Low-Capture-Switching-Activity Test Generation for Reducing IR-Drop in At-Speed Scan Testing

Xiaoqing Wen [1], Kohei Miyase [1], Tatsuya Suzuki [2], Seiji Kajihara [1], Laung-Terng Wang [3]
Kewal K. Saluja [4], and Kozo Kinoshita [5]

[1] Kyushu Institute of Technology
[2] Denso Techno Co., Ltd.
[3] SynTest Technologies, Inc.
[4] University of Wisconsin - Madison
[5] Osaka Gakuin University

Abstract. At-speed scan testing, based on ATPG and ATE, is indispensable to guarantee timing-related test quality in the DSM era. However, at-speed scan testing may incur yield loss due to excessive IR-drop caused by high test (shift & capture) switching activity. This paper discusses the mechanism of circuit malfunction due to IR-drop, and summarizes general approaches to reducing switching activity, by which highlights the problem of current solutions, i.e. only reducing switching activity for one capture while the widely used at-speed scan testing based on the launch-off-capture scheme uses two captures. This paper then proposes a novel *X*-filling method, called ***double-capture*** (***DC***) ***X-filling***, for generating test vectors with low and balanced capture switching activity for two captures. Applicable to dynamic & static compaction in any ATPG system, DC *X*-filling can reduce IR-drop, and thus yield loss, without any circuit/clock modification, timing/circuit overhead, fault coverage loss, and additional design effort.

Keywords: at-speed scan testing, capture switching activity, *X*-filling, test cube, ATPG, low power testing

1. Introduction

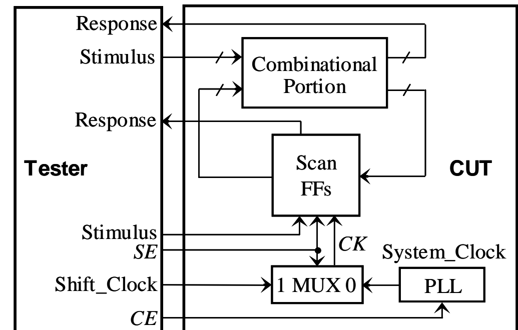
External scan testing, or simply ***scan testing***, is conducted by automatic test equipment (ATE) on a full-scan circuit with test vectors obtained through automatic test pattern generation (ATPG). It is the most widely used test methodology for achieving satisfactory test quality at acceptable test costs [1].

In a full-scan circuit, all functional flip-flops (FFs) are replaced with scan FFs of two operational modes. In ***shift mode***, scan FFs operate collectively as one or more shift registers (called scan chains), through which a test vector is applied by shift-in or a test response is obtained by shift-out, for the combinational portion of the circuit. In ***capture mode***, scan FFs operate individually as functional FFs and load the test response of the combinational portion for a test vector into themselves. This way, the problem of testing a sequential circuit is reduced to that of testing its combinational portion, in that it is enough to generate test vectors only for the combinational portion [1].

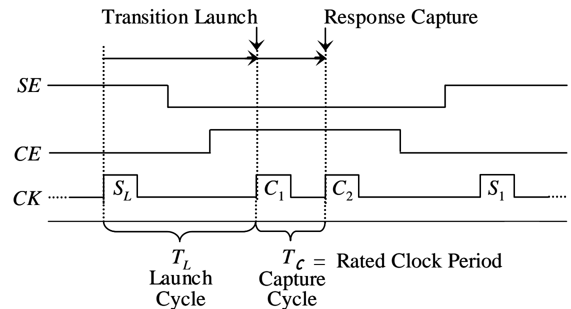
There are two types of scan testing: slow-speed and at-speed, depending on the interval between the time when a test stimulus is applied and the time when the corresponding test response is captured [2]. If the interval is equal to the rated clock period, the scan testing is called ***at-speed***, which is used to check for excessive delays caused by timing-related defects [3].

As feature sizes shrink into the deep submicron (DSM) scale and circuit speeds grow into the GHz domain, more chips fail due to timing-related defects [4]. This has made at-speed scan testing mandatory to guarantee delay test quality.

Fig. 1 shows a typical at-speed scan testing system, based on on-chip phase-locked loop (PLL) and the ***launch-off-capture*** clocking scheme [2, 3]. *SE* and *CE* are the scan and capture enable signals, respectively. A test vector is applied in shift mode (*SE* = 1) via a series of shift clock pulses with S_L as the last one. In capture mode (*SE* = 0), the PLL responds to the rising edge of *CE* to issue two capture clock pulses, C_1 and C_2, with the capture cycle T_C being equal to the rated clock period. C_1 launches transitions by the difference between the values shifted-in by S_L and the values captured by C_1, while C_2 captures the circuit response to the transitions in the at-speed manner. This way, timing-related defects can be detected.



(a) System view



(b) Launch-off-capture clocking scheme

Fig. 1. **At-speed scan testing system.**

However, the applicability of at-speed scan testing is being severely challenged by ***test-induced yield loss*** [5]. The reason is that the switching activity in scan testing is much higher than that in the functional operation [6, 7], due to three major factors: (1) high switching activity of test operations, (2) test vectors ignoring functional constraints, and (3) test clocking not allowed in the functional operation. Test-induced yield loss is caused by excessive ***instantaneous*** test power dissipation in both shift and capture mode, because FFs and/or PLL may malfunction due to power supply voltage drop and ground bounce [8-10]. This problem is rapidly worsening as feature sizes shrink below 0.18 micron.

1.1. Motivation

1.1.1. IR-Drop and Yield Loss

As illustrated in Fig. 2, a circuit can be seen as a network of cells (logic gates and FFs) existing between VDD and VSS grids. Whenever a cell switches its output, a dynamic current (I) will flow through the equivalent resistance (R) of the VDD/VSS grids and the cell network, causing a drop in the effective power supply voltage to a cell. This is called ***IR-drop*** [5, 8].

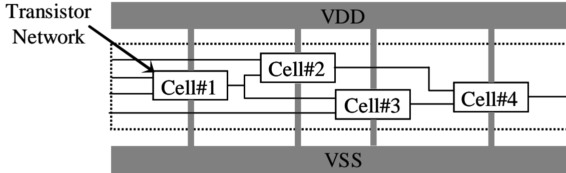


***Fig. 2*. Illustrative circuit with VDD/VSS grids.**

As illustrated in Fig. 3, high simultaneous switching activity (①) causes excessive IR-drop (②), resulting in performance degradation of transistors. For a FF consisting of degraded transistors, ***current-cycle malfunction*** (*CCM*) (③) may occur in the same cycle where the switching activity occurs. In addition, for a gate consisting of degraded transistors, its delay will increase, resulting in increased path delay in a circuit. A 10% IR-drop can increase path delay by as much as 30% [6]. Increased path delay may violate timing requirements at some FFs in the next cycle, resulting in ***next-cycle malfunction*** (*NCM*) (④). Note that the clock pulse P in Fig. 3 can be a shift clock pulse or a capture clock pulse, indicating that test-induced yield loss may occur in shift mode or capture mode or both.

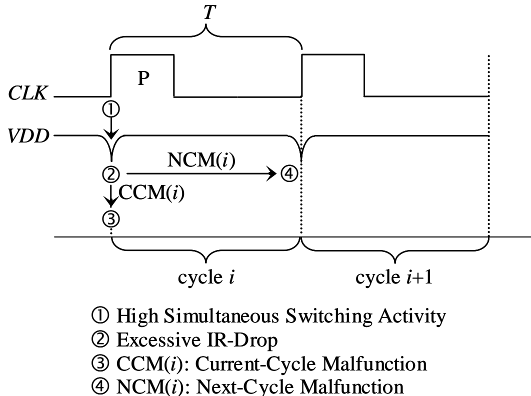


***Fig. 3*. Mechanism of IR-drop-induced malfunction.**

Note that the capture cycle (T_C in Fig. 1 (b)) is equal to the rated clock period, which is very short for a high-speed circuit. As a result, the risk for IR-drop-increased delay to cause timing violations, and thus next-cycle malfunction, is high. This indicates that at-speed scan testing is vulnerable to IR-drop.

1.1.2. Related Work

Excessive IR-drop during scan testing should be reduced by lowering switching activity. There are two types of switching activity reduction: ***shift switching activity reduction*** and ***capture switching activity reduction***, and two basic strategies: ***circuit/clock modification*** and ***test data manipulation***. Thus, as summarized in Table 1, there are four approaches, A~D, for switching activity reduction.

***Table 1*. Approaches to Switching Activity Reduction**

	Switching Activity Reduction Target		Advantages / Disadvantages
	Shift	Capture	
A	*Circuit /Clock*	*Circuit /Clock*	Significant reduction effect for IR-drop & test heat
			High overhead, fault coverage loss, test size increase, ATPG change, implementation difficulty
B	*Data*	*Data*	No circuit overhead
			Possibly insufficient reduction effect for IR-drop and test heat
C	*Data*	*Circuit /Clock*	Less overhead
			Fault coverage loss, test size increase, implementation difficulty, ATPG change
D	*Circuit /Clock*	*Data*	Good reduction effect for IR-drop & test heat, no fault coverage loss, no or minor ATPG change, easy implementation
			Highly effective capture IR-drop reduction solution needed

I. Shift Switching Activity Reduction

In shift mode, it is only necessary to guarantee that scan chains function properly. Thus, any circuit/clock change does not affect ATPG complexity and circuit timing. This makes shift switching activity reduction easy if the circuit/clock modification strategy [6, 7, 11-14] is used. This strategy can achieve significant and predictable reduction results, which are needed not only for IR-drop reduction but also for test heat reduction. Since many effective and practical solutions have already been proposed for shift switching activity reduction, this paper focuses on the less-researched issue of capture switching activity reduction.

II. Capture Switching Activity Reduction

Capture is directly related to circuit timing and ATPG complexity. Even a slight change to a circuit or its clocking scheme in capture mode may cause timing problems, substantial ATPG change, fault coverage loss, test data increase, or additional design efforts [15, 16]. Thus, it is preferable to use test data manipulation to reduce capture switching activity.

There are two approaches to test data manipulation: ***ATPG-based*** and ***X-filling***. The former directly generates logic values in a test vector so that switching activity is reduced [17, 18], while the latter assigns proper logic values to the unspecified bits (***X-bits***) in a ***test cube*** so that the resulting fully-specified ***test vector*** has lower switching activity [19-23].

Generally, *X*-filling is a preferred approach, since it can be used as a post-processing step in any ATPG flow, which requires little or no change to the ATPG algorithm. A few

X-filling methods have been proposed for reducing capture switching activity. For example, the method in [19] tries to minimize the output value difference at FFs before and after a capture; the method [21] is scalable in reducing average and peak switching activity during the capture cycle (T_C in Fig. 1 (b)) for a launch-off-capture-based delay test vector, by using a fast logic-value determination procedure based on signal probabilities; and the method in [22] tries to minimize the switching probability at each cell (gate or FF).

The biggest limitation of previous *X*-filling methods is that they only try to reduce switching activity for ***one*** capture pulse. Obviously, this may lead to unsatisfactory effect in IR-drop reduction for at-speed scan testing based on the launch-off-capture clocking scheme, which uses ***two*** capture pulses. This problem is addressed in this paper.

1.2. Our Approach

This paper proposes a novel approach to reducing capture switching activity for at-speed scan testing with the launch-off-capture clocking scheme. The basic idea is to make use of test cubes, i.e., input vectors with unspecified bits (*X*-bits), which can be obtained either during ATPG or after ATPG. We propose a novel method, called ***double-capture* (*DC*) *X-filling***, for algorithmically assigning 0's and 1's to the *X*-bits in a test cube so as to reduce its capture switching activity caused by the ***two*** capture pulses in the launch-off-capture clocking scheme of at-speed scan testing, in a balanced manner.

The DC *X*-filling method can be easily incorporated into dynamic or static compaction of any test generation flow, and the resulting "cool" test vectors can reduce capture switching activity without any area, timing, or fault coverage impact. As a result, test yield loss in capture mode can be efficiently lowered, thus greatly improving the applicability of at-speed scan testing with the launch-off-capture clocking scheme.

This paper is an extension of the basic *X*-filling technique proposed in [26]. In previous *X*-filling techniques for capture switching activity reduction, including [21] and [26], the WSA metric is used for evaluation. However, this metric ignores the distribution of capture switching activity, and thus cannot directly reduce its peak. This paper uses the FF-SWA metric, making the reduction efforts more directly targeted at FFs since the peak of capture switching activity of many circuits is related to FFs. In addition, the basic *X*-filling algorithm proposed in [26] is improved by taking the weight, i.e. the number of fanout branches + 1, of a FF into consideration. Furthermore, the feasibility of applying the proposed *X*-filling technique as a post-ATPG procedure is established by experimental results.

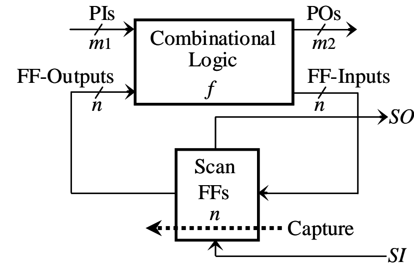
The rest of the paper is organized as follows: Section 2 formalizes the problem of capture switching activity reduction. Section 3 presents the DC *X*-filling method, and Section 4 describes its application in ATPG. Section 5 shows experimental results, and Section 6 concludes the paper.

2. Problem Formalization

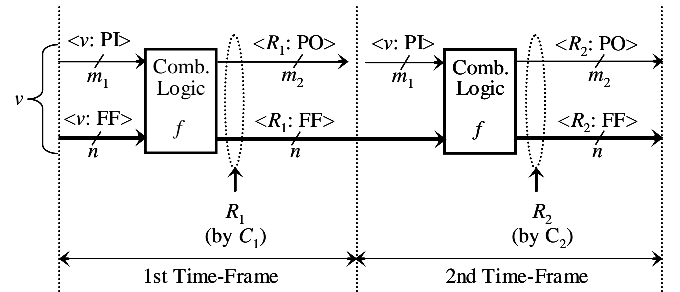
2.1. Circuit Model

Fig. 4 (a) shows a full-scan circuit, and Fig. 4 (b) shows its circuit model under the launch-off-capture clocking scheme illustrated in Fig. 1 (b).

In Fig. 4 (b), v is the input vector in the first time-frame. The bits in v related to primary inputs and FFs are denoted by <v: PI> and <v: FF>, respectively. The functional response of the combinational logic to v is $R_1 = f(v)$. The bits in R_1 related to primary outputs and FFs are denoted by <R_1: PO> and <R_1: FF>, respectively. <R_1: FF> is loaded by the first capture C_1 to replace <v: FF>, and <<v: PI>, <R_1: FF>> becomes the input vector in the second time-frame. The functional response of the combinational logic to this input vector is $R_2 = f($<<v: PI>, <R_1: FF>>$)$. The bits in R_2 related to primary outputs and FFs are denoted by <R_2: PO> and <R_2: FF>, respectively. <R_2: FF> is loaded by the second capture C_2 to replace <R_1: FF>. Note that the primary inputs do not change their values in the second time-frame, following the common industry practice [3].



(a) Full-scan circuit



(b) Circuit model

***Fig. 4*. Circuit model for the launch-off-capture scheme.**

2.2. DC X-Filling Problem

As shown in Fig. 4 (b), <v: FF> is replaced by <R_1: FF> at the first capture C_1, and <R_1: FF> is replaced by <R_2: FF> at the second capture C_2. If <v: FF> ≠ <R_1: FF> and/or <R_1: FF> ≠ <R_2: FF>, ***capture transitions*** will occur at the outputs of corresponding FFs for C_1 and/or C_2, resulting in capture switching activity.

Extensive gate-level simulation on ISCAS'89 circuits has confirmed that the dominant portion of capture switching activity occurs at FFs. For example, Fig. 5 shows the switching activity in the capture cycle (T_C in Fig. 1 (b)) at each logic level for s38584, whose maximum logic level is 24. Here, the logic level of a primary input or FF is set to 0, and the logic level of a logic gate is determined as 1 + (the maximum logic level among its inputs). Clearly, whether IR-drop can be sufficiently reduced largely depends on whether the switching activity at FFs can be sufficiently reduced. Therefore, it is reasonable to try to achieve capture switching activity reduction by reducing the number of capture transitions at FFs.

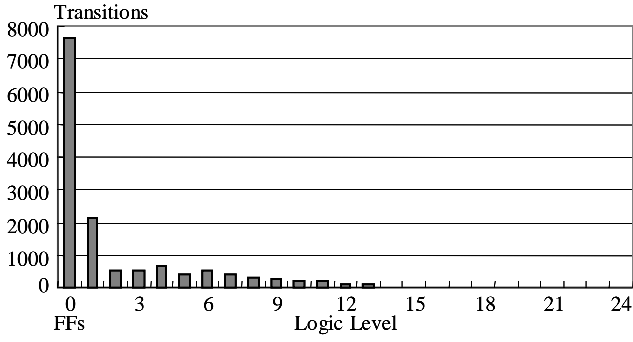


***Fig. 5*. Capture switching activity distribution of s38584.**

In Fig. 4 (b), it is clear that capture transitions for C_1 and C_2 can be reduced by minimizing the Hamming distance between <*v*: FF> and <R_1: FF> as well as the Hamming distance between <R_1: FF> and <R_2: FF>. In addition, this should be conducted in a balanced manner with respect to C_1 and C_2. Therefore, the DC *X*-filling problem can be formalized as follows:

DC X-Filling Problem: Given a test cube *v* with unspecified bits (*X*-bits) for a full-scan circuit with respect to the launch-off-capture clocking scheme using two captures (C_1 and C_2), assign logic values to all *X*-bits in *v* so that for the resulting fully-specified test vector, N_1, N_2, and $|N_1-N_2|$ are all minimized. Here, N_1 and N_2 are the numbers of capture transitions for the first capture C_1 and the second capture C_2, respectively.

3. DC *X*-Filling Algorithm

3.1. Basic Concepts

In Fig. 4 (b), suppose that *x*, *y*, and *z* are three bits in <*v*: FF>, <R_1: FF>, and < R_2: FF>, respectively, corresponding to the same FF. <*x*, *y*, *z*> is called a ***3-bit-tuple***. In addition, depending on how *X*-bits appear, 3-bit-tuples can be classified into 8 ***X-types*** as summarized in Table 2.

***Table 2*. *X*-Types**

Type	# of *X*'s	<*v*: FF>	<R_1: FF>	<R_2: FF>	Target Capture
1	0	b_1	b_2	b_3	N/A
2	1	X	b_2	b_3	C_1
3		b_1	X	b_3	C_1, C_2
4		b_1	b_2	X	C_2
5	2	b_1	X	X	C_1, C_2
6		X	b_2	X	C_1, C_2
7		X	X	b_3	C_1, C_2
8	3	X	X	X	C_1, C_2

(b_1, b_2, b_3: 0 or 1)

A 3-bit-tuple of Type-2 ~ Type-8 has at least one *X*-bit and can be used for capture transition reduction. In addition, 3-bit-tuples of different types may reduce capture transitions for different captures. For example, a Type-3 3-bit-tuple <b_1, *X*, b_3>, where $b_1 \neq b_3$, can be used to reduce capture transitions for C_1 if *X*-bit takes logic value b_1, or for C_2 if *X*-bit takes logic value b_3. This information is shown in Table 2 under "Target Capture".

3.2. General Procedure

Fig. 6 shows the DC *X*-filling procedure, which processes one 3-bit-tuple in each iteration in the following steps:

① ***X-Type Determination*** is to determine the *X*-types of all 3-bit-tuples.
② ***Target Capture Selection*** is to determine which capture, C_1 or C_2, should be targeted in the current iteration, in order to guarantee that capture transitions for C_1 and C_2 are reduced in a balanced manner.
③ ***Target 3-Bit-Tuple Selection*** is to pick up one 3-bit-tuple that has at least one *X*-bit and has the highest possibility of successfully reducing capture transitions for the target capture selected at ②.
④ ***X-Filling Operation*** uses ***assignment*** and ***justification*** techniques to find proper logic values for the *X*-bits in the test cube *v* to make necessary logic value(s) appear at the *X*-bit(s) in the target 3-bit-tuple selected at ③ in order to reduce capture transitions for the target capture selected at ②.
⑤ ***Logic Simulation*** is to propagate the impact of the newly determined logic values at *X*-bits in *v* to the whole circuit, since the *X*-types of some 3-bit-tuples may change accordingly.

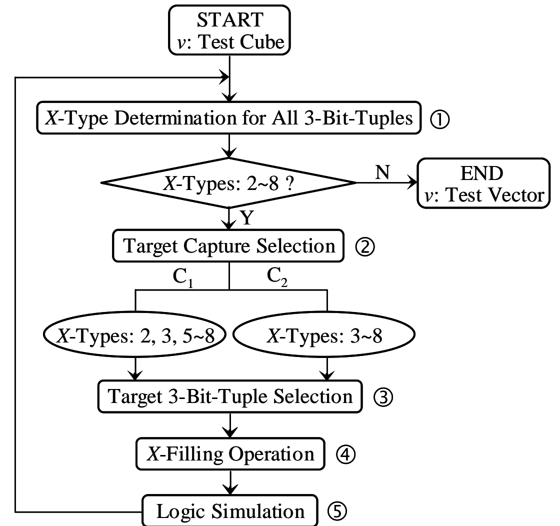
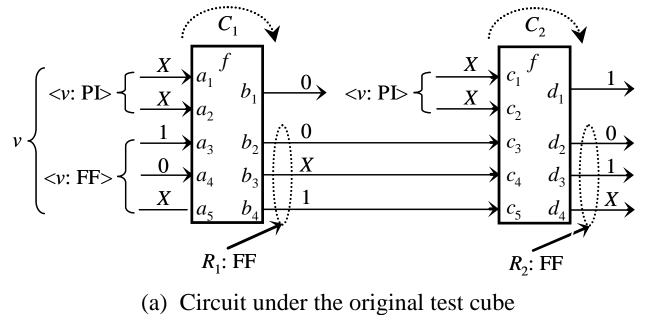


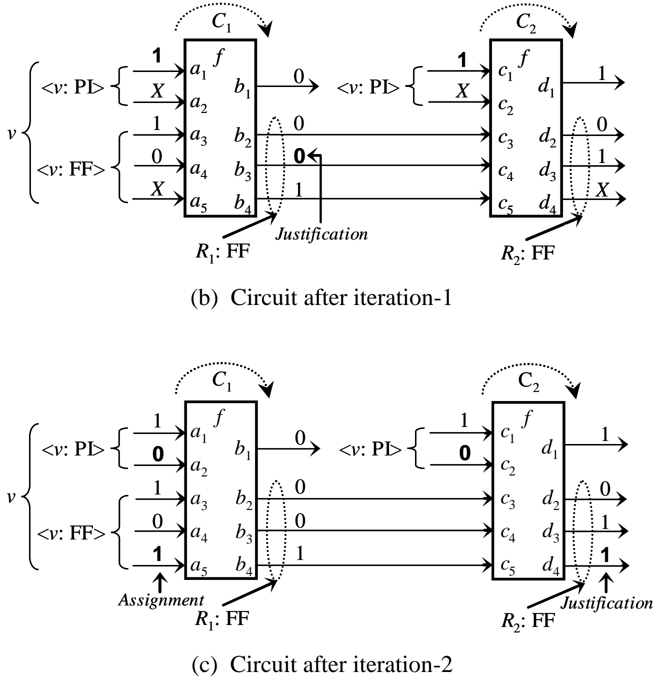
***Fig. 6*. DC *X*-filling procedure.**

3.3. Example

An example of DC *X*-filling is shown in Fig. 7, which consists of two DC *X*-filling iterations.



(a) Circuit under the original test cube



(b) Circuit after iteration-1

(c) Circuit after iteration-2

***Fig. 7*. Example of DC *X*-filling.**

***Iteration*-1**:

In Fig. 7 (a), there is one capture transition at C_1 but no capture transition at C_2 for <1, 0, 0>. Capture transition information for <0, *X*, 1> and <*X*, 1, *X*> is unclear due to *X*-bits. Thus, in order to achieve balanced capture transition reduction, it is necessary to reduce capture transitions at C_1. Although both <0, *X*, 1> and <*X*, 1, *X*> may achieve this goal, <0, *X*, 1> is selected since it has only one *X*-bit, making it easier to bring 0 to the *X*-bit to reduce capture transitions at C_1. Logic values in *v* needed for this goal are found by justifying 0 on b_3. The result is 1 for the *X*-bits on a_1 and c_1, as shown in Fig. 7 (b).

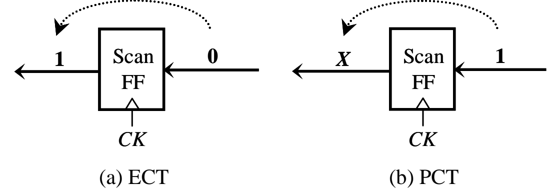
***Iteration*-2**:

In Fig. 7 (b), there is one capture transition at C_1 for <1, 0, 0> and one capture transition at C_2 for <0, 0, 1>. Capture transition information for <*X*, 1, *X*> is unclear due to *X*-bits. Thus, it is necessary to reduce capture transitions for both C_1 and C_2. <*X*, 1, *X*> is the only 3-bit-tuple for this goal, requiring 1 to appear on both *X*-bits in <*X*, 1, *X*>. Logic values in *v* needed for this goal are found by assigning 1 to the *X*-bit on a_5 and justifying 1 on d_4. The result of justification is 0 for the *X*-bit on a_2 and c_2. This way, <*X*, 1, *X*> becomes <1, 1, 1>, which causes no capture transition at both C_1 and C_2.

After two iterations of DC *X*-filling, the test cube *v* <*X*, *X*, 1, 0, *X*> becomes a fully-specified test vector <**1**, **0**, 1, 0, **1**>, as shown in Fig. 7 (c).

3.4. Target Capture Selection

The DC *X*-filling method dynamically selects a target capture in order to achieve a balanced reduction of capture transitions for the first capture C_1 and for the second capture C_2. The target capture selection heuristic is based on the ***total estimated capture transition activity*** (***TECTA***), which is calculated from ***existing capture transitions*** (***ECTs***) and ***potential capture transitions*** (***PCTs***) as illustrated in Fig. 8.



(a) ECT (b) PCT

***Fig. 8*. Existing and potential capture transitions.**

An ECT is a capture transition in the case where a logic value is loaded into a scan FF to replace a different logic value. An example of ECT is shown in Fig. 8 (a). On the other hand, a PCT is a capture transition in the case where a value v_2 is loaded into a scan FF to replace another value v_1, where either v_1 or v_2 or both are *X*-bits. An example of PCT is shown in Fig. 8 (b).

The probability of an ECT to occur is 100%; while the probability of a PCT to actually cause a real capture transition is 50% if it is simply assumed that all related *X*-bits in the PCT could take any logic value with equal probability. Based on this observation, *TECTA* for capture C_i (i = 1, 2), denoted by $TECTA_i$, can be calculated as follows:

$$TECTA_i = (\text{\# of ECTs for } C_i) + (0.5 \times (\text{\# of PCTs for } C_i))$$

Generally, the capture with higher *TECTA* is selected as the target capture, since the number of capture transitions for this capture is likely to be greater than that for the other capture, and hence it needs to be reduced first. An example is shown in Fig. 9, which has four 3-bit-tuples. In this case, C_1 is selected since $TECTA_1 > TECTA_2$.

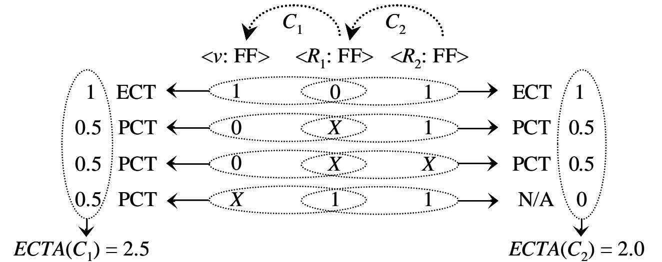


***Fig. 9*. Target capture selection.**

3.5. Target 3-Bit-Tuple Selection

Once a target capture is selected, it is necessary to further select a target 3-bit-tuple that has at least one *X*-bit and that has the highest possibility of successfully reducing capture transitions for the selected target capture.

As shown in the example of DC *X*-filling in Fig. 7, assignment and justification are used to determine logic values for *X*-bits in a test cube *v* to make required logic values appear at the *X*-bits in a 3-bit-tuple so that capture transitions are reduced. ***Assignment*** is to set a logic value to an *X*-bit in <*v*: FF> directly. Since any logic value can be loaded to any scan FF in shift mode for <*v*: FF>, assignment is straightforward and always successful. On the other hand, ***justification*** is to identify proper logic values for *X*-bits in *v* to make required logic values appear at the *X*-bits in <R_1: FF> or <R_2: FF>. Obviously, there is no guarantee that justification is always successful.

As a result, in target 3-bit-tuple selection, we first select a 3-bit-tuple that only needs assignment in *X*-filling. If there are

multiple choices of this kind, a 3-bit-tuple whose corresponding FF has the largest weight is selected each time.

Only when there is no such 3-bit-tuple, we select from 3-bit-tuples that need justification in *X*-filling, based on a heuristic measure. An example is shown in Fig. 10.

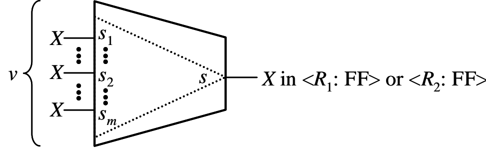


***Fig. 10. X*-Bit justification.**

In Fig. 10, there is one *X*-bit on line *s* on which justification is needed. Suppose that the level of *s* is L_s. Also suppose that *s* can reach *m X*-bit signal lines s_1, s_2, …, s_m corresponding to a test cube *v*, and that the levels of these signal lines are Ls_1, Ls_2, …, Ls_m. Here, levels are assigned from the output side toward the input side, and the highest level is denoted by *L*.

Conceptually, it is evident that if the more *X*-bit signal lines are reachable from *s* and the closer they are to *s*, then the easier it is to justify a logic value on *s*. Based on this observation, the ***justification easiness*** (***JE***) of *s*, denoted by *JE*(*s*), is calculated as follows:

$$JE(s) = \sum_{i}^{m} \frac{(L - |Ls - Ls_i|)}{L}$$

Obviously, the larger the value of *JE*(*s*), the easier the justification of a logic value on *s*.

Therefore, when it is necessary to select a 3-bit-tuple that needs justification, we first select from 3-bit-tuples with one *X*-bit in <R_1: FF> or <R_2: FF>. The *JE* value for the signal line with the *X*-bit is calculated, and the 3-bit-tuple of the largest *JE* value is selected. If there are only 3-bit-tuples that have two *X*-bits in <R_1: FF> and <R_2: FF>, the sum of the *JE* values for the signal lines with the *X*-bits is calculated, and the 3-bit-tuple with the largest sum of *JE* values is selected. In the case where multiple 3-bit-tuples have similar *JE* values, we select the one whose corresponding FF has the largest weight.

3.6. *X*-Filling Operation

After a target capture and a target 3-bit-tuple are selected, assignment and/or justification are conducted to determine logic values for *X*-bits in a test cube *v* in order to make required logic values appear at the *X*-bits in a 3-bit-tuple so that capture transitions are reduced.

Note that justification may fail. For example, for 3-bit-tuple <1, *X*, *X*>, the best choice is to make logic 1 appear at both *X*-bits. This choice is tried first by justification. If it fails, we then try the next-to-best choice of making logic 1 appear at the first *X*-bit and logic 0 at the second *X*-bit. If this justification also fails, we then try to make logic 0 appear at both *X*-bits. If this justification also fails, the last choice is to make logic 0 appear at the first *X*-bit and logic 1 at the second *X*-bit.

3.7. Practical Issues

3.7.1. Handling of *X*-Sources

In practice, a circuit may contain such *X*-sources as analog blocks, memories, un-initialized FFs, multiple clock domains, floating bus, inaccurate simulation models, etc. These *X*-sources, as well as *X*-bits in a test cube, may result in some *X*-bits in the corresponding test response at the inputs of FFs.

Different from *X*-bits existing in a test cube, above-mentioned *X*-sources are uncontrollable in that it is impossible to set an *X*-source to any required logic value. As a result, in the DC *X*-filling procedure, if justifying a logic value at an *X*-bit in a test response ends up needing to set a specific value at an *X*-source as the only choice, the justification is considered unsuccessful.

3.7.2. Application to Unconventional Scan Schemes

The conventional scan scheme uses one external scan input pin and one external scan output pin for each internal scan chain. Recently, some unconventional scan schemes, such as OPMISR, VirtualScan, EDT, SoCBIST, etc., have been proposed for reducing test data volume and test application time. These unconventional scan schemes can be classified into two groups: ***X-independent*** (OPMISR and VirtualScan) and ***X-dependent*** (EDT and SoCBIST), according to whether its fault detection capability depends on the use of *X*-bits in a test cube [2]. Obviously, the DC *X*-filling method readily works with any *X*-independent scan scheme.

As for *X*-dependent scan schemes, an interactive approach is needed. That is, *X*-bits are first utilized to guarantee the minimum fault detection capability. The remaining *X*-bits are used for detecting more faults or reducing capture test power with the DC *X*-filling method, as long as the resulting test cube can be compressed. Test power analysis may also be needed to determine which type of reduction should be targeted with *X*-bits: test data volume or test power dissipation.

4. Application of DC *X*-Filling in ATPG

DC *X*-Filling can be applied into any ATPG flow in a ***dynamic*** or ***static*** manner, depending on how test cubes are generated and processed for *X*-filling.

4.1. Dynamic Application

In each ATPG run, a primary fault is selected and a test cube is generated to detect it. This initial test cube usually contains a large number of *X*-bits. Conventionally, dynamic compaction is conducted by assigning logic values to the *X*-bits (either algorithmically or by random-fill) to detect as many secondary or fortuitous faults as possible, in order to reduce the total number of final test vectors.

In the new dynamic compaction flow as shown in Fig. 11, the *X*-bits are also used for reducing capture switching activity by DC *X*-filling. This is the ***dynamic application*** of DC *X*-filling.

In order to balance the conflicting needs of using *X*-bits for test data reduction and capture switching activity reduction, a concept, called ***X-usage control***, is introduced. As shown in Fig. 11, ***X-Limit*** is a user-specified percentage of original *X*-bits allowed for detecting secondary faults. First, *X*-bits are used for detecting secondary faults. A measure, ***X-Usage***, is updated each time when a secondary fault is detected. When *X-Usage* reaches *X-Limit*, the use of *X*-bits is switched to capture switching activity reduction with DC *X*-filling.

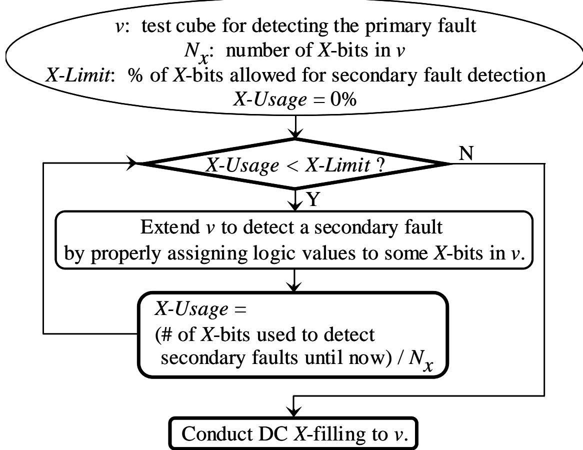


***Fig. 11*. Dynamic application of DC *X*-filling.**

4.2. Static Application

In ***static application***, a set of test cubes with *X*-bits is obtained and DC *X*-filling is conducted for each test cube to reduce capture switching activity, as illustrated in Fig. 12.

There are two approaches to obtaining test cubes for static application. One approach is to generate test cubes in ATPG by leaving *X*-bits alone without conducting random-fill, which is simple but often results in more test vectors. The other approach is to conduct ***X-identification*** for a set of fully-specified test vectors [24] to find bits (*X*-bits) that can actually be either 0 or 1 without causing any fault coverage loss.

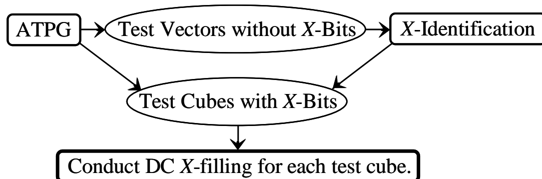
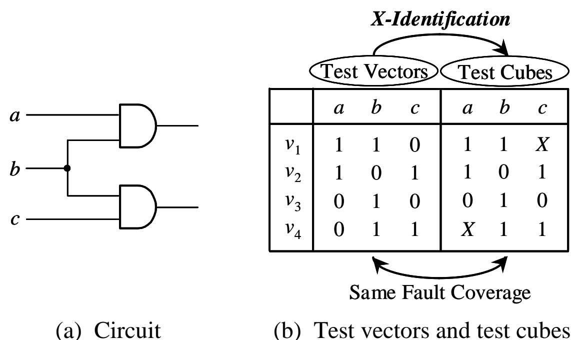


***Fig. 12*. Static application of DC *X*-filling.**

An example is shown in Fig. 13, where two *X*-bits are identified from the original test vectors without losing any fault coverage. Since the set of original test vectors can be generated with aggressive dynamic compaction, the number of test vectors in this approach is usually smaller than simply leaving X-bits unfilled in ATPG.



	a	b	c	a	b	c
v_1	1	1	0	1	1	X
v_2	1	0	1	1	0	1
v_3	0	1	0	0	1	0
v_4	0	1	1	X	1	1

***Fig. 13*. *X*-identification.**

5. Experimental Results

The DC *X*-filling algorithm was implemented in C, and experiments were conducted on eight ISCAS'89 benchmark circuits as shown in Table 3. Both dynamic and static applications were evaluated with an internally developed transition delay ATPG. Comparisons were made with the common practice of random-fill [1] and the state-of-the-art method of preferred-fill [21]. A workstation with a 2.6GHz CPU and 16GB memory was used for experiments.

***Table 3*. Circuit Statistics**

Circuit	# of PIs	# of FFs	# of Faults
s1423	17	74	2240
s5378	35	179	4924
s9238	19	228	10612
s13207	31	669	14740
s15850	14	597	17540
s35932	35	1728	53340
s38417	28	1636	48984
s38584	12	1452	52110

As described in 2.2, capture switching activity largely depends on the number of capture transitions at FFs. In the experiments, we further took the fanout branches of each FF into consideration in order to more accurately reflect parasitic capacitance at the FF. That is, we evaluated capture switching activity by a metric called ***weighted switching activity at FF*** (***FF-WSA***), which is the weighted sum of capture transitions at all FFs. Note that the weight of a FF is the number of its fanout branches + 1.

5.1. Dynamic Application Results

In dynamic application of DC *X*-filling shown in Fig. 11, it is necessary to set *X-Limit*. Generally, the smaller the *X-Limit*, the more test vectors will be generated since fewer secondary or fortuitous faults can be detected by one test vector. However, the smaller the *X-Limit*, the higher effect of DC *X*-filling in capture transition reduction will be obtained, since more *X*-bits are available for this purpose.

Extensive experiments on ISCAS'89 circuits have revealed that the number of test vectors will not grow too much if *X-Limit* is greater than a certain value, which can be as small as 20%. Fig. 14 shows the experimental results on two largest ISCAS'89 circuits. This fact is very useful in achieving a good balanced between test data reduction and capture transition reduction, by setting a properly-low *X-Limit*.

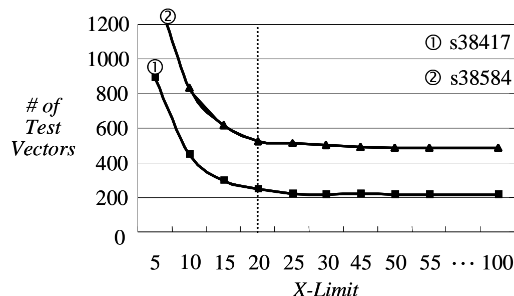


***Fig. 14*. Impact of *X-Limit*.**

Experiments with random-fill, preferred-fill [21], and DC *X*-filling in dynamic compaction were conducted for *X-Limit* = 20%, and the results are shown in Table 4.

In Table 4 (a), “Fault Cov.” shows the transition fault coverage. Under “# of Vec.”, “Ran.” shows the number of test vectors obtained by random-fill, and “Incr (%)” shows test vector count increase rates for [21] and DC. Here, “[21]” and “DC” indicate preferred-fill and DC *X*-filling, respectively.

From Table 4 (a), it can be seen that preferred-fill and DC *X*-filling have similar test size inflation. However, preferred-fill is faster than DC *X*-filling. This is because preferred-fill is a one-pass procedure, while DC *X*-filling is a multi-pass procedure. That is, preferred-fill processes all *X*-bits in a test cube simultaneously, while DC *X*-filling processes one 3-bit-tuple at a time. It is clear that the former has better scalability while the latter has better effectiveness.

In Table 4 (b), under “Reduction (%)”, the reduction rates of the peak WSA and peak FF-WSA values for the first capture (“C_1”) and second capture (“C_2”) are shown. In addition, the reduction rates of the difference between the peak FF-WSA for C_1 and C_2, are shown under “Diff.”. CPU time for both [21] and DC is shown in under “CPU (sec.)”.

***Table 4*. Results of Dynamic Application**

(a) Basic Results

Circuit	Fault Cov. (%)	# of Vec.			CPU (sec.)	
		Ran.	Incr. (%)			
			[21]	DC	[21]	DC
s1423	85.8	132	19.7	13.6	1	1
s5378	84.8	185	34.6	45.9	3	4
s9238	81.3	483	25.3	26.1	41	52
s13207	79.5	324	14.5	11.7	72	93
s15850	70.2	238	11.3	9.2	105	117
s35932	82.5	254	82.7	78.7	159	349
s38417	98.0	384	20.6	21.1	187	346
s38584	83.9	444	8.3	9.9	1025	1399
Ave.	---	---	27.1	27.0	199	295

(b) Reduction Results

Circuit	Reduction (%)								Diff. @ C_1 & C_2 (Peak FF-WSA)	
	Peak WSA				Peak FF-WSA					
	C_1		C_2		C_1		C_2		[21]	DC
	[21]	DC	[21]	DC	[21]	DC	[21]	DC		
s1423	11.4	-1.6	22.8	4.5	15.4	16.0	24.3	17.1	-27.6	10.3
s5378	5.5	30.7	-10.0	33.3	8.2	49.5	-4.9	48.3	37.8	52.2
s9238	15.4	20.1	6.5	13.0	15.3	47.8	30.6	27.1	-36.9	81.0
s13207	13.5	22.0	9.0	19.9	33.2	51.6	17.5	26.4	89.3	24.3
s15850	22.4	24.9	29.5	30.6	37.9	51.0	37.7	41.3	38.8	86.4
s35932	9.0	22.4	-0.8	4.3	31.4	35.8	21.1	32.2	-26.7	95.2
s38417	9.3	30.2	6.0	23.7	17.1	55.6	12.1	53.5	57.5	72.4
s38584	45.5	50.3	47.9	55.0	57.7	67.8	56.0	63.9	75.4	91.2
Ave.	16.5	24.9	13.9	23.0	27.0	46.9	24.3	38.7	26.0	64.1

On average, DC *X*-filling achieved 46.9%, 38.7%, and 64.1% reduction for the peak FF-WSA values at the first and second captures (C_1 and C_2), as well as for the difference in the peak FF-WSA values at C_1 and C_2, respectively, over that of random-fill. It is also clear that the result of DC *X*-filling is better than that of [21].

5.2. Static Application Results

In static application of DC *X*-filling, *X*-identification [24] was first conducted on a fully-specified test set to find *X*-bits without causing any fault coverage loss. As shown in Table 5 (a), “XID (%)” is the percentage of identified *X*-bits, which is 81.2% on average. The resulting test cubes with *X*-bits were then processed with preferred-fill [21] and DC *X*-filling. The results on capture switching activity reduction are shown in Table 5 (b).

***Table 5*. Results of Static Application**

(a) Basic Results

Circuit	Fault Cov. (%)	# of Vec.	XID (%)	CPU (sec.)	
				[21]	DC
s1423	85.8	76	55.5	0	0
s5378	84.8	178	75.1	0	1
s9238	81.3	376	77.3	1	8
s13207	79.5	309	91.8	1	25
s15850	70.2	218	86.4	1	11
s35932	82.5	337	96.7	4	224
s38417	98.0	270	77.8	3	94
s38584	83.9	410	88.9	6	163
Ave.	---	---	81.2	2	66

(b) Reduction Results

Circuit	Reduction (%)								Diff. @ C_1 & C_2 (Peak FF-WSA)	
	Peak WSA				Peak FF-WSA					
	C_1		C_2		C_1		C_2		[21]	DC
	[21]	DC	[21]	DC	[21]	DC	[21]	DC		
s1423	-1.1	4.8	25.2	14.3	13.5	23.6	23.0	17.8	-16.3	41.9
s5378	1.3	6.3	-7.9	9.5	13.8	25.5	0.0	17.0	32.6	37.0
s9238	11.1	23.0	1.8	12.1	-10.0	39.3	2.2	6.7	-54.0	42.5
s13207	2.7	11.0	-5.2	0.0	11.8	26.2	1.1	14.6	84.0	60.5
s15850	14.8	27.0	-1.4	4.3	18.9	42.0	7.3	16.0	60.0	66.5
s35932	10.0	7.9	11.3	9.3	29.4	42.6	27.2	27.1	56.1	-30.6
s38417	6.6	19.7	7.4	15.0	6.0	32.1	16.9	30.8	-234.1	-15.9
s38584	5.0	14.8	6.5	10.0	8.1	18.0	9.0	14.4	-7.5	76.6
Ave.	6.3	14.3	4.7	9.3	11.4	31.2	10.8	18.0	-9.9	34.8

On average, DC *X*-filling achieved 31.2%, 18.0%, and 34.8% reduction for the peak FF-WSA values at the first and second captures (C_1 and C_2), as well as for the difference in the peak FF-WSA values at the first and second captures, respectively, over that of original test sets. It is also clear that the result of DC *X*-filling is better than that of [21].

5.3. Discussion

Since DC *X*-filling uses accurate justification and implication, while preferred-fill [21] uses approximate probability calculation, to set logic values for *X*-bits, DC *X*-filling generally achieves better results. In addition, the weight of an FF is also taken into consideration in the DC *X*-filling algorithm. However, preferred-fill is fast since it is a one-pass procedure. This indicates that a trade-off between effectiveness and processing time may be achieved by combining the essences of DC *X*-filling and preferred-fill.

6. Conclusions

This paper addressed the problem of test-induced yield loss, by reducing capture switching activity in launch-off-capture-based at-speed scan testing that uses two captures. A novel *X*-filling method, called ***double-capture* (*DC*) *X-filling***, was proposed to achieve balanced reduction of capture switching activity at both captures, without any need for circuit/clock modification. Its effectiveness was confirmed by experiments.

Future work will address the following issues:

(1) The peak capture switching activity of some circuits may not be located at FFs. Thus, it is necessary to conduct circuit-specific analysis to reduce the actual peak.

(2) The actual effect of capture switching activity reduction needs to be evaluated by IR-drop / delay analysis based on information on layout, power grids, etc. Such evaluation is also needed for justifying the use of FF-WSA metric.

(3) The scalability of the capture switching activity reduction technique proposed in this paper needs to be improved by reducing the number of passes in *X*-filling.

References

[1] M. L. Bushnell and V. D. Agrawal, *Essentials of Electronic Testing for Digital, Memory & Mixed-Signal VLSI Circuits*. New York: Kluwer Academic Publishers, 2000.

[2] L.-T. Wang, C.-W. Wu, and X. Wen, (Editors), *VLSI Test Principles and Architectures: Design for Testability*, Elsevier, 2006.

[3] X. Lin, R. Press, J. Rajski, P. Reuter, T. Rinderknecht, B. Swanson, and N. Tamarapalli, "High-Frequency, At-Speed Scan Testing," *IEEE Design & Test of Computers*, pp. 17-25, September-October, 2003.

[4] S. Mitra, E. Volkerink, E. McCluskey, and S. Eichenberger, "Delay Defect Screening Using Process Monitor Structures," *Proc. VLSI Test Symp.*, pp. 43–52, 2004.

[5] J. Saxena, K. M. Butler, V. B. Jayaram, and S. Kundu, "A Case Study of IR-Drop in Structured At-Speed Testing," *Proc. Intl. Test Conf.*, pp. 1098-1104, 2003.

[6] P. Girard, "Survey of Low-Power Testing of VLSI Circuits," *IEEE Design & Test of Computers*, Vol. 19, No. 3, pp. 82-92, May/June 2002.

[7] N. Nicolici and B. Al-Hashimi, *Power-Constrained Testing of VLSI Circuits*, Kluwer Academic Publishers, 2003.

[8] J. Wang, D. M. H. Walker, A. Majhi, B. Kruseman, G. Gronthoud, L. E. Villagra, P. Wiel, and S. Eichenberger, "Power Supply Noise in Delay Testing," *Proc. Int'l Test Conf.*, Paper 17.3, 2006.

[9] M. Nourani, M. Tehranipoor, and N. Ahmed, "Pattern Generation and Estimation for Power Supply Noise Analysis," *Proc. VLSI Test Symp.*, pp. 439-444, 2005.

[10] A. Kokrady and C. P. Ravikumar, "Fast, Layout-Aware Validation of Test Vectors for Nanometer-Related Timing Failures," *Proc. Int'l Conf. on VLSI Design,* pp. 597-602, 2004.

[11] T. Yoshida and M. Watari, "MD-Scan Method for Low Power Scan Testing," *Proc. Intl. Test Conf.*, pp. 480-487, 2003.

[12] F. Corno, P. Prinetto, M. Redaudengo, and M. Reorda, "A Test Pattern Generation Methodology for Low Power Consumption," *Proc. VLSI Test Symp.*, pp. 35-40, 2000.

[13] R. Sankaralingam, R. Oruganti, and N. Touba, "Static Compaction Techniques to Control Scan Vector Power Dissipation," *Proc. VLSI Test Symp.*, pp. 35-40, 2000.

[14] A. Chandra and K. Chakrabarty, "Reduction of SoC Test Data Volume, Scan Power and Testing Time Using Alternating Run-Length Codes," *Proc. Design Automation Conf.*, pp. 673-678, 2002.

[15] K. Lee, T. Huang, and J. Chen, "Peak-Power Reduction for Multiple-Scan Circuits during Test Application," *Proc. Asian Test Symp.*, pp. 435-440, 2000.

[16] S. Wang and W. Wei, "A Technique to Reduce Peak Current and Average Power Dissipation in Scan Designs by Limited Capture," *Proc. Asian S. Pacific Design Automation Conf.*, pp. 810-816, 2007.

[17] F. Corno, P. Prinetto, M. Redaudengo, and M. Reorda, "A Test Pattern Generation Methodology for Low Power Consumption," *Proc. VLSI Test Symp.*, pp. 35-40, 1998.

[18] X. Wen, S. Kajihara, K. Miyase, T. Suzuki, K. K. Saluja, L.-T. Wang, K. S. Abdel-Hafez, and K. Kinoshita, "A New ATPG Method for Efficient Capture Power Reduction During Scan Testing," *Proc. VLSI Test Symp.*, pp. 58-63, 2006.

[19] X. Wen, Y. Yamashita, S. Morishima, S. Kajiihara, L.-T. Wang, K. K. Saluja, and K. Kinoshita, "On Low-Capture-Power Test Generation for Scan Testing," *Proc. VLSI Test Symp.*, pp. 265-270, 2005.

[20] W. Li, S. M. Reddy, I. Pomeranz, "On Reducing Peak Current and Power during Test," *Proc. ISVLSI*, pp. 156-161, 2005.

[21] S. Remersaro, X. Lin, Z. Zhang, S. M. Reddy, I. Pomeranz, and J. Rajski, "Preferred Fill: A Scalable Method to Reduce Capture Power for Scan Based Designs," *Proc. Int'l Test Conf.*, Paper 32.2, 2006.

[22] X. Wen, K. Miyase, T. Suzuki, Y. Yamato, S. Kajihara, L.-T. Wang, and K. K. Saluja, "A Highly-Guided *X*-Filling Method for Effective Low-Capture-Power Scan Test Generation," *Proc. Int'l Conf.* on Computer Design, pp. 251-258, 2006.

[23] K. M. Butler, J. Saxena, T. Fryars, G. Hetherington, A. Jain, and J. Levis, "Minimizing Power Consumption in Scan Testing: Pattern Generation and DFT Techniques," *Proc. Int'l Test Conf.*, pp. 355-364, 2004.

[24] K. Miyase and S. Kajihara, "XID: Don't Care Identification of Test Patterns for Combinational Circuits," *IEEE Trans. Computer- Aided Design*, Vol. 23, No. 2, pp. 321-326, 2004.

[25] A. H. El-Maleh and K. Al-Utaibi, "An Efficient Test Relaxation Technique for Synchronous Sequential Circuits," *IEEE Trans. on Computer-Aided Design*, Vol. 23, No. 6, pp. 933-940, 2004.

[26] X. Wen, Y. Yamashita, S. Morishima, S. Kajihara, L.-T. Wang, K. K. Saluja, and K. Kinoshita, "Low-Capture-Power Test Generation for Scan-Based At-Speed Testing," *Proc. Int'l Test Conf.*, Paper 39-2, 2005.