X-Tolerant Test Response Compaction

Subhasish Mitra
Intel

Michael Mitzenmacher
Harvard University

Steven S. Lumetta
University of Illinois at Urbana-Champaign

Nishant Patil
Intel

Editor's note:
Larger, denser designs lead to more defects; higher quality requirements and new test methods lead to an explosion in test data volume. Test compression techniques attempt to do more testing with fewer bits. This article summarizes one such method, X-compact, which addresses how unknowns, the bane of compression and logic BIST techniques, are eliminated.
—Scott Davidson, Sun Microsystems

DIGITAL CIRCUIT TESTING involves applying test patterns and observing the circuit's responses to the applied patterns. The tester compares the observed response to a test pattern with the expected response and declares a chip defective upon mismatch. Test engineers usually obtain the expected response through fault-free simulation of the circuit for the corresponding test pattern. Unfortunately, fault-free simulation cannot always determine the expected response to be 0 or 1. In that case, the expected response is an unknown, or X. Of course, an actual defect-free or defective chip will produce 0 or 1; however, because the simulated expected response is X, we cannot compare the actual chip's response with a golden reference. Hence, this test response is ignored during testing. Table 1 explains testing in the presence of Xs.

Table 2 summarizes the major sources of Xs in today's designs. Proper DFT techniques must be employed to minimize Xs. However, it is impractical to eliminate all X sources due to timing constraints, area overhead, simulation engine inefficiencies (such as 0-delay simulation), and inaccuracies in modeling the behaviors of certain memory, custom logic, and analog circuit blocks (also called black boxes). Some of these problems become visible very late in design or after IC manufacture, when it is difficult or impossible to insert additional DFT structures. Table 3 shows four industrial ASIC designs with their corresponding X densities—the percentage of response bits whose expected values are Xs.

For designs with traditional scan DFT, handling Xs in test responses is simple—the tester ignores expected test response bits with Xs. However, the presence of Xs poses a major challenge for designs using test compression[1] and BIST. For example, consider the use of classical signature analyzers, such as multiple-input signature registers (MISRs), for response compaction. Figure 1 (p. 568) shows an example. The outputs of four scan chains are connected to the MISR inputs. The initial MISR state is 0000. The figure shows the MISR states during the first four clock cycles. Xs appearing at scan chain outputs corrupt the MISR contents. After four clock cycles, the expected MISR signature obtained from fault-free simulation consists entirely of Xs. The tester ignores signature bits whose expected values are Xs during comparison of the expected signature with the actual signature. In this example, no comparison can be made.

Any response compaction technique must be able to detect a defective chip in the presence of residual Xs that neither DFT nor accurate modeling can eliminate. This article presents an overview of response compactor design techniques that we have developed.[2-4] These response compactors tolerate the presence of Xs with practically no impact on test quality. Depending on the number of Xs in a design, they can reduce test response data volume by up to three orders of magnitude, as supported by data from actual designs. No assumptions about defect behaviors are necessary. For example, it is not necessary to assume that all defects behave as sin-

0740-7475/05/$20.00 © 2005 IEEE Copublished by the IEEE CS and the IEEE CASS

gle stuck-at faults. Moreover, engineering change orders or test pattern changes after tapeout do not affect the response compaction hardware.

Two major types of techniques for test response compaction in the presence of Xs have appeared in the literature: fixing the Xs as 0s and 1s before they enter the compaction hardware,[5-9] and postprocessing response data to determine whether the tested IC is defective.[10] These approaches require significant tester support or knowledge of the exact positions of Xs in test responses, or both. Some of these techniques mask the outputs of entire scan chains, significantly affecting test quality in terms of defect coverage (not necessarily fault coverage). The X tolerance approach we describe here comes from the original X-compact idea (described in the next section) and imposes no such requirements, makes no assumptions about defect behaviors, and is ideal for BIST and test compression. Previous publications present a more detailed discussion of the benefits of these X-tolerant response compactors over other response compaction techniques.[3,4] Of course, other techniques and tester features can complement our technique.

X-tolerant response compaction theory

Suppose that n test response bits are compacted into m bits, $m < n$. We represent this situation with a matrix composed of n rows and m columns. Each row represents a bit in the uncompacted test response, and each column represents a bit in the compacted test response. The matrix entry corresponding to row i and column j is 1, if and only if bit j of the compacted test response depends on bit i of the uncompacted test response. Otherwise, the entry is 0. This matrix is called an X-compact matrix. We obtain a bit in the compacted response by calculating the XOR of bits in the uncompacted test response that have 1s in the compacted response bit's column.

Table 1. Testing in the presence of Xs.

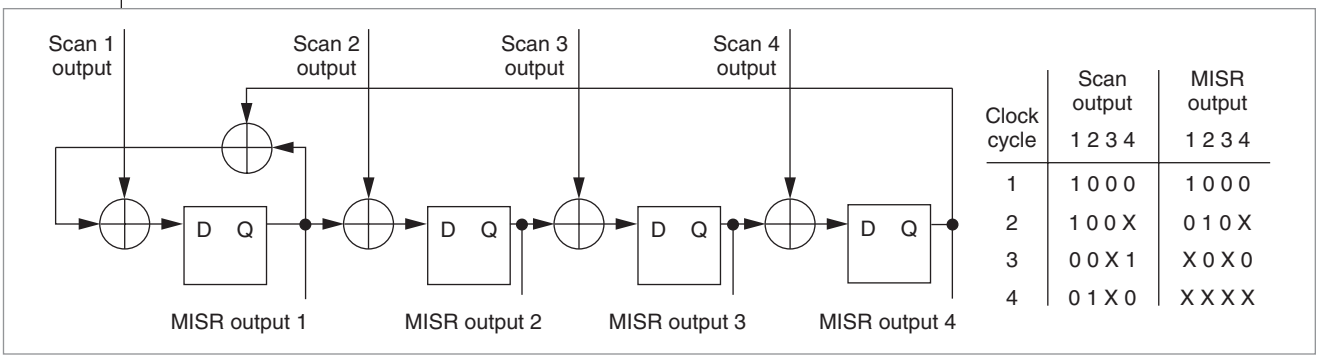
	Responses		
Test pattern	Expected (simulated)	Actual (silicon)	Test result
Pattern 1: The chip passes this test pattern since all response bits are either ignored or pass.	0	0	Pass
	1	1	Pass
	X	1	Ignore
	1	1	Pass
	X	0	Ignore
Pattern 2: The chip fails this pattern since there is one response bit that is not ignored and does not match the expected value.	0	0	Pass
	1	0	Fail
	X	0	Ignore
	1	1	Pass
	X	1	Ignore

Table 2. Major sources of Xs.

X sources	Explanation	Fixed by DFT?
Uninitialized bistables	Initial state unknown	Yes, with initialization circuitry
Bus contention, floating buses	Multiple bus drivers enabled or all bus drivers disabled	Yes, with ATPG constraints or test points
Black boxes	Structures for which creating accurate digital-logic models is difficult: embedded memory and mixed-signal blocks, custom circuitry	Sometimes, with isolation collars; occasionally difficult due to performance constraints
Multicycle paths	Circuit paths with delays greater than 1 clock cycle; Xs produced during delay testing	Very difficult
Multiclock-domain interactions	Fault-free simulation cannot accurately predict logic values	Very difficult
Postsilicon Xs	Mismatch between simulated and actual responses of known-good chips; mainly due to inaccurate simulation models	Impossible

Table 3. X densities in industrial ASIC designs.

Design	X management	X density (%)	Design type
1	Good	0.004	Network switch
2	Fair	0.02	Microprocessor chipset
3	Poor	0.15	Microprocessor chipset
4	Poor	0.3	Microprocessor chipset



Clock cycle	Scan output 1 2 3 4	MISR output 1 2 3 4
1	1 0 0 0	1 0 0 0
2	1 0 0 X	0 1 0 X
3	0 0 X 1	X 0 X 0
4	0 1 X 0	X X X X

Figure 1. Multiple-input signature register (MISR) example. Initial MISR state is 0000.

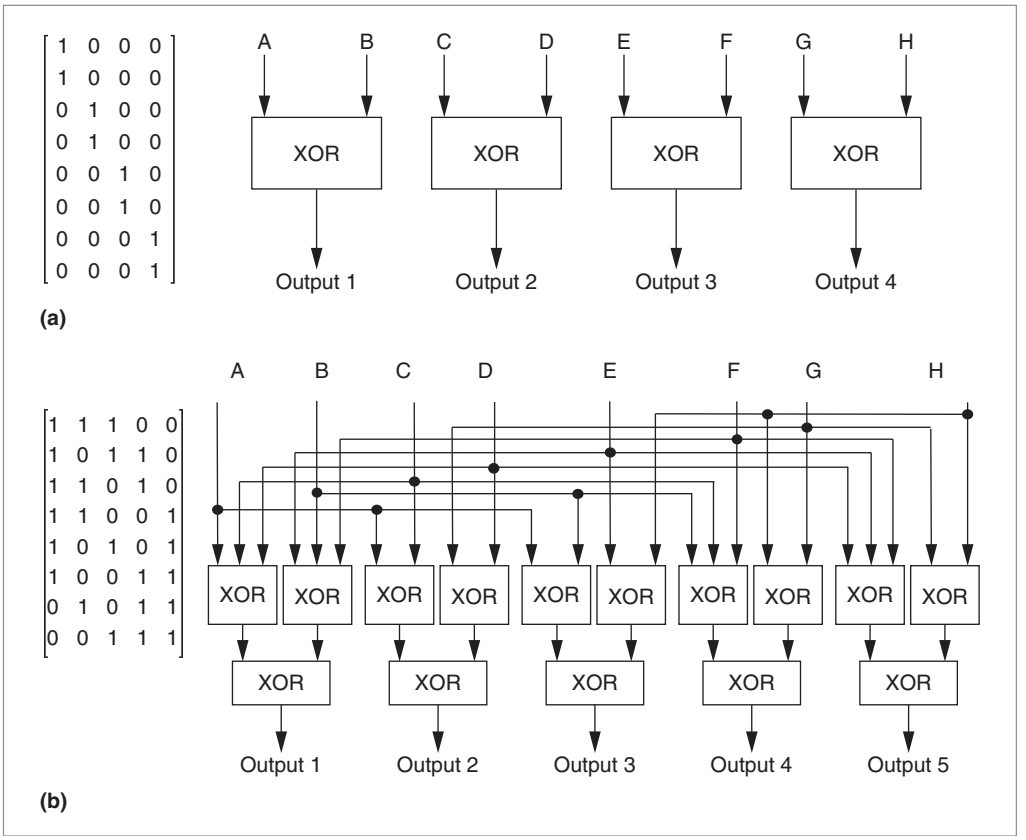


Figure 2. X-compact matrices and corresponding response compaction circuits: X-intolerant compactor (a); X-tolerant compactor (b).

The fundamental problem in designing response compactors is determining how to fill the X-compact matrix entries with 1s and 0s. For example, Figure 2 shows two compaction matrices and the corresponding response compaction circuits. Suppose that an error is captured in response bit A, and response bit B is X. The compactor in Figure 2a cannot report this error because the expected value of output 1 is X. However, the compactor in Figure 2b is able to report the error because output 2 depends on A and not B. Thus, the compactor in Figure 2b tolerates the X present at input B.

Deterministic coding techniques for designing X-compact matrices have been published in earlier works[2,3,11,12] and are not repeated here. Since the introduction of X-compact in 2002,[2] other researchers have proposed several techniques derived from it. Deterministic coding techniques are mainly useful during combinational compaction for test compression applications in which the uncompressed response bits represent scan chain outputs compacted by combinational logic circuitry.

We can also use stochastic coding techniques for designing X-compact matrices.[4] Stochastic coding is used mainly for signature analysis because of its implementation simplicity, but it can be used for combinational compaction as well. In the stochastic coding approach, we fill the X-compact matrix entries with 1s and 0s at random, so that the probability of assigning 1 to an entry is p, and the probability of assigning 0 to an entry is $1 - p$. The expected number of 1s in a row, or the row's expected weight, is then $m \times p$. A bit in the compacted response is a non-X bit if that bit's logic value in the expected compacted

Table 4. Response compaction for 1 million uncompacted test response bits for Table 3 designs.

Design	No. of Xs	No. of compacted bits	Compaction ratio	Probability of errors masked by Xs, $t = 1$	$t = 3$	$t = 5$
1	40	2,000	500	1.2×10^{-8}	$< 10^{-22}$	$< 10^{-36}$
		1,000	1,000	10^{-4}	$< 10^{-11}$	$< 10^{-18}$
		500	2,000	10^{-2}	1.9×10^{-6}	$< 10^{-9}$
2	200	6,500	153	6.5×10^{-6}	$< 10^{-15}$	$< 10^{-25}$
		2,400	420	10^{-2}	2×10^{-6}	$< 10^{-9}$
		1,250	800	10^{-1}	10^{-3}	1.2×10^{-5}
3	1,500	38,000	26	8×10^{-5}	$< 10^{-12}$	$< 10^{-20}$
		16,000	63	2×10^{-2}	$< 10^{-5}$	$< 10^{-8}$
		9,000	110	1.1×10^{-1}	1.3×10^{-3}	1.6×10^{-5}
4	3,000	76,000	13	8×10^{-5}	$< 10^{-12}$	$< 10^{-20}$
		30,000	34	2.5×10^{-2}	1.6×10^{-5}	$< 10^{-8}$
		18,000	56	1.1×10^{-1}	1.3×10^{-3}	1.6×10^{-5}

response is not X. An error in the uncompacted test response is masked if none of the non-X bits in the compacted response are erroneous.

Next, we ask, What is the probability that an error in an uncompacted test response bit gets masked when there are Xs in k other uncompacted test response bits? The probability is

$$[1 - p \times (1 - p)^k]^m \quad (1)$$

We derive Expression 1 from the fact that if an entry in the row corresponding to the erroneous bit is 0, the error does not affect the corresponding bit in the compacted response. If an entry in this row is 1, the error affects the corresponding compacted response bit (column) if and only if none of the k rows producing Xs have 1s in that column; otherwise, the error is masked. For a defective part to be detected, an error must affect at least one of the non-X bits in the compacted response. Expression 1 reaches its minimum when the following relationship is satisfied:[4] $p = 1/(k + 1)$.

If there are t error bits and k Xs in the uncompacted test response, the probability that no non-X bit of the compacted response is erroneous is

$$\left[1 - \sum_{1 \le i \le t,\, i \text{ is odd}} \binom{t}{i} p^i (1-p)^{t-i+k}\right]^m \quad (2)$$

The derivation of Expression 2 is similar to that of Expression 1. The only difference is that a bit in the compacted response is erroneous if and only if it is derived from an odd number of error bits in the uncompacted response, and it doesn't depend on any bit producing X in the uncompacted response. An even number of error bits affecting a given output is undetectable—a phenomenon called error cancellation. From practical experience, a pessimistic value of t is between 3 and 5.

For the Table 3 designs, Table 4 shows the corresponding probability that errors in the test response become masked because of the presence of Xs. Here, we use Expressions 1 and 2 with the value of p for which Expression 1 results in the minimum error-masking probability. Table 4 provides no insight into the response compactor's test architecture and actual logic design, which we detail later.

X-tolerant combinational compaction

X-tolerant combinational compactors (X-compactors) are combinational circuits that compact scan chain outputs at every scan cycle. An application of the scan clock to perform a shift operation is referred to as a scan cycle. The test equipment must collect the X-compactor outputs at every scan cycle and compare them with the expected response. For example, as Figure 3 shows, a design might contain 1,000 scan chains, which the X-compactor can reduce to only 20 outputs. Several industrial designs, including microprocessors, network processors, chipsets, and network storage controllers, are currently using such combinational X-compactors. X-compactor designs can guarantee high test quality (that is, reduced probability of error masking due to Xs) and can be optimized for reduced routing congestion.

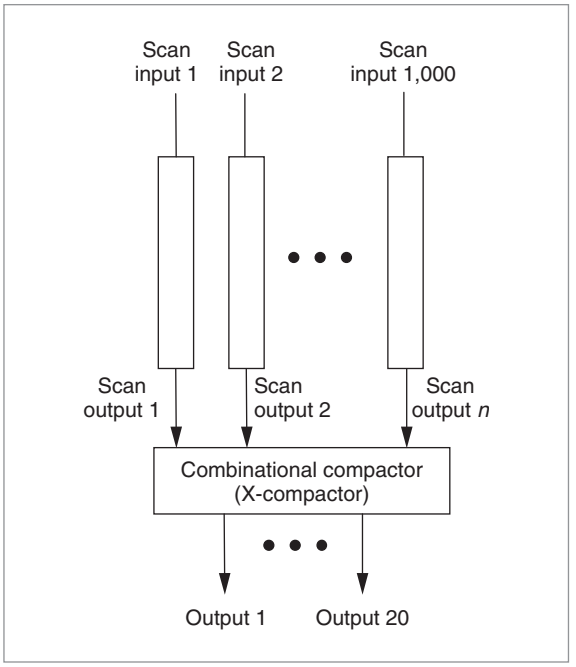


Figure 3. Scan architecture with X-compactor.

To design combinational compactors, we analyze the use of constant-weight X-compact matrices, in which every row has an equal and odd number of 1s. For an X-compactor with n inputs and m outputs, the X-compact matrix has n rows and m columns. Suppose that each row of the X-compact matrix has r 1s. The probability that an error at an input of the X-compactor will be masked when k other scan chains produce Xs is less than or equal to the following expression (for $k \times r \leq m$):

$$\frac{\binom{k \times r}{r}}{\binom{m}{r}} \tag{3}$$

Expression 3 follows from the fact that an error produced by row i will be masked if and only if at least one of the k rows corresponding to the rows producing Xs have 1s in the columns in which row i has 1s. The maximum number of columns masked by the k rows producing Xs is $k \times r$, giving

$$\binom{k \times r}{r}$$

combinations of r 1s that are masked. If the row corresponding to the error is given by one of these combinations, the error is masked. The total number of combinations is

$$\binom{m}{r},$$

giving Expression 3 as a bound on the probability of masking. All rows of the matrix are unique. Hence, an error produced by a scan chain will never be masked when another scan chain produces an X (that is, $k = 1$).[3] Table 5 summarizes the characteristics of X-compact matrices generated with this technique.

Table 6 shows the distribution of the number of Xs over scan cycles for the single-stuck-at test patterns of an ASIC design with 800 scan chains. Suppose that we implement the 800-to-25 X-compactor in Table 5 for this

Table 5. Characteristics of X-compact matrices for combinational compaction (where fan-out is the number of 1s in each row of the X-compact matrix).

	No. of	Compaction		Probability of an incorrect X-compactor input masked by k Xs		
No. of inputs	outputs	ratio	Fan-out	$k = 1$	$k = 2$	$k = 3$
500	25	20	5	0	2.6×10^{-3}	1.9×10^{-2}
	50	10	11	0	4.2×10^{-6}	3.3×10^{-4}
800	25	32	5	0	2.6×10^{-3}	1.9×10^{-2}
	50	16	11	0	4.8×10^{-6}	3.4×10^{-4}
1,000	25	40	5	0	2.7×10^{-3}	1.9×10^{-2}
	50	20	11	0	5.1×10^{-6}	3.5×10^{-4}
1,500	25	60	5	0	2.7×10^{-3}	1.9×10^{-2}
	50	30	11	0	4.7×10^{-6}	3.4×10^{-4}
2,000	25	80	5	0	2.7×10^{-3}	1.9×10^{-2}
	50	40	11	0	4.7×10^{-6}	3.4×10^{-4}

design. Because 33% of the scan cycles have only one X, 10% have only two Xs, and 2% have only three Xs, the following formula gives the overall probability that an incorrect scan chain output will be masked by Xs in a single scan cycle:

$$0.33p_{1x} + 0.10p_{2x} + 0.02p_{3x} = 6.4 \times 10^{-4}$$

where p_{1x} is the probability that an incorrect scan chain output is masked by one X; p_{2x}, the probability that an incorrect scan chain output is masked by two Xs; and so on.

As actual test data indicates, assuming that a defective chip produces incorrect outputs during at least two scan cycles is pessimistic and realistic. In that case, the overall probability that the incorrect outputs will be masked and the defective chip will not be detected is $(6.4 \times 10^{-4})^2 = 4 \times 10^{-7}$, a very small number.

This example demonstrates that X-compactors can obtain massive reduction in test data volume and test time with practically no impact on overall test quality. They are very effective for diagnosis and yield-learning, as demonstrated using actual silicon data.[13]

Table 6. Distribution of Xs over scan cycles for single-stuck-at test patterns of an ASIC design with 800 scan chains.

No. of Xs	Percentage of scan cycles
0	55
1	33
2	10
3	2

X-tolerant signature analysis

For simplicity, we first describe serial signature analysis, in which test response bits from a single source (such as a single scan chain) are compacted into a signature. We call this structure an X-SISR, which stands for X-tolerant single-input signature register or an X-tolerant *serial*-input signature register. Next, we describe the design of an X-tolerant multiple-input signature register or X-MISR.

Suppose that a source is serially producing n bits of test response data and that the signature consists of m bits. Here n and m are design parameters. As explained earlier, the stochastic coding approach fills the X-compact matrix (with n rows and m columns) with 1s and 0s such that the probability of assigning 1 is p, a parameter based on the number of Xs that must be tolerated.

How can we implement such a coding scheme in hardware? Given parameter p, a weighted random-pattern generator can generate 1s with probability p. The testing literature has extensively covered the design of logic structures that generate weighted random patterns.[14]

Figure 4 shows an X-SISR design. At each clock cycle, a test response bit is ANDed with the weighted random pattern generator outputs. Depending on where the 1s appeared, the XOR gate and feedback loop ensure that the corresponding signature bits will depend on this particular test response bit. The reader can prove the equivalence between this structure and the X-compact matrix structure described earlier.

It might seem that by ANDing on the response bits, we lose the capability of observing an error at a particular signature bit. However, because there are multiple signature bits, the probability that an error in the uncom-

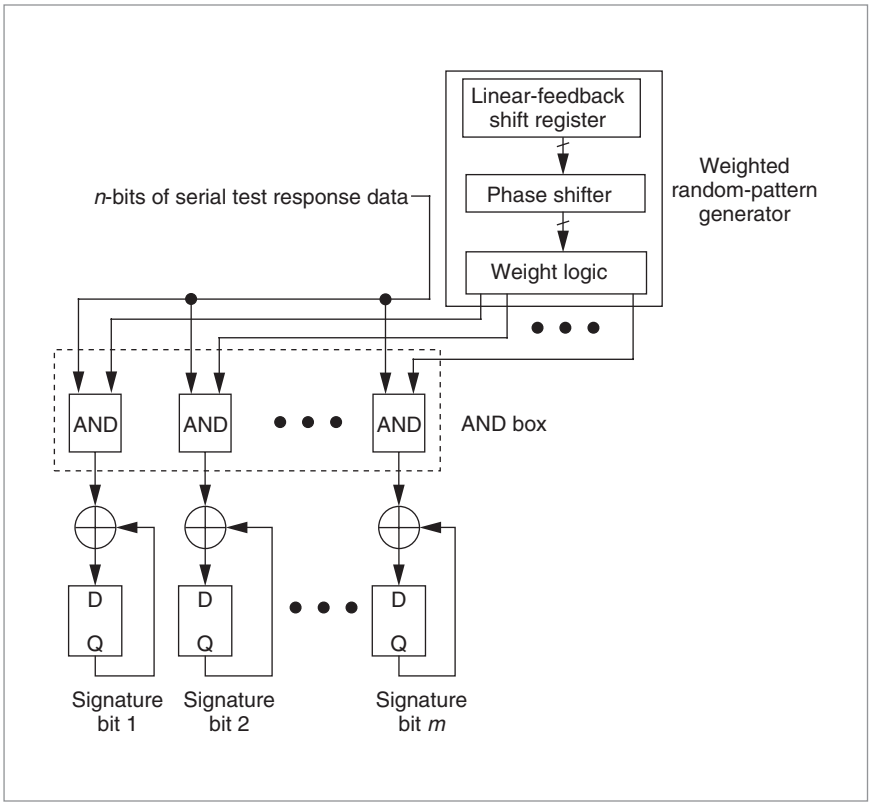


Figure 4. X-SISR design.

pacted response will not be registered in any of the signature bits is extremely small, even in the presence of Xs. This follows from Expressions 1 and 2 and Table 4, because the X-compact matrix structure using stochastic coding and the Figure 4 design are equivalent.

Depending on the value of m, the fan-out in Figure 4 might cause concern. For example, suppose that the value of m is 20, meaning that to implement the design in Figure 4, we need a fan-out of 20. Generally, scan chain shifting and scan chain output compaction occur at slow speed. Hence, the large fan-out might not be a problem because signal paths to the response compaction circuitry aren't timing critical. However, simple pipelining minimizes the fan-out problem. Using multiple weighted random-pattern generator circuits can reduce the fan-out associated with weighted random generators.

Figure 5 shows how we convert the X-SISR design into an X-MISR design. Suppose that there are s parallel test response inputs (which can be scan chains). For each parallel test response input, the X-MISR has m AND gates in a structure called an AND box (Figure 4 shows the components of an AND box).

X-tolerant signature analysis offers several benefits:

- It's simple to design: No complex controller is necessary. However, the design is more complex than traditional MISRs.
- It can tolerate several thousand Xs with practically no impact on test quality, unlike traditional MISRs.
- It requires very high-level design information on the expected number of Xs but no detailed information on the actual positions of Xs.
- It's nonintrusive to the design because no test points are inserted.
- It's friendly to engineering changes and test pattern changes after the chip is manufactured.
- It enables targeted improvement of test quality, even after manufacture, with two techniques: varying the intervals between intermediate signatures, and generating a huge number of independent X-compact matrices out of the signature-analysis hardware. Multiple test runs with the same test patterns but independently generated X-compactors are possible. Multiplying the probabilities of not detecting a defective chip in each test run significantly reduces the probability that a defective chip is not detected.
- Combining it with techniques such as I-compact[10] can further reduce the probability that Xs mask errors, at the expense of extra processing on the tester.

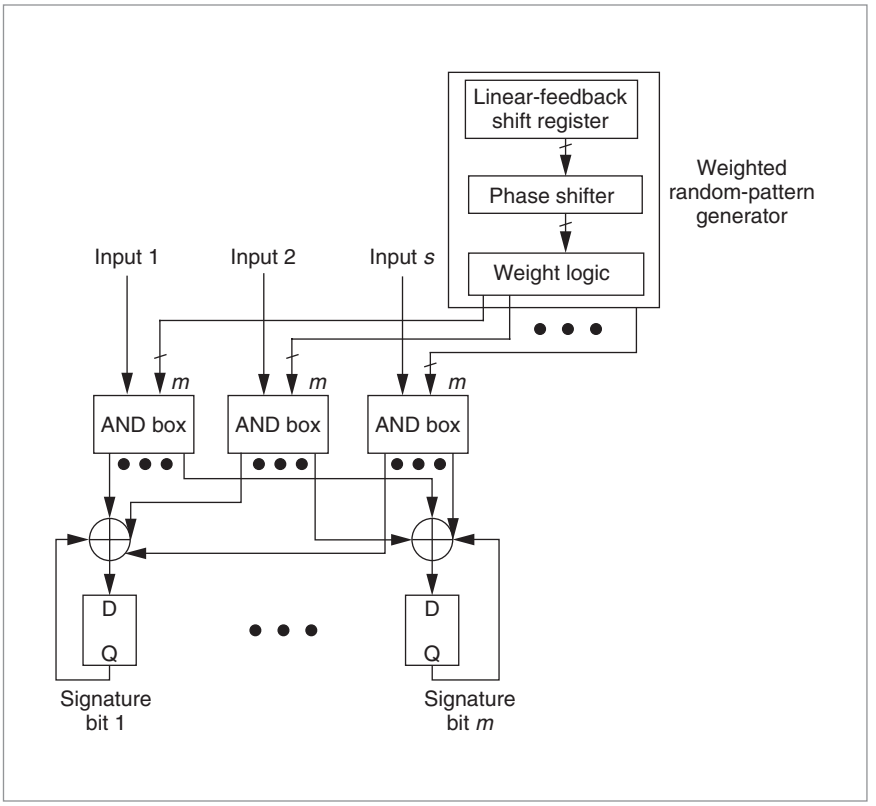


Figure 5. X-MISR design.

The routing overhead of X-tolerant signature analyzers must be suitably managed. Figure 6 shows one way to do this: using local X-MISRs and intermediate signatures. For example, suppose that design 3 in Table 4 has 1,000 scan chains, each 1,000 bits long. We break the design into 250 clusters, each containing four scan chains. For each cluster, we design a 20-bit X-MISR. Because only four scan chains drive each X-MISR, each X-MISR bit requires only four two-input XOR gates. We run the X-MISR for 500 cycles—this corresponds to 2,000 uncompacted response bits per cluster, which is compactable into 20 bits of signature with a high error detection probability (similar to Table 4, design 3, assuming a uniform distribution of Xs). Every 500 cycles, the 20-bit intermediate signatures from each cluster are transferred to a shadow register. The shadow registers of all clusters are configured into 10 scan chains. It takes 500 cycles to scan the intermediate signature bits from all clusters out of the

scan chains before another intermediate signature from the clusters is loaded into the shadow register.

This approach leads to an X-tolerant signature analysis technique that tolerates thousands of Xs but outputs only 10 response bits for observation on the tester in every scan cycle, despite the 1,000 scan chains. Thus, the test data volume and test time reduction obtained is two orders of magnitude for design 3 with its poorly managed Xs.

Applying X-MISR for BIST in the field involves the use of an on-chip memory (for example, cache memory or other storage areas available for BIST) that can be preloaded with expected signatures from an off-chip store, allowing the chip itself to compute a pass/fail result with no need to output any test data. As with other types of test data, the preloaded data can also be compressed for storage in a small ROM or flash memory.

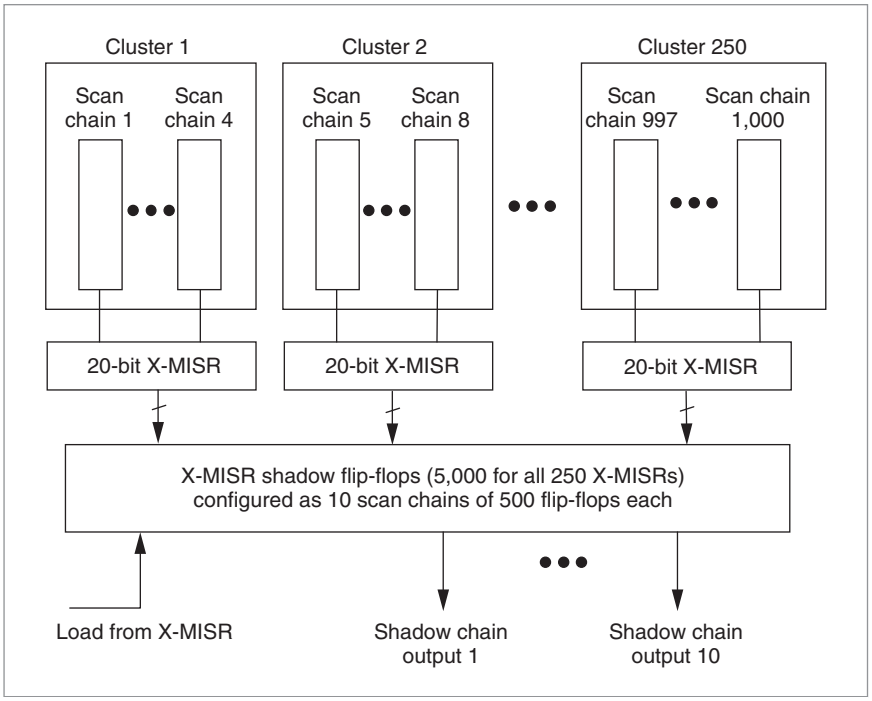


Figure 6. Scan architecture with local X-MISRs and intermediate signatures.

DFT ENGINEERS must spend serious effort to minimize Xs in future designs. It will be impossible to eliminate all Xs. X-tolerant response compactors are necessary for tolerating residual Xs to enable massive compaction with practically no impact on test quality. X-compactors are mainly useful for test compression purposes and provide up to 80 times the test response compaction of traditional scan. X-tolerant signature analyzers extend the X-compact concept to incorporate time compaction, thereby tolerating thousands of Xs and reducing test response data volume by 50 to 2,000 times relative to traditional scan. These signature analyzers are extremely beneficial for BIST because Xs can easily corrupt traditional MISR-based BIST signature analyzers. X-tolerant response compactors also enable efficient diagnosis essential to fast yield-learning, which will be the topic of a future paper. ■

■ References

1. "ITC 2002 Roundtable: Test Data Compression," *IEEE Design & Test*, vol. 20, no. 2, Mar.-Apr. 2003, pp. 76-87.
2. S. Mitra and K.S. Kim, "X-Compact: An Efficient Response Compaction Technique for Test Cost Reduction," *Proc. Int'l Test Conf.* (ITC 02), IEEE Press, 2002, pp. 311-320.
3. S. Mitra and K.S. Kim, "X-Compact: An Efficient Response Compaction Technique," *IEEE Trans. Computer-Aided Design*, vol. 23, no. 3, Mar. 2004, pp. 421-432.
4. S. Mitra, S.S. Lumetta, and M. Mitzenmacher, "X-Tolerant Signature Analysis," *Proc. Int'l Test Conf.* (ITC 04), IEEE Press, 2004, pp. 432-441.
5. C. Barnhart et al., "OPMISR: The Foundation for Compressed ATPG Vectors," *Proc. Int'l Test Conf.*, IEEE Press, 2001, pp. 748-757.
6. J. Rajski et al., "Embedded Deterministic Test for Low Cost Manufacturing Test," *Proc. Int'l Test Conf.* (ITC 02), IEEE Press, 2002, pp. 301-310.
7. P. Wohl et al., "Efficient Compression and Application of Deterministic Patterns in a Logic BIST Architecture," *Proc. 40th Design Automation Conf.* (DAC 03), ACM Press, 2003, pp. 566-569.
8. V. Chickermane, B. Foutz, and B. Keller, "Channel Masking Synthesis for Efficient On-Chip Test Compression," *Proc. Int'l Test Conf.* (ITC 04), IEEE Press, 2004, pp. 452-461.
9. E.H. Volkerink and S. Mitra, "Response Compaction with Any Number of Unknowns Using a New LFSR Architecture," *Proc. 42nd Design Automation Conf.* (DAC 05), ACM Press, 2005, pp. 117-122.
10. J.H. Patel, S.S. Lumetta, and S.M. Reddy, "Application of Saluja-Karpovsky Compactors to Test Responses

with Many Unknowns," *Proc. 21st VLSI Test Symp.* (VTS 03), IEEE Press, 2003, pp. 107-112.
11. S.S. Lumetta and S. Mitra, "X-Codes: Error Control with Unknowable Inputs," *Proc. Int'l Symp. Information Theory*, IEEE Press, 2003, p. 102.
12. S.S. Lumetta and S. Mitra, *X-Codes*: *Theory and Applications of Unknowable Inputs*, tech. report CRHC-03-08 (also UILU-ENG-03-2217), Center for Reliable and High-Performance Computing, Univ. of Illinois at Urbana-Champaign, 2003.
13. Z. Stanojevic et al., "Enabling Yield Analysis with X-Compact," to be published in *Proc. Int'l Test Conf.* (ITC 05), IEEE Press, 2005.
14. E.B. Eichelberger et al., *Structured Logic Testing*, Prentice Hall, 1991.

Subhasish Mitra is a principal engineer at Intel, a consulting assistant professor in the Electrical Engineering Department of Stanford University, and the associate director of the Stanford Center for Reliable Computing. He will join the Electrical Engineering Department of Stanford University as a full-time faculty member starting January 2006. His research interests include robust system design, VLSI design and test, VLSI CAD, and computer architecture. Mitra has a PhD in electrical engineering from Stanford University. He received the 2005 IEEE Circuits and Systems Society Donald O. Pederson Award, a Recognition Award from the Intel Mobility Group "for a breakthrough soft error protection technology," and the 2004 Intel Achievement Award, Intel's highest honor, "for the development and deployment of a breakthrough test compression technology."

Steven S. Lumetta is an associate professor of electrical and computer engineering and a research associate professor in the Coordinated Science Laboratory at the University of Illinois at Urbana-Champaign. His research interests include optical networking, computer systems, and digital-system testing. He has an AB in physics, an MS in computer science, and a PhD in computer science, all from the University of California, Berkeley.

Michael Mitzenmacher is a professor of computer science in the Division of Engineering and Applied Sciences at Harvard University. His research interests include Internet algorithms, hashing, load balancing, erasure codes, error-correcting codes, compression, bin packing, and power laws. Mitzenmacher has a BS in mathematics and computer science from Harvard and a PhD in computer science from the University of California, Berkeley.

Nishant Patil is pursuing a PhD in the Electrical Engineering Department of Stanford University; he was an intern at Intel when he did this work. His research interests include IC testing and computer architecture. Patil has a BS in electrical and computer engineering from Carnegie Mellon University.

■ Direct questions and comments about this article to Subhasish Mitra; smitra@crc.stanford.edu.