An Input Vector Monitoring Concurrent BIST Architecture Based on a Precomputed Test Set

Ioannis Voyiatzis, Member, IEEE, Antonis Paschalis, Member, IEEE, Dimitris Gizopoulos, Senior Member, IEEE, Constantin Halatsis, Frosso S. Makri, and Miltiadis Hatzimihail, Student Member, IEEE

Abstract—Built-In Self-Test (BIST) techniques constitute an effective and practical approach for VLSI circuits testing. BIST schemes are typically classified into two categories: offline and online. Input vector monitoring concurrent BIST schemes are a class of online techniques that circumvent the problems appearing separately in online and in offline BIST in a very effective way. The utilization of input vector monitoring concurrent BIST techniques provides the capability to perform testing at different stages, manufacturing, periodic offline (in special test mode), and concurrent online (in normal mode of operation). The input vector monitoring concurrent BIST schemes proposed so far have targeted either exhaustive or pseudorandom testing separately. In this paper, a novel input vector monitoring concurrent BIST scheme based on a precomputed test set is presented. The proposed scheme can perform both concurrent online and offline testing; therefore, it can be equally well utilized for manufacturing and concurrent online testing in the field. The applicability of the scheme is validated with respect to the hardware overhead and the time required for completion of the test in benchmark circuits. To the best of our knowledge, the proposed scheme is the first to be presented in the open literature based on a precomputed test set that can perform both concurrent online and offline testing.

Index Terms—Online testing, offline testing, self-testing, input vector monitoring, concurrent error detection.

1 INTRODUCTION

BUILT-IN Self-Test (BIST) techniques constitute an attractive and practical solution to the difficult problem of testing VLSI circuits and systems. The advantages of BIST include the capability of performing at-speed testing, very high fault coverage, elimination of test generation effort, and less reliance on expensive external testing equipment for applying and monitoring test patterns [1].

Offline BIST schemes cannot detect temporary faults that occur very often in modern VLSI circuits since the detection of such faults requires continuous monitoring of the Circuit Under Test (CUT) outputs. Furthermore, offline BIST, which is commonly applied during manufacturing testing and periodic maintenance testing, may drive down the dependability of the system due to the fact that an error that occurs between two consecutive applications of the periodic test may affect the circuit without being perceived. Hence, online/offline BIST schemes have been proposed [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27]; some of these schemes exploit small test sets and stall the normal operation of the CUT in order to apply the test patterns and verify the CUT responses. Since these schemes do not operate during normal operation of the CUT, they are referred to as nonconcurrent online BIST schemes.

In offline and in nonconcurrent online BIST, the normal operation of the CUT is stalled in order to perform the test. Thus, if the CUT is of critical importance for the function of the circuit, the total circuit performance is degraded. To avoid this performance degradation, input vector monitoring concurrent BIST techniques have been proposed which exploit input vectors arriving at the inputs of the CUT during normal operation (concurrent online mode) [2], [3], [4], [5], [6], [7].

The block diagram of an input vector monitoring concurrent BIST scheme is presented in Fig. 1. The CUT is a combinational circuit with \( n \) inputs and \( m \) outputs.

During the concurrent online mode (\( T_m = 0 \)), the CUT inputs, denoted by \( A = A[1 : n] \), are driven by the normal input vector (V[1 : n]). A is also driven to the Active test set Generator and Comparator (AGC), where it is compared to a set of active test vectors called active test set. If it is found that A matches one of the active test vectors, we say that a hit has occurred or that the input vector (V) has performed a hit.

When a hit occurs, a hit signal is generated from AGC and driven to a Response Verifier (RV) module; RV is a sequential comparator with \( m \) inputs whose contents are examined at the end of the test. If the examined content (termed response of the test) differs from that of the expected, one can deduce that an error has been detected and, thus, the circuit is faulty. An important evaluation measure of a concurrent BIST scheme is its concurrent test
latency (CTL), defined as the number of normal input vectors that must be applied to the CUT inputs while the CUT operates normally in order to expect that the concurrent test is over.

In case, during normal operation, the CUT does not receive all of the required input vectors, the circuit may sporadically operate in offline mode in order to complete the test. During offline mode ($T_m = 1$), the inputs to the CUT are driven by the $V[1:n]$ outputs of the AGC.

The pioneering work on input vector monitoring concurrent BIST was carried out by Saluja et al. (C-BIST, [2]). The AGC of C-BIST is a single Linear Feedback Shift Register (LFSR) and a comparator and, thus, the active test set consists of exactly one active test vector (the current value of the LFSR). During online mode, the input vector is compared with the unique active test vector. If the two vectors match, the LFSR proceeds to the next state changing the active test vector and the RV is enabled.

C-BIST has low hardware overhead but very high CTL (since, in every clock cycle, the input vector is compared against only one active test vector). To drive down the CTL, four techniques have been proposed so far, namely, the Multiple Hardware Signature Analysis Technique (MHSAT, [3]), the Order Independent Signature Analysis Technique (OISAT, [4]), windowed-Comparative Concurrent BIST (w-CBIST, [5]), and RAM-based Concurrent BIST (R-CBIST [6]), which decrease the CTL by increasing the number of active test vectors.

The abovementioned techniques target either exhaustive or pseudorandom testing, so the size of the active test set is large, imposing high CTL. In order to resolve this problem, Built-In Concurrent Self-Test (BICST, [7]) was proposed by Sharma and Saluja; when BICST is applied to an $n$-input $m$-output combinational CUT that can be tested with $T$ vectors, it utilizes a $T$-line $\times (n + m)$-column PLA; when a vector that belongs to the $T$-vector test set reaches the CUT inputs, the AND plane of the PLA is activated and the response of the CUT is compared to the output of the $m$ response columns of the PLA.

BICST cannot be utilized in offline mode in order to impose the test vectors required to test the CUT during manufacturing testing or periodic offline testing. In other words, in a case where a vector that belongs to the test set does not appear as an online input, the concurrent test can never be completed. Furthermore, an offline BIST scheme is used during manufacturing and periodic offline testing; therefore, the total hardware overhead is further increased.

In this paper, a novel input vector monitoring concurrent BIST technique is presented for Monitoring Input Vectors for concurrent testing based on a preComputed test SET (MICSET); the presented scheme is based on a test set stored in a mapping logic module which can be implemented with either random logic or a ROM whose address inputs are driven by a subset of the input bits of the CUT. MICSET can operate in offline mode, imposing the required vectors to the CUT inputs, and can thus be further utilized for 1) manufacturing, 2) nonconcurrent online, and 3) periodic offline testing. The time required to generate the offline vectors depends on $t$, the number of columns of the test matrix that are row-distinct, i.e., each $t$-tuple is different from any other $t$-tuple. The existence of such columns in benchmark test sets has been calculated both analytically and experimentally. To the best of our knowledge, the proposed scheme is the first to be presented in the open literature that can operate both in concurrent online and offline modes and is based on a precomputed test set.

This paper is organized as follows: In Section 2, the operation of the proposed scheme is presented and exemplified. In Section 3, we investigate the choice of $t$ inputs among the $n$ inputs of the CUT ($t \leq n$) in such a way that they can be used to drive the test generator module during offline testing; both analytical and experimental methods are utilized. In Section 4, the CTL is evaluated. In Section 5, the proposed scheme is extended for the case where CUTs with many inputs are to be tested, thus resulting in unacceptably high CTL. In Section 6, MICSET is compared to other schemes that have been proposed for concurrent testing with respect to their application on the well-known c6288 ISCAS’85 benchmark. Finally, in Section 7, we conclude this paper.

## 2 Operation of the Scheme

### 2.1 Online Operation

The idea underlying the online operation of the proposed scheme is presented in Fig. 2. Let us consider a combinational CUT with $n$ inputs and $m$ outputs, which can be tested with $T$ test vectors. Therefore, the test set is comprised of a $T$-row $\times n$-column test matrix. Among the $n$ columns of the matrix, we identify $t$ ($\lfloor \log_2 T \rfloor \leq t < n$) columns such that all $t$-tuples in the $T$ rows are distinct. These columns will be utilized to indicate addresses in the module where the remaining $(n - t)$ columns will be stored. In Fig. 2a, we have assumed, for the sake of simplicity of presentation and without loss of generality, that the distinct columns of the test matrix correspond to signals $A[1:t]$.

T logic cells denoted by $C_i$ ($i = 1, \ldots, T$) are also formed; each logic cell takes three inputs: the output of the decoder of the mapping logic module, the output of the comparator, and one reset signal. The schematic design of the cell $C_i$ is presented in Fig. 2b. The scheme operates as follows: Initially, all logic cells are reset. The CUT operates normally;
every time an input vector that belongs to the test set reaches the CUT inputs, the \( t \) bits indicate an address in the mapping logic. If the \((n - t)\) remaining bits of the input vector match the value stored in the OR-plane of the mapping logic (this means that the input vector belongs to the regarded test set), the output of the comparator is enabled. The corresponding cell \( C_i \) receives both \( \text{cmp} \) and \( D[i] \) signals and is enabled. The logic cell is designed in a way that can automatically track the setting status, as will be explained in the next paragraph. Hence, if it was not set previously (i.e., if the test pattern had not appeared to the inputs of the CUT), then its \( \text{hit}[i] \) output is triggered and the hit signal enables the RV to capture the CUT response to the input vector.

The logic cell operates as follows (Fig. 3): Initially, the reset signal is enabled \((= 1)\) and the hit is set to 0 (Fig. 3a). When the reset returns to 0, the hit continues to stay at 0 (Fig. 3b). When a hit is issued, both \( \text{cmp} \) and \( D[i] \) are 1; therefore, the output of the left AND gate of the cell is 1 and the hit is set to 1 (Fig. 3c). When the output of the bottom NOR gate takes the value of 1, the latch is set and its output is 0; therefore, the hit returns to 0 (Fig. 3d). After a hit has occurred, the \( \text{hit}[i] \) signal stays at 0 (Fig. 3e), even if both \( \text{cmp} \) and \( D[i] \) have the value of 1 (Fig. 3f).

For response verification purposes, Accumulator-Based Compaction (ABC) of the responses is issued. ABC is a well-known response verification technique that has been extensively studied and described [8] and has been shown to have aliasing properties similar to the best compactors based on Cellular Automata and Multiple Input Signature Registers (its asymptotic aliasing probability is \( 2^{-m} \), where \( m \) is the number of stages of the accumulator).

For example, let us consider a CUT with \( n = 6 \) inputs that can be tested with the five test patterns presented in Table 1 (columns are lettered from “a” to “f” in order to simplify the discussion).

In order to apply the MICSET scheme, we need \( t \geq \log_2 5 = 3 \) columns in which no two identical \( t \)-tuples exist. It is easy to see that columns a, b, and c hold this property and can, therefore, be used to address the OR-plane, which is made up of \( T = 5 \) words, each one

<table>
<thead>
<tr>
<th>Test Vectors for the Example Benchmark</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>
consisting of \((n-t) = (6-3) = 3\) bits. In Fig. 4, we present the MICSET scheme for the considered benchmark.

The module included in the dotted lines performs the comparison of the input vector against the precomputed test vectors of the test set in Table 1. Each of the five AND gates corresponds to a test vector. Thus, when \(V[1:6] = \{0,0,0,1,0,1\}\) reaches the CUT inputs for the first time, inputs \(V[1:3] = \{0,0,0\}\) and \(D[1] = 1\). Therefore, the outputs of the first (leftmost) and third (rightmost) OR gates are “1.” Thus, the outputs of the gates are compared against the bits \(V[4:6] = \{1,0,1\}\) of the input vector and the output of the comparator is enabled, setting cell C1 to 1 and enabling hit[1] (and, consequently, the hit signal).

### 2.2 Offline Operation

In order to perform offline testing with MICSET, we work as follows: A \(T_{in}\) signal is utilized which drives the select inputs of multiplexers feeding the inputs of the CUT and the address inputs of the mapping logic. In addition, a \(t\)-stage counter is issued which drives the address inputs of

the mapping logic during the offline test mode. When \(T_{in}\) is enabled, the CUT inputs are driven by 1) the \(t\) stages of the counter and 2) the \((n-t)\) outputs of the mapping logic. In fact, \(2^t\) pattern combinations are applied to the inputs of the CUT and the RV is enabled only for the applied patterns that belong to the active test set. For example, in Fig. 5, we present the mixed online/offline scheme for the testing of the benchmark in Table 1; the module in Fig. 5 applies the \(2^t = 2^3 = 8\) vectors shown in Fig. 5b.

### 3 Selection of the \(t\) Address Inputs

In order to apply the MICSET design method, we must identify \(t\), with \(\lceil \log_2 T \rceil \leq t \leq n\) columns in the test vector matrix, such that all \(T\) \(t\)-tuples are distinct. In the following, we shall formulate the problem and present a solution based on a greedy algorithm.

**Definition 1.** We say that \(t\) columns \(c_1, c_2, \ldots, c_t\), \((\lceil \log_2 T \rceil \leq t < n)\) in a \(T \times n\) test matrix are row-distinct if the maximum number of repetitions of the same \(t\)-tuple among the \(t\) columns and \(T\) rows is 1.

In our benchmark example in Table 1, columns d, e, and f are not row-distinct since the three-tuple \((0, 1, 0)\) appears twice. In addition, columns c, d, and e are not row-distinct since the three-tuple \((1, 0, 1)\) also appears twice. Columns a, b, and c are row-distinct.

Since the \(t\) address bits must be distinct, in order to address the contents of the \((n-t)\) remaining columns, the \(t\) columns must be row-distinct. We shall proceed in two ways. First, we shall present an analytical calculation of the probability that row-distinct columns can be found in an \(n\)-column \(\times\) \(T\)-row test matrix. Then, we shall provide a simulation-based method in order to find row-distinct columns in the matrices of benchmark circuits.

![Fig. 4. MICSET module for the benchmark example.](image1)

![Fig. 5. (a) Online/offline testing for the benchmark example. (b) Applied test patterns (shaded vectors do not belong to the test set).](image2)
3.1 Probabilistic Analysis

In this section, we shall present two lemmas that are indicative of the probability of existence of row-distinct columns. The proofs of the lemmas are provided in the Appendix.

Lemma 1. The probability $P_a$ that $t$ randomly chosen columns of a $T$-row matrix are row-distinct is

$$P_a = \frac{\prod_{i=0}^{T-1} (2^t - i)}{2^{t\times T}}.$$  (1)

Lemma 1 calculates the exact probability that if we randomly select $t$ columns out of $n$, these $t$ columns will be row-distinct.

Lemma 2. The probability $P$ that, among the $n$ columns of a $T \times n$ matrix there is at least one set of $t$ row-distinct columns is lower bounded by

$$P \geq L_b = 1 - (1 - P_a)^{n/t}.  \tag{2}$$

$P_a$ is the probability that $t$ randomly chosen columns of the matrix are row-distinct, as given by Lemma 1.

Lemma 2 provides a lower bound, i.e., a “safe” calculation that the probability that we will find a set of row-distinct columns will not be less than the presented bound. In fact, the experimental results (in Section 3.2) show that this bound is overly pessimistic.

For example, let us consider the well-known c6288 benchmark with $n = 32$ inputs and $m = 32$ outputs. This module can be tested with $T = 28$ test patterns [10]; therefore, $5 \leq \log_2 T \leq n$. In Fig. 6a, we present the probability $P_a$ that $t$ randomly chosen columns are row-distinct and the lower bound ($L_b$) that at least one set of row-distinct columns exists as a function of $t$. From Fig. 6a, we can see that for $t \geq 11$, the probability that at least one set of row-distinct columns exists is lower bounded by a number very close to 1. In Figs. 6b, 6c, and 6d, we present the same data for three other ISCAS benchmarks, namely, c432, c499, and c880, for which the respective values of $\log_2 T$ are 6, 7, and 8, respectively. We can easily see that the same situation also holds for these circuits as well.

3.2 Experimental Results

Although the results of Section 3.1 provide a safe calculation, in fact it would be mostly interesting to see what the situation is for real test sets. For this reason, we have coded a greedy procedure that searches among the $t$-tuples of columns increasing $t$ from $\log_2 T$ to $n$ until a set of row-distinct columns is found, as shown in Fig. 6. The procedure is given in the sequel.

Procedure 1:

```
for $t = \lceil \log_2 T \rceil$ to $n$ do
  for all $t$-combinations of $n$ columns
    if found row-distinct set of columns
      then exit
```

In order to give an essence of the value of $t$ for real-world circuits and test sets, we performed experiments utilizing Procedure 1 for test sets extracted by the COMPACTEST tool [10]. The results are presented in Table 2. In Table 2, the first column indicates the circuit name; the second and third columns indicate the number of circuit inputs and outputs, respectively; the fourth column indicates the number of test patterns ($T$) given by COMPACTEST, and the fifth column indicates the binary logarithm of $T$. In the sixth column, the value of $t$ for each of the benchmarks is calculated. We can see that, in all cases, $t$ is extremely close to $\log_2 T$, in fact much closer than the probabilistic results given by Lemmas 1 and 2. Therefore, we can confirm that the values given by (1) and (2) are indeed overly pessimistic.
TABLE 2
Experimental Validation of $t$ for Some of the ISCAS Benchmarks

<table>
<thead>
<tr>
<th>Circuit</th>
<th>n</th>
<th>m</th>
<th>T</th>
<th>$\log_T$</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>c432</td>
<td>36</td>
<td>7</td>
<td>45</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>c499</td>
<td>41</td>
<td>32</td>
<td>51</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>c880</td>
<td>60</td>
<td>26</td>
<td>53</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>c1355</td>
<td>41</td>
<td>32</td>
<td>86</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>c1908</td>
<td>33</td>
<td>25</td>
<td>116</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>c3540</td>
<td>50</td>
<td>22</td>
<td>145</td>
<td>8</td>
<td>10</td>
</tr>
<tr>
<td>c6288</td>
<td>32</td>
<td>32</td>
<td>28</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

4 Calculation of the CTL

An important evaluation measure for the performance of MICSET is the time required to complete the test while operating only in online mode, defined as the number of normal input vectors that must be applied to the CUT inputs while the CUT operates normally in order to expect that the concurrent test is over.

To compute the CTL of MICSET, we shall assume, as in [2], [3], [4], [5], [6], [7], that the probability of occurrence of any one pattern is independent of the occurrence of any other pattern and that the circuit input patterns are equally likely to occur during normal operation of the circuit. Although it has not been proven that the above two assumptions hold in practical circuits, it is (to the best of our knowledge) the only assumption that has been introduced in the literature for the appearance probabilities of the input vectors. The computation of the CTL for MICSET can be performed either analytically in a way similar to the one provided in [5] or by performing computer simulations [6].

4.1 Analytical Calculation

Let $h(i)$ denote the probability of a hit when $i$ vectors remain to be hit and $L(i)$ denote the corresponding mean number of cycles. The probability of a hit at the beginning of a test session, that is, when no test vector has reached the CUT inputs, is $T/N$, where $N = 2^n$ is the total number of vectors. Therefore, the mean number of cycles until a hit is achieved is $L(T) = N/T$ clock cycles. Every time a hit is performed, the hit probability is reduced. When only one vector remains to be hit, the hit probability is $1/N$ (and the corresponding number of cycles is $L(1) = N$). In general, when $i$ test vectors remain to be hit, the hit probability is $i/N$. The corresponding mean number of cycles $L(i)$ required for a hit is $L(i) = 1/h(i) = N/i$. Thus, the mean number of cycles required in order to receive the entire test set or, equivalently, the CTL is given by

$$CTL = \sum_{i=1}^{T} L(i) = N \times \sum_{i=1}^{T} \frac{1}{i}. \quad (3)$$

In order to give an essence of the CTL of MICSET for real-world circuits, in Table 3 we present the CTL calculated utilizing (3) for some of the ISCAS ’85 benchmarks. In Table 3, in the first column, we present the circuit name; in the second column, we present the calculated CTL in clock cycles.

From Table 3, it is derived that, for circuits having more than 40 inputs, i.e., c499, c880, c1355, and c3540, the value of the CTL becomes unrealistically high. In fact, such circuits are typically tested utilizing different techniques, like partitioning and pseudoexhaustive testing schemes. For circuits with fewer inputs, i.e., c432, c1908, and c6288, the application of MICSET may result in acceptable values of the CTL.

4.2 Simulation Experiments

In order to evaluate the CTL, a C program that simulates the operation of MICSET was developed. The program takes as inputs the number of inputs of the CUT ($n$); the test set ($T$) generates pseudorandom test vectors and compares them against the precomputed test set.

```c
int CTL(int n, int Test[T])
{
    N = 2^n; hits = 0; Tries = 0;
    for (i = 0; i < T; i++) L[i] = 0;
    while (hits < T)
    {
        V = rand(0, N - 1); Tries++;
        for (i = 0; i < T; i++)
            if ((V == Test[i]) && (L[i] == 0))
                {L[i] = 1; hits++;}
    }
    return(Tries);
}
```

The program takes as inputs the test set stored in an array Test[] and operates as follows. Initially, all latches of the AGC are set to zero ($L[i] = 0$). In every clock cycle, a new vector $V$ is chosen. If the incoming vector belongs to the test set and has not appeared before ($Test[i] == V$) & ($L[i] == 0$)), the respective latch is set. The procedure is repeated until all latches are set.

In the third column of Table 3, we present the simulation results for the benchmark circuits. By comparing the second and third columns of Table 3, we can conclude that the results match efficiently well.

It should be noted that, if the equal-probability assumption does not hold, i.e., if the input vectors are not equiprobable, and if the distribution of the input vectors is known, and some vectors are more frequent than others, then the test set can be selected in such a way that the vectors that are more probable to occur are selected and prestored.
5 Extension of MICSET for Instant Manifestation of Errors

In case the CUT has many inputs, it would be unacceptable to utilize an RV for the verification of the CUT outputs, since the time required to complete the concurrent test would be excessive. In such cases, a solution based on the comparison of the CUT outputs with the respective prestored expected responses should be adopted. Every time that an input vector belongs to the test set, the output of the CUT is compared against a prestored output response and, if the two responses differ, an error signal is generated. Therefore, an error is immediately revealed. For example, for the case of the C17 benchmark, a test set along with the expected responses is given as follows:

<table>
<thead>
<tr>
<th>Input vectors</th>
<th>Expected responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0101) (0101) (1001) (1111)</td>
<td>(01) (11) (00) (10)</td>
</tr>
</tbody>
</table>

In this case, the MICSET scheme can be extended as shown in Fig. 7.

It should be noted that, when the output comparison scheme exemplified in Fig. 7 is employed, the number of stored data increases. In order to drive down this overhead, output compaction schemes (e.g., [11], [12], [13], [14]) can be utilized to drive down the number of output bits.

6 Comparisons

In this section, MICSET will be compared against the BICST scheme proposed in [7] and the well-known Duplication with Comparison (DWC) scheme [28] with respect to its application for concurrent testing of the c6288 benchmark. This benchmark was chosen since, as shown in Table 3, the application of MICSET on this benchmark results in reasonable CTL (in a 200 MHz clock circuit, the CTL would be \(\approx 85\) seconds).

The hardware overhead is calculated using the gate equivalents as a metric. One gate equivalent or gate is the hardware equivalent of a two-input NAND gate. In Table 4, the hardware overhead of the various modules of MICSET has been calculated, following Fig. 5. For the calculations, in Table 4, we have considered that an \(n\)-input AND/OR gate is equivalent to \(n\) two-input NAND gates.


In the sequel, we shall study the applicability of 1) the proposed scheme, 2) the BICST scheme, and 3) Duplication with Comparison (DWC) [28] for the concurrent BIST of the c6288 ISCAS ’85 benchmark comprised of 2,416 gates. The DWC solution requires the duplication of the module, the
implementation of multiplexers at the CUT inputs, the test generator, and the comparator of the responses.

In Table 7, we compare the MICSET, the BICST, and the DWC schemes for the considered benchmark. In Table 7, in the first column, we present the compared scheme; in the second and third columns, we present the Test Pattern Generator (TPG) and RV method utilized (MICSET utilizes the same TPG for both RV implementations). In the fourth column, we present the hardware overhead in gate equivalents and, in the fifth column, the percentage hardware overhead of the scheme compared to the hardware overhead of the c6288 module. In the sixth column, we present the number of test patterns applied during the offline mode during manufacturing test and periodic offline test. In the seventh and eighth columns, the latency in seconds is presented, where a 200 MHz clock has been assumed; in the seventh column, the latency of fault detection is presented, i.e., the time required for a fault to be acknowledged, while, in the eighth column, the Latency of the Concurrent test is shown, i.e., the time required in order to assure that all of the tests of the test set have appeared to the CUT inputs and the CUT is free of the considered errors. In the ninth column, we present the faults that are assured to be detected. Finally, in the ninth column, we present the faults that escape detection during normal operation.

From Table 7, the following conclusions can be drawn:

- With respect to the Latency of the Concurrent test, the DWC scheme presents unacceptably high value, i.e., $2^{64}$ cycles = $2^{24}$ seconds = 5 million hours.
- With respect to the detected faults, the DWC scheme leaves no fault undetected during normal operation (ninth column). On the other hand, BICST and MICSET can, at the end of the concurrent test, guarantee that all modeled faults (i.e., the faults detected by the considered test set) do not exist.

In conclusion, which of the candidate solutions will be chosen depends on the specifications of the design. If a low-cost solution is targeted, MICSET with ABC may be utilized. On the other end, if instant manifestation of errors as well as a small number of offline test patterns is required and the hardware overhead of 177 percent can be paid, then DWC can be chosen. In-between schemes may be chosen to satisfy intermediate needs.

In order to investigate the hardware overhead in synthesized designs, we synthesized the compared schemes using a $35\mu m$ library and commercial synthesis and simulation tools. The operating frequency of the synthesized schemes scaled up to 200 MHz. The comparative areas occupied by each scheme were analogous to the ones shown in Table 7. For example, the area of MICSET (ACC) scheme was $\approx 150.950 \mu m^2$, while the area occupied by the DWC(LFSR) scheme was $\approx 220.000 \mu m^2$, i.e., $\approx 45$ percent higher. From Table 7, the gate count of the DWC(LFSR) scheme is 2,768, i.e., $\approx 45$ percent more than the gate count of the MICSET (ACC) scheme, which is 1,904 gates.

### 7 Conclusions

Input vector monitoring concurrent BIST schemes exploit vectors appearing during normal operation of the CUT for BIST purposes, thus avoiding performance degradation of the circuit’s normal operation. In this paper, a novel scheme for MICSET has been presented. MICSET can be utilized for both concurrent online and offline testing, therefore eliminating the necessity for a separate offline BIST mechanism. To the best of our knowledge, MICSET is the first scheme presented in the open literature that can perform both concurrent online and offline testing based on a precomputed test set. The applicability and effectiveness
of the proposed scheme has been investigated in comparison to the previously published schemes; the proposed scheme compares favorably both with BICST [7] and the well-known “DWC” scheme with respect to the trade-off of the hardware overhead and the number of tests applied during offline testing.

It should be noted that, since the hardware overhead of the proposed scheme depends on the number of test patterns in the test set, a reduction in the number of test patterns would have a consequent reduction on the MICSET overhead. Since current CAD tools tend to produce more efficient and compact test sets (comprised of fewer test patterns), the hardware overhead is expected to decrease. This direction is currently being investigated. Another direction toward reducing the hardware overhead of the scheme would be to utilize the same MICSET module in order to test more than one CUT, one after the other, in a cyclic manner. This direction is also the subject of ongoing research.

APPENDIX

PROOFS OF LEMMAS 1 AND 2

Lemma 1. The probability $P_a$ that $t$ randomly chosen columns of a $t$-row matrix are row-distinct is

$$P_a = \frac{2^t - t}{2^{t^2}}.$$  \hspace{1cm} (1)

Proof. Let $A$ denote the event that a binary matrix with $t$ rows and $t$ columns has no two identical rows. There are $2^t - t$ different ways for the matrix to appear. If the $t$ rows are to be distinct, then the first row can appear in $2^t$ different ways, the second row can appear in $2^t - 1$ different ways, the third in $2^t - 2$ ways, and so on. Thus, there are $2^t(2^t - 1)(2^t - 2) \cdots (2^t - t) \cdots (2^t - T + 1)$ ways; the $T$ rows can be distinct. Accordingly, the desired probability is

$$P_a = P(A) = \frac{2^t(2^t - 1)(2^t - 2) \cdots (2^t - T + 1)}{2^{t^2}}.$$ \hspace{1cm} \Box

Lemma 2. The probability $P$ that among the $n$ columns of a $T \times n$ matrix there is at least one set of $t$ row-distinct columns is lower bounded by

$$P \geq L_b = 1 - (1 - P_A)^{n/t}.$$  \hspace{1cm} (2)

Proof. First, we note that there are $\binom{n}{t}$ possible choices of $t$ out of $n$ columns which are equally likely to be selected. Let $A_i$ denote the event that at the $i$th choice, the $t$ selected columns out of $n$ are row-distinct, $i = 1, \ldots, \binom{n}{t}$. Obviously, the events $A_i$, $i = 1, \ldots, \binom{n}{t}$, are not mutually independent.

Then, the desired probability $P$ is given by

$$P = P\left(\bigcup_{i=1}^{\binom{n}{t}} A_i\right) = P\left(\left(\bigcap_{i=1}^{\binom{n}{t}} A_i^c\right)^c\right) = 1 - P\left(\bigcap_{i=1}^{\binom{n}{t}} A_i^c\right),$$

where $A_i^c$ denotes the complement of $A_i$. Let $B_i$ be the event that the $i$th, $(j+1)$th, $\ldots$, $(j+t-1)$th columns of the $T \times n$ matrix are row-distinct. Then, noting that $\{B_{it+1}, i = 0, 1, \ldots, [n/T] - 1\} \subseteq \{A_i, i = 1, \ldots, \binom{n}{t}\}$, $P(B_{it+1}) = P_{it}, i = 0, 1, \ldots, [n/T] - 1,$ and that the events $B_{it+1}$ are independent, we get

$$P\left(\left(\bigcap_{i=1}^{\binom{n}{t}} A_i^c\right)^c\right) = 1 - P\left(\bigcap_{i=0}^{[n/T] - 1} P(B_{it+1})\right) = \prod_{i=0}^{[n/T] - 1} (1 - P_{it}) = (1 - P_a)^{\binom{n}{t}}.$$ \hspace{1cm} \Box

The result follows.

In order to illustrate the proof of Lemma 2, let $n = 4$ and $T = 4$; hence, $t = 2$. Lemma 2 says that if we enumerate the columns by $1, 2, 3, 4$, there are six possible choices for the columns, namely, $(1, 2), (1, 3), (1, 4), (2, 3), (2, 4)$, and $(3, 4)$. Let $A_1$ be the event that columns $(1, 2)$ are row-distinct, $A_2$ be the event that columns $(1, 3)$ are row-distinct, and so on; $A_6$ is the event that columns $(3, 4)$ are row-distinct.

The events $B_i$ are defined as follows: $B_1$ is the event that columns $(1, 2)$ are row-distinct; $B_2$ is the event that columns $(2, 3)$ are row-distinct; $B_3$ is the event that columns $(3, 4)$ are row-distinct. There are $(4/2) = 2$ $B_{it+1}$ events, for $i = 0, 1$; more precisely, $B_{it+1}$ is the event that columns $(1, 2)$ are row-distinct and $B_{it+1} = B_3$ is the event that columns $(3, 4)$ are row-distinct.

It holds that $P(B_1) = P(B_2) = P_{it}$ hence, $P(B_1^c) = P(B_2^c) = 1 - P_a$. Since $B_1$ and $B_3$ are mutually independent, $P(B_1^c) \times P(B_2^c) = (1 - P_a)^2$.

Then,

$$P = P(A_1 \cup A_2 \cup A_3 \cup A_4 \cup A_5 \cup A_6) = P\left((A_1^c \cap A_2^c \cap A_3^c \cap A_4^c \cap A_5^c \cap A_6^c)^c\right) = 1 - P(A_1^c \cap A_2^c \cap A_3^c \cap A_4^c \cap A_5^c \cap A_6^c).$$

However, $\{B_1^c, B_3^c\} \subseteq \{A_1^c, A_2^c, A_3^c, A_4^c, A_5^c, A_6^c\}$, hence,

$$P(A_1^c \cap A_2^c \cap A_3^c \cap A_4^c \cap A_5^c \cap A_6^c) \leq P(B_1^c \cap B_3^c) = P(B_1^c) \times P(B_3^c) = (1 - P_a)^2.$$

Therefore,

$$P = 1 - P(A_1^c \cap A_2^c \cap A_3^c \cap A_4^c \cap A_5^c \cap A_6^c) \geq 1 - (1 - P_a)^2.$$ \hspace{1cm} \Box

ACKNOWLEDGMENTS

A preliminary version of this work was presented at the IEEE International Test Conference, 2005, Austin, Texas.

REFERENCES

H. Al-Asaad, B.T. Murray, and J.P. Hayes, “Online BIST for
A. Steininger and J. Vilanek, “Using Offline and Online BIST to
K.K. Saluja, R. Sharma, and C.R. Kime, “Concurrent Comparative
M. Chatterjee and D.K. Pradhan, “A BIST Pattern Generator
K. Chakrabarty, B.T. Murray, and J.P. Hayes, “Optimal Zero-
K. Chakrabarty and J.P. Hayes, “Zero-aliasing Space Compaction
K. Chakrabarty, B.T. Murray, and J.P. Hayes, “On-Line Built-In Self Test Architecture Based on a
J. Rajski and J. Tsyzer, “Test Responses Compaction in Accumulators with Rotate Carry Adders,”
F. Brglez and H. Fujiwara, “A Neutral Netlist of 10 Combinational
K. Chakrabarty, “Zero-Aliasing Space Compaction Using Multiple Parity Signatures,”
K. Chakrabarty, B.T. Murray, and J.P. Hayes, “Zero-Aliasing Space Compaction Using Linear
K. Chakrabarty, “Space Compaction of Test Responses Using Orthogonal Transmission Functions,”
H. Al-Asaad and M. Shringi, “On-Line Built-In Self-Test for Operational Faults,”
M. Chatterjee and M. Seuring, “Space Compaction of Test Responses Using Orthogonal Transmission Functions,”
M. Abramovici, C.E. Stroud, and J.M. Emmert, “Online BIST and BIST-Based Diagnosis of FPGA Logic Blocks,”
A. Steininger and J. Vilanek, “Using Offline and Online BIST to Improve System Dependability—The TTPC-C Example,”
H. Al-Asaad, B.T. Murray, and J.P. Hayes, “Online BIST for Embedded Systems,”
M. Abramovici, C. Stroud, B. Skaggs, and J. Emmert, “Improving On-Line BIST-Based Diagnosis for Roving STARs,”

Ioannis Voyiatzis received the BSc degree ininformatics, the MSc degree in electronics automation, and the PhD degree in computer science from the University of Athens, Greece. He is currently with the Department of Informatics at the Technological Educational Institute of Athens, Greece. His research interests include built-in self-test, online testing, and fault tolerance of circuits. He is the author of more than 30 technical papers published in refereed international scientific journals and conference proceedings and 10 books in the IT field. He is a member of the IEEE and the IEEE Computer Society and has been listed in Marquis Who's Who in Science and Engineering and Who's Who in the World.

Antonis Paschalis received the BSc degree in physics, the MSc degree in electronics and computers, and the PhD degree in computer science from the University of Athens, Greece, where he is now an associate professor in the Department of Informatics and Telecommunications. Previously, he was a senior researcher at the Institute of Informatics and Telecommunications, National Research Centre “Demokritos,” Athens. His current research interests are logic design and architecture, VLSI testing, processor testing, and hardware fault tolerance. He has published more than 110 papers and holds a US patent. He has served the test community as vice chair of the Communications Group of the IEEE Computer Society Test Technology Technical Council (TTTC) and participated on several organizing and program committees of international events in the area of design and test. He is a member of the IEEE and a Golden Core Member of the IEEE Computer Society.
Dimitris Gizopoulos received the Engineering Diploma from the University of Patras in 1992 and the PhD degree from the University of Athens in 1997. He is an assistant professor in the Department of Informatics at the University of Piraeus, Greece. His research interests include microprocessors and microprocessor-based systems design, test and fault tolerance, embedded systems design, test and reliability, as well as fault modeling, self-testing, and online testing. He has published more than 80 papers in peer-reviewed transactions, journals, and conference proceedings. He is a coinventor of a US patent, the author of a book, and an editor of a second in test technology. He is an associate editor of the IEEE Transactions on Computers, IEEE Design and Test of Computers, and Springer's Journal of Electronic Testing: Theory and Applications, as well as a guest editor for several special issues. He is involved in the organization and technical program of many international conferences. Since 2004, he has been a member of the Steering Committee of the IEEE International Test Conference (ITC) and, from 2001 to 2006, he was a member of the Steering Committee of the IEEE European Test Symposium (ETS). He was a general chair of the IEEE European Test Workshop in 2002 and the IEEE International On-Line Testing Symposium in 2003. He is a program chair of the IEEE International On-Line Testing Symposium in 2007 and 2008 and the IEEE International Symposium on Defect and Fault Tolerance in 2008. He is also a member of the Organizing and Program Committees of several IEEE conferences. He is member of the Executive Committee of the IEEE Computer Society Test Technology Technical Council (TTTC) with contributions to the Technical Meetings and the Tutorials and Education Groups. He is a senior member of the IEEE and a Golden Core Member of the IEEE Computer Society.

Constantin Halatsis received the BSc degree in physics and the MSc degree in electronics from the University of Athens in 1964 and 1966, respectively, and the PhD degree in computer science from the University of Manchester, United Kingdom, in 1971. From 1971 to 1981, he was a researcher with the Computer Centre, NRC Demokritos, Athens. In 1981, he became a full professor of computer science at the University of Thessaloniki, Greece, and, since 1988, he has been a full professor of computer science in the Department of Informatics and Telecommunications at the University of Athens. He has a wide area of research interests including logic design, computer architectures, fault-tolerant computing, artificial intelligence, and theory of computation. He is the author or coauthor of more than 100 technical papers published in refereed international scientific journals and conference proceedings.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.