2005

Low-Cost Concurrent BIST Scheme for Increased Dependability

Voyiatzis, Ioannis

IEEE

http://hdl.handle.net/11728/6502

Downloaded from HEPHAESTUS Repository, Neapolis University institutional repository
A Low-Cost Concurrent BIST Scheme for Increased Dependability

Ioannis Voyiatzis and Constantin Halatsis

Abstract—Built-In Self-Test (BIST) techniques constitute an attractive and practical solution to the problem of testing VLSI circuits and systems. Input vector monitoring concurrent BIST schemes can circumvent problems appearing separately in online and offline BIST schemes. An important measure of the quality of an input vector monitoring concurrent BIST scheme is the time required to complete the concurrent test, termed Concurrent Test Latency. In this paper, a new input vector monitoring concurrent BIST technique for combinational circuits is presented which is shown to be significantly more efficient than the input vector monitoring techniques proposed to date with respect to Concurrent Test Latency and hardware overhead trade-off, for low values of the hardware overhead.

Index Terms—Built-in self test, concurrent testing, input vector monitoring concurrent BIST.

1 INTRODUCTION

Built-In Self Test (BIST) techniques constitute an attractive and practical solution to the problem of testing VLSI circuits and systems [1]. Input vector monitoring concurrent BIST schemes exploit input vectors arriving at the inputs of the Circuit Under Test (CUT) during normal operation [2], [3], [4], [9]. Input vector monitoring Concurrent BIST schemes are applicable to combinational CUT’s and detect all permanent (as well as some of the transient) combinational (stuck-at) faults at the CUT.

The block diagram of an input vector monitoring concurrent BIST technique is presented in Fig. 1. The CUT is combinational, it has n inputs and m outputs and is tested exhaustively; hence, the test set size is \( T_s = 2^n \). The technique can operate in one of two modes, normal (when \( T_m = 0 \)) and test (\( T_m = 1 \)).

During normal mode the input vector to the CUT, denoted by \( A = A[n:1] \), is driven from the normal input vector \( V[n:1] \). A is also fed 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, the test vector corresponding to A is deactivated and the Response Verifier (RV) captures the CUT response to the input vector A. When all input vectors have performed a hit, the contents of the response verifier are examined, in order to take a decision as to whether a fault has occurred in the CUT.

In case that the CUT does not receive all input vectors during normal operation, the circuit may sporadically operate in test mode in order to complete the test. During test mode, the inputs to the CUT are driven from \( V'[n:1] \) that is generated by the AGC, and the RV is enabled in every test clock cycle.

Two crucial measures that affect the dependability of a Concurrent BIST scheme are the Concurrent Test Latency (CTL), i.e., the mean number of clock cycles required in order to complete the test while the CUT operates in normal mode only and the effective concurrent test latency (CTL_{eff}), defined as the time that the circuit must operate in normal mode so that errors existing in the circuit under test are revealed and an erroneous response is monitored at the response verifier. For the input vector monitoring concurrent BIST schemes proposed to date, CTL_{eff} is equal to CTL.

In this paper, a novel input vector monitoring concurrent BIST scheme is presented, termed window-monitoring Concurrent BIST (w-MCBIST) and its application is studied. A preliminary version of this work was presented in [11]. The presented scheme is shown to be superior of other input vector monitoring concurrent BIST schemes for low values of the hardware overhead. Furthermore, it can be utilized to drive down the value of the CTL_{eff} to any desired ratio of the CTL.

The paper is organized as follows: In Section 2, we introduce the w-MCBIST approach and, in Section 3, we calculate the value of its CTL. In Section 4, we compare w-MCBIST with the input vector monitoring concurrent BIST techniques that have been proposed to date. In Section 5, we study the implementation of w-MCBIST for concurrent testing of ROM modules. In Section 6, the CTL_{eff} of w-MCBIST is calculated. Finally, in Section 7, we conclude the paper.

2 THE w-MCBIST APPROACH

Let \( n \) denote the number of inputs and \( m \) the number of outputs of the combinational CUT to be exhaustively tested. The test set size is \( T_s = 2^n \). w-MCBIST is based on the separation of the test set into nonoverlapping subsets
(called windows) each one of size \( W_s = 2^w \), where \( 0 < w < n \); we shall denote with \( N_W \) the total number of windows; thus, \( N_W = T_s / W_s = 2^n / 2^w = 2^{n-w} \). A block diagram of w-MCBIST is presented in Fig. 2.

During normal operation, w-MCBIST exploits the arrival of any test vector belonging to a specific window, called active window. The input vector is compared against a set of \( W_s \) vectors that constitute the active window as follows. \( n-w \) bits of the \( n \)-bit input vector (denoted by \( d[w+1:n] \)) are compared with the outputs of the \( (n-w) \)-stage test generator and, if they match, cmp is enabled. The remaining \( w \) bits of the input vector are driven to the inputs of a \( w \)-/\( W_s \) decoder whose enable input is driven by cmp. Therefore, if the input vector belongs to the active window, one of the \( D[i] \) signals is enabled.

When all vectors of a window have performed a hit, the signal test generator enable (tg) triggers the test generator to the next state, in order to examine a new window. After the test generator has generated all its \( 2^n \) states, we examine the signature captured in RV and decide whether the CUT is faulty.

The logic module (Fig. 3) consists of \( W_s \) Logic Cells. Each LogicCell[\( i \)], \( 1 \leq i \leq W_s \), corresponds to a vector that belongs to the active window and can be either empty or full, depending on whether the corresponding vector has reached the CUT inputs during the examination of the current active window. This is indicated by the value of the signal full[\( i \)]. When all LogicCells are full, the signal tg enable is enabled; at the next clock cycle, all cells are emptied and the next active window is examined. The schematic of the LogicCell, based on a clocked RS-latch implemented with NAND gates, is also presented in Fig. 3.

Any time, w-MCBIST can switch from normal to test mode. If, at that time, some of the vectors belonging to the current window have reached the CUT inputs, the remaining vectors of the current active window are applied offline before proceeding to examine the next window.

The response verifier compresses the CUT outputs. Since in w-MCBIST the order of the vectors that perform hit
during the examination of a window is not fixed, the final signature must be independent of the order of the output vectors of the CUT. Thus, an order-independent RV is used with w-MCBIST. The Accumulator-Based Compaction is an order-independent response verification technique [12] that has been shown to have aliasing properties similar to the best compactors based on Cellular Automata [13] and Multiple Input Signature Registers [14] and is utilized with w-MCBIST.

### 3 Evaluation of the Concurrent Test Latency

To compute the Concurrent Test Latency of w-MCBIST, we shall assume 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 these 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 [2], [3], [4], [9].

The analytical computation of the Concurrent Test Latency for w-MCBIST requires the computation of the Window Latency (WL), i.e., the number of cycles required so that all the test vectors corresponding to a window are received during normal operation of the circuit, which is performed as follows:

Let $h(i)$ denote the probability of a hit when $i$ vectors remain to be hit in the current window, and $L(i)$ denote the corresponding mean number of cycles. The hit probability is $i/T_s$, and $L(i) = 1/h(i) = T_s/i$. Thus, WL is given by:

$$WL = \sum_{i=1}^{W_s} L(i) = T_s \times \sum_{i=1}^{W_s} \frac{1}{i}.$$  

Since the total number of windows is equal to $T_s/W_s$, CTL is given by:

$$CTL(w-MCBIST) = WL \times \frac{T_s}{W_s} = \frac{T_s^2}{W_s} \times \sum_{i=1}^{W_s} \frac{1}{i}. \quad (1)$$

The above formula has been utilized in Fig. 4 (next section). Furthermore, it has been utilized to calculate the CTL for the ROM modules as presented in Section 5. In order to experimentally validate our analytical results, we used a simulation program in the C programming language, and performed simulations for various values of $n$ and $W_s$. We performed 10 experiments for each case and considered the average value. In all cases, the simulated results differed less than 1 percent from those computed analytically.

### 4 Comparisons

In order to evaluate w-MCBIST, we compare it with the input vector monitoring concurrent BIST techniques proposed hitherto with respect to the value of CTL and the hardware overhead.

The pioneering work on input vector monitoring BIST was carried out by Saluja et al. (C-BIST, [2]). The Active test set Generator and Comparator of C-BIST comprises a single Linear Feedback Shift Register (LFSR) and a comparator and thus the active test set consists of only one active test vector, the current value of the LFSR. C-BIST has low hardware overhead but very high CTL; therefore, three scheme have been proposed to drive down the CTL by increasing the number of active test vectors [3], [4], [9].


In Table 1, we provide the formulas used to calculate the hardware overhead of MHSAT, OISAT ($K = 2^k$) and R-CBIST. In Table 1, we also present the hardware overhead, using the gate equivalents as a metric [5], as a function of the CUT inputs ($n$) and outputs ($m$) and the parameters of each scheme.
The hardware overhead of w-MCBIST is calculated, following Figs. 2 and 3, as follows: The hardware overhead of the LogiCell is five gates; the hardware overhead of the Logic is calculated by:

$$\text{Logic}(W_s) = \frac{W_s}{C_2} \text{LogicCell} + \text{AND}_{W_s} + \text{OR}_{W_s} + \text{NFSR}(w) = 7 \times W_s + 10 \times w.$$  

Overall, the hardware overhead of w-MCBIST is given by:

$$\text{w-MCBIST}(n, m, W_s) = \text{Mux}(n) + \text{Comp}(n - w) + \text{NFSR}(n - w) + \text{Logic}(W_s) + \text{Dec}(W_s) + \text{ABC}(m) = 3n + 4(n - w) + 10(n - w) + 7W_s + 10w + 2W_s + 18m \approx 17n + 9W_s + 18m \text{ gates}.$$  

In Fig. 4, the concurrent test latency is presented (in time units) as a function of the hardware overhead (in gate equivalents) for MHSAT, OISAT, R-CBIST, and w-MCBIST. For the calculation of the CTL of w-MCBIST, the formula (1) has been utilized, as well as the corresponding experimental results. A CUT with $n = 16$ inputs and $m = 16$ outputs and a 100 MHz clock has been considered. The points have been connected with power trend lines.

From Fig. 4, we can see that for the same hardware overhead, w-MCBIST achieves a significantly shorter Concurrent Test Latency than MHSAT and OISAT. Furthermore, if the demand for concurrent test latency is not less than 2 seconds, w-MCBIST achieves the same concurrent test latency with R-CBIST with significantly less hardware overhead. For example, for concurrent test latency equal to 3 seconds, the hardware overhead required by w-MCBIST is approximately 1,150 gates, while the same number for R-CBIST is approximately 1,550, i.e., 35 percent more. Thus, we conclude that w-MCBIST is significantly more efficient than the other input vector monitoring concurrent BIST schemes with respect to the Hardware Overhead—Concurrent Test Latency trade-off.

### 5 Case Study: Comparative Concurrent Testing of ROM Modules

ROM modules constitute critical parts in complex circuits; therefore, high-quality testing is performed on these modules. Concurrent testing is such a high-quality-testing scheme. Practical testing schemes for ROMs use exhaustive testing which is adequate for testing all logically testable combinational faults [7].

The percentage hardware overhead (HO%) is the ratio of the hardware overhead of the BIST circuitry over the hardware overhead of the CUT. In order to calculate the percentage hardware overhead of w-MCBIST, we generated ROMs of various sizes [8] and implemented w-MCBIST for the concurrent testing of these modules. In Table 2, we present the percentage hardware overhead for various values of the window size for w-MCBIST. In Table 2, we also present the Concurrent Test Latency of w-MCBIST as has been calculated by (1). For the calculations in seconds, a 100-MHz clock has been considered. From Table 2, it is derived that in the vast majority of practical cases a percentage hardware overhead less than 10 percent is imposed. This value is generally considered as an acceptable limit for the percentage hardware overhead. Moreover, the corresponding CTL for these cases also lies within acceptable limits. For example, for a 64K \times 8 ROM utilizing CTL with $W_s = 128$, $\text{HO} = 3.65\%$, and the corresponding CTL = 1.82 seconds.

### 6 Increasing Dependability

The dependability of a concurrent BIST technique is a function of 1) the probability that an error that has occurred in the CUT during its operation has been discovered, and 2) the time required so that the error be acknowledged, expressed by the $\text{CTL}_{\text{eff}}$.

For the case that only one signature is examined, at the end of the concurrent test, the effective concurrent test latency is equal to the concurrent test latency; Among the input vector monitoring techniques that have been presented in the literature to decrease the CTL of C-BIST, only w-MCBIST can utilize the multiple-signature concept to increase dependability. More precisely, $\text{CTL}_{\text{eff}}$ can be decreased by utilizing a well-established technique in the BIST literature, namely, multiple signature examination [10], as follows.

When more than one signatures are examined during the concurrent test, the effective concurrent test latency is calculated by:

#### Table 1: Calculation of the Hardware Overhead

<table>
<thead>
<tr>
<th>Technique</th>
<th>modules</th>
<th>gates</th>
</tr>
</thead>
<tbody>
<tr>
<td>f-stage NFSR</td>
<td>NFSR(f)</td>
<td>$f \times \text{DIFF} \times \text{non-linear circuitry}$</td>
</tr>
<tr>
<td>f-stage comparator</td>
<td>Comp(f)</td>
<td>$f \times \text{XOR}_2 \times \text{AND}_7$</td>
</tr>
<tr>
<td>f-output Decoder</td>
<td>Dec(f)</td>
<td>$&lt; f \times 2 \times \text{NAND}_2$</td>
</tr>
<tr>
<td>f-two-way multiplexers</td>
<td>Mux(f)</td>
<td>$f \times \text{MUX}_2$</td>
</tr>
<tr>
<td>f-stage accumulator</td>
<td>ABC(f)</td>
<td>$f \times (\text{DIFF} \times \text{Full Adder})$</td>
</tr>
<tr>
<td>f-stage incrementer</td>
<td>Inc(f)</td>
<td>$f \times \text{Half Adder}$</td>
</tr>
<tr>
<td>f-stage MISR</td>
<td>MISR(f)</td>
<td>$f \times (\text{DIFF} \times \text{XOR}_2)$</td>
</tr>
</tbody>
</table>

| MHSAT(n,m,K)         | $K \times (\text{Mux}(n) + \text{Comp}(n) + \text{NFSR}(n)) + K \times \text{MISR}(m)$ | 17Kn+12Km |
| OISAT(n,m,K)         | $K \times (\text{Mux}(n) + \text{Comp}(n) + \text{NFSR}(n)) + \text{ABC}(m)$ | 17Kn+18m |
| R-CBIST(n,m,w,R)     | $\text{Comp}(w) + \text{NFSR}(n-w) + \text{Inc}(w) + \text{RAM}$ | 11k + 2W + 18m + RAM |
CTLeff = \sum_{i=1}^{\text{number of signatures}} P_i \cdot CTL_i, \quad (2)

where \( P_i \) is the probability that the error(s) are revealed during the examination of the \( i \)-th signature, and \( CTL_i \) is the time required on order to expect that the test patterns corresponding to the \( i \)-th signature have arrived to the CUT inputs. Theorem 1 provides a formula to calculate the effective concurrent test latency when more than one signatures are examined during the concurrent test set. In the sequel, we denote by \( y \) the number of exciting input vectors, i.e., the vectors that exhibit an error on the CUT.

**Theorem 1.** If \( NW \) signatures are examined during exactly every \( T_s/NW \) vectors that perform hit, the effective concurrent test latency can be calculated by (3).

\[
\begin{align*}
\text{CTLeff} &= \frac{\text{CTL}}{NW} \times \left( 1 + \sum_{k=1}^{NW-1} \left( \frac{k}{NW} \right)^y \right) \\
&= \frac{\text{CTL}}{NW} \times \left( 1 + \sum_{k=1}^{NW-1} \left( \frac{k}{NW} \right)^y \right)
\end{align*}
\]

(3)

**Proof.** Let us assume that there are \( y \) exciting vectors and we examine \( NW = T_s/W_s \) signatures, where each signature is examined every exactly \( T_s/NW = W_s \) vector hits. Table 3 presents useful data for the calculations. The first column of Table 3 presents the number of the examined signatures, the second column presents the number of vectors that have performed hit until the corresponding signature is examined, the third column presents the probability that at least one vector exists during the examination of the window, times the fraction of the Concurrent Test Latency until this window is examined. Each quantity in the fifth column is the product of the probability that at least one vector exists during the examination of the window, times the fraction of the Concurrent Test Latency until this window is examined.

The sum of the elements of the last column is equal to \( CTL/NW \) multiplied by:

\[
P_1 + 2 \cdot (P_2 - P_1) + 3 \cdot (P_3 - P_2) + \ldots + NW \cdot (1 - P_{NW-1}) = NW - (P_1 + P_2 + P_3 + \ldots + P_{NW-1}).
\]

Then, by substituting the values of \( P_i \) from the third column of Table 3, we obtain formula (3).

In Fig. 5, the value of the quantity

\[
\frac{\text{CTLeff}}{\text{CTL}} = \frac{1}{NW} \times \left( 1 + \sum_{k=1}^{NW} W - 1 \left( \frac{k}{NW} \right)^y \right)
\]

is presented for various values of \( NW \) and \( y \).

From Fig. 5, it is easy to observe that \( \text{CTLeff} \) asymptotically tends to \( CTL/NW \), as the number of the activating

---

### TABLE 2
Percentage Hardware Overhead and CTL of w-MCBIST versus ROM Size

<table>
<thead>
<tr>
<th>ROM SIZE</th>
<th>( W_s=8 )</th>
<th>( W_s=32 )</th>
<th>( W_s=128 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>#words</td>
<td>word size</td>
<td>H0%</td>
<td>CTL cycles / seconds</td>
</tr>
<tr>
<td>16K</td>
<td>8</td>
<td>4.54%</td>
<td>91,196,160 (0.91 sec)</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>2.72%</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td></td>
<td>2.77%</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td></td>
<td>2.88%</td>
<td></td>
</tr>
<tr>
<td>64K</td>
<td>8</td>
<td>1.13%</td>
<td>1,459,138,560 (14.59 sec)</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>0.72%</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td></td>
<td>0.72%</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td></td>
<td>0.58%</td>
<td></td>
</tr>
</tbody>
</table>

---

### TABLE 3
Contribution of the Signatures to the Effective Concurrent Test Latency

<table>
<thead>
<tr>
<th>#sig</th>
<th>#vectors</th>
<th>( P_i )</th>
<th>( P ) (at least one vector, within window)</th>
<th>Contribution to CTLeff</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( W_s )</td>
<td>( P_1 = 1 - \left( \frac{NW-1}{NW} \right) )</td>
<td>( P_1 ) * CTLeff / NW</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>( 2 \cdot W_s )</td>
<td>( P_2 = 1 - \left( \frac{NW-2}{NW} \right) )</td>
<td>( P_2 - P_1 ) * CTLeff / NW</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>( 3 \cdot W_s )</td>
<td>( P_3 = 1 - \left( \frac{NW-3}{NW} \right) )</td>
<td>( P_3 - P_2 ) * CTLeff / NW</td>
<td></td>
</tr>
</tbody>
</table>

... \( NW-1 \) (\( (NW-1) \cdot W_s \)) \( P_{NW-1} = 1 - \left( \frac{1}{NW} \right)^y \) \( P_{NW-1} - P_{NW-2} \) * CTLeff / NW \( (P_{NW-1} - P_{NW-2}) \) * (\( (NW-1) \cdot W_s \))^y CTLeff / NW

\( NW \) \( T_s/W_s \cdot T_s \) \( T_s \) \( P_{NW} = 1 \) \( 1 - P_{NW-1} \) \( (1 - P_{NW-1}) \) * NW * CTLeff / NW
vectors increases. Thanks to the results of Theorem 1 and Fig. 5, one can reduce $CTL_{\text{eff}}$ to a desired ratio of CTL.

7 CONCLUSIONS

In this paper, a new input vector monitoring concurrent BIST scheme termed window-Monitoring Concurrent BIST (w-MCBIST) has been presented, which is more efficient than the other techniques already known from the open literature in terms of hardware overhead and Concurrent Test Latency for low values of the hardware overhead. The application of the proposed scheme for the concurrent testing of ROMs results in acceptable percentage hardware overhead.

An additional advantage of the presented scheme is that it can exploit the concept of multiple signature examination, decreasing both the effective concurrent test latency and the aliasing probability.

ACKNOWLEDGMENTS

The authors would like to thank the associate editor and the three anonymous referees for their constructive comments towards the improvement of this work.

REFERENCES

Ioannis Voyiatzis received the BSc degree in informatics and telecommunications, the MSc degree in electronics, and the PhD degree in computer science, all from the Department of Informatics and Communications, University of Athens, Greece. His research interests include Built-In Self Test, online testing, and fault tolerance of digital circuits. He is author of numerous technical papers published in refereed international scientific journals and conference proceedings and 10 books in the IT field.

Constantin Halatsis received the BSc degree in physics in 1964 and the MSc degree in electronics in 1966, both from the University of Athens. In 1971, he received the PhD degree in computer science from the University of Manchester, United Kingdom. From 1971 to 1981, he was a researcher with the Computer Centre of NRC Demokritos in Athens. In 1981, he became a full professor of computer science at the University of Thessaloniki, Greece, and, in 1988, he was invited to be a full professor of computer science in the Department of Informatics and Telecommunications of the University of Athens, where he is presently. He has a wide area of research interests including logic design, computer architectures, fault-tolerant computing, artificial intelligence, and theory of computation. He is author/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.