Open access • Journal Article • DOI:10.1109/TVLSI.2004.837985

Weighted pseudorandom hybrid BIST — Source link

A. Jas, C.V. Krishna, Nur A. Touba

Institutions: Intel, University of Texas at Austin

Published on: 01 Dec 2004 - IEEE Transactions on Very Large Scale Integration Systems (IEEE)

Topics: Fault coverage, Built-in self-test, Overhead (computing), Pseudorandom number generator and Random number generation

Weighted Pseudorandom Hybrid BIST

Abhijit Jas, *Member, IEEE*, C. V. Krishna, *Member, IEEE*, and Nur A. Touba, *Member, IEEE*

***Abstract*—This paper presents a new test data-compression scheme that is a hybrid approach between external testing and built-in self-test (BIST). The proposed approach is based on weighted pseudorandom testing and uses a novel approach for compressing and storing the weight sets. Three levels of compression are used to greatly reduce test costs. Experimental results show that the proposed scheme reduces tester storage requirements and tester bandwidth requirements by orders of magnitude compared to conventional external testing, but requires much less area overhead than a full BIST implementation providing the same fault coverage. No test points or any modifications are made to the function logic. The paper describes the proposed hybrid BIST architecture as well as two different ways of storing the weight sets, which are an integral part of this scheme.**

I. Introduction

THE MOVE to deep submicron technology has led to greater integration density. As the amount of logic on the chip has increased, the amount of test data volume required for testing such large designs is also growing rapidly. Moreover, current generation testers have limited speed, memory, and I/O channels, and upgrading to more advanced testers can be very costly. Hence, conventional external testing approaches where all test data is stored on the tester and transferred to and from the circuit-under-test (CUT) is becoming increasingly difficult. The limited test data bandwidth (product of the number of tester channels and clock speed) between the tester and the chip is becoming a major bottleneck that is expected to become much worse as the projections in [13] indicate. There is a need for new test data compression schemes that reduce test data bandwidth requirements and reduce tester storage requirements by orders of magnitude.

One well-known approach is to use built-in self-test (BIST). BIST involves performing test pattern generation and output response compaction on the chip. BIST has been studied for many years. The most economical BIST schemes are based on pseudorandom pattern testing. The problem with pseudorandom pattern testing, however, is that it generally does not provide high enough fault coverage due to the presence of random-pattern-resistant faults [7]. There are two solutions to this problem. One is to modify the circuit to eliminate the random pattern resistance by inserting test points [7], and the other is to modify the test pattern generator by adding additional hardware to generate patterns that detect the hard faults [8], [14], [15], [22]. Both approaches have significant drawbacks. Test point insertion requires modifying the function logic which can degrade system performance, and modifying the test pattern generator can require large amounts of additional silicon area.

In this paper, a new test data compression scheme is presented, which is a hybrid approach between BIST and external testing (preliminary results were published in [12]). The term "hybrid BIST" will be used in this paper to classify any scheme that involves combining external data from the tester along with BIST hardware on the chip to provide a hybrid test solution for a particular module or core. A hybrid BIST approach reduces the test data stored on the tester compared with full external testing, but does not require as much hardware overhead as full BIST. A simple approach for hybrid BIST is to use a STUMPS architecture [1] to apply pseudorandom patterns to detect the random pattern testable faults, and then use deterministic scan vectors from the tester to detect the hard faults. There have been a couple of case studies on using this approach for large industrial designs [11], [20]. The case study in [20] was done on the Motorola PowerPC microprocessor core, and the study in [11] was done on large ASIC designs. In [20], the reduction in external test storage requirements after using 500 K BIST patterns was around 30%. In [11], test points were inserted, but the reduction in test storage requirements after 262 K BIST patterns still ranged only from 35% to 55%. What these results indicate is that most of the vectors in a deterministic test set target hard faults which are missed by pseudorandom patterns. So a straightforward hybrid BIST approach where pseudorandom vectors are applied with BIST hardware followed by deterministic vectors from the external tester, can only achieve a limited reduction in tester storage requirements, generally not an order of magnitude reduction.

There are several other approaches that can be classified as hybrid BIST approaches. In [5], a hybrid BIST approach was proposed where some of the scan chains in a STUMPS architecture are filled with deterministic test data from the tester while the rest of the scan chains are filled from the pseudorandom pattern generator (PRPG). The set of scan chains receiving deterministic data is rotated in a round-robin fashion. This approach was applied to the Motorola PowerPC microprocessor core. Results indicated that the test storage requirements could be reduced by around 50% with this approach compared with 31% as was reported in [20] for using fully pseudorandom patterns followed by fully deterministic patterns.

Manuscript received January 5, 2004; revised June 6, 2004. This work was supported in part by the National Science Foundation under Grant MIP-9702236, and in part by the Texas Advanced Research Program under Grant 1997-003658-369.

A. Jas is with Intel Corporation, Austin, TX 78746 USA, and is also with the Computer Engineering Research Center, Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX 78712-1084 USA.

C. V. Krishna is with Cadence Design Systems Inc., Endicott, NY 13760 USA, and is also with the Computer Engineering Research Center, Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX 78712-1084 USA.

N. A. Touba is with the Computer Engineering Research Center, Department of Electrical and Computer Engineering, Engineering Science Building, University of Texas at Austin, Austin, TX 78712-1084 USA (e-mail: touba@ece.utexas.edu).

Digital Object Identifier 10.1109/TVLSI.2004.837985

The scheme described in [10] uses a test pattern generator that resembles a programmable Johnson counter and is called a "folding counter." The hardware is designed such that the folding counter can switch to a mode where it can generate a certain number of pseudorandom patterns to detect the easy faults within the circuit. A heuristic is then used to determine the minimal number of seeds for the folding counter such that the resulting folding counter sequences cover the deterministic test cubes (test vectors with don't cares) for the hard faults. The scheme in [17] uses a two-dimensional (2-D) approach where the deterministic test cubes for the hard faults are encoded as seeds of an LFSR (horizontal compression), and the seeds are again compressed into seeds of a folding counter sequence (vertical compression).

The hybrid BIST scheme proposed in [6] is based on a RESPIN architecture in which the test vectors for a core under test are decompressed by reusing scan chains of cores idle during that time. While this scheme provides up to two orders of magnitude compression compared with standard compacted test sets, it is applicable only for system-on-a-chip designs where idle scan chains can be reused.

The hybrid BIST scheme proposed in [16] uses a reconfiguration network to selected different sets of outputs from the LFSR when filling multiple scan chains. The idea is to cover a deterministic test cube by trying to select a set of LFSR outputs whose pseudorandom output will match the test cube. While this approach is efficient when the number of specified bits per test cubes is small, it does not scale well as the number of specified bits increases because the number of required configurations increases rapidly and correlations between test cubes cannot be exploited.

The new hybrid BIST approach described in this paper is based on weighted pseudorandom testing. Weighted pseudorandom testing has been studied extensively in the past. It involves biasing the generation of pseudorandom patterns toward those that detect the hard faults. A "weight" is assigned to each bit position in a test vector and corresponds to the probability of a "1" being generated at that bit position. Because of conflicting requirements for detecting hard faults in a circuit, multiple weight sets are generally required [24]. Some number of weighted pseudorandom patterns are generated for each weight set to detect all of the faults. There are two types of weighted pseudorandom testing schemes, one for external testing and one for BIST. For external testing, the weight sets are stored in the tester memory, and the weighted pseudorandom pattern generation is performed on the tester as each test vector is being transferred to the chip [23]. This approach reduces tester memory requirements, but it does not help with the test data bandwidth bottleneck problem because all of the test data still has to be transferred from the tester to the chip. The other scheme for weighted pseudorandom testing is to use it for standalone BIST. In this case, the weight sets are stored on the chip, and on-chip hardware is used to generate the weighted pseudorandom patterns [2], [18], [19] (or the hardware could be placed on a separate "test chip" [21]). The problem with a full BIST implementation of weighted pseudorandom testing is that storing the weight sets on the chip requires an enormous amount of area overhead.

The hybrid weighted pseudorandom scheme described here reduces tester storage requirements and solves the test data bandwidth bottleneck problem, but does not require the area overhead of a full BIST implementation. It uses three levels of compression to provide orders of magnitude reduction in tester storage requirements. In comparison with the approach of performing the weighted pattern generation on the tester, the proposed approach not only reduces tester memory requirements, but also reduces the test data bandwidth requirements from the tester to the chip. If weighted pattern generation is performed on the tester and then used to drive 32 scan chains, it requires 32 channels from the tester, whereas the proposed approach can drive the same number of scan chains with data coming from only a small number of channels from the tester. As system-on-a-chip designs become larger and more complex, this capability will be essential to keep test time down. Note that test time is lower bounded by the total amount of test data stored on the tester divided by the test data bandwidth between the tester and chip (which is limited by the number of I/O pins on the chip and I/O channels from the tester).

A simple approach for implementing a hybrid BIST weighted pseudorandom scheme would be to store all the weight sets on the tester, and then transfer one weight set at a time to the chip. After some number of weighted pseudorandom vectors are generated on the chip for one weight set, the next weight set could be transferred from the tester to the chip. The problem with this approach is that at least 2 bits (or more depending on the precision of the weights) are needed to encode the weight value for each scan element in a design. This means that the storage requirements on the chip for one weight set would be at least double the number of scan elements in the design which would be an enormous area overhead. Fortunately, it turns out that weight sets are highly compressible. This fact is greatly exploited in the scheme proposed in this paper. We present a novel hybrid BIST weighted pseudorandom testing scheme that uses only a small amount of data from the tester to significantly reduce BIST hardware requirements on the chip. The proposed approach reduces tester storage requirements by orders of magnitude compared to full external testing while requiring much less overhead than a full BIST approach that provides the same fault coverage. No test points or any modifications are made to the function logic. The proposed scheme requires adding only a small amount of additional hardware to the STUMPS architecture.

II. Overview of Proposed Scheme

This section describes the basic idea of the proposed scheme for hybrid BIST with weighted pseudorandom testing. Implementation details are explained in subsequent sections. Fig. 3 shows a block diagram of the test architecture. In this scheme, 3-valued weights are used (as was proposed in [19]), i.e., the three possible weights for a specific scan element are **0**, **1**, and **u** (which signifies "unbiased"). A weight of **0** forces the value of a particular scan element to 0, a weight of **1** forces it to 1, and a **u** means that the scan element takes on a value of 0 or 1 with equal probability. In Fig. 3, the scan elements of the chip have been configured into n scan chains each of which contains m scan elements (bits). Since a 3-valued weight system is being

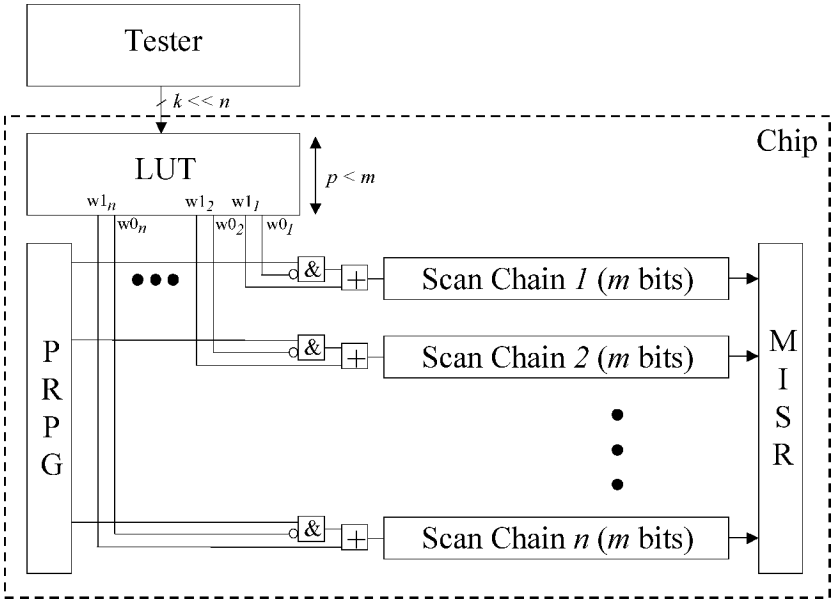

Fig. 1. Block diagram of proposed hybrid BIST test architecture.

used, two bits are required to store the weight for each scan element. The encoding used in this particular example for the three weights are $(w1_i, w0_i) = 01$ for weight **0**, 10 for weight **1**, and 00 for **u**. However, any other 3-valued encoding can be used.

Fig. 1 illustrates the STUMPS architecture for BIST where at each clock cycle, n pseudorandom bits generated by the PRPG are scanned into the n scan chains (one bit into each scan chain). However, in the proposed approach the pseudorandom bits are transformed by the logic at the input of the scan chains according to the weight bits $w1_i$, $w0_i$. The weight bits are stored in a look-up table (LUT) on the chip. At each clock cycle, the set of weights corresponding to the ith bit of every scan chain is looked up from the LUT and used to transform the pseudorandom bits coming out of the PRPG to generate the weighted pseudorandom bits which are then scanned into the scan chains. It takes m scan clock cycles to completely fill the n scan chains. Once the scan chains are filled (i.e., nm scan bits have been shifted into the scan chains) the system clock is applied and the output response is captured in the scan chains. This output response is shifted out and compacted in the multiple input signature register (MISR) as the next test vector is shifted in.

The $2n$ weight bits for the n scan chains $(w1_1, w0_1)$, $(w1_2, w0_2), \ldots, (w1_n, w0_n)$ are stored in one location of the LUT. At each clock cycle, the tester supplies an LUT index which is used to read the weights for the n bits from the LUT. These weights are then used to transform the n pseudorandom bits coming from the PRPG as they are shifted into the scan chains. The tester and the PRPG operate at the same clock frequency in a lock-step manner. The number of bits required for the LUT index depends on the size of the LUT. The number of bits, k, required for the index is generally much less than n. So in this scheme, k tester channels are being used to drive n scan chains, where k is much less than n. Hence, the tester bandwidth requirements are being reduced.

For each weight set, a sequence of m LUT indices are stored on the tester. If L weighted pseudorandom patterns are to be generated for each weight set, then the tester simply resends the sequence of indices for each weight set L times. Only one copy of the sequence of indices for each weight set needs to be stored in the tester memory.

There are three levels of compression in this scheme. The first level of compression is that only the unique parts of each weight set need to be stored in the LUT (this will be explained in detail in the next section), thus for each weight set there will much less than m rows in the LUT. The second level of compression is that each weight set is stored as a sequence of k-bit indices on the tester where k scales logarithmically with the number of rows in the LUT and is much less than n. The third level of compression is that each weight set is expanded into L weighted pseudorandom test patterns. These three levels of compression result in greatly reduced tester storage requirements and tester bandwidth requirements.

III. Determining Contents of LUT

The first step is to determine the weight sets that will be required to achieve the desired fault coverage. Any number of weighted pseudorandom patterns can be generated for each weight set with the scheme presented here. Many techniques for determining weight sets for a particular CUT have been proposed in the literature, e.g., [3], [19], and [23], etc. Any of these techniques can be used.

Given the weight sets, the next step is to determine the contents of the LUT. Fig. 2 illustrates how the weight sets are stored in the LUT and accessed during the testing. Let n denote the number of scan chains and m denote the number of bits in each scan chain. Thus the total number of scan elements is nm. As mentioned earlier, since a three-valued weight system is being used, two bits are required to store the weights for each scan element. The weights for the ith scan element of all the scan chains are stored in each location of the LUT so that they can all be read in the same clock cycle and used to transform the pseudorandom bits coming in from the PRPG. Thus, $(w1_1, w0_1), (w1_2, w0_2), \ldots, (w1_n, w0_n)$, denotes the weights

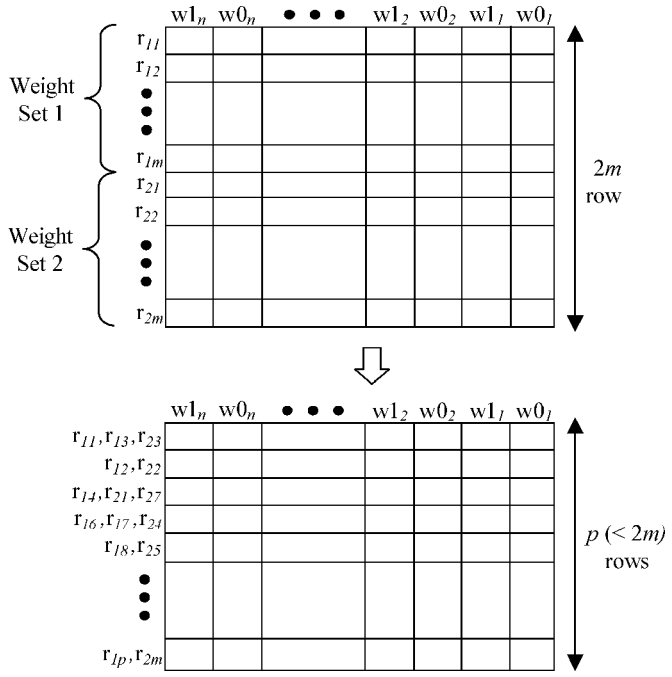

Fig. 2. Example storage of weight sets in LUT.

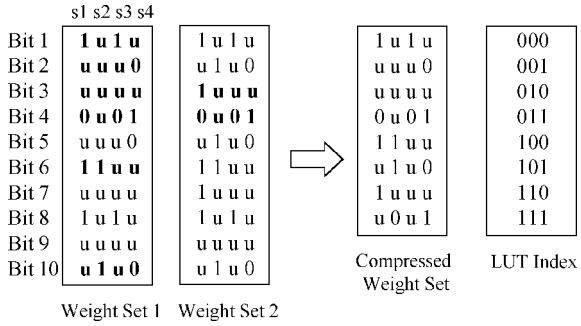

Fig. 3. Small example illustrating proposed approach.

represented by two bits for the i-th scan element in each of the n scan chains. The rows $r_{11}, r_{12}, \ldots, r_{1m}$ correspond to the first weight set. There is one row for each of the m bits of the scan chains. Rows $r_{21}, r_{22}, \ldots, r_{2m}$ correspond to the second weight set.

The LUT can be compressed to a great extent by merging identical rows as is illustrated in Fig. 2. This results in a lot of compression because many of the weight sets will have similar assignments in various rows in the LUT. Very often, the ith scan elements of the n scan chains will all be assigned $\mathbf{u}$. This common case reduces to a single row in the LUT. Reducing the size of the LUT has a two-fold advantage. Not only does it reduce hardware requirements, but it also reduces the size of the indices as fewer bits are now required to index the LUT.

Once the LUT has been constructed, then each weight set can be stored as a sequence of m indices on the tester where each index is $\lceil log_2 p \rceil$ bits wide where p is the total number of rows in the LUT.

A small example is shown in Fig. 3. The chip-under-test has 4 scan chains each of which is 10 bits long. Assume that 5 weighted pseudorandom patterns will be generated for each weight set for a total of 10 weighted pseudorandom patterns as shown in Fig. 4. In Fig. 4, the bold columns in the figure show the bits that are fixed because of the weight sets. The $\mathbf{u}$ bits take on a value of 1 or 0 randomly.

	scan chain 1	scan chain 2	scan chain 3	scan chain 4
weight 1	1uu0u1u1uu	uuuuu1uuu1	1uu0uuu1uu	u0u10uuuu0
weight 2	1u10u111uu	u1uu11uuu1	1uu0uuu1uu	u0u10uuuu0
weighted random pattern set 1	**1**0**1**0**11**0**1**00	0**11**00**1**0**1**0**1**	**1**0**1**0**11**0**1**00	**1**0**11**0**1**0**1**00
	11**00**1**11**10	0**111**0**11**0**11**	**1**0**1**0**00**1**1**0	**1**0**01**0**00**0**1**0
	11**00**0**1**0**111	1**0**1**0**1**1**0**00**1	**1**1**00**1**111**1	**1**0**01**0**0**1**0
	10**0**0**111**101	1**1**0**111**1**11**	**1**1**1**0**0**1**0**100	**0**0**0**1**0**1**1**00
	10**1**0**0**1**110	0**11**0**11**1**01**	**1**0**0**0**1**0**1**10	**0**0**11**0**11**000
weighted random pattern set 2	**1**0**1**0**111**100	**1**1**1**0**111**011	**1**0**0**0**0**0**1**110	**1**0**0**1**0**011u**0**
	10**1**0**111**101	0**1**0**1**1**1**0**101	**1**0**1**0**1**0**0**110	**1**0**0**1**0**0**0**0**1**0
	11**1**0**0**1**1**110	**1**1**0**1**111**001	**1**1**0**0**1**1**0**101	**1**0**0**1**0**1**1**000
	11**1**0**0**1**1**111	**1**1**0**1**1**1**0**001	**1**0**1**0**1**0**1**101	**0**0**11**0**0**1**1**00
	10**1**0**0**1**1**111	0**1**0**1**1**1**0**101	**1**1**0**0**0**1**1**100	**0**0**11**0**11**100

Fig. 4. Example of weighted pseudorandom patterns generated for 2 weight sets.

Fig. 3 shows how the weights will be stored in the LUT on the chip. Weights for a certain bit position for all the 4 scan chains will be stored in one location of the LUT. Thus every location of the LUT will be 8 bits wide (2 bits to represent each weight) and there will initially be 10 LUT locations for each weight set for the 10 bit positions in the scan chains. The scan chains are denoted by $s1$, $s2$, $s3$, and $s4$, and the bit positions are denoted by Bit i. Since the weight sets have a lot of similarities, they can be compressed to a great extent. In Fig. 3, the unique weight patterns are shown in bold. Thus, the duplicate patterns can be eliminated and only the unique patterns stored in the LUT. Hence, the LUT storage requirements can be reduced from 20 rows to only 8. Only a sequence of m LUT indices needs to be stored in the tester for each weight set. In this case, each index is only 3-bits wide since there are only 8 rows in the LUT.

IV. Hardware Implementation of LUT

The LUT can be implemented in hardware in a variety of ways. One simple way of implementing the LUT is by using a RAM. Generally there are a lot of RAM's already present in a design, so it may be possible to use one of them to serve as the LUT for this scheme. In this case, the contents of the LUT would be initially stored in the tester. Before the testing begins, the tester would initialize the RAM with the proper contents. The minimum size required for the RAM would depend on the number of rows in the LUT and the number of scan chains. Note that if the RAM is larger than necessary, this does not present a problem. In such a case, only a subset of the addressable locations in the RAM would be used, and only a subset of the data bits stored at each address would be used.

Another way of implementing the LUT would be to use a programmable logic array (PLA). Because most of the weight values are $\mathbf{u}$ (which could be encoded using one specified bit and one don't care), the number of rows in the PLA can be greatly minimized. A PLA provides a very compact and efficient implementation of the LUT.

One of the features of the proposed technique is that for each weight set, a sequence of m LUT indices are stored on the tester.

TABLE I
COMPARISON OF PROPOSED HYBRID BIST SCHEME WITH EXTERNAL TESTING

Circuit		Num. Weight Sets	Num. Scan Chains	RAM	PLA			Test Data		Test Data Compression Ratio
Name	Scan Elements			Size (Bytes)	Inputs	Outputs	Rows	Hybrid BIST	External Testing	
s13207	700	1	8	84	6	16	35	528	331800	628
			16	168	6	32	35	264		1257
			32	176	5	64	22	110		3016
s15850	611	4	8	198	7	16	74	2156	150306	70
			16	436	7	32	100	1092		138
			32	608	7	64	75	560		268
s38417	1664	10	8	642	9	16	255	18720	316160	17
			16	1580	9	32	358	9360		34
			32	2336	9	64	271	4680		67
			50	3100	8	100	227	2720		116
s38584	1464	3	8	114	6	16	44	3294	366000	111
			16	340	7	32	64	1932		189
			32	712	7	64	65	966		379
			50	887	7	100	53	630		581
composite	4439	10	8	772	9	16	295	49950	2104086	42
			16	2072	10	32	469	27800		76
			32	3088	9	64	361	12510		168
			50	3888	9	100	288	8010		263

If L weighted pseudorandom patterns are to be generated for each weight set, then the tester simply resends the sequence of indices for each weight set L times. Note that if the tester uses a slow clock, then the transfer of indices for all the L patterns might involve a long test time. One way to reduce the test time would be to also store the indices for the weight set in the RAM. Before pattern generation using a weight set, the tester initializes the RAM with a compressed weight set as well as its sequence of indices (hence, the RAM must be large enough to store the weight set and its sequence of indices). A faster on-chip clock can then be used to generate the L weighted patterns using that weight set and its sequence of indices. Some additional control circuitry would be required to cycle through the indices for the weight set (a counter can be used for this purpose). Once the required number of patterns has been generated using a weight set, another weight set and its corresponding set of indices can then be loaded from the tester into the RAM.

Note that the proposed method can also be used to implement traditional "stand-alone" BIST where no data comes from the tester. In this case, a ROM would be used. Both the contents of the LUT as well as the indices that would otherwise be stored on the tester can now be stored in a ROM. Some additional control circuitry would be needed to cycle through the sequence of indices for each weight set. Using a ROM can also help reduce the test application time since the weighted patterns can be applied at a speed higher than that of the tester clock.

V. EXPERIMENTAL RESULTS

Experiments were performed on the 4 largest ISCAS-89 circuits and a "composite" circuit which was formed by combining one instance of each of the 4 largest ISCAS-89 circuits. For each circuit, STUMPS architectures with different numbers of scan chains were constructed. First, 32 000 pseudorandom patterns were applied with the STUMPS architecture to detect the random pattern testable faults. Then the remaining random pattern resistant faults were targeted using weighted pseudorandom patterns based on the 3-valued weight system described in Section II. The weight sets were selected using the procedure described in [19] with 1000 weighted pseudorandom patterns being generated for each weight set. Table I shows the number of weight sets that were required for each circuit to achieve 100% coverage of detectable faults. For each circuit, Table I shows the total number of scan elements and the results for dividing them into different numbers of scan chains. In each case, the hardware requirements are shown for using either a RAM or a PLA to implement the LUT. The amount of test data that must be stored on the tester is shown for the proposed hybrid BIST approach which is equal to $(\text{num\_weight\_sets}) * (\text{inputs to PLA}) * (\lceil \text{scan elements/scan chains} \rceil)$ and is compared with the amount of test data required for conventional external testing where the uncompressed vectors are stored on the tester. The compression ratio for the test data storage requirements is shown. It is computed as follows:

$$\frac{(\text{test data for conventional external testing})}{(\text{test data for proposed hybrid BIST approach})}.$$

As can be seen, the tester storage requirements are reduced by orders of magnitude with the proposed hybrid BIST approach.

Table II shows a comparison of the proposed hybrid BIST approach versus an approach where BIST is used to detect the random pattern testable faults and then "top-up" deterministic test vectors are applied from the tester to detect the random pattern resistant faults. The "top-up" deterministic vectors were obtained by first applying 32 000 pseudorandom patterns using the STUMPS architecture, and then doing ATPG for the remaining undetected faults. The compression ratio for the test data storage requirements is shown. It is computed as:

$$\frac{(\text{test data for "top–up" test vectors})}{(\text{test data for proposed hybrid BIST approach})}.$$

TABLE II
COMPARISON OF PROPOSED HYBRID BIST SCHEME WITH BIST FOLLOWED BY TOP-UP TEST PATTERNS FROM TESTER

Circuit		Num. Scan Chains	Test Data		Test Data Compression Ratio
Name	Scan Elements		Hybrid BIST	BIST + Top-Up	
s13207	700	16	264	26600	101
s15850	611	16	1092	50102	46
s38417	1664	32	4680	158080	34
s38584	1464	32	966	61488	64
composite	4439	32	12510	421705	34

TABLE III
COMPARISON OF PROPOSED HYBRID BIST SCHEME WITH DETERMINISTIC BIST SCHEME IN [14]

Circuit		Hybrid BIST PLA				Deterministic BIST PLA				Area Overhead Compression Ratio
Name	Scan Elements	Inputs	Outputs	Rows	Area	Inputs	Outputs	Rows	Area	
s13207	700	6	16	35	0.4042	18	18	56	0.8698	2.1
s15850	611	7	16	74	0.7009	14	22	199	2.4907	3.5
s38417	1664	9	16	255	2.2061	34	68	509	16.4996	7.5
s38584	1464	6	16	44	0.4606	14	46	132	2.8289	6.1

As can be seen, the tester storage requirements are reduced by at least an order of magnitude in all cases with the proposed hybrid BIST approach based on weighted pseudorandom testing.

Table III shows a comparison of the area overhead for the proposed hybrid BIST approach compared with the best published results for deterministic BIST in [14] that provides the same fault coverage. This comparison assumes that a PLA implementation is used. It should be noted that for one or both of the techniques, a multilevel logic implementation may be more efficient than a PLA (this is in fact suggested in [14] for a STUMPS architecture). Also, note that the comparison only considers the area overhead of the PLA. It does not include the area overhead for the STUMPS architecture itself, however, the results in [14] indicate that the PLA area dominates the total area for BIST for these circuits. The compression ratio for the PLA area overhead requirements is shown. It is computed as follows:

$$\frac{(\text{PLA area for [14]})}{(\text{PLA area for proposed hybrid BIST approach})}.$$

As can be seen, the area overhead is greatly reduced with the proposed hybrid BIST approach. Note that the reduction becomes more pronounced for the larger circuits. For *s*38 417 and *s*38584, the hardware overhead is reduced by a factor of 7.5 and 6.1, respectively. Of course, it should be noted that the full BIST implementation in [14] does not have any tester storage requirements.

Fig. 5 shows how the hybrid BIST storage and the RAM size required for the proposed scheme vary when a different number of weighted pseudorandom patterns are generated for each weight set. The graph has been plotted with the number of weighted pseudorandom patterns per weight set varying from 32 to 32 000. As seen from the figure, as the number of patterns per weight set increases, both the hybrid BIST storage and the RAM size decreases. There is a tradeoff since increasing the number of patterns per weight set leads to an increase in the test application time. Thus, the proposed scheme provides a very easy way for the designer to trade off tester storage versus test time depending on the requirements.

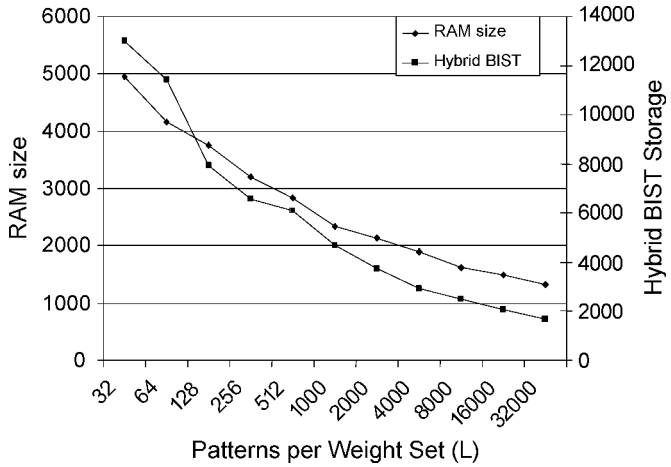

Fig. 5. Results for s38417 using different number of pseudorandom patterns for each weight set.

VI. CONCLUSION

The new test data compression scheme presented here combines BIST hardware with external data from the tester to provide a hybrid BIST solution. By using three levels of compression, the tester storage requirements are reduced by orders of magnitude compared to conventional external testing. Compared with using a deterministic BIST scheme to achieve the same fault coverage, it was shown that the area overhead on the chip can be significantly reduced. This new test data compression scheme provides another design point in addition to external testing or deterministic BIST that may be attractive in some test data compression scenarios. Note that in addition to the benefits of reducing tester storage and bandwidth requirements, the proposed approach also provides the benefits of weighted pseudorandom pattern testing in detecting nonmodeled defects.

REFERENCES

[1] P. H. Bardell and W. H. McAnney, "Self-testing of multichip logic modules," in *Proc. Int. Test Conf.*, 1982, pp. 200–204.

[2] F. Brglez, G. Gloster, and G. Kedem, "Hardware-based weighted random pattern generation for boundary scan," in *Proc. Int. Test Conf.*, 1989, pp. 264–274.

[3] M. Bershteyn, "Calculation of multiple sets of weights for weighted random testing," in *Proc. Int. Test Conf.*, 1993, pp. 1031–1040.

[4] K. Chakrabarty, B. T. Murray, and V. Iyengar, "Built-in test pattern generation for high-performance circuits using twisted-ring counters," in *Proc. VLSI Test Symp.*, 1999, pp. 22–27.

[5] D. Das and N. A. Touba, "Reducing test data volume using external/LBIST hybrid test patterns," in *Proc. Int. Test Conf.*, 2000, pp. 115–122.

[6] R. Dorsch and H.-J. Wunderlich, "Tailoring ATPG for embedded testing," in *Proc. Int. Test Conf.*, 2001, pp. 530–537.

[7] E. B. Eichelberger and E. Lindbloom, "Random-pattern coverage enhancement and diagnosis for LSSD logic self-test," *IBM J. Res. Develop.*, vol. 27, no. 3, pp. 265–272, May 1983.

[8] C. Fagot, P. Girard, and C. Landrault, "On using machine learning for logic BIST," in *Proc. Int. Test Conf.*, 1998, pp. 338–346.

[9] S. Hellebrand, S. Tarnick, J. Rajski, and B. Courtois, "Generation of vector patterns through reseeding of multiple-polynomial linear feedback shift registers," in *Proc. Int. Test Conf.*, 1992, pp. 120–129.

[10] S. Hellebrand, H.-G. Liang, and H.-J. Wunderlich, "A mixed mode BIST scheme based on reseeding of folding counters," in *Proc. Int. Test Conf.*, 2000, pp. 778–784.

[11] G. Hetherington, T. Fryars, N. Tamarapalli, M. Kassab, A. Hassan, and J. Rajski, "Logic BIST for large industrial designs: Real issues and case studies," in *Proc. Int. Test Conf.*, Sept. 1999, pp. 358–367.

[12] A. Jas, C. V. Krishna, and N. A. Touba, "Hybrid BIST based on weighted pseudorandom testing: A new test resource partitioning scheme," in *Proc. VLSI Test Symp.*, 2001, pp. 2–8.

[13] A. Khoche and J. Rivoir, "I/O bandwidth bottleneck for test: Is it real?," in *Proc. Int. Workshop Test Resource Partitioning*, Atlantic City, NJ, 2000.

[14] G. Kiefer and H.-J. Wunderlich, "Deterministic BIST with multiple scan chains," in *Proc. Int. Test Conf.*, 1998, pp. 1057–1064.

[15] G. Kiefer, H. Vranken, E. J. Marinissen, and H.-J. Wunderlich, "Application of deterministic logic BIST on industrial circuits," in *Proc. Int. Test Conf.*, 2000, pp. 105–114.

[16] L. Li and K. Chakrabarty, "Deterministic BIST based on a reconfigurable interconnection network," in *Proc. Int. Test Conf.*, 2003, pp. 460–469.

[17] H.-G. Liang, S. Hellebrand, and H.-J. Wunderlich, "Two-dimensional test data compression for scan-based deterministic BIST," in *Proc. Int. Test Conf.*, 2001, pp. 894–902.

[18] F. Muradali, V. K. Agarwal, and B. Nadeau-Dostie, "A new procedure for weighted random built-in self-test," in *Proc. Int. Test Conf.*, 1990, pp. 660–668.

[19] I. Pomeranz and S. M. Reddy, "3-weight pseudorandom test generation based on a deterministic test set for combinational and sequential circuits," *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 1050–1058, July 1993.

[20] M. Pressly, D. Das, and C. Hunter, "LBIST for PowerPC embedded core microprocessors: feasible or not?," in *Int. Workshop Microprocessor Test and Verification*, Atlantic City, NJ, 1999.

[21] A. P. Ströle and H.-J. Wunderlich, "TESTCHIP: A chip for weighted random pattern generation, evaluation, and test control," *IEEE J. Solid-State Circuits*, vol. 26, pp. 1056–1063, July 1991.

[22] N. A. Touba and E. J. McCluskey, "Altering a pseudorandom sequence of bits for scan-based BIST," in *Proc. Int. Test Conf.*, 1996, pp. 167–175.

[23] J. A. Waicukauski, E. Lindbloom, E. B. Eichelberger, and P. Forlenza, "A method for generating weighted random test patterns," *IBM J. Res. Develop.*, vol. 33, no. 2, pp. 149–161, Mar. 1989.

[24] H.-J. Wunderlich, "Multiple distributions for biased random test patterns," in *Proc. Int. Test Conf.*, 1988, pp. 236–244.

Abhijit Jas (S'94–M'01) received the B.E. degree in computer science and engineering from Jadavpur University, Calcutta, India in 1996. He secured the first rank among all graduating students from the college of engineering. He received the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Texas at Austin in 1999 and 2001, respectively.

He is currently with Intel Corporation, Austin, TX. His research interests are in VLSI testing, formal verification and related CAD algorithms.

Dr. Jas was a co-recipient of the 2001 Best Paper Award at the VLSI Test Symposium. He is a technical program committee member of the International Conference on VLSI Design 2005 and the International Test Synthesis Workshop 2005..

C. V. Krishna (S'01–M'04) received the B.E. degree in computer science and engineering from Jadavpur University, Calcutta, India, in 1999, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Texas at Austin in 2001 and 2004, respectively.

He currently works as a member of the consulting staff, Test Design Automation group at Cadence Design Systems, Inc, Endicott, NY. His research interests include test compression, design automation, design for testability, and built-in self-test.

Dr. Krishna was a co-recipient of the 2001 Best Paper Award at the VLSI Test Symposium. He was also a recipient of the IBM Ph.D. Fellowship for the years 2002–2004.

Nur A. Touba (S'88–M'96) received the B.S. degree in electrical engineering from the University of Minnesota, Twin Cities, MN, in 1990 and the M.S. and Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, in 1991 and 1996, respectively.

He is currently an Associate Professor at the University of Texas at Austin.

Dr. Touba received the National Science Foundation Early Faculty CAREER Award in 1997 and was a co-recipient of the 2001 Best Paper Award at the VLSI Test Symposium. He serves on the technical program committees of the International Test Conference, International Conference of Computer Design, Design Automation and Test in Europe Conference, International On-Line Test Symposium, European Test Symposium, Microprocessor Test and Verification Workshop, and the International Test Synthesis Workshop. He is on the editorial board of the *Journal of Low-Power Electronics* (JOLPE).