A Reconfigurable Parallel Signature Analyzer for Concurrent Error Correction in DRAM

PINAKI MAZUMDER, MEMBER, IEEE, JANAK H. PATEL, FELLOW, IEEE, AND JACOB A. ABRAHAM, FELLOW, IEEE

Abstract—**An efficient strategy to utilize a parallel signature analyzer (PSA) for concurrent soft-error correction in DRAM's is described. For a two-level w-bit, n-word memory system, the proposed technique needs only one additional chip as opposed to $\log_2 w+2$ in the conventional Hamming code. Such an error-correction circuit (ECC) significantly improves the reliability of the memory system.**

I. Introduction

The concept of fast data compaction by using a parallel signature analyzer (PSA) was originally proposed by Benowitz *et al.* [1]. Sridhar [2] had designed a testable memory architecture incorporating the PSA within the DRAM chip to test several cells on a row (word line) in a single memory cycle, and he demonstrated how to speed up the quadratic run time of the Walking 1's and 0's test procedure to linear run time. In order to test the memory cells in parallel, at first a test vector was sequentially scanned into the PSA. The content of the PSA was then used to write in parallel to multiple cells in the selected word line, and subsequently, when these cells were read in parallel, a signature was generated. To determine whether a DRAM chip is fault-free, the scan-out pin of the PSA (quotient bit) was continuously monitored, and the final signature at the end of the test procedure was verified. Using this test strategy, we have examined several other memory test algorithms, and noted that most of the functional test procedures, except Marching algorithms, can be substantially accelerated (as shown in Table I). The objective of this paper is to demonstrate how to utilize the presence of an on-chip PSA for correcting a single-bit soft error during the normal use of the memory chip. Several strategies [3]–[6] have been proposed in the past to correct soft errors in a memory system using an on-chip error-correction circuit (ECC). This paper demonstrates how to construct an on-chip ECC by reconfiguring the PSA during normal operation into a parity generator which detects the occurrence of a single-bit error.

The proposed scheme utilizes the two-level organization of a hierarchical memory system, and it requires one parity bit for each row (word line) in the DRAM chip to detect whether any error has occurred within the chip. A single-bit error in the memory system can be corrected by the proposed scheme by adding one extra chip containing the parity information for all memory words. Conventional memory systems use the Hamming code to correct a single-bit error and to detect a double-bit error (SEC/DED). In a two-level memory system, with n memory words having w bits/word, altogether a w number of $n\times 1$-b RAM chips is used. Thus, the Hamming code requires $\log_2 w+2$ additional chips to store the error checking bits in a code word. In a cost-efficient memory system design, the proposed error-correcting scheme provides an economy of $\log_2 w+1$ numbers of DRAM chips for a codeword with w information bits. Moreover, in the proposed scheme whenever a w-bit memory word is read out, $w\sqrt{n}$ DRAM cells in the memory system are sensitized to detect the occurrence of memory-cell upsets, while in the conventional Hamming code technique only w cells in the memory system are sensitized. By sensitizing $w(\sqrt{n}-1)$ extra memory cells, the proposed technique improves the reliability (MTBF) of the memory system considerably.

Manuscript received September 30, 1988; revised December 11, 1989. This research was supported in part by the NSF Research Initiation Awards under Grant MIPS 8808978, by SRC under Grant 86-12-109, by ONR under Grant N00014-85-K-531, by Bell Northern Research Laboratory, by Digital Equipment Corporation, and in part by the URI-Army under Grant DAAL-03-87-K0007.

P. Mazumder is with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109.

J. H. Patel is with the Coordinated Science Laboratory, University of Illinois, Urbana, IL 61801.

J. A. Abraham is with the Department of Electrical and Computer Engineering, University of Texas, Austin, TX 79712.

IEEE Log Number 9034581.

II. The Proposed Error-Correction Technique

A two-level memory organization of a w-bit, n-word memory system is shown in Fig. 1. The level-2 array consists of $w+1$ DRAM chips in which the last chip is used to store the parity bit of the word. Each memory chip consists of a two-dimensional array of $\sqrt{n}\times\sqrt{n}$ memory cells for data storage and an extra $\sqrt{n}$ memory cells (one on each column) to store the parity bit for the corresponding on-chip word line. Thus the proposed technique of concurrent error detection and correction uses a two-level parity coding. If $\oplus\sum_{i=0}^{w} P_i^1 \oplus P^2 = 1$, then a fault is sensitized, and by monitoring P_i^1 for all i and P^2, the faulty chip can be identified, where P_i^1 is the level-1 parity of the ith chip and P^2 is the level-2 parity bit, as shown in Fig. 2. If $P^2 = P_i^1 = 1$, then the selected cell in the ith chip is erroneous and is corrected by complementing its value. Table II shows all possible outcomes, and the conditions for occurrence of different types of errors. It can be seen that in row 2, even though the word is error-free, a single-bit soft error is detected, and it should be immediately diagnosed for reducing the soft-error rate. In the conventional system-level Hamming code, this soft error will be latent until the faulty bit is addressed for a READ operation. If before this faulty bit is read one more memory cell in the same location in another chip becomes faulty, then the Hamming code will not be able to correct the faulty bit, and thereby the reliability of the memory system will be poor. The proposed scheme can detect two single-bit errors, and it can automatically correct the addressed bit if it is faulty, as illustrated in row 4 of Table II. Fig. 3 illustrates the different examples of errors that can be corrected by the proposed code. Altogether there are $w+1$ memory chips, each of n cells (organized as $\sqrt{n}\times\sqrt{n}$ array), describing a solid of volume $|X|\times|Y|\times|Z|$. Each w-bit word consists of one bit from each chip. Such a line will be referred to as a $\langle x, y, Z\rangle$ line, shown by the dashed line in Fig. 3. Thus, each time a memory reference is made, $(w+1)\times\sqrt{n}$ cells will be sensitized. These cells will comprise a plane (X, y, Z) as shown in Fig. 3. If only one chip is completely faulty due to catastrophic failure or defective chip-select line, then such a fault, denoted by the (X, Y, z) plane, will be detected by the proposed scheme. If the word-line driver within a chip is faulty, then such a fault can be detected by the proposed scheme as long as the shaded plane contains no more than one such defective row. If the sense amplifier or bit line within a chip is defective, then such a fault, denoted by the line $\langle x, Y, z\rangle$, can also be detected by the proposed scheme.

In the READ mode, the parity bit of the selected level-1 word line is generated and checked with the content of the parity bit cell. Thus the PSA hardware has two functionalities: 1) updating

TABLE I
TEST SPEEDUP BY PSA

Algorithm	Test Size (Sequential)	Test Complexity (PSA)	Speedup	Fault Coverage
MSCAN	$4n$	$4n/p$	p	Not Reduced
Column Bar	$4n$	$4n/p$	p	Not Reduced
MATS	$4n$	$2n$	2	Reduced
Marching	$14n$	$6n$	1.3	Reduced
Walking 0/1	$2n^2+6n$	$2n^2/p+4n$	$p/(1+2p/n)$	Reduced
GALPAT	$4n^2+2n$	$2n^2/p+4n$	$p/(0.5+p/n)$	Reduced

Here p is the size of the PSA

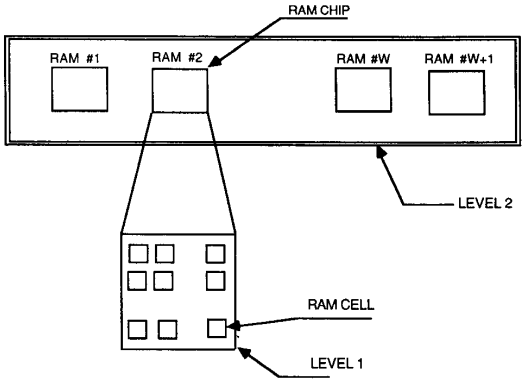

Fig. 1. Two-level memory system.

the parity bit and 2) generation of the parity bit, as discussed below.

A. Updating the Parity Bit

At first the whole memory is initialized to zero, and subsequently whenever a transition write is made in the DRAM chip, the parity bit is complemented. In order to ascertain whether a WRITE operation results in changing the content of a memory cell (i.e., a transition write), the selected cell is at first read and stored in the data-out buffer. The data to be written are available in the data-in buffer and are XORed with the value in the data-out buffer to determine whether the WRITE operation is a transition write. While reading the content of the desired memory cell, the parity bit can also be simultaneously read and stored in an additional buffer. The parity bit is toggled whenever a transition write is made. Because of the extra READ operation prior to a WRITE operation, the performance of the memory will be slightly degraded.

B. Generation of the Parity Bit

The parity bit can be generated by utilizing the XOR gates available in the PSA (shown in Fig. 4). In a DRAM, when a memory cell is read, all the bit lines are precharged and then the word line containing the desired cell is selected. The contents of all the cells in the selected word line can be simultaneously read from their respective bit lines. But only the content of the selected bit line is transferred to the data-output buffer. Thus in the normal mode of operation of the memory, the content of a complete word line can be accessed by the PSA. It can be seen from Fig. 4 that the bit lines are directly connected to the input of the XOR gates, while the other input of the XOR gate (in the *signature* mode) is connected to the output of the preceding XOR gate through the flip-flop of the preceding stage (in some cases an additional XOR gate which is used for feedback polynomial). In order to generate the parity bit for the word line, the flip-flops and the XOR gates in the feedback path in Fig. 4 should be bypassed.

In order to bypass the XOR gates in the feedback path, points A and B in Fig. 4 are disconnected and an additional switch is introduced, shown by the dotted box. The switch is driven by a signal called *TEST*. During the test mode, $TEST = 1$ and points A and B are connected through the switch, as shown in Fig. 4. During normal operation, when the PSA is reconfigured into ECC, $TEST = 0$ and the switch connects point B to ground; therefore all the XOR gates in the feedback path will be bypassed.

In order to understand how the flip-flops are bypassed, it is necessary to understand how the PSA circuit is implemented in the memory. Fig. 5 shows a typical MOS implementation of the jth flip-flop stage of the PSA in Fig. 4. The PSA is usually implemented in dynamic logic where ϕ_1 and ϕ_2 are used as two nonoverlapping clocks. During the test mode of the PSA, the signal $TEST = 1$ and the PSA can be operated in READ or signature mode by setting $WRITE = 0$ and $MODE = 1$, and in WRITE mode by setting $WRITE = 1$ and $MODE = 0$. In the *scan* mode, when the test data are serially loaded into the PSA, the signal lines are set to $MODE = 0$ and $WRITE = 0$. The flip-flop consists of two inverters $G2$ and $G3$ which are back-to-back connected in the test mode (when $TEST = 1$) by the transistor $Q7$ when clock $\phi_1 = 1$. In the normal mode, the feedback is removed by the signal $TEST = 0$ and the flip-flop degenerates into a cascade of inverters. In the signature mode, the signal at bit line B_j is XORed with the content of the preceding flip-flop stage, FF_{j-1}, by the pass transistors $Q1$ and $Q2$. This value is buffered into inverter $G1$, and when $\phi_2 = 1$, it is forwarded to the input of the flip-flop for storage. In the scan mode, the value of the preceding flip-flop stage FF_{j-1} is directly passed through the transistor $Q4$ and stored at the flip-flop FF_j. In the WRITE mode of the PSA, the flip-flop is isolated by the transistor $Q6$, which remains cut off. Transistor $Q8$ turns on and the value of FF_j is written on a memory cell being routed through the bit line B_j. In the normal mode of DRAM operation when the content of a memory cell is read, the bit values of all the cells in the corresponding word line of the DRAM appear at the inputs of the PSA. The inverters $G2$ and $G3$ simply forward the output of $B_j \oplus B_{j-1}$ to the next stage. Thus the PSA forms an XOR cascade and thereby a parity generator as shown in Fig. 6(a). In an n-bit RAM, the parity generator takes $O(\sqrt{n})$ time to detect a single-bit memory upset. This inordinate delay may reduce the effective use of memory cycles. This delay can be improved to $O(\log_2 n)$ by the addition of extra XOR gates to form a parity tree as shown in Fig. 6(b). The delay can be further reduced by a constant factor by bypassing transistors $Q5$ and $Q6$, and invert-

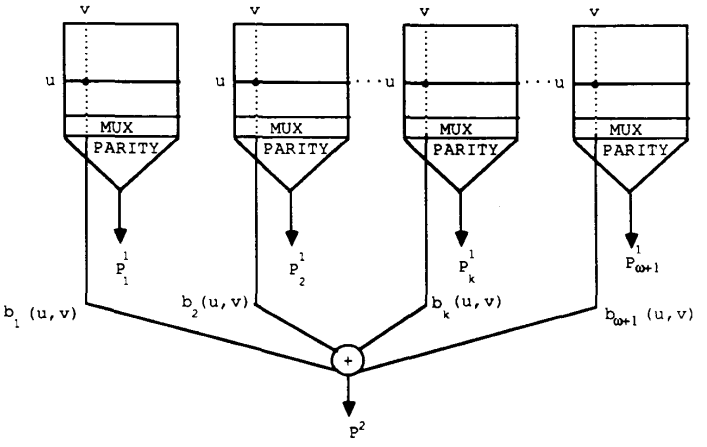

Fig. 2. Two-level parity code.

TABLE II
DIFFERENT TYPES OF ERRORS

P^2	P_i^1	P_j^1	Remarks
No Error	No Error	No Error	Error-Free Word
No Error	Error	No Error	Error-Free Word One 1-Bit Error Detected in i-th RAM
No Error	Error	Error	Two 1-Bit Errors in the Word Detected
Error	Error	No Error	One 1-Bit Error in the Word Detected & Corrected
Error	No Error	No Error	One 1-Bit Error in the Word Detected Two 1-Bit Error in i-th or j-th RAM Detected
Error	Error	Error	One 1-Bit Error in the Word Detected, Not Corrected One 1-Bit Error in i-th and j-th RAM Detected

P^2: Level-2 Parity Bit, P_i^1: Level-1 Parity in i-th Chip, P_j^1: Level-1 Parity in $j(\neq i)$-th Chip

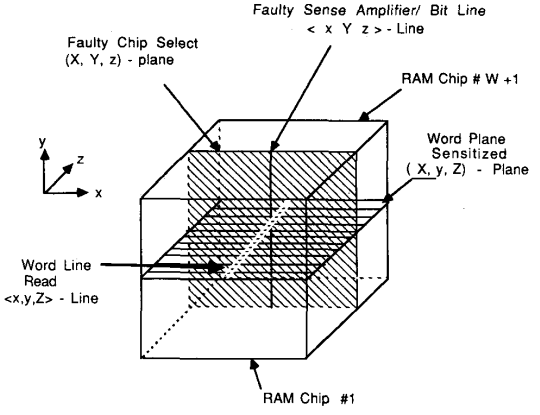

Fig. 3. Examples of correctable errors.

ers $G1$ and $G2$ of Fig. 5, by an extra pass transistor during the normal mode of DRAM operation.

The last bit B_m of the PSA is connected to the bit line of the parity bits. Thus the quotient bit of the PSA, which is available at the scan-out pin of the testable DRAM, indicates whether a single-bit error has occurred within the memory. While reading the content of a memory cell, if all the memory cells on the corresponding word line in the DRAM are correct, then the scan-out pin is at low voltage. On the contrary, if any of those memory cells is upset, the scan-out pin will be at high voltage. While reading any memory location inside a DRAM, if the scan-out pin voltage is high and the parity bit in level-2 memory indicates an error, then it is known that the memory location is faulty and its bit value should be complemented for error correction. On the other hand, if only the scan-out pin indicates an error and the parity bit of level-2 memory does not indicate any error, then the faulty cell lies on the corresponding word line. If an error is detected and not corrected, because it is not known which memory cell is faulty, then it is required to locate the faulty cell either by hardware or software. Product codes [5], [7] with orthogonal parity, or row and diagonal parity schemes applied over multiple word lines, are not suitable for memory applications. A bidirectional parity scheme [3], where all the information and parity bits in a rectangular code are stored on a single word line, can be used, but this will need about $2n^{1/4} \times n^{1/2}$ extra memory bits within each DRAM, since each row of $n^{1/2}$ bits will be organized as a square containing $n^{1/4}$ horizontal parities and $n^{1/4}$ vertical parities. So the strategy used here is to locate the faulty cells (if it cannot be readily located by the two-level parities) by sequentially reading the memory cells on the defective word line. Since the error rates are very low in the DRAM's with α-particle protective film, this on-line periodic removal of faulty bits ensures that no double error occurs on a word line, and thus the proposed scheme maintains high reliability. It may be noted that in the conventional Hamming coding

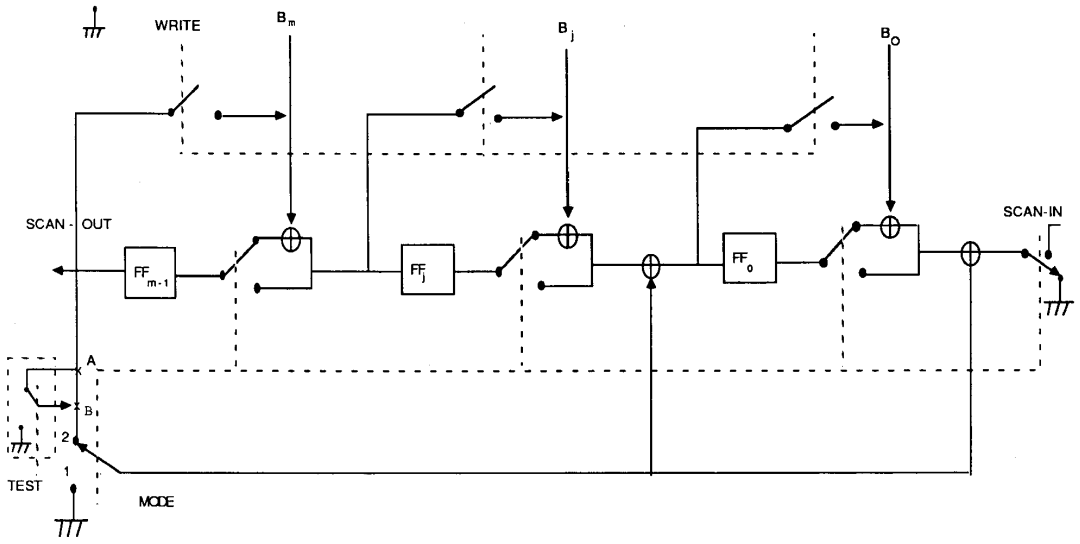

Fig. 4. Functional diagram of a PSA.

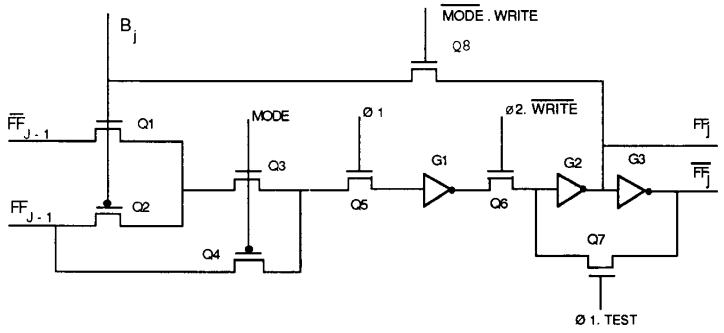

Fig. 5. MOS circuit of one cell of a PSA.

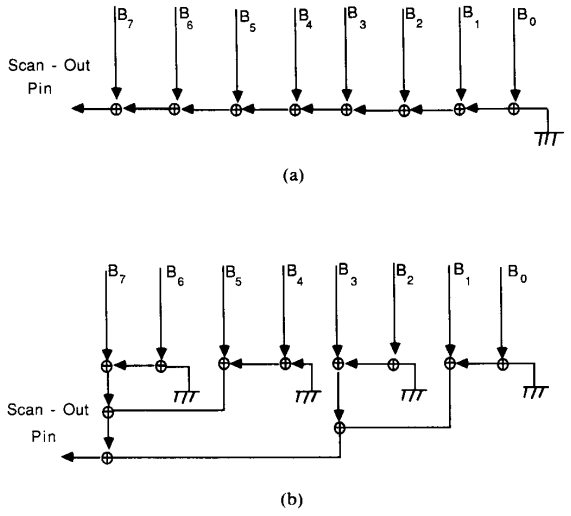

Fig. 6. Parity generator utilizing a PSA.

this cannot be done, because it only sensitizes the cell which is read out of the chip, and not the whole row (word line) in the chip.

III. Final Remarks

This paper demonstrates how to reconfigure the on-chip testability logic, such as the PSA, to correct a single-bit soft error in a two-level RAM system. The PSA can be integrated within a high-density DRAM to augment the testability and reduce the testing cost of the memory. The proposed scheme uses only one extra DRAM chip to store the parity bits of the system words, and it has very little overhead. The idea of reconfiguration of testable hardware into ECC further reduces the additional chip area. By making a simple reliability analysis, it can be shown that the MTBF for a DRAM with the proposed error is given by $1.25/\lambda n^{3/4}$, where n is the number bits in a chip and λ is the average chip failure rate. For a memory without any ECC the MTBF is given by $1/\lambda n$. Thus, the improvement in reliability due to the proposed error-correction scheme, defined by the ratio of these two MTBF's, is RIF = $0.8n^{1/4}$, and monotonically increases with the size of the memory array.

REFERENCES

[1] N. Benowitz *et al.*, "An advanced fault isolation system for digital logic," *IEEE Trans. Comput.*, vol. C-24, pp. 489–497, May 1975.
[2] T. Sridhar, "A new parallel test approach for large memories," in *Proc. Int. Test Conf.*, 1985, pp. 462–470.
[3] T. Mano *et al.*, "Circuit techniques for a VLSI memory," *IEEE J. Solid-State Circuits*, vol. SC-18, no. 5, pp. 463–469, Oct. 1983.
[4] J. Yamada, "Selector-line merged built-in ECC technique for DRAM's," *IEEE J. Solid-State Circuits*, vol. SC-22, no. 5, pp. 868–873, Oct. 1987.
[5] R. M. Tanner, "Fault-tolerant 256K memory designs," *IEEE Trans. Comput.*, vol. C-33, pp. 314–322, Apr. 1984.
[6] F. I. Osman, "Error-correction techniques for random-access memories," *IEEE J. Solid-State Circuits*, vol. SC-17, no. 5, pp. 877–881, Oct. 1982.
[7] W. W. Peterson and E. J. Weldon, in *Error Correcting Codes*. Cambridge, MA: MIT Press, 1972.

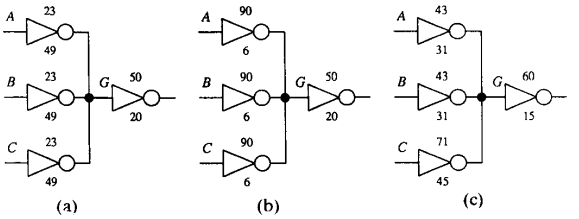

Fig. 1. Three GCMOS circuits with identical topology, but different functions: (a) OR3, (b) AND3, and (c) $A \cdot B + C$. Number above inverter is p-transistor width; number below is n-transistor width. All channel lengths = 3 μm.

Ganged CMOS: Trading Standby Power for Speed

KENNETH J. SCHULTZ, ROBERT J. FRANCIS, AND KENNETH C. SMITH, FELLOW, IEEE

***Abstract*—This correspondence presents ganged-CMOS logic (GCMOS), a technique employing CMOS inverters with their outputs shorted together, driving one or more encoding inverters. These encoding inverters, serving to quantize the nonbinary signal at the "ganged" node, effectively buffer it from external circuitry, thus allowing locally smaller noise margins. As demonstrated by two novel adders, GCMOS achieves higher speeds and lower input capacitances than static CMOS, at the expense of higher static power dissipation.**

I. INTRODUCTION

Fully complementary CMOS static circuits dissipate negligible dc power, can operate asynchronously, and do not require the routing of clock signals [1]. However, static circuits are generally slower than dynamic circuits. One way of overcoming this deficiency is to trade off standby power consumption for speed. Johnson [2] recently presented a novel CMOS NOR gate using inverters with their outputs shorted together. The design of such NOR's involves transistor ratioing to set appropriate high and low output levels.

The static power–speed trade-off is acceptable in localized applications, where there is a demonstrated need for a special function to operate particularly quickly. The static power dissipation is thus kept physically isolated to a few locations on an IC, and may in fact be insignificant in an environment where conventional static circuits operate near their maximum frequency, and thereupon dissipate considerable dynamic power. It may be noted, moreover, that the local nature of the concept does not preclude the use of other speed-enhancing techniques, and thus its use can provide the incremental delay improvement which may make a design feasible.

This correspondence extends the concepts in [2] to include buffering of the shorted or "ganged" node, thereby allowing the realization of more complex gates, and thus, the idea called "ganged-CMOS logic" (GCMOS). A number of sample circuits are presented. In particular, two novel adders are described and compared with an accepted conventional implementation. The ganged-CMOS adders provide lower input capacitance and faster carry propagation, for equally sized layouts.

Manuscript received December 13, 1989.
The authors are with the Department of Electrical Engineering, University of Toronto, Toronto, Ont., M5S 1A4, Canada.
IEEE Log Number 9035670.

II. GANGED CMOS

By buffering the ganged node with a simple CMOS inverter, a number of advantages are obtained. First of all, the ganged node is effectively isolated from external circuitry—its value is neither transmitted on long interconnect wires and corrupted by noise, nor does it drive complex gates, where any voltage exceeding a transistor threshold can cause a logic error. Essentially, one can tolerate much lower noise margins on a local node than on a global node. This benefit is enhanced by the inverter's inherent encoding action—its high gain results in a sharp distinction between low and high inputs. Furthermore, the inverter's switching point, while dependent on the square root of the p-n ratio, can be varied adequately by adjusting transistor geometries.

Fig. 1 shows the same circuit topology repeated three times; transistor widths are changed to realize three different functions. The first, in Fig. 1(a), implements an OR gate; only one of the three inputs need be high for the ganged node G to be forced well below the switching point of the buffer inverter. Similarly, the dimensions shown in Fig. 1(b) implement an AND gate. The circuit in Fig. 1(c) implements the logic function $A \cdot B + C$; the inverter driven by C is essentially "twice" as strong as those driven by A and B.

GCMOS results in a lower transistor count for more complex functions, as demonstrated in the examples that follow. Although it is true that the exclusive use of inverters limits the area-saving parallel and series layout of transistors, the lower transistor count overcomes this area deficiency. For further area savings, it is possible to group n-channel and p-channel transistors in common tubs, such that the inverter's two transistors are not constrained to be physically adjacent. It is also possible to alter p- and n-area requirements by enhancing the encoding-inverter threshold (for example, using narrower p-transistors for both input and encoding inverters, if the encoding-inverter switching point is lowered).

III. GLAD: GANGED-LOGIC ADDER WITH DOUBLE GANGED NODES

Two ganged nodes, with their input and output inverters weighted to implement different functions, are present in the adder circuit shown in Fig. 2. The full adder is realized using