Fault Models and Test Generation for IDDQ Testing

Yoshinobu Higami and Yuzo Takamatsu
Ehime University
Matsuyama, Japan
{higami, takamatsu}@cs.ehime-u.ac.jp

Kewal K. Saluja
University of Wisconsin
Madison, U.S.A.
saluja@engr.wisc.edu

Kozo Kinoshita
Osaka University
Osaka, Japan
kozo@ap.eng.osaka-u.ac.jp

Abstract— This paper surveys recent research related to IDDQ testing, particularly focuses on fault models and test generation methods.

(1) The paper provides a taxonomy of fault models that have been studied in literature, and classifies these models into a small set of faults.

(2) The paper describes efficient test generation methods and fault simulation methods. Test compaction methods, including reduction of the total number of test vectors and selection of IDDQ measurement vectors, are also described.

I. Introduction

Logic testing has been widely used in industrial testing environment for many years. However, it has become evident that for high complexity and high density digital circuits, many physical defects cannot be detected by logic testing alone. Measuring quiescent current, IDDQ testing, is now considered an important part of testing CMOS circuits, because it can detect physical defects which do not change output function but induce a large amount of quiescent current to flow in the presence of a fault in the circuit. The principal of IDDQ testing is based on the fact that in a fault-free CMOS circuit the amount of quiescent power supply current is negligible. Thus, if a physical defect induces a large amount of current, then such a defect can be detected by monitoring the current. Thus IDDQ testing technique offers a global observation point for a circuit under test. Another advantage of IDDQ testing is that the IDDQ-based test generation is easier than logic testing. Since IDDQ measurement point is a global observer, in IDDQ test generation, unlike logic testing, the propagation of a fault effect is not necessary.

Researchers have studied several problems relating to IDDQ testing, such as fault simulation, test generation and test compaction. Many papers have appeared in the journals dealing with IDDQ testing (see list of references at the end of this paper) including a special issue on IDDQ testing by Journal of Electronic Testing: Theory and Applications (JETTA) in 1992, a collection of papers on bridging faults and IDDQ testing by Malaiya and Rajsuman [24], and books on IDDQ testing by Rajsuman [27] and by Charkravarty and Thadikaran [7]. Actually, there are lots of approaches proposed to various problems. Thus this paper describe problems discussed in recent papers and presents directions and solutions for problems on IDDQ testing.

The paper contains the following topics.

1. Fault model: We describe stuck-at, open and bridging fault models. From the point of view of fault location, fault behavior, fault equivalence and containment relation, a taxonomy of each fault model is provided. Moreover fault extraction is discussed, because it is an inevitable problem for test generation.

2. Test generation: At first, we describe fault simulation methods which are able to deal with a large number of bridging faults. Test generation methods for intra-gate shorts and bridging faults for combinational and sequential circuits are provided. Recent researches are tabulated and efficient techniques for various purposes are described. Additionally we describe test compaction methods that reduce the total number of test vectors and the number of IDDQ measurement vectors.

This paper is organized as follows. In Sections II, we provide a taxonomy of fault models and summarize the fault extraction issues that are used to keep the problem tractable. In Section III, we describe fault simulation, test generation and test compaction methods. The conclusions of this paper are stated in Section IV.

II. Fault model

A. Stuck-at faults

In IDDQ testing, as in logic testing, stuck-at faults (SAF) have been often considered [1, 11, 20, 33]. Kondo and Cheng have classified SAFs into the following three types based on the logical behavior of the fault [20].

- Leakage stuck-at fault (LSAF): A faulty line has a same value as fault-free value, if the fault is activated.
- Pseudo stuck-at fault (PSAF): A faulty line has a stuck-at value, if the fault is activated.
- Generalize stuck-at fault (GSAF): A faulty line has an unknown value, if the fault is activated.

TABLE I
CLASSIFICATION OF BRIDGING FAULTS

	<SL, SL>	<SL, IN>	<IN, IN>	<SL∨IN, SL∨IN>
a. Same transistor	-	-	0. LE	-
b. Same gate	1. ITRA-BF	2. ITRA-SSI	3. ITRA-ISH	10. ITRA-SH
c. Different gates	4. ITER-BF	5. ITER-SSI	6. ITER-ISH	11. ITER-SH
d. b∨c	7. TBF	8. GSSI	9. GISH	12. USH

LE: leakage fault ITRA: intra-gate ITER: inter-gate
TBF: two-line bridging fault SSI: short between a signal line and an internal node
ISH: internal short SH: short
GSSI: global SSI GISH: global ISH USH: universal short

B. Open faults

Detectability of open faults has been analyzed by Singh et al. [31], where open faults are classified into five types. In this paper, open faults are classified into three types (see Fig. 1).

- Gate-input open: This locates at an input-line of a logic gate. It causes opens at both PMOS and NMOS transistors, and it is detectable by IDDQ testing as well as logic testing.
- Transistor-gate open: This locates at gate of a transistor. It is detectable by IDDQ testing, but difficult to detect by logic testing.
- Source-drain open: This locates at source or drain of a transistor. It is difficult to detect by IDDQ testing, but easy by logic testing.

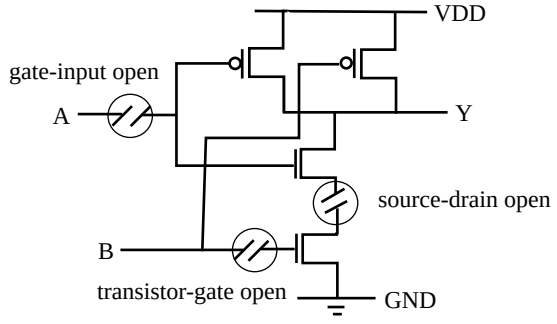

Fig. 1. Example of open faults

C. Bridging faults (BF)

In test generation for IDDQ testing, bridging faults (BFs) have been often considered as target faults. In previous literature, many different types of BFs have been considered and different terminologies have been used to describe these faults. Some of the more prevalent terminologies are: two-line bridging faults, internal shorts, external shorts, inter-gate shorts, intra-gate shorts, and leakage faults. In this paper, we define a BF as a short between two *nodes*, and denote a BF between node a and b as $< a, b >$. With this simple definition, BF model can represent various types of faults by an appropriate choice of set of *nodes* in the circuit as explained below. We classify BFs with respect to nodes and location of the fault. First we define a *node* to be one of the following two types:

- Internal node (IN): gate, source, drain, substrate (VDD, GND)
- Signal line (SL): input of a gate, output of a gate

Next we classify location of the two nodes, forming a BF, into the following three classes:

- In a same transistor
- In a same gate
- Between different gates

Based on the above definition and the division of nodes, classification of all BFs is shown in Table I. The second to fourth columns of this table show BF between SL and SL, SL and IN, IN and IN, respectively. The fifth column shows BF between arbitrary nodes. The containment relation between all possible BFs described in Table I is shown in Fig. 2 and as follows. ITRA-SH contains ITRA-BF, ITRA-SSI and ITRA-ISH. ITER-SH contains ITER-BF, ITER-SSI and ITER-ISH. The rows a, b, c show BFs in a transistor, in a gate, and between different gates, respectively. The row d represents arbitrary BFs in a circuit. TBF contains ITRA-BF and ITER-BF; GSSI contains ITRA-SSI and ITER-SSI; ITRA-ISH contains LE; and GISH contains ITRA-ISH and ITER-ISH. In this classification, USH is the largest class, which contains all other classes of BFs. Hence, USH contains BFs between arbitrary pair of nodes.

When the detection of BFs is discussed from the point of view of logic behavior, the following condition is often used. A BF is detected by IDDQ testing if the two nodes that are bridged (shorted) have opposite logic values, (0, 1) or (1, 0). Further, it is often assumed that in the faulty circuit two nodes that are physically connected have a same logic value. This assumes that a short to be an ideal short with 0 ohm resistance. Under this assumption, the set of internal nodes contains the set of all signal lines, therefore, ITRA-SH is equivalent to ITRA-ISH, ITER-SH is equivalent to ITER-ISH, and USH is equivalent to GISH.

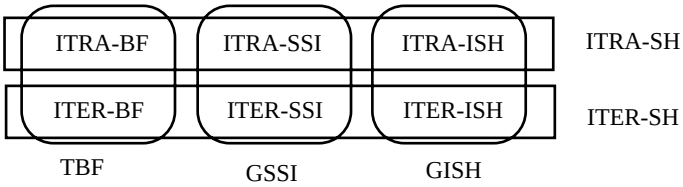

Fig. 2. Containment relation of fault models

D. Fault extraction

Choice of a fault model is only one aspect of the test process. Which faults should be included in a fault list is important as explained below. If target faults are selected based on ITRA-SH model, the number of all possible faults is proportional to the number of gates in the circuit. But if TBF, ITER-SH or USH constitutes the target faults, the number of all possible faults is proportional to square of the number of gates or signal lines. Thus, when any one of 4th to 12th classes of BFs in Table I are targeted, algorithms must be developed with consideration of the number of extracted faults.

Following different fault extractions have been reported.

1. Extraction based on layout information
2. Extraction of all possible faults
3. Random extraction from the set of all faults

Most recent papers on IDDQ test generation are listed in Table II along with the fault models used in these papers based on the terminologies developed by us above and the terminologies used by the authors. This table also lists the extracted faults.

III. Test generation

A. Fault simulation

As an efficient technique to calculate fault coverage of BFs, *node partition* technique has been used in [3, 4, 13, 17, 21, 22, 29, 32]. This technique simulates only a fault free circuit, and thus a large number of target faults can be dealt with. Moreover required memory space is proportional to the number of nodes of BFs. First all nodes are contained in one set. During fault simulation, sets of nodes are partitioned according to the signal values at nodes on application of each test vector. Consequently every set contains nodes which take same values for every test vectors. It is found that a BF between nodes in a same set is not detected because they do not take opposite values for any test vectors.

For fault diagnosis or test compaction, it is required to find all the test vectors which detect a target fault. If sequential circuits are target for such an objective, fault simulation should be performed with considering the propagation of effects of faults. By the simulation technique in [14, 16, 20], the value at faulty site is changed to X (don't care) when the fault is activated, and effect of the fault is propagated to the later timeframes.

B. Test generation for intra-gate shorts (ITRA-SHs)

When target faults are ITRA-SHs, test generation techniques for SAFs are efficient. It is known that test vectors which detect all detectable SAFs also detect all the detectable ITRA-SHs [17, 25]. For example, all the detectable ITRA-SHs in a 2-input NAND gate are detected by test vectors that set (0, 1), (1, 0) and (1, 1) at the two inputs. In previous literature, complete fault coverage is obtained for every ISCAS'85 benchmark combinational circuits [29].

However, occasionally a test vector that set (0, 0) may be necessary. For example, it is necessary when a high resistive short between the output of 2-input NAND gate and GND is considered. This is because large IDDQ is induced when both PMOS transistors are turned on, although smaller IDDQ is induced when only one PMOS transistor is turned on. Input vector (0, 0) may be also necessary for fault diagnosis. The amount of IDDQ is different between input vector (0, 0) and (0, 1) when a resistive short between the output of 2-input NAND gate and GND exists [12]. In general, higher amount of IDDQ is induced when input vector (0, 0) is applied, because the output is connected to VDD with lower resistance,

In [19], PODEM-based test generation is performed for input fault model. Input fault model is one of functional fault models of logic gates. Under this model, 2^k input vectors are generated for a k-input gate. All the faults which change the function of a gate are detected by 2^k input vectors. Obviously ITRA-SHs are covered by input fault model.

C. Test generation for two-line bridging faults (TBFs)

It is found that random testing is efficient for testing TBFs [17, 28, 29]. But to obtain complete fault coverage,

TABLE II
FAULT MODELS IN RECENT PAPERS

authors	fault (our terminology)	fault (authors' terminology)	extracted faults
Chakravarty & Thadikaran [3]	TBF	two line single BF	all
Dalpasso et al. [10]	TBF	TBF	random sample
Chen et al. [9]	TBF	BF	layout
Isern & Figueras[17]	ITRA-SH, TBF	internal-SH, external-SH	all
Mahlstedt et al. [23]	ITRA-SH, TBF	LE, ITRA-SH, inter-gate short	library-based, layout, random sample in experiments
Reddy et al. [29]	ITRA-SH, TBF	internal-BF, external-BF	all
Higami et al. [13]	ITRA-SH, TBF	internal-BF, external-BF	all
Bollinger & Midkiff [2]	USH	ITER-BF, ITRA-BF	layout (Carafe[18])
Chakravarty & Thadikaran[5, 6]	USH	BF	all
Lee et al. [21]	USH	TBF	all
Thadikaran et al.[32]	USH	metal-BF, all-BF, extracted-BF	all, topologically close pair
Maeda et al. [22]	USH	internal-SH external-SH	all

deterministic test generation is required for remaining undetected faults. One approach is to add an XOR gate. Let $< a, b >$ be a target BF. XOR gate is added such that a and b are inputs of the gate, and the output of the gate is an primary output. A test vector is generated to detect stuck-at-0 fault at the output of this XOR gate. If no test vector detects this stuck-at-0 fault, then BF $< a, b >$ is recognized to be undetectable.

Another technique to identify undetectable BFs is to compare functions of two signal lines. This can be implemented by using BDD [17]. If both functions of signal lines a and b are equivalent, then a and b always take same values and BF $< a, b >$ is recognized to be undetectable.

For TBFs in sequential circuits, random test vectors are also efficient. However, high fault coverage of TBFs is not always obtained by applying uniform random vectors for sequential circuits. One reason is that application of uniform random vectors to sequential circuits often leaves many flip-flops in the circuit unchanged. In such a case, weighted random vectors are efficient [13, 22]. "Weight" at each primary input is determined such that the frequency of logic 1 at each flip-flop approaches 50%.

We present a list of recent papers on test generation in Table III, where target faults, types of target circuits and applied techniques are briefly described.

In previous literature, 100% fault efficiency is obtained for TBFs in every ISCAS'85 benchmark combinational circuit [29]. For sequential circuits, 100% fault efficiency is difficult to obtain. Lee et al. used genetic algorithms [21], Thadikaran et al. used test generation method for stuck-at faults [32], and Maeda et al. used weighted random vectors and test generation method for stuck-at faults [22]. In these three papers, USHs are target faults.

D. Test compaction

In order to reduce test application time and memory space to store test data, it is important to reduce the number of test vectors.

Several efficient test compaction techniques for combinational circuits have been proposed. *Reverse order simulation* is a simulation technique with applying test vectors in the reverse order, i.e. the last generated test vector is applied first [19, 29]. This may make earlier generated test vectors unnecessary. This technique is simple, easy to implement and efficient for reducing test vectors. *Regeneration of test vectors based on essential fault information* has been proposed in [19]. Essential fault is defined as a fault which is detected by N or less number of test vectors among a generated test set, where N is pre-specified. A test vector is regenerated so that it detects more essential faults. As a result, a test vector which detects no essential faults becomes redundant, and it is removed. *Iterative improvement by changing one bit* has been proposed in [30]. This technique is simple and easy to implement. Random vectors are used as seeds, and the value at a primary input is inverted for one bit by one bit as long as at least one new fault is detected by a new test vector.

Test compaction techniques for sequential circuits have been proposed in [15]. The paper includes following five techniques; removal of inert subsequences, replacement of inert subsequences, removal with partial substitution, conditional removal of fault-detecting subsequences, conditional removal of inert subsequences. The techniques

TABLE III
Test generation (recent papers)

authors	fault	circuit	technique
Kondo & Cheng [20]	SAF	seq	TPG for SAF
Chen et al. [9]	TBF	seq	TPG for SAF, genetic algorithm
Dalpasso et al. [10]	TBF	comb	SPICE, BDD
Isern & Figueras[17]	INTRA-SH, TBF	comb	TPG for SAF
Reddy et al. [29]	INTRA-SH, TBF	comb	random tests, TPG for SAF
Mahlstedt et al. [23]	INTRA-SH, TBF	comb	random tests, deterministic TPG
Higami et al. [13]	INTRA-SH, TBF	seq	weighted random tests
Bollinger & Midkiff [2]	USH	comb	modified PODEM
Lee et al. [21]	USH	seq	genetic algorithm
Chakravarty & Thadikaran [6]	USH	seq	random tests, TPG for SAF
Thadikaran et al.[32]	USH	seq	TPG for SAF
Maeda et al. [22]	USH	seq	weighted random tests, TPG for SAF
seq: sequential circuit		comb: combinational circuit	

remove or replace inert subsequences as long as the state transition of a remaining test sequence is preserved. The fourth technique investigates if faults detected by a removed subsequence are also detected by a remaining test sequence. If all the faults are detected by a remaining test sequence, then a fault-detecting subsequence is removed.

Since measurement of IDDQ is a time-consuming process, reduction of IDDQ measurement vectors is more effective for reducing the total testing time. Therefore a method to select a small number of IDDQ measurement vectors has been proposed in [5, 6, 8, 14, 16, 20, 26]. Target faults are USHs in [5, 6, 14, 16], LEs in [8, 26], and SAFs in [20]. The problem of selecting minimum IDDQ measurement vectors can be formulated as a SETCOVER problem [5, 6]. To solve the problem, a method using a detection matrix is efficient [14, 16]. A detection matrix gives the information on the detectability of each fault by each test vector. Although a detection matrix is efficient for selecting IDDQ measurement vectors, large simulation effort is necessary for obtaining it. Moreover if a sequential circuit is a target, fault simulation must be performed with assuming the presence of faults. In a method of [14, 16], test vectors are classified based on essential fault information, and target faults for such simulation are selected by a heuristic technique. As a result, a small number of IDDQ measurement vectors are selected by a reasonable simulation time.

IV. Conclusion

In this paper we described several problems on IDDQ testing, particularly on fault models, fault simulation, test generation and test compaction, and we provided efficient methods among recent research. Although BF model has been considered in much of the literature, it often means different faults to different researchers due to lack of uniform terminology. We classified BFs with respect to the node set in a circuit and provided relations between different types of faults. We explained some of the techniques used for fault simulation and test generation. Since one of the important problem on IDDQ testing is to reduce testing time, we discussed techniques to reduce the number of test vectors and also the number of IDDQ measurement vectors.

In previous research most problems pertaining to combinational circuits testing have been solved, high fault coverage for SAFs, ITRA-SHs and BFs have been achieved, and compact test sets have been obtained. However, many of these problems for sequential circuits are still unsolved. In particular, the problems of sequentially untestable fault identification and testing time reduction require additional research and concerted effort.

References

[1] R. C. Aitken, "Extending the Psuedo-Stuck-At Fault Model to Provide Complete IDDQ Coverage," in *Proc. VLSI Test Symp.*, pp. 128–134, Apr 1999.

[2] S. W. Bollinger and S. F.Midkiff, "Test Generation for IDDQ Testing of Bridging Faults in CMOS Circuits," in *IEEE Trans. on Computer-Aided Design*, pp. 1413–1418, Nov. 1994.

[3] S. Chakravarty and P. Thadikaran, "Simulation and Generation of IDDQ Tests for Bridging Faults in Combinational Circuits," *IEEE Trans. on Computers*, vol. 45, pp. 1131–1140, Oct. 1996.

[4] S. Chakravarty and P. J. Thadikaran, "Simulation and Generation of IDDQ Tests for Bridging Faults in Combinational Circuits," in *Proc. VLSI Test Symp.*, pp. 25–32, Apr. 1993.

[5] S. Chakravarty and P. J. Thadikaran, "A Study of IDDQ Subset Selection Algorithms for Bridging Faults," in *Proc. Int. Test Conf.*, pp. 403–412, Oct. 1994.

[6] S. Chakravarty and P. J. Thadikaran, "Algorithm to Select IDDQ Measurement Points to Detect Bridging Faults," *Journal of Electronic Testing: Theory and Applications*, vol. 8, pp. 275–285, June 1996.

[7] S. Chakravarty and P. J. Thadikaran, *Introduction to IDDQ Testing*, Kluwer Academic Publishers, 1997.

[8] S. Chakravarty, S. T. Zachariah, and P. J. Thadikaran, "STBM: A Framework for Simulating and Selecting IDDQ Measurement Points for Leakage Faults," in *Dig. Int. Workshop on IDDQ Testing*, pp. 58–62, Nov. 1997.

[9] T. Chen, I. N. Haji, E. M. Rudnick, and J. H. Patel, "An Efficient IDDQ Test Generation Scheme for Bridging Faults in CMOS Digital Circuits," in *Dig. Int. Workshop on IDDQ Testing*, pp. 74–78, Nov. 1996.

[10] M. Dalpasso, M. Favalli, and P. Olivo, "Test Pattern Generation for IDDQ: Increasing Test Quality," in *Proc. VLSI Test Symp.*, pp. 304–309, Apr. 1995.

[11] R. Fritzemeier, J. Soden, R. Treece, and C. F. Hawlins, "Increasing CMOS IC Stuck-at Fault Coverage with Reduced IDDQ Test Sets," in *Proc. Int. Test Conf.*, pp. 427–435, Sep. 1990.

[12] A. E. Gattiker and W. Maly, "Current Signatures," in *Proc. VLSI Test Symp.*, pp. 112–117, Apr. 1996.

[13] Y. Higami, T. Maeda, and K. Kinoshita, "Sequential Circuit Test Generation for IDDQ Testing of Bridging Faults," in *Dig. Int. Workshop on IDDQ Testing*, pp. 12–16, Nov. 1997.

[14] Y. Higami, K. K. Saluja, and K. Kinoshita, "Observation Time Reduction for IDDQ Testing of Bridging Faults in Sequential Circuits," in *Proc. Asian Test Symp.*, pp. 312–317, Dec. 1998.

[15] Y. Higami, K. K. Saluja, and K. Kinoshita, "Static Test Compaction for IDDQ Testing of Sequential Circuits," in *Dig. Int. Workshop on IDDQ Testing*, pp. 9–13, Nov. 1998.

[16] Y. Higami, K. K. Saluja, and K. Kinoshita, "Efficient Techniques for Reducing IDDQ Observation Time for Sequential Circuits," in *Proc. Int. Conf. on VLSI Design*, pp. 72–77, Jan. 1999.

[17] E. Isern and J. Figueras, "Test Generation with High Coverages for Quiescent Current Test of Bridging Faults in Combinational Circuits," in *Proc. Int. Test Conf.*, pp. 73–82, Oct. 1993.

[18] A. Jee and F. J. Ferguson, "Carafe: An Inductive Fault Analysis Tool for CMOS VLSI Circuits," in *Proc. VLSI Test Symp.*, pp. 92–98, Apr. 1993.

[19] H. Kondo and K.-T. Cheng, "An Efficient Compact Test Generator for IDDQ Testing," in *Proc. Asian Test Symp.*, pp. 177–182, Nov. 1996.

[20] H. Kondo and K.-T. Cheng, "Driving Toward Higher IDDQ Test Quality for Seuqnetial Circuits: A Generalized Fault Model and Its ATPG," in *Dig. Int. Conf. on Computer-Aided Design*, pp. 228–232, Nov. 1996.

[21] T. Lee, I. N. Hajj, E. M. Rundnick, and J. H. Patel, "Genetic-Algorithm-Based Test Generation for Current Testing of Bridging Faults in CMOS VLSI Circuits," in *Proc. VLSI Test Symp.*, pp. 456–462, Apr. 1996.

[22] T. Maeda, Y. Higami, and K. Kinoshita, "Test Generation for Sequential Circuits under IDDQ Testing," *IEICE Trans. on Information and Systems*, vol. E81-D, pp. 689–696, July 1998.

[23] U. Mahlstedt, J. Alt, and M. Heinitz, "CURRENT: A Test Generation System for IDDQ Testing," in *Proc. VLSI Test Symp.*, pp. 317–323, Apr. 1995.

[24] Y. K. Malaiya and R. Rajsuman, *Bridging Faults and IDDQ Testing*, IEEE Computer Society Press, 1992.

[25] Y. K. Malaiya and S. Y. H. Su, "A New Fault Modeling and Testing Technique for CMOS Devices," in *Proc. Int. Test Conf.*, pp. 25–34, Oct. 1982.

[26] W. Mao and R. K. Gulati, "QUIETEST: A Methodology for Selecting IDDQ Test Vectors," *Journal of Electronic Testing: Theory and Applications*, vol. 4, pp. 349–357, 1992.

[27] R. Rajsuman, *Iddq Testing for CMOS VLSI*, Artech House Publisher, 1994.

[28] R. Rajsuman and D. A. Penry, "Coverage of Bridging Faults by Random Testing in IDDQ Test Environment," in *Proc. Int. Conf. on VLSI Design*, pp. 73–82, Jan. 1993.

[29] R. S. Reddy, I. Pomeranz, S. M. Reddy, and S. Kajihara, "Compact Test Generation for Briging Faults under IDDQ Testing," in *Proc. VLSI Test Symp.*, pp. 310–316, Apr. 1995.

[30] T. Shinogi and T. Hayashi, "A Simple and Efficient Method for Generating Compact IDDQ Test Set for Bridging Faults," in *Proc. VLSI Test Symp.*, pp. 112–117, Apr. 1998.

[31] A. D. Singh, H. Rasheed, and W. W. Weber, "IDDQ Testing of CMOS Opens: An Experimental Study," in *Proc. Int. Test Conf.*, pp. 479–489, Oct. 1995.

[32] P. Thadikaran, S. Chakravarty, and J. H. Patel, "Algorithm to Compute Bridging Fault Coverage of IDDQ Test Sets," *ACM Trans. on Design Automation of Electronic Systems*, vol. 2, pp. 281–305, July 1997.

[33] S. T. Zachariah and S. Chakravarty, "A Comparative Study of Psuedo Stuck-at and Leakage Fault Model," in *Proc. Int. Conf. on VLSI Design*, pp. 91–94, Jan. 1999.