JOURNAL OF ELECTRONIC TESTING: Theory and Applications, 2, 293-306 (1991)
© 1991 Kluwer Academic Publishers, Boston. Manufactured in The Netherlands.

Redundancy Effect on Yield of Binary Tree RAMs

B. CICIANI
Electronics Engineering Department, University of Rome II, Via Orazio Raimondo 8, 00173 Roma, Italy

Received November 5, 1990.
Editor: D.K. Pradhan

Abstract. Recently, a radically new RAM architecture was proposed by Jarwala and Pradhan [10] called TRAM architecture. A 64M version of this architecture is being prototyped by a DRAM manufacturer in Japan. The yield sensitivity of this binary Tree Dynamic RAMs (TRAMs) at variations in tree-depth and redundancy level is investigated in this article. It is analyzed not only the yield of all good chips, but also the probability of generating partially good chips. To this purpose a new stochastic yield model, overcoming the drawbacks of the existing ones, is used. The model is a straightforward one and easy to use in parametric studies of chip's yield versus redundancy level and reconfiguration strategies.

Key words: Fault-tolerant memories, semiconductor memory design, VLSI chip design, yield evaluation.

1. Introduction

The binary tree architecture is not only a versatile architecture for parallel processing applications [22, 8], but also a good answer to overcome the limits of traditional architecture memory chips. Indeed Jarwala and Pradhan [10] have shown that large dynamic RAMs implemented according to this architecture (which partitions the RAM into modules, each appearing as the leaf-node of a binary interconnect network) can be faster (in terms of lower access time as well as reduction in the refresh time) than those which use traditional implementations; besides testability is improved thanks to the reduction of the size of the array under test and thanks to parallel testing. These benefits, when there are no redundancies, are obtained at only a small increase in chip area.

This architecture is radically different from conventional RAM architecture and is the subject of significant industrial interest. In particular a 64M prototype of this design is being built by a DRAM manufacturer in Japan.

A major advantage of this architecture is its partitionability [10], which makes it easy to generate partially good products and therefore enhance the effective manufacturing yield. In their article [11] Jarwala and Pradhan evaluated the yield and wafer equivalent yield of the binary tree random access memory (TRAM) without considering the use of redundancy techniques [25, 20] in order to improve the yield figure of merit. Instead, in this article the yield sensitivity of this kind of organization at the variation of the tree depth and of the redundancy level will be investigated. As in [10] we shall analyze not only the yield of all good chips, but also the probability of generating partially good chips [25]. Yield improvement is obtained using redundancy, but redundancy is not free from penalties: insertion of spare rows and columns contributes to a larger chip area, and degradation of performance due to a longer access time can also occur [30, 6]. The presence of redundancy can also cause a loss of productivity, because the increase in chip density can result in the failure of the sparing circuitry [30]. Then, for any given tree depth the problem has to be dealt with of the choice of the redundancy level which gives the best benefit/cost ratio. A convenient figure of merit is the yield/area ratio [30]. A better benefit/cost figure of merit should include the access time for the memories and/or a testability index (as testing-cost at different chip processing phase [16], or testing time, or fault coverage), but it poses the problem of how to weigh these indices against the cost factor.

Finally, area evaluation in this paper will be performed according to the approaches introduced in [9] and [10], taking into account the reconfiguration strategy and associated redundancies. As for the yield-evaluation, we shall use a straightforward model which overcomes many of the drawbacks of the existing yield

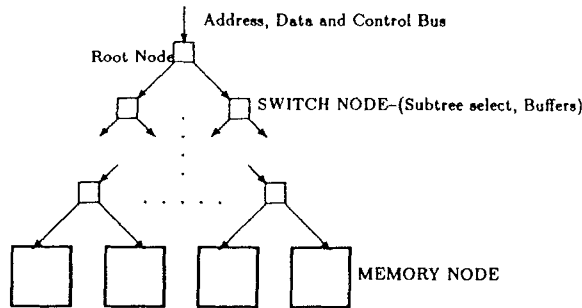


Fig. 1a. TRAM architecture—conceptual schematic.

models [4]. In Section 2 the binary TRAM architecture and properties is presented according to [10]. Section 3 presents the fault-tolerant techniques adopted for the binary TRAM architecture to improve the yield. In Section 4 the model for area cost is explained. Section 5 focuses the yield evaluation for all and partially good chips. Numerical results are presented in Section 6 along with further discussion. Concluding remarks appear in Section 7.

2. Architecture and Properties of Binary Tree Dynamic RAM

As specified in [10], the binary TRAM of size $N = 2^n$ is divided into modules, each of which appears as a leaf-node in a binary tree as shown in figure 1(a). The depth of the tree, the number of nodes and their size are related by:

$$N = 2^n = 2^{mf} 2^{l-1}$$

where N is the size of the RAM in bits, n the number of the address lines, l is the depth of the tree and 2^{mf} defines the size of each leaf-node.

Let $MF = 2^{mf}$ be the size of each leaf node and $Q = 2^{l-1}$ be the number of leaf-nodes, so that $N = MF \times Q$. Let each memory node be identified by B_i, each cell is identified by the address $A_{i,j}$, where $0 \leq j \leq MF - 1$ (address within the node) and $0 \leq i \leq Q - 1$ (node address).

In this binary TRAM the nodes are laid out using the well-known H-tree layout [10]. As organized in figure 1(b), the leaf nodes or memory nodes are in groups of four, forming an H-tree, which are further connected hierarchically. Each nonleaf node in the binary TRAM is a switch node, which is a simple 1-out-of-2 decoder with buffers. The memory nodes are memory modules which have the traditional four quadrant organization with independent control units. Thus a binary TRAM chip with depth equal to 1 is equivalent to a conventional memory chip [2]. Furthermore, when the depth is larger than 1 the address/data/control bus is connected to the root-node which in this case is a simple switch node. It decodes the most significant part

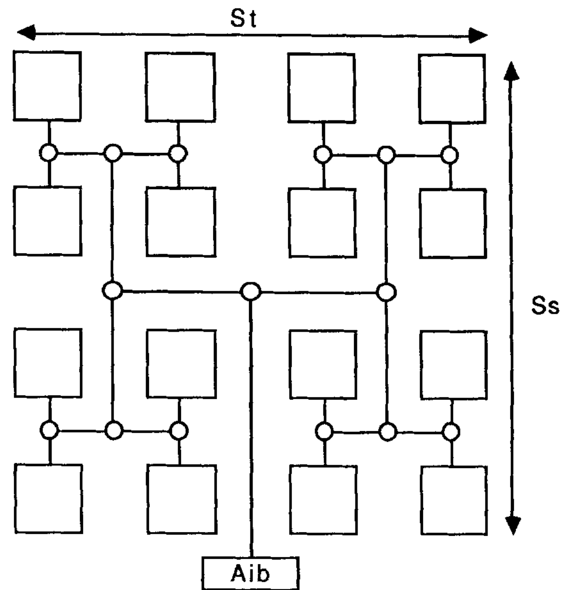


Fig. 1b. Area model for the redundant binary TRAM architecture.

of the address and generates a left- or a right-subtree select. The other signals are buffered and propagated down the tree. This action occurs repeatedly at each level until a single memory node is selected. The remaining address bits are then used to select a cell within the node.

The device has two modes of operation: a normal mode, in which it functions as a traditional RAM, and a test mode, in which the tester verifies the presence of faults through a testing procedure. Additional devices are introduced to support fault detection. Between every pair of nodes B_i and B_{i+1} ($O \leq i \leq Q - 2$), a comparator C_i (an EX-OR gate) is placed such that C_i compares the data between nodes B_i and B_{i+1}. The output of each comparator is an input to a distributed NOR gate, whose output is brought out as a—FAIL line as shown in figure 1(c). The structure of the comparators and the NOR gate has been referred to as built-in-test structure (BITS).

The TRAM architecture, as specified by its designers [10], has potential advantages over the conventional architecture, as may be briefly summarized here:

- *Easily Testable.* Partially self-testing with small testing times for very large RAM's, as shown in [10]. The use of this architecture results in significant saving in testing time in comparison with conventional architecture. For memories with more than 16 leaves the saving in testing time is more than 90%. Using the algorithm proposed in [21] the TRAM can be tested in ($16Q + 8 + 36N + 24N \log N$) operations. The proposed testing procedure has a high fault coverage: all detectable stuck-at faults in the tree decoder, stuck-at faults as well as the changing the functionality of the EX-OR gates in the BIT's, and stuck-at, two-coupling, and limited three-coupling faults in the memory nodes can be detected.
- *Low overhead.* The additional area required for large RAM's, without redundancy, is typically 8–20% more than for conventional memory architecture.
- *Improved Performance.* For large RAM's, this architecture is faster in comparison with conventional architecture, with a potential reduction in access time of about 30%. Refreshing the nodes in parallel will substantially reduce the amount of time the RAM is not available.
- *Partionable.* Partitionability makes it possible to generate partially good products. This can improve the effective yield.
- *Increased Reliability.* Single node failures are easily tolerated. If there is redundancy, the built-in testing capabilities can help to significantly reduce the mean time to repair; otherwise graceful degradation is straightforward.

3. Fault-Tolerant Characteristics

The binary TRAM architecture admits the introduction of both "static" and "operational" fault-tolerant strategies that can reconfigure the chip without loss of memory capacity, and mechanisms that, in the presence of faults can reconfigure the chip to obtain partially good chips [6]. While, in the first case (stand-by strategy) it is necessary to use spares to substitute faulty components, in the second (graceful degradation strategy), the memory remapping can be performed either internally to the chip or externally by specialized hardware (e.g., by memory management unit) or software (with some performance penalty).

In this article stand-by strategy is used for yield enhancement. Moreover, it is hypothesized that redundancies are present only at the leaf level. There are no redundancies at the interconnection bus or at switching node level since both kinds of components have a higher yield than memory and other logic support circuits (indeed, the interconnects require fewer mask layers, while the second ones are very simple logic circuits). In particular it is assumed that replacement of faulty components by redundant ones (word or bit lines) in the leaf-

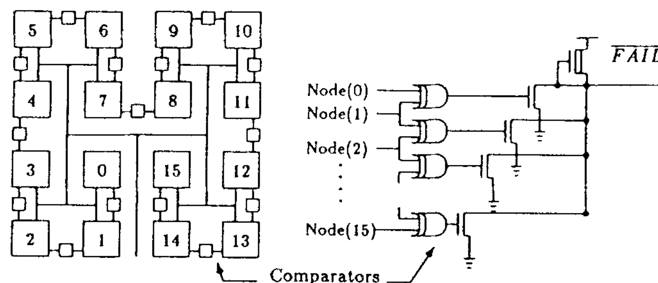


Fig. 1c. Test setup.

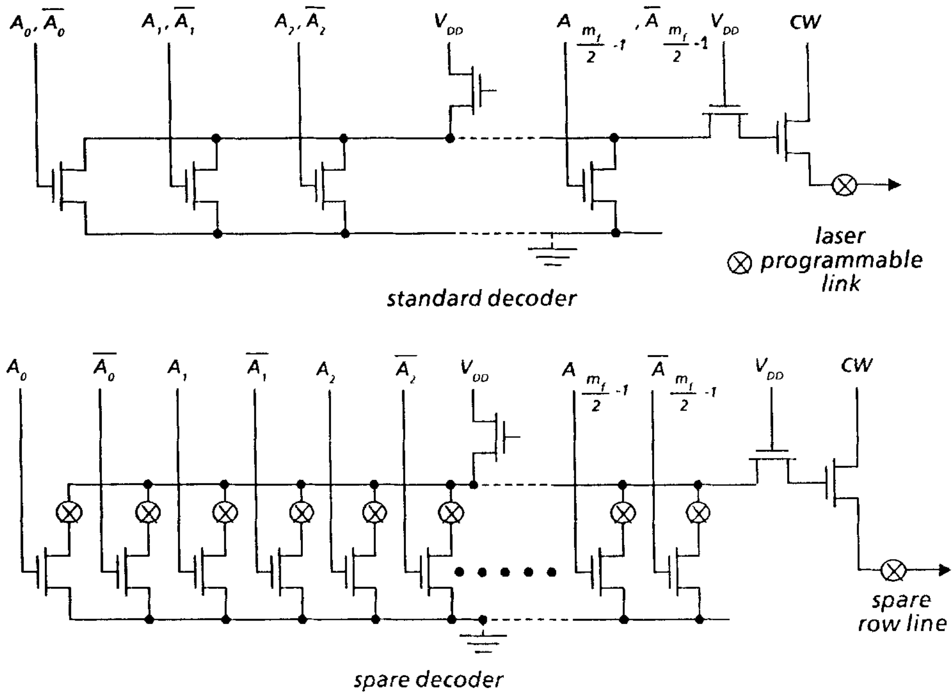


Fig. 2. Row-column replacement schema.

Table 1. Manufacturing fault type used in the leaf yield model.

1. Single cell	SC
2. Double cell on a word line	DCWL
3. Double cell on a bit line	DCBL
4. Single word line	SWL
5. Double word line	DWL
6. Single bit line	SBL
7. Double bit line	DBL
8. Leaf kill	LK

node is performed by use of fuse elements, as employed for the conventional architecture by AT&T Bell Labs, Hitachi, Toshiba, and Mitsubishi [20]. According to this technique, leaf static reconfiguration (i.e. elimination of faulty lines and replacement by spare ones) takes place by the firing of fuse elements (see Figure 2).

Based on the presence or absence of redundancies (eventually due to be exhausted) the reconfiguration strategy is organized in the following manner.

If a fault hits a component with available spares, then the reconfiguration strategy decisions are a function of the nature of detected fault and the redundancy level (see Section 5.2); otherwise, if a fault hits a component without spares then the reconfiguration strategy will work as follows:

- if the fault hits a line of the interconnection bus (address/control/data), or a switching node or the input buffers, then the entire memory chip is lost;
- if the fault hits a node-leaf and redundancies are exhausted then the leaf disconnection from the structure will be performed.

To implement the above mentioned reconfiguration strategies it is assumed to use the testing procedure adopted in [10].

4. Area-Cost Analysis

We shall base ourselves on the approach used in [9] and [10]; we need to repeat the analysis in order to take into account the presence of redundant rows and columns. Moreover, to simplify the modelling, we assume that the node leaf, as well as the TRAM chip, is organized in a square architecture.

In this article we are presenting parametric studies of chip's yield versus redundancy level and tree depth for different memory size. For this purpose we assume that the average number of manufacturing faults is proportional to the chip area circuits (see Section 5.2). For this reason we give the geometric values directly

Table 2. Leaf node circuits and their manufacturing fault densities for each type of fault (the density values are given in number-of-faults/cm²).

	SC	DCWL	DCBL	SWL	DWL	SBL	DBL	LK
Array cells	.482	.035	.035					
Word line				1.175	.31			
Bit line						.85	.19	
Standard column decoder						1	.5	
Spare column decoder						1	.75	
Standard row decoder				2	1			
Spare row decoder				2	1			
Sense amplifier						2	1.25	
Address buffer								.75
Data buffer								.75
Timing and control								.25

in μm and in μm^2 instead of in λ (technology minimum feature size) as proposed in [10]. The drawback of our approach is that for large memory size the die size might be physically unrealistic; on the contrary one has to hypothesize a change of technology (and then λ value) for each memory size, but in this case the drawback is the difficulty of determining a reasonable law which would relate the average number of manufacturing faults to the technology and consequently, to the memory size.

4.1. Leaf Node Geometric Area

The leaf node is based on traditional four quadrant RAM architecture. Besides, we need to take into account the presence of row/column redundancies. The circuit types present in the leaf node are listed in table 2.

To take into account the presence of fuses with the associated circuitry [23], from the point of view of the chip geometric area, we have to consider a cell array with standard row decoders with disable fuses, spare row decoders, standard column decoders with disable fuses, spare column decoders, sense amplifiers, address/data buffers, timing and control unit (see figures 2 and 3).

Supposing the presence of r redundant rows and columns, for a node leaf with 2^{mf} addressable bits, the array is composed of $(2^{mf/2} + r)^2$ cells (for the sake of simplicity we shall assume mf is always even); each cell has an area of C_a (expressed in μm^2, see table 4). The standard row/column decoder with disable fuses and spare row/column decoder are characterized by a width per bit to be decoded as K_r, K_c, K_r^s, K_c^s (expressed in μm) respectively. The sense amplifier height is characterized by a constant, K_s. The width is con-

Table 3. Interconnection structure circuits and their manufacturing fault densities (the density values are given in number-of-faults/cm²).

Circuit Type	Fault Density
Interconnection bus	0.25
Switching node	1.5
Input buffers	0.5

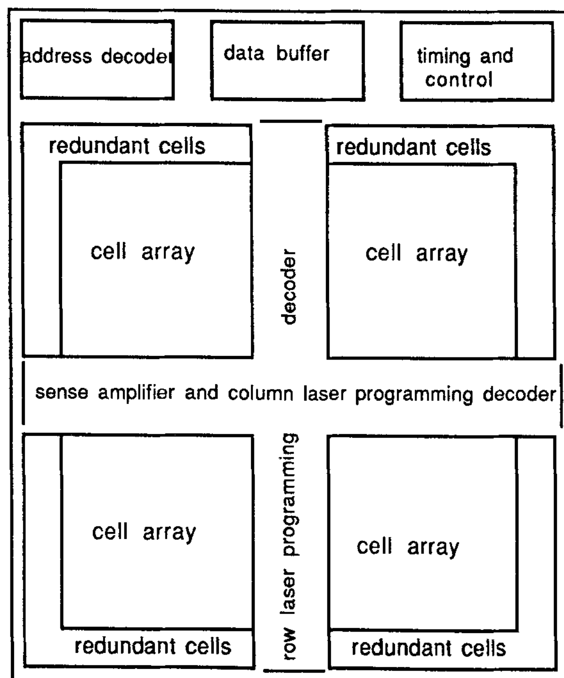


Fig. 3. Area model for a leaf-node of the redundant binary TRAM architecture.

trolled by the spacing and size of the cell. The address/data buffers and timing and control area are given by K_a (per each bit to buffer), K_d, and K_t, (expressed in μm^2) respectively. Moreover we hypothesize a location

Table 4. Design parameters.

Cell area	(C_a)	35 μm^2
Standard row and column decoder pitch/bit	(K_r, K_c)	25 μm
Spare row and column decoder pitch/bit	(K_r^s, K_c^s)	25 μm
Pitch of metal in TRAM bus structure	(K_b)	10 μm
Depth of sense amp/bit	(K_s)	100 μm
Area per each bit address latch	(K_a)	4000 μm^2
Data latch area	(K_d)	4000 μm^2
Timing and control area	(K_t)	100000 μm^2

for the standard decoder parallel to the spare decoders, since the spare decoders have twice the input than the standard ones (see figure 2), the area of a leaf-node can then be computed as:

$$A_{\text{leaf}} = [(2^{mf/2} + r) \cdot C_a^{1/2}) + K_r^s \cdot 2 \cdot mf/2] \times [(2^{mf/2} + r) \cdot C_a^{1/2}) + K_c^s \cdot mf/2 + K_s] + K_a \cdot mf/2 + K_d + K_t \qquad (1)$$

4.2. *Redundant Binary TRAM Geometric Area*

In [10] the assumptions made with regard to the bus structure implementation of the binary TRAM architecture are that the address bus is multiplexed. The lower address bits can be multiplexed; the upper bits propagate directly for subtree and leaf select. Therefore the bus carries $n - mf/2$ address lines, one data line, two lines for *Test* and *Fail* and one for *Read/-Write* signal. The comparators C_i present between every pair of nodes (see Section 2), being simple EX-OR gates, are not taken into account in the geometric area evaluation because, given their location, they are masked by the spacing and size of the tree interconnection bus. The geometric area of the redundant binary TRAM architecture can be computed (see figure 1) by using the following parameters: let W_b be the width of the bus, S_t be the length of the top of the chip and S_s be the length of the side of the chip. Therefore:

$$W_b = (n - mf/2 + 4)K_b \qquad (2)$$

where K_b is the pitch (the distance between neighboring signals),

$$S_t = S_s = N_t(A_{\text{leaf}})^{1/2} + (N_t - 1)W_b \qquad (3)$$

where N_t are the nodes on the horizontal side (in our case, mf being always even, we have $N_t = Q^{1/2}$).

Therefore, the area of the nodes and the bus structure is equal to S_sS_t.

Finally, certain input buffers would be required to drive the tree, whose area can be estimated as:

$$A_{ib} = (n - mf/2 + 4)K_a + K_d \qquad (4)$$

The area of the binary TRAM can therefore be approximately expressed as:

$$A_{BTRAM} = S_sS_t + A_{ib} \qquad (5)$$

5. Yield Evaluation for All and Partially Good Chip

Given the modular organization of binary TRAM, manufacturing yield for all good chips is given by:

$$Y_{AG} = Y_{IS}\,(Y_{LEAF})^Q \qquad (6)$$

where:

- Y_{IS} is the yield of the interconnection,
- Y_{LEAF} is the yield of a leaf-node,

On the other hand, if one is interested to obtain partially good chips too, then the yield evaluation has to take into account the fraction of usable capacity. C.H. Stapper, A.N. McLaren and M. Dreckmann in [25] have defined the *equivalent yield* as the fraction of usable capacity:

$$Y_{EQ} = Y_{AG} + (k/n)\,Y_{PE} \qquad (7)$$

where:

- Y_{AG} denotes the yield of all good chips;
- Y_{PE} denotes the yield of partially good chip (denotable also as $Y_{k/n}$);
- k/n is the fraction of usable capacity for partially good chips.

Assuming the chip is partitioned into n independent sections and that the probability has to be determined that k of the sections out of n are fault-free (after correction with redundancies, if applicable) Stapper et al. [25] proposed to evaluate $Y_{k/n}$ as:

$$Y_{k/n} = \binom{n}{k} Y_O\, Y_{CK}\, Y_{1/ns}^k\,(1 - Y_{1/ns})^{n-k} \qquad (8)$$

where:

- $Y_{1/ns}$ is the yield of each section;
- Y_{CK} is the yield of the fault fatal to the chip (chip-kill faults);
- Y_O is the gross yield.

Formula (7) takes into account only a fraction of usable capacity and Formula (8) the presence of exactly k fault-free sections out of n.

A generalization of Formula (8) could take into consideration any combination of fault-free sections, from the k-th to the $(n - 1)$th i.e.:

$$YG_{k/n} = \sum_{j=k}^{n-1} \binom{n}{j} Y_O \, Y_{CK} \, Y_{1/ns}^{j} \, (1 - Y_{1/ns})^{n-j} \quad (9)$$

while a generalization of Formula (7), considering more usable capacity fractions, simultaneously, can be expressed by:

$$Y_{GED} = Y_{AG} + \sum_{h=(k_1,k_2,\ldots,K_n)} (h/n) \, YG_{h/n}$$
$$(\text{for } k1 > k2 > \ldots > K_n) \quad (10)$$

where the summation for $YG_{k1/n}$ has as upper limit $n - 1$, the one for $YG_{k2/n}$ has $k_1 - 1$ and so on.

For example if $n = 16$ and $(k_1/n) = 0.75$, $(k_2/n) = 0.5$, $(k_3/n) = 0.25$, then

$$Y_{GED} = Y_{AG} + 0.75 \, YG_{12/16} + 0.50 \, YG_{8/16} + 0.25 \, YG_{4/16} \quad (11)$$

We want to remark that $YG_{12/16}$ is different from $YG_{3/4}$. Indeed the first one assumes that at least 12 out of 16 sections on the chip are good, and then the number of possible combinations to take into considerations is larger than in the second case.

Equivalent expression for Y_{GED} has been derived in [10] starting from a different analysis.

5.1. The Straightforward Yield Model

The yield evaluation formula essentially consists of two terms: the random defect (fault) statistics term and the term providing the probability of chip acceptability given n defects. The generalized negative binomial statistics defect model is taken as the first term of the formula. This model, taking into account the clustering phenomena, has been proven to be one of the statistics that best fits experimental data [26].

As far as the formula's second term is concerned, in the literature there is a general lack of yield formulas featuring both accuracy and ease of use. This is because yield formula accuracy requires a knowledge of the practically infinite series of all possible chip fault patterns. On the other hand, ease of use has been achieved only by introducing gross simplifying assumptions which reduce the formula's representativeness. In particular:

1. Some yield formulas do not take into consideration that there are cases in which replacement of defective components with redundant ones could not take place because of the *connectiveness constraints* imposed by logic circuitry. i.e. given K faulty components out of N, the system could still be reconfigured using the remaining $N - K$ components, without information about their effective connections.
2. Some other yield formulas do not allow for a comparison of different reconfiguration strategies, because they fail to consider the *functional relationships* existing between basic and redundant circuits.

In [4] a yield evaluation method has been introduced, which features ease of use without sacrificing representativeness and accuracy. Ease of use and representativeness are obtained by a double aggregation technique of the chip state space which replaces the infinite space of fault patterns with a finite state space of an easily defined Markov chain. Accuracy is preserved by control of the approximation bounds.

The proposed yield formula is expressed by

$$Y = \sum_{n=0}^{n*} p(n) \cdot \text{prob}(C \mid n) \quad (12)$$

where

- $p(n)$ is the probability there are n manufacturing faults in the chip;
- $\text{prob}(C|n)$ is the probability of chip acceptability given the presence of n manufacturing faults; and
- n^* is value of n corresponding to the choiced approximation level.

Let $M(n)$ denote the system configuration when there are n random spot faults. Introduce the Markov chain $M \equiv \{M(n), n = 0, 1, \ldots\}$ with state space $S = \{0, 1, \ldots, m\}$ and transition matrix $T \equiv [t(i, j)]$. State 0 denotes the absorbing state, i.e. the set of chip's unacceptable configurations; state m denotes the fault-free configuration and state $k \neq 0$, m intermediate acceptable configurations. The transition probability $t(i, j)$ gives the probability that, leaving from state i, the chip has to be reconfigured to state j by detection of one new fault. Basing on this, $\text{Prob}(C|n)$ can now be evaluated as:

$$\text{prob}(C|n) = \text{prob (the chain transits by } n \text{ steps from } OCm \text{ to } OCj \neq 0) = \text{Prob}(M(n) \neq 0 \mid M(0) = m) \quad (13)$$

and then:

$$\text{prob}(C|n) = \sum_{j=1}^{m} t^n(m, j) \tag{14}$$

where $t^n(m, j)$ is the (m, j)-th entry of the n-step transition matrix T^n.

The calculation algorithm for the yield calculation, according to formula (12), can be schematized as follows [4]:

1. Selection of the defect statistics model, $p(n)$, e.g., generalized negative binomial Poisson.
2. Definition of the Markov chain state-diagram for the architecture under consideration.
3. Derivation of the state-to-state transition probabilities, $t(i, j)$. These are functions of the redundancy level and the reconfiguration strategy.
4. Selection of the approximation level and derivation of the truncation level n^{**};

Adequate evaluation of the $p(n)$ formula above requires, as shown in [26, 27], knowledge of:

1. The *defect types* $(1, 2, \ldots, D)$ the manufacturing process gives rise to. D denotes the number of such types.
2. The *defect densities* $(d_1, d_2, \ldots, d_D)$ for each of the defect types above. Density d_i denotes the average number of type "i" defects per unit area.
3. The *fault types* $(1, 2, \ldots, F)$ originating from defects above. F denotes the number of such types. By λ_1 $(i = 1, 2, \ldots, F)$ the average number of type "i" faults per chip will be denoted.
4. The *logical circuit type* $((1, 2, \ldots, C_i)$ for $i = 1, 2, \ldots, F)$ originating type "i" faults.
5. The *fault clustering* parameter α.

Generalized statistics have turned out to be good models of the fault distribution in very large chips with internal defect clusters [27]. According to those statistics, one may write:

$$p(n) = \frac{\Gamma(n + \alpha)}{n!\,\Gamma(\alpha)} \frac{(\lambda/\alpha)^n}{(1 + \lambda/\alpha)^{n+\alpha}} \tag{15}$$

where:

$-\Gamma(\cdot)$ is the gamma function;

$-n$ is the number of faults on the chip;

$-\lambda = \lambda_1 + \lambda_2 + \ldots + \lambda_F$ is the total average number of faults per chip;

$-\alpha$ is the fault clustering parameter $(0 \le \alpha \le \infty)$. As shown in [27] $\alpha = 0$ denotes maximum clustering and $\alpha = \infty$ minimum clustering. Typical values are between .5 and 2.5.

Evaluation of formula (15) requires knowledge of the individual λ_i's values $(i = 1, 2, \ldots, F)$.

For the evaluation of the average number of faults per chip, in [26, 27] the authors proposed to relate faults to defect densities through the *critical area matrix*, i.e.,

$$\lambda = A_c\, d, \text{ or}$$

$$\lambda_i = \sum_{j=1}^{D} A_{ij}^c\, d_j \tag{16}$$

A_c is an array of numbers representing the sensitivity of a chip to random defects. This sensitivity to defects is obtained by calculating the area of the circuit in which a defect must fall in order to cause a fault [27]. For this reason in [2] we proposed to evaluate the critical area matrix terms, relating them to circuit types $((1, 2, \ldots, C_i)$ for $i = 1, 2, \ldots, F)$ originating *type-i* faults with their *physical area* (A_k) and a *fault-sensitivity-to-defects* parameter (S_{ijk}), that is, the ratio of the number of type "i" faults to the average number of type "j" defects (in the k-th circuit type). i.e.,

$$A_{ij}^c = \sum_{k=1}^{C_i} S_{ijk}\, A_k \tag{17}$$

then

$$\lambda_i = \sum_{j=1}^{D} \sum_{k=1}^{C_i} A_k\, S_{ijk}\, d_j \tag{18}$$

If the defect densities and the fault-sensitivity-to-defects parameters are known, then it is possible use formula (18); otherwise this formula can be used to generate λ_i parametrically to the physical area of the circuits, and thus formula (18) may be rewritten as:

$$\lambda_i = \sum_{k=1}^{C_i} \sum_{j=1}^{D} A_k\, S_{ijk}\, d_j = \sum_{k=1}^{C_i} A_k\, B_{ik} \tag{19}$$

where B_{ik} is the manufacturing fault density of circuit "k" type to generate the "i" fault type.

In the following to evaluate the average number of faults we will use formula (19).

5.2. Leaf Node Yield Evaluation

Concepts above will now be applied to the yield evaluation of a leaf-node. As just said in Section 3 we shall

assume that replacement of defective components (word or bit lines) by redundant ones is performed by use of fuse elements. According to this technique the chip reconfiguration (i.e. disablement of faulty lines and replacement by spare ones) takes place by the firing of fuse elements.

Assume a leaf node dynamic RAM is given, with $M = 2^{mf/2}$ bit lines and M word lines (i.e., with $M \times M \times 1$ bit), and assume there are r spare bit lines and r spare word lines (r is also called the leaf's "redundancy level").

In figure 4 the Markov chain of possible leaf states is represented. State $s \times t$ is the leaf configuration with $M + s$ fault free word lines and $M + t$ fault free bit lines. State $r \times r$ is the completely fault-free-state (the m state in formula 14). By the time all spare components have been used, the Markov chain has reached the 0×0 state. Unacceptable leaf configurations are aggregated into a single faulty absorbing-state (the state 0 in formula 14).

State to state transition are governed by the reconfiguration strategy. Assuming that after some reconfiguration step, the system is in the state $s \times t$ (i.e. there are $(M + s) \times (M + t)$ available cells) then the testing procedure can detect one of the following fault type that causes the following leaf reconfiguration:

(a) kill-leaf → then the entire leaf becomes unacceptable.
(b) single-cell → then:
  - if $s \geq t$ the bit line on which the faulty cell resides is replaced by a spare line, otherwise the corresponding word line is replaced;
  - if there are none word or bit line spare, then the entire leaf becomes unacceptable.
(c) double-cell → being these cells residing on a common word (bit) line, but on a pair of different bit (word) lines, then:
  - if $s \geq 1$ ($t \geq 1$), then the interested word (bit) line is replaced by a spare one;
  - if $s = 0$ and $t \geq 2$ ($t = 0$, $s \geq 2$), then the pair of interested bit (word) lines is replaced by spare ones;
  - if $s = 0$ and $t < 2$ ($t = 0$, $s < 2$), then the entire leaf becomes unacceptable.
(d) single-bit-line (single-word-line) → then:
  - if $t \geq 1$ ($s \geq 1$), then the faulty line is replaced by a spare line, otherwise the entire leaf becomes unacceptable.
(e) double-bit-line (double-word-line) on a pair of bit (word) lines out of M + t (M + s) available ones → then:
  - if $t \geq 2$ ($s \geq 2$), then the faulty lines are replaced by two spare lines, otherwise the entire leaf becomes unacceptable.
(f) double-bit-line (double-word-line) on a pair of bit (word) lines of which the first on one of $M + t$ ($M + s$) available ones and the other on one of the $r - t$ ($r - s$) unavailable ones → then:
  - if $t \geq 1$ ($s \geq 1$), then the faulty line is replaced by a spare line, otherwise the entire leaf becomes unacceptable.

For example for $r > s \geq 2$ and $r \geq t \geq 2$, or $r \geq s \geq 2$ and $r > t \geq 2$, the events above (a) through (f) give rise (see figure 4) to process transition out of state $s \times t$ to either state $(s - 1) \times t$, or to state $(s - 2) \times t$, or to state $s \times (t - 1)$, or to state $s \times (t - 2)$, or to the *unacceptable configuration* (absorbing state), or finally to $s \times t$ itself. In particular the transition to the absorbing state occurs when it is detected a fault on unreconfigurable circuits (such as timing and control circuit, data and address buffers); while the transition to the same state occurs either when the detected fault is located on replaced circuits or when more than one fault is located on the same circuit.

The transition probabilities for the Markov chain in figure 4 can be computed on the basis of the chip redundancies and the reconfiguration strategy outlined in paragraphs (a) through (f) above.

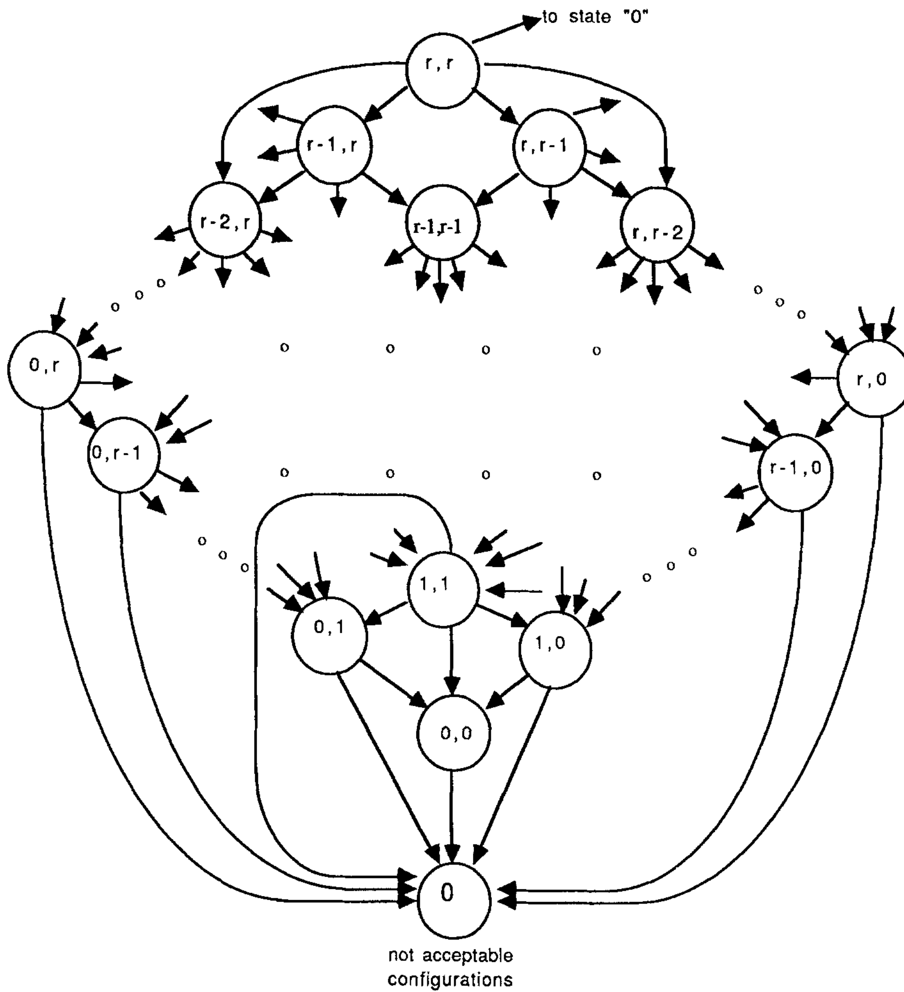
The manufacturing fault types considered for the state transition are listed in table 1 and are the same as those considered in [25]. Moreover, in table 2 are listed the possible sources and the the associated B_{ik} values for each manufacturing fault type.

5.3. *Yield Evaluation of Interconnection Structure*

As described in Section 3 we do not consider redundancies at the interconnection bus, the switching node level and the input buffers. Therefore the yield of the interconnection structure is the probability that there are no faults, that is:

$$Y_{IS} = \frac{1}{(1 + \lambda_{IS}/\alpha)^{\alpha}} \tag{20}$$

where $\lambda_{IS} = \lambda_{IB} + \lambda_{SWN} + \lambda_{AIB}$ is the total average number of manufacturing faults in the interconnection



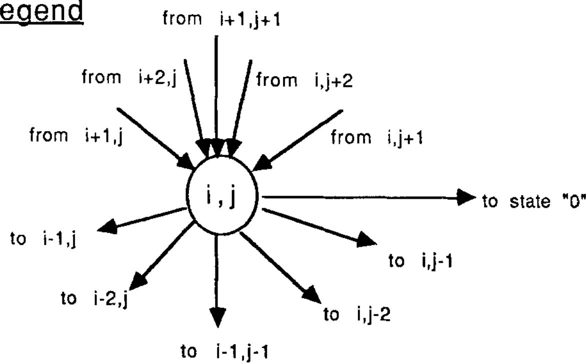


Fig. 4. Markov state transition diagram for a leaf $(M + r) \times (M + r) \times 1$ bit RAM.

structure: λ_{IB} the average of those in the interconnection bus, λ_{SWN} the average of those in the switching nodes, and λ_{AIB} the average number of manufacturing faults in the input buffers.

For this kind of manufacturing fault there is only one possible source, the bus line, the 1-out-of-2 decoder with buffers and the buffers respectively. So we deduce that:

$$\lambda_{SWN} = N_{SWN} A_{SWN} B_1 \quad (21)$$

$$\lambda_{IB} = A_{BUS} B_2 \quad (22)$$

$$\lambda_{AIB} = A_{ib} B_3 \quad (23)$$

where, A_{ib} is the area of the input buffers, N_{SWN} is the number of switch-nodes, equal to $2^{n-mf} - 1$; A_{SWN} is the physical area of a switch node and A_{BUS} is the physical area of the bus, equal to:

$$A_{BUS} = (n - mf/2 + 4) = K_b \times line_bus_length \quad (24)$$

where *line_bus_length* can be expressed as:

$$line_bus_length = \frac{A_{\text{leaf}}^{1/2}}{2} \left(2^{n-mf-1} + 2^{(n-mf-2)/2} + 6 \sum_{i=1}^{(n-mf-2)/2} 2^{n-mf-2-i}\right) \quad (25)$$

6. Numerical Evaluation

As said in introduction for any given tree-depth the problem has to be dealt with of the choice of the redundancy level which gives the best benefit/cost ratio. A convenient way of performing benefit/cost analysis it by introducing the figure of merit F, first used in [17], defined by:

$$F = A/Y_{AG} \quad (26)$$

where A is the chip area and Y_{AG} is the yield of all good chip. Figure F gives the average wafer area needed to produce an acceptable chip (indeed $1/Y_{AG}$ is the average number of chips one must produce to have an acceptable one).[1]

In the following, yield values are obtained with an approximation of 10^{-6}, and the F values in the tables are in cm^2. Moreover the graphs are obtained using as area parameters those listed in table 4, as leaf-node manufacturing fault density those listed in table 2, as interconnection structure manufacturing faults those listed in table 3.

Figures 5, 6, and 7 show the wafer area investment (F) versus the redundancy level for a binary TRAM with 1, 4, 16 and 64 leaves. Figures 5 and 6 are related to a 4 Mega bit memory, with a fault clustering parameter equal to 1 and 5, respectively, while figure 7 is related to a 1 Mega bit memory with a fault clustering parameter equal to 1.

From figures 5 and 6 it is possible to see that the clustering facter α can be neglected in the determination of the optimal redundancy level. Moreover it is possible to see that from the yield point of view there is no benefit in using memory with more leaves, this is true for each given memory dimension (see for example figure 7). This is due to the fact whatever the number of faults in the leaf, it is always possible to find a redundancy level that can tolerate it, indeed the area cost to increase the redundancy level is lower than to increase the tree-depth, then there is no benefit in dividing the leaves dimension. But as shown in [10] the more the leaves are, the lesser is the access-time, the refresh-time and the testing-time; besides increasing the number of spares increase the access-time, and then, if one

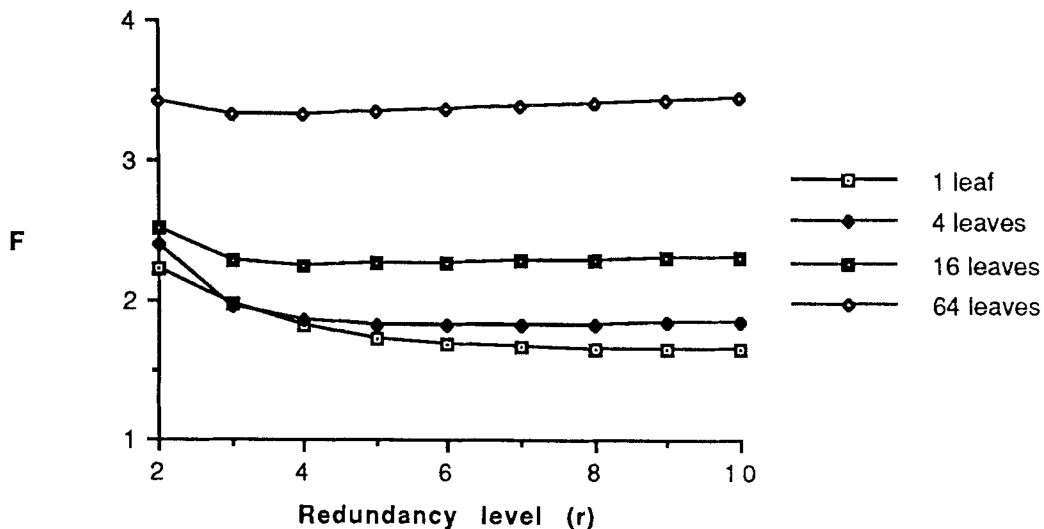


Fig. 5. Wafer area investment for a 4 Mega bit memory chip (with clustering parameter equal to 1) versus redundancy level.

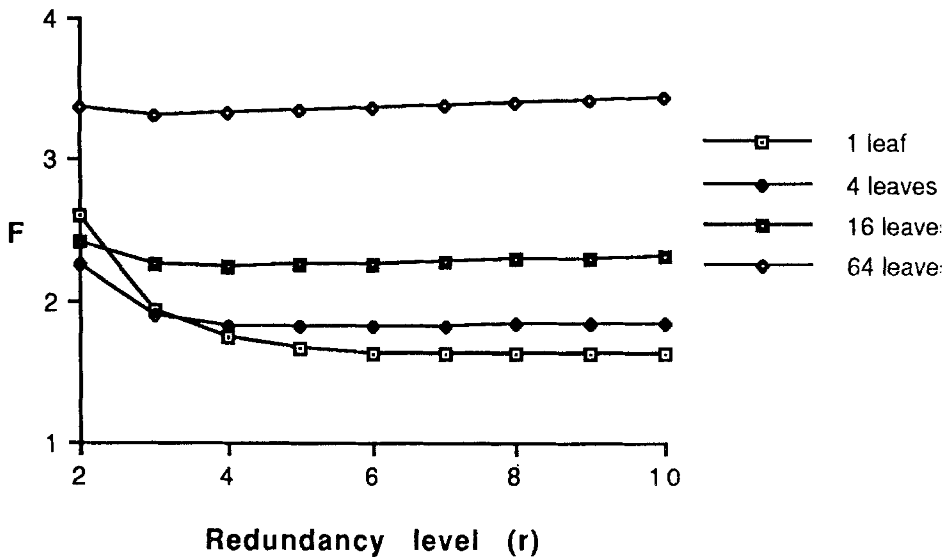


Fig. 6. Wafer area investment for a 4 Mega bit memory chip (with clustering parameter equal to 5) versus redundancy level.

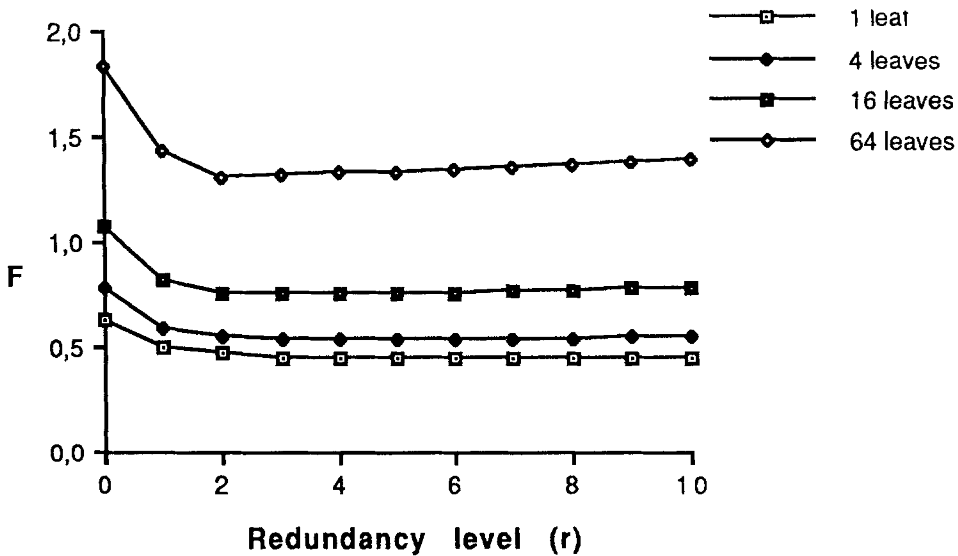


Fig. 7. Wafer area investment for a 1 Mega bit memory chip (with clustering parameter equal to 1) versus redundancy level.

wants a given performance level he cannot increase freely the leaf dimension and consequently the redundancy level.

Finally tables 5 and 6 show the sensitivity of the generalized yield (for all good chips and for .75 and .50 partially good chips) versus the reduncancy level for two binary TRAM of 4Mega bit with 4 and 16 leaves, respectively. The values are obtained using the same parameters used to obtain figure 5. From these tables we can extrapolate, without considering the gross yield phenomena, there is a benefit to use the partially good chips to increase the manufacturing chip productivity only when the yield for all good chip is very low (i.e. when there are no reconfiguration capabilities or when the level of redundancy is very far away from the optimal value). This last result is in according to the result found in [10] where the equivalent yield is evaluated without considering the redundancy presence.

7. Conclusions

The yield sensitivity of the binary TRAM structure to the variation of the tree-depth and the redundancy level has been investigated.

Table 5. Generalized yield for a 4 Mega-bit memory with 4 leaves.

Redundancy-level	Y	$Y_{GED}(.75)$	$Y_{GED}(.50)$
0	.1764	.4628	.6179
1	.6113	.8490	.8799
2	.7526	.9157	.9276
3	.9248	.9763	.9773
4	.9735	.9898	.9899
5	.9874	.9934	.9934
6	.9913	.9944	.9944
7	.9928	.9947	.9947
8	.9929	.9948	.9948

Table 6. Generalized yield for a 4 Mega-bit memory with 16 leaves.

Redundancy-level	Y	$Y_{GED}(.75)$	$Y_{GED}(.50)$
0	.1228	.7414	.7596
1	.6615	.9026	.9026
2	.8695	.9549	.9549
3	.9584	.9768	.9768
4	.9723	.9803	.9803
5	.9736	.9806	.9806
6	.9738	.9807	.9807
7	.9739	.9808	.9808
8	.9740	.9809	.9809

To this purpose a new yield model was used, which overcomes main drawbacks of the existing ones, is very easy to use and very precise; and a generalization of the equivalent yield concept used in [25] was introduced, in order to evaluate the probability of generating partially good chips.

From the result's analysis we found that, from the point of view of area-cost, the lesser the tree-depth the better the area/yield factor, since whatever the number of spot manufacturing faults in the memory leaf is, it is always possible to find a redundancy level that can tolerate it. But an increase in the leaf dimension increases the access time, the refresh time and the testing time. The increase of the access time is not only due to the increase of the leaf-size but also to the necessity of using additional redundancies to tolerate additional faults due to the larger dimension of the leaf.

Finally we found that, without considering the gross yield phenomena, one can use the partially good chips to increase the manufacturing chip productivity only when the yield for all good chip is very low (i.e., when there are no reconfiguration capabilities or the redundancy level is very far away from the optimal value).

Note

1. This figure of merit is roughly inversely proportional to the figure of merit more commonly used by the industry and called *circuit productivity* and defined as the fraction of usable chips per wafer [28]. The reason why we use F is its independency of knowledge about wafers' actual dimensions and about wafer surface utilization by chips shape and dimensions.

References

1. R. Abbott, K. Kokkonen, R.I. Kung and R.J. Smith, "Equipping a Line of Memories with Spare Cells," *Electronics*, pp. 127–130, July 28, 1981.
2. B. Ciciani and G. Iazeolla, "A Straightforward Yield Model for Fault-Tolerant VLSI Memory Chips," IFIP TC-10 Conference on "Design Methodology in VLSI and computer architecture," Sept. 1988, Pisa, Italy.
3. B. Ciciani and G. Iazeolla, "A Yield Model for the Evaluation of Topologically Constrained Chip Architectures," *IEEE Int. Conf. on Computer Design* (ICCD), Cambridge, MA, Oct. 1989.
4. B. Ciciani and G. Iazeolla, "A Markov-chain Based Yield Model for VLSI Fault Tolerant Chips," *IEEE Transactions on Computer-Aided Design*, 10 (2): 252–259, Feb. 1991.
5. J.A.B. Fortes and C.S. Raghavendra, "Gracefully Degradable Processor Arrays," *IEEE Trans. on Computers*, C-34 (11): 1033–1043, Nov. 1985.
6. K.E. Grosspietsch, "Schemes of Dynamic Redundancy for Fault Tolerance in Random Access Memories," *IEEE Trans. on Reliability*, 37 (3): 331–339, Aug. 1988.
7. J.D. Harden and N.R. Straden, "Architectural Yield Optimization for WSI," *IEEE Trans. on Computers*, 37 (1): 88–110, Jan. 1988.
8. E. Horowitz and A. Zorat, "The Binary Tree as an Interconnection Network: Applications to Multiprocessor Systems, and VLSI," *IEEE Trans. on Computers*, 30 (4): 247–253, Apr. 1981.
9. N.T. Jarwala and D.K. Pradhan, "Cost Analysis of on Chip Error Control Coding for Fault Tolerant Dynamic RAMs," *17th IEEE Fault Tolerant Computing Symposium*, pp. 278–283, 1987.
10. N.T. Jarwala and D.K. Pradhan, "TRAM: A Design Methodology for High-Performance, Easily Testable Multimegabit RAMs," *IEEE Trans. on Computers*, 37 (10): 1235–1250, Oct. 1988.
11. N.T. Jarwala and K.K. Pradhan, "Easily Testable High Speed Architecture for Large RAM," U.S. Patent No. 4, 833,677, May 23, 1989.
12. I. Koren and M. A. Breue, "On Area and Yield Considerations for Fault-Tolerant VLSI Processor Arrays," *IEEE Trans. on Computers*, 33 (1): 21–27, Jan. 1984.
13. I. Koren and D.K. Pradhan, "Introducing Redundancy into VLSI Designs for Yield and Performance Enhancement," *15th IEEE Fault Tolerant Computing Symposium*, pp. 330–334, 1985.
14. I. Koren and D.K. Pradhan, "Yield and Performance Enhancement Through Redundancy in VLSI and WSI Multiprocessor Systems," *Proceeding of the IEEE*, 74 (5): 699–711, May 1986.
15. I. Koren and D.K. Pradhan, "Modeling the effect of redundancy on Yield and Performance of VLSI Systems," *IEEE Trans. on Computers*, 36 (3): 344–355, March 1987.
16. W. Maly, A.J. Strojwas and S.W. Directo, "VLSI Yield Prediction and Estimation: A Unified Framework," *IEEE Transactions on Computer-Aided Design*, CAD-5 (1): 114–130, January 1986.

17. T.E. Mangir and A. Avizienis, "Fault-Tolerant Design for VLSI: Effect of Interconnect Requirements on Yield Improvement of VLSI Design," *IEEE Trans. on Computer*, 31 (7): 609–618, July 1982.
18. T.E. Mangir, "Sources of Failures and Yield Improvement for VLSI and Restructarable Interconnections for RVLSI and WSI: Part I—Sources of Failures and Yield Improvement for VLSI," *Proceedings of the IEEE*, 72 (6): 690–708, June 1984.
19. T. Mano, M. Wada, N. Ieda and M. Tanimoto, "A Redundancy Circuit for a Fault-Tolerant 256K MOS RAM," *IEEE Journal of Solid-State Circuits*, 17 (4): 726–731, August 1982.
20. W.R. Moore, "A Review of Fault-tolerant Techniques for the Enhancement of Integrated Circuit Yield," *Proceedings of the IEEE*, 74 (5): 684–698, May 1986.
21. C.A. Papachristou and N.B. Sahgal, "An Improved Method for Detecting Functional Faults in Semiconductor RAM's," *IEEE Trans. on Computers*, C-24: 110–116, Feb. 1985.
22. A.D. Singh, "A Reconfigurable Modular Fault Tolerant Binary Tree Architecture," *17th IEEE Fault Tolerant Computing Symposium*, pp. 298–304, 1987.
23. R.T. SMith, J.D. Chlipala, J.F.M. Bindels, R.G. Nelson, F.H. Fisher and T.F. Mantz, "Laser Programmable Redundancy and Yield Improvement in a 64K DRAM," *IEEE Journal of Solid-State Circuits*, SC-16 (5): 506–513, October 1981.
24. R.T. Smith, "Using a Laser Beam to Substitute Good Cells for Bad," *Electronics*, pp. 131-136, July 28, 1981.
25. C.H. Stapper, A.N. McLaren and M. Dreckmann, "Yield Model for Productivity Optimization of VLSI Memory Chips with Redundancy and Partially Good Product," *IBM J. Research Develop.*, 24 (3): 398–409, May 1980.
26. C.H. Stapper, F.M. Armstrong and K. Saji, "Integrated Circuit Yield Statistics," *IEEE Proc.*, 71 (4): 453–469, Apr. 1983.
27. C.H. Stapper, "On Yield, Fault Distributions, and Clustering of Particles," *IBM J. Research Develop.*, 30 (3): 326–338, May 1986.
28. C.H. Stapper, Private communication.
29. Chin-Long Wey, "On Yield Consideration for the Design of Redundant Programmable Logic Array," *IEEE Trans. on Computer Aided Design*, 7 (4): 528–535, April 1988.
30. Chin-Long Wey and F. Lombardi, "On the Repair of Redundant RAMs," *IEEE Trans. on Computer Aided Design*, 6 (2): 222–231, March 1987.

Bruno Ciciani is Research Associate of Computer Science at the University of Rome II. His researches include distributed computer systems, languages for parallel processing, fault-tolerant computing, and computer system performance/reliability evaluation.