High Defect Coverage with Low-Power Test Sequences in a BIST Environment

Patrick Girard, Christian Landrault, and Serge Pravossoudovitch
LIRMM

Arnaud Virazel and Hans-Joachim Wunderlich
University of Stuttgart

A new technique, random single-input change (RSIC) test generation, generates low-power test patterns that provide a high level of defect coverage during low-power BIST of digital circuits. The authors propose a parallel BIST implementation of the RSIC generator and analyze its area-overhead impact.

TESTING RANKS among the most expensive and difficult aspects of the circuit design cycle, driving the need for innovative solutions. To this end, researchers have proposed built-in self-test (BIST) as a powerful DFT technique for addressing highly complex VLSI testing problems. BIST designs include on-chip circuitry to provide test patterns and analyze output responses. Performing tests on the chip greatly reduces the need for complex external equipment.

The most commonly used fault model for BIST of digital systems is the classical single stuck-at fault model. However, in the new CMOS nanometer technologies, defects do not always behave as stuck-at faults do.[1] Therefore, test generation based on the stuck-at model alone is no longer sufficient for obtaining high defect coverage.[2] A straightforward solution covering many misbehaviors that can occur in digital circuits would be to use multiple test generation techniques, each targeting a specific fault type (such as stuck-at, delay, or bridging). However, this solution is costly and impractical from a BIST viewpoint.

An alternative solution might be the use of a single on-chip test pattern generator providing "universal" test sequences—sequences that target both conventional (stuck-at) and unconventional (delay, bridging, and stuck-open) fault types. The problem with this solution is that test application consumes excessive power because switching activity is significantly higher during test than during system mode.[3] A standard test pattern sequence produced by a linear feedback shift-register (LFSR) consists of random multiple-input change (RMIC) patterns with a switching-activity rate of 0.5 (that is, an equal likelihood of 0 and 1). In this article, we propose a new BIST test generation technique that reduces the switching activity of test patterns generated during BIST while increasing the defect coverage. The technique, called random single-input change (RSIC), generates test patterns capable of detecting many different faulty behaviors. Moreover, RSIC test sequences have a low rate of switching activity.

Basic definitions

Fault models describe a faulty system's logical behavior. In our work, we use the bridging, path delay, and single stuck-at fault models to analyze the effectiveness of test sequences produced by RSIC generation.

0740-7475/02/$17.00 © 2002 IEEE

Fault models

We can view a bridging fault as an unintentional short between two lines. This short can be a nonresistive short, such that the two lines are always brought into equilibrium at the same potential, or a resistive short, such that the shorted lines have different potentials. Researchers have proposed several models for resistive and nonresistive bridging faults, based on the popular wired-AND and wired-OR models used to model the effects of bridging faults in bipolar logic.[4] Although the coverage of a bridging fault by both wired-AND and wired-OR behavior does not always guarantee detection, these models are easy to use for bridging-fault simulation. To allow experiments with a larger number of benchmark circuits, the work described in this article used such models to represent bridging defects.

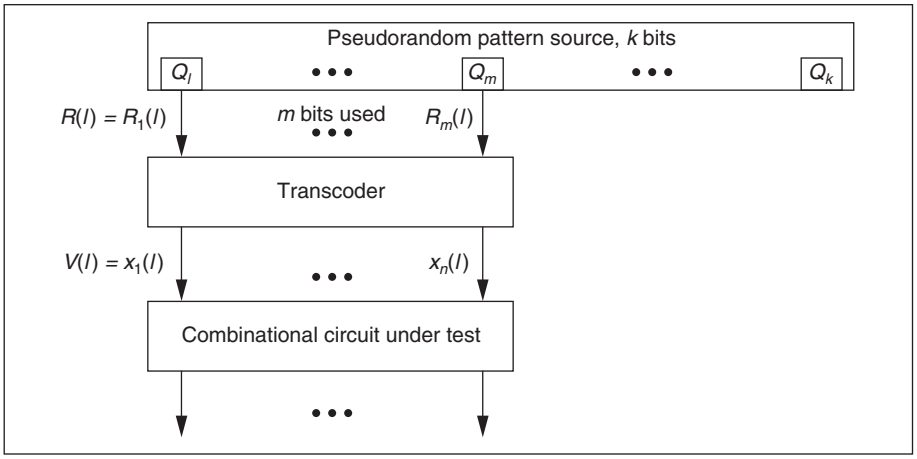


Figure 1. Generation principle of an RSIC test sequence.

Some defects in a manufactured circuit do not affect its logic function; rather, they change its delays, thus changing its operating speed. These are delay defects, and a widely studied delay fault model is the path delay fault model.[5] The main advantage of the path delay fault model is that it models the distributed delay defects more accurately than other delay fault models, particularly the gate delay fault model.

A test for a path is robust if it can detect a delay fault on that path irrespective of other delays and delay faults in the circuit; otherwise, it is nonrobust.[5] A robust test is preferable because a single defect usually affects many paths.

Test sequences

Two-pattern tests might differ in multiple bit positions. In that case, they are called multiple-input change (MIC) pattern pairs. Test pairs that differ in only one bit position are called single-input change (SIC) pattern pairs. We now define what an RSIC sequence should be in theory. Let

$$S = V(1)V(2) \ldots V(l) \ldots V(L)$$

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

$$V = \{V_0, V_1, \ldots, V_{2^n-1}\}$$

where V_j corresponds to the n-bit vector $(x_1, x_2, \ldots, x_n)$ associated with decimal value j. For example, for $n = 5$, $V_9 = 01001$; that is, $x_1 = x_3 = x_4 = 0$ and $x_2 = x_5 = 1$. In an RMIC sequence, the probability $\Pr[V(l) = V_j] = 2^{-n}$ for any l and any j, and the probability $\Pr[V(l) = V_j]$ is independent of the values $V(i)$, where $i = 1, 2, \ldots, l-1$.

In an RSIC sequence, this probability is

$$\Pr[V(l) = V_j \mid V(l-1) = V_k] = 1/n, \quad \text{if and only if } |j-k| = 2^a \qquad (1)$$

where $a \in \{0, n-1\}$. 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.

Hardware generation of random test sequences

Figure 1 represents the basic RSIC generation principle.[6] The structure uses a k-bit random source, which can be a random number obtained from a maximal-length LFSR. The value of vector $Q_1Q_2 \ldots Q_k$ changes at each clock cycle of the test session. At each time l, the transcoder uses a subset of m bits ($m \le k$). The transcoder transforms vector $R(l) = R_1(l)R_2(l) \ldots R_m(l)$ into n-bit vector $V(l) = x_1(l)x_2(l) \ldots x_n(l)$,

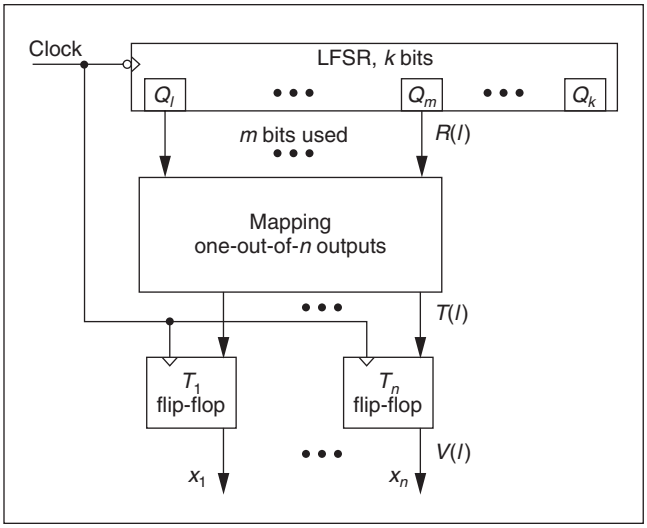


Figure 2. Hardware generation of an RSIC test sequence.

which is applied to a combinational circuit under test. The definition of an RSIC sequence requires the following conditions:

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

The period of the RSIC sequence generated from an appropriate structure according to the RSIC principle is $2 \times (2^k - 1)$,[6] if length k of the pseudorandom source obeys the relationship $2 \times (2^k - 1) > L$; that is,

$$k > \log_2(L/2 + 1) \approx \log_2(L/2)$$

Researchers have proposed structures for generating RMIC sequences in which two consecutive patterns are independent. René David[7] obtains RMIC test sequences from a combination of LFSRs in which more than one shifting (σ) occurs between two consecutive vectors.

Similarly, David obtains RSIC test sequences from the hardware structure shown in Figure 2. This structure maps vector $R(l)$ coming from the modified LFSR into a one-out-of-n vector $T(l)$ and applies every component of $T(l)$ to the input of a T flip-flop. Hence, given random vector $R(l)$, this mapping implies random trigger input $T_i(l) = 1$ and other trigger inputs $T_j(l) = 0$ for $j \neq i$. Therefore, $V(l)$ is similar to $V(l-1)$ apart from the value of x_i. The mapping cell must be a one-out-of-n decoder.

This hardware RSIC generator has the following theoretical properties:

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

$$\max_{j \in \{1,\ldots,n\}} b_j - \min_{j \in \{1,\ldots,n\}} b_j \leq 1.$$

RSIC test efficiencies

We performed tests to compare the effectiveness of RSIC test sequences with that of RMIC sequences in terms of delay, bridging, and single stuck-at fault coverage.

Delay fault testing

For delay fault testing, we used a case study corresponding to experiments performed with the combinational part of circuit s382 of the 1989 International Symposium on Circuits and Systems (ISCAS) benchmark set. We base our performance evaluation of the RSIC and RMIC test sequences on results we obtained using a deterministic automatic test-pattern generation (ATPG) tool. We use the following notations:

- *FC_ATPG robust*. Robust fault coverage obtained.
- *FC_ATPG nonrobust*. Nonrobust fault coverage obtained.
- *Ld*. Test length obtained by an ATPG tool on the path delay fault model with a complete efficiency.

Using the ATPG information, we computed the results provided by fault simulation with the RSIC and RMIC test sequences. We obtained the following efficiencies:

Table 1. Robust and nonrobust efficiencies of given RMIC and RSIC sequences run on a combinational part of the s382 benchmark circuit.

Multiple of deterministic test length	Length (no. of patterns)	RMIC sequence		RSIC sequence	
		Eff_R %	*Eff_NR* %	*Eff_R* %	*Eff_NR* %
—	10	12.78	16.49	1.70	1.63
—	100	25.57	52.04	14.35	15.40
Ld/4	349	30.97	73.97	24.57	25.61
Ld/2	699	33.09	86.78	43.47	46.05
Ld	1,398	33.24	91.01	52.84	57.22
2*Ld*	2,796	33.52	96.86	61.36	66.49
5*Ld*	6,990	33.52	99.31	76.56	84.33
10*Ld*	13,980	33.52	99.86	81.53	90.32
20*Ld*	27,960	33.52	99.86	86.65	96.19
50*Ld*	69,900	33.52	99.86	89.06	98.64
100*Ld*	139,800	33.52	99.86	90.19	99.73
—	1,000,000	33.52	99.86	90.48	100.00

- *Eff_R*. The robust fault efficiency achieved by RSIC or RMIC test sequences is

  $$Eff\_R = 100 \times (FC\ robust / FC\_ATPG\ robust)$$

  where *FC robust* is the fault coverage achieved by the fault simulation.
- *Eff_NR*. The nonrobust fault efficiency achieved by RSIC or RMIC test sequences is

  $$Eff\_R = 100 \times (FC\ nonrobust / FC\_ATPG\ nonrobust)$$

  where *FC nonrobust* is the fault coverage achieved by the fault simulation.

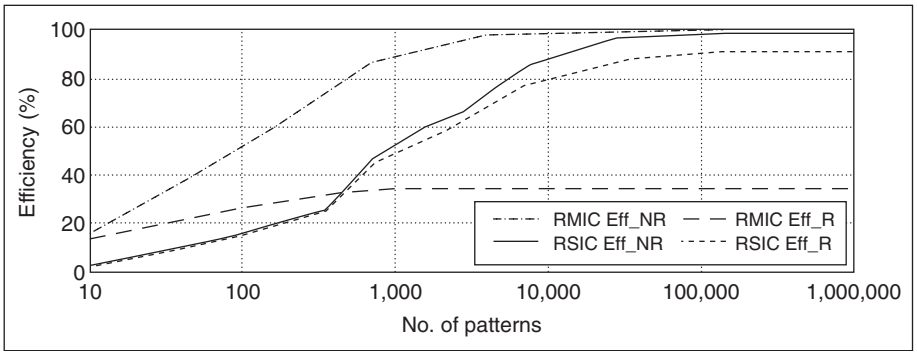


Figure 3. Efficiency comparison of RMIC and RSIC sequences on the s382 benchmark circuit.

Basic results. Table 1 presents simulation results obtained from one RMIC sequence and one RSIC sequence. The table shows results for various test lengths as multiples of required deterministic test length *Ld*. Figure 3 is a graphical representation of the results given in Table 1.

From these results, we make the following observations:

- The RSIC sequence's robust efficiency is close to its nonrobust efficiency, whereas there is a huge gap between these efficiencies for the RMIC sequence. The reason is that most of the tests in the RSIC sequence are robust because of the adjacency of successive vectors.
- For very short test lengths (which are uninteresting in practice), the RSIC test sequence's fault efficiencies are lower than those of the RMIC sequence. For medium-range test lengths, the RMIC sequence's nonrobust efficiency is higher than that of the RSIC sequence, whereas the RSIC sequence is higher in robust efficiency.
- For long test lengths, the RSIC test sequence can equal the RMIC sequence in nonrobust efficiency, but it is much higher in robust efficiency.

Table 2. Actual efficiencies for various success rate (S_{NR}) assumptions.

	RMIC sequence			RSIC sequence		
Length (no. of patterns)	*EFF* (S_{NR} = 10%)	*EFF* (S_{NR} = 50%)	*EFF* (S_{NR} = 90%)	*EFF* (S_{NR} = 10%)	*EFF* (S_{NR} = 50%)	*EFF* (S_{NR} = 90%)
10	12.60	13.98	15.36	1.63	1.60	1.58
50	21.92	27.88	33.83	8.51	8.78	9.06
100	26.98	36.82	46.66	13.86	14.25	14.64
349	33.69	49.67	65.65	23.66	24.05	24.43
699	36.72	56.67	76.62	41.92	42.88	43.84
1,398	37.24	58.71	80.18	51.08	52.71	54.34
2,796	38.03	61.57	85.11	59.32	61.23	63.13
6,990	38.26	62.71	87.16	74.14	77.03	79.91
13,980	38.31	62.96	87.61	79.01	82.27	85.54
27,960	38.31	62.96	87.61	83.98	87.53	91.07
69,900	38.31	62.96	87.61	86.30	89.86	93.42
139,800	38.31	62.96	87.61	87.38	90.92	94.47
1,000,000	38.31	62.96	87.61	87.65	91.19	94.73

In summary, the RMIC sequence achieves better nonrobust efficiency, and the RSIC sequence achieves better robust efficiency (for medium-range test lengths). Later, we try to find a criterion for selecting the best test sequence to target both robust and nonrobust efficiency.

Actual efficiency. During test application, an uncertainty arises as to what proportion of nonrobustly tested faults a sequence really detects. We define actual efficiency (*EFF*) as

$$EFF(S) = Eff\_NR \times S \quad (2)$$

Success rate S is the proportion of faults really detected during the test. For a robust test set, the success rate is $S_R = 1.0$, and *FC_robust* is the portion of the corresponding faults. The portion of faults that are nonrobustly testable but do not have a robust test pattern is *FC_nonrobust* – *FC_robust*, and the corresponding test set has success rate $S_{NR} < 1.0$. Hence, the overall success rate is

$$\begin{aligned} S &= FC\_robust \times S_R + (FC\_nonrobust - FC\_robust) \times S_{NR} \\ &= FC\_robust \times (1.0 - S_{NR}) + FC\_nonrobust \times S_{NR} \end{aligned}$$

Table 2 shows the efficiency results of applying Equation 2 to circuit s382 for the RMIC and RSIC test sequences and for S_{NR} = 10%, 50%, and 90%. For a long test length, actual efficiency is higher for the RSIC sequence than for the RMIC sequence, for all assumed values of S_{NR}.

These results demonstrate that even with a lower nonrobust delay fault coverage, RSIC test sequences often produce better test quality than do RMIC delay test sequences. This conclusion, drawn from a study of circuit s382, is also valid for most of the ISCAS89 benchmark circuits.[8]

Stuck-at and bridging fault testing

Table 3 shows that for stuck-at fault coverage, the effectiveness of RSIC generation in comparison with RMIC generation mainly depends on the test sequence's length. For lengths of 10,000, the fault efficiency of RMIC test sequences is slightly higher than that of RSIC test sequences. This is also true for lengths below 10,000. But when the length increases beyond 10,000, the efficiency of RSIC test sequences becomes comparable to that of RMIC test sequences. For a test length equal to that used for delay fault testing, the fault efficiency is nearly the same for both types of sequences.

The same conclusion is true for bridging-fault coverage, as Table 4 shows. For each

Table 3. Comparison of RSIC and RMIC generation for single stuck-at fault coverage.

Circuit	No. of faults	Length (no. of patterns)	RSIC efficiency (%)	RMIC efficiency (%)	Length (no. of patterns)	RSIC efficiency (%)	RMIC efficiency (%)
s386	360	10,000	100.00	100.00	73,600	100.00	100.00
s420	424	10,000	77.12	83.25	132,200	96.10	97.88
s510	538	10,000	100.00	100.00	136,600	100.00	100.00
s1238	1,332	10,000	90.99	97.94	499,200	99.84	100.00
s1494	1,489	10,000	99.66	99.94	351,800	100.00	100.00

Table 4. Comparison of RSIC and RMIC generation for bridging-fault coverage.

			RSIC sequences		RMIC sequences	
Circuit	No. of faults	Length (no. of patterns)	WAND&WOR coverage (%)	WAND||WOR coverage (%)	WAND&WOR coverage (%)	WAND||WOR coverage (%)
s386	390	73,600	99.23	100.00	99.23	100.00
s420	521	132,200	81.23	97.59	81.96	98.89
s510	1,073	136,600	100.00	100.00	100.00	100.00
s1238	2,924	499,200	99.90	100.00	99.90	100.00
s1494	3,851	351,800	99.95	100.00	99.95	100.00

bridging fault, we considered the following behaviors: WAND (wire-AND), WOR (wire-OR), WAND&WOR (the fault is tested if both behaviors are tested), and WAND||WOR (the fault is tested if at least one of the two behaviors is tested). Table 4 reports the results for the most representative WAND&WOR and WAND||WOR models.

RSIC generator implementation

A classical BIST architecture must incorporate a test pattern generator (TPG), a test response evaluator (TRE), and a BIST control unit. State-of-the-art implementations use either of two basic BIST execution options: the serial (test-per-scan) scheme or the parallel (test-per-clock) scheme.[9]

In serial BIST, the TPG shifts test vectors into a serial scan path and then applies them to the circuit under test. Next, it loads test responses into the scan chain and shifts them out to the TRE while shifting in a new test.

In parallel BIST, the TPG applies test vectors at every clock cycle. In the same clock cycle, the TRE captures test responses. This scheme leads to shorter test times than the test-per-scan scheme because the TPG generates a new test pattern in each clock cycle. This scheme can perform a high-speed test at system frequency without any clock delays for shifting. The parallel scheme's drawbacks are its hardware overhead and performance degradation.

BIST environment

The delay-fault detection capability offered by parallel BIST led us to propose a parallel BIST hardware implementation of the RSIC generator. Our test-per-clock design uses modified system registers including built-in logic block observers (BILBOs)[10] and concurrent built-in logic block observers (CBILBOs).[11]

Figure 4a (next page) presents the original circuit, which contains a combinational logic block, an internal state register, and registers at the primary inputs and outputs. Figure 4b shows the standard parallel BIST approach, which changes the input and output registers into BILBOs and the state register into a CBILBO.

Figure 4c shows our RSIC implementation. Here, we added a *k*-stage LFSR and mapping logic. The *k*-stage LFSR is a classical primitive polynomial LFSR. The mapping logic is a simple combinational structure allowing an *m*-bit vector provided by the LFSR to be mapped into

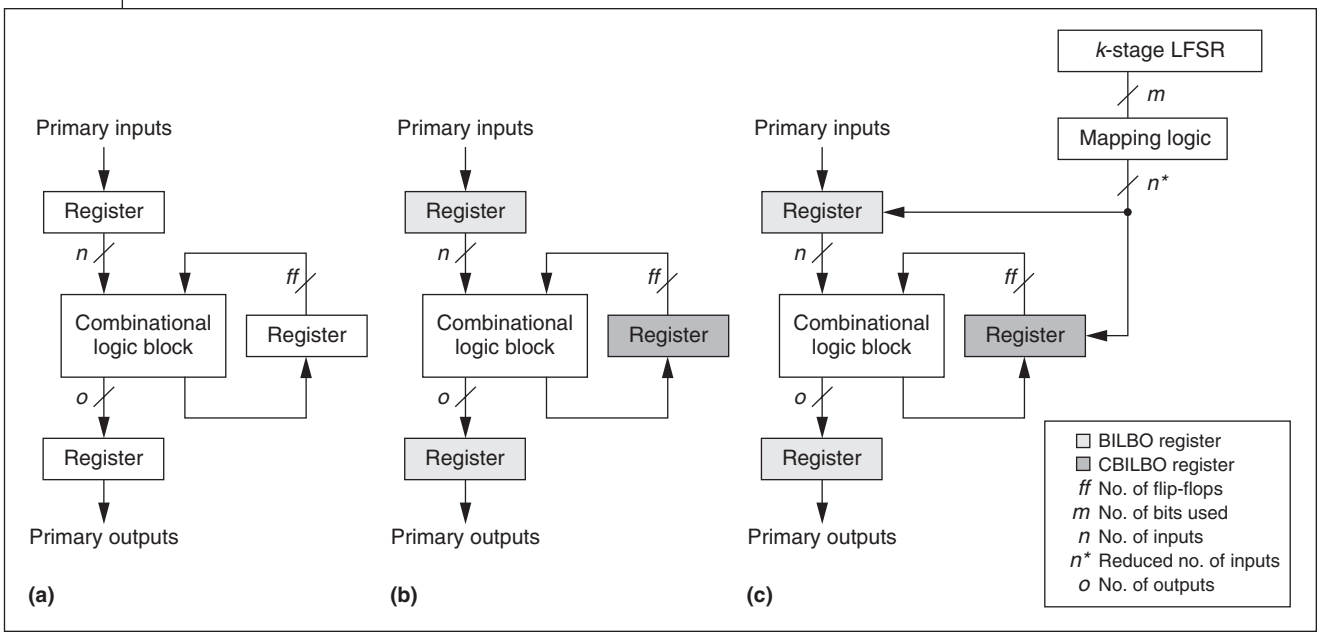


Figure 4. BIST implementations: original (a), classical parallel (b), and RSIC (c) designs.

a one-out-of-*n* vector. We further reduced this logic by identifying compatible inputs of each circuit. Compatible inputs are inputs that belong to the circuit's output cones.[12] We thus reduced the test of an *n*-input circuit to the test of an *n**-input circuit and also reduced the test length. Moreover, we modified the standard BILBO and CBILBO cells to enable hardware generation of RSIC test sequences from the structure in Figure 2 (in T flip-flop mode).

Area overhead

Table 5 shows the proposed RSIC generator's gate count overhead. The column *Cell_area* presents an estimate of the area (in cells only) in each original circuit. Column S_1 reports the area required to modify the registers into BILBOs and CBILBOs, as in the design in Figure 4b. Column S_2 gives the additional area required to transform the original circuit into an RSIC BIST design, as in Figure 4c. The last two columns show the area overhead the RSIC generator imposes on a conventional BILBO architecture, as determined by these two equations:

$$Cost_1 = S_2/S_1$$

$$Cost_2 = (S_2 + Cell\_area)/(S_1 + Cell\_area)$$

$Cost_1$ is the factor by which the RSIC hardware is larger than the RMIC hardware, and $Cost_2$ is the factor by which an RSIC-testable circuit is larger than a circuit with standard BIST.

The $Cost_2$ results show that the additional cost required to generate RSIC test sequences for each circuit is between 19% and 13% for the largest ISCAS89 circuits (s5378 through s38584). This percentage decreases as the circuit size increases. From these results, we can conclude that the RSIC generation method is a practical way to reach a high level of defect coverage for digital-circuit BIST.

Because of the RSIC generation technique's effectiveness, we plan to extend it to system-on-a-chip designs. In SoCs, an embedded processor could use software to generate RSIC sequences. This might increase test application time, but it would not require any extra hardware. ■

Acknowledgments

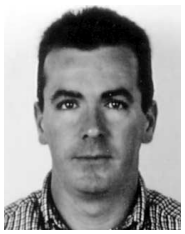
We thank René David of the Laboratoire d'Automatique de Grenoble for his participation in this work. DFG (Deutsche Forschungsgemeinschaft) grant WU 254/2-1 partially supported this work.

Table 5. Area overhead of the RSIC generator.

Circuit	No. of primary inputs	No. of primary outputs	No. of flip-flops	*Cell_area* (gate equivalents)	S_1 (gate equivalents)	S_2 (gate equivalents)	$Cost_1$	$Cost_2$
s510	25	7	6	435.5	190.5	456.0	2.394	1.424
s526	24	6	21	444.5	286.5	552.0	1.927	1.363
s641	54	24	19	789.0	488.5	867.0	1.775	1.296
s713	54	23	19	813.0	484.0	862.5	1.782	1.292
s820	23	19	5	698.5	228.5	490.5	2.147	1.283
s832	23	19	5	707.5	228.5	490.5	2.147	1.280
s953	45	23	29	822.0	513.5	860.5	1.676	1.260
s1196	32	14	18	890.0	337.5	647.5	1.919	1.253
s1238	32	14	18	919.5	337.5	647.5	1.919	1.247
s1423	91	5	74	1,325.0	954.5	1,620.5	1.698	1.292
s1488	14	19	6	1,166.5	195.0	413.0	2.118	1.160
s1494	14	19	6	1,173.5	195.0	413.0	2.118	1.159
s5378	214	49	179	3,803.5	2,441.0	3,579.0	1.466	1.182
s9234	247	22	228	6,292.5	2,811.0	4,210.5	1.498	1.154
s13207	700	121	669	11,321.5	8,382.0	12,120.5	1.446	1.190
s15850	611	87	597	12,175.0	7,324.5	10,602.0	1.447	1.168
s35932	1,763	320	1,728	29,224.5	21,474.0	28,652.5	1.334	1.142
s38417	1,664	106	1,636	28,850.5	19,421.5	26,527.5	1.366	1.147
s38584	1,464	278	1,452	30,898.5	18,007.5	24,525.5	1.362	1.133

References

1. R.C. Aitken, "Nanometer Technology Effects on Fault Models for IC Testing," *Computer*, vol. 32, no. 11, Nov. 1999, pp. 46-51.
2. P. Nigh et al., "An Experimental Study Comparing the Relative Effectiveness of Functional, Scan, I_{DDQ} and Delay Fault Testing," *Proc. 15th IEEE VLSI Test Symp.* (VTS 97), IEEE CS Press, Los Alamitos, Calif., 1997, pp. 459-464.
3. P. Girard, "Survey of Low-Power Testing of VLSI Circuits," *IEEE Design & Test of Computers*, vol. 19, no. 3, May-June 2002, pp. 82-92.
4. K. Mei, "Bridging and Stuck-At Faults," *IEEE Trans. Computers*, vol. 23, no. 7, July 1974, pp. 720-727.
5. G.L. Smith, "Model for Delay Faults Based upon Paths," *Proc. Int'l Test Conf.* (ITC 85), IEEE CS Press, Los Alamitos, Calif., 1985, pp. 342-349.
6. R. David et al., "On Hardware Generation of Random Single-Input Change Test Sequences," *Proc. 6th IEEE European Test Workshop* (ETW 01), IEEE CS Press, Los Alamitos, Calif., 2001, pp. 117-123.
7. R. David, *Random Testing of Digital Circuits: Theory and Applications*, Marcel Dekker, New York, 1998.
8. A. Virazel et al., "Delay Fault Testing: Choosing between Random SIC and Random MIC Test Sequences," *Proc. 5th IEEE European Test Workshop* (ETW 00), IEEE CS Press, Los Alamitos, Calif., 2000, pp. 9-14.
9. M.L. Bushnell and V.D. Agrawal, *Essentials of Electronic Testing*, Kluwer Academic, Boston, 2000.
10. B. Konemann, J. Mucha, and G. Zwiehoff, "Built-In Logic Block Observation Technique," *Proc. Int'l Test Conf.* (ITC 79), IEEE CS Press, Los Alamitos, Calif., 1979, pp. 37-41.
11. L.-T. Wang and E.J. McCluskey, "Concurrent Built-in Logic Block Observer (CBILBO)," *Proc. Int'l Symp. Circuits and Systems* (ISCAS 86), IEEE CS Press, Los Alamitos, Calif., 1986, pp. 1054-1057.
12. E.J. McCluskey, "Verification Testing: A Pseudo-Exhaustive Test Technique," *IEEE Trans. Computers*, vol. 33, no. 6, June 1984, pp. 541-546.



Patrick Girard is a full researcher in the Microelectronics Department of LIRMM (Laboratory of Informatics, Robotics, and Microelectronics of Montpellier) in France. His research interests include various aspects of digital testing with special emphasis on DFT, logic BIST, delay-fault testing, and low-power testing. Girard has a BSc and an MSc in electrical engineering and a PhD in microelectronics, all from the University of Montpellier. He is a member of the French National Center for Scientific Research (CNRS).

Christian Landrault is a research director at LIRMM. His research interests include fault modeling, testing, fault simulation, ATPG, DFT, and BIST. Landrault has a certified engineering degree from the Higher National School for Aeronautical Constructions, Toulouse, France. He has a PhD in automatic control and the Doctorat d'État in computer science, both from the National Polytechnic Institute of Toulouse.

Serge Pravossoudovitch is a professor in the Electrical and Computer Engineering Department of the University of Montpellier and a researcher at LIRMM. His research interests include delay-fault testing and power optimization during test. Pravossoudovitch has a PhD in electrical engineering from the University of Montpellier. He received the Doctorat d'État degree for his works on switch-level ATPG.

Arnaud Virazel is a research and teaching assistant in the Computer Architecture Laboratory of the University of Stuttgart, Germany. His research interests include delay-fault testing and BIST. Virazel has a BSc and an MSc in electrical engineering and a PhD in microelectronics, all from the University of Montpellier.

Hans-Joachim Wunderlich is the head of the Division of Computer Architecture at the University of Stuttgart. He has a Dipl.-Math. in mathematics from the University of Freiburg, Germany, and a PhD in computer science from the University of Karlsruhe, Germany.

■ Direct questions and comments about this article to Arnaud Virazel, Division of Computer Architecture, Univ. of Stuttgart, Breitwiesenstrasse 20-22, 70565 Stuttgart, Germany; virazela@informatik.uni-stuttgart.de, virazel@lirmm.fr.

For further information on this or any other computing topic, visit our Digital Library at http://computer.org/publications/dlib.