CMU-CS-83-133

THE CONTROL STORE AND REGISTER FILE DESIGN OF THE PROGRAMMABLE SYSTOLIC CHIP

Hank Walker

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

28 May 1983

Abstract

The design of a 64 word by 60-bit writeable control store and a 64 word by 9-bit register file is described. A dynamic three-transistor NMOS RAM cell is used to meet area constraints. Sense amplifiers reduce access time and bootstrapped logic reduces power dissipation. A shift register is used to load the control store. Experimental results indicate a typical read/write cycle time of 160ns and power dissipation of 150mW for the control store.

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.

Table of Contents

List of Figures

1. Introduction

The Programmable Systolic Chip (PSC) is a custom VLSI processor [Fisher 83, Dohi 83] designed to act as an element in a systolic array [Kung 80]. The PSC is a user-microprogrammable processor consisting of an 8 x 8 parallel multiplier/accumulator, an 8-bit ALU, three 9-bit input ports, three 9-bit output ports, and a 64 word by 9-bit register file. These units are connected by three 9-bit buses for high parallelism, and are controlled by a microsequencer and 64 word by 60-bit writeable control store (WCS). The PSC is designed to be fabricated through the MOSIS ARPA implementation service [Cohen 81]. This paper describes the design of the control store and register file.

An ISP [Barbacci 81] description and microcode simulator were developed for the PSC and used to code several important algorithms. From this work came the requirement that the control store contain 64 60-bit words. The system organization also requires that the WCS be loaded through a shift register. This shift register also improves testability since the rest of the PSC can be tested even if the control store does not work:

The original die size goal was 6.5mm x 6.5mm. The smallest six-transistor static RAM cell available in March 1982 was 24 x 29 lambda in area, resulting in a control store size of 1730 x 2050 lambda, which was too large for our application. A four-transistor dynamic cell design was considered [Walker 83]. This design had a 15 x 26 lambda cell and 1200 x 2000 lambda overall size. This RAM was also too large, and as it later turned out, too long. The only design that met the area requirements was a three-transistor dynamic RAM cell design. The cell has an area of 12.5 x 18 lambda, and the overall area is 1000 x 1625 lambda. This area is 68% of the area of the four-transistor design and 46% of the area of the static design. The actual size of the PSC is 7mm x 9mm. The memory height determines the long dimension of the chip. Since 9mm is already the maximum size allowed by the packaging, a larger control store will not fit.

A second reason for using the dynamic design was power consumption. A static RAM control store would consume 750mW or more [Frank 83]. Combined with the multiplier (200mW), a static register file (150mW), eight bus transceivers (50mW), 28 output pads (250mW), and other logic, this would have resulted in a typical chip power dissipation of more than 1.5 Watts, exceeding the package rating. Because many PSCs will be put together to form arrays, low power dissipation is especially important. The dynamic control store has a typical power dissipation of 150mW, keeping the chip power dissipation below 1.0 Watt.

The register file was originally specified to be 128 9-bit words. In order to save design time, it was decided to use the same basic circuit and layout design as the control store. However, a register file organized as a 128 x 9 array would be 320 x 2720 lambda in size, which is far too long. A convenient array shape would be 64 x 18, to match the height of the control store. However, the three-transistor RAM cell does not permit easy multiplexing of the outputs without using read-modify-write cycles. Instead it was decided to reduce the register file size to 64 9-bit words for the first PSC implementation.

2. Functional Description

2.1. Control Store

A block diagram of the writeable control store is shown in Figure 2-1. The signals to the memory include address input (AIN), data output (DOUT), VDD, GND, refresh write enable (REFRESHWE), shift register data in (SRIN), shift register data out (SROUT), RAM clocks (RAMPHI1, RAMPHI1BAR, and RAMPHI2), and the shift register clocks (SRPHI1 and SRPHI2). A write cycle takes place when WE is high on the rising edge of RAMPHI1. This edge initiates the read or write cycle. The address inputs are transparent on RAMPHI2. The bit lines are sensed on the rising edge of RAMPHI1BAR and precharged on RAMPHI2. The output latch is transparent when SRPHI2 is high. SRPHI1 is low except during shift register operation. With the data outputs numbered from left to right as 0 to 59, the shift register sequence from input to output is: 58, 57, 54, 53, 50, 49, 46, 45, 42, 41, 38, 37, 34, 33, 30, 29, 26, 25, 22, 21, 18, 17, 14, 13, 10, 9, 6, 5, 2, 1, 0, 3, 4, 7, 8, 11, 12, 15, 16, 19, 20, 23, 24, 27, 28, 31, 32, 35, 36, 39, 40, 43, 44, 47, 48, 51, 52, 55, 56, 59. This sequence was determined by layout considerations.

The system timing of the PSC is shown in Figure 2-2. The buses are precharged, the logic blocks compute, and the memories are accessed on clock phase 1. The buses are used and the logic blocks and memories are precharged on clock phase 2. Condition codes become available at the end of phase 1, and propagate through the microsequencer during phase 2 so that a new microinstruction address is ready at the beginning of the following phase 1. This address is in turn used to access a microinstruction that becomes available on phase 2. To the microcoder, this looks like a one-cycle branch delay.

As can be seen in the diagram, RAMPHI1 is the same as the system phase 1 during ordinary operation. RAMPHI2 and SRPHI2 go high before system phase 2 does. The reason for this is that the microinstruction must be stable before it can be used by the bus transceivers on phase 2.

Figure 2-3 shows the timing used for loading a word into the shift register and writing it into the

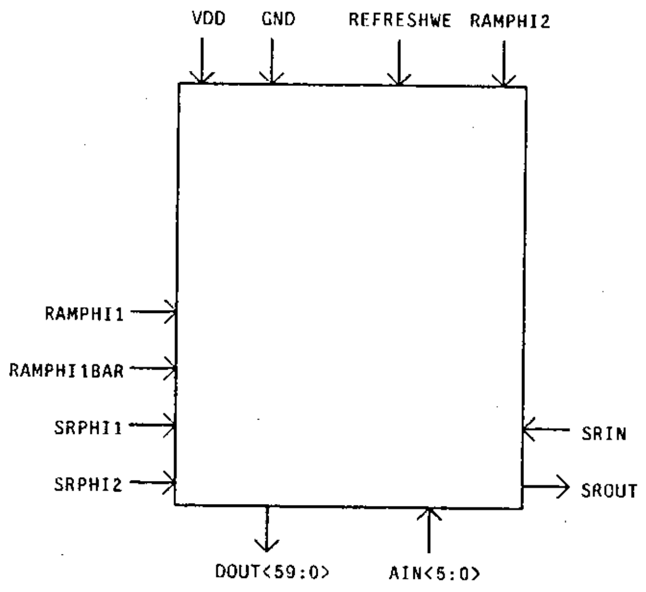

Figure 2-1: Writeable Control Store Block Diagram

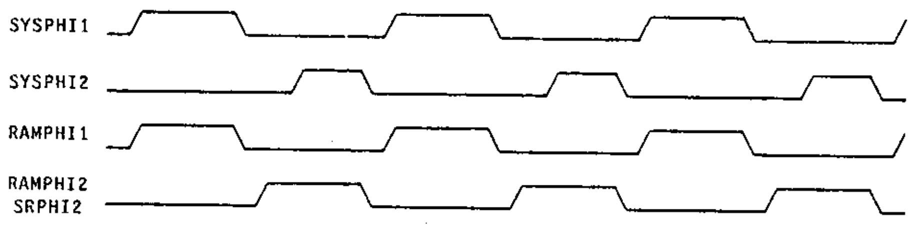

Figure 2-2: PSC System Timing

memory. This can be done without disturbing the rest of the chip logic since the WCS control lines are separate from the system clocks. Ordinarily this is of no consequence since loading occurs at boot time, but it does make testing much easier. The address used during microcode loading and refresh cycles is generated by an on-chip counter that is incremented on the falling edge of the REFRESH signal. A reset line is provided to synchronize the counters in an array.

Figure 2-4 shows the timing for a refresh cycle and neighboring compute cycles. Refresh of all of the chips in a systolic array is done in parallel. Since the system clocks are inactive, refresh does not change the chip's state, and so remains invisible to the programmer. One design alternative that was considered was to include a special refresh microinstruction field that would be executed at the end of the major microcode loop. But since a typical loop is less than ten instructions long, and as small as one instruction, this refresh technique would increase execution times by 10-100%, and require more on-chip clock logic, and was therefore rejected.

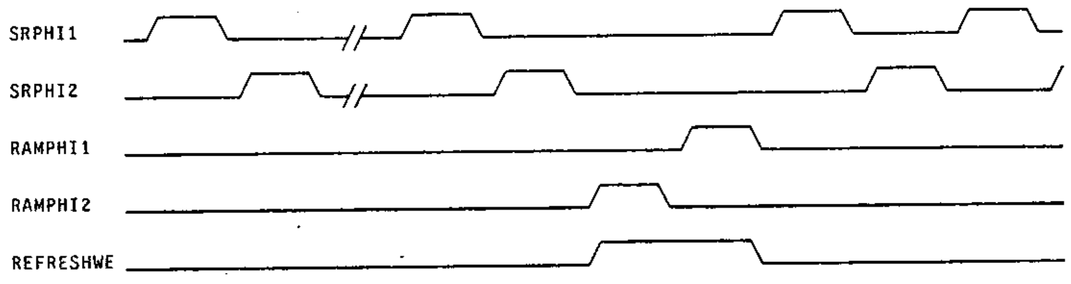

Figure 2-3: Shift Register Load and Write Cycle

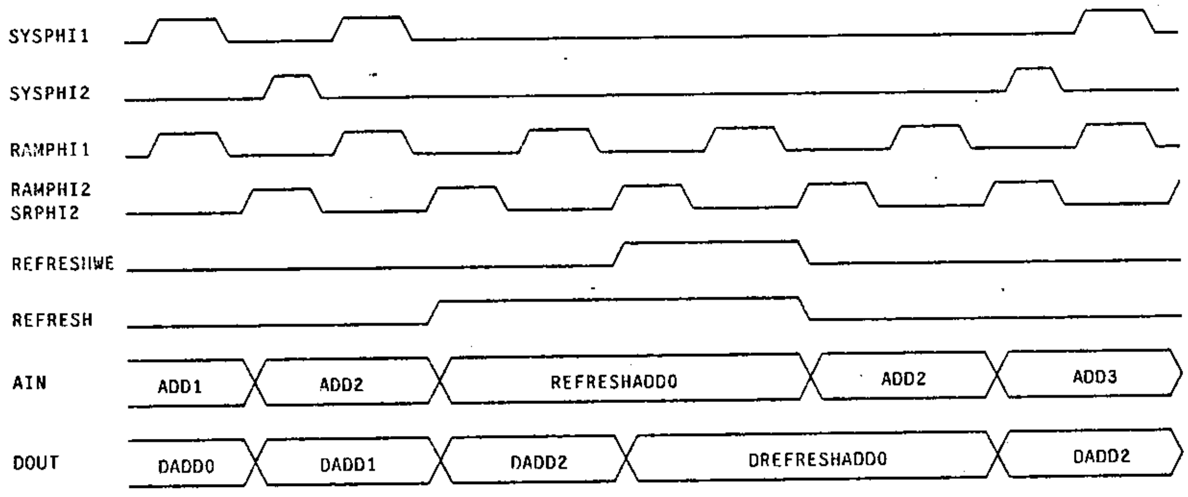

Figure 2-4: WCS Refresh Cycle

The control store timing described above is that used in the third pass of the PSC. The design used in the first two passes is described in Section 6.

2.2. Register File

A block diagram of the register file is shown in Figure 2-5. The signals are similar to, and in many cases the same as those used by the control store. The signals include VDD, GND, address input (AIN), data input (DIN), data output (DOUT), microcode write enable (MCWE), refresh write enable (REFRESHWE), REFRESH, RAMPHI1, RAMPHI1BAR, and RAMPHI2.

The register file does not include a shift register on its outputs. Instead it has logic for writing data

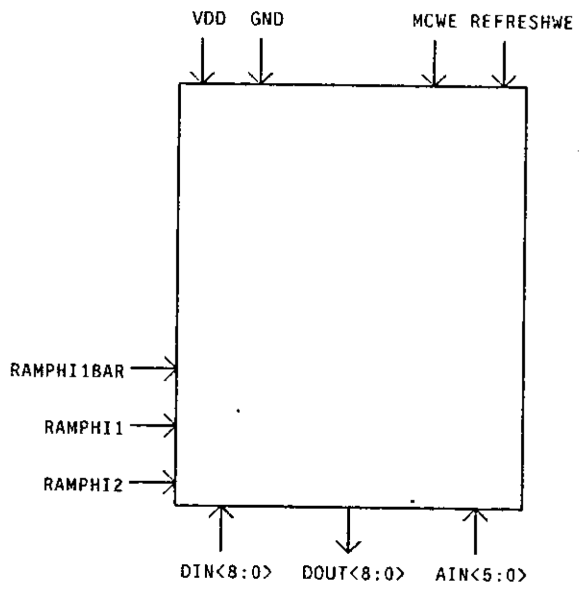

Figure 2-5: Register File Block Diagram

from a bus transceiver. Since the register file is under control of microcode, there is a separate write enable for microcode and for refresh. The REFRESH signal is used to indicate that data output should be fed back into the data input, and that REFRESHWE should be used instead of the MCWE.

The register file timing is the same as that of the control store. As is the case for the control store, the third pass of the PSC has slightly modified register file timing.

3. Circuit Design

3.1. Control Store

The control store circuits consist of seven basic blocks: A three-transistor RAM cell, column precharge, sense amplifier, shift register/output buffer, address driver, row decoder/driver, and clock logic. Schematic diagrams for these blocks are shown in Figures 3-1 and 3-2. The transistor sizes are given as width/length in lambda units. The PSC is fabricated with a lambda of two microns. As mentioned above, the original memory design was somewhat different than the final design, and is discussed in Section 6.

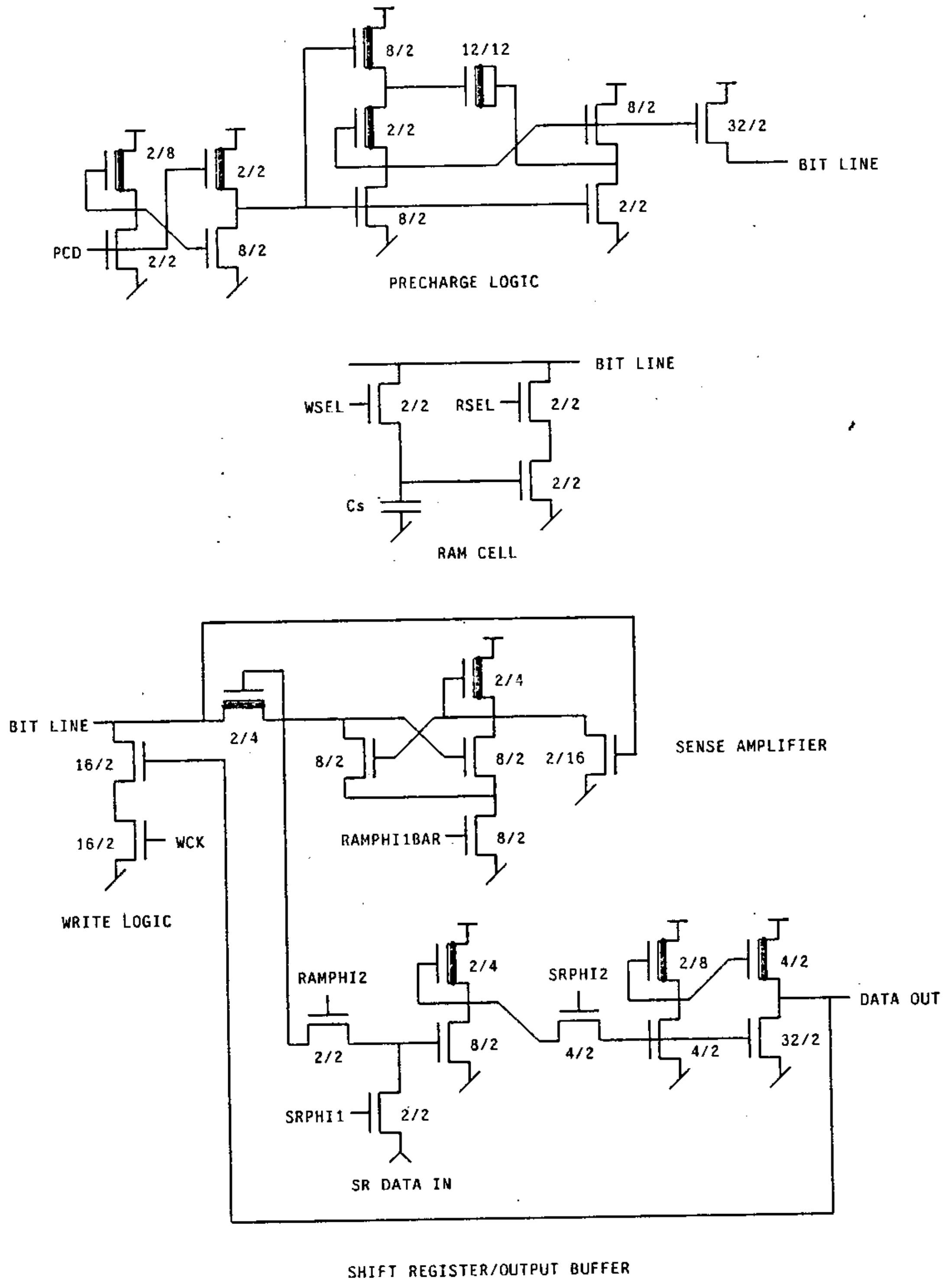

Figure 3-1: Control Store Circuit Schematic

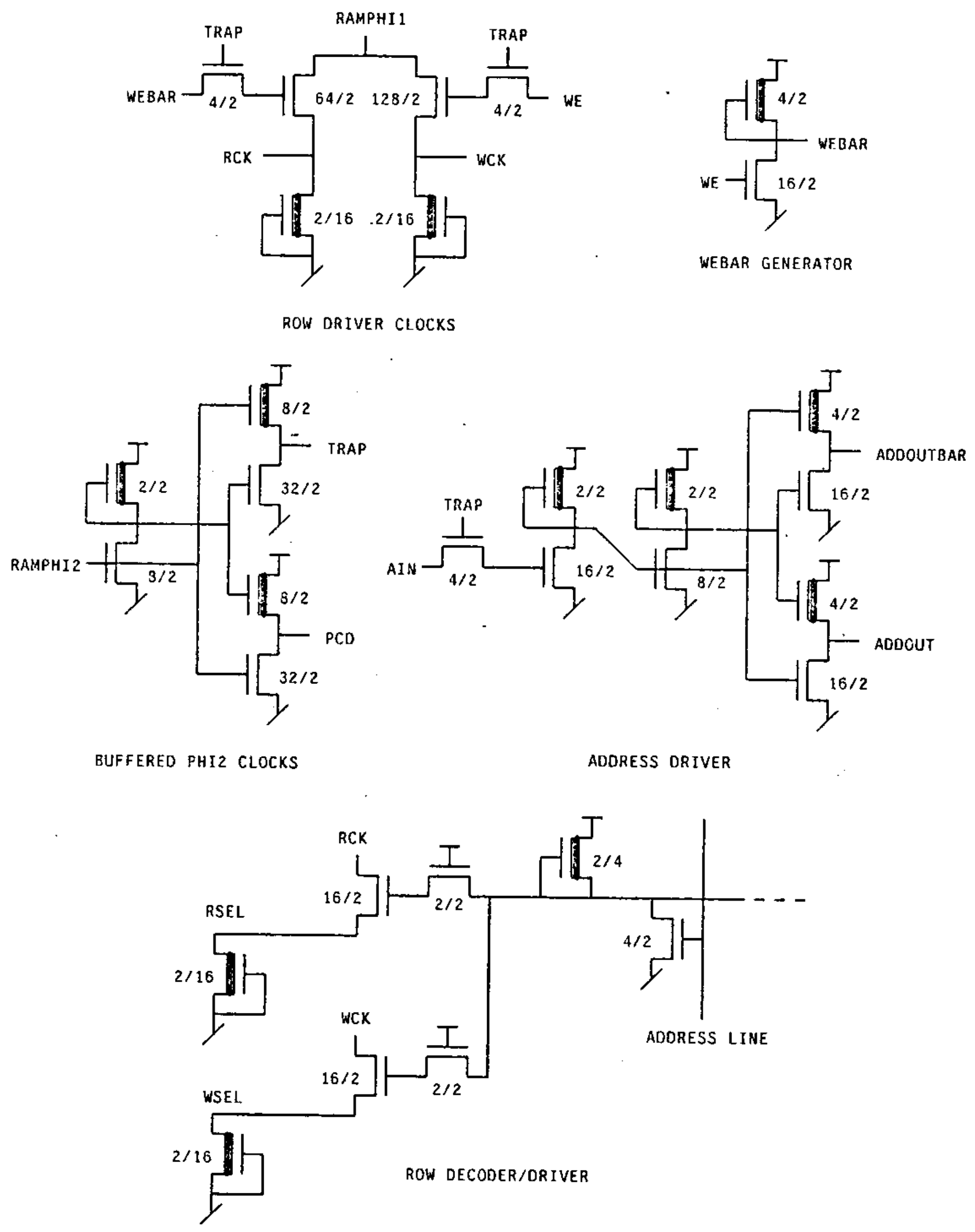

Figure 3-2: Control Store Circuit Schematic cont.

3.1.1. RAM Cell

The three-transistor RAM cell operates by having charge stored on the gate of the pulldown transistor. If 3.5V is stored on the gate, the cell will pull down the precharged bit line when read-selected. If 0V is stored in the cell, then the bit line will remain high. When the cell is write-selected, the bit line value is stored into the cell. The read and write select signals are driven high during RAMPHI1. The storage capacitance of the cell is 0.015pf. The bit line presents a load of about 1.2pf, and the cell can pull it down from 5V at a rate of 16ns/volt.

3.1.2. Column Precharge

The column precharge logic is an enhancement pullup transistor connected to a bootstrapped driver. The column is precharged when the input to the precharge logic is low, which occurs during RAMPHI2. (The active low circuit is a remnant of the original design.) Typical simulated precharge time is 15ns.

3.1.3. 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 input is connected to the bit line through a shield resistor. This allows the sense amplifier node to discharge without having to discharge the bit line capacitance, increasing speed and sensitivity. The reference node is connected to a very low ratio inverter. The inverter input is connected to the bit line, so that the reference voltage typically swings between 4.3V and 4.6V as the bit line falls from 5V to 4V. This provides a 0.6V voltage difference to sense. The sense amplifier discharges the node with the lower voltage when RAMPHI1BAR goes high. The discharge time is 4ns. During sensing, the pass transistor to the output buffer is turned off. This isolates the sense amplifier from the input capacitance of the buffer. This isolation more nearly equalizes the capacitance on each node of the sense amp, increasing sensitivity.

3.1.4. Shift Register/Output Buffer

The shift register is implemented as a standard two-phase dynamic shift register. The output buffer is formed by setting SRPHI1 low and parallel loading through an alternate path. The output driver and latch is formed by the second stage of the shift register cell, which is a large superbuffer. The output buffer can drive the 2pf microinstruction bus load in 20ns. Since the buffer also feeds the next stage of the shift register, this delay sets the limit on shift register clocking speed at about 20MHz. If shift registers in an array of chips are linked serially, the 30ns output pad delay would set the limit on operating frequency. In practice the clock speed will probably be 10MHz or less.

3.1.5. Write Logic

The write logic is implemented as a series pair of pulldown transistors. One gate is connected to the data output, and the other to the write clock (WCK). The bit lines are precharged, and then pulled down if 0V is to be stored into the cell. Pulldown time is 8ns.

3.1.6. Address Driver

The address driver is a pair of superbuffers generating complementary signals and includes an input latch. The address line load is about 1.3pf and is driven in about 9ns.

3.1.7. Row Decoder/Driver

The row decoder is implemented as a NOR array with a 22ns delay from address line input to the row driver output transistors. All deselected rows are held low. The row decoder output passes through isolation devices to the drive transistors for both the read and write lines for a word. The read line transistor is connected to the read clock (RCK) and the write line transistor is connected to WCK. The transistors in the selected row have a high voltage on their gates, so the rising edge of RCK or WCK bootstraps the corresponding drive transistor, allowing the word line to rise to V_{dd} with a delay of about 4ns on the read line and 5ns on the write line. The load presented by the word lines is a distributed RC with 0.8pf capacitance and 10k ohm resistance, with the driver in the middle of the line. Deselected word lines are held low by a pulldown resistor. This resistor presents a DC load of about 15uA on the read or write clock, and in turn on RAMPHI1.

3.1.8. Clock Logic

The clock logic generates several internal memory clocks from external clocks and control signals. The generated signals are WCK, RCK, TRAP, and Precharge Delayed (PCD). The input signals used are RAMPHI1, RAMPHI2, and REFRESHWE, and the RCK and WCK signals are gated versions of RAMPHI1. The gating devices are bootstrapped pass transistors. When REFRESHWE is high, WCK is activated, otherwise RCK is generated. TRAP is a buffered version of RAMPHI2 used to latch both REFRESHWE and AIN. PCD is not actually a delayed signal, but is RAMPHI2 inverted. The buffered signals and their names exist due to historical reasons. The RCK and WCK generator has a delay of 2ns driving loads of 1.3pf and 6pf respectively (excluding the word line). Note that these loads exceed that of the word line. Both the RCK and WCK lines have a pulldown resistor on them to clamp them low when they are inactive. This presents a DC load on RAMPHI1 of 15uA in addition to the load from the selected word line.

3.1.9. Read Cycle Timing

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

1. RAMPHI2 rises high, causing TRAP to go high, PCD to go low, the bit lines to be precharged, sampling new REFRESHWE and AIN values, and parallel loading the shift register with the data read on the previous cycle. REFRESHWE is low since this is a read cycle. The new address is driven onto the address lines by the address drivers. All but the selected row decoder output goes low. In phase with RAMPHI2, SRPHI2 rises high, allowing the data in the shift register to enter the output buffer and be driven onto the microinstruction bus.

2. RAMPHI2 falls low, causing TRAP to go low, PCD to go high, precharging to stop (and the bit lines to float), latching the AIN and REFRESHWE inputs, and shift register parallel inputs. At the same time, SRPHI2 falls low, latching the microinstruction in the output buffer.

3. RAMPHI1 goes high, causing RCK to rise high, which in turn drives the selected word read line (RSEL) high. Selected RAM cells with a high voltage stored in them begin discharging their bit lines. In parallel with RAMPHI1 going high, RAMPHI1BAR goes low, resetting the sense amplifiers.

4. RAMPHI1 goes low, causing RCK to go low, deselecting the word, allowing the bit lines to float. RAMPHI1BAR goes high, activating the sense amplifiers.

3.1.10. Write Cycle Timing

Write cycle timing is shown in Figure 3-4. A write cycle is similar to a read cycle with the following exceptions:

1. RAMPHI2 goes high, causing a high REFRESHWE input value to be read since a write cycle is to take place. If the data to be written is coming from a memory location (as in a refresh cycle), then SRPHI2 goes high in phase with RAMPHI2, allowing the data in the shift register to enter the output buffer. This data is driven out on the microinstruction bus and to the data input logic. If the data was loaded via the shift register, then SRPHI2 remains low, since the correct data is already present in the output buffer and the write logic. Refer to Figures 2-3 and 2-4.

2. RAMPHI2 goes low, latching the high REFRESHWE value.

3. RAMPHI1 goes high, causing WCK to rise high, driving the selected word write line (WSEL) high, and activating the write logic. Bit lines with a high voltage on their write logic are discharged, storing a low voltage in the selected RAM cells. Bit lines with a low

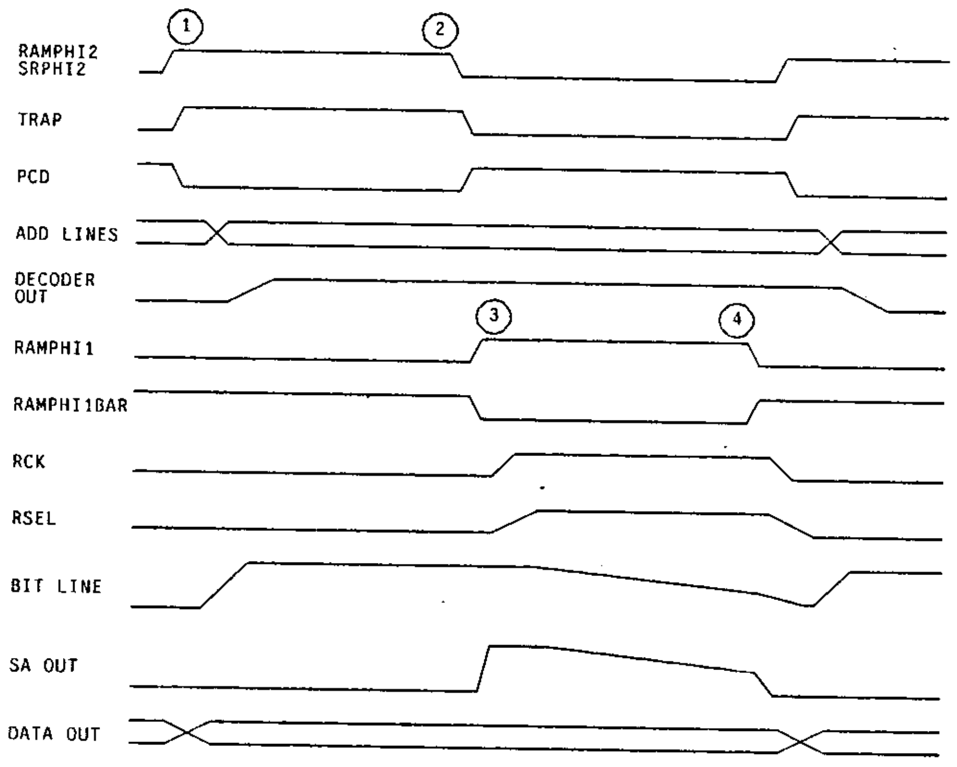

Figure 3-3: Control Store Read Cycle

voltage on their write logic float high, and store this high voltage in the selected RAM cells. In parallel with RAMPHI1 going high, RAMPHI1BAR goes low, resetting the sense amplifiers.

4. RAMPHI1 goes low, causing WCK to go low, deselecting the word. RAMPHI1BAR goes high, activating the sense amplifiers. Sense amplifier operation serves no useful purpose during write cycles, and does not affect operation of other circuits.

5. RAMPHI2 does not go high again until RAMPHI1 has been low long enough to completely discharge the write select line. This prevents precharge from overwriting a just-written low voltage with a high voltage.

3.1.11. Refresh Cycle Timing

A refresh cycle is implemented by doing a read cycle, latching the result in the output buffer, and then writing it back, as discussed in the previous section and as shown in Figure 2-4.

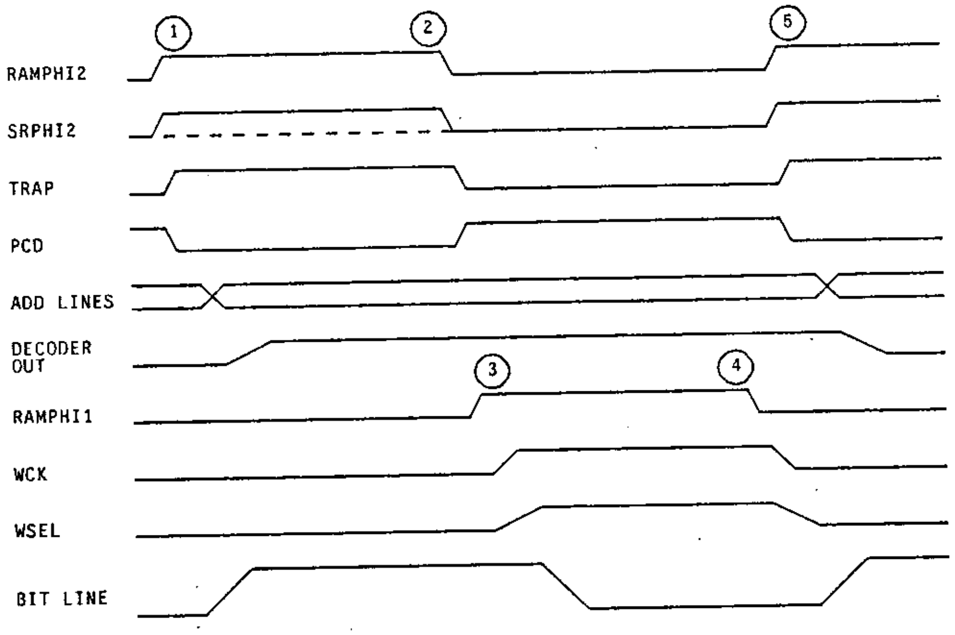

Figure 3-4: Write Cycle Timing Diagram

3.2. Register File

The register file circuits are the same as those in the control store, except that the sense amp output has been modified; there is only a simple output buffer rather than a shift register/output buffer, and there is a set of data input/output multiplexing logic. The clock logic has also been modified to select either the REFRESHWE or MCWE inputs. Schematics of the modified circuits are shown in Figure 3-5. Timing is the same as that for the control store.

3.2.1. Sense Amplifier

The sense amp has been modified so that it is no longer shielded from the output buffer input capacitance. The output buffer is smaller since it must only drive a bus transceiver rather than the microinstruction bus. The lower input loading does not upset the capacitance balance of the sense amplifier nodes.

3.2.2. Output Buffer

The shift register/output buffer in the control store has been replaced by an inverter pair and has a delay of 20ns.

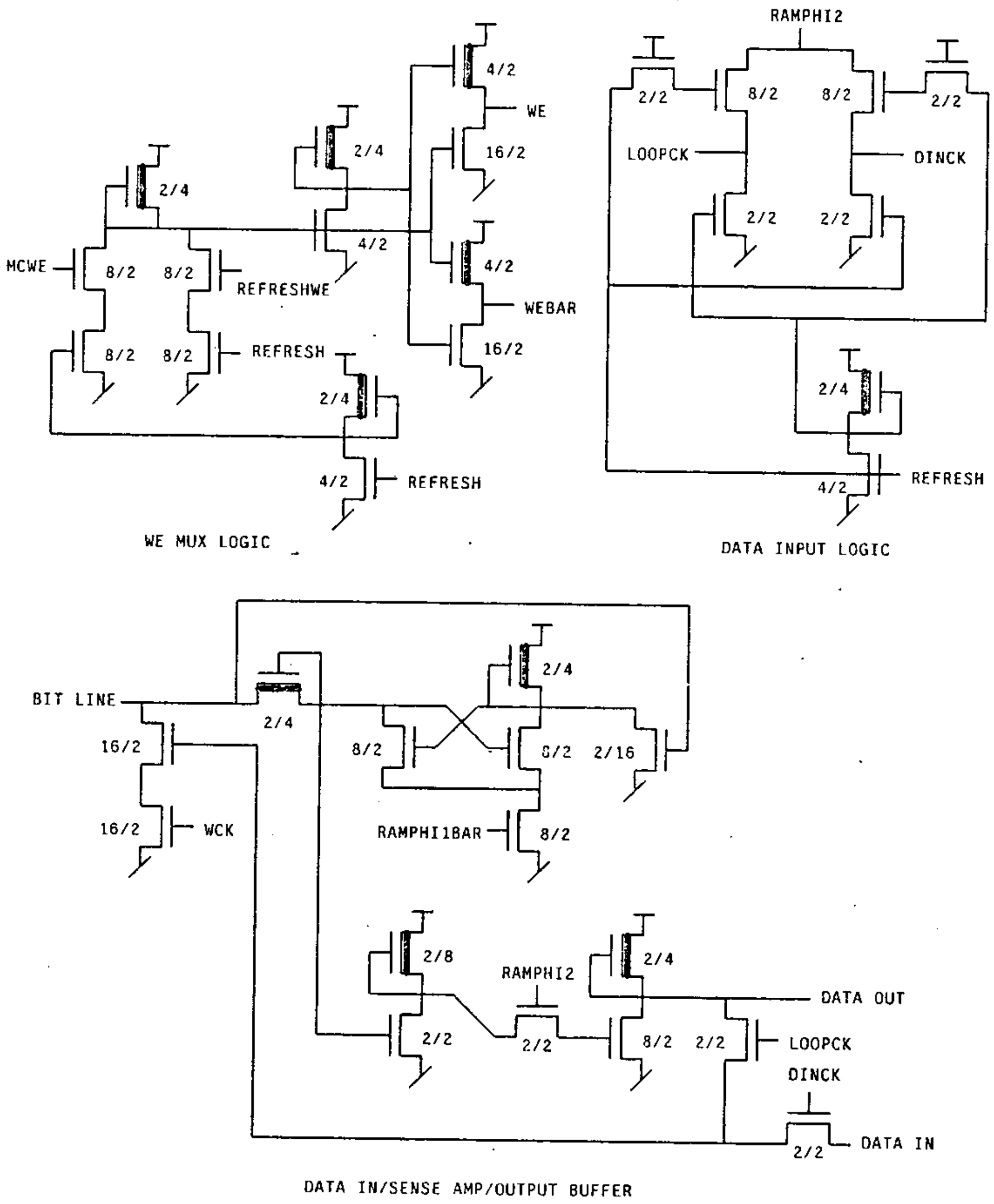

Figure 3-5: Register File Circuit Schematics

3.2.3. Input Logic

The data input comes either from the bus transceiver during write cycles, or from the data output during refresh cycles. Input multiplexing logic determines the source based on the REFRESH input. Because the data input is through a pass transistor, it takes longer for the write logic to pull down the bit line than in the control store; 18ns in this case.

3.2.4. Clock Logic

The clock logic has been modified so that when REFRESH is high, the REFRESHWE input is used, and when REFRESH is low, the MCWE input is used. Because the word lines are much shorter, and there are fewer write buffers in the register file, the RCK and WCK delays are only 2ns. This however, does not significantly change the cycle time.

4. Layout Design

The layout of the three-transistor RAM cell is shown in Figure 4-1. The size is 12.5 lambda wide by 18 lambda tall.

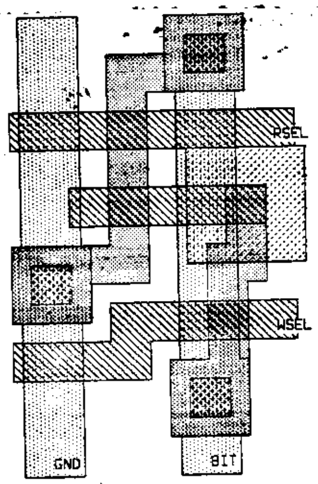

Figure 4-1: Three Transistor RAM Cell Layout

Note that the spacing rules for the buried contact are more aggressive than those specified by MOSIS. 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 lambda apart must remain two lambda away. As will be seen below, the narrow pitch of the cell leads to difficulties in laying out associated peripheral circuitry. In addition, the pitch leads to a tall and narrow control store, when a short fat one would be more desirable.

A row decoder and word line driver is shown in Figure 4-2. The decoder fits comfortably within the cell height. The drivers have difficulty fitting; thus there are a few small spacing violations to fit them in while maintaining a reasonable size. Note that the diffusion area connected to the WCK and RCK lines has been minimized to reduce what will already be high capacitance on the clock lines. Placing a separate word line driver on each side of the decoder would ease layout problems but result in larger area and higher capacitance on the clock lines.

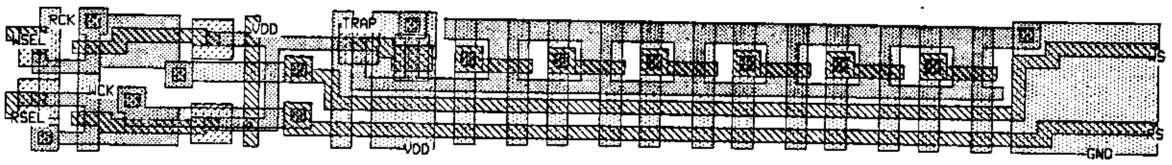

Figure 4-2: Row Decoder and Word Line Driver Layout

An address line driver layout is shown in Figure 4-3. The address driver for the least significant bit is slightly different to accomodate wiring at the right side of the decoder. Because the word line drivers are placed on the left of the decoder, the address drivers are placed next to the sense amplifiers of the right memory array. This leads to difficulties in routing control lines.

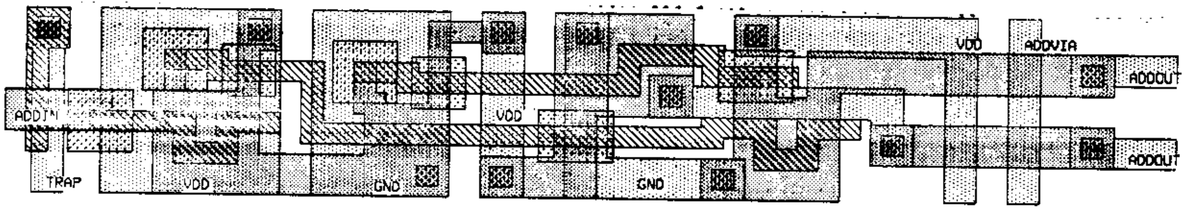

Figure 4-3: Address Line Driver Layout

The sense amplifier/shift register/output buffer layout for one bit line is shown in Figure 4-4. Due to pitch constraints, the cells for the odd and even bit lines must be stacked on top of one another. Despite the use of this stacked arrangement, several small spacing violations were still used to keep the cell size down. The stacking also results in these blocks accounting for 20% of the total height of the control store, grossly exceeding original estimates.

The precharge logic for one bit line is shown in Figure 4-5. Note that numerous design rule errors exist. However, these do not matter since every precharge cell is performing the same function. The logic is only broken down into a separate piece for each cell for convenience in changing the number of bits in a word (it did in fact change from 64 to 58 to 60 during the design process). The second poly

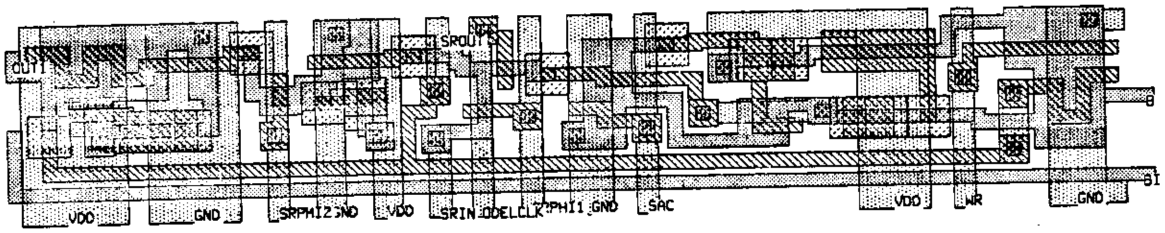

Figure 4-4: Sense Amplifier/Shift Register/Output Buffer Layout

line at the top of the cell forms the dummy word line that tracks word line delay. The top poly line is connected to the dummy line at the end of the array and brings the delayed signal back to the clock logic.

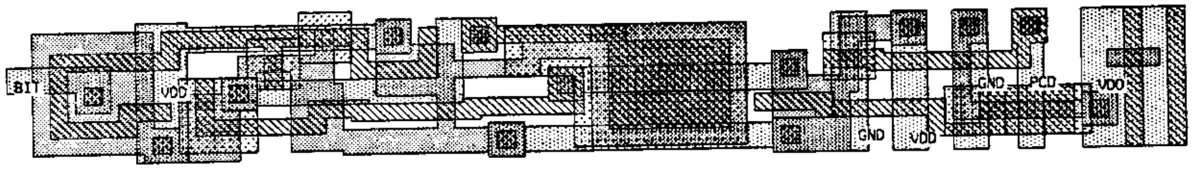

Figure 4-5: Precharge Logic Layout

The clock logic is shown in Figure 4-6. This block generates the WCK and RCK signals, TRAP, and the delayed precharge enable PCD.

A complete layout of the control store is shown in Figure 4-7. The register file layout is slightly different from the control store layout due to circuit differences. A layout of the entire PSC is shown in Figure 4-8. The control store and register file occupy the upper half of the chip.

5. Experimental Results

A test chip of the control store was designed and fabricated. Unfortunately test results did not become available before the first pass of the PSC was already being tested. A layout error resulted in the sense amplifier outputs being tied high, and was confirmed during PSC testing. However, the presence of the shift register meant that the rest of the PSC could be functionally tested, uncovering several other design errors elsewhere in the chip. A second pass corrected these errors, resulting in fully functional control store test chips and PSC chips. There was insufficient time to design a register file test chip, and we naively thought that the delay-tracking circuitry used in the control store would also work in the smaller register file, since it was only a slight modification of the basic design. We were wrong. The register file only worked intermitantly at low temperatures e.g., frozen with

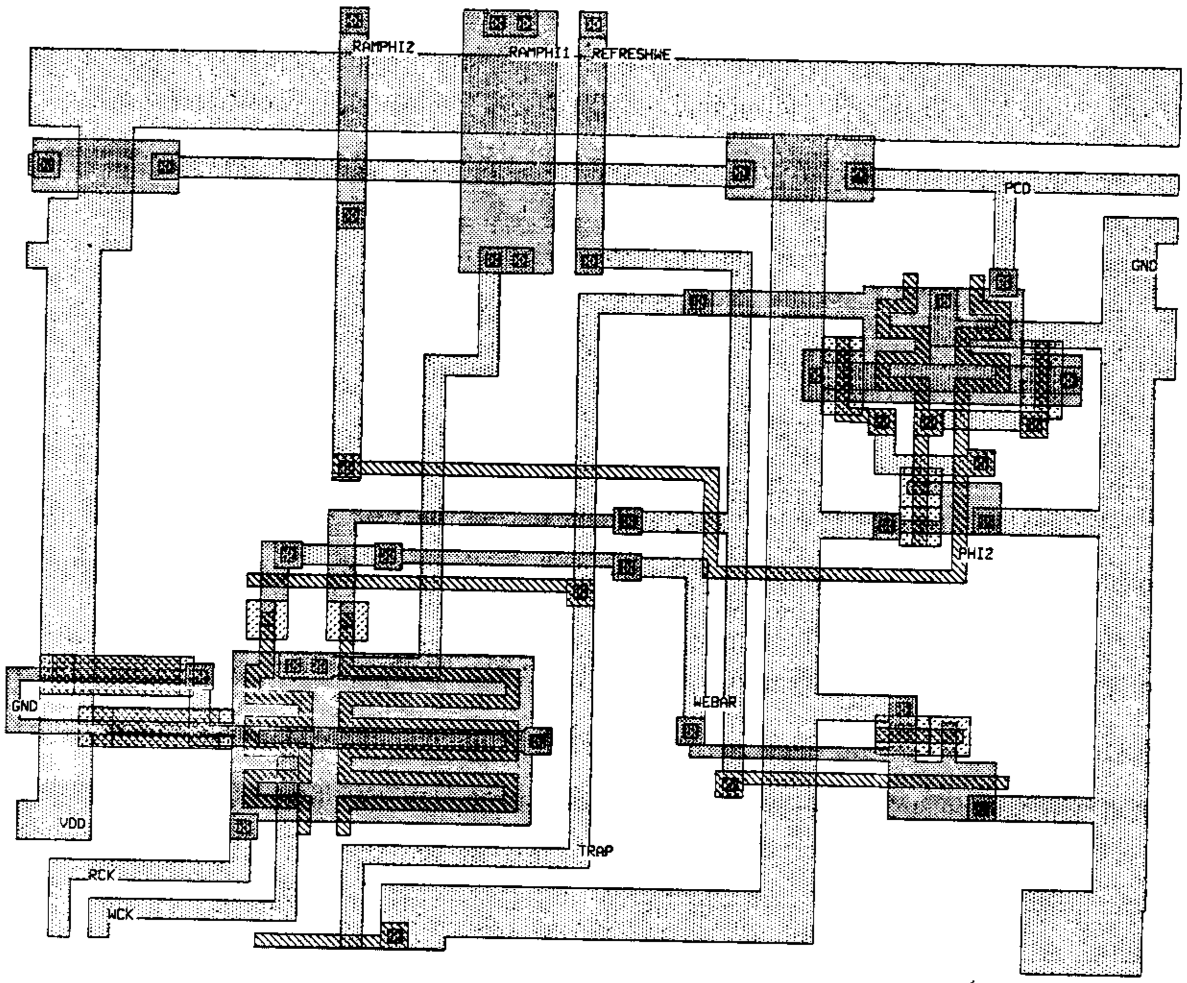

Figure 4-6: Clock Logic Layout

Quik-Freeze spray. A change in system timing temporarily circumvented the problem. Design changes discussed in Section 6 corrected the problem.

5.1. Low-Speed Testing

Low-speed testing of the PSCs and control store test chips was done with a 68000-based single board computer. C programs were used to read and write the test chip pins through PIAs. 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 clocks swinging between 0V and 4V.

As mentioned previously, the first control store test chip had a layout error resulting in all of the outputs tied high. The shift register was tested, with 3 of 5 from the MOSIS HP run M28S, 4 of 6 from the Comdial run M20V , and 4 of 6 from the HP run M20W working. Note that these are all very low yields since the shift register occupies only 1.2 square mm.

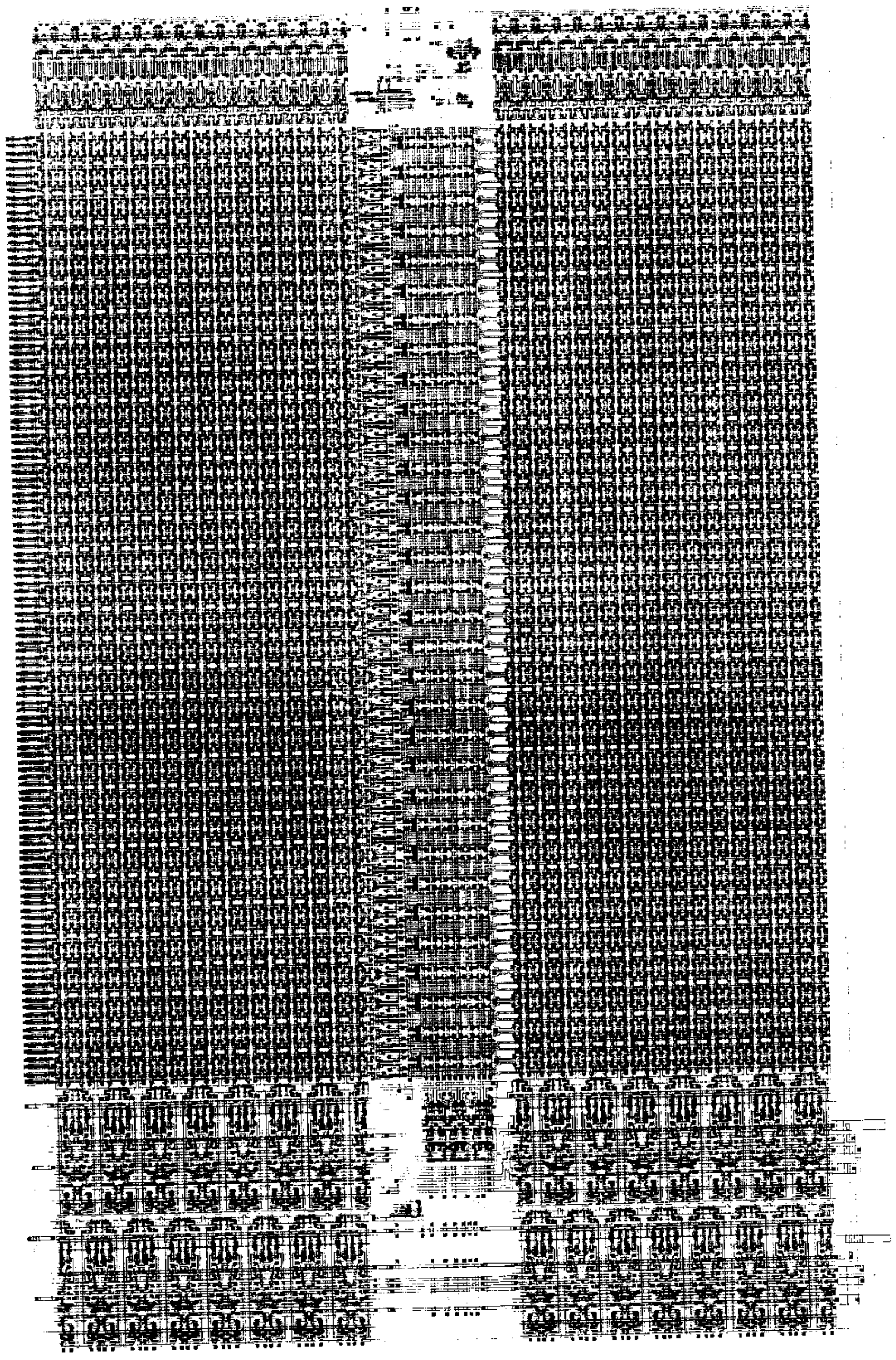

Figure 4-7: Writeable Control Store Layout

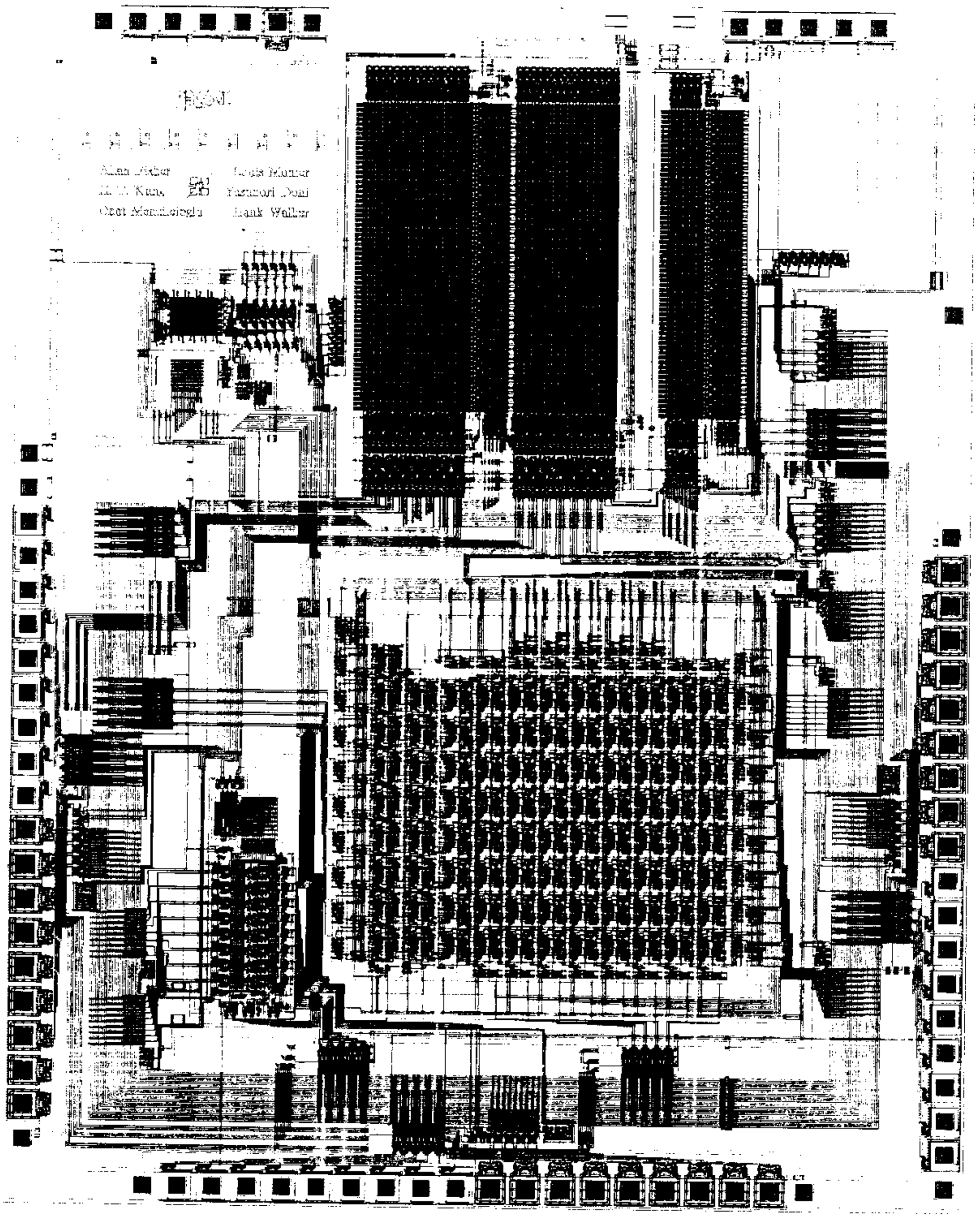

Figure 4-8: PSC Layout

The error on the first test chip was corrected and fabricated on HP run M31F. This chip was missing a wire that disabled the write logic to one-quarter of the array, but did not otherwise interfere with testing. Initial low-speed test results indicated that all outputs were tied high. It was discovered that the rise time on the PIA outputs was several hundred nanoseconds, and that the fall time was 50-100ns. The RAM was designed with work with rise and fall times of 10-20ns. Buffering the clocks eliminated this problem. The control store and register file were fully functional on 6 out of 26 PSCs on run M31F and 11 out of 19 PSCs on Comdial run M32I.

5.2. High-Speed Testing

High-speed testing of the M31F run control store test chip was done with a Tektronix DAS9000 analyzer. This system has a minimum time resolution of 40ns, and the available clock pulse widths in the range of interest were 40ns, 50ns, 80ns, and 100ns. Testing was done at room temperature with a 5V supply and clock waveforms swinging between 0V and 5V. Six of the eight chips worked at a pulse width of 80ns for RAMPHI1 and RAMPHI2, which corresponds to a cycle time of 160ns, where a cycle is one RAMPHI1 and one RAMPHI2 pulse. One chip was dead, and the other worked at a 200ns cycle time. This cycle time between 100ns and 160ns compares favorably to SPICE simulations using a cycle time of 100ns and the M31F run process parameters. The MOSIS on-chip ring oscillator frequency was 6.75MHz. The average control store power dissipation was 150mW. The maximum refresh time was 160us on four chips, 400us on two chips, and 1.6ms on one chip. These times are much lower than the refresh times obtained during functional testing, which were about one second at room temperature. The lower times are probably due to the very noisy test setup. All chips worked at an 80ns cycle time and 4ms refresh time when frozen with Quik-Freeze.

The output delay was 50ns, of which 30ns was due to the pad driver connected to a protoboard, cable, and scope probe. This delay made it necessary to use test patterns which hold the output stable for at least two pulse widths to obtain valid output.

6. Design History

As mentioned previously, the current control store and register file design is a slight modification of the original design. The old design used a delay line to ensure that a word was deselected before precharge began, to avoid destroying just-written data. The write and read clocks were sent through a dummy word line which tracked the delay in discharging a select line. This delayed clock was then used to enable the precharge logic. At the system level, it appeared that the falling edge of RAMPHI1 initiated precharge. This mechanism worked reasonably well in the control store, but appeared marginal in the register file, which has shorter word lines, and thus shorter delays. It appeared that precharge was not being delayed long enough to prevent overwriting cell data.

The solution to this problem was to move the guarantee of non-overlapping row select and precharge to the off-chip clock generator. Precharging is done on RAMPHI2. The time between RAMPHI1 going low (deselecting a word) and RAMPHI2 going high is made long enough to ensure that a word line is completely low before precharge is started (typically 10ns).

An additional change was made to clock the sense amplifiers on RAMPHI1BAR rather than RAMPHI2. In the original design, the sense amplifiers were clocked just as precharge was starting. This meant that RAMPHI2 had to go high just as RAMPHI1 fell, or even overlap, to ensure that the sense amplifiers could latch before the data on the bit lines was destroyed by precharging. By using RAMPHI2 for precharging and RAMPHI1BAR for latching, the sense amplifiers can settle during the time between RAMPHI1 going low and RAMPHI2 going high.

This change in clocking also eliminated another potential race condition. Originally, the sense amplifier output became invalid on the falling edge of RAMPHI2. However, this edge was also the time that the output buffer was being latched.[1] Since the sense amplifier output is now valid until RAMPHI1 goes high, these clocks are presently strictly non-overlapping.

As of this writing, the third version of the PSC is low-speed testing.

7. Future Design Changes

The precharge time can be made much longer while still meeting system timing requirements. This would mean that the bootstrapped driver could be eliminated, reducing the height of the RAM, as discussed in Section 4.

The original typical cycle time goal was 100ns. Test results indicate a cycle time between 100ns and 160ns. By precharging the bit line to 3.5V, an unaided RAM cell combined with an inverter on the output had a typical readout time of 45ns. This was clearly too slow since only 50ns was available for this phase of the clock, and the clock logic and row driver delays must also be included. Therefore, a sense amplifier had to be used to speed up readout. At the same time, the bit lines were charged all the way to 5V, since a RAM cell can pull down the bit line faster in this range. The PSC multiplier turns out to have a 160ns typical execution time. The worst case time cannot realistically be reduced

[1] In the control store, SRPHI2 operates in phase with RAMPHI2 during read cycles. In the register file, the sense amp and output latch were both clocked by RAMPHI2.

below 100ns, so the PSC cycle time cannot be easily reduced below 150ns (including precharge), and our current goal is a 200ns worst case cycle time. (We assume that worst case parts are twice as slow as typical parts). Therefore, sense amplifiers can be eliminated from the control store while still meeting the system timing requirements. As discussed in Section 4, this would significantly cut the height of the memories, thus reducing overall chip height.

The buffer logic used to generate TRAP and PCD are only present due to reasons discussed in Section 6. With suitable redesign of the precharge logic, both of these signals can be replaced by RAMPHI2.

The memories place a DC current load on RAMPHI1. When the clock is provided by a large off-chip driver this is no problem. But if the clock is generated on-chip, it is much preferable to have AC clock loads only. This can be accomplished by modifying the WCK and RCK generators, and the row driver logic, at a slight cost in area.

The circuit design for a simplified memory that would satisfy the needs of the PSC is shown in Figure 7-1. The RAM cells in this case have been increased in size. The increased speed provided by the larger devices is not necessary to meet timing requirements. The increased cell size of 14 x 19 lambda is needed so that peripheral logic can be made to fit in one cell pitch rather than being stacked. This results in a control store that is about 1075 x 1475 lambda in area, which is roughly the same area as the current design, but with a more desirable pitch.

8. Conclusions

8.1. Testing

The presence of the shift register in the microinstruction register was critical in debugging the PSC. Due to a rushed schedule, the memories were not debugged via test chips before the full PSC was fabricated. Do not repeat this mistake. Since the control store and register file did not work correctly on the first and second passes, the lack of a shift register would have meant several more design iterations to debug the design. In retrospect, additional shift registers on the register file and multiplier would have make debugging even easier since their inputs and outputs would have been directly accessable.

The second important feature of the PSC that greatly aided testing was the use of separate control pins for the memories. This meant that an additional level of control could be placed over them,

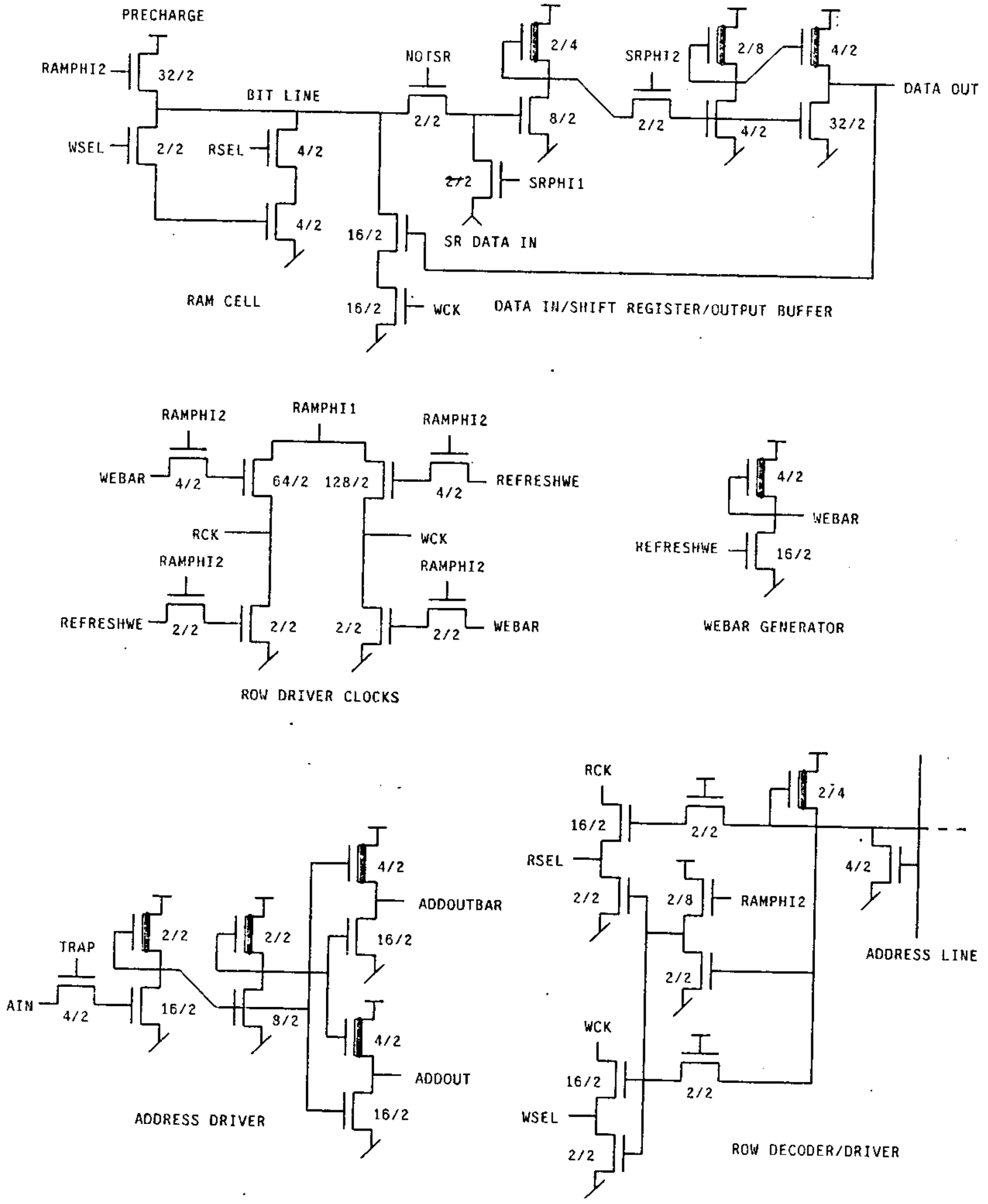

Figure 7-1: Simplied WCS Circuit Schematic

allowing the use of non-standard clocking to get around circuit bugs. The extra pins also meant that we could postpone decisions on system timing until late in the design process, or even alter the timing after-the-fact.

One bizarre problem that we had was with the 3M 84-pin ZIF sockets, which hold our 84-pin leadless chip carriers. The contacts are on 50 mil centers, and it is quite difficult to get the chip aligned in the socket, due to poor keying. The strange part is that it is possible to get the chip into the socket such that all but a few pins are seated properly. Since we cannot perform a continuity test, it is very difficult to determine whether a chip is dead, or just not mounted in the socket correctly. We have tried to get around this by inserting and removing chips from the socket several times when they appear to be bad. This seems to control the problem, which is even worse in prototype systems since they have several ZIF sockets. Once a package is seated, it appears to remain so, dispite jiggling. For future high-pinout designs we will use pin-grid packages, which use standard ZIF socket technology, and are available with higher pin counts.

Our single-board computer is adequate for most of our low-speed testing needs. The only problems that we had with it were that it was occasionally too slow, and that 32 I/O pins are frequently insufficient. The average instruction execution time of 5usec meant that loading the microcode shift register could take longer than the memory refresh time. Some amount of cleverness was needed to get around this problem. The limited number of I/O pins meant that a test jig had to be built to multiplex between the PSC I/O ports.

The DAS9000 can be used for limited high-speed testing. The major problems are that it is not fast enough, it does not have enough memory, and that it is a stand-alone system. The minimum time resolution on our version is 40ns. This is too slow if you are trying to characterize a simple part. The pattern generator has only a simple set of commands and 254 time steps. In order to load one microinstruction into the PSC, it is necessary to use 122 instructions. A more flexible set of instructions (more like a real programming language) would greatly reduce the program size. The pattern generator language bears no relationship to the C programs that we write for our low-speed tester. Therefore, we must write completely new routines. In addition, the DAS is not connected to a serial line or tape drive, so programs are lost on power-down or when a new program is entered. The data acquisition memory is also of limited size, making it difficult to use very many test patterns before they must be re-examined. Pulse generators would have been more convenient for testing the PSC memories.

8.2. Circuit Design

One lesson to be learned about the PSC memories is to be careful when trying any fancy circuit design. Debug it thoroughly using test chips. However, even this is not a guarantee of success. The delay tracking circuit worked only marginally. Since device parameters fluctuate wildly from month to month for MOSIS-fabricated chips, such marginally working circuits will have unacceptably low yield when large numbers of working parts are desired, as is the case with the PSC.

Tricky circuit designs can only be used if they are insensitive to large process fluctuations, particularly variations in device sizes. Worst case device models can be developed by examining the summary process parameters provided by MOSIS. A refinement is to develop worst case models for each manufacturer, and specify that a design should be fabricated only on a month that a particular manufacturer will be used. In general, this will result in unacceptable fabrication delays.

Worst case design is overly pessimistic. It is possible to develop a model of the fluctuations in a particular fabrication line, and then perform statistical process simulation in order to obtain more accurate device models for circuit simulation [Nassif 82]. However, this simulator requires knowledge of the fabrication process and test data. The test data is currently available in summary form, but can be obtained from MOSIS in its raw form. The fabrication sequence is not available, but can be guessed at.

A few useful circuit design principles are illustrated (or absent) in the memories. Speed can be gained by reducing voltage swings, such as precharging only to 3.5V. However noise margins must also be maintained. Current sensing is faster than voltage sensing since capacitors don't have to be charged. This is inherently more difficult in MOS than bipolar technologies, and was not used in the PSC. Differential voltage sensing is fast, and more insensitive to process fluctuations. Small, low-power, and fast clocks can be built using bootstrapped pass transistors, as was done in the clock drivers and row drivers.

In general it is my feeling that university groups interested in implementing architectures in VLSI should engage in circuit design primarily to obtain functionality. For example, in the PSC, a dynamic memory was used because simpler designs would have been too large. Circuit design to achieve high speed should be used only sparingly. Performance that is good enough not to be laughed at should be the goal, not matching industry's abilities.

8.3. Layout Design

The PSC memories used slightly aggressive buried contact spacing rules, and had occasional design rule violations. In retrospect these became yet another source of worry without providing a significant area savings. Data from a four-transistor dynamic RAM [Walker 83] show that these buried contact rules work well at a lambda of 2.5 microns, but poorly at a lambda of 2 microns. In addition, the yield of parts with these aggressive rules was significantly lower from some manufacturers, such as HP, than at others. In general, stick to the official design rules unless you have sufficient experience with the MOSIS manufacturers to be confident that your modified design rules will work reliably.

The need for global floorplanning early in a project is especially illustrated by the blank area on either side of the memories in the PSC layout. We explicitly avoided any sort of systems engineering other than timing. This allowed members of the design team to work completely independently. In retrospect, the floorplanning could have been done with little effort, and resulted in a much smaller chip. However this probably would not have resulted in significantly improved yields, because most of the saved area is currently unused. Less wasted area would lower chip costs, but this is somewhat irrelevant to us.

The most difficult layout task connected with the memories was wiring up the microcode bus. A global router would have saved us a significant amount of design time. A Mead-Conway design style tries to avoid routing by having a global bus run through all of the logic blocks, with all blocks designed to the same pitch. This was not possible in our case, making extensive routing was unavoidable. Logic blocks such as the multiplier and memories cannot be matched in pitch with embedded bussing without large area penalties. The current state of affairs is that everyone is afraid to modify the PSC design because of the rerouting that would be necessary.

8.4. Using MOSIS

After using MOSIS and MPC for several years, we have learned how to get the most out of it. The unique feature of MOSIS, other than elimination of the grunt work, is the fast turnaround. Turnaround has been as short as one month, or as long as three months, with the average about two months. Short turnaround means that it is possible to design, fabricate, and test all subsystems of a large chip separately. This isolates errors in the final system to the external interfaces and protocols. It has frequently been the case that we irrationally rushed completion of the entire design, so that it was fabricated before test chip data became available. The result has usually been wasted effort since bugs turned up in the test chip. The correct method is to take your time and do it right the first time.

Since there are usually more subsystems than designers on a university-based project, the delay between design submission and the return of fabricated parts can be used to design other subsystems. One area that we have found contributes much of the elapsed time in a project is a test jig setup and test program development. These should be completed before fabricated parts arrive, and in such a manner that testing can begin immediately. Failure to do rapid testing causes most of the benefits of fast turnaround fabrication to be lost.

We have discovered that there is a large variance in yield from run to run with the same manufacturer. Both Comdial and HP have produced chips with high and low yields. The Comdial run of the PSC had much higher yield than the various HP runs, but our sample is too limited to draw any conclusions.

8.5. Summary

As mentioned in the sections on circuit and layout design, don't be tricky unless you are certain of getting a large payback. Manpower is limited, and it is usually better to spend it on the higher levels of design. For universities, what Carver Mead has said is really true - good circuit and layout design will buy you factors of two, but good architecture will buy you orders of magnitude.

9. Acknowledgements

I began work on the PSC memories in March 1982 after an ARPA review. H. T. Kung asked me to design some memories that would be small enough to fit onto the chip. Allan Fisher worked with me to get the timing right, and wrote the programs for testing the memories in the PSC.

10. References

[Barbacci 81] M. R. Barbacci.
Instruction Set Processor Specifications (ISPS): The Notation and Its Applications.
IEEE Transactions on Computers C-30(1):24-40, January, 1981.

[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 83] E. H. Frank and R. F. Sproull.
A Self-Timed Static RAM.
In R. Bryant (editor), *Proceedings of the Third Caltech Conference on VLSI*, pages 275-285. Computer Science Press, Rockville, MD, March, 1983.

[Kung 80] H. T. Kung and C. E. Leiserson.
Algorithms for VLSI Processor Arrays.
In C. Mead and L. Conway (editor), *Introduction to VLSI Systems*, pages 271-292. Addison-Wesley, Reading, MA, 1980.

[Nassif 82] S. R. Nassif, A. J. Strojwas, and S. W. Director.
FABRICS II: A Statistica Simulator of the IC Fabrication Process.
In *Proceedings of the IEEE International Conference on Circuits and Computers*, pages 298-301. September, 1982.

[Walker 83] Hank Walker.
A 4-Kbit Four-Transistor Dynamic RAM.
Technical Report, Carnegie-Mellon University, Computer Science Department, May, 1983.