Carnegie Mellon University
Research Showcase

Computer Science Department | School of Computer Science

1-1-1983

A 4-kbit four-transistor dynamic RAM

Hank Walker
Carnegie Mellon University

Follow this and additional works at: http://repository.cmu.edu/compsci

Recommended Citation
Walker, Hank, "A 4-kbit four-transistor dynamic RAM" (1983). *Computer Science Department.* Paper 1475.
http://repository.cmu.edu/compsci/1475

This Technical Report is brought to you for free and open access by the School of Computer Science at Research Showcase. It has been accepted for inclusion in Computer Science Department by an authorized administrator of Research Showcase. For more information, please contact research-showcase@andrew.cmu.edu.

NOTICE WARNING CONCERNING COPYRIGHT RESTRICTIONS:
The copyright law of the United States (title 17, U.S. Code) governs the making of photocopies or other reproductions of copyrighted material. Any copying of this document without permission of its author may be prohibited by law.

CMU-CS-83-140

510.1302
C22r
83-140

A 4-KBIT FOUR-TRANSISTOR DYNAMIC RAM

Hank Walker

Department of Computer Science
Carnegie-Mellon University
Pittsburgh, PA 15213

23 June 1983

Abstract

The design of a 64-word by 64-bit dynamic RAM is described. The RAM cell is implemented as a four-transistor dynamic cell. Sense amplifiers are used to reduce access time. Bootstrapped logic reduces power dissipation. Experimental results indicate a typical cycle time of 160ns and power dissipation of 160mW.

Copyright © 1983 Hank Walker

Supported by the Defense Advanced Research Projects Agency, Department of Defense, ARPA Order 3597, monitored by the Air Force Avionics Laboratory under contract F33615-81-K-1539. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

University Libraries
Carnegie Mellon University
Pittsburgh PA 15213-3890

Table of Contents

1. Introduction
2. Functional Description
3. Circuit Design
    3.1. RAM Cell
    3.2. Column Precharge
    3.3. Sense Amplifier
    3.4. Write Logic
    3.5. Address Driver
    3.6. Row Decoder/Driver
    3.7. Output Buffer
    3.8. Read Cycle Timing
    3.9. Write Cycle Timing
4. Layout Design
5. Experimental Results
    5.1. Low-Speed Testing
    5.2. High-Speed Testing
6. Conclusions
    6.1. Circuit Design
    6.2. Layout Design
    6.3. Testing
        6.3.1. Test Programs
        6.3.2. Low-Speed Testing
        6.3.3. High-Speed Testing
        6.3.4. On-Chip Test Logic
    6.4. Summary
7. Acknowlegements
8. References

List of Figures

Figure 2-1: RAM Block Diagram
Figure 2-2: Read and Write Cycle Timing
Figure 3-1: RAM Circuit Schematic
Figure 3-2: Read Cycle Timing
Figure 3-3: Write Cycle Timing
Figure 4-1: Four Transistor RAM Cell Layout
Figure 4-2: Row Decoder and Word Line Driver Layout
Figure 4-3: Address Line Driver Layout
Figure 4-4: Sense Amplifier/Write Logic/Output Buffer Layout
Figure 4-5: Precharge Logic Layout
Figure 4-6: RAM Test Chip Layout

1. Introduction

Custom integrated circuits frequently make use of on-chip memory to form register files, status registers, stacks, etc. The Programmable Systolic Chip (PSC) [Fisher 83, Dohi 83] requires a 64-word by 60-bit writeable control store. The dynamic RAM described here was developed to satisfy this need. The design uses a 15 x 26 lambda four-transistor dynamic RAM cell and has a 1300 x 2000 lambda overall size. The typical cycle time is 160ns with a power dissipation of 160mW. The design was fabricated through the MOSIS ARPA implementation service [Cohen 81]. The four-transistor RAM turned out to be too large for the PSC application, and a three-transistor design was developed [Walker 83].

2. Functional Description

A block diagram of the RAM is shown in Figure 2-1. The signals to the RAM include address input (AIN), data input (DIN), data output (DOUT), VDD, GND, write enable (WE), address disable (DISABLE), sense amplifier clock (SAC), and precharge (PC). Write enable forces the data input values onto the bit lines when high. Address disable forces all select lines low when high. The sense amplifiers are activated on the rising edge of SAC. The outputs remain valid while SAC is high. The bit lines are precharged when PC is high. Read and write cycle timing is shown in Figure 2-2. Refresh occurs on a read cycle.

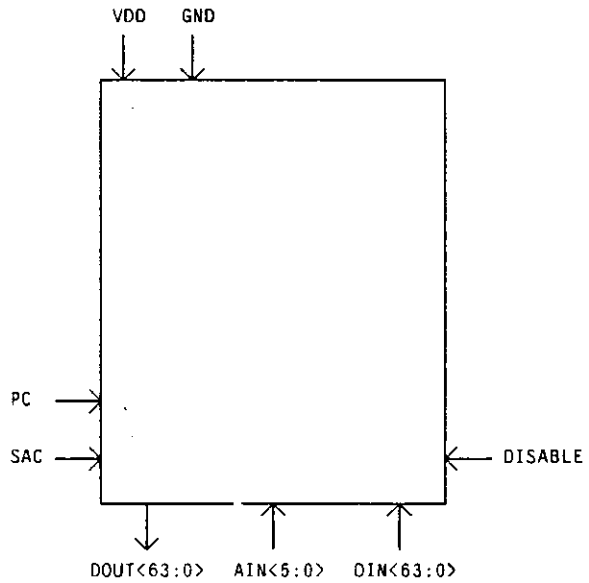

Figure 2-1: RAM Block Diagram

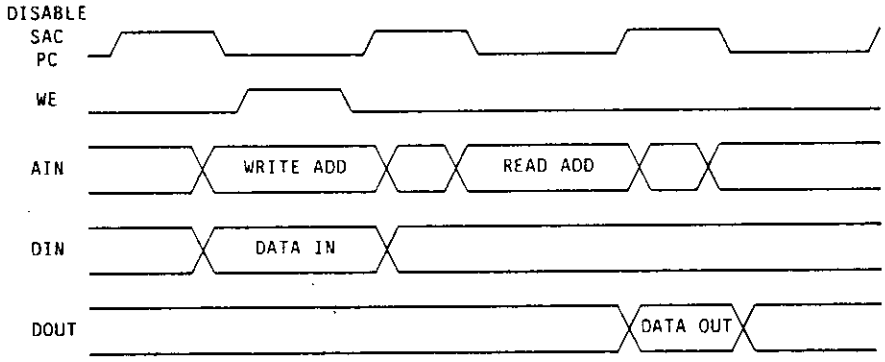

Figure 2-2: Read and Write Cycle Timing

3. Circuit Design

The RAM circuit design consists of seven basic blocks: A four-transistor RAM cell, column precharge, sense amplifier, write logic, address driver, row decoder/driver, and output buffer. Schematic diagrams of these blocks are shown in Figure 3-1. The transistor sizes are given as width/length in lambda units. The RAM was fabricated with a lambda of 2.0 microns and 2.5 microns.

3.1. RAM Cell

The four-transistor RAM cell is connected to a pair of bit lines (B and BBAR) and a select line (SELECT). The cell is written by setting one of the bit lines high, the other low, and the select line high. The side of the cell connected to the low bit line is discharged (if it is not already). The pulldown transistor connected to the opposite side of the cell is now turned off. This allows the side connected to the high bit line to be charged high. This high voltage turns on the pulldown transistor on the low side of the cell, re-enforcing the low voltage. This state will be maintained when the cell is deselected. The cell is dynamic since leakage will cause the high voltage to droop.

To read the cell, both bit lines are charged high, and the cell is selected. The side with a low voltage begins discharging its bit line. The side with the high voltage is refreshed by the high bit line voltage.

The storage capacitance on each side of the cell is 0.033pf. The bit line presents a load of 1pf, and the cell pulls it down from 5V at a simulated rate of 16ns/volt. The column precharge logic prevents the cell from discharging the bit lines below 2.75V. During high-speed operation, the cell does not have sufficient time to pull the bit lines this low.

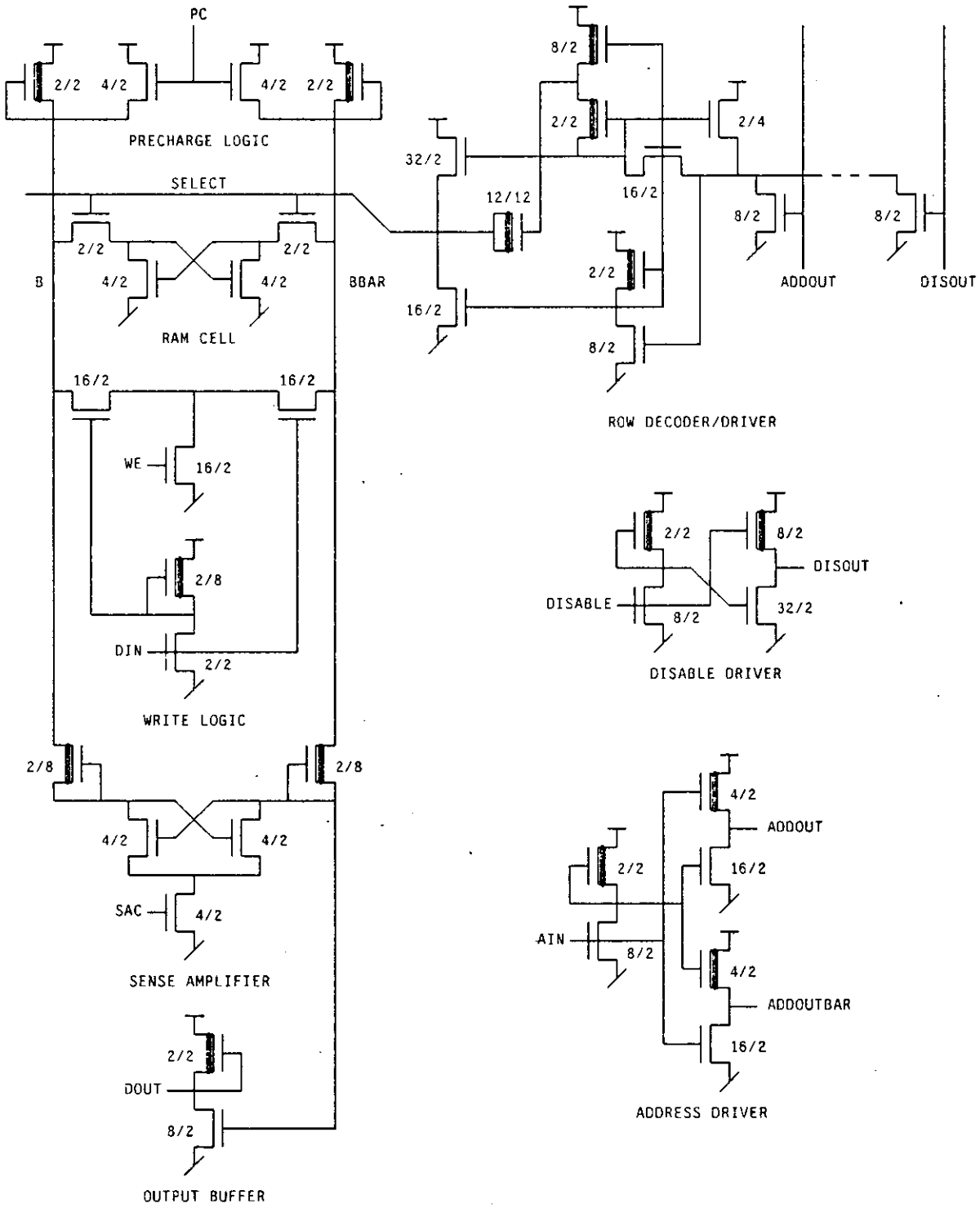

Figure 3-1: RAM Circuit Schematic

3.2. Column Precharge

The column precharge logic consists of a depletion and enhancement pullup transistor on each bit line in a column. The enhancement transistors are gated by the PC clock. During column precharge, the enhancement devices pulls the bit lines up rapidly. The depletion devices ensure that the bit lines are precharged to 5V. Typical simulated precharge time is 50ns. During sensing, the enhancement device is off, allowing the RAM cells to develop an adequate voltage difference on the bit lines.

3.3. Sense Amplifier

A sense amplifier is used to speed up readout. The sense amp is a standard cross-coupled pulldown pair, as is normally used in commerical dynamic RAMs. The sense amp inputs are connected to the bit line pair through shield resistors. These resistors allow the sense amplifier nodes to discharge without having to discharge the bit line capacitance, increasing speed and sensitivity. The sense amplifier discharges the node with the lower voltage when SAC goes high. The discharge time is 4ns. The RAM cell can provide a voltage difference of 1.4V 50ns after DISABLE goes low. Since the sense amplifier can detect voltage differences of 100mV or less, plenty of signal margin is provided. The sense amplifier is has a capacitance imbalance of 30% due to the output buffer input capacitance. This imbalance reduces sensitivity.

3.4. Write Logic

The write logic is implemented as pulldown transistors on the bit line pair in common with a pulldown transistor. The pulldown transistor is gated by WE. The other transistors are gated by the data input and its locally generated inverse. Writing is done by first precharging the bit lines, and then pulling one down. Pulldown time is 10ns.

3.5. Address Driver

The address driver is a pair of superbuffers generating complementary signals. The address line load is about 2pf and is driven high in about 20ns and low in 6ns.

3.6. Row Decoder/Driver

The row decoder is implemented as a NOR array. The pullup transistor is embedded in the row driver. All deselected rows are held low. One special address driver is connected to the DISABLE signal and holds all rows low when DISABLE is high. The row driver is a bootstrapped push-pull driver based on an INMOS design [Hardee 81]. The buffer drives the 1.6pf row select line high in 21ns when the row disable signal goes low. Rows are disabled with a delay of 16ns. A layout error resulted in the pullup and pulldown transistors in the buffer only half their design width. This has not had any

noticable effect on performance.

3.7. Output Buffer

The output buffer is an inverter connected to one side of the sense amplifier. It drivers 0.5pf high in 20ns and low in 15ns. The fall time is somewhat slow due to the slow precharge transition on the buffer input.

3.8. Read Cycle Timing

Read cycle timing is shown in Figure 3-2. A read cycle takes place as follows:

1. When PC and DISABLE are high, the bit lines are precharged, and the row decoder is disabled, forcing all select lines low. The sense amplifiers latch the previously sensed value while SAC remains high. This overlap of sensing and precharging is possible because the sense amp shield resistors allow a sense amp node to remain low when its corresponding bit line is high. The overlap is also possible because sensing is much faster than precharging, so the data can be latched in the sense amp before it is destroyed.

2. When PC goes low, precharging stops, and depletion-mode pullups maintain the bit lines at a high voltage. The read address has propagated through the address drivers so that the address lines have settled. DISABLE goes low, allowing the row decoders to activate, selecting one row. The row driver pulls this row high, allowing the selected RAM cells to begin discharging the bit lines. The sense amp resets in preparation for a new cycle.

3. When SAC, PC, and DISABLE go high, the sense amps latch the bit line value, the bit lines are precharged, and the select lines are forced low. The sensed value propagates through the output buffer.

3.9. Write Cycle Timing

Write cycle timing is shown in Figure 3-3. A write cycle takes place as follows:

1. When DISABLE and PC are high, the row decoder is disabled, the select lines are low, and the bit lines are precharged as during a read cycle. The state of the sense amplifiers is irrelevant during a write cycle.

2. When PC falls low, precharging stops. The write address has settled on the address lines. When DISABLE falls low, the appropriate row is selected.

3. The input data has propagated into the write logic. When WE goes high, this data is driven onto the bit lines by discharging one of the precharged lines in each column.

4. Writing stops when WE goes low. The depletion pullups begin charging the bit line that was just pulled low. Since the RAM cells are regenerative, they maintain their new data value while the bit lines are changing.

5. DISABLE and PC go high, deselecting the row and precharging the bit lines. SAC also

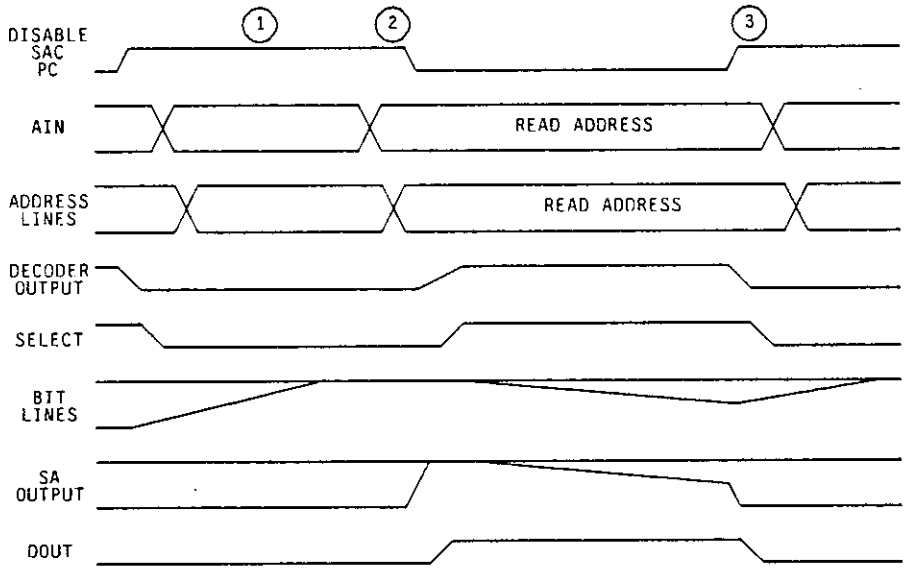

Figure 3-2: Read Cycle Timing

goes high, activating the sense amplifiers, which latch the just written data value.

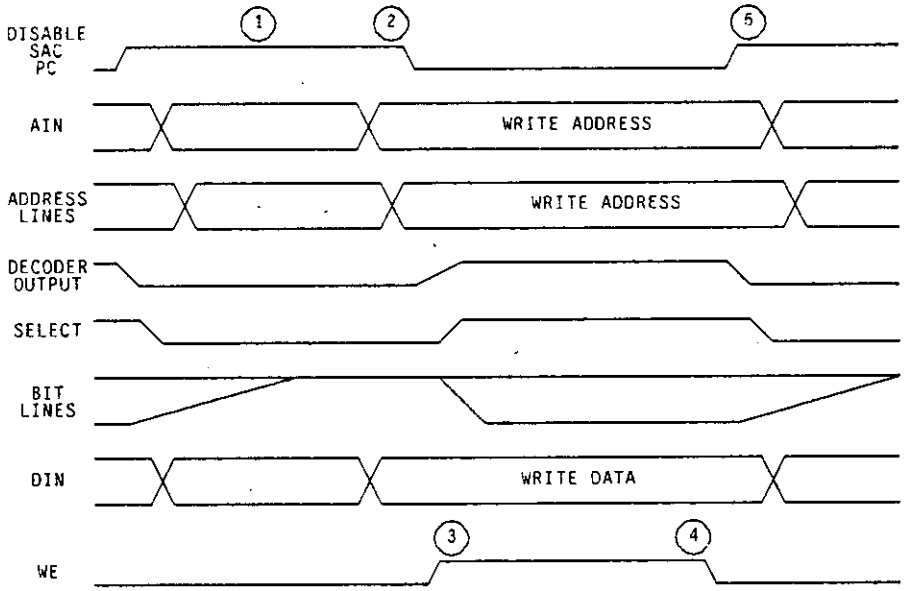

Figure 3-3: Write Cycle Timing

4. Layout Design

The layout of the four-transistor RAM cell is shown in Figure 4-1. The size is 15 lambda wide by 26 lambda tall.

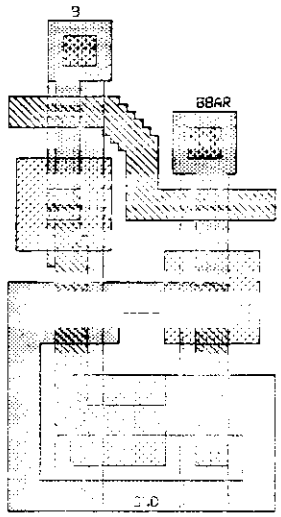

Figure 4-1: Four Transistor RAM Cell Layout

Note that the spacing rules for the buried contacts are more aggressive than the MOSIS rules. Rather than forcing all unrelated geometry to stay two lambda away from the buried region, the ordinary spacing rules are used except that poly and diffusion that are less than two lambdas apart must remain two lambdas away from the buried contact. As is discussed in Section 5, these rules do not result in significant yield loss for a lambda of 2.5 microns, but do result in poor yield for a lambda of 2.0 microns. These same rules were used on another RAM design using a lambda of 2.0 microns with no problems [Walker 83]. As will be seen below, the narrow pitch of the cell leads to layout difficulties in the associated peripheral circuitry.

A row decoder and word line driver is shown in Figure 4-2. The decoder and driver fit comfortably within the cell height. The driver sits to the left of the decoder, with the select line running through the decoder.

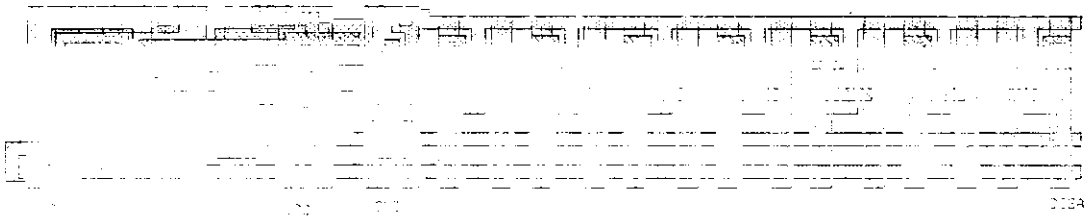

Figure 4-2: Row Decoder and Word Line Driver Layout

An address line driver layout is shown in Figure 4-3. The DISABLE driver is a single-output version of this design.

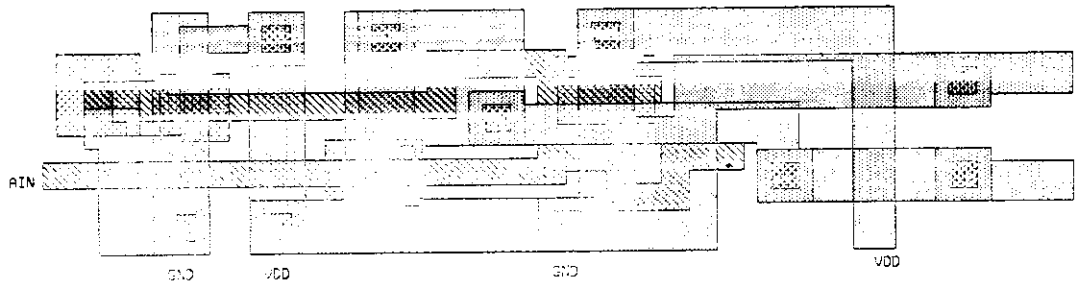

Figure 4-3: Address Line Driver Layout

The sense amplifier/write logic/output buffer layout is shown in Figure 4-4. Due to pitch constraints, the cells for odd and even columns must be stacked on top of one another. Despite this, the cell is still slightly larger than two RAM cell pitches. However metal ground lines run vertically through the RAM array every 4 cells to connect to the horizontal diffusion ground lines. This added overhead in the array means that the sense amplifier pitch matches the array pitch.

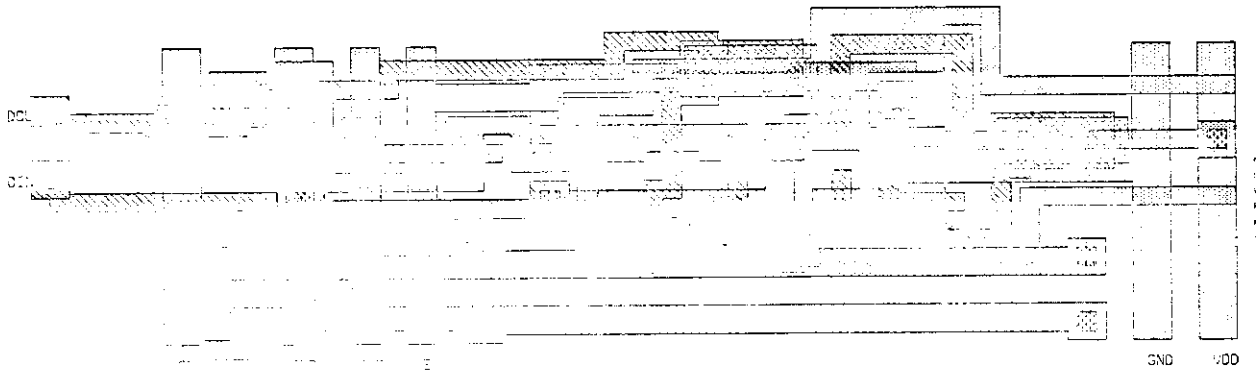

Figure 4-4: Sense Amplifier/Write Logic/Output Buffer Layout

The precharge logic for one column is shown in Figure 4-5.

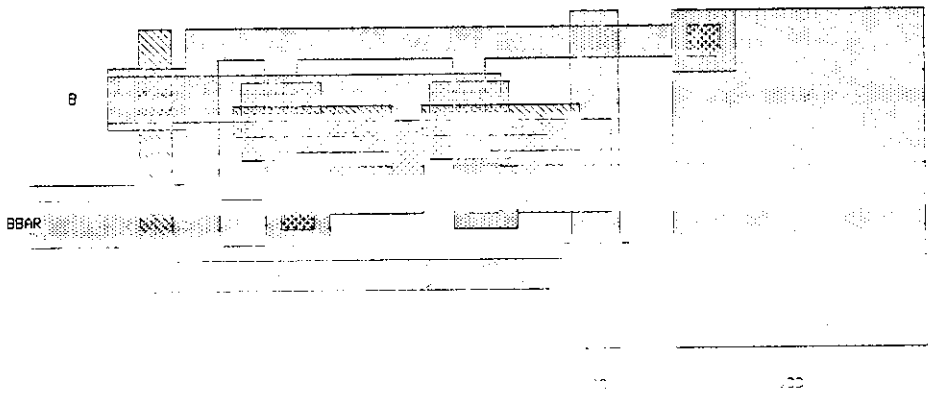

Figure 4-5: Precharge Logic Layout

A complete layout of the RAM test chip is shown in Figure 4-6.

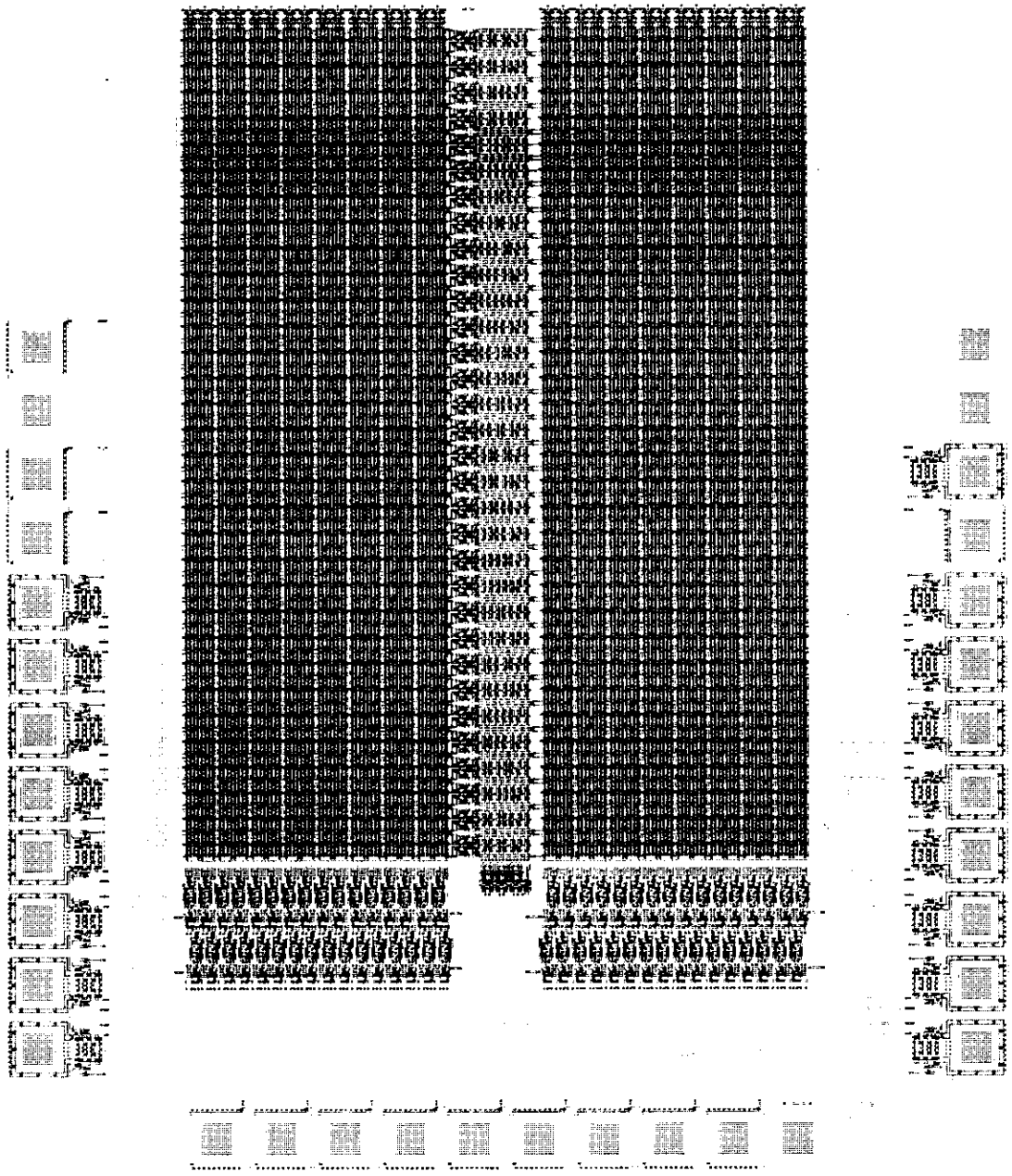

Figure 4-6: RAM Test Chip Layout

5. Experimental Results

A RAM test chip was designed and fabricated. This chip had the inner and outer four columns on each half of the array bonded out to pins, for a total of sixteen data I/Os. As noted previously, two versions were fabricated, ten chips using a lambda of 2 microns on lot M27O and eight chips using a lambda of 2.5 microns on lot M26L. All chips were tested at low speed, but only the M27O lot was tested at high speed.

5.1. Low-Speed Testing

Low-speed testing of the RAM test chips was done using a 68000-based single-board computer. C programs were used to read and write the chip pins through a parallel interface. Some lines had a TTL buffer inserted to reduce the interface rise and fall time of 50-100ns to 10ns. The average instruction took 5usec to execute, so the minimum clock pulse width was 10usec. Chips were tested at room temperature with a 5V supply and clock swings between 0V and 4V.

Three chips were fully functional in lot M26L. Fully functional in this case means that all of the sixteen visible columns worked. This also requires that the row decoder/drivers and select lines be functional. The other five chips have a variety of single-bit, partial column, partial row, full row, and full column failures.

None of the chips in lot M27O was fully functional. Three of the chips were completely dead, three were mostly dead, and the other four had a few bad bits. These results indicate that the yield is much too low for even our experimental purposes. This low yield is due to the aggressive buried contact rules used in the RAM cells.

5.2. High-Speed Testing

High speed testing of lot M27O was done using a Tektronix DAS9000 analyzer. This system has a minimum time resolution of 40ns, and the available clock pulse widths in the range of interest are 40ns, 50ns, 80ns, and 100ns. Testing was done at room temperature with a 5V power supply and clock waveforms swinging between 0V and 3.5V.

The chips worked (worked means that the good bits worked) with a minimum SAC/PC/DISABLE pulse width of 80ns, with a low time of 80ns, for a cycle time of 160ns. The chips did not work at a 100ns cycle time. The delay from the rising edge of SAC to data output is 25ns. The average current usage was 33.7mA active, and 21.7mA idle, for an active power dissipation of 158.5mW and idle dissipation of 108.5mW. Both the cycle time and power dissipation approximately agree with SPICE simulations. The columns that are not controllable have zero on their inputs. This is the maximum

power state, since the output buffer draws current. The chips worked at a refresh time of 40ms, but not 80ms at room temperature. This time is significantly shorter than the low-speed testing results, but the test setup was very noisy.

6. Conclusions

This section summarizes lessons learned from designing this memory. For other conclusions related to these, see [Walker 83].

6.1. Circuit Design

The four-transistor RAM cell has the advantage of providing a differential signal. This propoerty makes the cells easy to sense, write, and refresh. In contrast, the design of a three-transistor cell RAM was much more difficult [Walker 83]. Bootstrapped logic is used selectively to reduce power dissipation while maintaining speed in the row drivers. A much simpler design was later developed that trades a simple circuit design for a larger clock load [Walker 83]. In general, fancy circuit design should be used mainly to reduce power dissipation (but only when critical), increase chip functionality (by reducing layout area, allowing room for more logic), and to increase speed (if a large increase is possible). Note that these rules only apply to university design, not industrial design.

6.2. Layout Design

As noted in Section 4, the narrow column pitch made the peripheral circuitry difficult to lay out. Both stacking of the logic and area overhead due to ground lines had to be used to fit the sense amplfiers and write logic into place. In contrast, the row decoder/driver fit comfortably in the row pitch. A better approach might have been to make the RAM cell shorter and fatter, perhaps at some expensive in area. This area increase would be compensated by reduced peripheral logic area.

The aggressive buried contact rules allowed the RAM cell to be slightly smaller than if the standard MOSIS rules (which require all unrelated poly and diffusion to stay two lambdas away from the buried contact) were used. In retrospect this was a poor design choice since it resulted in substantial yield loss. On the other hand, these same rules were used in another design without problems [Walker 83].

6.3. Testing

Chip testing was by far the most difficult part of this project, representing half of the design time. Lessons were learned in the four basic areas of test programs, low-speed testing, high-speed testing, and on-chip test logic.

6.3.1. Test Programs

Our low-speed tester is a 68000-based single-board computer with parallel I/O ports. The device under test connects to these ports. We write and compile C programs on our VAX and download them to the board. This has proven to be an adequate method of low-speed testing. In particular, C makes an adequate test language when suitably extended with macros. The main advantage in using C is that it is our standard programming language.

We have discovered that a large fraction of the testing time is wasted by testing the wrong thing. We currently have no means of debugging our test programs. This frequently means that much time is wasted trying to discover why a chip is not working, only to find that the test program is incorrect. This discovery is often made by looking at output waveforms on an oscilloscope. The solution to this problem is to write timing simulators so that test programs can provide input stimulus and receive outputs from the simulator. This allows the test program to be debugged on a "design" that is defined to be correct.

Another major time waster is that we often go through a whole series of test programs. The designer is naturally impatient and writes a simple program to test basic chip functions. This usually verifies that the design is correct, but is not sufficient to check whether the chip is fully functional. This problem is particularly common with memory designs. In the end the designer must write a complete test program, and the total effort spent is larger than if these programs had been written at the beginning of testing. These complete test programs should be written while the design is being fabricated.

6.3.2. Low-Speed Testing

We have discovered a few limitations of our low-speed tester. Until recently we were using a 4MHz 68000, which often meant that test programs were too slow to adequately refresh dynamic logic. This problem has been solved by the use of higher-speed processors. Another limitation is the limited number of I/Os, 32 in this case. This is frequently insufficient to look at all data inputs and outputs at once. For high pin-out chips, a test board with the required multiplexers is used.

The setup time for the low-speed tester is relatively long. This is mainly due to the wiring connecting the test head to the chip on the protoboard. We have designed a programmable interface with a large number of I/Os that connects to every pin in a variety of sockets. This allows the configuration to be done in software.

6.3.3. High-Speed Testing

We have been using a Tektronix DAS9000 for high-speed testing. We have found that the DAS is extremely poorly suited to this task, and would strongly recommend that anyone considering purchasing a DAS for this application buy something else. If you already own a DAS, the best course would be to sell it. The main problems are the inflexibility of the pattern generator, and the small size of the pattern generator and data acquisition memory.

The pattern generator has only simple commands, such as branch to subroutine, unconditional branch, and hold for a certain number of cycles. There is no concept of a conditional branch, so FOR and WHILE loops cannot be implemented. All input data to a chip must be embedded in the test programs. Since there are only 254 memory locations, it is often impossible to load a chip with all the required data. This problem is particularly bad if the chip uses a shift register for data input.

In addition to these problems, the pattern generator outputs can only be tristated in groups of 16 pins, which means that for many of our chips, we have had to impose extra tristate buffers between the DAS and the chip, or to buy a pattern generator extension that provides a group of separately controllable tristate pins. This problem stems from the fact that the DAS was designed for debugging bus-oriented systems, while we need arbitrary bit manipulation capabilities.

We also have no software to interface the DAS to our VAX. Since our DAS does not have a tape drive, test programs must be keyed in by hand, and all data is lost on power-down. The test language is also unique to the DAS, so the system does not fit into our design environment.

The ideal solution to our high-speed test problems would be an ultra-fast microprocessor running C. Since this doesn't exist, and we cannot afford to buy a real integrated circuit test system, we are designing our own high-speed tester. The tester combines the flexible interfacing mentioned above with the ability to read and write patterns at high speed. The design is similar to that used in the DAS, where a microsequencer reads out a pattern memory, and stores results. Our microsequencer will be flexible enough to call procedures, implement conditional loops, and other desirable functions. Most of our chips cannot operate at very high speed. We feel that a cycle time of 100ns should be adequate for our needs. We do often need higher edge resolution than 100ns. Our tester chips therefore include programmable delay lines that provide 10ns resolution.

6.3.4. On-Chip Test Logic

A little bit of on-chip logic would greatly reduce the tester requirements. For example, timing samplers [Frank 81] allow high-speed snapshots of logic using low-speed equipment. Shift registers gain visibility that greatly simplifies test pattern generation and fault diagnosis. The area cost is small, so this logic can frequently be left in the production version and used for testing production chips (production in our case is several hundred chips). Care must be taken to minimize pinout requirements though. It has been our experience that we have plenty of spare pins for testing because we must use an over-sized package. For example, the PSC is mounted in an 84-pin leadless chip carrier, but only uses 74 pins including test pins. The PSC cannot use 64 pins because it needs 64 pins just for essential data, power, and clock lines. The PSC might squeeze into 68 pins, but the package must have a 400mil square cavity, and many 68-pin packages do not.

6.4. Summary

This paper has described the design of a 4-Kbit, four-transistor dynamic RAM. The lessons learned in the course of this work should reduce the design time for similar projects by 25% or more.

7. Acknowlegements

This work was started when H. T. Kung suggested that I design a memory for the PSC project [Fisher 83].

8. References

[Cohen 81] D. Cohen and G. Lewicki.
MOSIS - The ARPA Silicon Broker.
In C. L. Seitz (editor), *Proceedings of the Second Caltech Conference on VLSI*, pages 29-44. California Institute of Technology, Pasadena, CA, January, 1981.

[Dohi 83] Y. Dohi, A. L. Fisher, H. T. Kung, and L. M. Monier.
Architecture of the PSC: a Programmable Systolic Chip.
In *The 10th Annual International Symposium on Computer Architecture*. June, 1983.

[Fisher 83] A. L. Fisher, H. T. Kung, L. M. Monier, H. Walker, and Y. Dohi.
Design of the PSC: A Programmable Systolic Chip.
In R. Bryant (editor), *Proceedings of the Third Caltech Conference on VLSI*, pages 287-302. Computer Science Press, Rockville, MD, March, 1983.

[Frank 81] E. Frank and R. Sproull.
Two Timing Samplers.
In C. Seitz (editor), *Proceedings of the 2nd Caltech Conference on VLSI*, pages 127-134. Caltech, Computer Science Department 256-80, Pasadena, CA 91125, January, 1981.

[Hardee 81] Kim Hardee, and Rahul Sud.
A Fault-Tolerant 30 ns/375 mW 16K x 1 NMOS Static RAM.
IEEE Journal of Solid-State Circuits SC-17(5):435-443, October, 1981.

[Walker 83] Hank Walker.
The Control Store and Register File Design of the Programmable Systolic Chip.
Technical Report, Carnegie-Mellon University, Computer Science Department, May, 1983.