VLSI Final Delivery: Low-Power SRAM Design

Liam Colbert, Wolfgang Ploch, and Shah Zaib Hashmi
ECE 4332
University of Virginia
gvs7fq@virginia.edu, gex4kv@virginia.edu, npn8xg@virginia.edu

ABSTRACT

In this paper, Team Medium Large Scale will be describing the 1Mb SRAM memory that we designed alongside its functionality, key measurable metrics, special features, and the ability to which we met the specifications of the original project assignment.

1. INTRODUCTION

The Portable Instruments Company (PICo) requested bids to design low-power SRAM and high-speed caches that would be built and simulated within FreePDK 45nm technology. Team Medium Large Scale has elected to design a low power SRAM, with the main deliverable being its functionality alongside the optimization metric $(Active\ Energy\ Per\ Access)^2(Delay)(Area)(Idle\ Power)$.

1.2. Requirements

For this project we were required to work in teams of 3 designers to develop an embedded SRAM design for the PICo contract. We selected the Low Power SRAM memory which required that the word size would be 32 bits, the Address could be N bits, the Inputs would be ADDR$< N-1 : 0 >$, Din$< 31 : 0 >$, Read, Write, and CLK. The outputs would be Dout$< 31 : 0 >$, the Accesses/Cycles would be 1 read or write access per clock cycle, and finally the access time must be less than or equal to 3 cycles. Once our SRAM had been designed, we were required to simulate the clocking range, maximum clock period, reading and writing at global corners and variation temperatures, and implement any special features that would give our design the edge to win the contract.

1.3. Specifications

- 64 Blocks with each block being 16 rows of 32 32-bit words
- Total Capacity of 1.04 Mb memory
- Input: ADDR $< 15 : 0 >$
- Sense Amp and Write driver in each word column
- 6:64 Block Decoder, 4:16 Local Row Decoder, 32:5 Local Word Multiplexer, 64:6 Block Multiplexer
- Input ADDR $< 15 : 0 >$ With $< 15 : 9 >$ being for the Block Decoder and Multiplexer, $< 8 : 5 >$ for the Row Decoder, and $< 4 : 0 >$ for the Word Decoder.
- Maximum Clock Speed of 5.2 Mhz
- Output: Dout $< 31 : 0 >$

1.4. Block Diagram

The total memory architecture consisted of 64 blocks, in which each block contained 16 rows and 32 columns. We realized after simulations that we didn't account for how large an effect Elmore's delay would have on having 1024 bitcells in a row and how that huge delay would contribute to our overall power consumption because it took a significant amount of time to charge the bitcells.

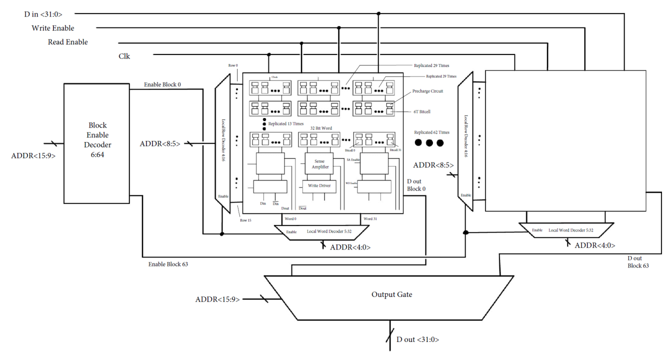

Fig. 1. SRAM Architecture Block Diagram

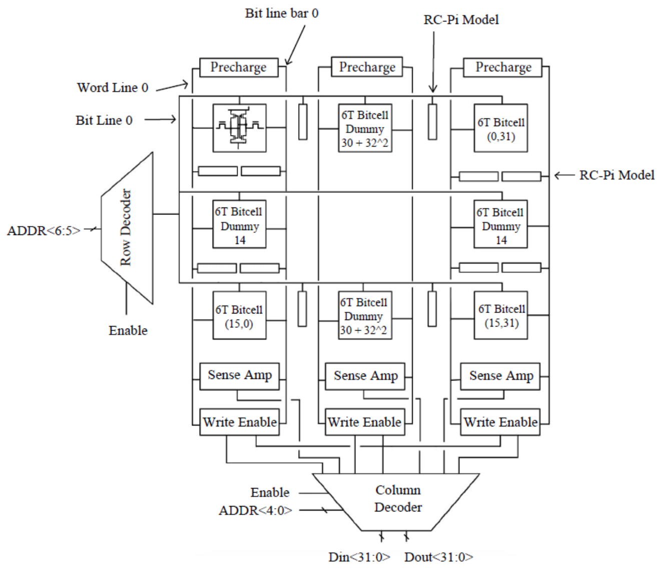

Fig. 2. Simulation Block Diagram

The 32 bit words contained 32 6T bitcells which connected to the necessary word lines and bit lines, along with each having a sense amplifier and read/write drivers. The block diagram used for the memory architecture can be seen in Fig. 1. A larger version of Fig. 1 and Fig. 2 can be found in the appendix for readability.

2. DESIGN AND ARCHITECTURE

2.1. Bitcell

To design the bitcell, the pull down NMOS in the inverter was chosen to have a width of 140 nm. The width of the access transistor was swept to get an optimal voltage at the access point, the ideal size was found to be 110 nm. The size of he pull up PMOS in the inverter was then swept and the optimal size was found to be 90 nm. This means that the cell ratio is 1.27 and the pull-up ratio is 0.82. These values correlate to other values seen. The schematic of the bitcell can be seen below in Fig. 3.

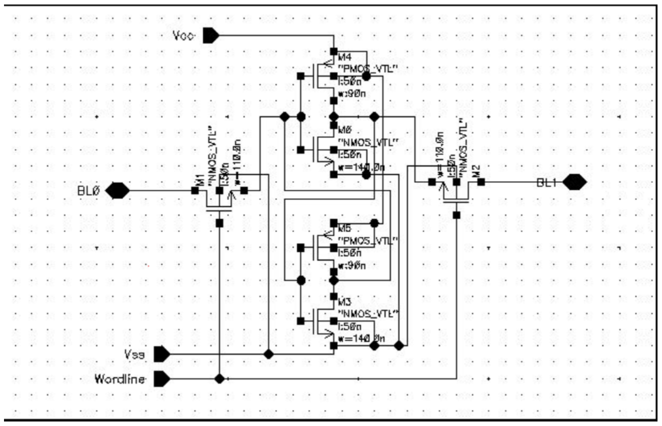

Fig. 3. Final Schematic of Bitcell Used

The input to one of the inverters was swept to create a butterfly plot of the bitcell.

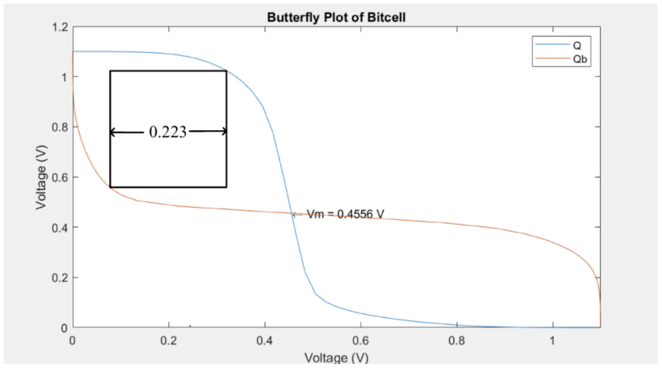

Fig. 4. Butterfly Plot of the Bitcell Showing the Hold Noise Margin

2.2. Decoders and Multiplexers

A 1:64 block decoder first selects a block and activates a 1:16 local row decoder to perform either a read or write operation. A 32 bit 32:1 local word multiplexer selects a word from within the block and sends it to a 32 bit 64:1 block multiplexer which outputs the selected word from the block. A transmission gate based architecture was chosen to design the decoders and multiplexers for its low power consumption and minimal delay. Initially, a 1:2 decoder was designed and then chained with two additional 1:2 decoders to create a 1:4. Then a 1:2 was chained with two 1:4s to create a 1:8. This process was repeated the desired number of outputs was achieved. The multiplexers were made in a similar fashion with the initial 2:1 being replicated 32 times to account for the 32 bit word inputs. A discharge NMOS transistor was added to the output/input of each decoder/multiplexer TX gate respectively. The gate of the discharge NMOS is connected to the same node as the PMOS gate such that it opens when the TX gate is closed. The purpose of this is to help discharge any floating nodes created when the select line changes. The input of the 1:64 and outputs of the 32 bit 64:1 were buffered with two back to back inverters to ensure signal stability.

2.3. Sense Amplifier

A differential sense amplifier was designed to read the information from the bitlines and decide whether the data at an address was a 1 or a 0. A differential amplifier was chosen that monitors the bitlines but does not drive them to save the power it would take to drive them. This differential amplifier sense amp was ultimately not used because it refused to work in the simulations. The approach taken instead was to use three inverters in series to read the bitline. The inverter draw some active power when switching which was accounted for in the metrics. The first inverter has a large NMOS so that is it easy to turn off, the output needs to change when the input is low because of the behavior of the bitlines. The last two inverters are generically sized. This triple inverter can be seen in Fig. 5.

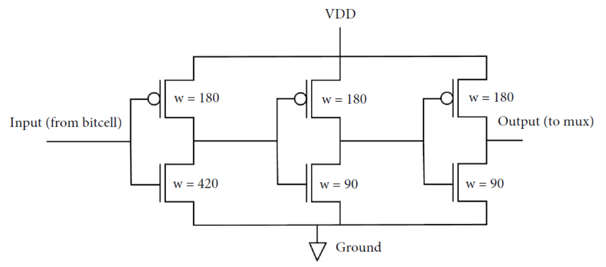

Fig. 5. Triple Inverter Used to Read Bitlines

The triple inverter will take longer to setup than a differential sense amp that drives the bitlines but the clock speed it low enough that this does not matter.

2.4. Array

To determine the sizing of the bitcell array, we first considered what the trade offs of too many / too little blocks in the array would have on our SRAM. In the case of too little blocks, we would have massive amounts of Elmore delay and parasitic capacitance's sitting on all of the lines of the SRAM, while in the case of too many blocks in the array, the overhead required to design and build all of the necessary decoders and multiplexers could outweigh the decrease in delay gained by dividing the SRAM into blocks. When we decided to go with a block size of 16 rows by 32 columns as described in the next section, we then needed a minimum of 32 block to achieve the necessary total memory capacity as described by the requirements. We then built our array of 32 blocks in a 8x4 square with there being 8 rows and 4 columns to shorten the "length" between our decoders and the block they were accessing.

2.5. Block Size

Simulations of the bitline current and wordline current were used to determine the block size. The current draw was approximated as a triangle to simplify the calculations. Different amount of bitcells in per wordline and bitline were used to determine the optimal ratio of size. The lengths and widths of the triangles for the wordline charging can be seen in Table I and the bitline charging can be seen in Table II.

TABLE I. WORDLINE CHARGING

Number of Words	$I_{peak}(\mu Amps)$	Triangle Base (ns)
1	157.16	1.19
2	162	2.3
4	161	4.25
8	161	7.96
16	161	16.52
32	161	37.29
64	161	70.40

TABLE II. BITLINE CHARGING

Number of Bit cells	$I_{peak}(\mu Amps)$	Triangle Base (ps)
4	75.92	347.47
8	78.09	606.81
16	79.4	1020
32	83.60	1882
64	86.15	4.457

It was seen that for the same amount of bitcells, the wordline charging energy was higher than the bitline charging energy. This is why the block was designed to have 1024 horizontally and 16 bitcells vertically. This was not the greatest decision because the increased time increased the power drawn by the memory.

3. SIMULATION AND RESULTS

3.1. Functionality

A TT simulation of the memory can be seen be seen in Fig. 6 and Fig. 7. The test reads and writes a 0 and 1 to the bitcell continuously showing that both operations work. The bitline data shows the need for the low threshold inverter to read the data correctly. The test also shows the sloping behavior of the wordline. The wordline is the limiting factor in speed of the device. There is no latch in the design so the output is only valid when the clock signal is high.

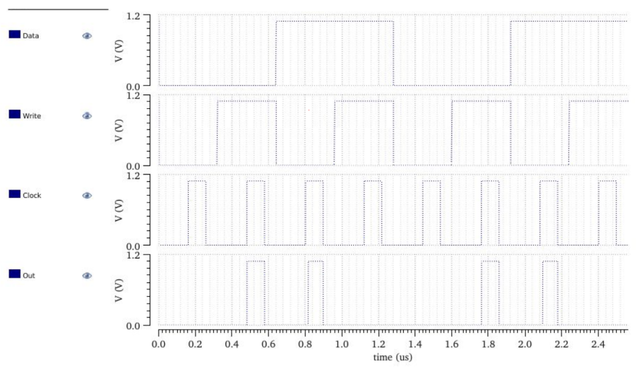

Fig. 6. TT Simulation Showing Write, Clock, Data, and Out

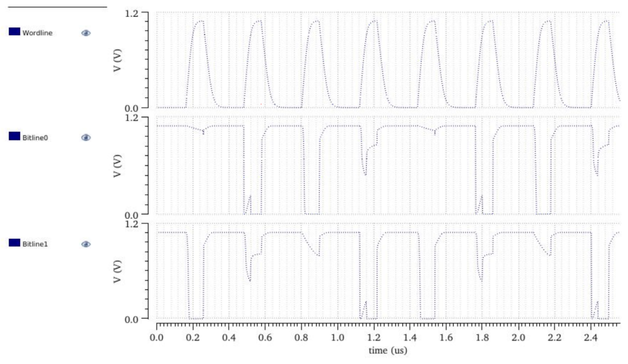

Fig. 7. TT Simulation Showing Bitline0, Bitline1, and Wordline

Enlarged versions of Fig. 6 and Fig. 7 are shown in Appendix A for readability at the end of this report.

3.2. PVT

The SRAM was simulated for worst case read and write (Farthest bitcell from the bitline drivers and wordline drivers). It was simulated for TT, FF, SS, SF, and FS; all of which passed. At TT, the SRAM was tested at 0°, 27°, and 50° C, all of which tests passed. It was also simulated with VCC set to 1.21v (VCC+10%) and 0.99v (VCC-10%), both of which passed as well.

4. METRIC CALCULATIONS

4.1. Area

The area was found by adding the widths of every NMOS and PMOS device in the overall schematic. Table III below shows a breakdown of the area per component and the quantity of each component in the SRAM. The total area of the design is $855.42mm$.

TABLE III. AREA BREAKDOWN

Component	Area (nm)	Quantity	Total Area (mm)
Bitcell	680	1048576	713.0
Sense Amplifier	1140	65536	74.71
BL Driver	450	65536	29.49
Data Inverter	251.25	2048	0.5146
AND Gate	810	64	0.05184
Data Access	150.7	131072	19.75
1:64 Decoder	45630	1	0.04563
1:16 Decoder	10800	64	0.6912
32:1 32b Mux	714240	64	45.71
64:1 32b Mux	1460160	1	1.460

4.2. Delay

The read and write delay was found by adding the amount of time it took the change in address took to propagate through the decoders, the time to charge the wordline, the clock to output delay of the bitcell, and the time to propagate through the multiplexers. The read time was $119.69ns$ and the write time was $103.85ns$. The write time was better because the clock to output delay was better than the read clock to output delay. The worst delay for calculating the metric was the read delay.

4.3. Idle Power

The idle power was found by taking the current that all the bitcells, multiplexers, and internal multiplexers and logic of one of the blocks. This is because only one of the blocks is on at a given time. This current was then multiplied by $1.1v$ to get the power in watts. The total idle power was $44mW$. This idle power seems high but the memory was large with is 1 Mb capacity. Most of the idle power came from the static bitcell current draw because of how many of them there were and the inability to turn them off. The decoders and multiplexers had a negligible idle power of $1.407\mu W$.

4.4. Energy Per Access

To calculate the read and write energies, the active energy of the bitcell, bitlines, wordline, and the triple inverters were accounted for. The bitcell active does not apply to the read energy because the value in the bitcell never changes for a read. The bitline and, inverter, and wordline active energies were the same for a read and a write. The energy for charging two bitlines associated with 16 bitcells and the parasitic capacitances and resistances was $5.724 * 10^{-14}J$, and therefore $1.832 * 10^{-12}J$ for 32 of them. Because the words are organized so there is one per row, the bitline energy is multiplied by 32. Only one wordline needs to be charged and this energy was $3.302 * 10^{-12}J$. The active energy of the triple inverter was multiplied by 32 because the worst case is all of the 32 inverters associated with a word change during a read or a write. The energy for 32 triple inverters was $9.356 * 10^{-11}$. The energy that one bitcell drew when being written to a different value than it previously stored was $9.46 * 10^{-13}$. The worst case when writing a word is that all the bits change so the total energy contribution is 32 times this value: $3.03 * 10^{-11}$. It can be seen that the majority of the power draw comes from the triple inverters. This is because there are so many of them that need to be used because the amount of bitcells in the row. In future designs, this number should be reduced. The final read energy is $98.69pJ$ and the final write energy is slightly higher at $128.97pJ$. The total energy is found by averaging 5 reads and a write and it is $103.74pJ$.

4.5. Final Metric

TABLE IV. FINAL METRICS BREAKDOWN

Metric	Quantity
$\frac{Energy}{Access}^2(Idle)(Delay)(Area)$	9.378*10^{-26}
Total Area (mm)	885.42
Read Energy (pj)	98.69
Write Energy (pj)	128/97
Average Energy/Access (pj)	103.74
Read Delay (ns)	119.69
Write Delay (ns)	103.85
Total Delay (ns)	223.54
Idle Power (mW)	44

5. CONCLUSION

In conclusion, our design Low Power SRAM has a final measurable metric of $9.378 * 10^{-26}$. This SRAM was fully functional and simulated for functionality at all PVT corners and at TT. These simulation included individual simulation on all designed components along with overall read/write functionality of the whole SRAM. The bitcell sizing was swept to determine optimal values for the access, pull-up, and pull-down transistors and the bit-lines and world-lines were swept to determine what our optimal block and array size would be. The sense amps allowed for quicker read times that reduced the amount of voltage swing our word lines would experience, reducing power consumption. The multiplexers and decoders implemented pass gates to further minimize power consumption. With all of this information Team Medium Large scale can confidently report that we have thus completed all requirements for the proposal bid for a Low-Power embedded SRAM from PICo.

6. ACKNOWLEDGEMENTS

We would like to thank Professor Ben Calhoun along with our Teaching Assistants Anjali Agrawal and Aki Takana for their assistance and guidance throughout this semester and for this project. We would also like to thank the University of Virginia Charles L. Brown Department of computer and electrical engineering for the equipment they have provided to us and access to the University's resources.

7. REFERENCES

[1] B. H. Calhoun, R. T. Howe, and C. G. Sodini, Design principles for digital CMOS integrated circuits: the modular series of microelectronic device circuit design. Allendale, NJ: NTS Press, 2012.

APPENDIX A
OVERFLOW FIGURES

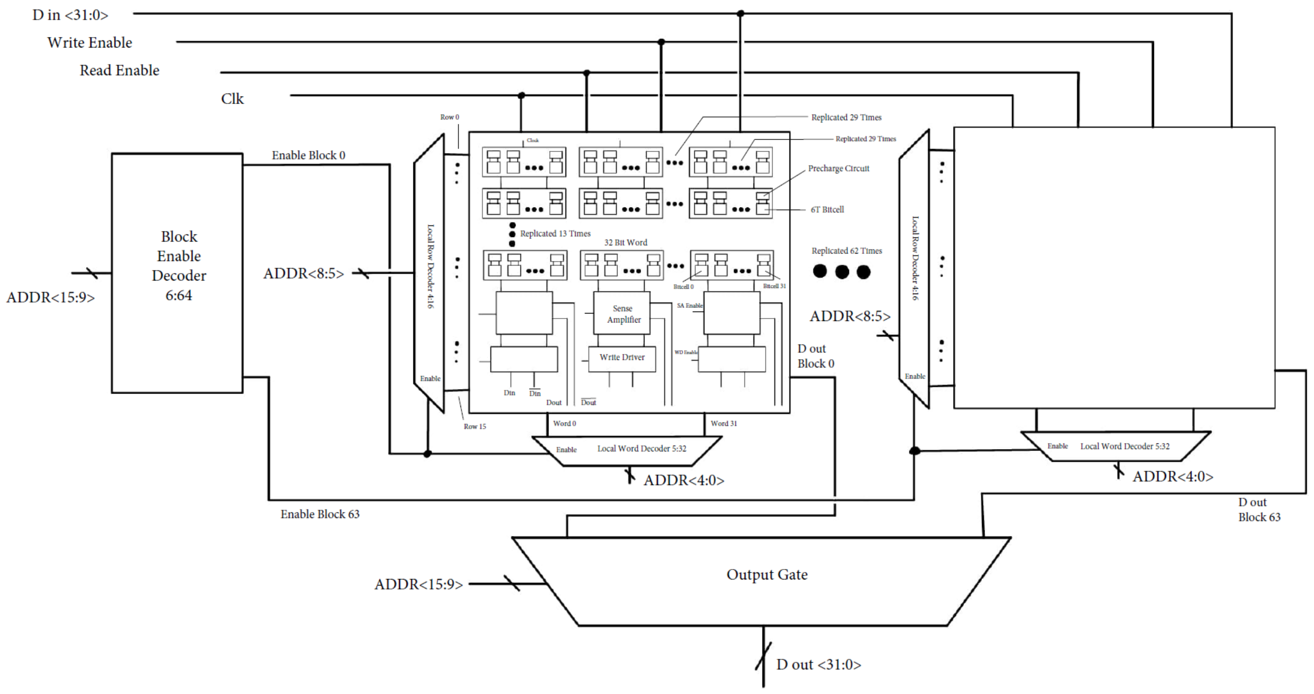

Fig. 8. SRAM Architecture Block Diagram (Enlarged)

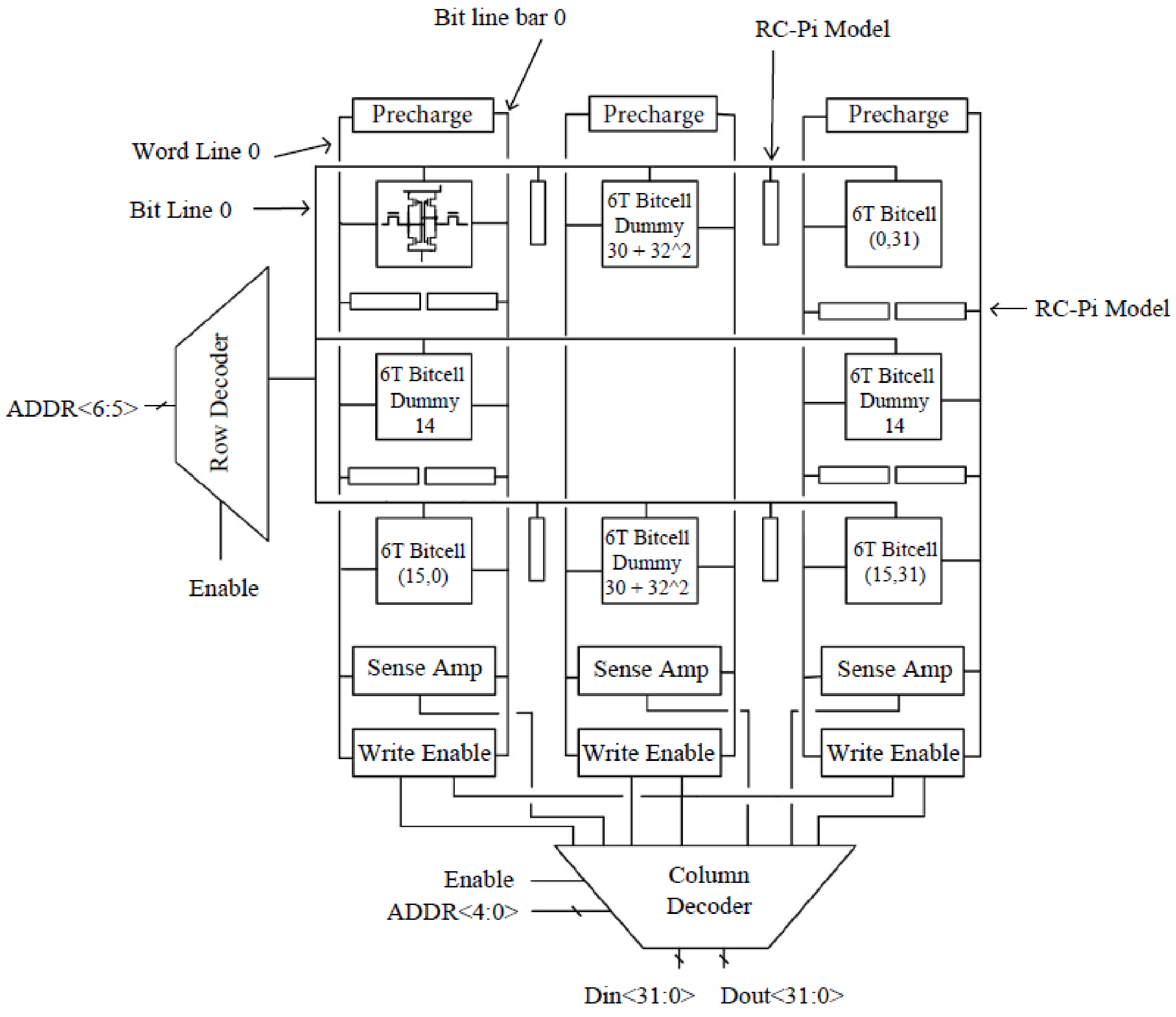

Fig. 9. Simulation Block Diagram (Enlarged)

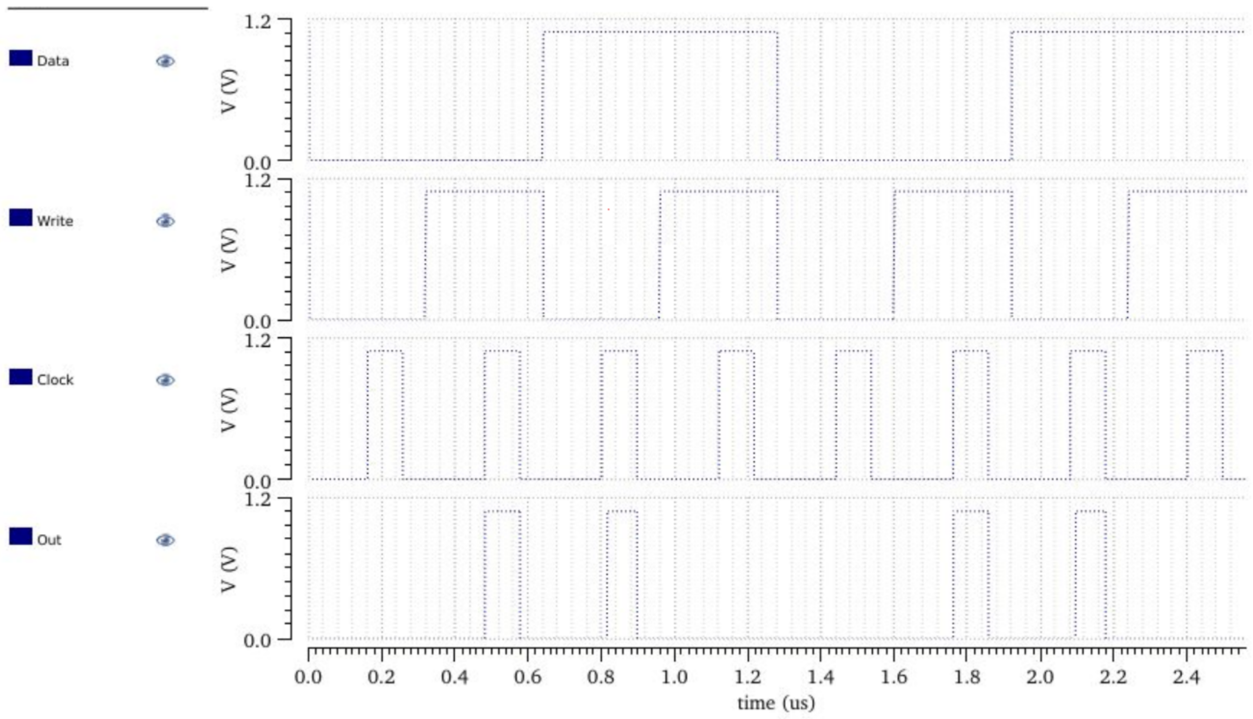

Fig. 10. TT Simulation Showing Write, Clock, Data, and Out (Enlarged)

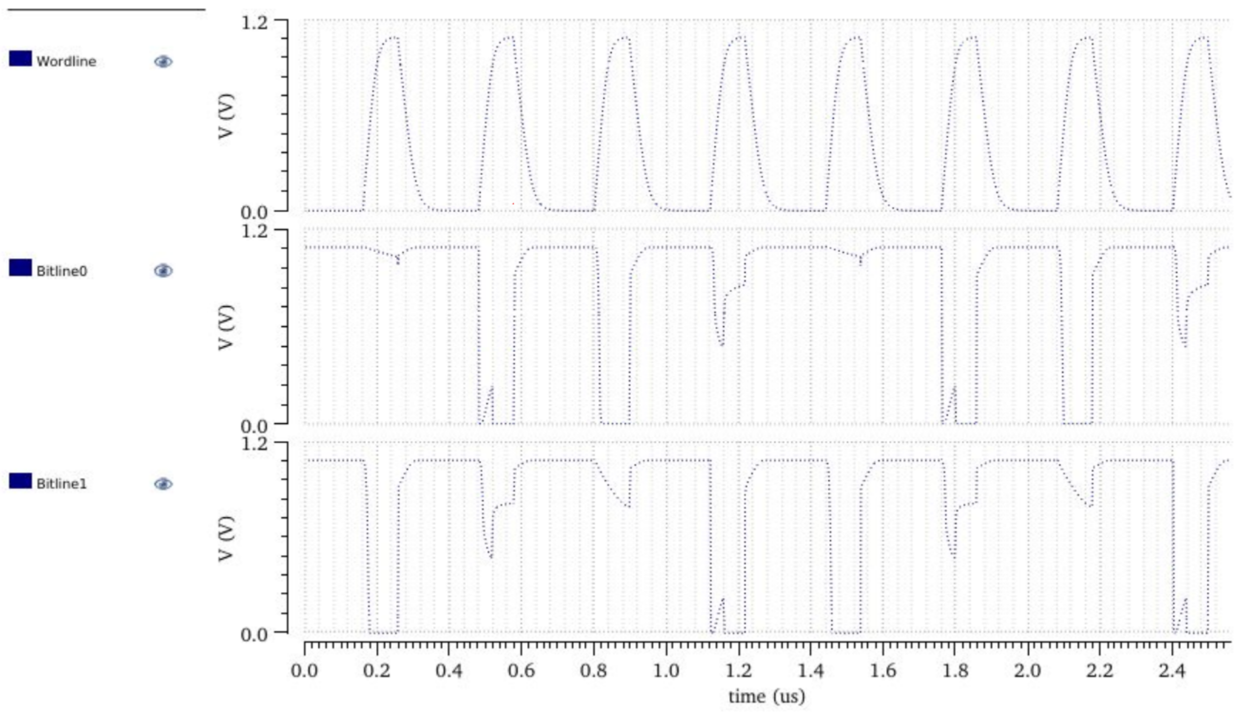

Fig. 11. TT Simulation Showing Bitline0, Bitline1, and Wordline (Enlarged)