# Gate-Sizing-Based Single $V_{dd}$ Test for Bridge Defects in Multi-Voltage Designs

Saqib Khursheed, Bashir M. Al-Hashimi, *Fellow, IEEE,* Krishnendu Chakrabarty, *Fellow, IEEE* and Peter Harrod *Senior Member, IEEE*

***Abstract*—The use of multiple voltage settings for dynamic power management is an effective design technique. Recent research has shown that testing for resistive bridging faults in such designs requires more than one voltage setting for 100% fault coverage; however switching between several supply voltage settings has a detrimental impact on the overall cost of test. This paper proposes an effective gate sizing technique for reducing test cost of multi-$V_{dd}$ designs with bridge defects. Using synthesized ISCAS and ITC benchmarks and a parametric fault model, experimental results show that for all the circuits, the proposed technique achieves single $V_{dd}$ test, without affecting the fault coverage of the original test. In addition, the proposed technique performs better in terms of timing, area, and power than the recently proposed test point insertion technique. This is the first reported work that achieves single $V_{dd}$ test for resistive bridge defects, without compromising fault coverage in multi-$V_{dd}$ designs.**

***Index Terms*—Gate Sizing, Test Cost, Resistive Bridging Faults, Multiple-$V_{dd}$ designs, Design for Testability**

## I. INTRODUCTION

RESISTIVE bridging faults (RBF) represent a major class of defects for deep submicron CMOS and can constitute 50% or more, of total defect count [1]. A bridge is defined as an un-wanted metal connection between two lines of the circuit, which may deviate the circuit from its ideal behavior. Resistive bridges have received increased attention on modeling, simulation and test generation [2]–[10]. Typically, a multi-$V_{dd}$ design has a set of discrete supply voltage settings it can switch between depending on the current workload and power saving mode [11]. Manufacturing test needs to ensure that such a design operates correctly over the entire set of supply voltage settings, while keeping the overall cost of test low.

It has been shown in [4] and more recently in [10] that the fault coverage of a test set targeting RBF can vary with the supply voltage used during test. This means that, depending on the operating $V_{dd}$ setting, a given RBF may or may not affect correct operation of the design. Consequently, to ensure high fault coverage for a design that needs to operate at a number of different $V_{dd}$ settings, it is necessary to perform testing at more than one $V_{dd}$ to detect faults that manifest themselves only at particular $V_{dd}$. It was shown in [10] that the majority of circuits (8 out of 12) require testing at more than one voltage setting to achieve 100% fault coverage, which means that the ATE (Automatic Test Equipment) will have to switch between different voltage settings to apply the test. Switching between different $V_{dd}$ settings during test is not desirable and can impact the cost of test. Therefore it is important to reduce the number of test $V_{dd}$ settings to one $V_{dd}$ leading to reduction in test cost.

To the best of our knowledge, the only investigation that addresses test cost reduction through minimizing the number of test $V_{dd}$ settings for multi-$V_{dd}$ designs has been presented in [10]. It demonstrates that test point insertion (TPI) can be used to reduce the number of $V_{dd}$ settings during test, without affecting the fault coverage of the original test, thereby reducing test cost. A drawback with the TPI technique [10] is that it does not guarantee a single $V_{dd}$ test and usually results in more than one test $V_{dd}$ setting. This paper proposes a new and more effective technique for reducing test cost of multi-$V_{dd}$ designs with bridge defects. It targets resistive bridges that cause faulty logic behavior, to appear at more than one test $V_{dd}$ setting, and uses gate sizing (GS) to expose the same physical resistance of the bridge at a single test $V_{dd}$. The number of test voltages is then reduced, minimizing test cost. We show that unlike TPI, it is possible to achieve single $V_{dd}$ test without affecting the fault coverage of the original test.

In this paper, we present a gate sizing technique with two different algorithms to identify bridges requiring multiple $V_{dd}$ settings for detection. The first algorithm is *Deterministic* that utilizes only SAT-based test generation procedure [10] to identify bridges that require multiple $V_{dd}$ settings for detection and marks their driving gates for replacement. The second

Manuscript received on 28th Oct, 2009 and revised on 4th Feb, 2010. The authors would like to acknowledge EPSRC (U.K) for funding this work in part under grant nos. EP/E035965/1 and EP/H011420/1. The work of K. Chakrabarty was supported in part by the US National Science Foundation under grant no. ECCS-0823835. A preliminary version of this work has been published at DATE'09 (http://eprints.ecs.soton.ac.uk/17047).

S. Khursheed and B. M. Al-Hashimi are with the School of Electronics and Computer Science, University of Southampton, SO17 1BJ, UK. (email: ssk06r@ecs.soton.ac.uk; bmah@ecs.soton.ac.uk)

K. Chakrabarty is with the department of Electrical and Computer Engineering, Duke University, USA. (email: krish@ee.duke.edu)

P. Harrod is with ARM Limited, Cambridge, UK. (email: peter.harrod@arm.com)

Copyright © 2010 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending an email to pubs-permissions@ieee.org.

algorithm is *Probabilistic* that is motivated by an observation discussed in [10] that SAT-based test generation can take up to 71% of total time inside SAT engine and attempts to reduce the number of times SAT engine is invoked thereby reducing computation time. These two algorithms present a trade-off between accuracy and speed; experimental results show an improvement of up to 50% in computation time. This paper also evaluates the impact on timing, area and power of the proposed technique, and comparison with the TPI [10] shows that the proposed gate sizing technique performs better in terms of these three parameters. In comparison to the original design, the proposed technique has minimal impact on area and power, while timing has improved for many designs.

The paper is organized as follows: Section II gives an overview of resistive bridge behaviour in multi-$V_{dd}$ design. The motivation for using gate sizing to reduce the number of test $V_{dd}$ settings is discussed in Section III. Section IV presents the proposed gate sizing technique. Experimental results are reported in Section V, and Section VI concludes the paper.

## II. Preliminaries

To explain the proposed gate sizing technique, it is necessary to discuss some concepts related to resistive bridging faults and their behavior in the context of multi-$V_{dd}$ designs. A typical bridge fault behavior is illustrated in Fig. 1. Fig. 1-A shows a resistive bridge, D1 and D2 are the gates driving the bridged nets, while S1, S2 and S3 are successor gates; the output of D1 is driven high and the output of D2 is driven low. The dependence of the voltage level on the output of D1 ($V_O$) on the equivalent resistance of the physical bridge is shown in Fig. 1-B (based on SPICE simulation with 0.12$\mu$m library). To translate this analog behavior into the digital domain, the input threshold voltage levels $V_{th1}$ and $V_{th2}$ of the successor gates S1 and S2 have been added to the $V_O$ plot. The logic threshold of a gate input is defined as the input voltage at which the output reaches half of the supply voltage, while other inputs of the gate are at non-controlling value(s) [12]. For each value of the bridge resistance $R_{sh} \in [0, \infty)$, the logic values read by inputs $I_1$ and $I_2$ can be determined by comparing $V_O$ with the input threshold voltage of the corresponding input. These values are shown in the second part of Fig. 1-B (marked as "digital domain"). Crosses are used to mark the faulty logic values and ticks to mark the correct ones. It can be seen that, for bridges with $R_{sh} > R_2$, the logic behavior at the fault site is fault-free (all inputs read the correct value), while for bridges with $R_{sh}$ between 0 and $R_2$, one or more of the successor inputs are reading a faulty logic value. A number of bridge resistance intervals can be identified based on the corresponding logic behavior. For example, bridges with $R_{sh} \in [0, R_1]$ exhibit the same faulty behavior in the digital domain (all successor inputs read the faulty logic value). Similarly, for bridges with $R_{sh} \in [R_1, R_2]$, successor gate S2 reads the faulty value, while S1 reads the

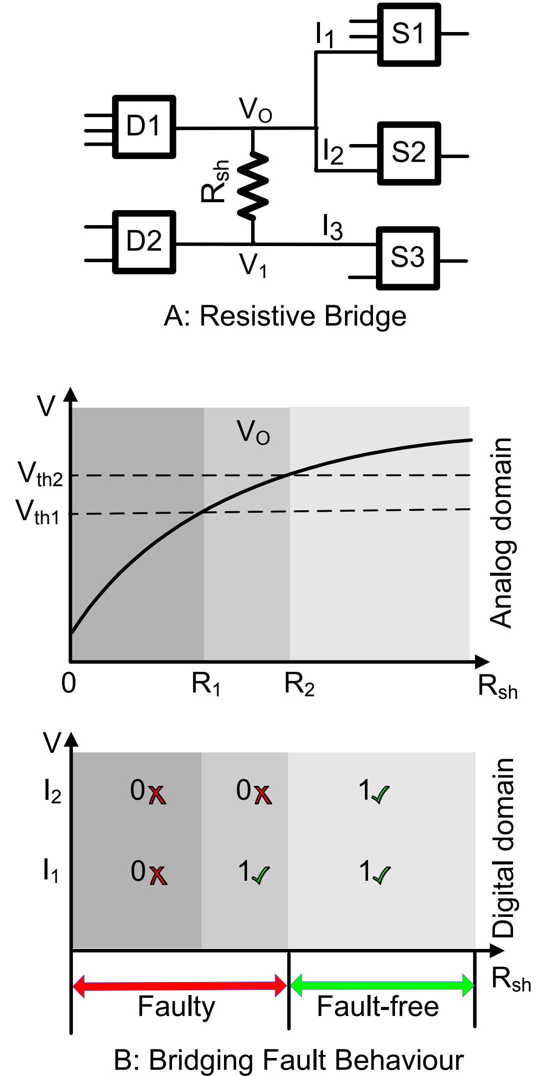


Fig. 1. Bridge fault example and its behaviour in analog and digital domain

correct value; and finally for $R_{sh} > R_2$, all the successor gates read the correct logic value. Consequently, each interval $[R_i, R_{i+1}]$ corresponds to a distinct logic behavior occurring at the bridge fault site. This distinct logic behavior at the fault site is referred to as *Logic Fault* and constitutes the following: boolean input to driving gates, resistance range coverage, $V_{dd}$ setting and boolean values interpreted by driven inputs of successor gates.

Next, consider Fig. 2, which shows the relationship between the voltage on the output of gate D1 (Fig. 1-A) and the bridge resistance for two different supply voltages $V_{ddA}$ and $V_{ddB}$ [2], [4]. Fig. 2 also shows how the analog behavior at the fault site translates into the digital domain. Using similar explanation (as for Fig. 1-B), we can see that two distinct Logic Faults LF1 and LF2 can be identified for each $V_{dd}$ setting. However, because the voltage level on the output of D1 does not scale linearly with the input threshold voltages of S1 and

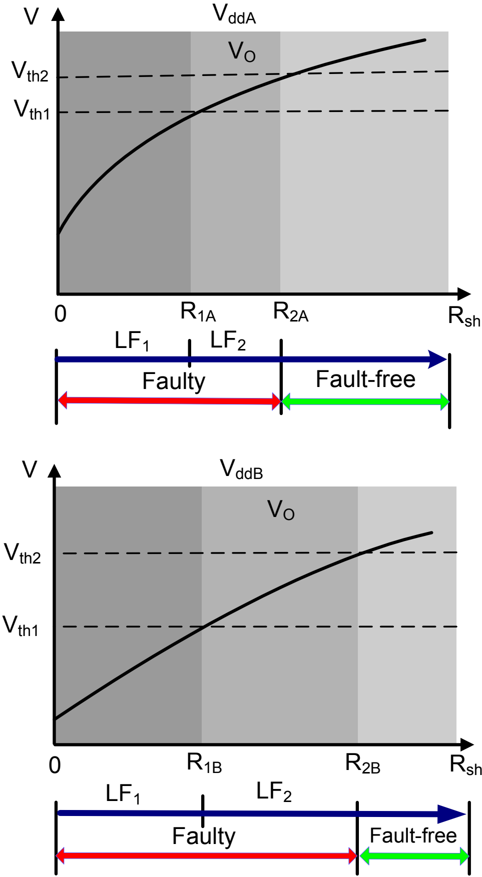


Fig. 2. Bridge fault behaviour at two different voltage settings

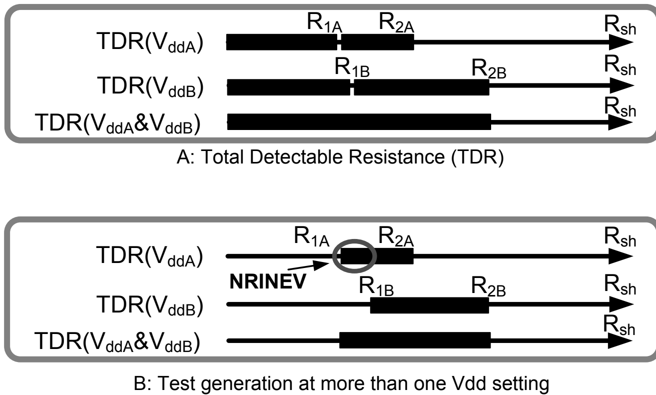


Fig. 3. Resistance coverage at two different voltage settings

S2 when changing the supply voltage, the resistance intervals corresponding to LF1 and LF2 differ from one supply voltage setting to another [10]. Fig. 3 shows the Total Detectable Resistance (TDR) for the LFs detected at two voltage settings separately and combined as well. Furthermore, this means that in the case of Fig. 3-A, the complete range of physical defects can be covered alone at $V_{ddB}$.

Consider a case where logic fault LF1 covering a resistance range ([0, $R_{1A}$] at $V_{ddA}$, and [0, $R_{1B}$] at $V_{ddB}$) becomes undetectable, in which case Fig. 3-B shows the detectable resistance intervals at two voltage settings. For a certain bridge the *Essential* $V_{dd}$ setting is the one at which the highest resistance interval is detected, which is $V_{ddB}$ in this case. From a test generation point of view, essential $V_{dd}$ has to be included in test generation, as the highest resistance interval, of certain bridge(s), exists at the essential voltage setting(s). On the other hand, non-essential voltage settings ($V_{ddA}$ in this case) are included in test generation only because some non-redundant intervals are detectable at non-essential voltage setting(s). These intervals are referred to as NRINEV (Non-Redundant Intervals at Non-Essential $V_{dd}$) [10]. One such NRINEV interval is highlighted in Fig. 3-B. Therefore test generation tool uses extra $V_{dd}$ settings to cover such NRINEV intervals in order to achieve 100% fault coverage.

The only investigation to reduce the number of test voltages for resistive bridging faults is presented in [10], which utilizes Test Point Insertion (TPI). Test points are used to provide additional controllability and observability at the fault-site to detect NRINEV intervals at essential $V_{dd}$, which are otherwise redundant (at essential $V_{dd}$) and therefore help reducing the number of test $V_{dd}$ settings. Fig. 3-B shows that the resistance range marked as NRINEV is covered at essential $V_{dd}$ ($V_{ddB}$) by providing additional controllability and observability using test points. TPI has shown reduction in the number of test $V_{dd}$ setting(s) but it has some limitations. Experimental results presented in [10] show that the TPI is unable to reduce to single test $V_{dd}$ for the majority of circuits (10 out of 13 circuits require more than one test $V_{dd}$). This is because TPI cannot reduce the number of test $V_{dd}$ setting(s) below the number of essential $V_{dd}$ setting(s). This can be understood from the following explanation. In Fig. 1-A, the gates used for driving the bridge (D1, D2) and the driven gates (S1, S2, S3) influence the number of essential $V_{dd}$ setting(s) in a circuit. For the same circuit, assume that D1 is driving high and D2 is driving low, the output of D2 ($V_1$) on the equivalent resistance of the physical bridge is shown in Fig. 4, which shows that higher resistance range is covered at 1.2V (non-preferred test $V_{dd}$) than at 0.8V (preferred test $V_{dd}$) as $R_{1.2V} > R_{0.8V}$. This means that 1.2V becomes essential test $V_{dd}$ and TPI has to include it for 100% fault coverage, as the resistance range covered at 1.2V cannot be covered at 0.8V. The TPI has some limitations (not limited to the technique proposed in [10]) that to increase the fault coverage and to reduce test cost it may be necessary to introduce extra overhead on timing, area and power as is the case with [13]–[15].

## III. Impact of Gate Sizing on Test $V_{DD}$ Reduction

Gate sizing has been used to enhance timing performance of designs and more recently to tackle soft error rate in logic circuits [16]. It was shown in [17] that bridges driven by gates with equal drive strength are likely to be detected at higher $V_{dd}$ settings. We investigate the effect of gate sizing on the behavior of resistive bridging faults, and how it can

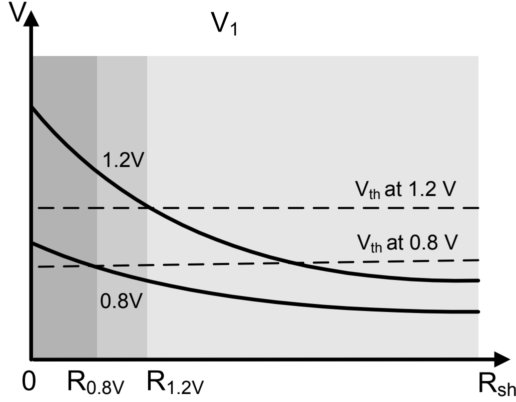


Fig. 4. Resistance range detection at different voltage settings

be used to propagate faulty behavior, such that a higher physical resistance is exposed at a single $V_{dd}$ setting (thereby reducing the number of essential test $V_{dd}$ settings to one). The limitations of TPI can be addressed by adjusting the driving gates (D1, D2) or driven gates (S1, S2, S3) at the fault-site. The driving/driven gates can be adjusted by two approaches, which include the following:

- Modifying logic threshold of driven gates,
- Modifying drive strength of driving gates.

### *A. Modifying logic threshold of driven gates*

In this case, the logic threshold of the driven gate is adjusted such that a higher resistance range is detectable at the lowest $V_{dd}$ setting. This observation is further elaborated in Fig. 5, where the logic threshold of the same gate inputs as for Fig. 4, is reduced by gate-sizing. Therefore, the highest resistance interval is exposed at the lowest $V_{dd}$ setting since $R_{0.8V} > R_{1.2V}$, which facilitates test generation at the lowest $V_{dd}$ setting.

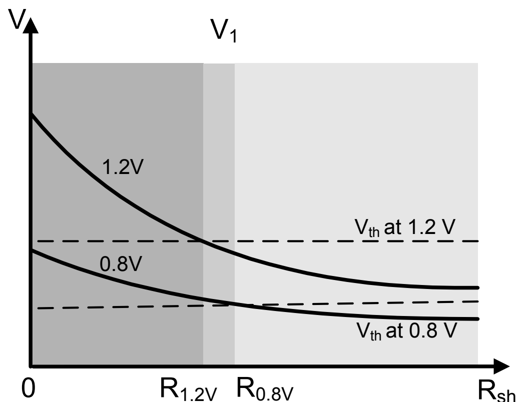


Fig. 5. Resistance range detection after adjusting logic thresholds of the driven gates

The logic threshold can be adjusted by altering the width/length of the PMOS/NMOS transistor connected to the

TABLE I
TRANSISTOR WIDTH MODIFICATION FOR ALTERING LOGIC THRESHOLD

| Gate (Input) | $W_p/W_n$ * | | Logic Th. Diff. |
|---|---|---|---|
| | **Original** | **Re-designed** | **@ 0.8V $V_{dd}$** |
| **4 Input NAND (C)** | 0.64/0.46 | 3.09/0.24 | -80 mV |
| **4 Input NAND (B)** | 0.64/0.46 | 5.22/0.24 | -100 mV |
| **5 Input AND-NOR (B)** | 0.94/0.64 | 6.79/0.24 | -140 mV |

* Width is in $\mu$m

particular gate input, or by using the body bias effect. For an inverter it is given by [18]:

$$V_{in} = \frac{V_{DD} + V_{tp} + V_{tn}\sqrt{\frac{\beta_n}{\beta_p}}}{1 + \sqrt{\frac{\beta_n}{\beta_p}}} \quad (1)$$

where, $V_{in}$ is the voltage at the input of the gate, $V_{DD}$ is the supply voltage, $V_{tp}$ is the threshold voltage of the PMOS transistor, $V_{tn}$ is the threshold voltage of the NMOS transistor.

$$\beta = \mu C_{ox}\left(\frac{W}{L}\right) \quad (2)$$

where, $\beta$ is the MOS transistor gain factor, $\mu$ is the effective surface mobility of the carriers, $C_{ox}$ is the gate oxide capacitance. From (1), it can be seen that a variation in $W_p$ and $W_n$ can alter the logic thresholds of a given gate input. This observation was used to conduct some experiments using $0.12\mu$m ST Microelectronics library. The transistor widths (connected to the gate input of interest) are varied to reduce the logic threshold, while operating at 0.8V $V_{dd}$. For all the considered cases, the targeted change in logic threshold was -80 mV or less to detect the fault at the lowest $V_{dd}$ setting, as that exposes higher resistance at the lowest $V_{dd}$ setting. The resultant widths for some of the transistors are shown in Table I, where the first column shows the gate for which the logic threshold is varied, followed by the $(W_p/W_n)$ ratios of the original design and that of the re-designed gates. The last column shows the difference in logic thresholds as a result of gate-sizing. It can be seen that for all the cases the ratio between $(W_p/W_n)$ is much higher than usually suggested design rule ratio of $(W_p/W_n) \approx 1.5 - 2.5$ [18]. The ratios (in Table I) result in unbalanced charging/discharging time ($t_{phl}$ and $t_{plh}$) and violate design rules. For these reasons, modification of logic thresholds of the driven gates is not further considered to achieve single $V_{dd}$ test. We also examined body biasing to vary the logic threshold but preliminary examination did not provide sufficient variations. For the cases considered, it resulted in $\approx 20$ mV variation in logic threshold (operating at 0.8V $V_{dd}$) at the targeted gate input. Therefore logic threshold modification either by changing (width, length) ratios or by body biasing did not provide sufficient change in logic threshold voltages, and therefore these two methods are not pursued further to achieve single $V_{dd}$ test.

### B. Modifying drive strength of driving gates

The drive strength of the gates driving the bridged nets can be adjusted to increase the voltage on the bridged nets ($V_1$ Fig. 1-A, where D1 is driving high and D2 is driving low). This increase in voltage level can help expose higher resistance at the lowest $V_{dd}$ setting thereby reducing the number of essential $V_{dd}$ settings; additionally it can also be used to cover NRINEV (Non-Redundant Interval at Non-Essential $V_{dd}$) intervals at the lowest $V_{dd}$ setting. This concept is illustrated in Fig. 6, which shows the same pair of bridged nets as Fig. 4, i.e., the logic thresholds of the driven gates remain the same. It can be seen that the voltage level $V_1$ has increased such that $R_{0.8V} > R_{1.2V}$, as a result of increasing the drive strength of the gates driving the bridge. This means that during test pattern generation, logic fault at 0.8V will be targeted leading to single $V_{dd}$ test.

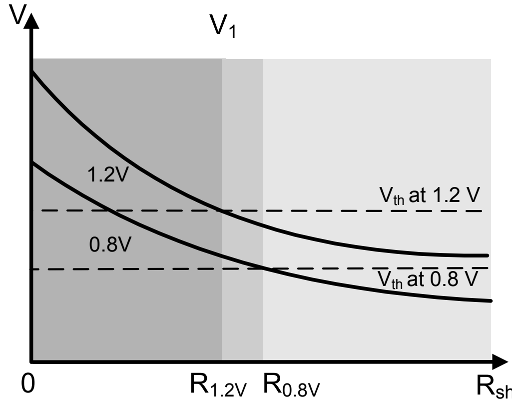


Fig. 6. Resistance range detection after adjusting the drive strength of the gates driving the bridge

The drive current of an NMOS transistor operating in *active* region is [18].

$$I_{ds} = \beta \left[ (V_{gs} - V_t) V_{ds} - \frac{V_{ds}^2}{2} \right] \quad (3)$$

where, $I_{ds}$ is the drain-source current, $\beta$ is the gain factor expressed by (2), $V_{gs}$ represents the gate-source voltage and $V_t$ is the transistor threshold voltage.

From 3, it can be observed that the drive current $I_{ds}$ is directly proportional to the gain factor $\beta$ (in saturation and active modes), which in turn is directly proportional to the $W/L$ of the transistor. Thus replacing a gate with another having higher value of $\beta$ (for transistors feeding the output) results in higher drive strength. This is feasible since, different versions of functionally equivalent gates are usually available in the gate library.

We conducted an experiment to analyze the impact of increasing drive strength of gates driving the bridged nets on resistance coverage of bridge defects. For this purpose 10 circuits were synthesized using 0.12 $\mu$m STMicroelectronics gate library and Synopsys design compiler. A fault simulator and test pattern generator from [10] is used to determine the detectable resistance range at three $V_{dd}$ settings, i.e., 0.8V, 1.0V, and 1.2V. For each design, a bridge is inserted at a location that requires one or more $V_{dd}$ setting for complete resistance coverage; unique resistance range at each $V_{dd}$ setting is recorded that is not detectable at other $V_{dd}$ settings. This is followed by replacing the gate with another having higher drive strength and repeating the procedure to determine the change in resistance coverage at each $V_{dd}$ setting. The results are shown in Table II. As can be seen, the resistance range for all the circuits has increased and for each design, 0.8V $V_{dd}$ setting alone covers maximum resistance range, which is not covered at any other $V_{dd}$ setting. For instance, a bridge in the design c2670 covers 0 to 0.4 kΩ at 1.0V and 0.2 to 0.5 kΩ at 1.2V in original design. After increasing the drive strength of the driving gate the resistance range at 0.8V increased substantially from 0 to 3.4 kΩ; resistance coverage at 1.2V is covered completely at 0.8V and this is why it is not shown in the table. A similar trend is observed for the rest of the benchmarks shown in Table II. From this experiment two key observations are made:

- The detectable resistance range of a bridge defect can be increased by increasing the drive strength of driving gate. This is further shown in Fig. 7, which shows higher defect resistance range is covered by replacing a gate (driving high, D1 as in Fig. 1) with higher drive strength gate $I_{ds2}$, which is greater than $I_{ds1}$.
- This increase is much higher at 0.8V than other voltage settings and for all the cases 0.8V alone captures the unique detectable resistance range.

These observations are exploited by the proposed gate sizing technique to achieve single $V_{dd}$ test.

TABLE II
RESISTANCE RANGE COVERAGE AT 3 DIFFERENT $V_{DD}$ SETTINGS BY INCREASING THE DRIVE STRENGTH OF GATES DRIVING THE BRIDGE

| Ckt | Before Gate Sizing | | | After Gate Sizing |
|---|---|---|---|---|
| | 1.2V | 1.0V | 0.8V | 0.8V |
| ISCAS-85 Benchmarks | | | | |
| c432 | 0-1 kΩ | | | 0-4.3 kΩ |
| c1355 | 0.9-1.3 kΩ | | 0-1.2 kΩ | 0-4.5 kΩ |
| c1908 | 0-1.8 kΩ | | | 0-6.3 kΩ |
| c2670 | 0.2-0.5 kΩ | 0-0.4 kΩ | | 0-3.4 kΩ |
| c3540 | 0-0.6 kΩ | | | 0-3.3 kΩ |
| ITC-99 Benchmarks | | | | |
| b01 | 0.9-1.1 kΩ | 0.8-1 kΩ | 0-0.9 kΩ | 0-4.3 kΩ |
| b02 | 0.5-1.5 kΩ | | 0-1.3 kΩ | 0-4.6 kΩ |
| b03 | | | 0-7.3 kΩ | 0-7.9 kΩ |
| b04 | 1.8-2.2 kΩ | 2.2-2.6 kΩ | 0-1.3 kΩ | 0-8.3 kΩ |
| | | | 2.9-3.3 kΩ | |
| b05 | 0-0.8 kΩ | | | 0-1.7 kΩ |

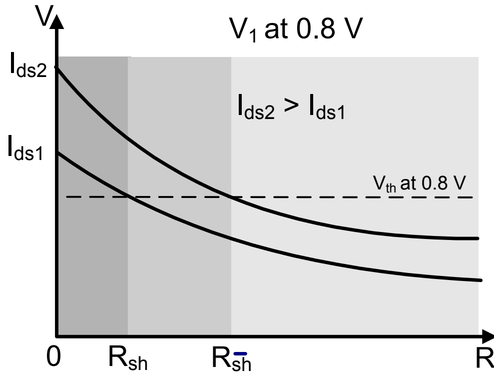


Fig. 7. Higher drive strength results in increasing the covered resistance range of a bridge defect at same $V_{dd}$ setting

## IV. Proposed Technique for Single $V_{DD}$ Test

This section presents two gate sizing algorithms to reduce the number of test $V_{dd}$ setting(s) for resistive bridge defect. Both algorithms consist of two phases: gate(s) identification and replacement, during which they identify the gates that should be replaced (for single $V_{dd}$ test), followed by test generation phase on the modified circuit to achieve single $V_{dd}$ test set. The process of gate identification for replacement distinguishes the two proposed algorithms. The first algorithm capitalizes on test generation method from [10] to identify bridges that require more than one $V_{dd}$ setting for complete fault coverage and is referred to as the *Deterministic Algorithm*. The second algorithm is based on a probabilistic method to identify bridge location(s) that may need more than one $V_{dd}$ setting and is referred to as the *Probabilistic Algorithm*. The two algorithms show a trade-off between accuracy and speed as discussed in Section V.

### A. Deterministic Algorithm

We briefly describe the *Deterministic Algorithm*[1] (DA). It is included because the *Probabilistic Algorithm* (PA) uses the same flow and the two algorithms are compared in Section V demonstrating the trade-off between accuracy and speed.

The algorithm (Fig. 8) starts by test generation (test generation follows the method presented in [10]) and marks all the bridges, which require test generation at higher than the lowest $V_{dd}$ setting. All such bridges are placed in $TargetBridgeList$ and all the driving gates of the respective bridges are marked as potential candidates for gate replacement. The algorithm then solves a minimum set covering problem that identifies the minimum number of driving gates, such that all the bridges are covered. The selected gates are placed in $minGatesList$ (step-2). The algorithm then takes each selected gate in $minGatesList$ and replaces it with another having higher drive strength from the gate library (step 3-5). After updating the netlist, the algorithm generates a test set considering complete bridge list and finally returns with an updated netlist and a new test set.

[1]This algorithm was presented in our earlier publication in DATE 2009 and can be downloaded from http://eprints.ecs.soton.ac.uk/17047.

**Input:** Netlist
**Output:** Test Set, Modified Netlist
1: Compute $TargetBridgeList$ by running test generation using the netlist
// *M*ark the bridges that require test at additional
// *v*oltage setting(s)
2: Compute minimum number of driving gates $minGatesList$ across complete $TargetBridgeList$ by solving a minimum set cover
3: **for all** $minGatesList$ **do**
4: Replace the selected gate with another having higher drive strength.
5: **end for**
6: Generate Test Set for the modified netlist using complete bridge list.
7: **return** $(Modified\,netlist,\,Test\,set)$

Fig. 8. Deterministic Algorithm

### B. Probabilistic Algorithm

This algorithm reduces run-time to identify bridge locations for gate replacement. An experiment conducted using 12 different ISCAS-85 and ISCAS-89 benchmarks and the SAT-based test generator [10] used in this work show that on average, 49% of total time spent during test generation is taken by the SAT engine [19], and it can take as much as 71% of total time [20]. The SAT-solver has exponential worst-case complexity [20] and therefore the purpose of the *Probabilistic* algorithm is to restrict its usage thereby reducing run-time. In the *Deterministic Algorithm* bridge locations for gate replacement are identified by invoking test generator [10] in step-1, as shown in Fig. 8. The *Probabilistic Algorithm* (PA) aims at reducing run-time by selectively using the test generator (and therefore SAT-solver) and does not use it by default. The PA is invoked as step-1 of gate sizing technique (to compute $TargetBridgeList$) without affecting rest of algorithmic flow, shown in Fig. 8.

As discussed in Section II, a bridge defect consists of a number of logic faults at each $V_{dd}$ setting (Fig. 2); all logic faults per bridge constitute its fault domain. Since bridges requiring higher $V_{dd}$ settings for detection are only targeted for gate replacement, the probabilistic algorithm categorizes the $V_{dd}$ setting of each bridge location, which is used to decide whether gate replacement is required or not. Each bridge location is categorized by computing the probability of detecting logic faults at higher $V_{dd}$ settings in comparison to those at the lowest $V_{dd}$ setting. Therefore, a bridge with higher probability of fault detection at the lowest $V_{dd}$ setting is not targeted for gate replacement. Probability based categorization

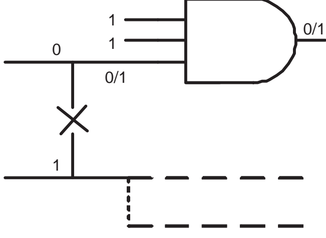


Fig. 9. Observability calculation

and comparison of logic faults reduces the need of invoking test generator, thereby speeding up the gate sizing technique.

This is achieved by assigning a detection value (DV) to each logic fault, which is a probabilistic estimate of controllability and observability of each logic fault in the fault domain (all logic faults per bridge) and represents the probability of fault detection. Eq. (4) is used as a comparative measure between different logic faults belonging to the same bridge and puts more emphasis on local analysis at the fault-site by taking into account signal probabilities of nets associated with the fault site. It uses signal probabilities to determine the likelihood of activating the fault and its effect reaching output of gates fed by the bridge (for example output of "AND" gate as in Fig. 9), the number of gates fed by the "AND" gate, and its minimum distance from the primary outputs as in [21]. In our experiments, signal probabilities are calculated by simulating pseudo-random patterns, however other analytical methods for estimating signal probability can be used for this purpose just as well. In a given circuit, signal probability per net is found by assigning a probability of 1(0) by carrying out logic simulations on the circuit using pseudo-random test patterns, until the probability of 1(0) do not change in last 200 iterations on any net. The number of iterations (200) is found by experimenting with different number of iterations from 50 to 300, and with 200 iterations, probability values are stable for all benchmarks.

DV is assigned by computing probability of fault activation and fault effect observation at the output of gates fed by the bridge. DV(LF) is the detection value per logic fault, which is computed for each candidate logic fault as follows:

$$DV(LF) = C(LF) \cdot O(LF) \tag{4}$$

where C(LF) is a probabilistic measure of the logic fault controllability, O(LF) is a probabilistic measure of observability of the fault at the outputs of gates fed by the bridge

$$C(LF) = \prod_{i=1}^{n} (Prob(i)) \tag{5}$$

where $n$ is the cumulative number of inputs of the two gates driving the bridged nets and *Prob(i)* is the signal probability of logic value required by the LF on input $i$

$$O(LF) = \sum_{i=1}^{m} \frac{(f(X)) \cdot G_i}{D(PO)_i} \tag{6}$$

where $m$ is the number of gates fed by the bridged nets, which propagate the faulty value to their outputs, $G$ is the number of gates fed by each such gate, and $D(PO)$ is the minimum distance of fault observing gate fed by the bridge from primary output(s). $f(X)$ is the probability that the fault effect is propagated through gate $X$, computed as follows:

$$f(X) = \frac{\sum_{j=1}^{k} \prod_{i=1}^{l} SP_{i,j}}{2^l} \tag{7}$$

where $k$ is the number of input combinations which propagate the fault effect to the output of successor gate $X$, $l$ is the number of inputs of gate $X$ which are not fed by the bridge, and $SP_{i,j}$ is the probability of having the value corresponding to input combination $j$ on input $i$. For example, a 3-input AND gate fed by the bridge (as shown in Fig. 9) there is one input configuration which will propagate the fault (0/1) to its output out of the 4 possible combinations on the two inputs which are not fed by the bridge. Assuming the "1" probabilities of the inputs which are not driven by the bridge to be 0.4 and 0.7 respectively, the probability of this gate propagating the fault to its output is $\frac{(0.4*0.7)}{4} = 0.07$.

The categorization of bridge defect to a specific $V_{dd}$ is shown in Fig. 10. It shows $V_{dd}$ specific logic faults, with respective resistance range and detection value, where $DV \in [0, 1]$. Fig. 10-a shows all logic faults of a bridge, including one at the highest $V_{dd}$ setting (black bar) and the lowest $V_{dd}$ setting (gray bar) with their respective DV. As can be seen, the resistance range covered at the highest $V_{dd}$ setting has lower DV than 3 overlapping logic faults at the lowest $V_{dd}$ setting. It means that the probability of this bridge to be detected at the highest $V_{dd}$ setting is 3 times lower than that of the lowest $V_{dd}$ setting. Similarly, a bridge resistance at the highest $V_{dd}$ is shown in Fig. 10-b, which shows the complete resistance range overlap by 2 logic faults, each with higher and lower DV, at the lowest $V_{dd}$ setting. The Probabilistic algorithm uses this type of comparison to ease bridge identification (requiring gate replacement) without invoking computationally expensive test generator [10].

Since logic circuits have different depths, topologies and design styles, a challenge is to establish a generic set of criteria to categorize bridges according to their $V_{dd}$ setting - more importantly the criteria should hold on a wide variety of benchmarks and cover the worst case scenario for each design. For this reason, we performed a detailed analysis using 23 benchmarks, with various gate counts, design styles (ISCAS 85, 89 and ITC 99) and in total more than 110,000 bridge locations. After detailed analysis, a set of criteria is formulated to categorize a bridge to the lowest $V_{dd}$ test setting. A bridge is referred to as *Low $V_{dd}$ Bridge*, if its resistance ranges across all logic faults at higher $V_{dd}$ settings are completely overlapped

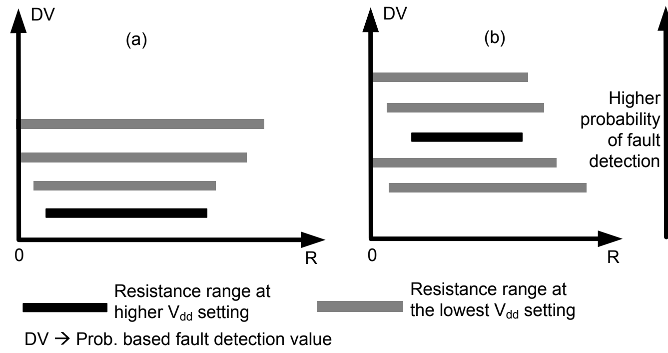


Fig. 10. Logic faults comparison using probability based detection value

by those at the lowest $V_{dd}$ setting, using one of the following criteria:

1) Resistance range at higher $V_{dd}$ is covered by at least 3 logic faults, at the lowest $V_{dd}$ setting, with higher detection value, as shown in Fig. 10-a.
2) Resistance range is covered by 2 logic faults with higher detection value, AND at least 2 logic faults with lower detection value, as shown in Fig. 10-b.
3) Resistance range covered by 1 logic fault with higher detection value, AND at least 15 logic faults with lower detection value.
4) Resistance range covered by at least 20 logic faults with lower detection value.

The above listed criteria is developed after detailed experimentation using benchmarks shown in Table III. The number of overlapping logic faults represent the worst case scenario over all bridges per design. It is used by the algorithm to cover a subset of bridges requiring the lowest $V_{dd}$ test. Such bridge locations do not need gate replacement and therefore reduce the number of calls to the SAT-solver made by the test generator. The rest of the bridges are categorized as *Gray Zone* bridges as they may need gate replacement to achieve single $V_{dd}$ test. Only for these bridges, the algorithm uses test generation to determine the exact $V_{dd}$ test setting for detection of each such bridge. As a result of test generation, the Gray Zone bridges are categorized as either requiring High $V_{dd}$ or Low $V_{dd}$ test for detection. The above list of criteria serves as a useful filter to distinguish the bridges requiring High $V_{dd}$ test and results in speeding up the process of gate identification for replacement by reducing the use of test generator.

To further evaluate the above listed criteria, we conducted experiments using benchmarks shown in Table III, with higher limits on the number of overlapping logic faults. In all cases, the experiments resulted in only converting Low $V_{dd}$ bridges to Gray Zone bridges, without affecting those requiring High $V_{dd}$ test. This and the above mentioned reasons indicate that the developed set of criteria for bridge identification, employed by the *Probabilistic* algorithm, is expected to hold on other benchmarks just as well. However, there is still a non-zero probability of missing out a bridge (Gray Zone bridge identified as low $V_{dd}$) in another experimental setup. Such corner

**Input:** Netlist, Bridge locations
**Output:** Categorize all bridge locations to either of the two categories: Low $V_{dd}$, High $V_{dd}$

1: Compute signal probabilities on all nets
2: **for all** Bridge locations **do**
3: Generate a list of logic faults candidates at each $V_{dd}$ setting
4: Retain unique logic faults at all $V_{dd}$ settings
5: **for all** LF candidates **do**
6: Compute DV(LF)
7: **end for**
8: Sort all logic faults using their respective DV(LF)
9: Categorize bridge location to either Low $V_{dd}$ or Gray Zone
10: Invoke test generator for Gray Zone Bridge and categorize it as either High $V_{dd}$ or Low $V_{dd}$ Bridge
11: Update $(TargetBridgeList)$
12: **end for**
13: **return** $(TargetBridgeList)$

Fig. 11. LF Ranking and Bridge Categorization of PA

cases will be identified by step-6 of the algorithm presented in Fig. 8 resulting in an additional iteration of the algorithm.

The Probabilistic Algorithm (PA) is shown in Fig. 11. It uses signal probabilities to quantify the effort required by a logic fault for detection. In step-3 of PA (Fig. 11), the algorithm generates all logic faults per bridge and in step-4, it removes non-unique logic faults that are completely covered by identical logic fault at another $V_{dd}$ setting. Two logic faults at different $V_{dd}$ settings are identical if the input assignments to gates feeding the bridge are same along with the logic values interpreted by gates driven by the bridge. Such logic faults are distinguished by resistance range and $V_{dd}$ setting at which they appear. This step reduces total candidate logic faults and is used to speed up the search and bridge categorization process. Next, in steps 5 to 7, the algorithm computes the detection value DV(LF) for each candidate logic fault using Eq. (4)-(7).

Logic faults are sorted using their respective *Detection Value* (DV), and are then categorized into two different categories (Low $V_{dd}$ or Gray Zone) using the above mentioned set of criteria (Fig. 10). For bridges that falls into "Gray Zone", test generator [10] is invoked, which identifies exact $V_{dd}$ setting of each bridge location in Gray Zone. Bridges requiring higher $V_{dd}$ test are marked by $TargetBridgeList$, and this process is repeated for all bridge locations. Finally the PA returns to step-2 of the algorithm shown in Fig. 8 with updated $TargetBridgeList$ that is used to compute minimum number of gates for replacement using set covering technique.

It should be noted that the minimum set covering technique (step-2, Fig. 8) is useful for area minimization and has shown positive results for almost all the cases considered. However, in a few cases (less than 10), increasing the drive strength of a gate may make the fault redundant (un-detectable) at all $V_{dd}$

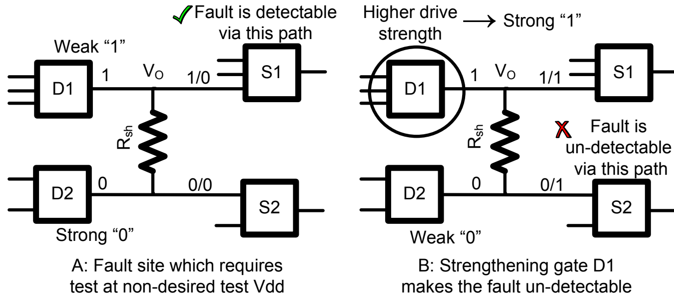


Fig. 12. Fault Redundancy due to gate selection by minimum set cover

settings. This is explained using Fig. 12, which shows a fault-site with driving gates D1 (driving high), D2 (driving low) while S1 and S2 are successor gates. Consider Fig. 12-A and assume that the output of D1 is a weak “1” and the output of D2 is a strong “0”. This results in S1 reading a faulty logic value at its input (shown as 1/0), while S2 reads the correct logic value in both good/faulty circuits. Furthermore, assume that the fault effect is propagated to the primary output via S1 and results in test generation at a non-desired voltage setting. Now consider Fig. 12-B, which shows that gate D1 is selected by the minimum set cover and is replaced by a gate with higher drive strength. Due to this change in drive strength, D1 outputs a strong “1” and D2 outputs a weak “0”, which results in S2 reading a faulty logic value (shown as 0/1) but this faulty logic value does not reach the primary output and therefore the fault becomes un-detectable. In such cases, the drive strength of both the driving gates (D1 and D2) is adjusted, such that higher resistance is exposed at the lowest $V_{dd}$ setting (Fig. 6) while ensuring that the fault is detectable. Therefore it is worth mentioning that for a few bridges, gate replacement and test generation may be repeated for fault detection at the lowest $V_{dd}$ setting.

## V. Experimental Results

The proposed technique for reducing test $V_{dd}$ settings is validated using ISCAS’85, ’89 and ITC 99 full scan circuits. The benchmark circuits are synthesized using ST Microelectronics 0.12 $\mu$m gate library. Synopsys Design Compiler$^{TM}$ (DC) is used for synthesis, as well as, to evaluate timing, area and power. Default options of DC are used for synthesis without specifying any time constraints on any design. The generated netlist is then used for gate identification and replacement to achieve single $V_{dd}$ test. The setup uses non-feedback bridges only and an exhaustive bridge list is generated by considering all possible pairs of nets in the netlist, up to a maximum of 10,000 pairs. This increases the total number of bridges for all the circuits and therefore, creates more challenging test cases than coupling capacitance based post-layout extracted bridge list. The number of bridge locations using coupling capacitance based extraction for ISCAS designs vary from 47 to 943, for c432 and s15850 respectively using the same gate library [22]. It should be noted that the number of extracted

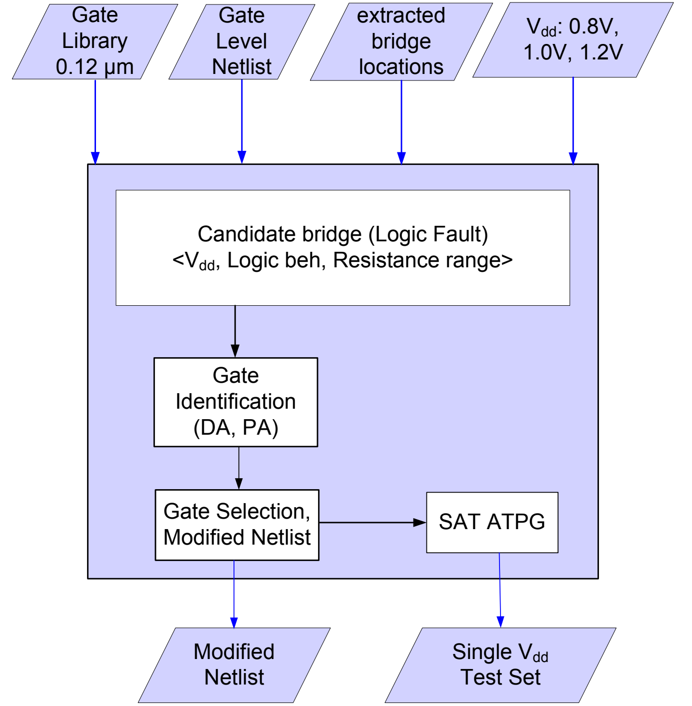


Fig. 13. Test generation flow

bridges depend on the type of gates available in the gate library that are used during synthesis. The use of compound gates (with up to 9 inputs) reduces the gate count (in comparison to 2 input AND/OR gates) resulting in reducing the number of extracted bridge locations. For the same reason, in a recent study reported in [23] the experimental setup uses the number of gates multiplied by 10 to determine the total number of random bridges to be considered.

All experiments are conducted using three $V_{dd}$ settings: 0.8V, 1.0V, and 1.2V. The selection of $V_{dd}$ settings is similar to a commercial microprocessor (TransMeta Crusoe TM5800) [24], that varies $V_{dd}$ settings from 0.9V to 1.3V and is synthesized using 0.13 $\mu$m cell library. The test generation flow used by the proposed gate sizing technique is shown in Fig. 13. The benchmarks used, total number of gates and extracted bridges for each circuit are tabulated in Table III. For these benchmarks, we conducted an experiment to determine the fault coverage only at 0.8V $V_{dd}$ setting (without applying test at higher test voltages) and our experimental results show that the fault coverage is 90% (as in case of s9234) or higher when the test is applied only at 0.8V $V_{dd}$ setting. In a recent study conducted using ISCAS benchmarks and layout extracted bridges, the fault coverage is 93.95% or higher when the test is applied only at 0.8V $V_{dd}$ setting [20]. This setup is used to conduct two sets of experiments.

### A. Test $V_{dd}$ Reduction Using GS

This experiment has two objectives: firstly, to show the impact of the proposed gate sizing (GS) algorithms *Determin-*

TABLE III
BENCHMARKS

| ISCAS 85, ISCAS 89 | | | ITC 99 | | |
|---|---|---|---|---|---|
| Ckt | # Gates | # Bridges | Ckt | # Gates | # Bridges |
| c432 | 93 | 1,094 | b01 | 26 | 142 |
| c1355 | 226 | 6,563 | b02 | 15 | 33 |
| c1908 | 205 | 7,986 | b03 | 63 | 350 |
| c2670 | 269 | 10,000 | b04 | 208 | 7,228 |
| c3540 | 439 | 10,000 | b05 | 315 | 10,000 |
| c7552 | 731 | 9,998 | b06 | 33 | 203 |
| s344 | 62 | 469 | b07 | 170 | 6,447 |
| s382 | 74 | 1,146 | b08 | 86 | 1,350 |
| s386 | 63 | 1,625 | b09 | 75 | 729 |
| s838 | 149 | 5,737 | b10 | 88 | 1,923 |
| s5378 | 578 | 9,933 | | | |
| s9234 | 434 | 10,000 | | | |
| s15850 | 1578 | 10,000 | | | |

*istic* and *Probabilistic*, to achieve single $V_{dd}$ test. Secondly, to compare the two proposed algorithms in terms of the number of gates replaced by each and their respective run-time. Table IV tabulates the total number of test $V_{dd}$ setting(s) required by the original design (labeled as Orig.) and compares it with those generated by TPI [10] (labeled, TPI)[2], and the proposed gate sizing algorithms (labeled DA, PA). As can be seen, the proposed algorithms (DA, PA) are able to achieve 100% fault coverage at a single test $V_{dd}$. This is unlike TPI, which requires two or more test $V_{dd}$ settings for a large number of circuits. Furthermore, TPI is unable to reduce any test $V_{dd}$, in case of c432 and c1908. The last column of Table IV shows the number of gates replaced by the proposed algorithms (DA, PA) and the number of test points (control/observation points) added by the TPI[3]. The number of gates replaced by the two algorithms ranges from 1-18, while the TPI has added up to 42 test points. The total number of gates replaced by the two GS algorithms (or added by the TPI) is shown in the last row of Table IV. The computation time of the proposed gate sizing algorithms is less than the TPI as it uses a simple set covering algorithm (Step-2, Fig. 8) for reducing the number of gates to be replaced, while the TPI uses a complex control point minimization algorithm [10]. The number of gates replaced by the PA is higher for certain circuits than the DA, as in case of c432. This is because of step-4 of the *Probabilistic* algorithm (PA) (Fig. 11) that removes non-unique logic faults to speedup the algorithm. To investigate the increased gate count, we analyzed a bridge in c432 that is marked for gate replacement by the PA. The bridge has the following three logic faults: LF1@1.2V (0-1000 Ω), LF2@0.8V (0-800 Ω) and LF3@0.8V (800-1200 Ω). Furthermore, LF1 and LF2 are identical in terms of input assignments to the gates feeding the bridge and the logic values interpreted by the gates fed by the bridge. Since LF1 covers higher resistance than LF2, the algorithm removes LF2. With the removal of LF2, the bridge is marked for gate replacement, as 1.2V $V_{dd}$ setting is required for complete resistance coverage.

We also analyzed the detectable resistance of neighboring nets (potential bridges) that may be affected by re-sizing of gates, and compared the detectable resistance range before and after gate sizing. It was found that around 75% of the bridges sharing the net driven by the re-sized gates has their detectable resistance range increased, while the resistance range has reduced for the rest of 25% bridges, however it is always $\geq 1K\Omega$ of detectable resistance after re-sizing. These bridges are not further re-sized because it was reported in [25] that around 96% of the bridges have their resistance range $\leq 1K\Omega$, however the proposed gate sizing technique can be repeated for such bridge locations, if higher detectability is required. The detectable resistance range is increased for a large majority of bridges because a bridge location consists of a large number of logic faults, where total number of logic faults depends on the number of possible combinations to activate the bridge and the number of gates fed by the bridge. For each bridge location, the test generator determines the total detectable resistance range using all possible logic faults. Therefore resistance range covered by an individual logic fault is less important than the total detectable resistance considering all logic faults. From the experimental results, it is evident that the proposed gate sizing technique guarantees single $V_{dd}$ test for all designs, while increasing the detectable resistance range for a large majority of bridges.

Table V shows the categorization of bridges to *Low* $V_{dd}$ and *Gray Zone* by step-9 of the *Probabilistic* algorithm (Fig. 11). As can be seen, for all the circuits, on average 45% and up to 71.6% of total bridges are identified as “Low $V_{dd}$” without using computationally expensive (SAT-solver based) test generator. These bridges are accurately identified by using probability based bridge identification criteria (Fig. 10).

To get an insight into the computation time of the proposed algorithms *(Probabilistic, Deterministic)*, see Table VI, which shows the comparison of total number of SAT calls and respective run-times of the two algorithms. The *Probabilistic* algorithm has significantly reduced the total number of computationally expensive SAT calls, for all benchmark designs, and on average it achieves 2.6X reduction in the total number of SAT calls in comparison to the *Deterministic* algorithm. The run-time (of PA and DA) is shown in column 3 of Table VI, and the last column shows the relative run-time by the PA in comparison to the DA. The last two rows show the sum and average of the number of SAT calls and run-time for all designs. As can be seen, the PA results in a significant speed up for a large majority of circuits (up to 50% time reduction, in case of c2670), this is especially noticeable for larger circuits for e.g., b04, c2670, c3540, c7552, s9234, and s15850 that show significant speed up. However, because of the setup time of PA (step-1 and steps 5-7 shown in Fig. 11), it is more time

[2]TPI results may vary from those reported in [10] because of difference in logic threshold voltages.

[3]The number of test points is the sum of control and observation points.

TABLE IV
RESULTS OF THE PROPOSED GATE SIZING ALGORITHMS (DA, PA) AND COMPARISON WITH TPI

| Ckt | Test $V_{dd}$ settings | | | No. of Gates | | |
|---|---|---|---|---|---|---|
| | Orig. | TPI [10] | DA, PA | DA | PA | TPI [10] |
| c432 | All* | All | 0.8V | 2 | 3 | 0 |
| c1355 | All | 0.8V | 0.8V | 4 | 4 | 10 |
| c1908 | 1.2V, 0.8V | 1.2V, 0.8V | 0.8V | 3 | 3 | 0 |
| c2670 | All | 1.2V, 0.8V | 0.8V | 6 | 6 | 19 |
| c3540 | All | 1.0V, 0.8V | 0.8V | 7 | 8 | 7 |
| c7552 | All | 0.8V | 0.8V | 1 | 1 | 1 |
| s344 | 1.2V, 0.8V | 0.8V | 0.8V | 1 | 1 | 1 |
| s382 | 1.2V, 0.8V | 0.8V | 0.8V | 2 | 2 | 5 |
| s386 | All | 1.2V, 0.8V | 0.8V | 7 | 7 | 4 |
| s838 | All | 0.8V | 0.8V | 14 | 14 | 28 |
| s5378 | All | 1.0V, 0.8V | 0.8V | 9 | 12 | 9 |
| s9234 | All | 1.0V, 0.8V | 0.8V | 6 | 13 | 2 |
| s15850 | All | 0.8V | 0.8V | 8 | 9 | 3 |
| b01 | All | 0.8V | 0.8V | 1 | 1 | 1 |
| b02 | 1.2V, 0.8V | 0.8V | 0.8V | 1 | 1 | 2 |
| b03 | 0.8V | 0.8V | 0.8V | 0 | 0 | 0 |
| b04 | All | 0.8V | 0.8V | 8 | 8 | 4 |
| b05 | All | 0.8V | 0.8V | 18 | 18 | 42 |
| b06 | 0.8V | 0.8V | 0.8V | 0 | 0 | 0 |
| b07 | All | 1.2V 0.8V | 0.8V | 9 | 10 | 10 |
| b08 | All | 0.8V | 0.8V | 4 | 4 | 8 |
| b09 | 1.2V, 0.8V | 0.8V | 0.8V | 2 | 2 | 2 |
| b10 | All | 0.8V | 0.8V | 4 | 5 | 5 |
| Total No. of Gates | | | | 117 | 132 | 163 |

*All = 0.8V, 1.0V, 1.2V

PA → Probabilistic Algorithm, DA → Deterministic Algorithm

efficient for larger designs and smaller designs do not show improvement, as is the case with s382, s386, b01, b08, and b10. The results presented in Table VI has shown encouraging results in terms of reducing SAT calls and minimizing run-time. It should be noted that the proposed GS technique is an offline process carried out only once during the design flow. However, we are currently investigating the use of faster ATPG engines to further improve the run-time of the algorithm.

### *B. Impact on Timing, Area and Power*

This experiment compares timing, area and power (dynamic and leakage) of the original design, the proposed Gate Sizing, and the TPI. Fig. 14 shows the timing performance. As can be seen, the proposed GS technique has little effect on timing when compared to the original design. This is because it replaces small number of gates. On average, for circuits shown in Table IV, it has replaced only 3% of the total number of gates. For some circuits the proposed GS technique has improved timing due to larger and faster gates. This is unlike the case with the TPI, where the timing was negatively affected because of the test points inserted in the critical path. For example, in case of s386, TPI has inserted 1 test point, and

TABLE V
BRIDGE CATEGORIZATION BY THE PROBABILISTIC ALGORITHM

| Ckt. | # Bridges | Prob. Search Space | |
|---|---|---|---|
| | | Gray Zone | Low $V_{dd}$ |
| c432 | 1,094 | 339 | 755 |
| c1355 | 6,563 | 3762 | 2,801 |
| c1908 | 7,986 | 4776 | 3,210 |
| c2670 | 10,000 | 2842 | 7,158 |
| c3540 | 10,000 | 3282 | 6,718 |
| c7552 | 9,998 | 6203 | 3795 |
| s344 | 469 | 234 | 235 |
| s382 | 1,146 | 803 | 343 |
| s386 | 1,625 | 751 | 874 |
| s838 | 5,737 | 3916 | 1821 |
| s5378 | 9,933 | 4886 | 5047 |
| s9234 | 10,000 | 5363 | 4637 |
| s15850 | 10,000 | 5899 | 4101 |
| b01 | 142 | 78 | 64 |
| b02 | 33 | 21 | 12 |
| b03 | 350 | 195 | 155 |
| b04 | 7,228 | 3497 | 3,731 |
| b05 | 10,000 | 4468 | 5,532 |
| b06 | 203 | 148 | 55 |
| b07 | 6,447 | 3489 | 2,958 |
| b08 | 1,350 | 860 | 490 |
| b09 | 729 | 542 | 187 |
| b10 | 1,923 | 1189 | 734 |

TABLE VI
TIMING COMPARISON OF DETERMINISTIC AND PROBABILISTIC ALGORITHMS

| Ckt. | Total SAT runs | | Time (min) | | $\frac{PA}{DA}$ |
|---|---|---|---|---|---|
| | PA | DA | PA | DA | |
| c432 | 1816 | 7379 | 1.52 | 1.93 | 0.78 |
| c1355 | 5821 | 19128 | 24.37 | 28.88 | 0.84 |
| c1908 | 8940 | 13766 | 23.83 | 25.52 | 0.93 |
| c2670 | 7416 | 50488 | 117.68 | 237.33 | 0.50 |
| c3540 | 10790 | 44908 | 75.62 | 135.75 | 0.56 |
| c7552 | 18454 | 32877 | 225.25 | 396.95 | 0.57 |
| s382 | 1363 | 2119 | 1.65 | 1.28 | 1.29 |
| s386 | 2190 | 7770 | 2.35 | 2.27 | 1.04 |
| s838 | 6187 | 14586 | 19.82 | 22.28 | 0.89 |
| s5378 | 9450 | 31269 | 310.00 | 336.20 | 0.92 |
| s9234 | 12669 | 37064 | 723.60 | 947.60 | 0.76 |
| s15850 | 12580 | 20598 | 4513.1 | 5896.18 | 0.77 |
| b01 | 166 | 338 | 0.05 | 0.02 | 3.0 |
| b02 | 44 | 63 | 0.02 | 0.02 | 1.0 |
| b04 | 6884 | 13803 | 33.78 | 41.15 | 0.82 |
| b07 | 8527 | 25631 | 29.32 | 31.45 | 0.93 |
| b08 | 3697 | 9883 | 2.08 | 1.93 | 1.08 |
| b10 | 4601 | 8258 | 2.03 | 1.55 | 1.31 |
| Total | 121595 | 339928 | 6106.1 | 8108.3 | 0.75 |
| Avg. | 6755.3 | 18884.9 | 339.2 | 450.5 | 0.75 |

PA → Probabilistic Algorithm, DA → Deterministic Algorithm

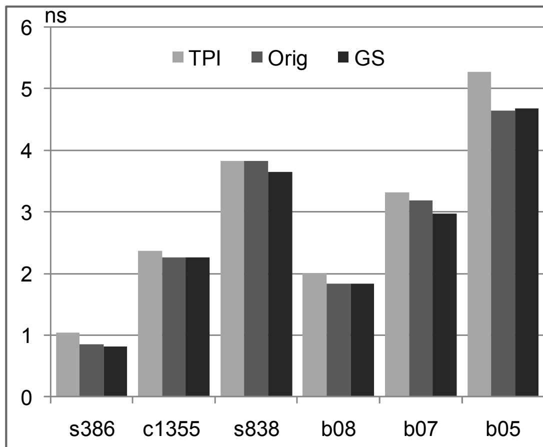


Fig. 14. Impact of Gate Sizing on timing performance and comparison with the original and the TPI [10].

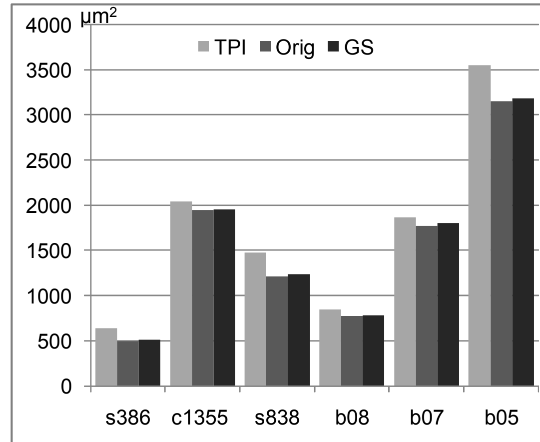


Fig. 15. Impact of Gate Sizing on area overhead and comparison with the original and the TPI [10].

converted a non-critical path into a critical path, while the GS technique has replaced a gate with a bigger gate, in the critical path, resulting in reduced timing. On comparing the delay of the longest path in the original and the GS modified designs, it was found that the longest path in the original design has a delay of 0.85 ns. On the other hand, the GS has replaced a gate in the longest path with a bigger gate thereby reducing the delay of the longest path to 0.78 ns (from 0.85 ns in the original design). As a result the second longest path in the original design with a delay of 0.82 ns, became the longest path in the GS modified design.

Similarly, comparison of area overhead is shown in Fig. 15 for the three designs. The proposed GS technique results in a slightly higher area overhead in comparison to original designs; however, it is less than the TPI for all circuits. Finally, comparison of dynamic and leakage power is shown in Fig. 16 and Fig. 17 respectively. It can be seen that the proposed gate sizing technique slightly increases the power budget in comparison to the original design; however, it is less than the TPI in all cases. High power consumption of the TPI is because of additional switching activity and leakage power of added test points. In case of GS, switching activity does not change in comparison to the original design but load capacitance and leakage power increases due to bigger gates, leading to higher dynamic and leakage power. The impact on leakage power can be reduced by using high-$V_t$ transistors in non-critical paths of the design [11].

## VI. Conclusion

This paper has proposed gate sizing to reduce test cost of multi-$V_{dd}$ designs with bridge defects, by reducing the number of test voltage settings. It has been shown, that it is possible to achieve 100% fault coverage using a single $V_{dd}$ test setting. This represents an improvement on the recently proposed TPI technique [10] which mostly requires two or more test $V_{dd}$ settings to achieve complete fault coverage. In this paper, we

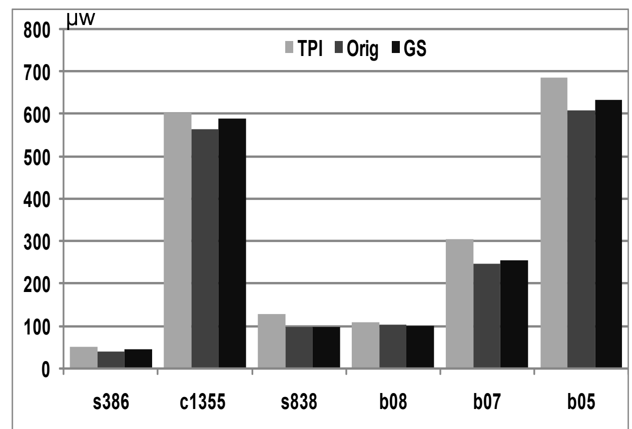


Fig. 16. Impact of Gate Sizing on dynamic power and comparison with the original and the TPI [10].

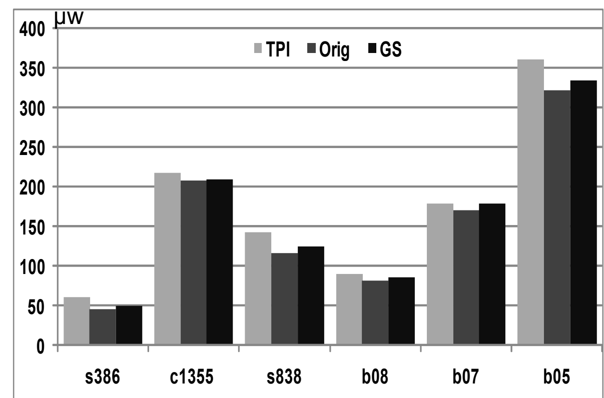


Fig. 17. Impact of Gate Sizing on leakage power and comparison with the original and the TPI [10].

have presented two algorithms to identify gates for replacement to achieve single $V_{dd}$ test, these algorithms show a trade-off between accuracy and speed. The proposed gate sizing technique has little effect on timing, area and power when compared with the original design (prior to gate sizing) and outperforms the TPI in terms of these three parameters.

## ACKNOWLEDGMENT

The authors would like to thank the reviewers for their comments which helped improve the quality of this paper.

## REFERENCES

[1] F. Ferguson and J. Shen, "Extraction and simulation of realistic CMOS faults using inductive fault analysis," in *Proceedings IEEE International Test Conference (ITC)*, Sep. 1988, pp. 475–484.
[2] M. Renovell, P. Huc, and Y. Bertrand, "Bridging fault coverage improvement by power supply control," in *Proceedings of the VLSI Test Symposium (VTS)*, Apr. 1996, pp. 338–343.
[3] V. R. Sar-Dessai and D. M. H. Walker, "Resistive bridge fault modeling, simulation and test generation," in *Proceedings of the International Test Conference (ITC)*, Sep. 1999, pp. 596–605.
[4] P. Engelke, I. Polian, M. Renovell, B. Seshadri, and B. Becker, "The pros and cons of very-low-voltage testing: an analysis based on resistive bridging faults," in *Proceedings of the VLSI Test Symposium (VTS)*, Apr. 2004, pp. 171–178.
[5] P. Engelke, I. Polian, M. Renovell, and B. Becker, "Simulating resistive bridging and stuck-at faults," *IEEE Transactions on Computer-Aided Design*, vol. 25, no. 10, pp. 2181–2192, Oct. 2006.
[6] M. Renovell, F. Azais, and Y. Bertrand, "Detection of defects using fault model oriented sequences," *Journal of Electronic Testing: Theory and Applications*, vol. 14, no. 1–2, pp. 13–22, Feb. 1999.
[7] T. Maeda and K. Kinoshita, "Precise test generation for resistive bridging faults of CMOS combinational circuits," in *Proceedings IEEE International Test Conference (ITC)*, Oct. 2000, pp. 510–519.
[8] G. Chen, S. Reddy, I. Pomeranz, J. Rajski, P. Engelke, and B. Becker, "An unified fault model and test generation procedure for interconnect open and bridges," in *Proceedings IEEE European Test Symposium (ETS)*, May 2005, pp. 22–27.
[9] P. Engelke, I. Polian, M. Renovell, and B. Becker, "Automatic test pattern generation for resistive bridging faults," *Journal of Electronic Testing: Theory and Applications*, vol. 22, no. 1, pp. 61–69, Feb. 2006.
[10] S. Khursheed, U. Ingelsson, P. Rosinger, B. M. Al-Hashimi, and P. Harrod, "Bridging fault test method with adaptive power management awareness," in *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 6, Jun. 2008, pp. 1117–1127.
[11] M. Keating, D. Flynn, R. Aitken, A. Gibbons, and K. Shi, *Low Power Methodology Manual: For System-on-Chip Design*. Springer, 2007.
[12] J. Segura, J. Rossello, J. Morra, and H. Sigg, "A variable threshold voltage inverter for CMOS programmable logic circuits," *Solid-State Circuits, IEEE Journal of*, vol. 33, no. 8, pp. 1262–1265, Aug 1998.
[13] M. Abramovici, M. Breuer, and A. Friedman, *Digital Systems Testing and Testable Designs*. IEEE Press, 1998.
[14] N. A. Touba and E. J. McCluskey, "Test point insertion based on path tracing," in *Proceedings IEEE VLSI Test Symposium (VTS)*, Apr. 1996, pp. 2–8.
[15] I. Pomeranz and S. Reddy, "Design-for-testability for path delay faults in large combinational circuits using test points," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, no. 4, pp. 333–343, Apr. 1998.
[16] Q. Zhou and K. Mohanram, "Gate Sizing to Radiation harden Combinational logic," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 25, no. 1, pp. 155–166, Jan. 2006.
[17] R. Rodriguez-Montanes, D. Arumi, and J. Figueras, "Effectiveness of very low voltage testing of bridging defects," *Electronics Letters*, vol. 42, no. 19, pp. 1083–1084, Sept. 2006.
[18] N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI design: a systems perspective*. Addison-Wesley Publishing Co., 1994.
[19] "zChaff Boolean Satisfiability problem solver," Oct. 2009. [Online]. Available: http://www.princeton.edu/˜chaff/zchaff.html
[20] U. Ingelsson, "Investigation into voltage and process variation-aware manufacturing test," *Ph.D Thesis, University of Southampton*, July 2009. [Online]. Available: http://eprints.soton.ac.uk/67440/
[21] S. Chandra and J. Patel, "Experimental evaluation of testability measures for test generation (logic circuits)," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 8, no. 1, pp. 93–97, Jan 1989.
[22] S. Khursheed, B. M. Al-Hashimi, S. M. Reddy, and P. Harrod, "Diagnosis of multiple-voltage design with bridge defect," *IEEE Transactions on Computer-Aided Design*, vol. 28, no. 3, Mar. 2009. [Online]. Available: http://eprints.ecs.soton.ac.uk/16964/
[23] P. Engelke, B. Becker, M. Renovell, J. Schloeffel, B. Braitling, and I. Polian, "SUPERB: Simulator utilizing parallel evaluation of resistive bridges," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 14, no. 4, pp. 1–21, Aug. 2009.
[24] "TransMeta Crusoe TM5800," Oct. 2009. [Online]. Available: http://www.cpu-world.com/CPUs/TM5800/
[25] R. Rodriguez-Montanes, E. Bruis, and J. Figueras, "Bridging defects resistance measurements in a CMOS process," *Proceedings of the International Test Conference (ITC)*, pp. 892–899, Sep 1992.

**Saqib Khursheed** received the B.E. degree in Computer Engineering from NED University, Karachi, Pakistan, in 2001, the M.Sc. degree in Computer Engineering from King Fahd University (KFUPM), Dhahran, Saudi Arabia, in 2004, and the Ph.D. degree in Electronics and Electrical Engineering from University of Southampton, Southampton, U.K., in 2010.

From 2005 to 2007, he served as a Lecturer with KFUPM. Currently he is working as a Research Fellow in the School of Electronics and Computer Science, University of Southampton. His research interests include fault modeling, diagnosis and test generation for deep submicron defects.

**Bashir M. Al-Hashimi** (M'99-SM'01-F'09) received the B.Sc. degree (with 1st-class classification) in Electrical and Electronics Engineering from the University of Bath, UK, in 1984 and the Ph.D. degree from York University, UK, in 1989. Following this he worked in the microelectronics design industry and in 1999, he joined the School of Electronics and Computer Science, Southampton University, UK, where he is currently a Professor of Computer Engineering and Director of the Pervasive Systems Center. He has authored one book on SPICE simulation, (CRC Press, 1995), and coauthored two books, Power Constrained Testing of VLSI circuits (Springer, 2002), and System-Level Design Techniques for Energy-Efficient Embedded Systems (Springer, 2004). He edited the book, System-on-Chip: Next Generation Electronics (IEE Press, 2006). He has published over 240 papers in journals and refereed conference proceedings.

Prof. Al-Hashimi is a Fellow of the IEE and a Fellow of the British Computer Society. He is the Editor-in-Chief of the IEE Proceedings: Computers and Digital Techniques, and a member of the editorial board of the Journal of Electronic Testing: Theory and Applications (JETTA), Journal of Embedded Computing, and Journal of Low Power Electronics. He was the General Chair of the 11th IEEE European Test Symposium (UK 2006), Technical-Programme Chair of DATE 2009, and the General Chair of DATE 2011. He is the coauthor of two Best Paper Awards: the James Beausang at the ITC 2000, relating to low power BIST for RTL data paths, and at the CODES-ISSS Symposium 2009, relating to low-energy fault-tolerance techniques. He is a co-author of a paper on test data compression which has been selected for a Springer book featuring the most influential work over the ten years of the DATE conference.

**Krishnendu Chakrabarty** (M'92-SM'00-F'08) received the B. Tech. degree from the Indian Institute of Technology, Kharagpur, in 1990, and the M.S.E. and Ph.D. degrees from the University of Michigan, Ann Arbor, in 1992 and 1995, respectively. He is now Professor of Electrical and Computer Engineering at Duke University. He is also a member of the Chair Professor Group (honorary position) in Software Theory at the School of Software, Tsinghua University, Beijing, China. Prof. Chakrabarty is a recipient of the National Science Foundation Early Faculty (CAREER) award, the Office of Naval Research Young Investigator award, the Humboldt Research Fellowship from the Alexander von Humboldt Foundation, Germany, and several best papers awards at IEEE conferences. His current research projects include: testing and design-for-testability of integrated circuits; digital microfluidics and biochips, circuits and systems based on DNA self-assembly, and wireless sensor networks. He has authored nine books on these topics, published over 330 papers in journals and refereed conference proceedings, and given over 130 invited, keynote, and plenary talks.

Prof. Chakrabarty is a Fellow of IEEE, a Golden Core Member of the IEEE Computer Society, and a Distinguished Engineer of ACM. He was a 2009 Invitational Fellow of the Japan Society for the Promotion of Science (JSPS). He is a recipient of the 2008 Duke University Graduate School Dean's Award for excellence in mentoring, and the 2010 Capers and Marion McDonald Award for Excellence in Mentoring and Advising, Pratt School of Engineering, Duke University. He served as a Distinguished Visitor of the IEEE Computer Society during 2005-2007, and as a Distinguished Lecturer of the IEEE Circuits and Systems Society during 2006-2007. Currently he serves as an ACM Distinguished Speaker, as well as a Distinguished Visitor of the IEEE Computer Society for 2010-2012. He is the Editor-in-Chief for IEEE Design & Test of Computers and for ACM Journal on Emerging Technologies in Computing Systems. Prof. Chakrabarty is also an Associate Editor of IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on Circuits and Systems II, and IEEE Transactions on Biomedical Circuits and Systems. He serves as an Editor of the Journal of Electronic Testing: Theory and Applications (JETTA).

**Peter Harrod** (M'80-SM'99) graduated with a BSc(Eng) from the University of the Witwatersrand in 1976 and with MSc and PhD degrees from the University of Manchester Institute of Science and Technology in 1978 and 1982 respectively.

He is a Consultant Engineer at ARM Ltd in Cambridge, UK where he works on embedded CPU designs. He has a particular interest in the areas of design for test and debug. Dr. Harrod is a Fellow of the IET and has served on several IEEE standards and conference program committees.