64-KByte Sum-Addressed-Memory Cache with 1.6-ns Cycle and 2.6-ns Latency

Raymond Heald, Ken Shin, Vinita Reddy, I-Feng Kao, Masood Khan,
William L. Lynch, Gary Lauterbach, and Joe Petolino

***Abstract*— Address base-plus-offset summing is merged into the decode structure of this 64-KByte (512-Kbit), four-way set-associative cache. This address adder avoids time-consuming carry propagation by using an $A + B = K$ equality test. The combined add and access operations are implemented using delayed-reset logic and a 0.25-μm process. This wave pipelined RAM achieves a 1.6-ns cycle time and 2.6-ns latency for the combined address add and cache access.**

I. Introduction

FIRST-LEVEL cache RAM blocks for the next generation of high-speed processors must be both higher capacity and greater throughput than prior generations. The goal for this design was a 64-KByte RAM block in 0.25-μm technology operating at 600 MHz. The 64-KByte RAM block can be assembled from 4-KByte arrays, which have array access of less than 1.5 ns. However, the wire delay to distribute addresses to the individual array blocks and return the data output to the edge of the RAM adds more than 1 ns to the 64-KByte access time. Since this will not fit in a single 1.67-ns clock cycle, techniques were developed to combine the RAM access with the address offset add from the pipe stage ahead of the cache.

This cache, like many processor caches, is indexed with base + offset addresses. To access such a cache, the address addition must be done first. Thus, the address-adder delay increases the total cache latency. Incorporation of the address base + offset add function into the cache decoder can eliminate significant pipeline latency in a high-speed processor. The circuit presented here combines the sum-addressed-memory (SAM) concept [1]–[3] with delayed-reset logic circuitry [4]–[6]. This combination enables cache access with a two-cycle latency for a 600-MHz processor (third-generation superscalar processor implementing the Sparc V9 64-b architecture [7]).

In the next section, the SAM concept will be introduced with the resulting single and multibit SAM logical equations. Then the SAM operation will be discussed and the column and array decoders shown. Next, the circuits used to implement the SAM decoder will be examined. This will be followed by the decoder realization for the 64-KByte cache and the read critical path discussion. Last, the test chip results will be examined and the work summarized.

Manuscript received April 2, 1998; revised June 8, 1998.

R. Heald, K. Shin, V. Reddy, I.-F. Kao, W. L. Lynch, G. Lauterbach, and J. Petolino are with Sun Microelectronics, Palo Alto, CA 94303 USA.

M. Khan was with Texas Instruments, Stafford, TX 77477 USA. He is now with Hitachi Micro Systems, Inc., San Jose, CA 95101 USA.

Publisher Item Identifier S 0018-9200(98)07050-4.

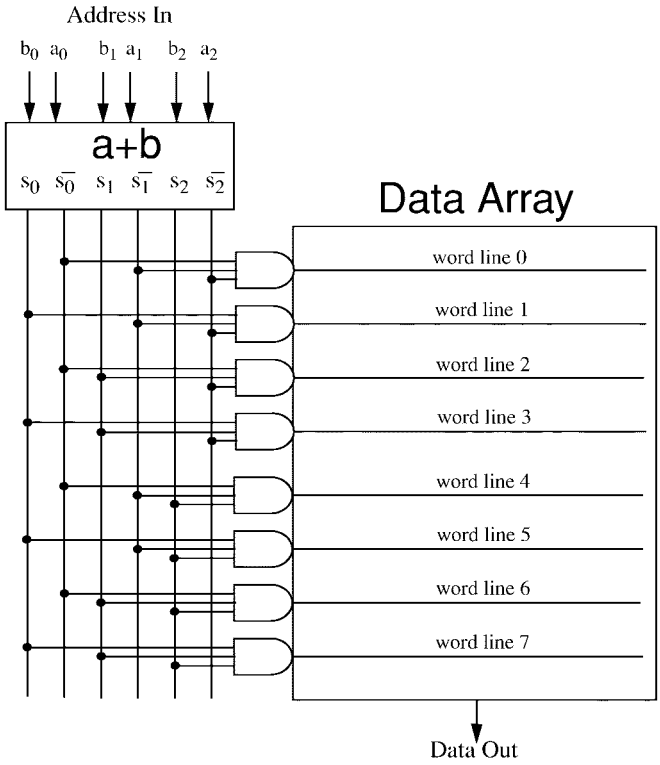


Fig. 1. Addition-indexed memory.

II. SAM Decoder Concept

To perform the address-offset add and then access a large cache within two cycles, it is desirable to embed the offset adder within the RAM decoder as shown in Fig. 1. However, the carry propagate chain of a straightforward adder-decoder combination is still a very slow path. To achieve a significant speed improvement over the speed of a separate adder block followed by a RAM block, a logical refinement is used, the SAM decoder. The SAM decoder takes advantage of the extensive parallelism of conventional RAM decoders, together with the fact that unlike generating an adder sum, an $A+B=K$ (constant K) equality test requires no time-consuming carry propagation [8]. This result can be seen by taking the three n-bit two's complement vectors: $A = a_{n-1}a_{n-2}\cdots a_0$, $B = b_{n-1}b_{n-2}\cdots b_0$, and $K = k_{n-1}k_{n-2}\cdots k_0$. The required carry-in c_i^{in} and required carry-out c_i^{out}, of each bit position i are uniquely determined if $A + B = K$. If for every bit

position the required carry-out from the previous bit position is the required carry-in to give k_i, then $A+B=K$.

The required carry-in and -out equations are the normal adder carry equations where the generate and propagate terms g_i and p_i are evaluated using the k_i for each bit position i. Maintaining the notation of previous publications [1], [8], the AND and OR functions are represented here by $\wedge$ and $\vee$, respectively. The equations for the carry-out of the $i-1$ stage and into the i stage are

$$\begin{aligned} c_{i-1}^{\text{out}} &= \left(p_{i-1} \wedge \overline{k_{i-1}}\right) \vee g_{i-1} \\ &= \left((a_{i-1} \oplus b_{i-1}) \wedge \overline{k_{i-1}}\right) \vee (a_{i-1} \wedge b_{i-1}) \\ c_i^{\text{in}} &= k_i \oplus p_i = k_i \oplus (a_i \oplus b_i). \end{aligned}$$

These equations' required-carry terms depend only on a_i, b_i, and k_i of their respective bit positions. So the equality $c_i^{\text{in}} = c_{i-1}^{\text{out}}$ depends only on two adjacent bits with no multiple bit propagation requirements.

A RAM decoder can be imagined to perform a large AND function of the address lines, where these AND gates test for a sum equal to the constant K for each word line K. To perform this parallel test, the "address" sent to the final decoder cannot depend on K. This requires some extra logic within the predecoder circuitry to precondition the addresses prior to the final decoder. The preconditioning is most easily done a bit at a time, resulting in equations for the carry-in equals carry-out equality e_i at each bit position. For each bit position there are four cases of $e_i^{k_i k_{i-1}}$

$$\begin{aligned} e_i^{00} &= (a_i \oplus b_i) \oplus \overline{(a_{i-1} \vee b_{i-1})} \\ e_i^{01} &= (a_i \oplus b_i) \oplus \overline{(a_{i-1} \wedge b_{i-1})} \\ e_i^{10} &= \overline{(a_i \oplus b_i)} \oplus \overline{(a_{i-1} \vee b_{i-1})} \\ e_i^{11} &= \overline{(a_i \oplus b_i)} \oplus \overline{(a_{i-1} \wedge b_{i-1})}. \end{aligned}$$

For the low-order bit position, the carry-in c_{-1} takes the place of the $i-1$ terms, giving

$$\begin{aligned} e_0^{0c} &= (a_i \oplus b_i) \oplus \overline{c_{-1}} \\ e_0^{1c} &= \overline{(a_i \oplus b_i)} \oplus \overline{c_{-1}}. \end{aligned}$$

The block diagram of an eight-entry RAM with SAM decoding is shown in Fig. 2. Note that the four cases of $e_i^{k_i k_{k-1}}$ cause the number of address lines per bit in the decoder to double from two (true and complement of each address bit) to four, while each decoder word line AND contains the same number of terms as in a conventional decoder array. Thus, the SAM decoding resembles two-bit predecoding in wire count, but the final decoder retains the logical complexity of a single-stage decoder. Hence, the single-bit SAM structure does not reduce the complexity of the word-line decoder. This is not a concern for small arrays. But for large arrays, it is important to do enough logic in the predecoder that the final decoder can be built efficiently. For these large arrays, faster access can be achieved by generating the AND of the SAM outputs in groups of two or more bit positions. The two bit SAM equations are

$$\begin{aligned} e_i^{000} &= e_i^{00} \wedge e_{i-1}^{00} \quad e_i^{001} = e_i^{00} \wedge e_{i-1}^{01} \\ e_i^{010} &= e_i^{01} \wedge e_{i-1}^{10} \quad e_i^{011} = e_i^{01} \wedge e_{i-1}^{11} \\ e_i^{100} &= e_i^{10} \wedge e_{i-1}^{00} \quad e = e_i^{10} \wedge e_{i-1}^{01} \\ e_i^{110} &= e_i^{11} \wedge e_{i-1}^{10} \quad e_i^{111} = e_i^{11} \wedge e_{i-1}^{11}. \end{aligned}$$

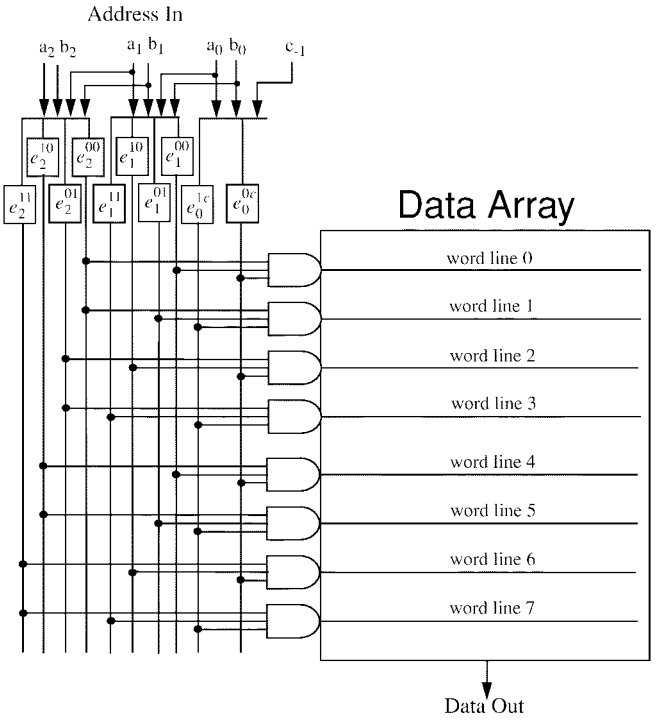


Fig. 2. SAM indexed memory.

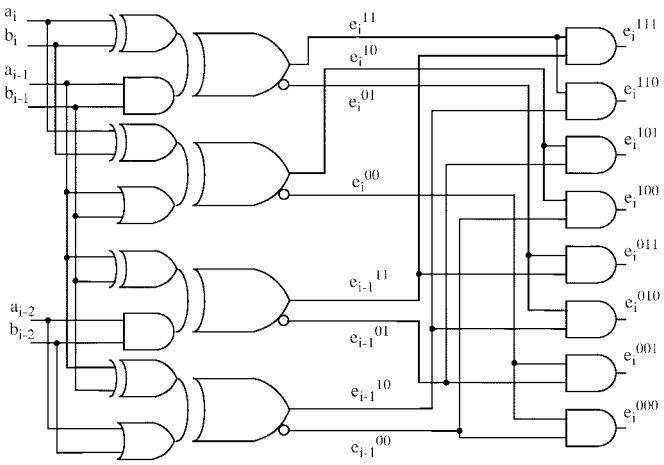


Fig. 3. SAM 2-bit predecoder logic.

As before, the low-order bits contain the carry-in terms, resulting in

$$\begin{aligned} e_1^{00c} &= e_1^{00} \wedge e_0^{0c} \quad e_1^{01c} = e_1^{01} \wedge e_0^{1c} \\ e_1^{10c} &= e_1^{10} \wedge e_0^{0c} \quad e_1^{11c} = e_1^{11c} \wedge e_0^{1c}. \end{aligned}$$

These multibit SAM decoder signals reduce the complexity and delay of the final decoder in the same way as a conventional predecoder reduces the complexity and delay of its final decoder. In the two-bit SAM predecoder logic shown in Fig. 3, the predecoder uses groups of three base and three offset bits to produce two out of eight predecoded signals for the final decoder. This two hot encoding scheme is chosen to allow parallel evaluation of the carry-in value. Unique word-line addressing prior to carry-in determination is achieved by dividing the array into segments such as odd and even lines.

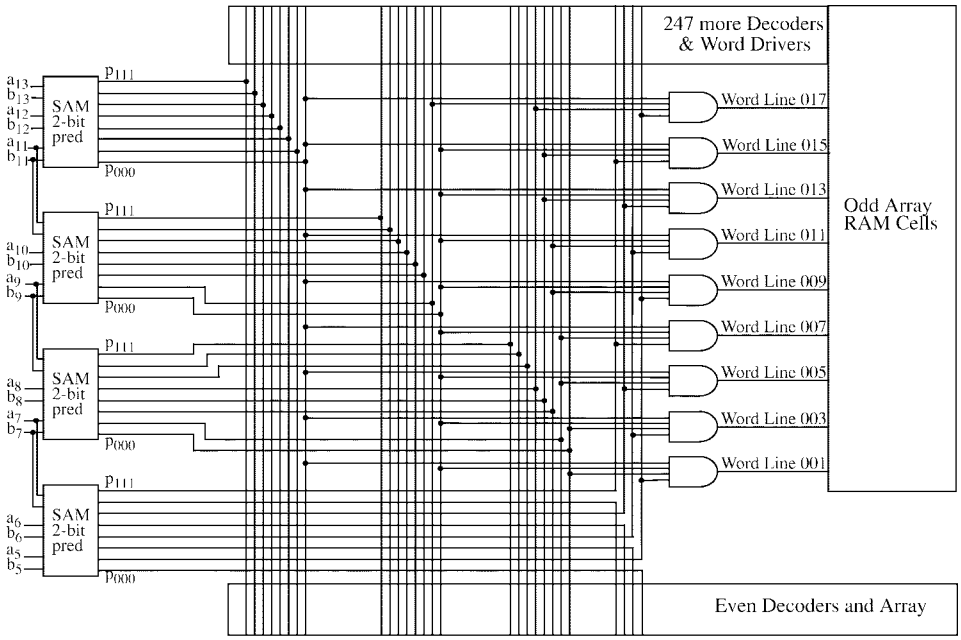


Fig. 4. SAM row decoder for 64-K cache.

III. SAM Cache Operation

A SAM cache logically resembles the organization shown in Fig. 2 but will generally have many more entries and possibly a multibit SAM predecoder. The base and offset of the address arrive at the SAM decoder from the register file, bypass network, or logical unit. A cache normally does not address the RAM with the low-order bits of the address. The low-order bits are used only for line offset. SAM decoding cannot decode these low-order bits as they are bytes within the double word or larger cache segment being addressed. But the SAM decoder requires the carry-in to the addressed segment to select the correct segment. For example, the carry-out from the low-order bits addition is needed to determine whether an odd or an even line is being addressed. Care is needed to prevent this from becoming the critical path in a SAM cache with the low latency of the SAM circuitry. A solution to this carry-in slow path is to use the late-arriving carry-in to select between an odd–even pair of lines read from two banks of the cache. Each pair of word lines in a SAM decoder is either an odd–even or an even–odd pair. Thus, a small SAM addressed array may require twice as many bit lines and/or twice as many word lines as would be needed if the carry-in were available at the beginning of the access. However, most physical RAM implementations already contain array segmentation for speed reasons.

Since SAM addressing uses the correct address on the first access, it avoids any overhead for retrying on a mispredicted address as happens with XOR addressed caches [1]. In addition, SAM generates the actual address so no duplicates or mispredicted addresses are created. Last, the SAM cache retains the hit rate of a normally addressed cache because it *is* a normally addressed cache, just one with a lower latency.

IV. SAM Decoder in a 64-KByte Cache

For a 64-KByte, four-way SAM cache with 512 lines of 256 bits each, the selected line is addressed with the SAM row decoder shown in Fig. 4. As is shown in this figure, address bits 13 through 5 are used for addressing the word lines. The outputs of the least significant SAM predecoder are divided between the odd and even arrays to allow unique access within each array. The low-order bits 4 and 3 select the double word within the line, and bits 2 through 0 generate the carry_out_2 for the column decoder. The column decoder, shown logically in Fig. 5, combines the carry_out_2, c2, from these low-order bits with bits 3 and 4 to provide a unique one-out-of-four double-word selection. The column decoder timing is less critical than the row decoder, so the carry circuit does not limit

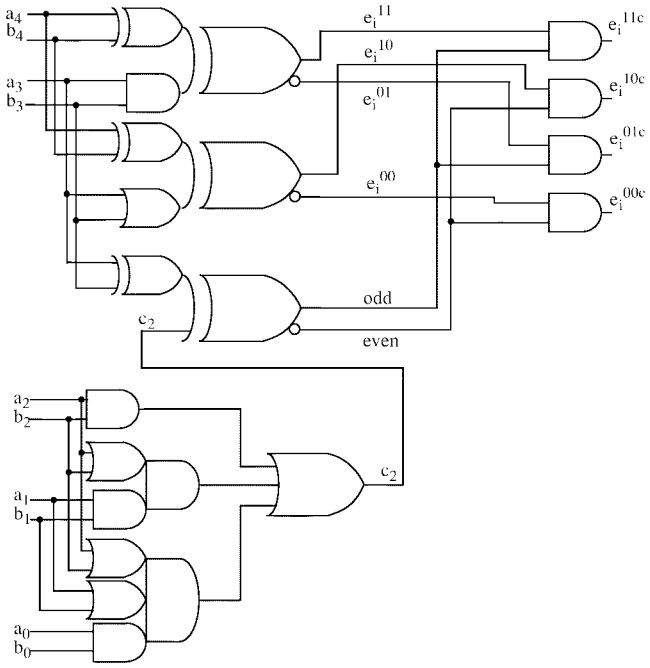


Fig. 5. SAM one-of-four column decoder.

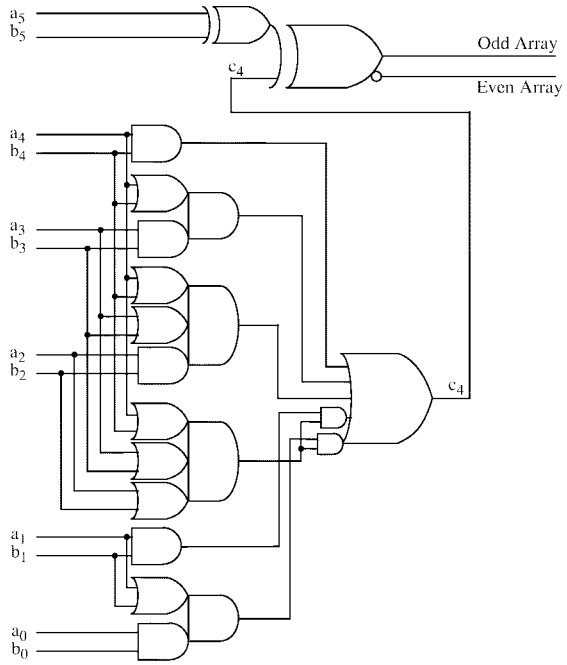


Fig. 6. SAM array decoder.

the overall RAM access time. The array decoder in Fig. 6 uses a similar combination of carry and XOR circuits to produce the odd or even array choice in time to power the correct sense amplifiers and multiplex circuits at the array output. Fig. 6 shows that bits 4 through 0 generate the carry_out_4, which is added to the bit 5 sum to determine the odd/even select. This is a fairly complex logical operation but can often be completed in time to power only the needed sense amplifiers. For a 64-bit output with four-way set associativity, this is a significant power savings.

The SAM structure requires 14 bits from each of two addresses to be routed from the data path to the RAM block. This is twice as many as a conventional RAM of the same size. But its overall impact on routing to and within a large cache with 256 inputs and 64 outputs is negligible. The remainder of the data path is relatively conventional. The bank or way select is determined by a SAM-addressed 8-bit micro-TAG subset of the complete TAG accessed in parallel with the data array. The four 8-bit outputs of this micro-TAG are compared while the large data-cache access is under way. The way-select arrives in time to drive a four-to-one multiplexer located between the sense amplifiers and the data aligner. Following the data aligner, the data are buffered by the data output drivers and are sent to the execution unit.

A. Delayed-Reset Circuit Implementation

For highest speed with a limited power budget, delayed-reset logic is used to realize this SAM cache. Delayed-reset logic [4] is similar to postcharge logic and self-resetting logic used in other high-speed RAM designs [5], [6]. It allows the high-speed signal propagation and high fan-up of domino logic while locally generating the reset for each stage. Local reset saves power since only nodes that undergo a transition during the logic propagation are driven by a local recovery signal.

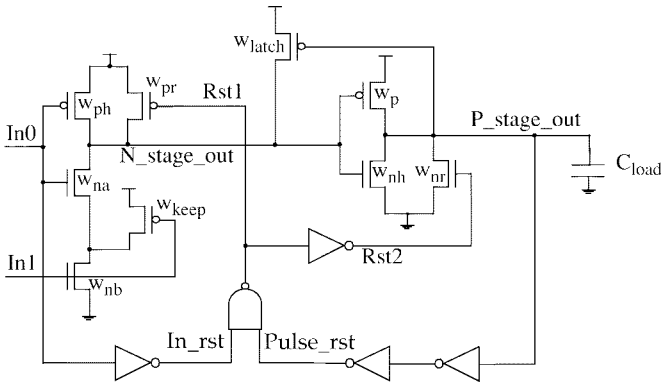


Fig. 7. Delayed-reset NAND2.

This recovery signal is a delayed version of the signal that just had a transition, so no global recovery clock is needed. In a logical structure where a very small percentage of the gates undergo a transition in a given cycle, such as a decoder, a significant power savings results from not clocking the quiet gates. Also, the pulse is completed without requiring a clock edge. This allows multiple accesses to propagate through the delayed-reset logic chain simultaneously without clock blockages. This is of special interest in a pipelined RAM as it permits a RAM latency greater than a single cycle without introducing latches or flip-flops at some inconvenient place in the RAM access path. Delayed-reset logic has the added benefit of being logically immune to the hazard of momentary short circuit current caused by the reset signal arriving early. This further controls the RAM power. A delayed-reset NAND2 circuit is shown in Fig. 7. This shows the forward path from In0 and In1 through the series NMOS devices to N_stage_out and the PMOS inverter stage to P_stage_out. The reset paths from In0 via In_rst and from P_stage_out via Pulse_rst control the first stage reset with Rst1 and the second stage reset with Rst2. Other logic blocks are built in a similar way.

The extension of these techniques to include the SAM decoder only requires revised predecoder and column decoder circuits. Delayed-reset logic (and most other cousins of domino logic) works best in high fan-out situations. Hence, the chosen 2-bit SAM predecoder realization in Fig. 8 combines the 2-bit XOR and input AND/OR logic into a single complex NMOS stack. The output AND is done in the P-stage, which drives the predecoder outputs and the wires leading to the final decoder. Some keeper and reset devices have been omitted from this schematic for clarity. They are only to insure glitch-free operation at any speed and are not important in the understanding of normal operation.

Comparing this predecoder to a standard one-out-of-eight tree predecoder shows that the block delay increases by about 110 ps with the SAM predecoder, or about one loaded gate delay. As the 2-bit SAM predecoder is a two-out-of-eight output, the final row decoder, while conventional in design, has a slightly longer delay due to the additional input. Overall, this SAM decoder gives a cache access of only 200 ps greater than a normal cache while eliminating the need for a separate address add stage prior to cache access.

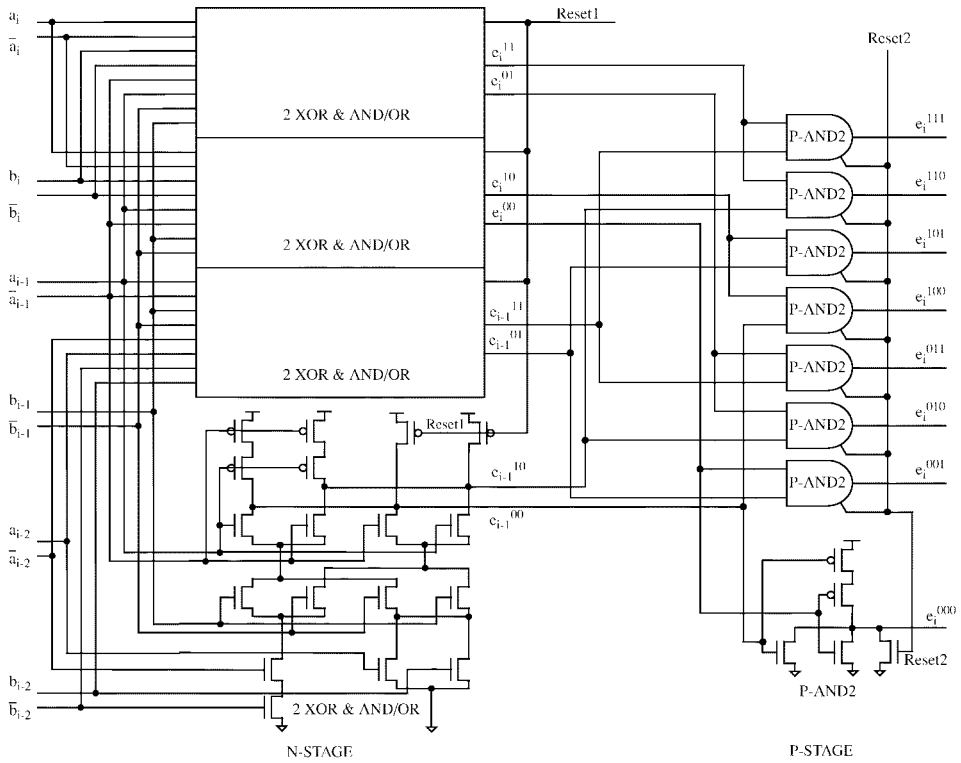


Fig. 8. SAM predecoder circuit realization.

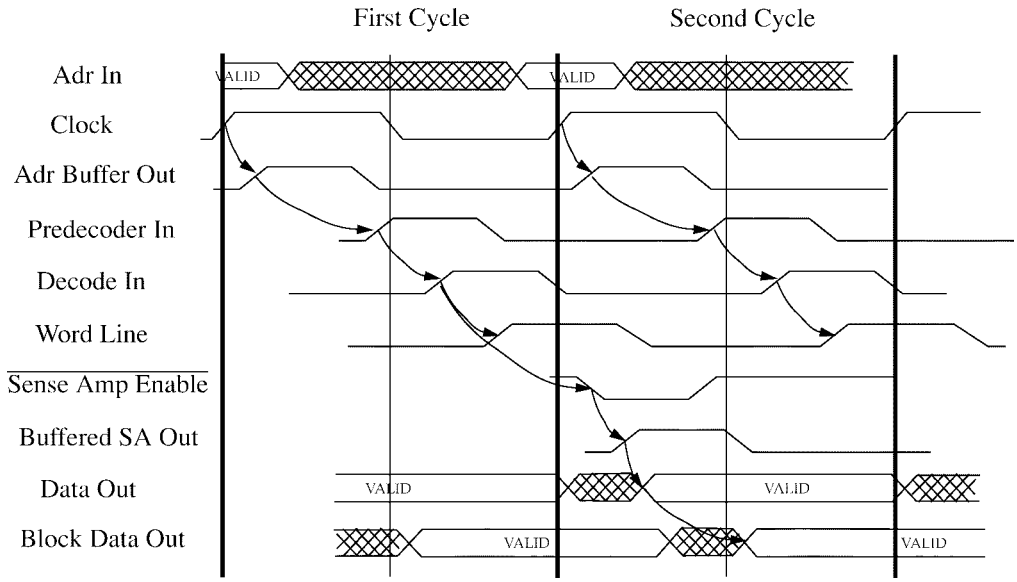


Fig. 9. Read cycle access path.

B. Array Organization and Operation

The SAM decoder circuitry drives an array of six-transistor memory cells organized in 16 blocks of 256 bits by 128 words. The read and write column decoders select the four double words for output or input as needed. The sense amplifier is a low-power CMOS latch feeding a four-way hit-multiplex circuit. The hit-multiplex output is fed through an alignment multiplex and exits the SAM cache block. The 8-bit micro-TAG is included in the SAM cache block and is SAM accessed in parallel with the cache. It contains 8 bits of the complete tag and is placed between the cache array blocks to minimize wire delay. Its small size allows the access and hit compare to take place in the time the cache is accessed and the data amplified by the sense amplifier. Hence, the hit signal is generated just in time to control the four-way multiplex circuit.

Each 4-KByte RAM block contains two spare rows and two spare columns in the memory array. This redundancy is controlled by laser-programmed metal fuses. The latency

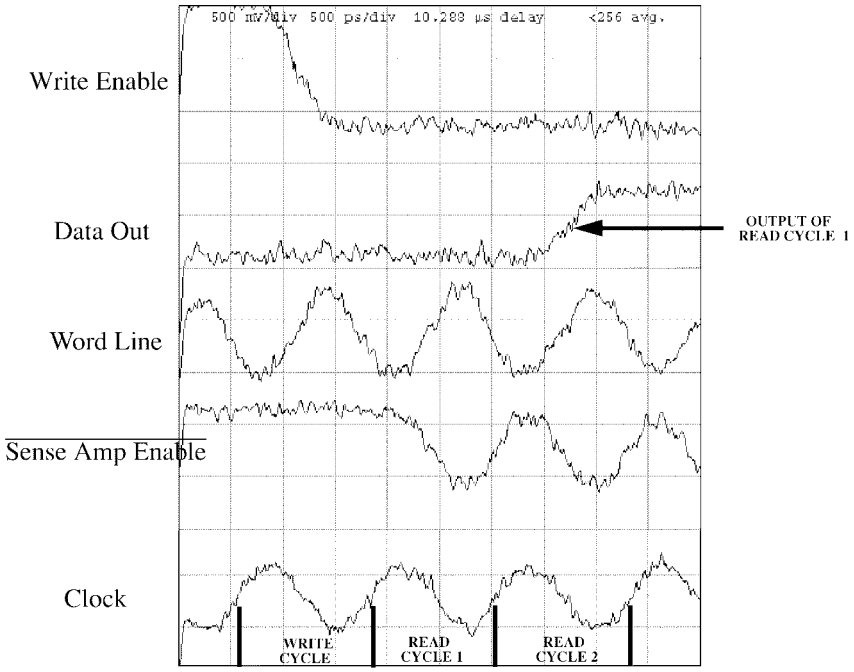


Fig. 10. Test circuit waveforms.

TABLE I
SAM CACHE AND PROCESSOR TECHNOLOGY

Drawn transistor	.25 μ
Metal system	Six metal layers
Memory cell type	Six-transistor CMOS—Two-layer metal
Memory cell size	3.56 μ × 4.32 μ
Cache storage	64 K Bytes data + 2 K Bytes μ-TAG
Cache organization	512 lines × 4 ways × 256 bits
Total cache transistors	3.9 million
Cache block size	5.2 × 4.2 mm
Cycle time	1.6 ns
Access time	2.6 ns
Cache power	1.9 W
Clock frequency	600 MHz
Power supply	1.8 V

increase is kept low by including the column redundancy switch in the column decoding.

The RAM latency is controlled without flip-flops in the memory-cell access path by adding latency control latches to the output path. These latches synchronize the cache output to the processor clock for any clock frequency by employing odd/even latches and multiplex structures. When the clock frequency is so slow that the cache access is completed within a cycle, the latch output is disabled until the next cycle. At high frequencies, the cache requires more than a single cycle to complete the access. Then the latch is transparent when the data arrives and adds very little to the access time.

Wave forms of a read access progressing through the circuit are shown in Fig. 9. The cycle begins with the two sets of addresses being sampled by the leading edge of the clock. The address flip-flops generate the pulsed true and complement address signals, which are then routed to the individual 4-KByte arrays with significant wire delay. The local blocks then SAM-decode the addresses and access the selected word line. After a differential has been established on the bit lines, the global sense-amplifier enable signal is pulsed low and inverted within each individual sense amplifier to provide a well-controlled sensing operation. The buffered sense amplifier output is selected by the way-select multiplexer, converted to a static signal at the latency latch, and sent to the block output. In this example, the clock frequency is high enough that the latency latch is transparent when the output data arrive showing the normal circuit operation.

V. PROCESS TECHNOLOGY AND TEST CHIP

The target process used for the SAM cache and the 600-MHz processor containing it is a 0.25-μm (drawn) transistor process with six layers of metal. The dual-layer metal six-transistor with second-metal bit lines cell is 15.38 μm^2 in area, and the 64-Kbyte SAM data cache occupies 17.2 mm^2 of the 330-mm^2 processor die. The SAM decoder as realized is 33% wider than a similar decoder without the SAM logic. This is a total increase in the RAM width and area due to the SAM of about 4%. With the micro-TAG, comparators, data aligner, and global power and signal channels included, the SAM data cache block size is 21.8 mm^2. The technology is summarized in Table I.

A bit slice of the RAM has been fabricated and tested to check data flow and timing in the 0.25-μm six-metal process. The decoder and output structure were simplified to fit the test environment. The test-structure critical path consists of: address-input flip-flop, delayed-reset non-SAM predecoder, delayed-reset decoder, word line, memory cell, bit line, sense amplifier, and output latency latch. The test chip did not duplicate the full chip latency as the wiring needed for a test chip differs from that of a cache-processor combination. However, the cycle time limiting word line, bit line, and sense-amplifier cycling were proven to work above the target cycle

time. The path operates at 720 MHz at 1.6 V and 100 C. This compares well with the simulated speed of 760 MHz. The wave forms in Fig. 10 show the test circuit in operation at room temperature with an 800-MHz cycle time. The sequence is of a write operation followed by one read operation and a portion of a second read. The signal noise is due to the use of an IDS-10 000 electron beam test setup to observe these high-speed internal signals. The write cycle begins with the sampling of the write enable "high" value at the rising edge of the clock. The address and data input values are also sampled at this clock edge. The global sense-amplifier strobe $\overline{\text{Sense Amp Enable}}$, does not pulse during the write operation to minimize power consumption and noise generation. The second positive going clock begins the read access resulting in the positive pulse of the selected word line (Word Line) about 600 ps later. The negative going global sense amplifier strobe, about 700 ps following the clock, triggers the sense amplifiers through the local inverters. The data output (Data Out) appears long after the sense amplifier strobe due to the latency control circuitry. In the 64-KByte cache, more delay will exist between the clock and array access due to wiring. Hence, the cache output delay is not limited by the latency circuit. But the wave pipelining permits the high data throughput corresponding to the clock frequency while still allowing low-frequency testing.

VI. Conclusion

A 64-KByte cache has been designed that includes the address adder in the RAM decoder. The use of an $A+B=K$ equality test allows the address add to be incorporated without a long propagation chain. An efficient combined XOR and AND/OR gate executes the added required logic with only 200 ps of added decoder latency. The delayed-reset logic in the decoder and a 2-bit SAM decoder structure allow very-high-speed logic throughput within a power budget while permitting pipelined operation. These combined factors give a 2.6-ns latency and 1.6-ns cycle time for the 64-KByte RAM. This SAM decoder permits the address add plus the cache access to be completed within two cycles in this 600-MHz processor.

Acknowledgment

The authors gratefully acknowledge the support of their colleagues in the Sun Microsystems, Microelectronics Division, SRAM group. Special thanks for their help in preparation of this paper go to J. Kaku, K. Tam, L. Rodgers, and S. Mehrotra.

References

[1] W. Lynch, G. Lauterbach, J. Petolino, and D. Poole, "Low-latency cache indexing: XOR and SAM caches," Sun Microsystems Labs, Inc., Tech. Notes SMLI93-0350, 1994.

[2] R. Heald *et al.*, "64 KByte Sum-Addressed-Memory Cache with 1.6 ns cycle and 2.6 ns latency," in *ISSCC Dig. Tech. Papers*, Feb. 1998, pp. 216–217.

[3] J. Silberman *et al.*, "A 1.0 GHz single-issue 64 b power PC integer processor," in *ISSCC Dig. Tech. Papers*, Feb. 1998, pp. 230–231.

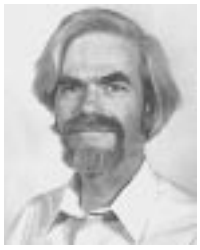
[4] L. Lev *et al.*, "A 64-b microprocessor with multimedia support," *IEEE J. Solid-State Circuits*, vol. 30, pp. 1227–1238, Nov. 1995.

[5] R. Heald and J. Holst, "A 6-ns cycle 256 kb cache memory and memory management unit," *IEEE J. Solid-State Circuits*, vol. 28, pp. 1078–1083, Nov. 1993.

[6] T. Chappell *et al.*, "A 2-ns cycle, 3.8-ns access 512-kb CMOS ECL SRAM with a fully pipelined architecture," *IEEE J. Solid-State Circuits*, vol. 26, pp. 1577–1585, Nov. 1991.

[7] D. L. Weaver and Tom Germond, Eds., *The SPARC Architecture Manual, Version 9.* Englewood Cliffs, NJ: Prentice-Hall, 1994.

[8] J. Cortadella and J. M. Llaberia, "Evaluation of $A+B=K$ conditions without carry propagation," *IEEE Trans. Comput.*, vol. 41, pp. 1484–1488, Nov. 1992.



Raymond Heald received the B.S., M.S., and Ph.D. degrees in electrical engineering from the University of California, Berkeley, in 1965, 1972, and 1975, respectively.

He currently is the Technical Lead for the global SRAM design group at Sun Microsystems Microelectronics Division, Palo Alto, CA. He began the circuit design realization of the SAM cache in delayed-reset logic. In addition to his technical direction of the SPARC SRAM circuits, he works with global power and I/O engineers as well as packaging experts to maintain high-quality signals and power supplies needed for these high-speed RAM designs. Prior to joining Sun, he designed RAM blocks and other circuitry for the Clipper family of microprocessors at Fairchild and Intergraph.

Ken Shin received the B.S.E.E. degree from Rensselaer Polytechnic Institute, Troy, NY, in 1986.

Since 1993, he has been involved in the circuit design of UltraSPARC microprocessors at Sun Microsystems, Palo Alto, CA. Presently, he manages a single-port SRAM design team for the UltraSPARC III processor. Prior to joining Sun, he was with IBM and Texas Instruments.

Vinita Reddy was born in Delhi, India. She received the M.S. degree in physics from the University of Delhi, India, and the M.S.E.E. degree from the University of Wisconsin, Madison, in 1984.

She joined Philips Semiconductor in 1984 and worked on the design of several memories and programmable logic devices from definition all the way to production. In 1992, she joined Sun Microsystems, Palo Alto, CA, where she has been involved in design and development of high-speed megacells and SRAM's for the UltraSPARC I and the UltraSPARC II processors. Currently, she is a Staff Engineer on the global SRAM design team for the UltraSPARC III.

I-Feng Kao received the B.S.E.E. degree from the University of Texas at Austin in 1995 and the M.S.E.E. degree from Stanford University, Stanford, CA, in 1996. As a Member of Technical Staff at Sun Microsystems, Palo Alto, CA, he is presently working on the data cache of the UltraSPARC III microprocessor. His interests include high-speed CMOS circuits and SRAM designs.

Masood Khan received the B.S. degree in electronics engineering technology from Texas Southern University, Houston, and the B.Sc. degree in applied physics from Karachei University, Pakistan.

He was with Texas Instruments Inc. He joined Sun Microelectronics, Palo Alto, CA, in 1992. Since then, he has been engaged with Sun Microelectronics design teams on timing, clock distribution, and SRAM design areas for SuperSPARC and UltraSPARC processors. On UltraSPARC III, he contributed to the design of the SAM Data Cache and physical TAG arrays. He joined Texas Instruments' ASP design group in 1983, where he developed memory compiler methodology for DSP and microcontroller products.

William L. Lynch received the B.S.E.E. and B.S.Cpr.E. degrees in 1982 and the M.S.E.E. degree in 1983 from Iowa State University of Science and Technology, Ames. He received the Ph.D. degree from Stanford University, Stanford, CA, in 1993.

For the last five years, he has been designing and implementing SPARC processors for Sun Microsystems, Palo Alto, CA. Currently, is an Architect of UltraSPARC III+.

Gary Lauterbach is an Engineer and Chief Architect for the UltraSPARC-III microprocessor at Sun Microsystems. He is responsible for research, design, and development of high-performance processors.

Joe Petolino received the B.S.E.E. degree from the Massachusetts Institute of Technology, Cambridge, in 1973 and the M.S. degree in computer engineering from Stanford University, Stanford, CA, in 1978.

He has been designing SPARC processors for Sun Microsystems since 1986. Prior to that, he did cache memory design for Amdahl mainframes. He has received several U.S. and foreign patents related to caches.