PAPER *Special Section on Test and Verification of VLSIs*

On Detection of Bridge Defects with Stuck-at Tests*

Kohei MIYASE[†a)], ***Member***, **Kenta TERASHIMA**[†], ***Nonmember***, **Xiaoqing WEN**[†], **Seiji KAJIHARA**[†], ***Members***, ***and*** **Sudhakar M. REDDY**[††], ***Nonmember***

SUMMARY If a test set for more complex faults than stuck-at faults is generated, higher defect coverage would be obtained. Such a test set, however, would have a large number of test vectors, and hence the test costs would go up. In this paper we propose a method to detect bridge defects with a test set initially generated for stuck-at faults in a full scan sequential circuit. The proposed method doesn't add new test vectors to the test set but modifies test vectors. Therefore there are no negative impacts on test data volume and test application time. The initial fault coverage for stuck-at faults of the test set is guaranteed with modified test vectors. In this paper we focus on detecting as many as possible non-feedback AND-type, OR-type and 4-way bridging faults, respectively. Experimental results show that the proposed method increases the defect coverage.

key words: *defect based testing, test vector generation, test vector modification, bridging faults, fault extraction*

1. Introduction

In the past, it was considered that test sets for stuck-at faults could detect enough defects. However, while a test set for stuck-at faults is still necessary to detect logical faults, this test set is not sufficient to cover non-stuck-at faults. This is because behavior of defects in recent VLSIs is becoming more and more complicated than before. Consequently the goal of test pattern generation need to change in order to cover more complex fault models than stuck-at fault [1]–[6]. Defect based testing (DBT) is a test technology that addressed this issue [1]–[3].

A possible way to detect unmodeled defects is to generate tests that detect each stuck-at fault n times [7]. This method is referred to as n-detection test. Since a stuck-at fault is activated and propagated by n different test vectors in an n-detection test set, defects on the site of the stuck-at fault are likely detected. Thus the n-detection test set has higher defect coverage. Furthermore, the method is easy to be implemented since all we have to do is to generate test vectors for single stuck-at faults. However, the number of test vectors increases in proportion to n, which leads to the increasing of test application time.

When we generate a test set assuming a more complex fault model, the test set may detect a larger number of defects than a test set for stuck-at faults does. The problem of such a way is that the test set size for the complex fault model may become larger than the one for stuck-at faults. It may be possible that test data volume would exceed the limit of tester memory as well as increase test application time if we use or add a test set targeting a complex fault model. For example, in case of bridging faults, the number of the fault is on the order of L^2 for a circuit with L lines [8], [9]. Obviously targeting all the bridging faults in ATPG is extremely expensive for today's large scale circuits.

Detection of many defects with a small size of a test set is really important to achieve higher test quality and to reduce the test cost of test application. However, it is impossible to detect all conceivable defects with a small test set. Therefore it is strongly required to concentrate detection of a small number of defects that are more realistic than others. In case of bridging faults, there are many unrealistic bridging faults, e.g. two lines are not physically close. Therefore extraction of realistic bridging faults has been proposed [2], [10].

Similarly, it is also important to concentrate detection of defects that are more likely to occur than others. Nowadays, information obtained from fault analysis has been fed back to testing, i.e., if we have data on type and location of defects which are likely to occur, we can concentrate detection of such defects.

Recently, methods that allow us to modify test vectors without increasing the size of a test set and without changing stuck-at fault coverage have been proposed [11], [12]. The methods identify don't-care bits in fully specified test vectors. Then they assign logic values to don't-cares in order to satisfy some purposes such as test compression [12]–[14] or test power reduction [15]. However there is no work to assign logic values in order to detect additional bridging faults. In this paper, we propose a novel method to generate a test set that not only guarantees to detect stuck-at faults but also maximizes the detection of faults from another fault model. We can select the fault model from realistic faults extraction [2], [10] or information on defects that likely occur in the manufacturing process. According to such information the proposed method modifies a test set generated for stuck-at faults so as to additionally detect other faults.

Manuscript received April 11, 2007.
Manuscript revised August 6, 2007.

[†]The authors are with the Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology, Iizuka-shi, 820–8502 Japan.
[††]The author is with the Department of Electrical and Computer Engineering, University of Iowa, USA.

*This work is based on, K. Miyase, K. Terashima, S. Kajihara, X. Wen, S.M. Reddy, "On Improving Defect Coverage of Stuck-at Fault Tests," in Proc. of Asian Test Symposium, pp.216–223, Dec. 2005.

a) E-mail: k_miyase@cse.kyutech.ac.jp

DOI: 10.1093/ietisy/e91–d.3.683

In this paper, we concentrate the detection of defects modeled by non-feedback AND/OR-type bridging faults or non-feedback 4-way bridging faults [1], [8].

The proposed method is divided into two steps. In the first step, given a fully specified test vectors for stuck-at faults, we identify as many don't-cares as possible in the test vectors. Even though arbitrary logic values are assigned to the don't-cares, stuck-at fault coverage of the initial test set is still guaranteed. In the next step we assign logic values to don't-cares so that detection of the bridging faults is maximized. The conditions to detect a bridging fault are to detect the stuck-at fault on one of the lines in the pair of bridged lines and to set the opposite value on the other line of the bridged lines or a combination of above conditions. Since the conditions are comprised of conditions to detect a stuck-at fault, we employ a dynamic compaction technique of ATPG [16] to assign logic values to bridged lines. As a result we obtain a test set for stuck-at faults which detects many bridging faults without increasing the number of test vectors. Experimental results for ISCAS benchmark circuits show that the test sets obtained by the proposed method detect more bridging faults than the test sets initially generated for stuck-at faults. We also compare the fault coverage with n-detection test sets.

This paper is organized as follows. In Sect. 2, we review a method of test vector modification and dynamic compaction. We propose a method to improve defect coverage in Sect. 3. In Sect. 4 we describe how to improve coverage of non-feedback AND/OR-type bridging faults or non-feedback 4-way bridging faults with the proposed method. Next, we give experimental results and comparison for benchmark circuits in Sect. 5. Finally, we conclude this paper in Sect. 6.

2. Preliminaries

2.1 Don't-Cares in Test Vectors

Don't-cares (Xs) in test vectors play an important role for testing logic circuits today. Depending on logic values assigned to the Xs, different features can be imparted to the test vectors. For example the existence of Xs generally facilitates test compression [12]–[14]. There are two ways to obtain test vectors that include Xs. One of them is not to specify logic values to Xs (unspecified bits) just after test vectors are generated. The Xs are left unspecified until all test vectors are generated. The drawback of this method is that more test vectors are generated [17], [18] because this method misses accidental detection of faults with random fill or static/dynamic compaction methods [16]. Another method to obtain tests with Xs is identification of entries in a fully-specified test set that can be Xs [11], [12]. The methods of [11], [12] can identify the input values in test vectors that can be set to Xs with reasonable computing time. Furthermore even in compacted test sets these methods show that up to 50% of the inputs in the test sets can be set to Xs. Therefore we can effectively modify test vectors without increase of test vectors to achieve higher defect coverage by filling the Xs appropriately.

2.2 Dynamic Compaction

Dynamic compaction is known as a classic technique to reduce the number of test vectors during ATPG [16]. The concept is to detect as many yet undetected faults as possible by each newly generated test vector. Usually just after test generation for a fault, there are many unspecified values left in the test vector. Dynamic compaction assigns logic values to the unspecified values using ATPG in order to detect other undetected faults.

3. Detection of Desirable Defects

3.1 Overview of the Proposed Method

We propose a method to detect defects modeled by non-stuck-at faults in addition to stuck-at faults. At first the proposed method identify Xs in test vectors, then assign logic values to Xs in order to detect additional non-stuck-at faults. Users of the method can select the additional fault model. Since we only use Xs in test vectors for the additional fault detection in turn we don't add new test vectors, the number of faults additionally detected is limited. Therefore we concentrate a small number of faults to be detected. The faults should be realistic or likely occur. We can obtain such information from fault extraction [2], [10] or information of fault analysis.

Figure 1 shows the flow of the proposed method. Given a test set T for stuck-at faults, we first identify as many positions as possible that can be set to X in order to obtain test set T' with Xs. Before logic value assignment to Xs, we select a fault model and reduce the number of faults that we actually try to detect in the proposed method. Then we assign logic values to the Xs so as to detect targeted faults. Finally we obtain the modified test set T''.

3.2 Target Faults

When we select target faults, there are several concepts. In this section we introduce three concepts to select target faults based on fault extraction, characteristic of defects, and location.

3.2.1 Selection of Faults Based on Fault Extraction

Considering bridging faults, there are many unrealistic bridging faults, e.g. two lines are not physically close. In the case we can know some defects never occur, faults modeling the defects shouldn't be selected. To target only realistic faults in ATPG, methods to extract realistic bridging faults has been proposed [2], [10]. The proposed method in this paper can concentrate such faults.

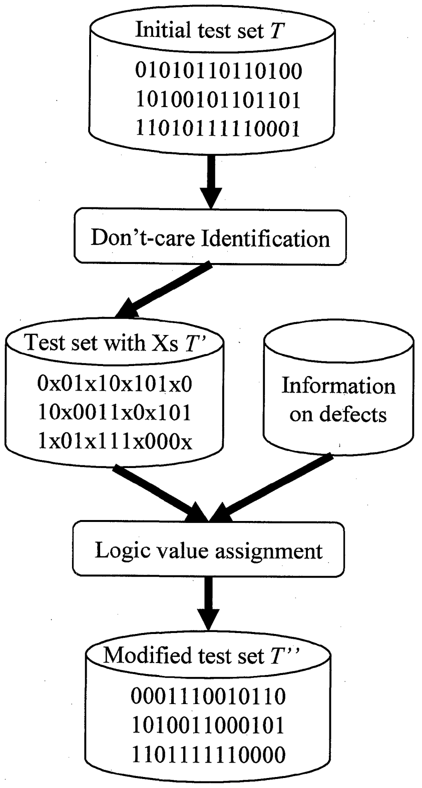


Fig. 1 Flow of test vector modification.

3.2.2 Selection of Faults Based on Characteristic of Defects

When information obtained from fault analysis indicates that defects often occur with a particular behavior, we select fault model representing the defects such as 4-way bridging faults [1], [8], or X-faults [4]. The conditions to detect a stuck-at fault are also necessary conditions for detection of other logical faults. Therefore a test set for stuck-at faults already has potential to detect them.

3.2.3 Selection of Faults Based on Location

Recently sophisticated works of fault location have been proposed [19], [20]. One can use statistics of fault locations for many failed chips to determine sites that are likely to have defects. When information acquired from fault analysis does not indicate the characteristic of defects but only sites where defects likely occur, we modify the initial test set such that stuck-at faults on the suspicious sites are detected multiple times. The concept is similar to n-detection [7], but not all faults are required to be detected n-times. Therefore we can detect fault n-times at the suspicious cites without increase of the test set size.

4. Detection of Non-feedback Bridging Faults

In this paper, we assume that defects which occurred in many circuits under test are bridges and that the behavior of the defects corresponds to non-feedback AND/OR-type bridging faults or 4-way bridging faults. Obviously we can't detect all bridging faults with test vector modification only, since the number of bridging faults in a circuit with L lines is $O(L^2)$. Instead, we focus on a small number of faults, and then we attempt to detect as many bridging faults as possible without increasing the number of tests generated for single stuck-at faults.

4.1 Condition of Bridging Fault Detection

Here we define conditions of detection for non-feedback AND-type bridging fault, non-feedback OR-type bridging fault, and non-feedback 4-way bridging fault.

Definitions:

Definition 1: For bridge between lines (a, b), non-feedback AND-type bridging fault can be detected if stuck-at 0 fault on line a (b) can be detected and logic value of line b (a) is 0.

Definition 2: For bridge between lines (a, b), Non-feedback OR-type bridging fault (a, b) can be detected if stuck-at 1 fault on line a (b) can be detected and logic value of line b (a) is 1.

Definition 3: For bridge between lines (a, b), Non-feedback 4-way bridging fault can be detected if a test set satisfies the following four conditions.
(1) Stuck-at 0 fault on line a can be detected and logic value of line b is 0.
(2) Stuck-at 0 fault on line b can be detected and logic value of line a is 0.
(3) Stuck-at 1 fault on line a can be detected and logic value of line b is 1.
(4) Stuck-at 1 fault on line b can be detected and logic value of line a is 1.

4.2 Logic Value Assignment for Detection of Non-feedback Bridging Faults

After we obtain a test set including Xs, we assign logic values to Xs so as to detect non-feedback bridging faults. Since each condition of detection described in Sect. 4.1 is covered by the condition to detect a stuck-at fault, we can detect a non-feedback AND-type, OR-type, or 4-way bridging fault using the technique of dynamic compaction [16]. In this section, to simplify the explanation we only show a method for non-feedback AND-type bridging faults. We can easily modify the method in order to detect non-feedback OR-type or 4-way bridging fault.

When we assign logic values to Xs to detect a targeted fault, there are three cases. In the following we explain how to assign logic values for each case.

Case 1: In this case a stuck-at 0 fault can be detected on one of the bridging lines, and the logic value is an X on the other bridging line for same test. In order to detect the bridging

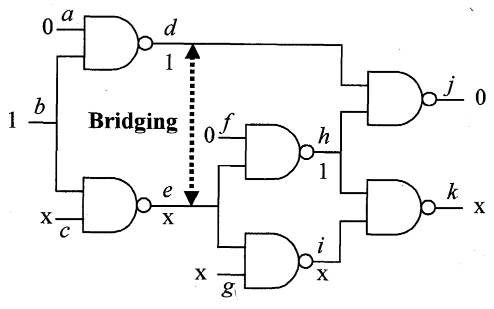


Fig. 2 Logic value assignment for Case 1.

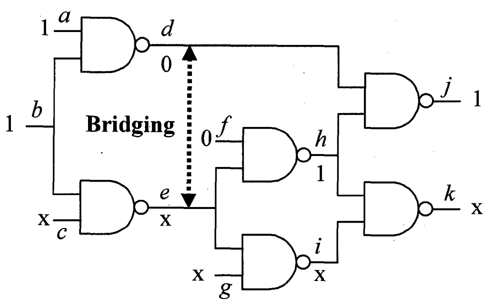


Fig. 3 Logic value assignment for Case 2.

fault, we assign logic values to primary inputs that satisfy the conditions to detect the bridging fault. In this case the detection of a stuck-at 0 on a bridged line is already satisfied. Therefore, we attempt to set 0 on the other line of the bridge. We show an example in Fig. 2. Suppose that there is a non-feedback AND-type bridging fault between lines d and e, and that we have obtained a partially specified test vector $v = \langle a, b, c, f, g\rangle = \langle 0, 1, \mathrm{x}, 0, \mathrm{x}\rangle$ by X-identification. We perform fault simulation with the test vector, and we find out that the stuck-at 0 fault on line d can be detected. In this case we set 0 to line e. This operation corresponds to activation of the fault site in ATPG using dynamic compaction. In order to set 0 on line e, we assign 1 to line c. Finally we obtain test vector $v = \langle a, b, c, f, g\rangle = \langle 0, 1, 1, 0, \mathrm{x}\rangle$ which detects the non-feedback AND-type bridging fault between lines d and e.

Case 2: Suppose that a logic value on one of the bridged lines is 0, and the logic value on the other bridging line is an X. This case satisfies the condition that a logic value is 0 on one of the bridged lines. So, we assign logic values to primary inputs so as to detect the stuck-at 0 fault on the other line. An example is shown in Fig. 3. Suppose that there is a non-feedback AND-type bridging fault between lines d and e, and that we have obtained partially specified test vector $v = \langle a, b, c, f, g\rangle = \langle 1, 1, \mathrm{x}, 0, \mathrm{x}\rangle$ by X-identification. We try to detect stuck-at 0 fault on line e. In order to set 1 to line e for the activation of the stuck-at 0 fault, we assign 0 to line c and we also assign 1 to line g for the propagation of the fault. As a result we obtain test vector $v = \langle a, b, c, f, g\rangle = \langle 1, 1, 0, 0, 1\rangle$ which detects the non-feedback AND-type bridging fault between lines d and e.

Case 3: Suppose that the logic values on both the bridged lines are Xs. In this case, we assign logic values to primary inputs so as to detect a stuck-at 0 fault on one of the bridging lines and to set 0 on the other bridging line. An example is given in Fig. 4. Assume that there is a non-feedback AND-type bridging fault between lines d and e, and that we have obtained a partially specified test vector $v = \langle a, b, c, f, g\rangle = \langle \mathrm{x}, 1, \mathrm{x}, 0, 0\rangle$ by X-identification. We assign 0 to line a, and 1 to line c, and we can detect stuck-at 0 fault on line d and set 0 on line e. As a result we obtain test vector $v = \langle a, b, c, f, g\rangle = \langle 0, 1, 1, 0, 0\rangle$ which detects the non-feedback AND-type bridging fault between lines d and e.

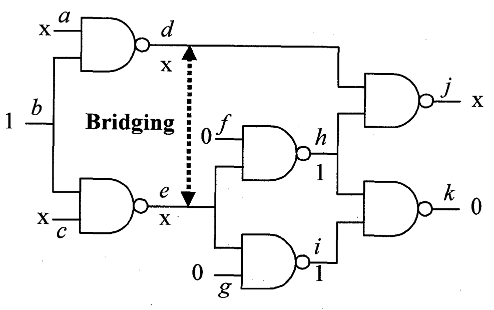


Fig. 4 Logic value assignment for Case 3.

```
Procedure to detect bridge faults (C, T')
Circuit C; Test set with Xs T';
{
    fault_simulation_for_BF(T');
    BF = collect_undetected_BF();

    For each test vector t' in T'{
        fault_simulation_for_SF(t');
        logic_assignment_Case1(BF);
        BF = BF-detected_bf;
        logic_assignment_Case2(BF);
        BF = BF-detected_bf;
        logic_assignment_Case3(BF);
        BF = BF-detected_bf;
        fault_simulation_for_BF(t');
    }
    return modified T''
}
```

Fig. 5 Procedure of logic assignment.

4.3 Procedure of Logic Value Assignments

In Fig. 5, we show the procedure to detect non-feedback AND/OR-type bridging faults with test vector modification. After we obtain a test set T' with Xs, we first perform fault simulation for the targeted non-feedback bridging faults and collect undetected faults. After that, for each test vector t', we perform fault simulation for stuck-at faults. Then we apply three cases of logic value assignments proposed in the previous section in sequence using the results of fault simulation for stuck-at faults. After each assignment, we remove the bridging faults bf which can be detected by the assignments. Finally we obtain the modified test set T'' which detects non-bridging AND/OR-type faults in addition to stuck-at faults. When we detect non-feedback 4-way bridging faults with test vector modification, the procedure of logic value assignment is basically the same as Fig. 5. Since there are four conditions to detect a single 4-way bridging fault, multiple test vectors are required. In the procedure, we remove the bridging faults after the four conditions have been satisfied.

5. Experimental Results

We implemented the proposed method using C programming language on a PC (OS: FreeBSD 6.0-Release, CPU: Intel Xeon 2.0 GHz, Memory 4 GB), and applied to full-scan versions of ISCAS'89 benchmark circuits. For a test set to be modified by the proposed method we used a compacted test set for single stuck-at faults [21]. Since layout information or fault analysis information was not available, we didn't select target faults as described in Sect. 3.2. We picked up bridging faults randomly. However, we focused on a small number of faults to be detected by the proposed method. Then we show the increase of fault coverage when we concentrated to detect the small number of faults. As targeted fault models, we used non-feedback AND-type, OR-type, and 4-way bridging fault, respectively.

Table 1 shows statistics of circuits which used in the experiment. The third column shows the number of test vectors, "%X" shows the percentage of Xs identified in the test vectors. Although the test set is highly compacted, the average of the percentage of Xs is more than 70%. The high percentage of Xs implied that we can assign a lot of logic

Table 1 Circuit information.

circuits	#PIs+PPIs	# test vectors	%X
s5378	214	100	73.3
s9234	247	111	69.2
s13207	700	235	92.0
s15850	611	97	77.3
s35932	1763	12	36.2
s38417	1664	87	74.8
s38584	1464	114	81.2
average			72.0

Table 2 Experimental results for non-feedback AND-type bridging faults.

circuits	# bridge faults	# detected faults		fault coverage (%)		time(sec)
		original	proposed	original	proposed	
s5378	27615	27455	27348	99.4	99.0	53.4
s9234	47420	45680	45987	96.3	97.0	156.0
s13207	69845	69123	68948	99.0	98.7	330.7
s15850	82655	81053	81479	98.1	98.6	446.8
s35932	188300	163295	164542	86.7	87.4	2199.1
s38417	200405	199332	199709	99.5	99.7	2544.1
s38584	200810	194513	194948	96.9	97.1	2578.4
average				96.5	96.8	

Table 3 Experimental results for non-feedback OR-type bridging faults.

circuits	# bridge faults	# detected faults		fault coverage (%)		time(sec)
		original	proposed	original	proposed	
s5378	27615	27428	27393	99.3	99.2	87.9
s9234	47420	45854	46075	96.7	97.2	258.6
s13207	69845	68549	68783	98.1	98.5	568.2
s15850	82655	80824	81169	97.8	98.2	749.7
s35932	188300	172982	174900	91.9	92.9	3743.7
s38417	200405	199497	199945	99.5	99.8	4369.5
s38584	200810	192090	194330	95.7	96.8	3465.0
average				97.0	97.5	

Table 4 Experimental results for non-feedback 4-way bridging faults.

circuits	# bridge faults	# detected faults		fault coverage (%)		time(sec)
		original	proposed	original	proposed	
s5378	27615	17482	18180	63.3	65.8	84.0
s9234	47420	19308	22761	40.7	48.0	467.8
s13207	69845	38869	48085	55.7	68.8	2352.7
s15850	82655	43061	54806	52.1	66.3	2095.5
s35932	188300	52040	56680	27.6	30.1	1102.6
s38417	200405	133281	154225	66.5	77.0	5057.8
s38584	200810	75144	100810	37.4	50.2	13940.3
average				49.0	58.0	

Table 5 Comparison of 4-way brigding faults coverage with N-detection tests.

circuits	proposed method		original (N = 1)		N = 2	
	# test vectors	fault coverage (%)	# test vectors	fault coverage (%)	# test vectors	fault coverage (%)
s5378	100	65.8	100	63.3	206	79.5
s9234	111	48.0	111	40.7	229	59.7
s13207	235	68.8	235	55.7	469	76.9
s15850	97	66.3	97	52.1	206	71.0
s35932	12	30.1	12	27.6	32	46.8
s38417	87	77.0	87	66.5	228	85.3
s38584	114	50.2	114	37.4	291	51.9
average		58.0		49.0		67.3

values to Xs so as to detect additional bridging faults.

Table 2, Table 3, and Table 4 shows experimental results when we applied the proposed method for non-feedback AND-type, OR-type, 4-way bridging fault, respectively. The second column shows the number of non-feedback bridging faults we targeted. The column "# detected faults" shows the number of faults detected by original test sets and test sets modified by our proposed method, respectively. The next column shows the fault coverage of original test sets and modified test sets. The last column shows CPU time in seconds. Every table shows that the proposed method could increase the fault coverage. When we targeted non-feedback AND-type and OR-type bridging faults, the fault coverage increased by only 0.3% and 0.5% in the average. This is because almost all faults can be detected by the original test sets. As for 4-way bridging fault, the fault coverage increased by 9% in the average. For the largest circuit s38584, the fault coverage increased by about 13%. The CPU time for 4-way bridging fault is longer than AND-type and OR-type bridging fault since we try to detect more undetected faults and conditions to detect 4-way bridging fault is more complex.

Table 5 shows the comparison with the N-detection test sets ($N = 2$). While the fault coverage of the proposed method is lower than the one of the 2-detection test sets, the number of test vector of the proposed method is smaller. For the largest circuit s38584, the fault coverage is pretty close. However the number of test vectors of 2-detection test set is more than double the one of test vectors modified by the proposed method. It can be seen that many additional bridging faults can be detected with logic value assignment since the percentage of Xs is very high for the circuit. For the larger circuits, it is said more than 90% of Xs are included in test vectors. Therefore the proposed method could work more efficiently.

6. Conclusions

This paper proposed a method to detect bridge defects with a stuck-at test set. In order to detect fault model other than stuck-at fault, it identifies Xs in test vectors, then assign logic values. Since the proposed method only modified test vector without losing stuck-at fault coverage and without increase of test vectors, there are no negative impacts on test data volume and test application time. In the experimental results we focused on detecting as many as possible non-feedback AND-type, OR-type and 4-way bridging faults, respectively. Experimental results show that the proposed method increased the non-feedback 4-way fault coverage by 9% in the average.

References

[1] S. Chakravarty, A. Jain, N. Radhakrisnan, E.W. Savage, and S.T. Zachariah, "Experimental evaluation of scan tests for bridges," Int'l Test Conf., pp.509–518, 2002.

[2] S. Chakravarty, Y. Chang, H. Hoang, S. Jayaraman, S. Picano, C. Prunty, E.W. Savage, R. Sheikh, E.N. Tran, and K. Wee, "Experimental evaluation of bridge patterns for a high performance microprocessor," VLSI Test Symposium, pp.337–342, 2005.

[3] E.N. Tran, V. Krishna, S. Zachariah, and S. Chakravarty, "Logic proximity bridges," Int'l Test Conf., Paper 31.1, 2005.

[4] X. Wen, H. Tamamoto, K.K. Saluja, and K. Kinoshita, "Fault diagnosis for physical defects of unknown behaviours," Asian Test Symp., pp.236–241, 2003.

[5] B. Kruseman, A. Majhi, C. Hora, S, Eichenberger, and J. Meirlevede, "Systematic defects in deep sub-micron technologies," Int'l Test Conf., pp.290–299, 2004.

[6] S. Irajpour, S.K. Gupta, and M.A. Breuer, "Timing-independent testing of crosstalk in the presence of delay producing defects using surrogate fault models," Int'l Test Conf., pp.1024–1033, 2004.

[7] S.M. Reddy, I. Pomeranz, and S. Kajihara, "Compact test sets for high defect coverage," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.16, no.8, pp.923–930, Aug. 1997.

[8] V. Krishnaswamy, A.B. Ma, and P. Vishakantaiah, "A study of bridging defect probabilities on a Pentium (tm) 4 CPU," Int'l Test Conf., 2001, pp.688–695, 2001.

[9] M. Abramovici, M.A. Breuer, and A.D. Friedman, Digital Systems Testing and Testable Design, Computer Science, Rockville, MD, 1990.

[10] I. Pomeranz and S.M. Reddy, "Using dummy bridging faults to define reduced sets of target faults," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.25, no.10, pp.2219–2227, Oct. 2006.

[11] K. Miyase and S. Kajihara, "XID: Don't care identification of test patterns for combinational circuits," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.23, no.2, pp.321–326, Feb. 2004.

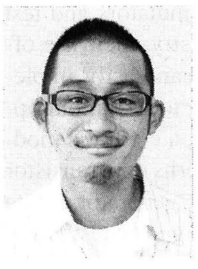
[12] A. El-Maleh and A. Al-Suwaiyan, "An efficient test relaxation technique for combinational & full-scan sequential circuits," VLSI Test Symp., pp.53–59, 2002.

[13] A. Chandra and K. Chakrabarty, "Test data compression and test resource partitioning for system-on-a-chip using frequency-directed run-length (FDR) codes," IEEE Trans. Comput., vol.52, no.8, pp.1076–1088, Aug. 2003.

[14] A. Wurtenberger, C.S. Tautermann, and S. Hellebrand, "Data compression for multiple scan chains using dictionaries with corrections," Int'l Test Conf., pp.926–935, 2004.

[15] X. Wen, Y. Yamashita, S. Kajihara, L.-T. Wang, K.K. Saluja, and K.

Kinoshita, "On low-capture-power test generation for scan testing," VLSI Test Symposium, pp.265–270, 2005.
[16] P. Goel and B.C. Rosales, "Test generation and dynamic compaction of tests," Digest of Papers 1979 Test Conf., pp.189–192, Oct. 1979.
[17] B. Koenemann, C. Barnhart, B. Keller, T. Snethen, O. Farnsworth, and D. Wheater, "A smart BIST variant guaranteed encoding," Asian Test Symp., pp.325–330, 2001.
[18] R. Sankaralingam, R.R. Oruganti, and N.A. Touba, "Static compaction techniques to control scan vector power dissipation," VLSI Test Symp., pp.35–40, 2000.
[19] T. Bartenstein, D. Heaberlin, L. Huisman, and D. Sliwinski, "Diagnosing combinational logic designs using the single location at-a-time (SLAT) paradigm," Int'l Test Conf., pp.287–296, 2001.
[20] D. Lavo, I. Hartanto, and T. Larrabee, "Multiplets, models, and the search for meaning: Improving per-test fault diagnosis," Int'l Test Conf., pp.250–259, 2002.
[21] S. Kajihara, I. Pomeranz, K. Kinoshita, and S.M. Reddy, "Cost-effective generation of minimal test sets for stuck-at faults in combinational logic circuits," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.14, no.12, pp.1496–1504, Dec. 1995.

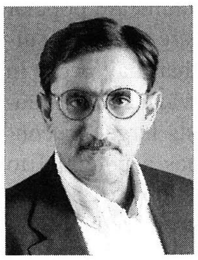


Kohei Miyase received the B.E., M.E., and Ph.D. degrees in Computer Science and Systems Engineering from Kyushu Institute of Technology, Japan, in 2000, 2002 and 2005, respectively. From 2005 to 2007, he was a researcher in Innovation Plaza Fukuoka, Japan Science and technology Agency, Japan. From 2007, he has been an Assistant Professor of Kyushu Institute of Technology. His research interests include test compaction, test compression, design for testability, low power test, and fault diagnosis. He received the Excellent Student Award of the IEEE Fukuoka Section in 2005. He is a member of the IEEE and the IPSJ.

Kenta Terashima received the B.E., M.E. degrees in Computer Science and Systems Engineering from Kyushu Institute of Technology, Japan, in 2005 and 2007, respectively. From 2007, he is joined Fujitsu Limited, Electronic Devices Business Unit. His research interests include VLSI test.

Xiaoqing Wen received the B.E. degree from Tsinghua University, Beijing China, in 1986, the M.E. degree from Hiroshima University, Hiroshima, Japan, in 1990, and the Ph.D degree from Osaka University, Osaka, Japan, in 1993. From 1993 to 1997, he was a Lecturer at Akita University. He was a Visiting Researcher at University of Wisconsin, Madison, U.S.A., from Oct. 1995 to March 1996. He joined SynTest Technologies, Inc., U.S.A., in 1998, and served as its CTO until 2003. In 2004, he joined the Kyushu Institute of Technology, Iizuka, Japan, where he is currently a Professor. His research interests include VLSI test, diagnosis, and testable design. He is a member of the IEEE and the REAJ.

Seiji Kajihara received the B.S. and M.S. degrees from Hiroshima University, Japan, and the Ph.D. degree from Osaka University, Japan, in 1987, 1989, and 1992, respectively. From 1992 to 1995, he worked with the Department of Applied Physics, Osaka University, as an Assistant Professor. In 1996, he joined the Department of Computer Science and Electronics of Kyushu Institute of Technology, Japan, where he is a Professor currently. His research interest includes test generation, delay testing, and design for testability. He received the Young Engineer Award from IEICE in 1997, the Yamashita SIG Research Award from IPSJ in 2002, and the Best Paper Award from IEICE in 2005. Dr. Kajihara is a member of the IEEE and the IPSJ.



Sudhakar M. Reddy received the B.Sc. degree in Physics and B.E. degree in electronics and communication engineering from Osmania University, Hyderabad, India, M.E. degree from the Indian Institute of Science, Bangalore, India and the Ph.D. degree in electrical engineering from the University of Iowa, Iowa City, Iowa. Since 1968, he has been a member of the faculty of the Department of Electrical and Computer Engineering, University of Iowa, where he is currently a University of Iowa Foundation Distinguished Professor. He served as the Chair of the Department from 1981 to 2000. Dr. Reddy has published over five hundred papers in the areas of test and design for test of digital VLSI circuits, coding theory and fault-tolerant computing. He is a Life Fellow of IEEE. He received a Von Humboldt Senior Research Fellowship in 1995 and the first life time achievement award from the VLSI Design Conference in 2000. Dr. Reddy has served twice as a guest editor and as an associate editor of the IEEE Transactions on Computers. He has served as an associate editor of the IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. He has served on numerous program committees of conferences and workshops. He was the technical program chair of the 1989 Fault-Tolerant Computing Symposium.