A Scan Partitioning Algorithm for Reducing Capture Power of Delay-Fault LBIST

Nan Li
Royal Institute of Technology
Stockholm, Sweden
nan3@kth.se

Elena Dubrova
Royal Institute of Technology
Stockholm, Sweden
dubrova@kth.se

Gunnar Carlsson
Development Unit Radio, Ericsson AB
Stockholm, Sweden
gunnar.carlsson@ericsson.com

***Abstract*—It is well-known that high power consumption in test mode can cause problems such as overheating and IR-drop which have negative effect on circuit reliability and yield. The problem is particularly hard in the case of at-speed delay-fault testing where it cannot be mitigated by lowering the clock frequency. The difficulty increases even further if pseudo-random rather than ATPG patterns are used for testing. ATPG patterns can be chosen selectively, as well as re-ordered and specified in a power-friendly manner. Pseudo-random test patterns are much harder to control. In this paper, we present a scan partitioning algorithm for reducing capture power targeting delay-fault LBIST. The algorithm uses a novel weighted S-graph model in which the weights are determined by signal probability analysis. Our experimental results show that, on average, the presented method reduces average capture power by 50% and peak capture power by 39% with less than 2% loss in the transition fault coverage.**

I. Introduction

It is known that the power consumption of a circuit in test mode is much higher than in normal operational modes, due to excessive switching activity during test application [1]. This results in a series of problem including circuit overheating and IR-drop, which in turn decreases reliability and yield [2]. For delay fault testing, this problem is particularly hard since it cannot be mitigated by lowering the clock frequency. *Launch-off-Capture* (LoC, also known as *broadside testing*) is a widely used method for testing transition faults, the most common delay fault model [3]. In LoC, after a test pattern is shifted into the scan flip-flops, two consecutive capture clock pulses are applied with the interval equal to the target clock frequency.

A commonly used method for reducing the average and peak power consumption in Automatic Test Pattern Generation (ATPG) based LoC testing is *scan partitioning*, where only a subset of scan chains is activated in each shift or capture clock cycle [4]. While scan partitioning introduces no penalty for shift power reduction besides the extended test application time, for capture power reduction, it comes at a cost of fault coverage drop, due to capture violation [5]. Capture violation is caused by the dependency among flip-flops from different scan chain groups. Therefore, the key problem is to find a partitioning of scan flip-flops so that the overall dependency among different partitions is minimized.

The problem of excessive test power consumption becomes even more difficult in Logic Built-In Self-Test (LBIST), which uses pseudo-random test patterns. First, due to the random nature of the test patterns, the switching activity of LBIST is even higher than the one of ATPG-based testing [6]. Second, in ATPG, it is possible to select and specify the test patterns in a way that power consumption in test mode is reduced. There is much less flexibility to control test patterns in LBIST, where they are generated by a pseudo-random pattern generator.

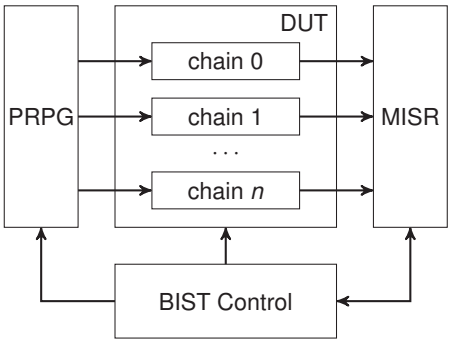

Fig. 1. A typical LBIST architecture.

In this paper, we present a scan partitioning algorithm targeting capture power reduction in LBIST for LoC testing. A novel weighted S-graph model is used to partition scan chains, where the weights are determined by signal probability analysis. The presented method is referred to as Signal Probability Enhanced Partitioning (SPEP).

The rest of the paper is organized as follows. Section II explains fundamental concepts and definitions. Section III introduces related work. Section IV presents our scan partitioning algorithm. Section V shows experimental results. Section VI concludes the paper and discusses future work.

II. Background

A. Logic Built-In Self-Test

Logic Built-In Self-Test (LBIST) is used in manufacturing and in-field test of integrated circuits [7]. Instead of using an external Automatic Test Equipment (ATE) to apply test patterns and to analyze test response, LBIST uses an on-chip Pseudo-Random Pattern Generator (PRPG) and an on-chip Multiple-Input Signature Register (MISR), both controlled by an on-chip BIST controller. A typical LBIST architecture in STUMPS configuration [8] consists of a Linear Feedback Shift

978-3-9815370-4-8/DATE15/©2015 EDAA

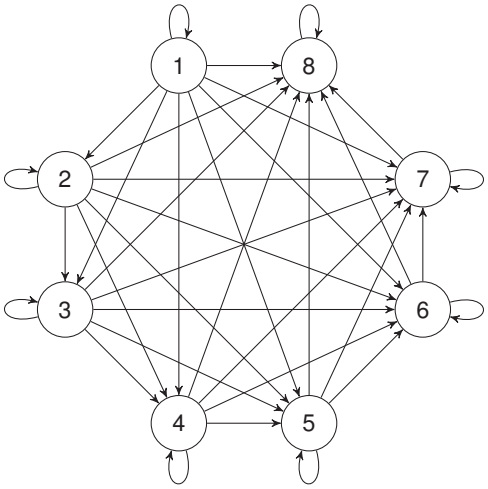

Fig. 2. S-graph for ISCAS'89 benchmark s208.

Register (LFSR) as the PRPG, a MISR, and a BIST control block which generates test clocks and control signals, as shown in Figure 1. The design under test is assumed to be scan-ready.

B. S-graph

S-graphs [9] are widely used in designing of scan structures, such as partial scan [9], [10] design and scan partitioning [4], [5], [11], [12].

Definition 1: An *S-graph* of a sequential circuit is a directed graph $G = (V, E)$, where the vertex set V is the set of flip-flops in the circuit, and edge $\langle u, v\rangle \in E$ iff there is a combinational path from an output of flip-flop u to an input of flip-flop v.

Figure 2 shows an example S-graph of an ISCAS'89 benchmark s208.

In scan partitioning, we try to minimize the dependency between each pair of partitions. This problem is modeled with an S-graph partitioning problem with minimum cut and balanced constraint on the size of each partition. Assuming that we apply capture clocks in the natural order (i.e., partition $1, 2, \cdots, n$), then a Violation Edge (VE) is defined as an edge in the S-graph that points from a partition with lower index to a partition with higher index. An S-graph partitioning with one or more VEs can potentially cause a capture violation during test. The problem of S-graph partitioning can then be formulated as follows. Given an S-graph $G = (V, E)$ and integer n, $n > 1$, find an n-partition of G, $\{G_1, G_2, \cdots, G_n\}$, so that the size of the VE-set

$$\{\langle u, v\rangle \in E | u \in G_i, v \in G_j, \text{and } i < j\}$$

is minimized.

III. Related Work

Many efforts have been made to reduce capture power in ATPG-based testing and in LBIST.

There are two types of methods that can be used for ATPG-based testing. The first type reduces capture power by making test patterns power-friendly [13], [14]. These methods utilize the large portion of don't-care bits (X-bits) in ATPG patterns.

The second type introduces new Design-For-Test (DFT) strategies, such as scan partitioning (also called segmented scan) [4], [5], [11], [12], [15]. With these techniques, only a subset of scan cells are activated in shift and capture cycles, so the power consumption is reduced by a factor that is positively correlated to the number of partitions. The problem is modeled as minimization of the number of VEs in the S-graph in these techniques. As a trade-off for reducing power consumption, the design will suffer from extended test application time, lower fault coverage, and potential area overhead.

For LBIST, it is less efficient to directly control test patterns. Therefore, reducing power consumption in LBIST normally involves modification of LBIST structures. In [16], gating logic is used to partially mask scan cell activities. A low power BIST scheme based on circuit partitioning is presented in [17]. A substitute for an LFSR is introduced in [18] to generate power-aware pseudo-random patterns. In [19], a scan partitioning method based on controllability/observability analysis is presented. However, all of the above methods that reduce capture power for LBIST target stuck-at fault model. Their applicability and efficiency for delay fault model is unknown.

IV. Signal Probability Enhanced Partitioning

A. Motivation

An S-graph contains simplified dependency information among the flip-flops. It shows if the input of a flip-flop depends on the output of another flip-flop or not. However, it does not reflect the degree of dependency.

As an example, consider two flip-flops a and b with an inverter connecting them in serial

$$b = \text{NOT}(a).$$

In this case, a change in flip-flop a causes a change in b with 100% probability. Now if we use an AND gate instead of the inverter

$$b = \text{AND}(a, c),$$

then, b follows the change of a only if the other input of the AND gate, c, is 1. If c is not a constant 1, the probability that b will change when a changes is less than 100%. We can say that flip-flop b is less dependent of flip-flop a than in the first case.

In the context of scan partitioning, the *dependency* of a flip-flop a to a flip-flop b can be defined as the probability that the input of b will change when the output of a changes.

To illustrate how much the dependency values vary for the same circuit, we plotted the dependency among flip-flops in ISCAS'89 benchmark s38584, as shown in Figure 3. We can see that, out of 17,973 edges in the S-graph, only 7,037 edges represent a dependency greater than 1%. All the remaining edges represent less than 1% probability that a flip-flop is dependent of another flip-flop. Yet in the traditional S-graph,

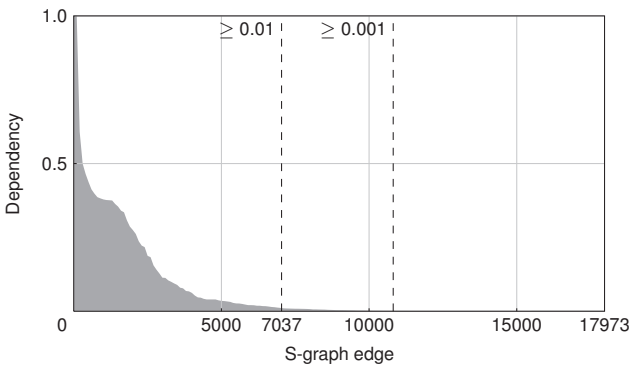

Fig. 3. Dependency among flip-flops of benchmark s38584.

TABLE I
SIGNAL PROBABILITY EXPRESSIONS FOR COMMON LOGIC GATES.

Type of g	Inputs	Signal probability $p(g)$
NOT	a	$1-p(a)$
AND	a, b	$p(a)p(b)$
OR	a, b	$1-(1-p(a))(1-p(b))$
NAND	a, b	$1-p(a)p(b)$
NOR	a, b	$(1-p(a))(1-p(b))$
XOR	a, b	$(1-p(a))p(b)+p(a)(1-p(b))$
XNOR	a, b	$p(a)p(b)+(1-p(a))(1-p(b))$

they are considered equally significant to the more dependent connections.

Compared to pseudo-random patterns used by LBIST, the test patterns generated by Automatic Test Pattern Generation (ATPG) algorithms may be biased (e.g., contain a lot of zeros). Therefore, an analysis based purely on a circuit structure does not provide accurate dependency information for ATPG-based testing. However, for LBIST, due to the randomness of the test patterns, a probabilistic analysis is sufficient to provide near-accurate information on the dependency.

B. Signal Probability

Probabilistic models have been widely used in testability analysis [20], [21] and fault coverage estimation [22], [23].

The *signal probability* of a node a is a real number within the interval $[0, 1]$ reflecting the probability that the signal value of a equals one [24]. For a combinational logic block, given a set of signal probabilities of the inputs, we can derive the signal probability of each node in the block. Assuming independent inputs, the signal probabilities for some common logic gates are shown in Table I.

Due to the presence of re-converging paths, input signals of a logic gate may not be independent, therefore having correlated signal probabilities. Deriving the signal probability for a circuit with re-converging paths is an NP-hard problem [21]. Several heuristics exist for estimation of signal probability [25]–[28]. The most straight-forward estimation is to simply ignore the correlation of the signals [25]. This results in a fast, linear algorithm. According to an evaluation in [21], it is generally not worth paying the price for methods with higher complexity as they do not generate results that are good enough. In the presented work, we adopt this simple approach for signal probability estimation.

Algorithm 1 `CalcSigProb`(G, F): Calculate signal probability for internal nodes, ignoring signal correlation. Return the signal probability for each node in $G \cup F$. g^i denotes the ith input node of g.

```
1:  for all f ∈ F do
2:      p(f) ← 0.5
3:  end for
4:  for all g ∈ G in topological order do
5:      if g is a NOT gate then
6:          p(g) ← 1 − p(g^1)
7:      end if
8:      if g is an n-input AND gate then
9:          p(g) ← ∏_{i=1}^{n} p(g^i)
10:     end if
11:     if g is an n-input OR gate then
12:         p(g) ← 1 − ∏_{i=1}^{n} (1 − p(g^i))
13:     end if
14:     if g is an n-input NAND gate then
15:         p(g) ← 1 − ∏_{i=1}^{n} p(g^i)
16:     end if
17:     if g is an n-input NOR gate then
18:         p(g) ← ∏_{i=1}^{n} (1 − p(g^i))
19:     end if
20: end for
21: return p
```

The pseudo-code for the signal probability calculation algorithm is shown in Algorithm 1. In the description, G denotes the set of all logic gates; F denotes the set of all flip-flops. The return value p is a mapping of type $G \cup F \rightarrow [0, 1]$, where $p(n)$ is the estimated signal probability of node n. First, we assign a 0.5 signal probability to the output node of all flip-flops. Then, with a single traversal of the logic gates in topological order we assigns signal probability for each gate. We implement calculation for five types of gates in the algorithm, because these are the gates used in ISCAS'89 benchmarks. More gates may be easily added into the algorithm if other gate libraries are used.

C. Dependency Estimation

After assigning signal probabilities for nodes in the circuit, we estimate the dependency between each pair of flip-flops.

The *dependency* of nodes b on node a, denoted $d_a(b)$ is defined as a probability that the change in the value of a causes the change in the value of b. For any logic gate g, given the dependency of each of its inputs on some source node a and the signal probability of its inputs and output, we can estimate the dependency of g on a. Let $s_i(g)$ denote the probability that g is sensitive to its ith input. The dependency of g on a

TABLE II
SENSITIVITY CALCULATION FOR COMMON LOGIC GATES.

Type of g	Inputs	$s_a(g)$	$s_b(g)$
NOT	a	1	-
AND	a, b	$p(b)$	$p(a)$
OR	a, b	$(1-p(b))$	$(1-p(a))$
NAND	a, b	$p(b)$	$p(a)$
NOR	a, b	$(1-p(b))$	$(1-p(a))$
XOR	a, b	1	1
XNOR	a, b	1	1

is estimated by

$$d_a(g) = 1 - \prod_{i=1}^{n} (1 - d_a(g^i) \cdot s_i(g)),$$

where n is the number of g's inputs, and g^i is the ith input of g. This estimation ignores the case that multiple inputs change at the same time.

The sensitivity $s_i(g)$ is the probability that the change of ith input causes the change of the output of g, and it purely depends on the signal probability of the other inputs. Table II shows the calculation of $s_i(g)$ for different types of gates.

The algorithm of dependency estimation is shown in Algorithm 2. First, choose a source flip-flop f. Set the dependency value of f to be 1, other flip-flops to be 0. Then, for each logic gate g in topological order, derive the dependency values $d_f(g)$. The function `Sens` returns the sensitivity $s_i(g)$ that is used in the calculation of $d_f(g)$. If the output of g is the input of some flip-flop f_{dst}, then $d_f(g)$ is stored in the dependency matrix m as the dependency of flip-flop f_{dst} on flip-flop f. Go back to the first step until all flip-flops have been chosen to be the source flip-flop. Now the dependency between any pair of flip-flops (f_1, f_2) is stored in the dependency matrix m as $m(f_1, f_2)$ and $m(f_2, f_1)$.

Algorithm 2 `CalcDep`(G, F, p): Estimate dependency between each pair of flip-flops. Return the dependency matrix m. g^i denotes the ith input node of g.

```
for all f ∈ F do
    d(f) ← 0
end for
for all f ∈ F do
    d(f) ← 1
    for all g ∈ G in topological order do
        n ← number of inputs of g
        d(g) ← 1 − ∏_{i=1}^{n} (1 − d(g^i) · Sens(g, i, p))
    end for
    for all f_dst ∈ F do
        m(f, f_dst) ← d(input of f_dst)
    end for
    d(f) ← 0
end for
return m
```

Algorithm 3 `Sens`(g, i, p): Return the probability that the output of gate g is sensitive to the ith input of g, given the signal probability. g^i denotes the ith input node of g.

```
if g is a NOT gate then
    return 1
end if
if g is an AND gate then
    return p(g)/p(g^i)
end if
if g is an OR gate then
    return (1 − p(g))/(1 − p(g^i))
end if
if g is a NAND gate then
    return (1 − p(g))/p(g^i)
end if
if g is a NOR gate then
    return p(g)/(1 − p(g^i))
end if
```

D. Weighted S-Graph

A traditional S-graph is an unweighted graph. An edge in an S-graph indicates there is a logical path between its terminals. In the presented SPEP method, we extend this model by assigning a weight to each edge that is equal to the dependency between the the two flip-flops. The new model retain more information of the circuit, yet it can still be efficiently handled by existing graph partitioning methods. Therefore, it is expected to provide a better partitioning compared to traditional S-graphs. Additionally, for extremely large circuits, or when using time-consuming graph partitioning algorithms, with the new model it is possible to remove edges with very low weights, so the problem complexity decreases.

Partitioning of a weighted S-graph is accomplished using the multi-level k-way partitioning algorithm [29]. The algorithm consists of three phases. First, the closely related vertices are merged into a single vertex. This is done multiple times so the resulting graph is sufficiently small (typically less than 100 vertices). The second phase involves partitioning the coarsened graph using a greedy algorithm [30]. The partitions are balanced to a certain degree, since the algorithm keeps track of the number of vertices in each partition. Finally, the partitioned graph is projected back to the original graph in the reverse order as it is coarsened, with balancing and refinement in each step. The authors of [29] implemented the algorithm in the publicly available software METIS.

V. EXPERIMENTAL RESULTS

A. Comparison with Minimum VE Partitioning

The presented SPEP algorithm is compared with the minimum VE partitioning method, which minimizes the number of violation edges. We applied both algorithms on eight largest circuits from ISCAS'89 benchmark set [31]. For each circuit, the flip-flops are split into two partitions using both algorithms.

TABLE III
CAPTURE POWER REDUCTION, FAULT COVERAGE DROP, AND RUN TIME COMPARISON. TWO SCAN PARTITIONS.

Circuit	# FFs	# Patterns	Minimum VE partitioning				SPEP (presented)			
			ACPR[1] (%)	PCPR[1] (%)	DCOV[2] (%)	CPU time[3] (s)	ACPR[1] (%)	PCPR[1] (%)	DCOV[2] (%)	CPU time[3] (s)
s1423	74	100K	46.06	12.57	5.21	1.39	51.14	36.15	5.02	1.27
s5378	179	100K	49.72	33.52	2.15	1.04	49.06	40.16	0.93	0.99
s9234	228	100K	50.34	27.03	2.69	1.07	48.56	26.12	0.88	0.94
s13207	669	100K	52.08	37.06	2.94	1.58	49.68	37.06	3.28	1.55
s15850	597	100K	51.67	31.89	6.70	2.13	49.79	36.70	1.66	1.36
s35932	1728	100K	49.84	46.76	0.28	2.97	50.05	47.41	0.31	2.76
s38417	1636	100K	50.89	36.15	0.77	4.88	55.49	45.43	2.07	5.08
s38584	1452	100K	50.17	43.77	0.18	4.17	49.92	41.79	0.37	4.17
Average			50.10	33.59	2.62		50.46	38.85	1.82	

[1] ACPR/PCPR: average/peak capture power reduction. [2] DCOV: decrease in fault coverage. [3] CPU time does not include simulation time.

Then, 100,000 pseudo-random LFSR patterns are simulated for the transition fault coverage and average/peak capture power. The simulation is performed using an in-house cycle based simulator. Weighted Switching Activity (WSA) is used as an estimation of the power consumption. The WSA of a node is calculated as the number of switchings on the node multiplied by the number of fan-outs of the node plus one [16].

All experiments are carried out on a PC with Intel i5 3.20 GHz quad-core CPU and 4GB RAM.

The results are shown in Table III. The columns "ACPR" and "PCPR" show average and peak capture power reduction in percentage, respectively. The columns "DCOV" show the fault coverage drop caused by scan partitioning. The columns "CPU time" show the run time of the scan partitioning algorithm. It does not include the run time of the simulation.

As we can see, both methods reduce average capture power by $\sim$50%. This is expected since only one half of the scan cells are activated at each capture cycle. The peak capture power reduction is lower because of the randomness of the patterns. Compared to minimum VE partitioning, the presented SPEP method achieves 0.36% larger reduction in average capture power, and a rather significant 5.26% larger reduction in peak capture power. Since peak power has a stronger impact on IR-drop, it is preferable to maximally reduce the peak capture power.

As a price to pay for the lower capture power, the fault coverage decreases because the partitioning potentially makes some faults more difficult, or even impossible, to detect. Applying the SPEP method causes 1.82% fault coverage drop, which is 0.8% less than the minimum VE partitioning.

Run time is another interesting parameter to compare. Since we used the same back-end graph partitioning algorithm and the minimum VE partitioning method uses simpler unweighted S-graphs, it is reasonable to expect SPEP method to be slower. However, as we can see from Table III, for many benchmarks the SPEP algorithm runs faster than the minimum VE partitioning. A possible reason for this might be that assigning weights to the edges of an S-graph reduces the decision complexity of the graph partitioning algorithm.

B. Power Reduction Comparison with Other Low-Power Testing Techniques

In Table IV, we compare the presented method with other test power reduction techniques [5], [12], [17]. The method in [17] works on LBIST, but it targets only stuck-at faults, and the paper presents results for overall (shift and capture) power reduction. The methods in [12] and [5] reduce the capture power in transition fault testing, but they use ATPG patterns. We understand that the results are not strictly comparable; they are only intended to show that our method reduces power at the same level as other test power reduction techniques.

VI. CONCLUSION AND FUTURE WORK

We presented a novel scan partitioning method targeting the reduction of capture power for delay fault testing in LBIST. We extended the existing partitioning methods based on S-graphs by assigning the dependency value to the edges. The presented method is able to achieve 50% reduction in average capture power, and 39% reduction in peak capture power, with less than 2% loss in transition fault coverage.

Future work includes developing a more accurate model considering geometrical proximity of scan cells and power planning.

REFERENCES

[1] Y. Zorian, "A distributed BIST control scheme for complex VLSI devices," in *VLSI Test Symposium, 1993. Digest of Papers., Eleventh Annual 1993 IEEE*, pp. 4–9, April 1993.

[2] D. Lee, E. Volkerink, I. Park, and J. Rearick, "Empirical validation of yield recovery using idle-cycle insertion," *Design Test of Computers, IEEE*, vol. 24, pp. 362–372, July 2007.

[3] J. Savir and S. Patil, "Broad-side delay test," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 13, pp. 1057–1064, Aug 1994.

[4] P. Rosinger, B. Al-Hashimi, and N. Nicolici, "Scan architecture for shift and capture cycle power reduction," in *Defect and Fault Tolerance in VLSI Systems, 2002. DFT 2002. Proceedings. 17th IEEE International Symposium on*, pp. 129–137, 2002.

TABLE IV
POWER REDUCTION EFFECTIVENESS COMPARED TO OTHER LOW-POWER TECHNIQUES.

Circuit	Results of [17] (BIST, shift + capture)		Results of [12] (ATPG, capture)		Results of [5] (ATPG, capture)		SPEP (presented) (BIST, capture)	
	APR[1] (%)	PPR[1] (%)	APR[1] (%)	PPR[1] (%)	APR[1] (%)	PPR[1] (%)	APR[1] (%)	PPR[1] (%)
s1423	50.7	31.9	-	-	-	-	51.14	36.15
s5378	-	-	34.71	25.59	-	-	49.06	40.16
s9234	49.5	33.8	7.20	24.02	47.56	43.01	48.56	26.12
s13207	46.3	40.5	24.44	29.38	48.38	44.13	49.68	37.06
s15850	48.7	40.3	10.26	15.95	42.06	49.46	49.79	36.70
s35932	-	-	19.66	17.12	-	-	50.05	47.41
s38417	60.2	55.8	15.87	24.02	48.88	44.64	55.49	45.43
s38584	46.0	27.6	17.62	18.35	49.14	46.75	49.92	41.79
Average	50.23	38.32	18.54	22.06	47.20	45.60	50.46	38.85

[1] APR/PPR: average/peak power reduction.

[5] Z. Chen, K. Chakrabarty, and D. Xiang, "MVP: Minimum-violations partitioning for reducing capture power in at-speed delay-fault testing," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 30, pp. 1762–1767, Nov 2011.

[6] S. Wang and S. Gupta, "DS-LFSR: a new BIST TPG for low heat dissipation," in *Test Conference, 1997. Proceedings., International*, pp. 848–857, Nov 1997.

[7] E. McCluskey, "Built-in self-test techniques," *IEEE Design and Test of Computers*, vol. 2, pp. 21–28, 1985.

[8] P. H. Bardell, W. H. McAnney, and J. Savir, *Built-in Test for VLSI: Pseudorandom Techniques*. New York, NY, USA: Wiley-Interscience, 1987.

[9] K.-T. Cheng and V. Agrawal, "A partial scan method for sequential circuits with feedback," *Computers, IEEE Transactions on*, vol. 39, pp. 544–548, Apr 1990.

[10] D. Lee and S. Reddy, "On determining scan flip-flops in partial-scan designs," in *Computer-Aided Design, 1990. ICCAD-90. Digest of Technical Papers., 1990 IEEE International Conference on*, pp. 322–325, Nov 1990.

[11] H. Ko and N. Nicolici, "Automated scan chain division for reducing shift and capture power during broadside at-speed test," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 27, pp. 2092–2097, Nov 2008.

[12] Z. Zhang, S. Reddy, I. Pomeranz, J. Rajski, and B. Al-Hashimi, "Enhancing delay fault coverage through low power segmented scan," in *Test Symposium, 2006. ETS '06. Eleventh IEEE European*, pp. 21–28, May 2006.

[13] X. Wen, Y. Yamashita, S. Morishima, S. Kajihara, L.-T. Wang, K. Saluja, and K. Kinoshita, "Low-capture-power test generation for scan-based at-speed testing," in *Test Conference, 2005. Proceedings. ITC 2005. IEEE International*, pp. 10 pp.–1028, Nov 2005.

[14] S. Remersaro, X. Lin, Z. Zhang, S. Reddy, I. Pomeranz, and J. Rajski, "Preferred fill: A scalable method to reduce capture power for scan based designs," in *Test Conference, 2006. ITC '06. IEEE International*, pp. 1–10, Oct 2006.

[15] K.-J. Lee, S.-J. Hsu, and C.-M. Ho, "Test power reduction with multiple capture orders," in *Test Symposium, 2004. 13th Asian*, pp. 26–31, Nov 2004.

[16] S. Gerstendrfer and H.-J. Wunderlich, "Minimized power consumption for scan-based BIST," *Journal of Electronic Testing*, vol. 16, no. 3, pp. 203–212, 2000.

[17] P. Girard, L. Guiller, C. Landrault, and S. Pravossoudovitch, "Low power BIST design by hypergraph partitioning: methodology and architectures," in *Test Conference, 2000. Proceedings. International*, pp. 652–661, 2000.

[18] N. Ahmed, M. Tehranipour, and M. Nourani, "Low power pattern generation for BIST architecture," in *Circuits and Systems, 2004. ISCAS '04. Proceedings of the 2004 International Symposium on*, vol. 2, pp. II–689–92 Vol.2, May 2004.

[19] N. Basturkmen, S. Reddy, and I. Pomeranz, "A low power pseudo-random BIST technique," in *Computer Design: VLSI in Computers and Processors, 2002. Proceedings. 2002 IEEE International Conference on*, pp. 468–473, 2002.

[20] H.-J. Wunderlich, "PROTEST: A tool for probabilistic testability analysis," in *Proceedings of the 22Nd ACM/IEEE Design Automation Conference*, DAC '85, (Piscataway, NJ, USA), pp. 204–211, IEEE Press, 1985.

[21] P. Camurati, P. Prinetto, and M. Reorda, "Random testability analysis: comparing and evaluating existing approaches," in *Computer Design: VLSI in Computers and Processors, 1988. ICCD '88., Proceedings of the 1988 IEEE International Conference on*, pp. 70–73, Oct 1988.

[22] E. McCluskey, S. Makar, S. Mourad, and K. Wagner, "Probability models for pseudorandom test sequences," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 7, pp. 68–74, Jan 1988.

[23] W. Huang, M. Lightner, and F. Lombardi, "Predicting fault coverage for random testing of combinational circuits," in *Proc. IEEE Int. Test Conf*, pp. 843–848, 1987.

[24] K. Parker and E. McCluskey, "Probabilistic treatment of general combinational networks," *Computers, IEEE Transactions on*, vol. C-24, pp. 668–670, June 1975.

[25] F. Brglez, P. Pownall, and R. Hum, "Applications of testability analysis: From ATPG to critical delay path tracing," in *Proceedings of the 1984 International Test Conference on The Three Faces of Test: Design, Characterization, Production*, ITC'84, (Washington, DC, USA), pp. 705–712, IEEE Computer Society, 1984.

[26] J. Savir, G. S. Ditlow, and P. H. Bardell, "Random pattern testability," *IEEE Transactions on Computers*, vol. C-33, pp. 79 – 90, Jan. 1984.

[27] B. Krishnamurthy and I. Tollis, "Improved techniques for estimating signal probabilities," *Computers, IEEE Transactions on*, vol. 38, pp. 1041–1045, Jul 1989.

[28] S. Ercolani, M. Favalli, M. Damiani, P. Olivo, and B. Ricco, "Estimate of signal probability in combinational logic networks," in *European Test Conference, 1989., Proceedings of the 1st*, pp. 132–138, Apr 1989.

[29] G. Karypis and V. Kumar, "Parallel multilevel series k-way partitioning scheme for irregular graphs," *SIAM Review*, vol. 41, no. 2, pp. 278–300, 1999.

[30] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 359–392, 1998.

[31] F. Brglez, D. Bryan, and K. Kozminski, "Combinational profiles of sequential benchmark circuits," in *Circuits and Systems, 1989., IEEE International Symposium on*, pp. 1929–1934 vol.3, May 1989.