LETTER Special Section on Test and Verification of VLSIs

Post-BIST Fault Diagnosis for Multiple Faults

Hiroshi TAKAHASHI[†a)], **Yoshinobu HIGAMI**[†], *Members*, **Shuhei KADOYAMA**[†], *Student Member*, **Yuzo TAKAMATSU**[†], **Koji YAMAZAKI**[††], **Takashi AIKYO**[†††], *and* **Yasuo SATO**[†††], *Members*

SUMMARY With the increasing complexity of LSI, Built-In Self Test (BIST) is a promising technique for production testing. We herein propose a method for diagnosing multiple stuck-at faults based on the compressed responses from BIST. We refer to fault diagnosis based on the ambiguous test pattern set obtained by the compressed responses of BIST as *post-BIST fault diagnosis* [1]. In the present paper, we propose an effective method by which to perform post-BIST fault diagnosis for multiple stuck-at faults. The efficiency of the success ratio and the feasibility of diagnosing large circuits are discussed.

key words: *post-BIST fault diagnosis, multiple stuck-at faults, combinational circuits, pass/fail information*

1. Introduction

Built-In Self-Test (BIST) is as effective method of testing a circuit under test (CUT) in the production test. Previous research has focused on developing methods to increase the diagnostic information derived from the results of BIST [3], [6]. It is difficult to identify true the failing test patterns of BIST due to the large number of test patterns applied to CUT and the high degree of the output response compaction using the signature analyzer.

Therefore, we summarize the solution for developing the diagnosis method after BIST.

1) On the fault diagnosis after BIST, we have to develop the diagnosis method by using the passing test pattern set and the ambiguous failing test pattern set that consists of the failing test pattern set and several passing test patterns.

2) We have to develop the diagnosis method without using the information (locations and faulty values) derived from failing scan-cells and failing primary outputs.

We refer to the method for diagnosing the failed LSI based on the result of BIST as *the post-BIST fault diagnosis* [1].

In [1], we proposed a method for diagnosing single stuck-at faults under the BIST environment. However, it is difficult for the method in [1] to apply the faulty circuits with multiple stuck-at faults directly. In practice, a faulty circuit is more likely to have stuck-at faults at multiple locations. Therefore, we propose an effective method to diagnose the failed LSI with multiple stuck-at faults based on the result of BIST [2]. We use large benchmark circuits designed at the Semiconductor Technology Academic Research Center (STARC) [5] to confirm the feasibility of diagnosing the large circuits on the post-BIST fault diagnosis.

The remainder of the paper is organized as follows: In Sect. 2, we propose a method of post-BIST fault diagnosis for multiple stuck-at faults based the pass/fail information. In Sect. 3, we evaluate the effectiveness of the proposed method of post-BIST fault diagnosis by experiments conducted on benchmark circuits and prove the feasibility of diagnosing multiple stuck-at faults by post-BIST fault diagnosis.

2. Post-BIST Fault Diagnosis for Multiple Stuck-at Faults

2.1 Outline of the Proposed Method

First, we present some definitions.

Definition 1: For a test pattern applied to the CUT, if at lest one primary output value of CUT is different from that of the expected fault-free circuit, then the test pattern is called a *failing test pattern*. On the other hand, if all values at primary outputs of CUT applying a test pattern are the same as those of the expected fault-free circuit, then the test is called a *passing test pattern*.

In the present study, we assume that the proposed diagnosis method is performed under the BIST environment presented in [3], [6]. Under the BIST environment presented in [3], [6], the large number of test patterns is divided into intervals of a fixed number of test patterns. When the response signature for the interval is fault-free, the test patterns in the passing interval are the passing test patterns. A set of passing test patterns denotes *a passing test pattern set (PT_set)*. When the response signature for the interval is faulty, the methods proposed in [3], [6] can be used to identify the suspected failing test patterns from the test patterns in the failing interval. However, the failing test pattern set identified by the methods described in [3], [6] includes the passing test patterns in CUT due to the high degree of the output response compaction by using the signature analysis.

Manuscript received April 11, 2007.
Manuscript revised August 14, 2007.

[†]The authors are with the Department of Electrical and Electronic Engineering and Computer Science, Graduate School of Science and Engineering, Ehime University, Matsuyama-shi, 790–8577 Japan.

[††]The author is with the School of Information and Communication, Meiji University, Tokyo, 168–8555 Japan.

[†††]The authors are with the Semiconductor Technology Academic Research Center (STARC), Yokohama-shi, 222–0033 Japan.

a) E-mail: takahasi@cs.ehime.ac.jp

DOI: 10.1093/ietisy/e91–d.3.771

Therefore, the failing test pattern set obtained by the methods in [3], [6] denotes *the ambiguous failing test pattern set (AFT_set)*. The *AFT_set* consists of $Tf \cup Tp$, where Tf denotes the set of failing test patterns and Tp denotes the set of passing test patterns. However, the same Tp is not included in the *PT_set*

It is not necessary for the proposed method to use all failing test patterns for the CUT. We use part of the failing test patterns for the CUT to diagnose the multiple stuck-at faults.

The fault model assumed herein is multiple stuck-at faults in combinational circuits.

Definition 2:

Let *the candidate faults* be the faults that may exist in the CUT. Let *CF_set* be the set of candidate faults.

Let *the non-existing faults* be the faults that may not exist in the CUT.

The proposed method consists of the main procedure and the post procedure. The main procedure determines the candidate faults using the ambiguous failing test pattern set and the passing test pattern set. The post procedure scores the candidate faults based on detection times and structural information.

The pass/fail state of the failing test pattern or the passing test pattern depends on whether the test pattern can detect the candidate fault. We use a single stuck-at fault simulation to obtain the pass/fail state. The detection times of the candidate fault $f\alpha$ under the ambiguous failing test pattern set or the passing test pattern set indicates the number of test patterns that can detect the candidate fault $f\alpha$.

2.2 Main Procedure Using Failing Test Patterns and Passing Test Patterns

Main procedure of the multiple fault diagnosis deduces candidate faults based on the following observations.

Observation 1: The number of detected faults in every test pattern $t \in AFT\_set$ is not always the same.

In order to prevent the increase in the number of candidate faults, the failing test pattern with a small number of detected faults is selected first.

Under multiple stuck-at faults, we must consider the fault masking relationship for the actual faults. Therefore, we use the following observation to deduce the non-existing faults from the set of candidate faults.

Observation 2: Let us consider the candidate fault $F\alpha$ deduced by the diagnosis method using failing test patterns. If the candidate fault $F\alpha$ is detected by several passing test patterns, the candidate fault $F\alpha$ might be the non-existing fault in CUT.

Main procedure: Method for deducing candidate faults:

Inputs for main procedure:

- *AFT_set* and *PT_set*
- The threshold value N for the detection times under the passing test patterns: N is used to prevent missing the actual faults from the set of candidate faults.
- The set of faults for the diagnosis using the ambiguous failing test pattern set (*SIM_set*): Initial set of *SIM_set* is the set of representative faults.
- Set of candidate faults (*CF_set*): Initial set of *CF_set* is empty.

Output of main procedure:

- Set of candidate faults deduced by the main procedure

Step 1 Perform a single stuck-at fault simulation with each failing test pattern $t \in AFT\_set$ to calculate the number of faults $\in SIM\_set$ detected by each failing test pattern, where the test patterns that cannot detect any fault $\in SIM\_set$ are removed from the *AFT_set*. In this step, we use the single stuck-at fault simulation without fault dropping.
Sort the test patterns in *AFT_set* in ascending order of the number of detected faults in *SIM_set*. We obtain the ordered *AFT_set* with respect to the number of faults detected by the test patters.

Step 2 Repeat the procedures of from Step 3 to Step 6 until all test patterns in the ordered *AFT_set* are selected.

Step 3 Select the top ranked test pattern from the unselected test patterns in the ordered *AFT_set*. The number of faults detected by the top ranked test pattern is the smallest number among the unselected test patterns.

Step 4 Perform the single stuck-at fault simulation with fault dropping. Add the faults $\in SIM\_set$ that can be detected by the selected test pattern to *CF_set*.

Step 5 Perform the single stuck-at fault simulation for each fault in *CF_set* with all test patterns in *PT_set*. From the result of the single stuck-at fault simulation without fault dropping we identify the candidate faults that can be detected by N or more passing test patterns. Remove the candidate faults that can be detected by N or more passing test patterns from *CF_set*.

Step 6 Perform the following procedure for all unselected test patterns in *AFT_set*.
Perform the single stuck-at fault simulation for all candidate faults in *CF_set* with the unselected test pattern. We check whether the test pattern in *AFT_set* can detect at least one candidate fault in *CF_set*. If the test pattern in *AFT_set* cannot detect any candidate fault in *CF_set*, leave the test pattern in *AFT_set* as the unselected test pattern. If the test pattern in *AFT_set* can detect at least one candidate fault in *CF_set*, remove the test pattern from *AFT_set* as the selected test pattern. In this step, we use the single stuck-at fault simulation without fault dropping.

2.3 Post Procedure: Method for Scoring Candidate Faults Based on Detection Times and Structural Information

On the post procedure, we estimate the possibility for candidate faults. We rank the candidate faults in *CF_set* according to the estimated values for the candidate faults.

We introduce the following estimations (E_1, E_2, and E_3) to calculate the estimation $E_{f_{\alpha i}}$ for the candidate fault $f_{\alpha i}$.

We assume that the actual faults can be detected by many failing test patterns. We also assume that the actual faults can be detected by a small number of passing test patterns. The estimated values E1 and E2 take the above qualitative relationships. E3 takes the depth of the fault.

For $f_{\alpha i}$ $(i = 1, 2, \ldots, n)$, we calculate the following estimated values, where n is the total number of candidate faults.

estimated value E_1:

We sort n candidate faults in descending order of detection time for n candidate faults on the test patterns in AFT_set. The rank of $f_{\alpha i}$ is denoted by $[FT\_rank(f_{\alpha i})]$.

$$E_1(f_{\alpha i}) = \frac{1}{FT\_rank(f_{\alpha i})} \tag{1}$$

estimated value E_2:

We sort n candidate faults in ascending order of detection time for n candidate faults on the test patterns in PT_set. The rank of $f_{\alpha i}$ is denoted by $[PT\_rank(f_{\alpha i})]$.

$$E_2(f_{\alpha i}) = \frac{1}{PT\_rank(f_{\alpha i})} \tag{2}$$

estimated value E_3 [4]:

The level of $f_{\alpha i}$ is denoted as $[Level(f_{\alpha i})]$. The level of an output of the gate is calculated in the same manner described in [4]. We assume that all primary outputs have level 1. *MAX_LEVEL* is the maximum level of the output of the gate in the circuit.

$$E_3(f_{\alpha i}) = \frac{Level(f_{\alpha i})}{MAX\_LEVEL} \tag{3}$$

In the post-procedure, we calculate the estimation $E_{f_{\alpha i}}$ for $f_{\alpha i} \in CF\_set$ $(i = 1, 2, \ldots, n)$ to rank the candidate fault $f_{\alpha i}$.

$$E_{f_{\alpha i}} = E_1(f_{\alpha i}) \times E_2(f_{\alpha i}) \times E_3(f_{\alpha i}) \tag{4}$$

3. Experimental Results for Multiple Stuck-at Faults

Experiments using the proposed method for multiple stuck-at faults were performed for ISCAS'85 and full-scan versions of ISCAS'89 benchmark circuits and for two large circuits in STARC03 benchmark circuits [5]. The STARC03 benchmark circuits are designed to evaluate various tools and methods for SoC (System On Chip) design at the Semiconductor Technology Academic Research Center (STARC). The specifications of the STARC03 benchmark circuits are shown in Table 1. The STARC03 benchmark circuits used in this experiment are the full-scan circuits.

The program was run on a computer having a Pentium 4 (3.4-GHz) CPU and 3 GB of memory. In these experiments, double stuck-at faults, or quadruple stuck-at faults, were injected randomly. We used 1,024 random patterns. The accuracy of the ambiguous failing test pattern set is 95%, where the accuracy of the ambiguous failing test pattern set is defined as follows:

$$\begin{aligned}&\text{Accuracy of } AFT\_set(\%)\\ &= \frac{\#\_of\_true\_failing\_test\_patterns\_in\_AFT\_set}{\#\_of\_test\_patterns\_in\_AFT\_set}\\ &\quad \times 100\end{aligned} \tag{5}$$

We randomly select the passing test patterns from among the passing test patterns for CUT and add the selected passing test patterns to the ambiguous failing test pattern set.

In this experiment, we set N to 20 in Step 5 of the main procedure. In Step 5 of the main procedure, N is determined by the results of a preliminary experiment. We perform the single stuck-at fault simulation for several faulty circuits in order to count how many times actual faults are detected by the passing test patterns.

Table 2 shows the average CPU time consumed by the main procedure, the average CPU time consumed by the post procedure, the average hit ratio (*hit 2, hit1*) of double faults, and the average success ratio (*SR*) for each benchmark circuit with double stuck-at faults. Because of space limitations, we do not show the results of small circuits in ISCAS benchmark circuits.

According to researchers at the Semiconductor Technology Academic Research Center (STARC), the desired value of the *success ratio* is equal to 20 faults, or five faults from the top ranked candidate fault. In each circuit, the results for 20 faults from the top ranked candidate fault are shown in the first line. In addition, the results for five faults from the top ranked candidate fault are shown in the second line.

In the present study, the *hit ratio* is a generic term for *hit Ns*. *Hit N* is defined as the ratio of successfully containing N actual fault(s) within 5 (20) candidate faults from the top ranked candidate fault. In other words, hit Ns shows the distribution of the success ratio.

The number of cases in which N actual faults are included in less than or equal to 5 (20) candidate faults is denoted as the # of N_ successful cases. Hit N is defined as follows:

$$hitN(\%) = \frac{\#\_of\_N\_successful\_cases}{\#\_of\_faulty\_circuits} \times 100 \tag{6}$$

The success ratio (*SR* is defined by the following equation. The number of cases in which at least one actual fault is included in less than of equal to 5 (20) faults from the top ranked candidate faults is denoted as the # of successful cases.

$$success\_ratio(SR) = \frac{\#\_of\_successful\_cases}{\#\_of\_faulty\_circuits} \times 100 \tag{7}$$

A success ratio of 100% indicates that the proposed method is able to diagnose at least one stuck-at fault correctly within 5 (20) candidate faults from the top ranked

Table 1 Specifications of STARC03 benchmark circuits.

circuits	# of inputs	# of outputs	# of lines	# of gates	# of faults
STARC03_ct1	1,486	1,307	98,659	44,516	117,664
STARC03_ct2	18,373	18,179	263,455	107,685	285,076

Table 2 Experimental results for double faults under accuracy of ambiguous failing test set = 95%.

circuits	CPU (sec.) for main procedure	CPU (sec.) for post procedure	CF	hit2	hit1	SR(%)
c6288	34.19	0.15	20	98	2	100
			5	93	6	99
c7552	8.81	0.17	20	92	8	100
			5	76	24	100
cs9234	5.64	0.23	20	58	38	96
			5	25	58	83
cs15850	18.84	0.57	20	74	23	97
			5	41	46	87
cs35932	318.17	2.01	20	81	19	100
			5	80	20	100
cs38417	122.57	1.96	20	90	10	100
			5	52	40	92
cs38584	145.23	2.06	20	65	35	100
			5	55	43	98
STARC03_ct1	1885.2	3.8	20	96	4	100
			5	89	11	100
STARC03_ct2	9069.8	45.4	20	58	40	98
			5	51	43	94

Table 3 Experimental results for quad. faults under accuracy of ambiguous failing test set = 95%.

circuits	CPU (sec.) for main procedure	CPU (sec.) for post procedure	CF	hit4	hit3	hit2	hit1	SR (%)
c6288	51.06	0.38	20	80	20	0	0	100
			5	0	2	96	2	100
c7552	14.08	0.26	20	64	32	4	0	100
			5	8	38	48	4	98
cs9234	9.03	0.34	20	8	46	28	18	100
			5	2	14	30	46	92
cs15850	31.52	0.81	20	24	40	24	12	100
			5	0	24	38	32	94
cs35932	462.97	9.03	20	6	28	46	20	100
			5	0	18	60	22	100
cs38417	176.86	2.73	20	34	46	20	0	100
			5	2	26	34	26	88
cs38584	221.19	2.87	20	8	34	58	0	100
			5	2	28	58	12	100
STARC03_ct1	2443.6	5.9	20	54	42	4	0	100
			5	36	48	14	2	100
STARC03_ct2	11224.2	61.1	20	0	26	64	10	100
			5	0	16	64	18	98

candidate fault for all faulty circuits used in this experiment.

Table 3 shows the results for quadruple faults for the benchmark circuits, respectively.

From the experimental results shown in the tables, it is clear that high success ratios of approximately 98% are obtained by the proposed method. Even though the fault multiplicity increases, the success ratio is not degraded.

Most of the CPU time was spent on the main procedure.

The experimental result for quadruple faults indicate that the CPU time for diagnosing the STARC03_ct2 circuit that consists of 100 K gates is approximately three hours. Therefore, the proposed diagnosis method is feasible for diagnosing large circuits.

The proposed method of post-BIST fault diagnosis gives good diagnostic results in practical CPU times. We believe that the proposed method is more amenable to the diagnosis of multiple stuck-at faults in post-BIST fault diagnosis because the proposed method does not use any information about the locations of the primary outputs having faulty responses in CUT.

4. Conclusions

In order to provide high-quality post-BIST fault diagnosis, we have proposed a method for improving the diagnostic accuracy for multiple stuck-at faults based on only pass/fail

information. From the experimental results, we confirmed the feasibility of diagnosing multiple stuck-at faults on the post-BIST fault diagnosis. Therefore, we believe that the proposed method is effective for post-BIST fault diagnosis.

Further study is necessary to clarify the relationship between the number of failing test patterns used in the diagnosis and accuracy of the diagnostic result.

Acknowledgment

This work was supported in part by the Semiconductor Technology Academic Research Center (STARC) through a research project.

References

[1] H. Takahashi, Y. Tsugaoka, H. Ayano, and Y. Takamatsu, "BIST based diagnosis using ambiguous test set," Proc. 18th Int. Symp. Defect and Fault Tolerance in VLSI Systems, pp.89–96, Oct. 2003.

[2] H. Takahashi, S. Kadoyama, Y. Higami, Y. Takamatsu, K. Yamazaki, T. Aikyo, and Y. Sato, "Effective post-BIST fault diagnosis for multiple faults," Proc. 21st Int. Symp. Defect and Fault Tolerance in VLSI Systems, pp.401–409, Oct. 2006.

[3] J. G-Dastidar, D. Das, and N.A. Touba, "Fault diagnosis in scan-based BIST using both time and space information," Proc. Int. Test Conf., pp.95–102, Sept. 1999.

[4] X. Wen, T. Miyoshi, S. Kajihara, L.-T. Wang, K.K. Saluja, and K. Kinoshita, "On per-test fault diagnosis using the X-fault model," Proc. Int. Conf. on Computer-Aided Design, pp.633–640, Nov. 2004.

[5] Y. Sato, S. Hamada, T. Maeda, A. Takatori, Y. Nozuyama, and S. Kajihara, "Invisible delay quality-SDQM model lights up what could not be seen," Proc. Int. Test Conf., no.47.1, Oct. 2005.

[6] C. Liu and K. Chakrabarty, "Failing vector identification based on overlapping intervals of test vectors in a scan-BIST environment," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.22, no.5, pp.593–604, 2003.