need for address pointers and accompanying increment/decrement logic for stack management.

The dual-stack module is intended primarily for application in stack-based VLSI processors requiring two high-performance LIFO stacks. This scheme offers better performance than dynamically partitioned stacks implemented in random access memories and is more compact than two separate LIFO hardware stacks of fixed size.

Acknowledgment

The authors are grateful to Tektronix Inc. and the Hewlett-Packard Company for donations of equipment and software. They would also like to thank C. Knighton, R. Robinson, and Y. Luan for their valuable contributions to this project.

References

[1] A. Haley, "The KDF.9 computer system," in *Proc. AFIPS FJCC*, vol. 22, 1962, pp. 108–120.
[2] D. Bursky, "Optimized processor handles Forth, and more," *Electron. Des.*, Nov. 1986.
[3] *RTX2000(TM) Real-Time Express Microcontroller, Advance Information*, Harris Corp., Melbourne, FL, 1988.
[4] R. Goodman, "An arithmetic-stack processor for high level language execution," in *Proc. 1988 Rochester Forth Conf. Programming Environments* (Rochester, NY), June 1988.
[5] K. Winters, "A mesh-structured processor architecture based on a stack-oriented instruction set," in *Proc. 1988 Rochester Forth Conf. Programming Environments* (Rochester, NY), June 1988, pp. 140–142.
[6] *Scalable CMOS Design Rules*, Rev. 6, MOSIS Service, USC/Inform. Sci. Inst., Marina del Rey, CA, 1988.
[7] M. McDonnell, "Stack logic design for the bridger array processor," Electron. Res. Lab, Montana State Univ., Bozeman, Rep. 62352-1, Mar. 18, 1988.

A Static RAM Chip with On-Chip Error Correction

TZI-DAR CHIUEH, MEMBER, IEEE,
RODNEY M. GOODMAN, MEMBER, IEEE,
AND MASAHIRO SAYANO

Abstract—**This paper describes a 2-kb CMOS static RAM with on-chip error-correction capability (ECCRAM chip). The chip employs the linear sum code (LSC) technique to perform error detection and correction. The ECCRAM chip has been fabricated in a double-metal scalable CMOS process with 3-μm feature size. Testing results of the actual chip shows a significant improvement in random error tolerance.**

I. Introduction

As the scale of computers grows, there is a tendency to put as much memory in a system as possible. Nowadays, thanks to the advent of VLSI technology, four million bits of information can be squeezed into a silicon chip. However, as more and more circuitry is packed into a chip, the reliability of memory chips becomes lower. Memory modules built from these high-density chips are vulnerable to environmental radiation and processing defects. Several researchers have proposed schemes for correcting soft and hard errors in semiconductor memory [1]–[4]. However, they are either not powerful enough, i.e., not able to correct many errors, or too complicated to be integrated onto the same chip.

Manuscript received August 11, 1989; revised February 26, 1990. This work was supported in part by NSF Grant MIP-8711568.

T.-D. Chiueh was with the Department of Electrical Engineering, California Institute of Technology, Pasadena, CA 91125. He is now with the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan 10764.
R. M. Goodman and M. Sayano are with the Department of Electrical Engineering, California Institute of Technology, Pasadena, CA 91125.

IEEE Log Number 9037795.

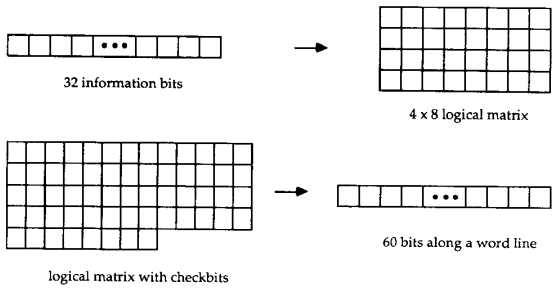

Fig. 1. Physical and logical locations of memory bits on a word line.

In this paper we describe a prototype chip, a 2-kb static RAM with on-chip error correction capability (ECCRAM). This chip employs the linear sum code (LSC) scheme developed by Fuja *et al.* [5]. Although the chip size is small, the prototype illustrates the power of the LSC scheme, which easily scales to large dynamic RAM implementation. Furthermore, small robust static RAM's are of great interest for space-borne applications. The overall chip design is divided into two phases. First, we designed a static RAM and simulated critical components of this RAM by SPICE to determine certain parameters for optimal performance. Next, we built the error correction circuitry, laid the complete chip out, and tested the prototype chip.

In Section II a brief introduction of the LSC scheme is given, and the static RAM and the error-correction circuits are presented in Section III. Section IV describes the testing results of the ECCRAM chip. Finally, conclusions are drawn in Section V.

II. Error Correction Using a Linear Sum Code

The principles of LSC's can be found in [5]; here only a brief introduction will be given. LSC was devised specifically for correcting errors in semiconductor memory chips. In the LSC scheme, bits addressed by the same word line constitute a code block; bits that carry information are arranged in the form of a logical matrix (see Fig. 1). Encoding is then performed along every row as well as every column of a logical matrix.

In the prototype chip, each word line has 32 information bits, arranged as a 4×8 logical matrix. The (13,8) single-error-correcting–double-error-detecting (SEC-DED) Hamming code is used to encode along the rows, while a parity check bit is added to each column (refer to Fig. 1). The number of parity bits per word line is $4\times5+8=28$, which is quite large compared to 32—the number of information bits per word line. Nonetheless, we feel justified since the objective of this prototype chip is to demonstrate the feasibility of the LSC scheme. Besides, as can be seen from [5], the ratio of parity bits to information bits drops drastically as memory size grows. For instance, if 1024 information bits are to be packed on a row in a 1-Mb RAM chip, only 192 parity check bits are needed to implement the LSC scheme.

In order to minimize the complexity of the LSC encoding and decoding circuitry, we chose, as the horizontal subcode, an

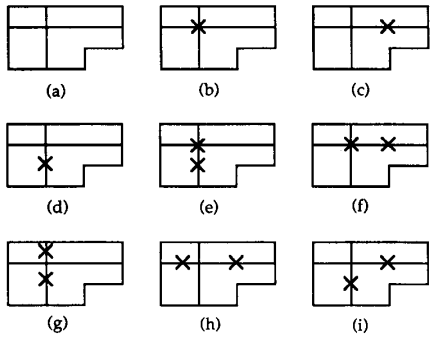

Fig. 2. Nine possible error patterns with no more than two erroneous bits.

odd-weight SEC–DED (13,8) Hamming code [6] with the following parity check matrix:

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \quad (1)$$

The syndrome vector, which is essential in locating errors in a noisy word, is given by the product of the parity check matrix (H) and the noisy word. This code requires little circuitry in syndrome vector calculation, in that double-error detection is easily done by checking if the number of ONE's in the syndrome vector is even and if the syndrome vector is nonzero (note that EXCLUSIVE-ORing any two columns component-wise yields a vector with even number of ONE's).

The minimum Hamming distance of the aforementioned LSC, i.e., (13,8) Hamming code + parity check, is $4+2-1=5$ (see [5]); hence it can correct two errors in the logical row and the logical column to which the addressed bit belongs (two-error-tolerating code). The decoding strategy of the LSC is to first decode the horizontal code word (the logical row) and the vertical code word (the logical column), respectively. Nine possible error patterns of no more than two errors in the logical row and the logical columns are depicted in Fig. 2. A case-by-case explanation is given in the following:

a) no vertical error and no horizontal error: no error;
b) vertical error and single horizontal error at the addressed bit: one error, the addressed bit;
c) no vertical error and single horizontal error in another bit: one error in the logical row, and the addressed bit is correct;
d) vertical error and no horizontal error: one error in the logical column, and the addressed bit is correct;
e) no vertical error and single horizontal error at the addressed bit: two errors in the logical column, and the addressed bit is incorrect;
f) vertical error and double horizontal errors: two errors in the logical row, and the addressed bit is incorrect;
g) no vertical error and no horizontal error: two errors in the logical column, and the addressed bit is correct;
h) no vertical error and double horizontal error: two errors in the logical row, and the addressed bit is correct;
i) vertical error and single horizontal error in another column: the addressed bit is correct.

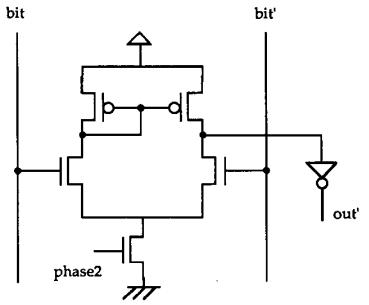

Fig. 3. Sense-amplifier circuit.

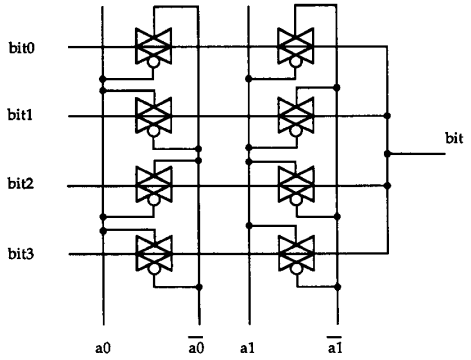

Fig. 4. Column-decoder circuit.

From the above, we conclude that the addressed bit is erroneous when there is a horizontal error (cases b) and e)), or when there is a vertical error and double horizontal errors are detected (case f)).

III. Circuit Design

A. Static RAM

The static RAM consists of four components, described as follows.

1) RAM Cell: We use the conventional six-transistor RAM cell made up of two coupled inverters holding one bit of information, which can be accessed through two transmission gates activated by the word line.

2) Sense Amplifier: The sense amplifier chosen is a simple design [7], nevertheless, SPICE simulation shows that it is good enough for this particular application (see Fig. 3).

3) Row Decoder: A static NAND–NOR type decoder is used.

4) Column Decoder: We chose a column tree decoder [8] made up of chains of transmission gates enabled by the address lines to do column decoding (see the 4MUX and 8MUX blocks in the next subsection). At any time, only one chain will be activated and a connection is set up between the output and one of the inputs. Notice that because of the bilateral characteristic of transmission gates, this decoder can work in both directions; i.e., it can be used as a multiplexer or a demultiplexer depending on which end is the driving input (refer to Fig. 4).

After some SPICE simulation, it was decided that the transistors in the word-line drivers were made eight times as wide as the minimum-size transistor, i.e., 32λ wide and 4λ long. The transmission-gate transistors in the column decoders are made three times as wide as the minimum-size transistor; the pull-down

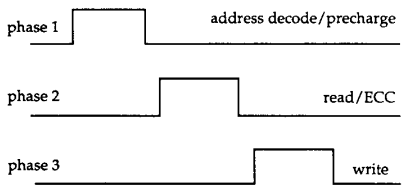

Fig. 5. Three-phase nonoverlapping clock used in the ECCRAM chip.

transistors as well as all current-source transistors in the sense amplifiers are made minimum sized so that more cells can be put on a chip.

B. Error-Correcting Circuits

For each memory access, it is necessary to first read the addressed bit, modify it, and then write it back. The reason for this read–modify–write cycle is twofold. 1) When the accessed bit is incorrect, it is better to make the correction rather than leave it alone. If one does so, the errors in the RAM may accumulate or, even worse, more errors may be generated through the encoding process. 2) During a write operation, the vertical check bit (vchk) and the five horizontal check bits (hchk0–hchk4) must be modified (the encoding process). In order to do the encoding efficiently, one first reads the addressed bit and compares it with the information bit at the input. If they are the same, nothing needs be done; otherwise, the check bits need to be modified accordingly.

To accommodate the read–modify–write cycle in every memory access, we adopted a three-phase nonoverlapping clock as shown in Fig. 5. Phase one is reserved for row address decoding and bit-line precharging. During phase two, the current-source transistors in the sense amplifiers are turned on and the sense amplifiers begin to sense the differences in bit-line voltages. After retrieving all relevant information, i.e., the logical row and the logical column that the addressed bit is in, error-correction operation is carried out. The information bit and the check bits are written back to the memory during phase three.

Fig. 6 depicts the block diagram of the 2K×1 on-chip ECCRAM chip. Now, let us briefly explain how the chip achieves error recovery and how each block works. In each memory access the 64×60 static RAM block outputs 60 b, 32 of which are information bits and the remaining 28 of which are check bits for error correction. In a write operation, depending on whether modification is required by the encoding process, some or all of the following 7 b may be modified: the information bit, the vertical check bits, and the five horizontal check bits. However, only the information bit may be modified in a read operation. The four-to-one multiplexers (4MUX) pick the 13 b in the logical row that the addressed bit is in, i.e., data0–data7 and hchk0–hchk4. The bit lines of these 13 cells (databit0–databit7, databit0′–databit7′, hchkbit0–hchkbit4, and hchkbit0′–hchkbit4′) are also selected by these multiplexers. The PCG block consists of EXCLUSIVE-OR gates, which compute all eight vertical parities (vp0–vp7) by modulo-2 summing all 5 b in each logical column, respectively. Also, note how the eight-to-one multiplexer (8MUX) does the secondary column decoding by selecting the addressed bit (data) together with its two bit lines databit and databit′ out of data0–data7, databit0–databit7, and databit0′–databit7′. On top of that, 8MUX selects the vertical error signal verror, the vertical check bit vchk, and its two bit lines vchkbit and vchkbit′ out of vp0–vp7, vchk0–vchk7, and vchkbit0–vchkbit7 and vchkbit0′–vchkbit7′, respectively.

The PLA block stores information about the parity check matrix H, which is necessary for computing the horizontal error signal herror and updating horizontal check bits. Blocks SYNDGEN and ERRSIGEN work together to produce the error signals of the horizontal subcode, i.e., herror, which indicates a single error at the addressed bit, and doublerror, which means there are two errors in the logical row. At first, the SYNDGEN block generates five syndrome bits (sy0–sy4) according to the following formulas:

$$\begin{aligned} sy0 &= data0 \oplus data1 \oplus data2 \oplus data3 \oplus data6 \oplus hchk0 \\ sy1 &= data0 \oplus data1 \oplus data4 \oplus data5 \oplus data7 \oplus hchk1 \\ sy2 &= data2 \oplus data3 \oplus data4 \oplus data5 \oplus hchk2 \\ sy3 &= data0 \oplus data2 \oplus data4 \oplus data6 \oplus data7 \oplus hchk3 \\ sy4 &= data1 \oplus data3 \oplus data5 \oplus data6 \oplus data7 \oplus hchk4 \end{aligned} \quad (2)$$

where $\oplus$ is the EXCLUSIVE-OR function. This syndrome vector (sy0–sy4) is matched against the column in the parity check matrix H that the addressed bit corresponds to (col0–col4, output of PLA). If there is an exact match, then herror becomes a ONE; i.e.,

$$herror = (sy0 \odot col0) \wedge (sy1 \odot col1) \wedge (sy2 \odot col2) \wedge (sy3 \odot col3) \wedge (sy4 \odot col4) \quad (3)$$

where $\odot$ is the EXCLUSIVE-NOR function and $\wedge$ is the logical AND function. At the same time the modulo-2 sum of sy0–sy4 is computed; if the result is even and not all the five syndrome bits are ZERO, then doublerror becomes a ONE. In other words

$$doublerror = \{\neg (sy0 \oplus sy1 \oplus sy2 \oplus sy3 \oplus sy4)\} \wedge (sy0 \vee sy1 \vee sy2 \vee sy3 \vee sy4) \quad (4)$$

where $\neg$ is the logical negation function and $\vee$ is the logical OR function. The LOGIC block then generates the error signal (err), which indicates if the addressed bit is incorrect, by computing herror $\vee$ (verror $\wedge$ doublerror).

The three update blocks BITUPDATE, VPUPDATE, and HPUPDATE are responsible for modifying the addressed bit, the vertical parity check bit, and the five horizontal parity check bits, respectively. At first, the corrected output bit (Dout) is obtained by EXCLUSIVE-ORing the raw data bit (data) with the error signal (err). At this juncture if a read operation is being carried out, then Dout is written back to correct the possible soft error in the accessed memory cell. Also, Dout is available to the circuits outside the ECCRAM chip. On the other hand, if it is a write operation, then Din gets written back. Furthermore, all the vertical and horizontal parity check bits need to be updated in order to perform encoding. This process is done by first calculating the EXCLUSIVE-OR of Din and Dout; if the result is a ONE, the vertical check bit as well as all the horizontal check bits that check the addressed bit are complemented; otherwise they stay intact. We also include a control signal (eccenb) to enable/disable the error-correcting-code encoding and decoding processes. When this signal is high, normal encoding/decoding procedures are performed on every memory access; otherwise the ECCRAM acts like a normal static RAM without error-correction capability.

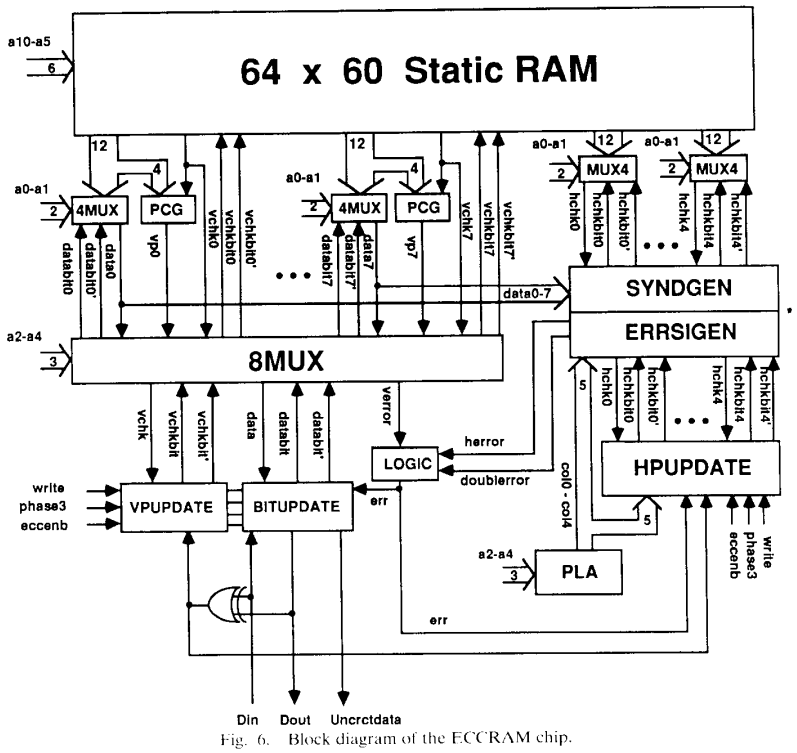

Fig. 6. Block diagram of the ECCRAM chip.

As can easily be seen, extra circuits for the LSC scheme are: PCG, SYNDGEN, ERRSIGEN, PLA, LOGIC, BITUPDATE, VPUPDATE, and HPUPDATE, which constitute less than 10% of the total chip area.

IV. Testing Results of the ECCRAM Chip

The ECCRAM chip was then fabricated through MOSIS (see Fig. 7) and tested. The testing procedure deals mainly with the case when the RAM is inflicted with random 1-b errors. No testing on row failures was done since the LSC scheme cannot function properly if the whole row is defective.

The testing procedure is as follows. At first, the chip is cleared. This loads all memory locations with zero bit values. Random addresses are then selected and random data (ZERO or ONE) are written into those addresses. A random data entry pattern is produced by repeating this 4096 times. This random data entry pattern is recorded into the ECCRAM chip with the error correction enabled; therefore, the parity check bits are all set as required by the encoding algorithm. Next, error correction is disabled, and a predetermined number of bits, again randomly selected, are flipped, which serves to simulate alpha-particle errors or random memory cell failures. Finally, the error correction is again enabled and the data in the chip are read sequentially from the lowest valued address to the highest. These results are compared with the initial data pattern; the number of discrepancies between the original and recorded data patterns is counted.

To obtain a statistically accurate value for the resulting bit-error probability, this cycle of writing a data pattern, inducing errors, and reading back the corrected version was repeated 200 times for each number of induced errors. For each trial a different data pattern and error pattern were generated.

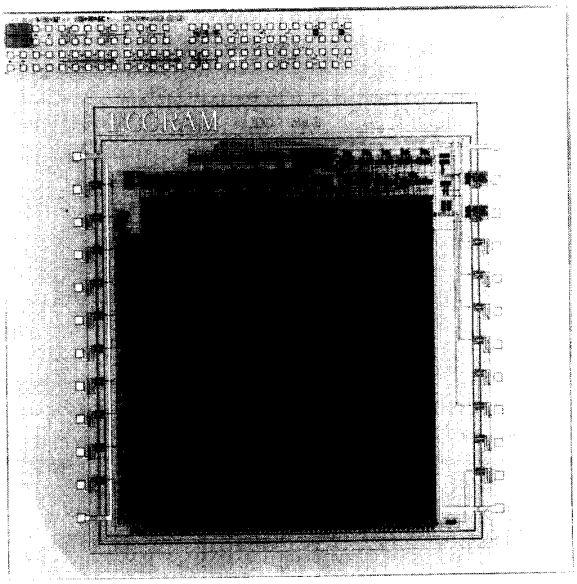

Fig. 7. Microphotograph of the ECCRAM chip.

A computer simulation of the chip was conducted to test the accuracy of the results. This model mimicked every circuit of the ECCRAM chip. The data are recorded in arrays, and parity check bits are computed. Reading and writing data to the chip with error correction activated requires that the syndromes and error signals be computed and the data be updated accordingly. The simulation was conducted using identical testing procedures

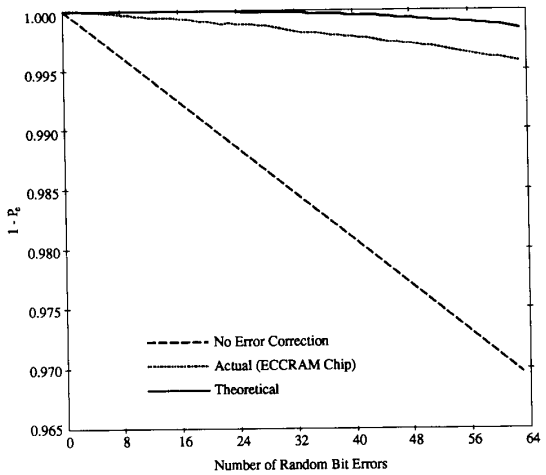

Fig. 8. Performance of the ECCRAM chip with comparison to the theoretical LSC scheme and the no error-correction case.

as the actual chip. An identical pseudorandom number generator was used in both cases, and both the simulations and the chip tests were run for 200 trials. The results are plotted as the bit-error probability—the relative number of bits that remain in error after error correction compared to the total number of bits read—for cases of 0 to 63 random errors. Also plotted are the cases when no error correction is done and when the ECCRAM is used (see Fig. 8).

As can be seen in Fig. 8, the computer simulation predicts a lower bit-error probability (and therefore a higher probability that the bits are correct) than that observed in the actual ECCRAM chip. Further logical testing of the circuits shows that the error-correction circuitry and memory cells did perform as expected; this discrepancy was found to be caused by a minor error in the memory cell selection circuitry.

Conceptually, however, the on-chip error correction does operate as expected, and the resulting decrease in bit-error probability is large.

V. Conclusions

In this paper, we present a static random access memory chip with on-chip error correcting capability (the ECCRAM chip). The coding scheme adopted in this chip is the linear sum code (LSC), which is designed with a view to efficiently correcting errors in memory chips. Unlike the simple parity-check memory modules, which can detect one error in, say, 8 b, the LSC-based ECCRAM is capable of correcting error at any addressed bit as long as there are no more than two errors in the 17 b (the logical row and column) associated with that addressed bit. The results show that significantly larger error recovery capability is present in the ECCRAM chip compared to memory chips without error correction.

References

[1] L. Edwards, "Low cost alternative to Hamming codes corrects memory errors," *Computer Design*, pp. 143–148, July, 1981.
[2] T. Mano, J. Yamada, J. Inoue, and S. Nakajima, "Circuit techniques for a VLSI memory," *IEEE J. Solid-State Circuits*, vol. SC-18, no. 5, pp. 463–469, Oct. 1983.
[3] F. I. Osman, "Error-correction technique for random access memories," *IEEE J. Solid-State Circuits*, vol. SC-17, no. 5, pp. 877–881, Oct. 1982.
[4] J. E. O'Toole, T. M. Trent, and W. D. Parkinson, "256K dynamic error corrected RAM," Micron Technologies, Inc., Boise, ID, memo, 1982.
[5] T. Fuja, C. Heegard, and R. M. Goodman, "Linear sum code for random access memories," *IEEE Trans. Computers*, vol. 37, no. 9, pp. 1030–1042, Sept. 1988.
[6] C. L. Chen and M. Y. Hsiao, "Error-correcting codes for semiconductor memory applications: A state-of-the-art review," *IBM J. Res. Develop.*, vol. 28, no. 2, pp. 124–134, Mar. 1984.
[7] O. Minato *et al.*, "2K×8 bit Hi-CMOS static RAM's," *IEEE J. Solid-State Circuits*, vol. SC-15, no. 4, pp. 656–660, Aug. 1980.
[8] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design—A System Perspective*. Reading, MA: Addison-Wesley, 1985.

An Improved Latching Pulse Design for Dynamic Sense Amplifiers

JIANN S. YUAN, MEMBER, IEEE, AND
JUIN J. LIOU, MEMBER, IEEE

Abstract—**A generalized optimal latching pulse for dynamic sense-amplifier design has been derived. The model equations account for threshold voltage imbalance, bit-line capacitance imbalance, current gain imbalance, gate capacitance and intra-bit-line capacitive coupling effect, channel-length modulation, source-body effect, and temperature sensitivity in a unified manner. Computer simulations of the analytical equations, including those effects, are presented. A design implementation for fast sense-amplifier operation is also presented to demonstrate the utility of the model equations for practical application.**

I. Introduction

As dynamic memories become faster, the stored charge in a given bit gets smaller. The detection of small voltage signals becomes very difficult in sense-amplifier operation, especially for multimegabit DRAM's where significant bit-line coupling noise occurs during initial sensing and amplification [1]. The available time required to properly sense the data, amplify, and detect it correctly decreases for high-speed DRAM's [2]. In addition, the process imbalance and high-temperature operation slow down sensing speed. Thus, it is essential to develop a generalized optimum sensing waveform under constraints of coupling noise, process imbalance, sensing speed, power dissipation, temperature variation, and maximum layout area.

The sense-amplifier (SA) operations have been investigated by numerous authors in the past 15 years [3]–[7]. Generally, the optimum waveform derived by Linch and Boll [3] was improved by [5], [7] including the threshold voltage V_T, bit-line capacitance C_b, and current gain β imbalances. We extend their analysis to take into account the intra-bit-line capacitive coupling of a sense amplifier in high-density megabit DRAM's and physical effects of channel-length modulation, body effect, and temperature fluctuation.

In this paper, generalized equations accounting for the above-mentioned effects are presented in a unified manner. The model equations are derived in Section II. Computer simulated

Manuscript received July 28, 1989; revised June 4, 1990.
J. S. Yuan was with Texas Instruments Incorporated, Dallas, TX. He is now with the Department of Electrical Engineering, University of Central Florida, Orlando, FL 32816.
J. J. Liou is with the Department of Electrical Engineering, University of Central Florida, Orlando, FL 32816.
IEEE Log Number 9038257.