Low-Transition Test Pattern Generation for BIST-Based Applications

Mehrdad Nourani, Senior Member, IEEE, Mohammad Tehranipoor, Member, IEEE, and Nisar Ahmed, Student Member, IEEE

Abstract—A low-transition test pattern generator, called the low-transition linear feedback shift register (LT-LFSR), is proposed to reduce the average and peak power of a circuit during test by reducing the transitions among patterns. Transitions are reduced in two dimensions: 1) between consecutive patterns (fed to a combinational only circuit) and 2) between consecutive bits (sent to a scan chain in a sequential circuit). LT-LFSR is independent of circuit under test and flexible to be used in both BIST and scan-based BIST architectures. The proposed architecture increases the correlation among the patterns generated by LT-LFSR with negligible impact on test length. The experimental results for the ISCAS'85 and '89 benchmarks confirm up to 77 percent and 49 percent reduction in average and peak power, respectively.

Index Terms—Built-In Self-Test, LFSR technique, low-power pattern generation, switching activity, reliability.

1 INTRODUCTION

POWER dissipation is a challenging problem for today’s system-on-chips (SoCs) design and test. In general, the power dissipation of a system in test mode is more than in normal mode [1]. Four reasons are blamed for power increase during test [26]:

1. high-switching activity due to nature of test patterns,
2. parallel activation of internal cores during test,
3. power consumed by extra design-for-test (DFT) circuitry, and
4. low correlation among test vectors.

This extra power consumption (average or peak) can create problems such as instantaneous power surge that cause circuit damage, formation of hot spots, difficulty in performance verification, and reduction of the product yield and lifetime. Solutions that are commonly applied to alleviate the excessive power problem during test include reducing frequency and test partitioning/scheduling to avoid hot spots. The former disrupts at-speed test philosophy and the latter may significantly increase the time.

Built-In Self-Test (BIST) is a DFT methodology that aims at detecting faulty components in a system by incorporating the test logic on chip. BIST is well known for its numerous advantages such as at-speed testing and reduced need for expensive external automatic test equipment (ATE). In BIST, a linear feedback shift register (LFSR) generates pseudorandom test patterns for primary inputs (for a combinational circuit) or scan chain inputs (for a sequential circuit). On the observation side, a multiple input signature register (MISR) compacts test responses received from primary outputs or scan chain outputs. Unfortunately, BIST-based structures are very vulnerable to high-power consumption during test. Test vectors, applied to a circuit under test at nominal operating frequency, often cause more average and/or peak power dissipation than in normal mode. The main reason is that the random nature of patterns generated by an LFSR significantly reduces the correlation not only among the patterns but also among adjacent bits within each pattern.

1.1 Prior Work

Several techniques have been reported to cope with high-power consumption in BIST-based architectures. These techniques can be classified into three broad categories as follows:

- **System-Level Partitioning and/or Scheduling.** The technique proposed in [1] consists of a distributed BIST control scheme that simplifies BIST architecture for complex ICs, especially during higher levels of test activity. This approach can schedule the execution of every BIST element to keep the power dissipation under a specified limit. A partitioning method using hypergraph is employed in [10] to lower power in BIST designs. References [11] and [12] are two optimization techniques (using mixed-ILP and ILP, respectively) that insert idle time windows in the test schedule to make sure that power and other thermal constraints are satisfied.

- **Correlation-Driven Enhancement of LFSR.** A BIST strategy called dual-speed LFSR is proposed in [2] to reduce the circuit’s overall switching activities. This technique uses two different-speed LFSRs to control those inputs that have elevated transition densities.
The low-power test pattern generator presented in [3] is based on cellular automata and reduces the test power in combinational circuits. Another low-power test pattern generator based on a modified LFSR is proposed in [4]. This scheme reduces the power in CUT in general and clock tree in particular. A low-power BIST for data path architecture, built around multiplier-accumulator pairs, proposed in [5], is circuit dependent. This dependency implies that nondetecting subsequences must be determined for each circuit test sequence. A low-power BIST based on state correlation analysis is proposed in [6].

Modifying the LFSR by adding weights to tune the pseudorandom vectors for various probabilities decreases energy consumption and increases fault coverage [7], [8]. A low-power random pattern generation technique to reduce signal activities in the scan chain is proposed in [9]. In this technique, an LFSR generates equally probable random patterns. The technique generates random but highly correlated neighboring bits in the scan chain, reducing the number of transitions and, thus, the average power.

• **Power-Driven Control of LFSR.** The authors of [13] proposed a method to select an LFSR’s seed to reduce the lowest energy consumption using a simulated-annealing algorithm. Test vector inhibiting techniques [14], [15], [16] filter out some nondetecting subsequences of a pseudorandom test set generated by an LFSR. These architectures apply the minimum number of test vectors required to attain the desired fault coverage and therefore reduce power.

Many low-power strategies have been proposed for full-scan [21], [22] and BIST/scan-based BIST architectures [17], [18], [19], [20]. The architecture proposed in [17] modifies the scan-path structure such that the CUT’s inputs remain unchanged during a shift operation. A test pattern generator for scan-based BIST was proposed in [18] which reduces the number of transitions that occur at scan inputs during scan-shift operation. The authors of [19] proposed a pseudorandom BIST scheme to reduce the switching activity in the scan chains. The activity and correlation in CUT are controlled by limiting the scan shifts to a portion of the scan chain structure using scan chain disable control.

### 1.2 Contribution and Paper Organization

This paper presents a new test pattern generator for low-power BIST (LT-LFSR), which can be employed by both combinational and sequential (scan-based) architectures. The proposed technique increases the correlation in two dimensions: 1) the vertical dimension between consecutive test patterns (Hamming Distance) and 2) the horizontal dimension between adjacent bits of a pattern sent to a scan chain. Reducing the switching activity in turn results in reducing the power consumption, both peak and average. The conventional LFSR structure will be modified such that it automatically inserts intermediate patterns between its original pairs. The intermediate patterns are carefully chosen using two techniques (that is, bipartite and random injection) and impose minimal time increase to achieve a desired fault coverage. The favorable features of LT-LFSR in terms of performance, coverage, and average/peak power consumption are verified using the ISCAS’85 and ISCAS’89 benchmarks.

The rest of this paper is organized as follows: Section 2 describes our motivation of designing a new random pattern generator. Section 3 describes the randomness of test patterns generated by our proposed techniques. Section 4 describes implementation of the two proposed techniques (RI and Bipartite) for low-power test pattern generation and combines them to design our LT-LFSR. Section 5 discusses some practical aspects of LT-LFSR. The experimental results are discussed in Section 6. Finally, the concluding remarks are in Section 7.

### 2 BACKGROUND AND MOTIVATION

#### 2.1 Behavior and Applications of LFSR

Random pattern generators such as LFSR usually generate very low correlated patterns. Assume that $T_i = \{t_i^1, t_i^2, \ldots, t_i^n\}$ and $T_{i+1} = \{t_{i+1}^1, t_{i+1}^2, \ldots, t_{i+1}^n\}$ are two consecutive patterns. The number of bits in the test patterns ($n$) is equal to either the number of PIs or the length of the scan chain in the CUT. If $T_i$ is used for combinational circuits, then it is applied to PIs. If $T_i$ is a pattern generated to be used in sequential circuits, it is applied to the scan-in pin (SI) of a scan chain in the circuit.

#### 2.2 Test per Clock versus Test per Scan

- **Test per Clock.** Assume that $T_i$ and $T_{i+1}$ are two consecutive patterns, and the number of bit changes (transitions) between two consecutive patterns $(\sum_{j=1}^{n} |t_j^i - t_j^{i+1}|)$ is high. If low-correlated patterns are applied to PIs of combinational circuits (see Fig. 1a), they generate a high number of transitions at the PIs, which in turn results in huge number of switching activities in CUT. The transitions between two consecutive test patterns are shown in Fig. 1c.

- **Test per Scan.** Assume that bit $j$ of patterns is shifted into the scan chain and the number of transitions among the adjacent bits going into the scan chain $SC_{j}(\sum_{i=1}^{n} |t_j^i - t_j^{i+1}|)$ is high. Fig. 1b shows a test-per-scan architecture that uses a random pattern generator, that is, LFSR. It also uses MISR as the signature analyzer at the output of the scan chains to receive the responses. If low-correlated patterns are used for testing sequential circuits, they will result in a large number of transitions in scan chains and combinational block during shifting the patterns into the scan chains. The transitions between two adjacent bits when shifted into a scan-in chain are shown using arrows in Fig. 1c.

#### 2.3 Motivation and Focus

Almost all of the proposed techniques of random pattern generators reduce the transitions either within the patterns or between the patterns [4], [23], [9]. In this paper, our goal is to design a new random pattern generator that reduces...
the total number of transitions among the adjacent bits in each random pattern (horizontal dimension) and transitions between two consecutive random patterns (vertical dimension) as well (see Fig. 1c). In other words, the new low-transition random pattern generator increases the correlation between and within patterns and can therefore be used for any combinational or sequential circuits.

We propose a random pattern generator that combines two methods of test pattern generation called Random Injection (RI) and Bipartite LFSR. Briefly, the RI method inserts a new intermediate pattern between two consecutive test patterns by positioning a random-bit \(R_i\) in the corresponding bit of the intermediate pattern when there is a transition between corresponding bits of pattern pairs. The Bipartite LFSR generates an intermediate pattern using one half of each of the two consecutive random patterns. The main advantage of our proposed technique is that it can be used for both combinational and sequential circuits and the randomness quality of patterns does not deteriorate.

There are two additional favorable features in our low-power LFSR.

1. Both the peak and average power consumptions are reduced. The peak power reduction to alleviate the thermal and signal integrity problems during test is often the main goal. However, reducing the average power will improve reliability. Moreover, in some cases, reducing the average test power is quite beneficial. For example, some portable devices need to be self-tested periodically during their lifetime cycle [26].

2. The second feature of our LT-LFSR is its negligible effect on the fault coverage convergence. By injecting intermediate patterns into our technique, the correlation among patterns will change. However, in Section 5, we will show that the effect on performance for achieving a target fault coverage is negligible. Note that, even though intermediate patterns are generated between consecutive patterns, the test length (number of patterns required to achieve a target fault coverage) compared to a conventional random pattern generator is quite close. This is achieved by preserving the randomness of the inserted patterns. We will show our evidence using both the ISCAS’85 (combinational) and ISCAS’89 (sequential) benchmarks in Section 6.

3 Random-Bit Injection Methodologies

3.1 Definition of Randomness Metric

Many researchers used entropy as a measure of randomness metric [32], [33]:

\[
H = - \sum_{i=1}^{r} p_i \cdot \log_2 p_i,
\]

where \(p_i\) is the probability that the signal is in state \(i\) and \(r\) denotes total number of states. This metric can quantify how the quality of pseudorandom values deteriorates if there is a biased change in bit selection or sequencing. More specifically, for an \(n\)-bit perfect random generator, we have \(r = 2^n\) and \(p_i = 1/2^n\) and, thus, the entropy will be \(H_{max} = n\), reflecting the maximum randomness. For a nonideal random generator, we get \(0 \leq H \leq n\). To make it easier for computation in an \(n\)-bit LFSR, if \(p_{ij} (p_{ij})\) denotes the probability of having 0 (1) in bit \(b_j\), then we approximate its entropy by adding the entropy of individual bits:

\[
H \approx - \sum_{j=1}^{n} (p_{0j} \cdot \log_2 p_{0j} + p_{1j} \cdot \log_2 p_{1j}).
\]

3.2 Randomness in Conventional LFSR

LFSR units are expected to generate pseudorandom patterns that behave quite close to ideal random numbers \((H \approx n)\). To show this better, we analyzed the first 10,000 patterns generated by a 20-bit LFSR with polynomial \(f(x) = x^{20} + x + 1\). The results are shown in Figs. 2, 3, and 4 for conventional LFSR.

Fig. 2 shows that, if the number of patterns chosen \((N = 10,000\) here) is large, each bit \(b_j (1 \leq j \leq 20)\) would almost equally get 0s and 1s. In practice, depending on the polynomial used in the LFSR, the randomness is not perfect. That is why, in Fig. 2, for example, for LFSR, we get around 4,930 zeros (that is, \(p_{0j} \approx 0.493\) and \(H \approx 19.72\)) instead of exactly 5,000 (\(H_{max} = 20\)). Figs. 3 and 4 picture distribution
of bit transitions vertically (between two consecutive patterns fed to a combinational circuit) and horizontally (among adjacent bits chosen from one bit position and fed to a sequential circuit), respectively. The normal curve behavior in Fig. 3 is expected due to close-to-perfect randomness of bits generated in an LFSR. Note carefully that, in Fig. 4, it is expected that the curves for bipartite and RI-LFSR will be identical as the total number of transitions among adjacent bits chosen from one bit position and sent into a scan chain serially remains the same due to the relation (see Fig. 1).

It is also expected that the number of transitions for LFSR become almost twice that of the other two methods. This is because, for the case of LFSR, it generates 10,000 patterns. However, for bipartite-LFSR and RI-LFSR, only 5,000 of those patterns are used, in which another 5,000 patterns are added to lower the transitions. This way we will be able to compare three methods for the same number of total patterns (that is, 10,000).

3.3 Randomness in Bipartite LFSR
The implementation of an LFSR can be changed to improve some design features, such as power, during test. However, such a modification may change the order of patterns or insert new patterns that affect the overall randomness. For example, suppose that $T_i$ and $T_{i+1}$ are two consecutive...
patterns generated by an n-bit LFSR. The maximum number of transitions will be n when \( T^i \) and \( T^{i+1} \) are complements of each other. One strategy, used in [23] to reduce number of transitions to maximum of \( n/2 \), is to insert a pattern \( T^i \), half of which is identical to \( T^i \) and \( T^{i+1} \). This Bipartite (half-fixed) strategy is shown symbolically in Fig. 5.

The Bipartite strategy guarantees the transition change to be limited to \( n/2 \) between two consecutive patterns. However, it deteriorates the randomness to \( H = n/2 \). Intuitively, the worst-case scenario (\( H = 0 \)) belongs to a case in which all transitions happen in the same half that we fix. In this case, \( T^i \) and \( T^{i+1} \) will be identical and adding \( T^i \) has no significance for fault detection. It only prolongs the test. To see the randomness drop more clearly, we repeated the same experiment for a modified 20-bit LFSR and the results are shown in Figs. 2, 3, and 4.

### 3.4 Randomness in RI-LFSR

To preserve the randomness of patterns, instead of Bipartite strategy, we randomly inject a value in bit positions, where \( t^i_j \neq t^{i+1}_j \). Briefly,

\[
\begin{align*}
    t^i_j &= \begin{cases} 
        t^i_j & \text{if } t^i_j = t^{i+1}_j \\
        R & \text{if } t^i_j \neq t^{i+1}_j
    \end{cases}
\end{align*}
\]

Fig. 6 shows this symbolically. The shaded cells show those bit positions where \( t^i_j \neq t^{i+1}_j \). We insert a random bit (shown as \( R \) in \( T^{i+1} \)) if the corresponding bits in \( T^i \) and \( T^{i+1} \) are different. Note that, since such bits are uniformly distributed and we also replace them with another random value, the overall randomness remains unchanged, that is, \( H_{\text{max}} = n \). Unfortunately, the maximum bit transition can no longer be guaranteed, although the expected number of transitions (mean value in the normal distribution) will be \( n/2 \).

To verify the high randomness of this strategy, we repeated the same experiment for a modified 20-bit LFSR and the results are shown in Figs. 2, 3, and 4.

In Section 4, we will show how to design and mix these two strategies (Bipartite and RI) to have an LFSR in which the maximum number of transitions is guaranteed to be \( n/4 \), while the randomness of patterns is largely preserved.

### 4 Low-Transition Linear Feedback Shift Register Architecture

#### 4.1 Implementing the RI Technique

The RI technique (Section 3.4) inserts a new test pattern \( T^{i+1} \) between these two test patterns such that the sum of the PI’s activities between \( T^i \) and \( T^{i+1} \) (\( N^{i,\text{trans}} \)) and \( T^i \) and \( T^{i+1} \) (\( N^{i+1,\text{trans}} \)) is equal to the activities between \( T^i \) and \( T^{i+1} \) (\( N^{i,i+1,\text{trans}} \)) or, briefly,

\[
N^{i,\text{trans}} + N^{i+1,\text{trans}} = N^{i+1,\text{trans}}.
\]

For example, after generating \( T^i \), \( N^{i,\text{trans}} \) is partitioned into two parts, \( N^{i,\text{trans}} \) and \( N^{i+1,\text{trans}} \), which reduce the patterns’ switching activity. When two same-position bits in \( T^i \) and \( T^{i+1} \) are equal, the same bit is placed in the same position in \( T^{i+1} \). When there is a transition between two corresponding bits in \( T^i \) and \( T^{i+1} \), the RI method injects random-bit (\( R \)).

Fig. 7 shows a small example of generating an intermediate pattern using the RI technique. The shaded bits in \( T^i \) and \( T^{i+1} \) show that the number of transitions between \( T^i \) and \( T^{i+1} \) is 10 before inserting \( T^{i+1} \). For example, after generating \( T^{i+1} \) using the RI method, as shown in Figs. 6 and 4 or 4 and 6 (depending on \( R = 0 \) or 1), transitions exist between \( T^i \) and \( T^{i+1} \) and \( T^i \) and \( T^{i+1} \), respectively. There are a maximum of six transitions for the RI technique regardless of \( R = 0 \) or \( R = 1 \). In general, for \( n \)-bit vectors if \( m \) (\( m \leq n \)) transitions exist between \( T^i \) and \( T^{i+1} \):

\[
\begin{align*}
    \text{Worst Case : } N^{i,\text{trans}} &= 0, N^{i+1,\text{trans}} = m \quad \text{(or vice versa)}, \\
    \text{Best Case : } N^{i,\text{trans}} &= N^{i+1,\text{trans}} = m/2.
\end{align*}
\]

\[
\sum_{j=1}^{n} |t^i_j - t^{i+1}_j| + \sum_{j=1}^{n} |t^i_j - t^{i+1}_j| = \sum_{j=1}^{n} |t^i_j - t^{i+1}_j|.
\]
Fig. 8 shows the RI unit that generates intermediate patterns. \( R \) is a random bit which can come from one of the outputs of a random pattern generator (for example, LFSR) itself.

### 4.2 Implementing Bipartite LFSR Technique

This technique inserts an intermediate test pattern \( T_{i+1} \) between two consecutive random patterns \( T_i \) and \( T_{i+1} \) such that the transitions between \( T_i \) and \( T_{i+1} \) are reduced. In this technique, each half of \( T_{i+1} \) is filled with half of \( T_i \) and \( T_{i+1} \):

\[
T_{i+1} = \left\{ t_i^1, \ldots, t_i^n, t_{i+1}^1, \ldots, t_{i+1}^n \right\}. \tag{7}
\]

In this method, an LFSR is divided into two halves by applying two complementary (nonoverlapping) enable signals. In other words, when one half is working, the other half is in idle mode. An LFSR including flip-flops with enable is shown in Fig. 9a. Fig. 9b shows the architecture of the Bipartite LFSR to generate intermediate pattern \( T_{i+1} \). \( en_1 \) and \( en_2 \) are two nonoverlapping enable signals. When \( en_1en_2 = 10 \), the first half of LFSR is working, whereas, with \( en_1en_2 = 01 \), the second half works. The shaded flip-flop is added to the Bipartite LFSR architecture to store the \( n/2 \)th bit of LFSR when \( en_1en_2 = 10 \) and send its value into the \((n/2 + 1)\)th flip-flop when the second half becomes active (\( en_1en_2 = 01 \)). Note carefully that the new (shaded) flip-flop does not change the characteristic function of LFSR. The LFSR’s operation is effectively split into two halves, and the shaded flip-flop is an interface between these two.

This method is similar to the proposed LPATPG in [23] and Modified Clock Scheme LFSR [4]. Although the basic idea of Bipartite LFSR is not new, the LT-LFSR architecture is much more efficient in terms of randomness of patterns and power. In [23], the authors used two \( n \)-bit random pattern generators and \( n \times 2 \times 1 \) multiplexers, but we only add one flip-flop to an \( n \)-bit LFSR. Therefore, the area overhead of Bipartite LFSR is much lower than LPATPG. In [4], an \( n \)-bit LFSR is divided into two \( n/2 \)-bit LFSRs, which together reduce the CUT and clock tree power consumption. The drawback of this technique is that it reduces the randomness property of the LFSR due to dividing it into two smaller LFSRs and it also requires generating and distributing two nonoverlapping clocks (with half frequency), which in turn increases the area overhead.

Our Bipartite LFSR keeps the randomness property of the \( n \)-bit LFSR intact and it also reduces the overall power consumption of Bipartite LFSR compared to LFSR because, in each period of the clock, half of the LFSR is in idle mode. Fig. 9c shows a small example of inserting an intermediate pattern \( T_{i+1} \) between two consecutive patterns \( T_i \) and \( T_{i+1} \) using a 16-bit Bipartite LFSR. This reduces the bit transitions among patterns from \( N_{\text{trans}} = 10 \) to \( N_{\text{trans}} = 7 \) and \( N_{\text{trans}} = 3 \).

### 4.3 Implementing Low-Transition Linear Feedback Shift Register Architecture

We combine our two proposed techniques of pattern generation (RI and Bipartite LFSR) for low-power BIST. The new LT-LFSR generates three intermediate patterns \( T_{i+1}, T_{i+2}, \) and \( T_{i+3} \) between \( T_i \) and \( T_{i+1} \). We embed these two techniques into a bit-sliced LFSR architecture to create LT-LFSR, which provides more power reduction compared to having only one of the R-Injection and Bipartite LFSR techniques in an LFSR. This may seem to prolong test session by a factor of 4. However, due to the high
randomness of the inserted patterns, many of the intermediate patterns can do as well as patterns generated by an LFSR in terms of fault detection. In fact, in Section 6, we show that the overall number of LT-LFSR patterns to hit a fault coverage target is quite close to the number of conventional LFSR patterns.

Fig. 10 shows LT-LFSR with RI and Bipartite LFSR included. The LFSR used in LT-LFSR is an external-XOR LFSR. As shown, an injector circuit taps the present state (conventional LFSR patterns, as shown in Fig. 9. MUXs select either the injection bit or the exact bit in LFSR. One very small (46 gates, see Section 6) finite-state machine (FSM) controls the pattern generation process as follows:

1. **Step 1.** \( e_{n1}e_{n2} = 10, \) \( s_{el1}s_{el2} = 11. \) The first half of LFSR is active and the second half is in idle mode. Selecting \( s_{el1}s_{el2} = 11, \) both halves of LFSR are sent to the outputs \( (O_1 \) to \( O_n). \) In this case, \( T^1 \) is generated.

2. **Step 2.** \( e_{n1}e_{n2} = 00, \) \( s_{el1}s_{el2} = 10. \) Both halves of LFSR are in idle mode. The first half of LFSR is sent to the outputs \( (O_1 \) to \( O_{n/2}), \) but the RI injector circuit outputs are sent to the outputs \( (O_{n/2} \) to \( O_n). \) \( T^2 \) is generated.

3. **Step 3.** \( e_{n1}e_{n2} = 01, \) \( s_{el1}s_{el2} = 11. \) The second half of LFSR works and the first half of LFSR is in idle mode. Both halves are transferred to the outputs \( (O_1 \) to \( O_n) \) and \( T^2 \) is generated.

4. **Step 4.** \( e_{n1}e_{n2} = 00, \) \( s_{el1}s_{el2} = 01. \) Both halves of LFSR are in idle mode. From the first half, the injector outputs are sent to the outputs of LT-LFSR \( (O_1 \) to \( O_{n/2}) \) and the second half sends the exact bits in LFSR to the outputs \( (O_{n/2+1} \) to \( O_n) \) to generate \( T^3 \).

5. **Step 5.** The process continues by going through Step 1 to generate \( T^{n+1}. \)

LT-LFSR reduces the transitions between consecutive patterns that can be used for test-per-clock architecture. The generated patterns can also be used for test-per-scan architecture to feed scan chains with a lower number of transitions. We will discuss this more in Section 5.

5 **Practical Aspects**

5.1 **Time-Fault Coverage Relationship in LT-LFSR**

Suppose a conventional LFSR generates \( N \) patterns for a maximum fault coverage \( (FC^*) \) for a CUT. Since LT-LFSR adds three intermediate patterns between LFSR patterns, it generates a total of \( 4N - 3 \) patterns. Although the worst-case scenario seems to quadruple the overall test time, this never happens in practical cases when the goal is to hit a target fault coverage. Fig. 12 is an intuitive illustration of this fact. The FC curve for the majority of circuits rises exponentially (for example, point FC1 after \( N/10 \) patterns in LFSR) and then continues toward \( FC^* \) logarithmically. In LT-LFSR, after \( 4N/10 \) patterns, we will be at FC1 (worst case) or higher since all of those \( N/10 \) LFSR patterns are included. After that, an absolute worst-case (pessimistic) scenario is a case to hit \( FC^* \) at \( 4N. \) In all of the examples we tried so far, this never happened because the random nature of patterns is preserved in LT-LFSR and almost all of the original LFSR patterns are generated much earlier than the 4N point. For example, for the s13207 ISCAS’85 benchmark, the required number of patterns to hit \( FC^* = 97.7\% \) are 77,696 and 78,832 for LFSR and LT-LFSR, respectively. This is only a 1.5 percent increase for a large (about 8,500 gates) circuit. Empirically, \( FC^* \) (or a higher point) is often hit in 0.9 N to 1.3 N range using LT-LFSR patterns, as shown in Fig. 12. Our experimental results shown in Section 6 also confirm this.

<table>
<thead>
<tr>
<th># clk</th>
<th>pattern</th>
<th>en1 en2 sel1 sel2</th>
<th>LT-LFSR (R=0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( T^1 )</td>
<td>1 0 1 1</td>
<td>1010 1011</td>
</tr>
<tr>
<td>2</td>
<td>( T^{11} )</td>
<td>0 0 1 0</td>
<td>1010 1111</td>
</tr>
<tr>
<td>3</td>
<td>( T^{12} )</td>
<td>0 1 1 1</td>
<td>0100 0101</td>
</tr>
<tr>
<td>4</td>
<td>( T^{13} )</td>
<td>0 0 0 1</td>
<td>1111 0101</td>
</tr>
<tr>
<td>5</td>
<td>( T^{14+1} )</td>
<td>1 0 1 1</td>
<td>0101 0101</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>... ... ...</td>
<td>... ... ...</td>
</tr>
</tbody>
</table>

Fig. 11 shows patterns generated using an 8-bit LP-LFSR with polynomial \( x^3 + x + 1 \) and seed = 01001011. As shown, between two consecutive patterns \( T^n \) and \( T^{n+1}, \) three intermediate patterns are generated as \( N^{n+1}_1, \) \( N^{n+1}_2, \) \( N^{n+1}_3, \) and \( N^{n+1}_4, \) respectively. This reduction of transitions eventually reduces the average and peak power during test.
statistical analysis. Using a lesser number of patterns (for example, 0.9 N to hit $FC^*$) for some examples is not a surprise as the intermediate patterns inserted by LT-LFSR were good enough to catch some of the hard-to-detect faults. The rate of growth of fault coverage for LFSR and LT-LFSR patterns were performed for one of the ISCAS’89 benchmarks in Section 6.

5.2 Performance Drawback
The additional components in LT-LFSR impose extra delay, which in turn causes slight performance degradation compared to its LFSR counterpart. Our implementation using Synopsys Design Compiler and 0.18 µm library indicates that, in the worst-case scenario, using an LT-LFSR circuit, a maximum of 0.1 ns is added to the critical path delay of the unit. For example, for a circuit originally running at 100 MHz, the delay added by LT-LFSR makes the circuit run at 99.1 MHz, which is a negligible reduction.

5.3 BIST Applications
Figs. 13a and 13b show the application of LT-LFSR in test-per-clock (BIST) and test-per-scan (Scan-BIST) architectures, respectively. LT-LFSR can replace LFSR in all applications to generate and feed pseudorandom test patterns into the circuit or scan chains. Specifically, the patterns applied through a scan chain reduce the scan-in power because of a lesser number of transitions generated inside the patterns. Note that we have not used phase shifter in this technique and that is mainly because the phase shifter may change the patterns at its output and that may not result in high test power reduction.

5.4 Power Consumption of Low-Transition Linear Feedback Shift Register
The power consumption of LT-LFSR itself is also reduced due to using the Bipartite LFSR technique. Only half of the LT-LFSR components are clocked in each cycle. Fig. 15 indicates this behavior. At each of the four steps of pattern generation process, either half of the flip-flops or half of the RI units becomes active by $en_1en_2$ and $sel_1sel_2$, respectively. In an LFSR, all flip-flops are clocked at the same time in each clock cycle and, thus, its power consumption is much higher than LT-LFSR. See Section 6 for more statistics.

5.5 Circuit-Independent Structure
Several methods were proposed for low-power BIST using test vector inhibiting [14], [15], [16] to filter out some nondetecting subsequences of a pseudorandom test set generated by an LFSR. These methods result in more power reduction, but have high area overhead. More importantly, they are customized for the CUT (test pattern-dependent) and need to start with a specific seed. Therefore, a preprocessing step is required to obtain the nondetecting subsequences and seed. Although LT-LFSR is totally independent of CUT and no preprocessing is needed to obtain a seed. LT-LFSR has a flexible structure that can replace a conventional LFSR in any circuit.

5.6 Randomness in LT-LFSR
Figs. 16, 17, and 18 show the high randomness of 10,000 LT-LFSR patterns generated under polynomial $x^{20} + x + 1$. As seen in Fig. 16, the number of 0s and 1s is almost equal, which indicates very good randomness for each bit. Fig. 17 shows that a curve has been shifted to the left compared to LFSR’s curve in Fig. 3. This is expected as, by inserting three transitions exists in each scan chain ($SC_1$ through $SC_8$). This shows that LT-LFSR is quite capable of reducing transitions in each test pattern applying to scan chain.

Fig. 12. Time-coverage relationship in LT-LFSR.

Fig. 13. Using LT-LFSR in BIST architectures: (a) test-per-clock and (b) test-per-scan.

Fig. 14. Test patterns generated using an 8-bit LT-LFSR used for test-per-scan architecture.
patterns (see Fig. 11), the maximum number of transitions will drop to \( n/4 \) (five in our case). Finally, Fig. 18 shows that number of transitions in each bit position if that position feeds a scan chain. Again, this is almost four times better than conventional LFSR (Fig. 4).

6 EXPERIMENTAL RESULTS

6.1 Simulations Setup and Implementation

In our experimentation, we used polynomial \( x^n + x + 1 \) for both LFSR and LT-LFSR of different lengths. The results are shown for both combinational and sequential ISCAS ('85 and '89) benchmarks. We have selected the four largest ISCAS'85 and four largest ISCAS'89 benchmarks in our experiments. All circuits are synthesized using Synopsys' Design Compiler [24]. The same tool is used for scan chain insertion for ISCAS'89 benchmarks. Twenty scan chains were inserted into these ISCAS'89 benchmarks. The circuits are optimized using the Artisan TSMC library based on 0.18 \( \mu m \) technology. Fault coverage is obtained using the TetraMax tool [24] from Synopsys. Power consumption has been measured at the gate level using PrimePower [24], assuming a power supply voltage of 1.8 V. PrimePower reports the entire power consumed in the circuit-under-test and that includes scan-in, power consumed in combinational blocks, and scan-out power. The simulation is performed with back-annotation using a standard delay format (SDF) file containing the delay information of each gate in the netlist. This process is performed for all two test data sets, that is, LFSR and LT-LFSR.

Below, we summarize the steps used in obtaining the fault coverage and the required number of test patterns for LFSR and LT-LFSR:

1. First, the test patterns are generated using an LFSR written in C++.
2. The required number of test patterns (\( N_p \) of LFSR) to target a certain fault coverage (\( FC \)) is obtained using a Fault Simulator in TetraMax [24].
3. Low-power test patterns are generated using LT-LFSR with the same seed as used for LFSR in Step 1, again written in C++.
4. Repeat Step 2 to achieve the same \( FC \) for LT-LFSR patterns. Note that TetraMax has an option that asks the user to enter the desired fault coverage. Here, we are trying to compare the required number of patterns for both LFSR and LT-LFSR that achieve the same fault coverage. Therefore, the same fault coverage obtained from Step 2 is used as the target fault coverage in this step. The required number of low-power patterns (\( N_p \) of LT-LFSR) to meet the same \( FC \) is obtained.
Table 1 shows the specifications of the ISCAS benchmarks and the number of test patterns \((N_p)\) required to hit a target fault coverage \((FC^*)\) for LFSR and LT-LFSR. This table also compares our results with techniques proposed in [23] and [4] for the number of patterns and fault coverage. Reference [23] and [4] report results only on combinational and sequential benchmarks, respectively. Reference [4] seems to limit \(N_p\) and, thus, achieves a lower fault coverage.

In general, the performance of both LFSR and LT-LFSR \((N_p \text{ to hit } FC^*)\) is seed and polynomial independent. According to this table, to hit the target \(FC^*\), LT-LFSR uses at most \(+/-10\) percent more/less patterns than that of LFSR for the majority of the benchmarks. As seen in a few cases (for example, c1908 and c5315), \(N_p\) slightly (13.3 percent and 1.6 percent, respectively) drops, showing that some of the intermediate patterns did a good job in fault detection. We used 50 different seeds for 10 different polynomials in our experiments and the results were almost the same as what was shown in the table. This confirms that the performance is seed and polynomial independent.

The comparison between LT-LFSR and other techniques such as those proposed in [14], [15], [16] is not feasible since those are conceptually different. These techniques try to filter the nondetecting vectors that result in reduction in the average test power and may also reduce the peak power. However, if the pattern that causes the peak power is a detecting pattern, then it will not be masked and the peak power will not be reduced. A preprocessing step is also required to find the nondetecting patterns. The technique proposed in [17] reduces the power during test by suppressing the output switch during shifting. The techniques proposed here and in [14], [15], [16] are different from that in [17] since these techniques work on test patterns. Compared to all of these techniques, our proposed technique reduces the switching activity among patterns and this has given us a significant reduction in both peak and average power.

### 6.2 The Rate of Growth of LT-LFSR Fault Coverage

Fig. 19 shows the rate of the growth of fault coverage for the s38584 benchmark. The figure shows that the new LT-LFSR increases the fault coverage almost the same way as an LFSR does. The empirical results shown in Table 1 and in this figure verify the argument in Section 5 that the required number of LT-LFSR patterns to provide a target fault coverage \((FC^*)\) does not quadruple. In fact, due to preserving randomness in LT-LFSR, the number of patterns (and, therefore, the required time) to hit \(FC^*\) remains quite close to the number of LFSR patterns.

### 6.3 Average and Peak Power Reduction

Table 2 shows the average and peak power of LFSR and LT-LFSR for ISCAS benchmarks. As expected, LT-LFSR significantly reduces the average and peak power. Table 3 shows the average and peak power reduction of LT-LFSR compared to LFSR, that is, \(\Delta P_{avg} = \frac{P_{avg}(LFSR) - P_{avg}(LT-LFSR)}{P_{avg}(LFSR)}\). As shown, LT-LFSR reduces up to 77 percent and 49 percent

![Fig. 19. The rate of growth of fault coverage for the s38584 benchmark.](image)

### TABLE 1

<table>
<thead>
<tr>
<th>Circuit</th>
<th>PI</th>
<th>PO</th>
<th>Circuit Size Gates+FFs</th>
<th>(FC^*) %</th>
<th>(N_p)</th>
<th>LFSR</th>
<th>LT-LFSR</th>
<th>(FC^*)</th>
<th>(N_p)</th>
<th>[23]</th>
<th>(FC^*)</th>
<th>(N_p)</th>
<th>[4]</th>
</tr>
</thead>
<tbody>
<tr>
<td>c1908</td>
<td>33</td>
<td>25</td>
<td>880+0</td>
<td>95.9</td>
<td>996</td>
<td>863</td>
<td>95.3</td>
<td>1116</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>c2670</td>
<td>233</td>
<td>140</td>
<td>1193+0</td>
<td>91.5</td>
<td>1952</td>
<td>1988</td>
<td>84.3</td>
<td>2940</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>c3540</td>
<td>50</td>
<td>22</td>
<td>1669+0</td>
<td>97.8</td>
<td>1164</td>
<td>1052</td>
<td>92.3</td>
<td>1049</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>c5315</td>
<td>178</td>
<td>123</td>
<td>2307+0</td>
<td>99.7</td>
<td>1129</td>
<td>1111</td>
<td>98.4</td>
<td>1034</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>s13207</td>
<td>62</td>
<td>152</td>
<td>7951+638</td>
<td>97.7</td>
<td>77696</td>
<td>78832</td>
<td>NA</td>
<td>NA</td>
<td>92.4</td>
<td>9942</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>s15850</td>
<td>77</td>
<td>150</td>
<td>9782+534</td>
<td>98.8</td>
<td>96640</td>
<td>96413</td>
<td>NA</td>
<td>NA</td>
<td>90.6</td>
<td>9533</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>s38417</td>
<td>28</td>
<td>106</td>
<td>22179+1636</td>
<td>97.5</td>
<td>115882</td>
<td>116208</td>
<td>NA</td>
<td>NA</td>
<td>91.7</td>
<td>9601</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>s38584</td>
<td>38</td>
<td>304</td>
<td>19253+1426</td>
<td>99.3</td>
<td>79712</td>
<td>79360</td>
<td>NA</td>
<td>NA</td>
<td>94.1</td>
<td>9645</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

### TABLE 2

<table>
<thead>
<tr>
<th>Circuit</th>
<th>(P_{avg\ LFSR}) [(\mu)W]</th>
<th>(P_{peak\ LFSR}) [(\mu)W]</th>
<th>(P_{avg\ LT-LFSR}) [(\mu)W]</th>
<th>(P_{peak\ LT-LFSR}) [(\mu)W]</th>
</tr>
</thead>
<tbody>
<tr>
<td>c1908</td>
<td>5.7</td>
<td>26.7</td>
<td>1.4</td>
<td>15.8</td>
</tr>
<tr>
<td>c2670</td>
<td>26.4</td>
<td>103.2</td>
<td>6.1</td>
<td>66.3</td>
</tr>
<tr>
<td>c3540</td>
<td>12.9</td>
<td>69.6</td>
<td>4.2</td>
<td>40.7</td>
</tr>
<tr>
<td>c5315</td>
<td>38.8</td>
<td>219.9</td>
<td>11.3</td>
<td>137.2</td>
</tr>
<tr>
<td>s13207</td>
<td>745</td>
<td>4735</td>
<td>301</td>
<td>2917</td>
</tr>
<tr>
<td>s15850</td>
<td>783</td>
<td>5904</td>
<td>297</td>
<td>3591</td>
</tr>
<tr>
<td>s38417</td>
<td>1770</td>
<td>15394</td>
<td>792</td>
<td>8527</td>
</tr>
<tr>
<td>s38584</td>
<td>2466</td>
<td>19880</td>
<td>1051</td>
<td>10170</td>
</tr>
</tbody>
</table>
of the average and peak power, respectively. Compared to [23] and [4], our technique, in most cases, provides a larger reduction of the average and peak power.

6.4 Instantaneous Power

Instantaneous power (that is, power surge between two consecutive patterns) can put a lot of stress on circuits (for example, formation of hot spots) and, thus, is a matter of concern. Our LT-LFSR significantly lowers the chance of instantaneous power violations. Fig. 20 shows the instantaneous power waveform for the first 500 patterns applied using LFSR and LT-LFSR for c880 benchmark. The parameter $P_{thr}$ represents the instantaneous power limit set by the user. For this particular benchmark, $P_{thr} = 9.0 \mu W$. The test patterns generated by LT-LFSR cross this limit much less frequently than LFSR patterns. In this particular benchmark in the same simulation period, LT-LFSR patterns violate the power limit ($P_{thr}$) only 21 times, whereas LFSR patterns violate $P_{thr}$ 106 times. The more violations there are of such a limit, the more chances there are of damaging the circuit.

6.5 Power Consumption of LFSR and LT-LFSR

We also explored the power consumption of LFSR and LT-LFSR themselves used in the benchmarks. Table 4 compares the power consumption of the RI of LT-LFSR, including its FSM and LFSR. Depending on the size, the power consumption of LT-LFSR is 14-22 percent less than the same size of LFSR.

6.6 Area Overhead

As mentioned before, FSM can be a part of an on-chip BIST controller to control the test pattern generation process. The size of FSM is fixed, that is, 46 equivalent NAND gates. Table 5 shows the area increase for ISCAS benchmarks when they use LT-LFSR instead of LFSR. As seen in the table, using LT-LFSR, the overall test area overhead increases up to 13 percent. Compared to a conventional LFSR, test overhead is almost negligible, especially for large circuits such as s38417 and s38584.

7 CONCLUSION

This paper presents a new low-power LFSR to reduce the average and peak power of combinational and sequential circuits during the test mode. The switching activity in the CUT and scan chains and, eventually, their power consumption are reduced by increasing the correlation between patterns and also within each pattern. The experimental results indicate up to 77 percent and 49 percent reduction in the average and peak power, respectively, with test overhead less than 13 percent. This is

---

**Table 3**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>c1908</td>
<td>75.48</td>
<td>40.67</td>
<td>32.00</td>
</tr>
<tr>
<td>c2670</td>
<td>77.08</td>
<td>35.76</td>
<td>90.00</td>
</tr>
<tr>
<td>c3540</td>
<td>67.39</td>
<td>41.46</td>
<td>60.00</td>
</tr>
<tr>
<td>c5315</td>
<td>70.85</td>
<td>37.60</td>
<td>60.00</td>
</tr>
<tr>
<td>s13207</td>
<td>58.89</td>
<td>41.15</td>
<td>NA</td>
</tr>
<tr>
<td>s15850</td>
<td>62.21</td>
<td>39.18</td>
<td>NA</td>
</tr>
<tr>
<td>s38417</td>
<td>55.25</td>
<td>44.61</td>
<td>NA</td>
</tr>
<tr>
<td>s38584</td>
<td>57.38</td>
<td>48.84</td>
<td>NA</td>
</tr>
</tbody>
</table>

**Table 4**

<table>
<thead>
<tr>
<th>Circuit</th>
<th>LFSR Bitwidth</th>
<th>Power Consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[LFSR] [µW]</td>
<td>[LT-LFSR] [µW]</td>
</tr>
<tr>
<td>c2670</td>
<td>233</td>
<td>6.4</td>
</tr>
<tr>
<td>c3540</td>
<td>50</td>
<td>4.1</td>
</tr>
<tr>
<td>c5315</td>
<td>178</td>
<td>5.9</td>
</tr>
<tr>
<td>s13207</td>
<td>62</td>
<td>5.0</td>
</tr>
<tr>
<td>s38584</td>
<td>38</td>
<td>4.3</td>
</tr>
</tbody>
</table>

**Table 5**

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Test Overhead (LFSR)%</th>
<th>Test Overhead (LT-LFSR)</th>
</tr>
</thead>
<tbody>
<tr>
<td>c1908</td>
<td>9.2</td>
<td>11.0</td>
</tr>
<tr>
<td>c3540</td>
<td>10.7</td>
<td>12.3</td>
</tr>
<tr>
<td>s38417</td>
<td>0.7</td>
<td>0.8</td>
</tr>
<tr>
<td>s38584</td>
<td>0.8</td>
<td>0.9</td>
</tr>
</tbody>
</table>

Fig. 20. Instantaneous power in LFSR and LT-LFSR.
achieved with almost no increase in test length to hit a
target fault coverage. LT-LFSR significantly reduces the
achieved with almost no increase in test length to hit a
achieved with almost no increase in test length to hit a
achieved with almost no increase in test length to hit a
achieved with almost no increase in test length to hit a
achieved with almost no increase in test length to hit a
achieved with almost no increase in test length to hit a

ACKNOWLEDGMENTS

A preliminary version of this paper was published in the
Proceedings of the 14th IEEE Asian Test Symposium (ATS '05).

REFERENCES

[1] Y. Zorian, “A Distributed BIST Control Scheme for Complex VLSI
[3] F. Corno, M. Rebaudengo, M. Reorda, G. Squillero, and M.
Violante, “Low Power BIST via Non-Linear Hybrid Cellular
Wunderlich, “A Modified Clock Scheme for a Low Power BIST
State Correlation Analysis," Proc. VLSI Test Symp., pp. 267-272,
2003.
Partitioning: Methodology and Architectures,” Proc. Int’l Test
and Power-Constrained Test Scheduling for System-on-a-Chip,”
Scheduling Environment,” IEEE Trans. Computers, vol. 54, no. 12,
Figueras, S. Manich, P. Teixeira, and M. Santos, “Low Energy
BIST Design: Impact of the LFSR TPG Parameters on the Weighted
Test Vector Inhibiting Technique for Low Power BIST Design,”
C. Landrault, S. Pravossoudovitch, P. Teixeira, and M. Santos,
European Test Workshop, pp. 165-170, 1999.
Consumption for Scan-Based BIST,” Proc. Int’l Test Conf., pp. 77-
84, 1999.
BIST with Smoother and Scan-Chain Reorder,” Proc. IEEE Asian
Power Testing by Test Vector Ordering with Vector Repetition,”
Based on LFSR Reseeding,” Proc. IEEE Int’l Conf. Computer Design,
Consecutive Test Vectors in a Random Sequence for Energy-
2005.
with Applications to Datapath Synthesis,” Proc. Design Automation

Mehrdad Nourani received the BSc and MSc
degrees in electrical engineering from the
University of Tehran, Iran, in 1986 and the PhD
degree in computer engineering from Case
Western Reserve University, Cleveland, Ohio,
in 1993. In 1994, he was a postdoctoral fellow in
the Department of Computer Engineering at
Case Western Reserve University. He served in
the Department of Electrical and Computer
Engineering at the University of Tehran from
1995 to 1998 and the Department of Electrical Engineering and
Computer Science at Case Western Reserve University from 1998 to
1999. Since August 1999, he has been on the faculty of the University of
Texas at Dallas, where he is currently an associate professor of
electrical engineering and a member of the Center for Integrated Circuits
and Systems (CICS). His current research interests include design for
testability, system-on-chip testing, signal integrity modeling and test,
application specific processor architectures, packet processing devices,
high-level synthesis, and low-power design methodologies. He has
published more than 120 papers in journals and refereed conference
proceedings. He is a senior member of the IEEE and a member of the
IEEE Computer Society and ACM SIGDA. He received the Texas
Telecommunications Consortium Award in 1999, the Clark Foundation
Research Initiation Grant in 2001, the US National Science Foundation
Career Award in 2002, and the Cisco Systems Inc. URP Award in 2004.
He received a best paper award at the 2004 International Conference on
Computer Design.
Mohammad Tehranipoor received the BSc degree from the Amirkabir University of Technology (Tehran Polytechnic University), Iran, the MSc degree from the University of Tehran, and the PhD degree from the University of Texas at Dallas, in 1997, 2000, and 2004, respectively, all in electrical engineering. He is currently an assistant professor of electrical and computer engineering at the University of Connecticut. He was a faculty member in the Computer Science and Electrical Engineering Department at the University of Maryland Baltimore County (UMBC) from 2004 to 2006. His current research interests include computer-aided design and test for CMOS VLSI designs and emerging nanoscale devices, design-for-testability, at-speed test, secure design, and IC trust. He has published more than 70 journal articles and refereed conference papers in the area of VLSI design and test. He has published two books, Nanometer Technology Designs—High-Quality Delay Tests and Emerging Nanotechnologies—Test, Defect Tolerance and Reliability, in addition to two book chapters. He was a recipient of a best paper award at the 2005 VLSI Test Symposium (VTS), a best paper candidate at the Design Automation Conference (DAC) in 2006, a recipient of a best panel award at VTS ’06, and a recipient of top-10 paper recognition at the 2005 International Test Conference (ITC). He has served on the program committees of several workshops and conferences in addition to serving as session chair for many technical events. He has served as a guest editor of the Journal of Electronic Testing: Theory and Applications (JETTA) on “test and defect tolerance for nanoscale devices” and IEEE Design and Test of Computers on “IR-drop and power supply noise effects on very deep-submicron designs.” He has also served as the program chair of the 2007 IEEE Defect-Based Testing (DBT) Workshop. He will also be serving as co-program chair of the 2008 International Defect and Fault Tolerance in VLSI Systems (DFT). He serves as an associate editor of JETTA. He is a member of the IEEE, the IEEE Computer Society, the ACM, and ACM SIGDA.

Nisar Ahmed received the BE degree in electronics and communication engineering from Osmania University, India, and the MS degree in electrical engineering from the University of Texas at Dallas. He is currently working toward the PhD degree in electrical and computer engineering at the University of Connecticut. His current research interests include computer-aided design and test, at-speed low-cost test, and IR-drop effects on delay test. He is a student member of the IEEE and the IEEE Computer Society.