Random Adjacent Sequences: An Efficient Solution for Logic BIST

René M. G. David, Patrick Girard, Christian Landrault, Serge Pravossoudovitch, Arnaud Virazel

RSIC Generation: A Solution for Logic BIST

R. David[1], P. Girard[2], C. Landrault[2], S. Pravossoudovitch[2], A. Virazel[2]

[1] *Laboratoire d'Automatique de Grenoble, BP 46, 38402 St-Martin-d'Hères, France*
Rene.David@inpg.fr
[2] *Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier*
161, rue Ada, 34392 Montpellier Cedex 05, France
<name>@lirmm.fr

Abstract

High defect coverage requires good coverage of different fault types. In this paper, we present a comprehensive test vector generation technique for BIST, called Random Single Input Change (RSIC) generation, that can be used to generate tests for many arbitrary misbehaviors that can occur in digital systems, thus providing a single on-chip test generation solution. By proving the effectiveness of universal test sequences produced by such a generation technique in detecting stuck-at, path delay and bridging faults, we demonstrate that using RSIC generation is one of the best and most practical way to reach a high level of defect coverage during BIST of digital circuits.

1. Introduction

In the near future, the recent advances in deep-submicron IC process technology and core-based IC design technology will lead to a widespread use of logic BIST in industry. This is confirmed by the ITRS (International Technology Roadmap for Semiconductors) statement that by 2014 it may cost more to test a transistor than to manufacture it unless techniques like logic BIST are employed [1]. Logic BIST, which test logic circuits through the use of built-in pattern sources and response evaluators, offers a number of advantages compared with external testing, which is becoming more and more difficult and costly [2].

A first problem that occurs with external testers is that they are several times slower than the circuits they have to test. Purchasing high speed testers that meet the performances of new designs requires a huge investment. Moreover, even with those high speed testers, it is not always possible to have a timing accuracy comparable to the IC internal speed [2]. In this context, BIST represents an attractive test solution since it allows at-speed testing (*i.e*. test at the intended operating speed of the circuit), thus solving timing accuracy and test time related problems encountered with traditional external testers. In addition, BIST drastically reduces the amount of test data exchanged with the tester, thus reducing the need for complex external testing equipment. BIST can hence be run on a very low cost tester. Finally, BIST may relieve the problem of tester capacity (very often, external testers do not have enough memory to store the entire test set to cover stuck-at and path delay faults, and tester reloads become necessary [3]) and the problem of low accessibility of internal nodes of the design, which increases the test complexity [4].

Another strong demand for BIST solutions comes from providers of pre-designed and pre-verified modules (cores). An increasing part of microelectronic systems is now implemented using embedded cores, and BIST has been shown to be the appropriate method for testing complex core-based systems [5]. Core-based System-on-a-Chip (SoC) testing introduces new difficulties into the test process caused by the increase complexity of the chip, the reduced accessibility of the cores and the higher heterogeneity of the modules. Equipping the cores with BIST features solves the accessibility problem and helps to preserve the intellectual property of the design as less test information about the core has to be given to the user.

The growing need for performance-related testing (called delay testing) also demands BIST solutions. Deep-submicron technologies introduce new performance-related defect types, and the increasing clock frequencies in high-speed designs impose aggressive timing margins. While the internal clock frequencies have risen by 30% per year, the accuracy of external test equipment has improved at a rate of only 12% per year [1]. Hence, it is becoming increasingly difficult to do performance-related testing using external test equipment. BIST may solve this problem since it allows to do accurate delay testing and precision measurement on-chip.

Test pattern generation for BIST can be broadly classified into deterministic, mixed-mode, pseudo-random (flat or weighted) and exhaustive generation. Deterministic BIST consists in storing pre-computed test patterns on-chip by using a special-purpose hardware [6,7,8]. Deterministic BIST has been shown to be efficient in terms of fault detection but the price for obtaining complete fault coverage usually is a relatively large silicon area overhead. *Moreover, deterministic BIST is always related to a specific fault model*. Mixed-mode BIST consists in using a limited number of pseudo-random patterns to eliminate easy to detect faults and deterministic patterns to cover the remaining random patterns resistant faults [9]. The deterministic patterns

are either stored on-chip in a compressed format and expanded during BIST [10-13] or directly embedded into an LFSR sequence by "bit-fixing" or "bit-flipping" techniques [14-17]. Mixed-mode BIST is less hardware consuming compared with deterministic BIST, but is also less efficient in terms of test quality (fault coverage vs test length) and is often extremely tailored to a specific circuit. *Moreover, mixed-mode BIST is related to a specific fault model*. Pseudo-random BIST consists in applying test patterns that exhibit randomness but which are generated using special-purpose hardware (LFSR or Cellular Automata) and are thus repeatable [18-25]. Pseudo-random BIST is the least hardware consuming solution but requires test lengths which may be long in some cases. *Pseudo-random BIST is not related to a specific fault model.* Exhaustive (or pseudo-exhaustive) BIST consists in applying all (or nearly all) the logic combinations at the circuit inputs [26-28]. *Pseudo-exhaustive BIST is not related to a specific fault model*, but cannot be used in practice due to high testing time and circuit segmentation requirement.

The goal in this paper is to propose a new test generation technique for BIST that can be used to generate tests for many arbitrary misbehaviors that can occur in digital systems, thus providing a single on-chip test generation solution. The reason for using such kind of "universal" test sequences can be explained as follows. The classical single-stuck fault (SSF) model continues to be the most commonly used fault model in digital systems testing. However, defects in new nanometer CMOS technologies do not always behave like stuck-at faults [29]. Therefore, test generation based on the SSF model alone is no longer sufficient for obtaining high defect coverage [30,31]. On the other side, the use of multiple test generation techniques that each targets a specific fault type (such as transition, bridging or stuck-open fault type) would be costly and unpractical from a BIST point of view. For this reason, we propose an alternative solution consisting in the use of a single on-chip test pattern generator providing universal test sequences that target both conventional (stuck-at) and non-conventional (delay, bridging, stuck-open) fault types.

The question now is: what kind of BIST test pattern generation do we need to use for generating such "universal" test sequences? According to the above presentation of BIST test generation approaches, the only practical solution is to use pseudo-random BIST as this approach is not related to a specific fault model, *i.e.* test patterns are determined irrespective of the targeted fault type. However, the only problem that remains with pseudo-random BIST is the test length, which is acceptably long to test for SSFs, but which is prohibitively long to test for delay faults [32]. A solution to solve this problem is to use Random Single Input Change (RSIC) test sequences, that have been shown to be efficient to test path delay faults with a reasonable test length [33]. RSIC test sequences are single input change (also called adjacency) test sequences that have to be generated in such a way that they have practically the properties of pure random sequences. This can be performed through the use of an LFSR with improved random properties for example. Moreover, the selection of the changing bit (between two consecutive patterns) has also to be performed randomly. This can be done through the use of a special hardware structure as the one described in Section 2.2.

The goal of this paper is to propose a new test pattern generation technique for BIST, called RSIC generation, and to analyze the effectiveness of "universal" test sequences produced by such a generation technique in detecting stuck-at, path delay and bridging faults. For this purpose, we proceed in two steps. First, we remind the results already published and demonstrating the effectiveness of RSIC test sequences in terms of delay fault coverage. Next, we present new results in terms of SSF and bridging fault coverage achieved by universal test sequences produced from a RSIC generation. We therefore demonstrate that using RSIC generation is one of the best and most practical way to reach a high level of defect coverage during BIST of digital circuits. This is obtained at the expense of a test length which is reasonably increased compared with the test length for SSF testing alone. Note that in addition to stuck-at, path delay and bridging faults, stuck-open faults (which also require two-pattern tests) can also be detected by "universal" test sequences.

The remainder of this paper is organized as follows. Section 2 gives preliminary definitions on the considered fault models and the test sequences used. Section 3 highlights the effectiveness of RSIC generation, first for delay fault detection and next for SSF and bridging fault detection. Concluding remarks are given in Section 4.

2. Basics and background

2.1 Basics on fault models

Fault models describe the logical behavior of a faulty system [34]. In this work, several fault models have been considered to exhibit the effectiveness of test sequences produced from a RSIC generation. The classical and most widely-used SSF model is obviously considered. Two other fault models are also used: the bridging fault (BF) model and the path delay fault (PDF) model. Basic definitions and notations on these fault models are now given.

A bridging fault can be viewed as an unintentional short between two lines. This short can be a non-resistive short such that the two lines are always brought into equilibrium at the same potential or a resistive short such that the potential between the shorted lines is different. Therefore, after the popular wired-AND and wired-OR models used to model the effects of bridging faults in bipolar logic [35], several models for the resistive and non-resistive fault types of bridging faults were proposed. However, these models suffer from a number of limitations. The primary drawback of the non-resistive fault model is that it is not guaranteed to detect resistive bridging faults. While resistive BF models are more realistic, they fall short of modeling all bridging type anomalies since the number of possible BF anomalies is intractable for most circuits [36].

Although the coverage of a bridging fault by both wire-AND and wire-OR behavior does not always guarantee detection, these models have the advantage to be easy to consider during BF simulation. Moreover, it has been proven recently that a high gate level SSF coverage implies high BF coverage [37]. Consequently, the rest of this paper is based on the use of such models to represent bridging defects that occur in digital circuits. The BF site extraction problem has not been dealt with since the lists of realistic bridging faults considered during the evaluation of RSIC test sequences were provided by the Lisboa Technical University (INESC/IST).

Some defects in a manufactured circuit do not affect the logic function of the circuit, rather they change the delays of some gates, thus altering the speed at which the circuit can operate. Such defects are referred to as delay defects and delay fault testing is the term used for the methodology of detecting such defects by checking whether a manufactured circuit meets its timing characteristics. The fault model most widely studied in this context is the path delay fault model [38]. The main advantage of using the path delay fault model is that it models the real distributed delay defects very well compared with other existing delay fault models, namely the gate delay fault model and the transition delay fault model. However, an important feature of the path delay fault model is that the single fault assumption is not realistic since a single defect usually affects a large number of paths. For this reason, a robust test is preferred to detect a path delay fault. A test for a path is robust if it can detect a delay fault on the path irrespective of other delays and delay faults in the circuit, otherwise it is non-robust [38].

Detection of path delay faults requires two-pattern tests. An initialization vector is applied and the circuit is allowed to stabilize. Then, the test vector is applied and the circuit outputs are sampled at clock speed. The response is then compared to that of the fault-free circuit to determine the presence or the absence of a delay fault. Hence, correct operation of a circuit at the intended speed can only be guaranteed if there is no path delay that exceeds the value determined by the clock period [39].

2.2 Theoretical basis of RSIC generation

In general, two-pattern tests may differ in multiple bit positions. In this case, they are called multiple input change (MIC) pattern pairs. Test pairs that differ only in one bit position are called single input change (SIC) pattern pairs. Let us now define what a RSIC sequence should be from a theoretical point of view (we assume equal likelihood of all vectors). Let

$$S = V(1)V(2)...V(l)...V(L), \quad (1)$$

be a test sequence composed of L successive n-bit vectors $V(l)$. Each vector takes a value from the set

$$V = \{V_0, V_1, ..., V_{2^n-1}\}, \quad (2)$$

where V_j corresponds to the n-bit vector $(x_1,x_2, \ldots,x_n)$ associated with the decimal value j. For example, for n = 5, V_9 = 01001, i.e., $x_1 = x_3 = x_4 = 0$ and $x_2 = x_5 = 1$. In a Random MIC (called RMIC) sequence, the probability $\Pr[V(l) = V_j] = 1 / 2^n$ for any l and any j, and the probability $\Pr[V(l) = V_j]$ is independent of the values $V(i)$, where i = 1, 2, …, l–1. In a RSIC sequence, this probability is:

$$\Pr[V(l) = V_j \mid V(l-1) = V_k] = \frac{1}{n}, \text{iff } |j - k| = 2^a, \quad (3)$$

where a is a non-negative integer. In other words, for any $l > 0$, $V(l)$ differs from $V(l–1)$ by exactly one bit randomly drawn, and this bit must be independent of the bits previously drawn.

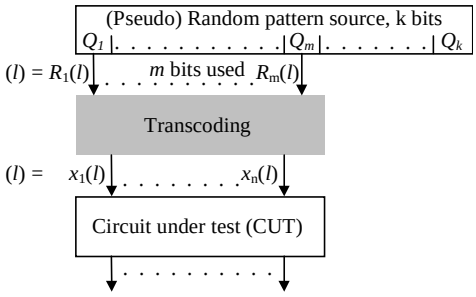

Figure 1: Principle of generation of a RSIC test sequence

Figure 1 represents the basic principle of an RSIC generation. This principle is taken from [40]. Basically, a k-bit random (in fact pseudo-random) source is used; this source may be a random number obtained from a maximal length LFSR. The value of the vector $Q_1Q_2....Q_k$ changes at each clock cycle of the test session. At each time l, a subset of m bits is used ($m \le k$). The vector $R(l) = R_1(l)R_2(l)....R_m(l)$ is transformed into a n-bit vector $V(l) = x_1(l)x_2(l)....x_n(l)$ which is applied to the circuit under test. In order to agree with the definition of a RSIC sequence given above, the following points have to be addressed:

1) $R(l)$ is independent of $R(l–1)$.
2) The transcoding between $R(l)$ and $V(l)$ satisfies Equ.(3), at least approximately.
3) The period of the sequence $S = V(1)V(2)\ V(l)\ V(L)$ is at least equal to the test length L.

Solutions to points 1 and 2 are given in [40]. The third point is addressed in [41] where it is demonstrated that the period of the RSIC sequence generated from an appropriate structure based on the above principle is $2 \times (2^k–1)$. For this point to be verified, the length k of the pseudo-random source (i.e., an LFSR) must be chosen such that $2 \times (2^k – 1) > L$, i.e.,

$$k > \log_2\left(\frac{L}{2}+1\right) \approx \log_2 \frac{L}{2} \quad (4)$$

The advantage of using RSIC generation is that the probability of delay test invalidation due to hazards or multiple delay faults is greatly reduced as a single transition is applied at the primary inputs of the CUT at each clock cycle of the test session. Moreover, RSIC test pairs are sufficient to detect all robustly detectable path delay faults [38]. Finally, the universe of pattern pairs considered for SIC (or RSIC) generation ($O(n \cdot 2^n)$) is significantly smaller than that for MIC (or RMIC) generation ($O(2^{2n})$). Hence, RSIC fault coverage can be higher than RMIC fault coverage for the same test length [23].

The structures producing MIC or SIC test sequences with random properties similar to those discussed above were taken from [40]. More specifically, RMIC test sequences have been obtained from a modified LFSR in which more than one shifting are used between two consecutive vectors. An example of such a modified LFSR is shown in Figure 2. Assuming a test sequence of 2-bit vectors applied to the CUT, the classical LFSR (Figure 2.a) is replaced by a modified LFSR with a number of shiftings • = 2 that allows two consecutive vectors to be independent (Figure 2.b).

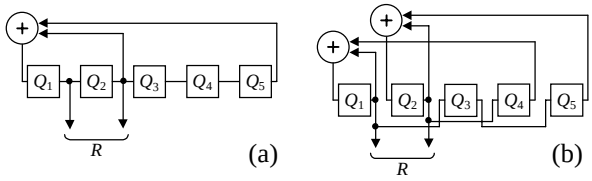

Figure 2: Generation of a RMIC sequence by an LFSR with more than one shifting

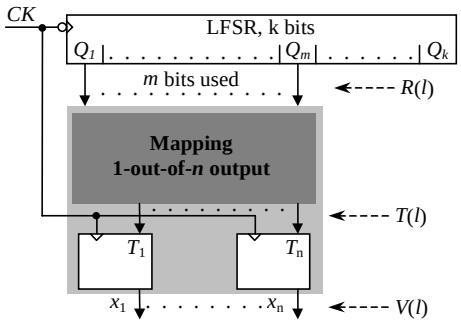

Figure 3: Hardware generation of a RSIC test sequence

Similarly, RSIC test sequences have been obtained from the hardware structure proposed in [40] and reproduced in Figure 3. The vector $R(l)$ coming from the modified LFSR is mapped into a 1-out-of-n vector $T(l)$. Every component of $T(l)$ is applied to the input of a T flip-flop. Hence, given a random vector $R(l)$, it implies a random trigger input $T_i(l) = 1$ while other trigger inputs $T_j(l) = 0$ for $j \neq i$. Hence, $V(l)$ is similar to $V(l-1)$ apart the value of x_i. This hardware RSIC generator has a number of theoretical properties:

1) The LFSR is of maximal-length (period 2^k-1), and $2 \times (2^k-1) > L$,
2) $m \geq \lceil \log_2 n \rceil$,
3) The number σ of shiftings is such that: $1 \leq \sigma < 2^k - 1$, and σ does not share a common factor with 2^k-1,
4) Let b_j be the number of m-bit vectors associated with T_j; $\max_i b_j - \min_i b_j \leq 1$.

More details and discussion on the RSIC generator and the corresponding properties can be found in [40] and in [41]. In particular, it is demonstrated in [41] that random testing from the above hardware structure provides the same results than those obtained from a software generation, which is easy to perform thanks to the random function Rand() of the C language. This confirms the fact that pure random test sequences can be materially generated.

3. Effectiveness of RSIC generation

The aim of this section is to demonstrate the effectiveness of RSIC generation, first for delay fault detection and next for SSF and bridging fault detection. For this purpose, we proceed in two steps. First, we remind the results presented in [42] on the comparison between random and pseudo-random generation of SIC or MIC sequences, and in [33] on the comparison between RSIC and RMIC generation. These results were obtained based on the path delay fault model. Next, we present new results on the detection capability of RSIC generation for SSF and bridging fault coverage. A synthesis is drawn at the end of this section.

3.1 Comparison between random and pseudo-random generation

All the generated sequences are pseudo-random by construction (*i.e.* repeatable). In the sequel, comparison is made between two kinds of generation. The first one is the usual generation : at time l, the vector made of n bits in the LFSR is applied. After a shifting in the LFSR, a new n-bit vector is applied at time l+1. For short, this generation will be called *usual pseudo-random* (in case of SIC sequence, there is an additional transformation as describe in Section 3.3). The second kind of generation (software or hardware, as it will be specified) is such that the sequence has properties very close to a pure random sequence (RSIC or RMIC sequence). For short, such a sequence will be called *random* in the sequel.

In order to validate our statement that SIC test sequences must be random (rather than usual pseudo-random) to be efficient in detecting conventional and non-conventional faults, we first compare random and usual pseudo-random generation in [42]. The results obtained demonstrate that using random test pairs (produced from a software generation) to test path delay faults in a given circuit produces higher delay fault coverage than that obtained with usual pseudo-random test pairs. The comparison has been further extended, showing that the same conclusion can be derived when SSF or bridging fault coverage is targeted rather than delay fault coverage [42].

Circuit	L	Robust Eff		Non-Robust Eff	
		RSIC	SIC	RSIC	SIC
s510	136600	91.11 %	75.95 %	100 %	100 %
s1238	499200	89.11 %	74.94 %	97.16 %	93.55 %
s1494	351800	94.51 %	81.47 %	100 %	99.54 %
s3330	158620	62.28 %	14.14 %	77.63 %	17.01 %
s5378	297300	64.77 %	46.18 %	67.23 %	56.95 %

Table 1: Comparison between RSIC and Pseudo-random SIC - delay fault efficiency

A sample of the results given in [42] is reported in Table 1 and Table 2. Table 1 lists the robust and non-robust delay fault coverage achieved by RSIC test sequences and usual pseudo-random SIC test sequences respectively. The results are expressed in terms of fault efficiency. Similarly, Table 2 lists the robust and non-

robust delay fault efficiency achieved by RMIC test sequences and usual pseudo-random MIC test sequences respectively.

Circuit	L	Robust Eff		Non-Robust Eff	
		RSIC	SIC	RSIC	SIC
s510	136600	31.78 %	23.76 %	100 %	71.14 %
s1238	499200	29.35 %	18.42 %	99.84 %	70.32 %
s1494	351800	30.83 %	23.24 %	99.78 %	72.62 %
s3330	158620	23.47 %	23.18 %	90.71 %	90.62 %
s5378	297300	24.51 %	17.87 %	97.09 %	65.09 %

Table 2: Comparison between RMIC and Pseudo-random MIC - delay fault efficiency

As can be seen on these results, the delay fault efficiency obtained with random (RSIC or RMIC) test sequences is always better than that obtained with usual pseudo-random (SIC or MIC) test sequences. Hence, these results clearly illustrate the fact that random testing is more efficient than usual pseudo-random testing for delay fault detection.

Remark : In order to evaluate the delay fault coverage of the experimented test sequences, we used an industrial test evaluation package, TestGen of Synopsys [43], to perform test validations. An important comment on the validation results given in this paper is that the ISCAS'85 circuits family has not been used in our experiments. This is because the number of path faults in these circuits is too huge, and TestGen is unable to generate the corresponding delay fault dictionaries for fault simulation. Hence, only combinational part of the ISCAS'89 benchmark circuits have been experimented.

3.2 Comparison between RSIC and RMIC generation

According to the above results, it can be stated that random testing is more efficient than usual pseudo-random testing. So, the next step has been to concentrate on this kind of testing and compare RSIC and RMIC generation for delay fault detection.

Circuit	L	RSIC	RMIC
s510	136600	91.11 %	31.78 %
s1238	499200	89.11 %	29.35 %
s1494	351800	94.51 %	30.83 %
s3330	158620	62.28 %	23.47 %
s5378	297300	64.77 %	24.51 %

Table 3: Comparison between robust fault efficiency of RSIC and RMIC testing

A first comparison between SIC and MIC generation for delay fault detection has already been proposed in [23], where it is shown that SIC test sequences are more efficient than MIC test sequences when a high robust delay fault coverage is targeted. However, the study considers usual pseudo-random vectors produced from an LFSR and concentrates the analysis on the coverage of faults having at least one robust test. For these reasons, the efficiency of RSIC test sequences for delay fault testing was further analyzed in [33]. The performance measurement took into account both robust tests and non-robust tests, and a random generation was assumed. The main conclusion drawn from this study is that, even with a lower non-robust delay fault coverage, a RSIC test sequence may often give rise to a better test quality than that obtained with RMIC delay test sequences.

To illustrate our statement, a sample of the results given in [33] is reported in Table 3. As can be seen, these results only concern the robust fault efficiency. For results concerning non-robust fault efficiency, the reader can refer to the discussion developed in [33].

3.3 New results on the effectiveness of RSIC generation

In this subsection, we present new results on the effectiveness of RSIC generation for SSF and bridging fault coverage. We proceed in the same way than that used in our previous evaluations, *i.e.* we compare the efficiency of RSIC test sequences to that of RMIC test sequences (for SSF and bridging fault detection). Results in terms of SSF coverage are given in Table 4. Results in terms of bridging fault coverage are given in Table 5. Bridging faults simulated to evaluate the efficiency of RSIC test sequences are realistic bridging faults provided by the Lisboa Technical University (INESC/IST - Portugal). For each fault, we considered the following behaviors: WAND (Wire-AND), WOR (Wire-OR), WAND&WOR (the fault is tested if the two behaviors are tested), WAND||WOR (the fault is tested if at least one of the two behaviors is tested). As the WAND&WOR and the WAND||WOR behaviors are more representative, only results on these models are reported in Table 5. Note that results in Table 4 are expressed in terms of fault efficiency (*i.e.*, coverage of detectable faults) while results in Table 5 are expressed in terms of fault coverage (we have no information on whether some bridging faults are redundant or not).

By looking at the results in Table 4, it can be seen that the effectiveness of RSIC generation compared to RMIC generation mainly depends on the test length L of the SSF sequence. For a test length L =10 000, the fault efficiency of RMIC test sequences is slightly higher than that of RSIC test sequences. This is also true for test lengths L < 10 000 (on the considered circuits). But when the test length increases, the efficiency of RSIC test sequences becomes comparable to that of RMIC test sequences. For a test length equal to the test length used for delay fault testing (*cf.* sections 3.1 and 3.2), the fault efficiency is nearly the same for both type of test sequences. This is the most important result we can derive from these experiments: for a sufficiently long RSIC test sequence, the SSF efficiency is comparable to that of a RMIC test sequence.

Actually, the same conclusion can be derived when bridging fault detection is considered. This is illustrated in Figure 4, which compares the evolution of the bridging fault coverage of RSIC and RMIC test sequences for circuit s1494. The RSIC test sequence is less efficient for short test lengths, but becomes as efficient as the RMIC test sequence for test lengths L ≥

Circuit	#faults	L	RSIC	RMIC	L	RSIC	RMIC
s386	360	10000	100 %	100 %	73600	100 %	100 %
s420	424	10000	77.12 %	83.25 %	132200	96.10 %	97.88 %
s510	538	10000	100 %	100 %	136600	100 %	100 %
s1238	1 332	10000	90.99 %	97.94 %	499200	99.84 %	100 %
s1494	1 489	10000	99.66 %	99.94 %	351800	100 %	100 %

Table 4: Comparison between RSIC and RMIC generation for SSF efficiency

Circuit	#faults	L	RSIC		RMIC	
			WAND&WOR	WAND||WOR	WAND&WOR	WAND||WOR
s386	390	73600	99.23 %	100 %	99.23 %	100 %
s420	521	132200	81.23 %	97.59 %	81.96 %	98.89 %
s510	1 073	136600	100 %	100 %	100 %	100 %
s1238	2 924	499200	99.9 %	100 %	99.9 %	100 %
s1494	3 851	351800	99.95 %	100 %	99.95 %	100 %

Table 5: Comparison between RSIC and RMIC generation for bridging fault coverage

Circuit	L	SSF Eff	Bridging FC	Delay Robust Eff	Delay Non-Robust Eff
s386	73600	100 %	99.23 %	98.79 %	100 %
s420	132200	94.10 %	81.23 %	66.86 %	68.62 %
s510	136600	100 %	100 %	91.11 %	100 %
s1238	499200	99.84 %	99.9 %	89.11 %	97.16 %
s1494	16383	100 %	99.95 %	94.51 %	100 %

Table 6: Synthesis of the results demonstrating the effectiveness of RSIC generation

1000. For a test length equal to the test length used for delay fault testing (L = 351800), the bridging fault coverage is equal to that of a RMIC test sequence. This conclusion is also true for the other experimented circuits as illustrated in Table 5.

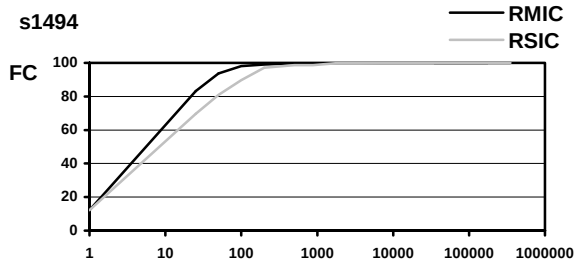

Figure 4: Comparison between bridging fault coverage of RSIC and RMIC test sequences

The fact that RSIC generation requires sufficiently long test sequences to reach the same level of test quality as that of RMIC generation (for SSF and Bridging fault coverage) confirms the basic assumption according to which producing efficient universal test sequences from RSIC generation (for SSF and Bridging fault coverage) can only be done at the expense of a longer test time. But this is not a problem! According to the results reported previously on delay fault testing by RSIC test sequences, we know that quite long test lengths are required. So, the only way to cover both SSF, delay and bridging faults with universal RSIC test sequences is to use test lengths which are at least as long as test lengths for delay fault detection. In this context, RSIC generation is as efficient as RMIC generation for SSF and bridging fault coverage.

To summarize, a synthesis of the results discussed in this section is proposed in Table 6 (note that results in terms of bridging fault coverage are not provided for circuits s3330 and s5378 because we do not have any list of realistic bridging faults for these circuits). For a reasonably long test length, one can verify the efficiency of RSIC test sequences for both conventional (stuck-at) and non-conventional (delay, bridging) fault detection. Note that this effectiveness can be further improved (especially for delay testing) by increasing the length of the "universal" test sequence.

4. Conclusion and future work

In this paper, we have presented a comprehensive test generation technique for BIST, called RSIC generation, that can be used to generate tests for many arbitrary misbehaviors that can occur in digital systems. We have demonstrated the effectiveness of universal test sequences produced by such a generation technique in detecting stuck-at, path delay and bridging faults.

Although the effectiveness of the proposed generation technique has been clearly shown, it is possible to continue this study by looking at a tradeoff between the proposed RSIC generation and the conventional MIC generation. In other words, it would be interesting to analyze the efficiency of test sequences in which two or three bits rather than one bit change between two consecutive vectors. This solution would

probably reduce the test length while achieving the same fault and defect coverage. The problem in this case would be to find an appropriate hardware generating structure. This problem will be consider in our future work.

Acknowledgement

The authors would like to thank Pr. P. Teixeira and his team from the Technical University of Lisboa (INESC/IST) who provided us with the lists of realistic bridging faults experienced.

References

[1] Semiconductor Industry Association (SIA), "International Technology Roadmap for Semiconductors (ITRS) ", 1999 Edition.
[2] Y. Zorian, "Testing the Monster Chip", IEEE Spectrum, Vol. 36, N° 7, pp. 54-60, 1999.
[3] G. Hetherington, T. Fryars, N. Tamarapalli, M. Kassad, A. Hassan and J. Rajski, "Logic BIST for Large Industrial Designs: Real Issues and Case Studies", IEEE Int. Test Conf., pp. 358-367, 1999.
[4] J. Rajski and J. Tyszer, "Arithmetic Built-In Self-Test for Embedded Systems", Prentice Hall PTR, 1998.
[5] H.J. Wunderlich, "BIST for Systems-on-a-Chip", INTEGRATION, the VLSI Journal, Vol. 26, pp.55-78, 1998.
[6] R. Dandapani, J. Patel and J. Abraham, "Design of Test Pattern Generators for Built-In Test", IEEE Int. Test Conf., pp. 315-319, 1984.
[7] C.W. Starke, "Built-In Test for CMOS Circuits", IEEE Int. Test Conf., pp. 309-314, 1984.
[8] G. Edirisooriya and J.P. Robinson, "Design of Low Cost ROM Based Test Generators", IEEE VLSI Test Symp., pp. 61-66, 1992.
[9] C. Fagot, P. Girard and C. Landrault, "On Using Machine Learning for Logic BIST", IEEE Int. Test Conf., pp. 338-346, 1997.
[10] B. Koenemann, "LFSR-Coded Test Patterns for Scan Designs", IEEE Euro. Test Conf., pp. 237-242, 1991.
[11] S. Hellebrand, J. Rajski, S. Tarnick, S. Venkataraman and B. Courtois, "Built-In Test for Circuits with Scan Based on Reseeding of Multiple-Polynomial Linear Feedback Shift Registers", IEEE Trans. on Computers, Vol. 44, N° 2, pp. 223-233, February 1995.
[12] S. Hellebrand, H.J. Wunderlich and A. Hertwig, "Mixed-Mode BIST Using Embedded Processors", Journal of Electronic Testing: Theory and Applications (JETTA), Vol. 12, N°1/2, pp. 127-138, February/April 1998.
[13] K. Chakrabarty, B.T. Murray and V. Iyengar, "Built-In Test Pattern Generation for High Performance Circuits Using Twisted-Ring Counters", IEEE VLSI Test Symp., pp. 22-27, 1999.
[14] N.A. Touba and E.J. McCluskey, "Altering a Pseudo-Random Bit Sequence for Scan-Based BIST", IEEE Int. Test Conf., pp. 167-175, 1996.
[15] N.A. Touba and E.J. McCluskey, "Applying Two-Pattern Tests Using Scan-Mapping", IEEE VLSI Test Symp., pp. 393-397, 1996.
[16] C. Fagot, O. Gascuel, P. Girard and C. Landrault, "A Ring Architecture Strategy for BIST Test Pattern Generation", IEEE Asian Test Symposium, pp. 418-423, 1998.
[17] G. Kiefer; H. Vranken, E.J. Marinissen and H.J. Wunderlich, "Application of Deterministic Logic BIST on Industrial Circuits", IEEE Int. Test Conf., pp. 105-114, 2000.
[18] J. Savir and P.H. Bardell, "On Random Pattern Test Length", IEEE Trans. on Computers, Vol. C-33, N° 6, pp. 467-474, 1984.
[19] T. Williams, "Test Length in a Self-Testing Environment", IEEE Design & Test of Computers, Vol. 2, N° 2, pp. 59-63, 1985.
[20] K.D. Wagner, C.K. Chin and E.J. McCluskey, "Pseudo-Random Testing", IEEE Trans. on Computers, Vol. C-36, N° 3, pp. 332-343, 1987.
[21] H.J. Wunderlich, "Self Test Using Unequiprobable Random Patterns", IEEE Int. Symp. on Fault-Tolerant Computing, pp. 258-263, 1987.
[22] S. Pilarski, A. Pierzynska, "BIST and Delay Fault Detection", IEEE Int. Test Conf., pp. 236-241, 1993.
[23] W. Wang and S.K. Gupta, "Weighted Random Robust Path Delay Testing of Synthesized Multilevel Circuits", IEEE VLSI Test Symp., pp. 291-297, 1994.
[24] P. Girard, C. Landrault, V. Moreda and S. Pravossoudovitch, "An Optimized BIST Test Pattern Generator for Delay Testing", IEEE VLSI Test Symposium, pp. 94-99, 1997.
[25] C. Fagot, O. Gascuel, P. Girard and C. Landrault, "On Calculating Efficient LFSR Seeds for Built-In Self Test", IEEE European Test Workshop, pp. 7-14, 1999.
[26] E.J. McCluskey, "Verification Testing: A Pseudo-Exhaustive Test Technique", IEEE Trans. on Computers, Vol. C-33, N° 6, pp. 541-546, 1984.
[27] G.L. Craig and C.R. Kime, "Pseudo-Exhaustive Adjacency Testing: A BIST Approach for Stuck-Open Faults", IEEE Int. Test Conf., pp. 126-137, 1985.
[28] A. Vuksic and K. Fuchs, "A New BIST Approach for Delay Fault Testing", IEEE VLSI Test Symp., pp. 284-288, 1994.
[29] R.C. Aitken, "Nanometer Technology Effects on Fault Models for IC Testing", IEEE Computer, Vol. 32, N° 11, pp. 46-51, 1999.
[30] P. Nigh, W. Needham, K. Butler, P. Maxwell and R. Aitken, "An Experimental Study Comparing the Relative Effectiveness of Functional, Scan, Iddq and Delay Fault Testing", IEEE VLSI Test Symp., pp. 459-464, 1997.
[31] S.C. Ma, P. Franco and E.J. McCluskey, "An Experimental Chip to Evaluate Test Techniques Experiment Results", IEEE Int. Test Conf., pp. 663-672, 1995.
[32] C.Chen and S.K. Gupta, "BIST Test Pattern Generators for Stuck-Open and Delay Testing", IEEE Euro. Design & Test Conf.., pp. 289-296, 1994.
[33] A. Virazel, R. David, P. Girard, C. Landrault, and S. Pravossoudovitch, "Delay Fault Testing: Effectiveness of Random SIC and Random MIC Test Sequences", IEEE European Test Workshop, pp. 9-14, 2000.
[34] M. Abramovici, M. Breuer and A. Friedman, "Digital System Testing and Testable Design", IEEE Press, Piscataway, NJ, 1990.
[35] K. Mei, "Bridging and Stuck-at Faults", IEEE Trans. on Computers, Vol. C-23, N° 7, pp. 720-727, 1974.
[36] D. Lavo, B. Chess, T. Larrabee and F. Ferguson, "Diagnosing Realistic Bridging Faults with Single Stuck-at Information", IEEE Trans. on CAD, Vol. C-17, N° 3, pp. 255-267, 1998.
[37] S. Ma, I. Shaik and R. Fetherston, "A Comparison of Bridging Fault Simulation Methods", IEEE Int. Test Conf., pp. 587-595, 1999.
[38] G.L. Smith, "Model for Delay Faults Based upon Paths", IEEE Int. Test Conf., pp. 342-349, 1985.
[39] A. Krstic and K.T. Cheng, "Delay Fault Testing for VLSI Circuits", Kluwer Academic Publishers, Boston, 1998.
[40] R. David, "Random Testing of Digital Circuits: Theory and Applications", Marcel Dekker, Inc., New York, 1998.

[41] R. David, P. Girard, C. Landrault, S. Pravossoudovitch and A. Virazel, “On Hardware Generation of Random Single Input Change Test Sequences”, proposed to IEEE European Test Workshop, 2001.
[42] P. Girard, C. Landrault, S. Pravossoudovitch and A. Virazel, “Comparison Between Random and Pseudo-Random Generation for BIST of Delay, Stuck-at and Bridging Faults”, IEEE On-Line Testing Workshop, pp. 121-126, 2000.
[43] TestGen, version Tg4.1 User Guide, Synopsys Inc., 1999.