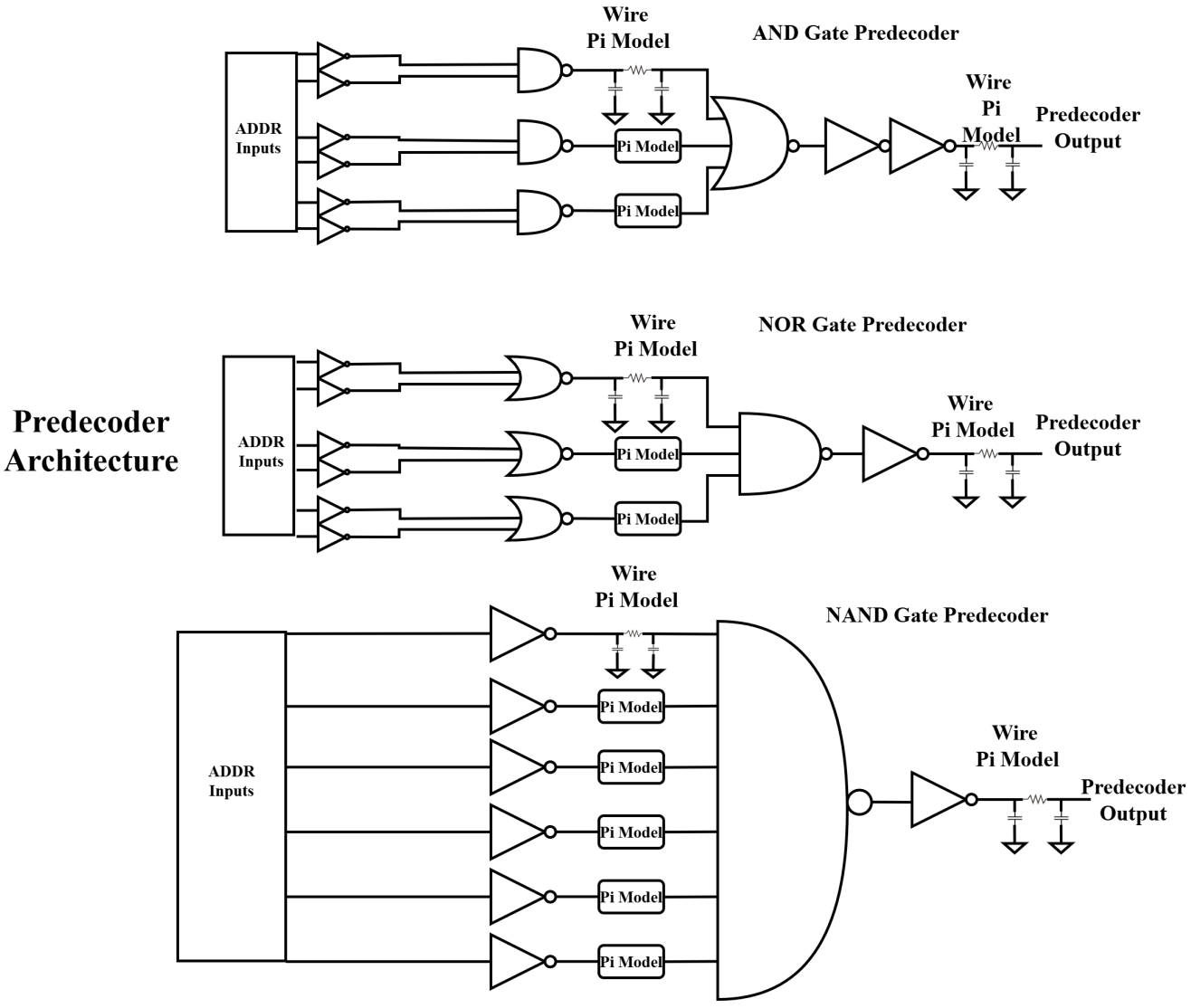

Fig. 14. Decoder Comparison Schematic

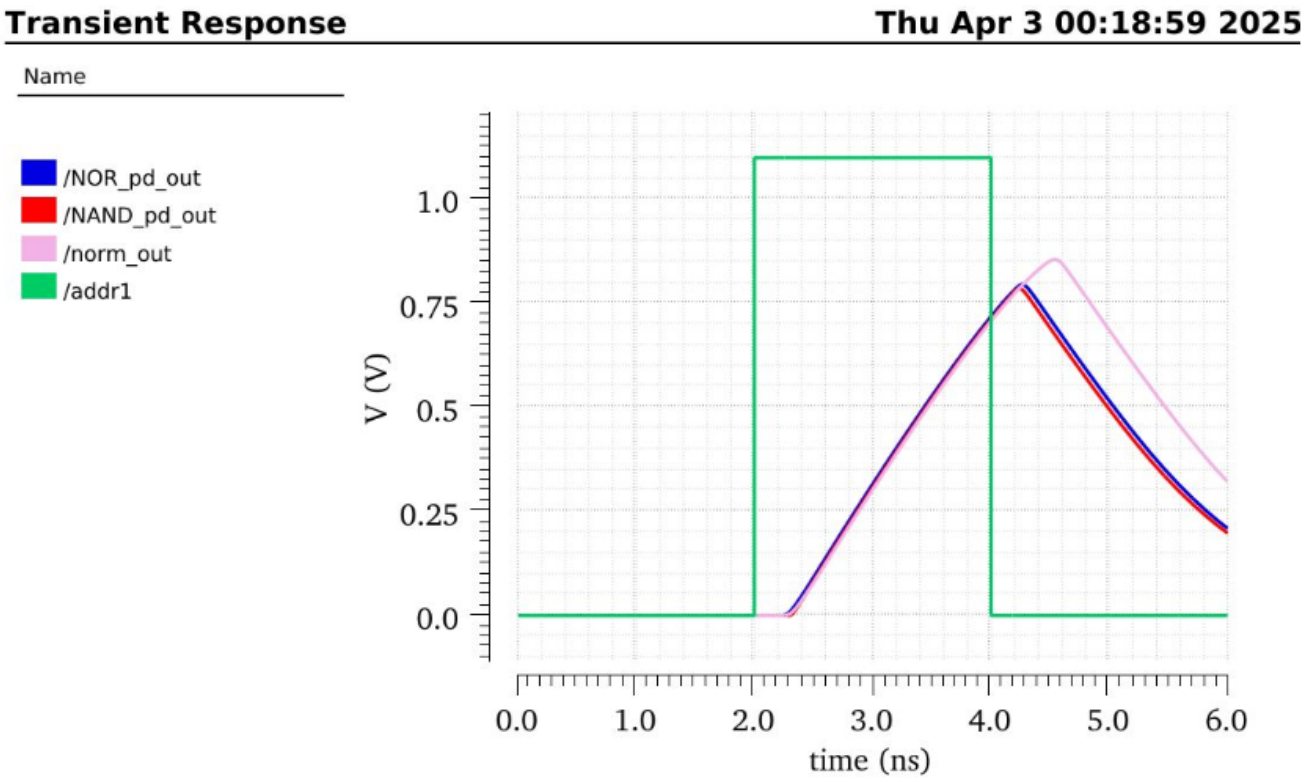

Fig. 15. Decoder Transient Simulation

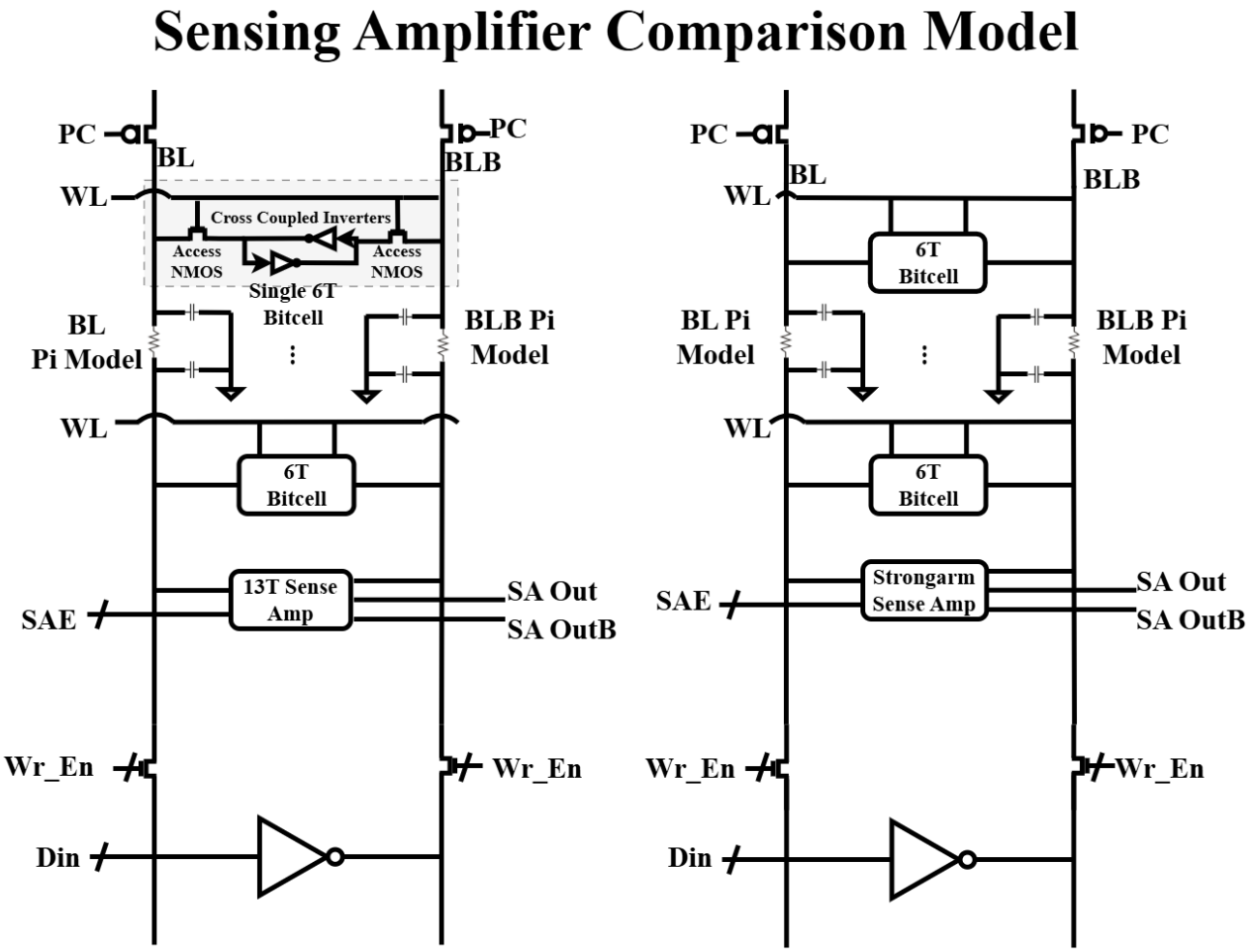

Fig. 16. Sensing Amplifier Testbench Schematic

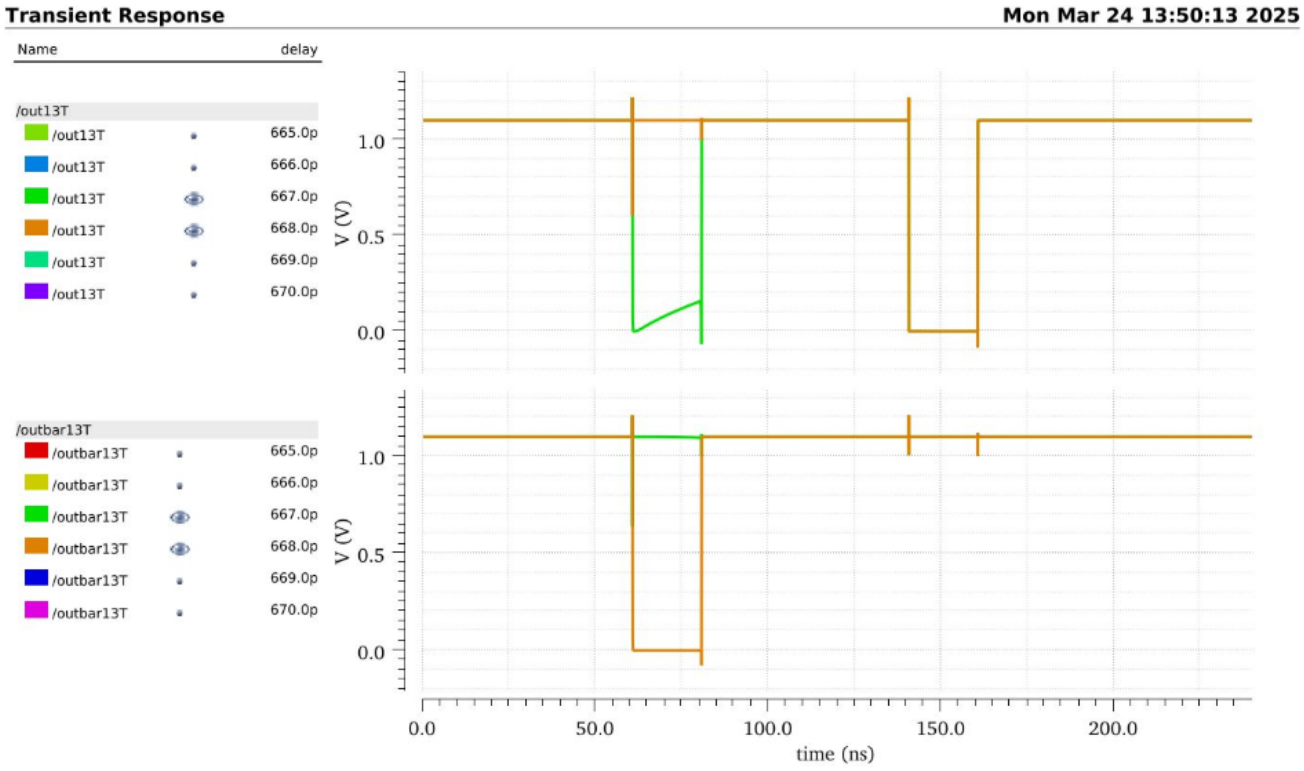

Fig. 17. 13T Sense Amplifier Transient

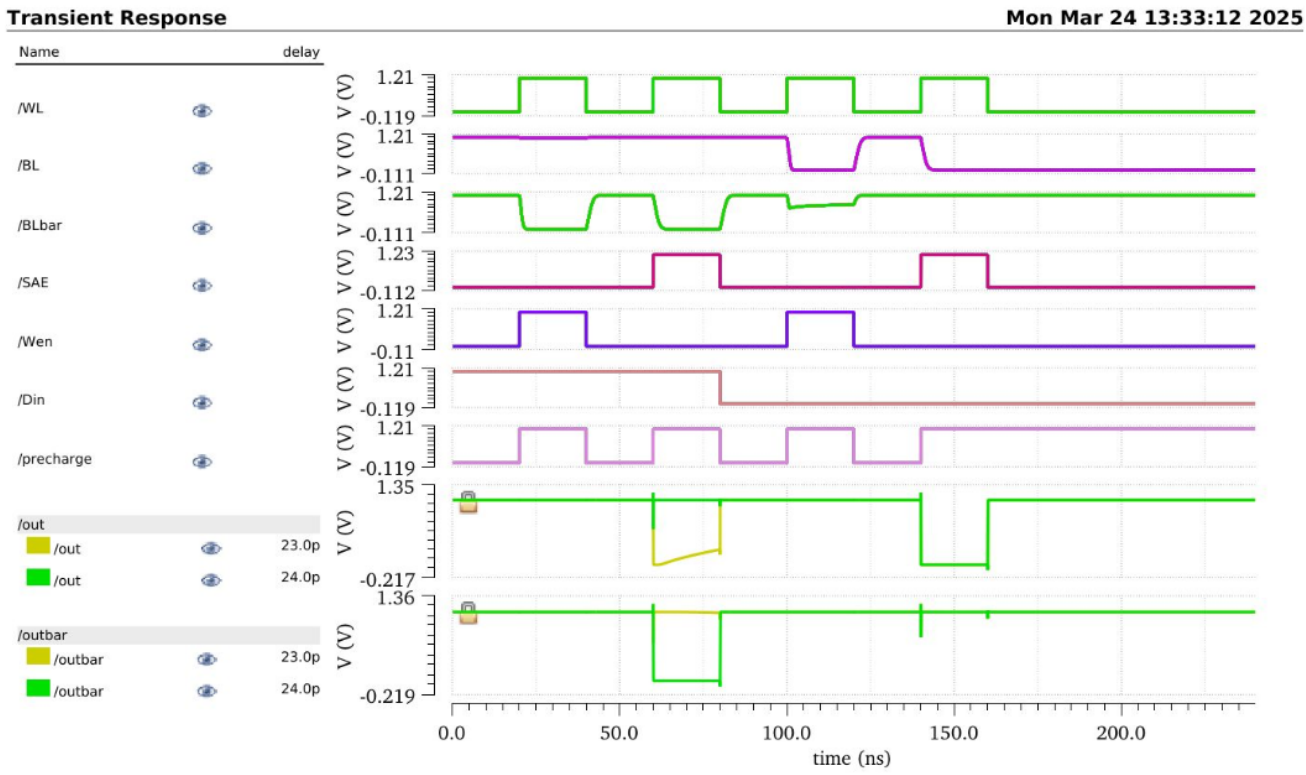

Fig. 18. Strongarm Sense Amplifier Transient

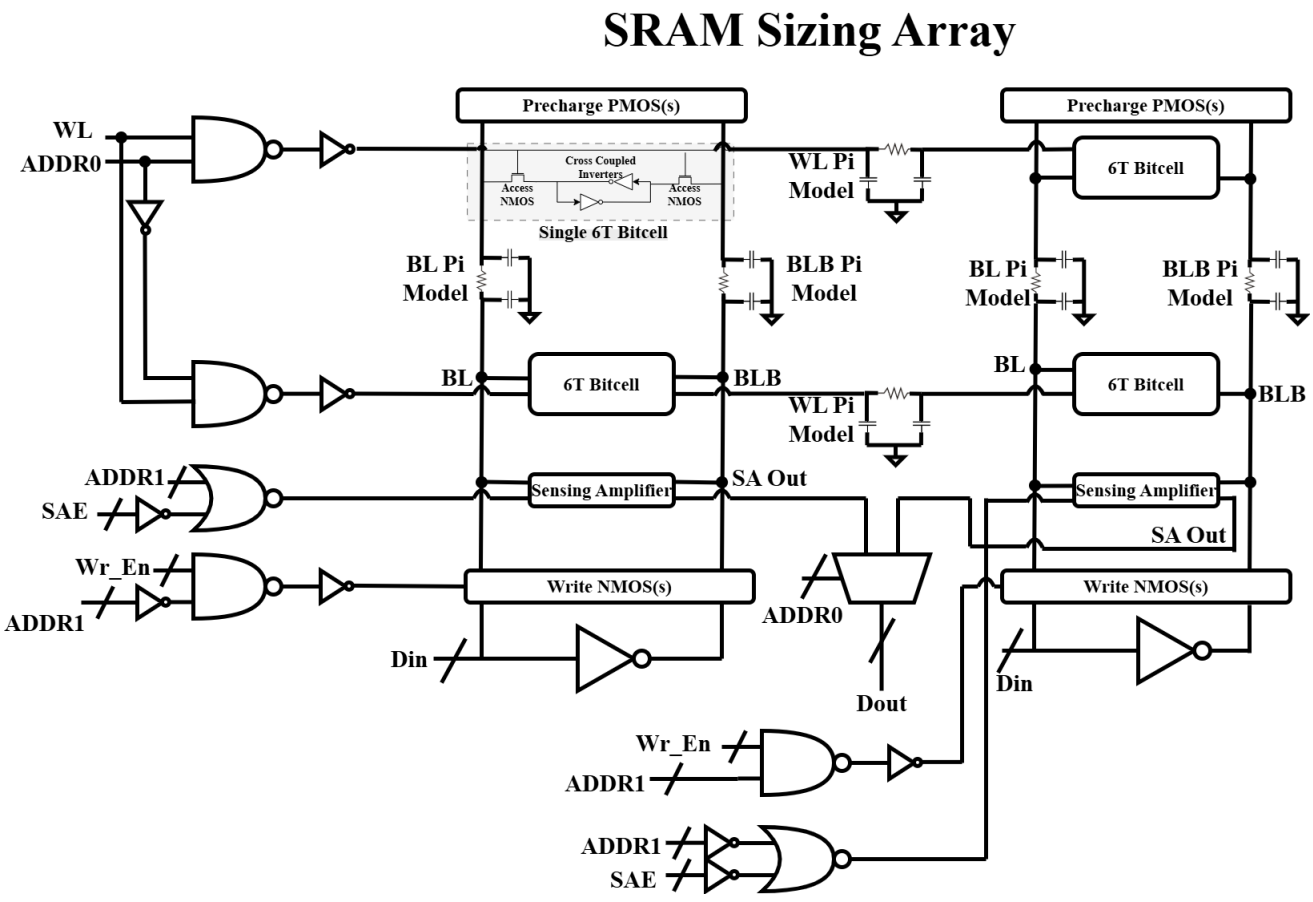

Fig. 19. SRAM Array Sizing Testbench Schematic

Transient Response **Mon Apr 28 00:34:18 2025**