Unknown-Tolerance Analysis and Test-Quality Control for Test Response Compaction using Space Compactors

Mango C.-T. Chao, Kwang-Ting Cheng,

Dept. of ECE, UC Santa Barbara
Santa Barbara, CA, USA
{mango,timcheng}@ece.ucsb.edu

Seongmoon Wang, Srimat Chakradhar,
Wen-Long Wei

NEC Labs. America
Princeton, NJ, USA
{swang, chak, wwei}@nec-labs.com

ABSTRACT

For a space compactor, degradation of fault detection capability caused by the masking effects from unknown values is much more serious than that caused by error masking (i.e. aliasing). In this paper, we first propose a mathematical framework to estimate the percentage of observable responses under unknown-induced masking for a space compactor. We further develop a prediction scheme which can correlate the percentage of observable responses with the modeled-fault coverage and with a n-detection metric for a given test set. As a result, the quality of a space compactor can be measured directly based on its test quality, instead of based on indirect metrics such as the number of tolerated unknowns or the aliasing probability. With the prediction scheme above, we propose a construction flow for space compactors to achieve the desired level of test quality while maximizing the compaction ratio.

Categories and Subject Descriptors

B.8.1 [**Hardware**]: Reliability, Testing, and Fault-Tolerance

General Terms

Design

Keywords

Test response compaction, design for test

1. INTRODUCTION

The greatest barrier to effective test-response compaction is the presence of unknown values (or unknowns) among scan-out responses. If no unknown value exists, a *time compactor*, such as MISRs (Multiple Input Signature Registers), can compress an infinitely long output sequence into a fixed-length signature and guarantee a negligible aliasing probability. The method in [1] applies the concept of MISR for response compaction. However, even one unknown in the scan-out responses would result in an unpredictable good-circuit signature for MISR, from which no faulty-circuit signature could be differentiated. The methods in [2], [3] and [4] attempt to mask unknowns before they reach the time compactor. However, [2] and [3] require pattern-dependent circuitry and [4] requires a special ATPG to support its compaction scheme. Besides, they may mask some known scan-out responses in addition to unknowns.

The *selective compactors* proposed in [5] and [6] select only the scan-out responses with faulty values for observation and hence avoid the masking effect due to unknowns. All other responses are discarded. However, an important assumption of structural testing is that the patterns for the target modeled faults could detect many unmodeled faults as well. Therefore, discarding a majority of scan-out responses would degrade the overall test quality.

Another class of response compactors, called *space compactors*, allow unknown values feeding to the compactor but use an XOR matrix to reduce the probability that a response is masked by the unknown values [8] [9]. The methods proposed in [10] and [11] further use storage elements along with an XOR matrix to improve the compaction ratio and unknown tolerance. The methods proposed in [8], [9], [10] and [11] can guarantee some degree of error detection in the presence of one unknown. However, for high-ratio response compaction, a large number of responses will be processed by the space compactor, and hence the probability that more than one unknown concurrently appears at the inputs of the compactor is high. For such a multiple-unknown situation, some responses may become unobservable due to the masking of unknowns. Different configurations of a space compactor may result in different percentages of responses being observable. The method proposed in [12] analyzes this observable percentage for a special case of a convolutional compactor [11], which contains only one output. A tree-search-based approach proposed in [13] can calculate the observable percentage for a convolutional compactor. However, this tree-search-based approach may not be scalable for a large convolutional compactor.

For the space compactors which can tolerate more than one unknown, the method proposed in [14] uses a binary linear code with Hamming distance d to detect a varying number of errors with a varying number of unknowns up to d-2. In [15], the authors propose a convolutional compactor which can guarantee a certain level of error detection if the total number of unknowns appearing during a period of scan-out cycles is below a target threshold. Due to a more restrictive construction rule in [14] and [15], the methods proposed in [14] and [15] result in limited compaction ratio. In addition, if the number of unknowns exceeds the target threshold, its percentage of observable responses remains unclear.

More importantly, none of these space-compaction schemes can correlate its unknown-tolerant compaction to a test-quality metric such as fault coverage. One-unknown tolerance, multiple-unknown tolerance, or any error-masking probability is not a direct index to the coverage loss caused by the compactor. In fact, our experimental results indicate that this coverage loss is somewhat inversely propor-

DAC 2006, July 24–28, 2006, San Francisco, California, USA.
Copyright 2006 ACM 1-59593-381-6/06/0007 ...$5.00.

tional to the percentages of observable responses under unknown-induced masking for a given test set. Therefore, it is desirable to estimate the percentage of observable responses for a space compactor and then correlate this observable percentage to its coverage loss. In other words, we attempt to measure the quality of a space compactor based on the fault coverage or other test-quality metrics, not on the number of unknowns tolerated or other error-masking probabilities.

In this paper, we develop a design flow for constructing space compactors to achieve a desired level of test quality while maximizing compaction ratio. We first derive an equation to predict the observable percentage of scan-out responses for a given XOR matrix and unknown percentage. Based on this equation, we can efficiently build a space compactor to support a maximal number of scan chains and also observe a desired level of responses. Furthermore, for a given test set, we can predict the coverage of the target fault model and the BCE [16] for different observable percentages. Therefore, when designing a space compactor, we start from specifying a desired level of both target-fault coverage and BCE for a given test set. With this desired coverage, we can determine its corresponding percentage of observable responses. Last, we construct the space compactor with maximal compaction ratio to satisfy this observable percentage, and, in turn, satisfy the desired test quality (i.e. fault coverage and BCE).

The rest of the paper is organized as follows. In Section 2, we first show that the error masking (aliasing) barely affects fault coverage for a space compactor. The unknown-induced masking is the major concern for coverage loss. In Section 3, we describe a mathematical formula for predicting the percentage of observable responses under unknown-induced masking. We also show how to use this mathematical analysis to construct a space compactor with a maximal compaction ratio and a desired observable percentage. In Section 4, we propose a simulation-based scheme to predict the stuck-at-fault coverage and BCE for different observable percentages, so that we can correlate the observable percentage of a space compactor directly with the test quality. Section 5 makes some concluding remarks.

2. MASKING EFFECTS OF XOR MATRIX

Two types of masking effects may affect the fault detection for a space compactor. The first is *error masking* or so-called *aliasing*. Error-masking means that multiple error values cancel one another out through the XOR matrix and result in the same values as those of the fault-free circuit at the outputs of the space compactor. The second masking effect is *unknown-induced masking*. Unknown-induced masking occurs when a compacted output response in the fault-free circuit is infected by unknown values. Once the compacted output response of the fault-free circuit becomes unknown, it is impossible to differentiate any faulty response from this fault-free response.

In practice, error masking (aliasing) rarely affects the fault coverage. This is because few faults will propagate all of the faulty values to the space compactor in the same scan-shift cycle under a test pattern. Even if all of the faulty values arrive in the same scan-shift cycle, aliasing may not happen at every output. A fault is un-detected only if all its faulty values are aliasing at every output and for every test pattern. In Table 1, we show the results of using a simple XOR tree to compact the responses of 50 scan chains into 1 output for four large ISCAS and ITC circuits. In this experiment, we generated ATPG patterns for stuck-at faults without applying any compaction scheme. Then we ran another stuck-at-fault simulation, applying a 50-to-1 XOR tree. No unknown values existed in this experiment. Therefore, the fault coverage loss (if any) is solely due to error-masking.

In Table 1, Column 2 lists the number of ATPG patterns. Column 3 lists the number of detected faults without using any compactor. Column 4 lists the number of un-detected faults using the 50-to-1 XOR tree. Column 5 lists the number of faults which produce aliasing faulty values in at least one test pattern. Column 6 lists the ratio of the aliasing fault count to the total detected fault count in percentage. The results indicate that only a very small portion of faults can generate aliasing faulty values (0.189% at most), even though the compactor in use has a high chance of error masking (50 chains to only one output). Furthermore, those faults generating aliasing faulty values on one pattern may not generate aliasing faulty values on the other patterns. The results in Table 1 show that all aliasing faults can eventually be detected for s35932, s38417, and s38584. Only 15 faults are un-detected for b17. In summary, the fault coverage loss due to error masking is trivial, if any, and thus is not a major issue in designing a space compactor.

circuit	# of pttn	det flt w/o compactor	un-det flt with compactor	aliasing flt	aliasing flt %
s35932	27	43105	0	35	0.081
s38417	189	39177	0	18	0.046
s38584	191	42013	0	16	0.038
b17	536	66638	15	126	0.189

Table 1: Error masking by applying a 50-to-1 simple XOR tree.

On the other hand, unknown-induced masking influences the fault coverage more significantly than error masking. When multiple unknowns go to a space compactor, some responses may not be observed at all due to unknown-induced masking, and the fault coverage is directly proportional to the percentage of observable responses. In this paper, we focus on optimizing the percentage of observable responses with a given unknown percentage. Therefore, our rules for constructing an single-weight Xor matrix is different from the rules of the X-Compactor, which requires an odd *weight* (number of Xor gates each column) and a unique matrix column for each chain. An odd weight and unique columns can reduce the aliasing probability. However, it may not always be helpful in reducing the probability of unknown-induced masking. For our Xor matrix, the weight can be arbitrary and the matrix columns can be identical.

3. OBSERVABLE PERCENTAGE UNDER UNKNOWN-INDUCED MASKING

3.1 Mathematical Analysis

This paper uses single-weight space compactors, in which the number of Xor gates is identical for every column of the Xor matrix. We first give the definition of an *observable response* under unknown-induced masking. A response (at an input of the space compactor) is observable if the response can be propagated to at least one output of the compactor whose fault-free value is not unknown. If all outputs to which this response is propagated are unknowns, then it is an *unobservable response*. We use an exemplary X-Compactor in Figure 1 to show how unknown-induced masking can affect the percentage of responses being observable. This exemplary X-Compactor contains a 20-to-6 XOR matrix, and each scan chain is connected to 3 outputs through the XOR matrix (i.e. weight of 3).

In Figure 1, if there is only one unknown among the 20 responses at the input of the compactor, the remaining 19 responses are all observable. If there are two unknowns among the 20 responses, some of the known responses become unobservable. The number of unobservable responses depends on the locations of the two unknowns. For example, if the two unknowns come from chains 1 and 2, the good-circuit results at 4 outputs (Z_1 to Z_4) become unknown. We define an output with an unknown value in a good-circuit signature as an *unknown-infected output*. These four unknown-infected outputs will make the responses at chains 5 and 11 unobservable. If the two unknowns come from chains 1 and 8, then 5 outputs (Z_1 to

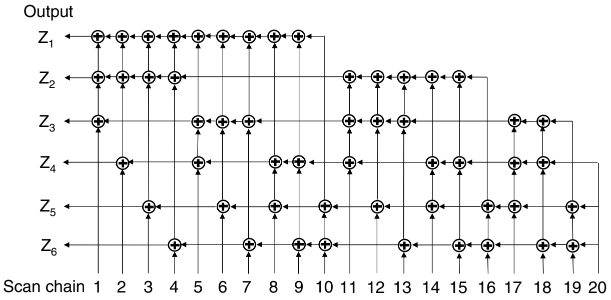
Output
Z1
Z2
Z3
Z4
Z5
Z6
Scan chain 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

Figure 1: **An 20-to-6 exemplary X-Compactor.**

Z_5) become unknown-infected outputs. These five unknown-infected outputs will make the responses at chains 2, 3, 5, 6, 11, 12, 14, and 17 unobservable. If the two unknowns come from chains 1 and 20, then all remaining 18 known responses will be unobservable. Also, if there are more unknowns in the responses, it is likely to have more unknown-infected outputs and hence more unobservable responses. In addition, the number of supported scan chains (i.e. input size of the compactor) and the number of outputs of the compactor will also affect the percentage of observable responses.

In the following, we derive an equation to predict the percentage of observable responses (called *observable percentage* in the rest of the paper). The input parameters for calculating the observable percentage include the number of scan chains S (i.e. the number of inputs of the compactor), the number of outputs of the compactor M, the weight of an XOR matrix W (i.e. the number of outputs a response would propagate to), and the probability that a scan-out response is an unknown (p). Figure 2 illustrates the concept behind our mathematical derivation for the observable percentage. We first define two random variables N and K: N is the number of unknowns among the S responses (shown in the left-hand side of Figure 2), and K is the number of unknown-infected outputs among M outputs (shown in the right-hand side of Figure 2). In the derivation, we first calculate the probability density function for K based on the given N unknowns (mapping from the left to the right in Figure 2). Then based on the K unknown-infected outputs, we calculate the probability that a response (among the S responses) is masked by those K unknown-infected outputs (mapping from the right to the left in Figure 2).

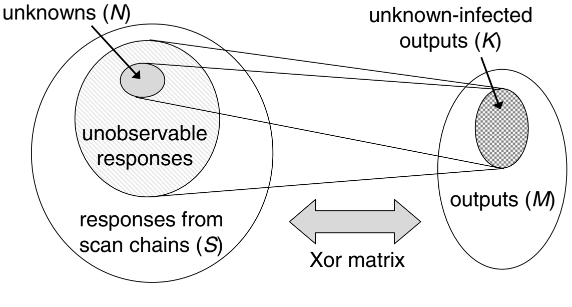

Figure 2: **Illustration for the mathematical derivation of observable probability.**

Given the number of outputs M, the number of unknowns N, and the weight of the XOR matrix W, the probability that k unknown-infected outputs are among the outputs is:

$$Prob\{K=k\} \quad = \quad \frac{\binom{M}{k}}{\binom{M}{W}^N} \sum_{j=W}^{k} (-1)^{k-j} \cdot \binom{k}{j} \cdot \binom{j}{W}^N \qquad (1)$$

where $\binom{i}{j}$ is the number of different combinations of choosing j items from i items. The value of K ranges from W to $min(M, N \cdot W)$.

If there are K unknown-infected outputs among the M outputs, the probability that a response is unobservable (i.e. all mapped outputs of this response are unknown-infected outputs) is $\binom{K}{W}/\binom{M}{W}$, where $\binom{K}{W}$ and $\binom{M}{W}$ represent the number of possible W-output combinations from the K unknown-infected outputs and the M outputs, respectively. Then, the expectation of a response's unobservable probability, $f(K) = \binom{K}{W}/\binom{M}{W}$, is:

$$\begin{aligned} E[f(K)] &= \sum_{k=W}^{min(M,N\cdot W)} \frac{\binom{k}{W}}{\binom{M}{W}} \cdot Prob\{K=k\} && (2) \\ &= \sum_{j=0}^{W} (-1)^j \cdot \left(\frac{\binom{M-j}{W}}{\binom{M}{W}}\right)^N \cdot \binom{W}{j} && (3) \end{aligned}$$

The number of unknowns (N) in Equation 3 is a random variable. N can be represented by the following binomial distribution if the probability of a scan-out response being an unknown (p) is the same for every response:

$$Prob\{N=n\} \quad = \quad \binom{S}{n} \cdot p^n \cdot (1-p)^{S-n} \qquad (4)$$

We can now re-express the $E[f(K)]$ in Equation(3) as a function of N (denoted as $E_N[f(K)]$), and a response's unobservable probability (UP) becomes:

$$\begin{aligned} UP &= \sum_{n=0}^{S} Prob\{N=n\} \cdot E_n[f(K)] && (5) \\ &= \sum_{j=0}^{W} (-1)^j \cdot \binom{W}{j} \cdot \left(p \cdot \frac{\binom{M-j}{W}}{\binom{M}{W}} + 1 - p\right)^S && (6) \end{aligned}$$

Last, the observable percentage (OP) can be expressed as:

$$\begin{aligned} OP &= 1 - UP \\ &= 1 - \sum_{j=0}^{W} (-1)^j \cdot \binom{W}{j} \cdot \left(p \cdot \frac{\binom{M-j}{W}}{\binom{M}{W}} + 1 - p\right)^S && (7) \end{aligned}$$

The detailed steps for derivation from Equation (2) to (3) and Equation (5) to (6) are omitted due to the space limitation.

An important assumption in the above derivation is that unknowns are evenly distributed among the responses. In practice, however, this assumption does not hold. A small portion of the scan chains produce the majority of unknowns ([4] [11] [17]). Fortunately, the above prediction equation still matches the observable percentage well within an acceptable level of accuracy even under the condition of clustered unknowns. Table 2 compares the observable percentage predicted by Equation 7 with the simulation results, for $M = 10$, $S = 100$, $p = 1\%$, and different weights (W). In all the following simulations, we randomly generate 90% of unknowns from 10% of scan chains, and then report the observable percentage of 1-million samples. The average and maximum errors reported in Table 2 are 0.35% and 0.56%, respectively.

weight (W)	1	2	3	4	5	6	avg
predicted obs. %	90.48	95.22	95.52	94.44	92.30	88.99	
simulated obs. %	90.40	95.08	95.31	93.90	91.74	88.50	
error	0.08	0.14	0.21	0.54	0.56	0.49	0.35

Table 2: **Observable percentage with respect to different weights (given 10 outputs, 100 chains, and 1% unknowns).**

We further validate the accuracy of Equation 7 by varying other input parameters. In Table 3, we tried different unknown percentages (p) and fixed the other input parameters ($M = 10$, $S = 100$, $W = 3$). The average and maximum errors are 0.24% and 0.52%, respectively. In Table 4, we tried different numbers of scan chains and fixed the other input parameters. The average and maximum errors are 0.37% and 0.47%, respectively.

unknown % (p)	0.25%	0.50%	0.75%	1.00%	1.50%	2.00%	avg.
predicted obs. %	99.54	98.62	97.26	95.52	91.09	85.70	
simulated obs. %	99.49	98.51	97.10	95.31	90.73	85.18	
error	0.05	0.11	0.16	0.21	0.36	0.52	0.24

Table 3: Observable percentages with respect to different unknown percentages (p) (given 10 outputs, 100 chains, and weight of 3).

chains (S)	100	200	300	400	500	600	avg.
predicted obs. %	95.52	85.67	73.35	60.69	48.95	38.74	
simulated obs. %	95.31	85.36	73.01	60.25	48.48	38.30	
error	0.21	0.31	0.34	0.44	0.47	0.44	0.37

Table 4: Observable percentages with respect to different numbers of scan chains (S) (given 10 outputs, 1% unknowns, and weight of 3).

3.2 Finding Maximal Compaction Ratio for A Desired Observable Percentage

The above mathematical analysis enables an efficient and effective prediction of the observable percentage. It avoids a tedious, brute-force simulation. With this equation, we can easily derive the observable percentage for various configurations of an XOR matrix and further construct a space compactor with a maximal compaction ratio.

The input parameters of this construction procedure are: (1) the number of ATE channels (outputs M) that can be used to observe the scan-out responses, (2) the unknown percentage among the responses (p), and (3) a desired level of observable percentage (op). The goal of this construction procedure is to find out the maximal number of scan chains (S) and an optimal weight (W) under which the observable percentage is higher than the desired level (op). Note that we allow the use of identical Xor columns. Therefore, there is no upper bound for the number of supported scan chains as long as its observable percentage can meet the desired level.

The search for the maximal number of scan chains and optimal weight is straightforward. For each number of scan chains, we just list the observable percentage with respect to each possible weight (i.e. 1 to $M/2$), and then choose the maximal number of scan chains along with a weight which can achieve a higher observable percentage than op. For example, given 8 outputs, 0.5% unknowns, and a desired observable percentage of 90%, Table 5 lists the predicted observable percentages of each weight for different numbers of scan chains in units of 20. Computing those predicted observable percentages takes only seconds. As the table shows, the maximum number of scan chains is 240 and the optimal weight is 2 for the desired observable percentage of 90%. In a similar manner, we can further continue the experiment to search for a more specific number of scan chains within 240-260.

Maximizing the number of scan chains can reduce the test data volume stored for the good-circuit results of the space compactor. Also, it can shorten the test application time because the length of the scan chains becomes shorter. However, if there is a limit on the number of scan chains, we could connect multiple scan cells of a scan chain to the compactor as shown in Figure 3. In this triple-connection example, once the output result of the compactor is observed by the ATE for one scan-out cycle, the results for the next two scan-shift cycles will be discarded. In this way, the test data volume can be reduced although the time application time remains the same.

chains	W=1	W=2	W=3	W=4
160	90.49	94.79	94.72	92.99
180	89.36	93.87	93.62	91.48
200	88.25	92.91	92.45	89.89
220	87.15	91.92	91.21	88.21
240	86.03	90.89	89.92	86.45
260	85.00	89.83	88.57	84.64
280	83.94	88.74	87.17	82.78

Table 5: Predicted observable percentage with respect to different numbers of chains and weights, given $M = 8$ and $p = 0.5\%$.

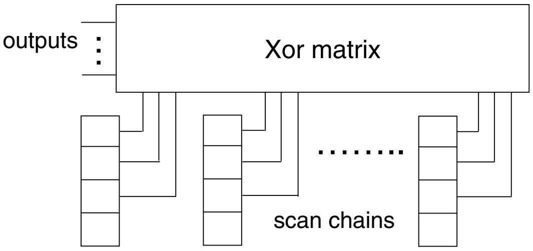

Figure 3: Multiple connection of one scan chain.

4. OBSERVABLE PERCENTAGE VS. TEST QUALITY

4.1 Fault Coverage and BCE for Different Observable Percentages

The next question we attempt to answer is: how would the observable percentage be related to the test quality? As an attempt to make some observations before answering this question, we conducted experiments on four large ITC and ISCAS circuits (with more than 1000 flip-flops) to observe the trend of observable percentages on the test quality. We used the same stuck-at-fault patterns as those used in Table 1. The patterns were generated by an industrial, in-house ATPG. We ran stuck-at-fault simulation using these patterns and randomly selected scan-out responses for observation based on a given observable percentage. Table 6 lists the stuck-at-fault coverage and the coverage loss for different observable percentages. The definition of coverage loss is the coverage of observing all responses ($obs.\% = 1.0$) minus the coverage of the given observable percentage. The results show that the coverage loss with respect to an observable percentage is highly circuit-dependent. Using the average to predict the coverage loss would result in a significant error.

ckt	obs. %	0.1	0.3	0.5	0.7	0.9	0.95	1.0
b17	coverage	74.74	84.93	88.57	90.50	91.62	91.84	92.04
	cov. loss	17.30	7.11	3.47	1.54	0.42	0.20	0.00
s35932	coverage	51.61	78.20	86.24	89.43	91.08	91.34	91.54
	cov. loss	39.93	13.34	5.30	2.11	0.46	0.20	0.00
s38417	coverage	87.10	95.78	97.73	98.80	99.29	99.46	99.53
	cov. loss	12.43	3.75	1.8	0.73	0.24	0.07	0.00
s38584	coverage	77.91	89.29	93.22	95.07	96.06	96.29	96.44
	cov. loss	18.53	7.15	3.22	1.37	0.38	0.15	0.00
avg.	cov. loss	22.05	7.84	3.45	1.44	0.38	0.16	0.00

Table 6: Stuck-at-fault coverage loss with respect to different observable percentages.

We conducted similar experiments for another important test quality metric, BCE [16], which quantifies the impact of multiple-detection of the stuck-at faults and can also approximate the bridge-fault coverage. Table 7 lists the BCE's and the BCE loss for different observable percentages. A similar trend was observed as in Table 7. The BCE and the loss vary from circuit to circuit. Therefore, we can conclude that the circuit and the test patterns must be taken as input parameters for accurate analysis.

ckt	obs. %	0.1	0.3	0.5	0.7	0.9	0.95	1.0
b17	BCE	66.16	77.77	82.16	84.65	86.23	86.56	86.86
	BCE loss	20.70	9.09	4.70	2.21	0.63	0.30	0.00
s35932	BCE	32.88	61.49	74.08	80.50	84.59	85.32	86.00
	BCE loss	53.12	24.51	11.92	5.50	1.41	0.68	0.00
s38417	BCE	77.76	89.96	92.87	94.40	95.14	95.33	95.43
	BCE loss	17.67	5.47	2.56	1.03	0.29	0.10	0.00
s38584	BCE	68.81	81.82	86.94	89.73	91.43	91.77	92.06
	BCE loss	23.25	10.24	5.12	2.33	0.63	0.29	0.00
avg.	cov. loss	28.69	12.33	6.08	2.77	0.74	0.34	0.00

Table 7: BCE loss with respect to different observable percentages.

4.2 Prediction of Fault Coverage and BCE

In this session, we propose a one-time fault-simulation-based method to predict both the stuck-at-fault coverage and BCE for any given list of observable percentages. The inputs to this prediction method include a test set, the circuit, and a list of observable percentages. The outputs are the predicted fault coverage and BCE for each of the given observable percentages in the list. In this prediction method, we first fault-simulate the given pattern set assuming all responses to be fully observable. During the fault simulation, we compute two additional statistics for each fault. One is the number of patterns that a fault f is detected by, denoted as DN_f. The other statistic is the total number of outputs (i.e. scan-out responses) that a fault can be propagated to, for the whole pattern set. We denote it as ON_f for fault f. After the fault simulation, we calculate the fault coverage and BCE for each observable percentage based on the DN_f and ON_f statistics.

We first show the prediction for fault coverage. A fault cannot be detected if all ON_f observation outputs cannot be observed under an observable percentage (denoted as *op*). Therefore, the probability that a fault f can be detected (denoted as det_prob_f) under a given observable percentage is:

$$det\_prob_f \quad = \quad 1 - (1 - op)^{ON_f} \tag{8}$$

The predicted fault coverage (denoted as FC) of a given observable percentage will then be:

$$FC \quad = \quad \frac{1}{|F|} \sum_{f \in F} det\_prob_f \tag{9}$$

where F is the set of faults and $|F|$ is the size of the fault set.

We then show the prediction for BCE. The BCE proposed in [16] is defined as:

$$BCE \quad = \quad \frac{1}{|F|} \sum_{f \in F} (1 - 0.5^{N_f}) \tag{10}$$

The key parameter in Equation 10 is the number of detecting patterns (N_f) for each fault. N_f is equal to DN_f when all responses can be observed. Here, we attempt to calculate the expectation of the BCE function for a fault f (denoted as $BCE_f = 1 - 0.5^{N_f}$), for given *op*, DN_f, and ON_f.

For each detecting pattern pn of fault f, the probability that f remains detected under the observable percentage op depends on the number of observation outputs for pn. For a fault f, we denote the detection probability of a pattern pn under op as $DP_{f,pn}(op)$, and the number of observation outputs of a pattern pn as $NOO_{f,pn}$. The detection probability $DP_{f,pn}(op)$ can therefore be expressed as:

$$DP_{f,pn}(op) \quad = \quad 1 - (1 - op)^{NOO_{f,pn}} \tag{11}$$

In our experiments, we found that the numbers of observation outputs are very similar for most detecting patterns. So we could approximate $NOO_{f,pn}$ as $\frac{ON_f}{DN_f}$ for every detecting pattern. That is, we assume the detection probability $DP_{f,pn}(op)$ of a fault f to be the same for each of its detecting patterns. Therefore we use a function a_f to replace $DP_{f,pn}(op)$ as follows:

$$a_f \quad = \quad 1 - (1 - op)^{\frac{ON_f}{DN_f}} \tag{12}$$

Because each of the detecting patterns has the same detection probability, we could model the number of detecting patterns of a fault f (N_f in Equation 10) using the following binomial function, for a given observable percentage *op*:

$$Prob\{N_f = n\} = \binom{DN_f}{n} \cdot (a_f)^n \cdot (1 - a_f)^{DN_f - n} \tag{13}$$

The expectation of the BCE function for a fault f (BCE_f) can then be calculated using the following equation:

$$\begin{aligned} E[BCE_f] \quad &= \quad \sum_{n=0}^{DN_f} (1 - 0.5^n) \cdot Prob\{N_f = n\} && (14) \\ &= \quad 1 - (1 - 0.5 \cdot a_f)^{DN_f} && (15) \end{aligned}$$

Last, we use the expectation value of BCE_f in Equation 15 to estimate the BCE for a given *op*:

$$BCE \quad = \quad \frac{1}{|F|} \sum_{f \in F} E[BCE_f] \tag{16}$$

In Tables 8 and 9, we compare the predicted stuck-at-fault coverages (prd. cov) and the predicted BCE's (prd. BCE) with the simulation results (sim. cov and sim. BCE). The results show that the average errors of the stuck-at-fault coverage are 0.02%, 0.08%, 0.06%, and 0.03% respectively for the four circuits. The average errors of the BCE are 0.14%, 0.08%, 0.14%, and 0.02% respectively. These tables show the accuracy of the proposed prediction scheme based on Equations 9 and 16.

Simulation for the BCE calculation takes longer than the fault simulation for the coverage calculation. For the BCE calculation, a fault cannot be dropped until the number of detections exceeds a threshold when the change of BCE resulting from any more detections of the fault can be ignored. In the proposed prediction scheme, we drop a fault with more stringent constraints because we need to consider the accuracy of the BCE's for all *op*'s in the given list. In this case, the threshold of the number of detections for fault-dropping depends on the predicted BCE of the lowest *op*. Note that the most time-consuming part of the proposed prediction scheme is fault simulation. Calculating the predicted coverage and the BCE for each *op* based on Equations 9 and 16 takes only a small fraction of the total runtime. In Table 10, we list the runtime of the original BCE calculation (a) and our prediction scheme for 20 *op*'s (b), and for 40 *op*'s (c). The lowest *op* in this experiment is 0.5. The results show that the extra

ckt	obs. %	0.5	0.6	0.7	0.8	0.9	0.95	avg
b17	sim. cov.	88.57	89.62	90.50	91.14	91.62	91.84	
	prd. cov.	88.58	89.66	90.48	91.12	91.63	91.85	
	error	0.01	0.04	0.02	0.02	0.01	0.01	0.02
s35932	sim. cov.	86.24	88.01	89.43	90.39	91.08	91.34	
	prd. cov.	86.05	88.12	89.51	90.45	91.09	91.34	
	error	0.19	0.11	0.08	0.06	0.01	0.00	0.08
s38417	sim. cov.	97.73	98.37	98.80	99.11	99.29	99.46	
	prd. cov.	97.87	98.44	98.85	99.14	99.37	99.45	
	error	0.14	0.07	0.05	0.04	0.08	0.00	0.06
s38584	sim. cov.	93.22	94.32	95.07	95.62	96.06	96.29	
	prd. cov.	93.19	94.26	95.04	95.63	96.09	96.27	
	error	0.03	0.05	0.03	0.02	0.03	0.01	0.03

Table 8: Comparison between the simulated coverage (sim. cov.) and the predicted coverage (prd. cov.) for different observable %.

ckt	obs. %	0.5	0.6	0.7	0.8	0.9	0.95	avg
b17	sim. BCE	82.16	83.54	84.65	85.53	86.23	86.56	
	prd. BCE	82.46	83.79	84.75	85.60	86.29	86.59	
	error	0.30	0.25	0.10	0.07	0.06	0.03	0.14
s35932	sim. BCE	74.08	77.58	80.50	82.78	84.59	85.32	
	prd. BCE	73.90	77.71	80.59	82.82	84.59	85.33	
	error	0.18	0.13	0.09	0.04	0.00	0.01	0.08
s38417	sim. BCE	92.87	93.76	94.40	94.84	95.14	95.33	
	prd. BCE	93.18	93.97	94.53	94.93	95.22	95.33	
	error	0.31	0.21	0.13	0.09	0.08	0.00	0.14
s38584	sim. BCE	86.94	88.56	89.73	90.68	91.43	91.77	
	prd. BCE	86.94	88.53	89.75	90.70	91.46	91.78	
	error	0.00	0.03	0.02	0.02	0.03	0.01	0.02

Table 9: Comparison between the simulated BCE (sim. cov.) and the predicted BCE (prd. cov.) for different observable %.

runtime required for the proposed prediction scheme is quite low, in comparison with the original BCE calculation. Also, adding more op's in the list requires little extra runtime. Please also note that, we calculate both the predicted stuck-at-fault coverage and the BCE with one-time fault simulation. So the runtime reported in Table 10 already includes the runtime for stuck-at-fault coverage prediction.

ckt	original (a)	20-op prd. (b)	40-op prd. (c)	(b)-(a)	(c)-(b)
b17	114.7	126.4	128.2	11.7	1.8
s35932	6.4	7.7	8.8	1.3	1.1
s38417	23.1	26.9	27.5	3.8	0.6
s38584	20.1	23.0	23.3	2.9	0.3

Table 10: Runtime comparison (in seconds) between original BCE simulation and our prediction scheme.

4.3 Construct Space Compactors with Desired Level of Test Quality

In this section, we summarize the flow for constructing an optimal space compactor. Given the circuit, the structural test set, the number of available ATE channels M for observing responses, and a desired level of stuck-at-fault coverage or BCE (c_level), we identify the configuration of the space compactor and the number of scan chains based on the following steps:

1. Extract unknown percentage (xp) from the responses of the test set using logic simulation.

2. List the predicted fault coverages and BCEs for different observable percentages based on the test set (as in Section 4.2).

3. Choose a desired observable percentage (op) satisfying c_level.

4. Construct the compactor with a maximal number of scan chains (max_c) based on given M, xp, and op (as in Section 3.2).

5. Divide the scan cells into max_c chains or use the multiple-connection scheme for each chain (as discussed in Figure 3).

5. CONCLUSION

This paper first shows that the error-masking effect has little impact on fault coverage for designing a space compactor. The effect of unknown-induced masking is, in fact, the main concern. We propose an analytical framework for predicting the percentage of observable responses under unknown-induced masking for a space compactor. This analysis is also valid for clustered unknowns. Based on this analysis, we can efficiently construct a space compactor with a maximal compaction ratio for achieving a desired level of observable percentage. Furthermore, we propose a simulation-based scheme to correlate an observable percentage of responses to its fault coverage and BCE. Enabled by this scheme, we can efficiently evaluate the quality of a space compactor based on the test-quality metrics such as fault coverage and BCE, instead of the indirect test-quality metrics like the number of tolerated unknowns or the aliasing probability.

6. REFERENCES

[1] C. Barnhart, V. Brunkhorst, F. Distler, O. Farnsworth, B. Keller, and B. Koenemann, OPMISR: The Foundation for Compressed ATPG Vectors, *IEEE Int'l Test Conference*, pp.748-757, 2001.

[2] I. Pomeranz, S. Kundu, and S. M. Reddy, Masking of Unknown Output Values during Output Response Compression by Using Comparison Units, *IEEE Tran. on Computers*, Vol. 53, No. 1, Jan, pp.83-89, 2004.

[3] Y. Tang, H.-J. Wunderlich, H. Vranken, F. Hapke, M. Wittke, P. Engelke, I. Polian, and B. Becker, X-Masking During Logic BIST and Its Impact on Defect Coverage, *IEEE Int'l Test Conference*, pp.442-451, 2004.

[4] V. Chickermane, B. Foutz, and B. Keller, Channel Masking Synthesis for Efficient On-Chip Test Compression, *IEEE Int'l Test Conference*, pp.452-460, 2004.

[5] P. Wohl, J. A. Waicukauski, S. Patel, and M. B. Amin, X-Tolerant Compression and Application of Scan-ATPG Patterns in a BIST Architecture, *IEEE Int'l Test Conference*, pp.727-736, 2003.

[6] J. Rajski, J. Tyszer, M. Kassab, and N. Mukherjee, Selective Linear Compactor of Test Responses with Unknown Values, USA pending patent application, 2002.

[7] H. Tang, C. Wang, J. Rajski, S. M. Meddy, J. Tyszer, and I. Pomeranz, On Efficient X-handling Using a Selective Compaction Scheme to Achieve High Test Response Compaction Ratios, *IEEE Int'l Conference on VLSI Design*, pp.59-64, 2005.

[8] S. Mitra, and K. S. Kim, X-Compact: an efficient response compaction technique for test cost reduction, *IEEE Int'l Test Conference*, pp.311-320, 2002.

[9] P. Wohl, and L. Huisman, Analysis and Design of Optimal Combinational Compactors, *IEEE VLSI Test Symposium*, pp. 101-106, 2003.

[10] C. Wang, S. M. Reddy, I. Pomeranz, J. Rajski, and J. Tyszer, On Compacting Test Response Data Containing Unknown Values, *ACM/IEEE Int'l Conference on Computer Aided Design*, pp.855-862, November 2003.

[11] J. Rajski, C. Wang, and S. M. Reddy, Convolutional Compaction Of Test Responses, *IEEE Int'l Test Conference*, pp. 745-754, 2003.

[12] Y. Han, Y. Hu, H. Li, and X. Li, Theoretic Analysis and Enhanced X-Tolerance of Test Response Compact based on Convolutional Code, *IEEE Asia and South Pacific Design Automation Conference*, pp. 53-58, 2005.

[13] M. Arai, S. Fukumoto, and K. Iwasaki, Analysis of Error-Masking and X-Masking Probabilities for Convolutional Compactors, *IEEE Int'l Test Conference*, paper 24.1, 2005.

[14] J. H. Patel, S. S.Lumetta, and S. M. Reddy, Application of Saluja-Karpovsky Compactors to Test Responses with Many Unknowns, *IEEE VLSI Test Symposium*, pp. 107-112, 2003.

[15] J. Rajski, and J. Tyszer, Synthesis of X-tolerant Convolutional Compactors, *IEEE VLSI Test Symposium*, pp. 114-119, 2005.

[16] B. Benware, C. Schuermyer, N. Tamarapalli, K. Tsai, S. Ranganathan, R. Madge, J. Rajski, and P. Krishnamurthy, Impact of Multiple-Detect Test Patterns on Product Quality, *IEEE Int'l Test Conference*, pp.1031-1040, 2003.

[17] E. H. Volkerink, and S. Mitra, Response Compaction with any Number of Unknowns Using a New LFSR Architecture, *ACM/IEEE Design Automation Conference*, pp.1031-1040, 2003.