A Dynamic Test Compaction Procedure for High-quality Path Delay Testing

Masayasu Fukunaga[1], Seiji Kajihara[2], Xiaoqing Wen[2],
Toshiyuki Maeda[3], Shuji Hamada[3], and Yasuo Sato[3]

[1]Fujitsu Ltd., Kawasaki, Japan
fukunaga-m@jp.fujitsu.com
[2]Kyushu Institute of Technology, Iizuka, Japan
{kajihara, wen}@cse.kyutech.ac.jp,
[3]Semiconductor Technology Academic Research Center, Yokohama, Japan
{maeda,hamada.shuji,satoh.y}@starc.or.jp

Abstract - We propose a dynamic test compaction procedure to generate high-quality test patterns for path delay faults. While the proposed procedure generates a compact two-pattern test set for paths selected by a path selection criterion, the generated test set would detect not only faults on the selected paths but also faults on many unselected paths. Hence both high test quality by detecting untargeted faults and test cost reduction by reducing test patterns can be achieved. Experimental results show that the proposed procedure could generate a compact test set that detect many untargeted path delay faults certainly, compared with the static test compaction method previously proposed in [15].

I. Introduction

Path delay fault model [1] is known as a powerful delay fault model to detect defects which lead to the timing violation. Since the path delay fault model models localized as well as distributed excessive delays, test patterns generated for a path delay fault can detect most of other types of delay faults such as gate delay faults [2] on the path. However it is practically impossible to generate test patterns for all paths because there are a huge number of paths in a logic circuit. Therefore we need to select only a subset of paths and target it in test generation.

In path selection for test generation, it is important to select paths which are likely to be faulty, i.e. longer paths. Path selection criteria are categorized into two approaches, which are based on static timing analysis and statistical or dynamic timing analysis. The former selects structurally longest paths in the circuit, and are categorized into two approaches further. One approach is to select *N* longest paths in order of the path length. The length of any selected path is longer than the length of any unselected path. However since the selected paths may not be distributed all over the circuit and may be locally concentrated in a part of the circuit. The other approach is to select a set of paths which contains at least one of the longest paths through each line [3-8]. If we select paths based on this approach, the selected paths would be distributed all over the circuit. However the structurally longest paths may not be actual longest paths in a manufactured circuit due to process variation and/or circuit noise [9,10]. On the other hand, [11,12] tried to select actual longest paths by using statistical or dynamic timing analysis. However it is difficult to know exact delay distribution of manufactured circuits. In addition, the longest paths may be different for each manufactured circuit. Hence actual longest paths cannot be selected necessarily.

A test generation method proposed in [13,14] selects two subsets of paths. For paths in the primary set consisting of structurally longest paths, test patterns are guaranteed to be generated. For paths in the secondary set consisting of next longest paths, fault detection is not guaranteed, but it is considered so as to maximize accidental detection by the test patterns for paths in the primary set.

Recently an idea of test generation for a given set of path delay faults was proposed [15]. A test generation procedure based on the idea can bring an effective solution for two major problems in test generation for path delay faults, namely reducing the number of test patterns and achieving high fault coverage against process variation and noise. In test compaction, while each two-pattern test is generated for more than one fault in the given fault list as well as ordinary test compaction methods, the faults simultaneously detected are selected such that paths with the faults have many cross points. When crossing paths on which there is a common gate are tested simultaneously, non-target paths consisting of partial paths of the paths can be accidentally sensitized and faults on the non-targeted paths can be detected simultaneously too. Note that the accidentally detected faults may not be included in the fault list. Hence even if longer paths in a manufactured circuit are not structurally long paths and not included in the target fault list, the compact test set generated by the method would detect the longer untargeted faults. Although a test compaction procedure based on this idea was given in [15], it is a simple static test compaction [16] and hence the advantage of the idea has not been derived very much.

In this paper we propose a dynamic test compaction procedure based on the concept described in [15]. Dynamic compaction [16] is a well-known compaction procedure with a higher ability of compaction than static compaction. While the proposed procedure selects a primary target fault and secondary target faults from the fault list, the secondary targeted faults are selected such that there are many cross points with the primary fault or other secondary faults

0-7803-9451-8/06/$20.00

processed in the test pattern. Experimental results showed that the size of test sets generated by the proposed dynamic compaction procedure is about six times smaller than that of uncompacted test sets, and 1.5 times smaller than that generated by the static test compaction procedure described in [15]. Though final fault coverage of the test sets by the proposed procedure is lower a little due to the much smaller size of the test sets, certainly detected faults by specified inputs at test generation are increased.

This paper is organized as follows. In Section 2, we explain the method of test compaction for high-quality path delay testing described in [15]. In Section 3, we propose a dynamic test compaction procedure to detect many untargeted path delay faults. In Section 4, experimental results and discussions are given. Finally, we conclude this paper in Section 5.

II. Path Delay Tests with Process Variation Tolerance

In [15] an idea of test generation for path delay faults has been proposed. The compaction method tries to generate not only compact test patterns for a given set of path delay faults but also detect many path delay faults not included in the fault set. The basic idea of this method is to test path delay faults on crossing paths simultaneously. Fig. 1 illustrates a simple example. Assume that there are two paths, PI_1-g-PO_2, and PI_2-g-PO_1 which cross at a gate g. If test patterns are generated for PI_1-g-PO_2 and PI_2-g-PO_1 separately, path delay faults on PI_1-g-PO_1 and PI_2-g-PO_2 may not be tested by the generated test patterns. On the other hand, if paths PI_1-g-PO_2 and PI_2-g-PO_1 are tested by a same test pattern t, then t can test both PI_1-g-PO_1 and PI_2-g-PO_2 simultaneously. When faults on the two paths, PI_1-g-PO_1 and PI_2-g-PO_2 are included in the fault list, we have no need to generate additional test patterns for them. This situation would lead to efficient generation of a compact test set. Even when two paths, PI_1-g-PO_1 and PI_2-g-PO_2 are not included in the fault list, t would enhance fault coverage.

In order to simultaneously test crossing paths on which there is a common gate, the paths must be satisfied with following conditions:

(1) The crossing paths have same transition at the common gate each other.
(2) The transition at the common gate is from the controlling value [22] of the gate to the non-controlling value.

Since paths that are likely to be faulty should be tested, longer paths are selected according to a criterion. Test patterns generated would detect path delay faults on the selected paths certainly if they are testable. However, it is difficult to predict the delay size of a path in manufactured circuits because of process variation or noise. As a result, there remain paths that are more likely to be faulty than the selected ones and the generated test patterns might miss a fault on the paths. Test patterns generated by this method, however, would detect not only faults on the selected paths but also some faults on unselected paths. If the unselected paths whose faults are accidentally detected consist of parts of the selected paths, the length of the unselected paths is relatively long because the selected paths are long. Therefore the test patterns potentially compensate the detection of untargeted faults.

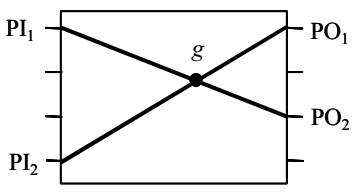


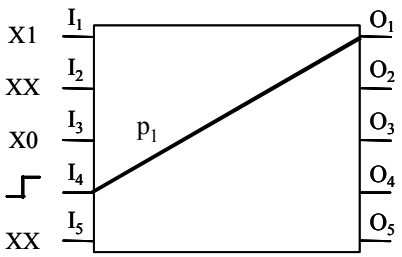
Fig. 1: Two paths with one cross point.

III. The proposed dynamic compaction

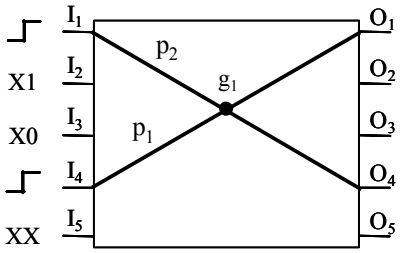
A. Outline of the Proposed Procedure

In this section we propose a dynamic compaction procedure to generate test patterns for a given set of path delay faults which are selected with a criterion of path selection.

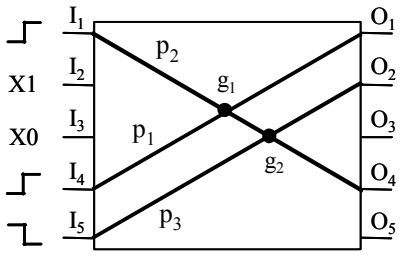
First we pick up an undetected fault from the fault list and generate an initial test pattern with unspecified bits for the fault. We call the fault a primary fault. Next, another



(a) Initial path selection and test generation



(b) Dynamic compaction for *p2*



(c) Dynamic compaction for *p3*

Fig. 2: Outline of the proposed procedure

undetected fault, which is called a secondary fault, is picked up from the fault list. When the secondary fault is selected, we care whether the path of the secondary fault crosses the path of the primary fault or not. Test generation for the secondary fault tries assigning logic values to unspecified bits in the initial test pattern for the primary fault.

Fig. 2 illustrates the outline of the proposed dynamic compaction procedure. Suppose that path p_1, p_2 and p_3 is included in the given fault list. First, we pick up an undetected fault and generates an initial test pattern like Fig. 2 (a), where the path of the primary fault is path p_1, and the generated initial test pattern is (I_1, I_2, I_3, I_4, I_5) = (*X*, *X*, *X*, *0*, *X*), (*1*, *X*, *0*, *1*, *X*). Next we search an undetected fault in the fault list that satisfies the following conditions:

(1) The fault is detected by filling unspecified bits of the initial test pattern.
(2) The path of the fault produces more sensitized paths additionally by crossing the path of the primary fault than any other undetected fault in the fault list.

In Fig. 2(b), assume that the path p_2 is satisfied with above conditions, and that the new test pattern, (I_1, I_2, I_3, I_4, I_5) = (*0*, *X*, *X*, *0*, *X*), (*1*, *1*, *0*, *1*, *X*), is generated by dynamic compaction. Two paths p_1 and p_2 are tested by the test pattern simultaneously, and more two additional paths, I_1-g_1-O_1, I_4-g_1-O_4 can be tested by the test pattern. After that we repeat to select secondary faults and to assign values for the detection of the secondary faults as long as there is a candidate of the secondary fault. Fig. 2(c) illustrates the repetition of the dynamic compaction. Path p_3 is processed after the dynamic compaction for p_2. In this case, more three additional paths can be tested by the test pattern, (I_1, I_2, I_3, I_4, I_5) = (*0*, *X*, *X*, *0*, *1*), (*1*, *1*, *0*, *1*, *0*). Finally, five paths in the given fault list can be tested by one test pattern, and five paths are tested by the test pattern additionally.

B. Selection of a primary fault

In selecting a primary fault, it is important to select one which gives a chance to cross with other paths to be selected as secondary faults. Such a path as a primary fault would satisfy the conditions below.

(1) The path has more off-inputs on each gate having the transition from the controlling value of the gate to the non-controlling value.
(2) The path has more fan-out branches associated with the path.

Fig. 3 shows two examples. Assume that two sub-circuits in Fig. 3 are a part of a logic circuit, and the path p_1, (*a*-*c*-*d* with rising transition at *a*), and the path p_2, (*g*-*i*-*k* with falling transition at *g*) are included in the fault list. The path p_1 can cross a path including *b*-*c*-*e* with rising transition at *b* and a path *b*-*c*-*f* with rising transition at *b*. If the three paths can be tested by the same test pattern, six paths can be tested additionally. On the other hand, the path p_2 can cross only a path *h*-*i*-*k* with falling transition at *b*. No other paths can be crossed with the two paths. Therefore when we select a primary fault to generate an initial test pattern, we had better care the above two conditions.

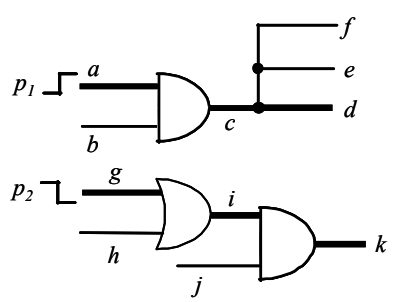


Fig. 3: Deference of the first selected path.

C. Detailed procedure

We implemented a dynamic test compaction procedure shown below. This procedure generates test patterns for faults in a given fault list with heuristics to detect as many untargeted path delay faults as possible by crossing paths in the fault list. Note that, in this procedure, P_{init} is the given fault list, and T_{fin} is the final test pattern set.

Step1: Select a primary fault p from P_{init}, which is undetected and satisfied with two conditions described in the previous section. If P_{init} is empty, go to Step 8. Otherwise go to Step 2.

Step2: Remove p form P_{init}, and go to Step 3.

Step3: If p is untestable, go to Step 1. Otherwise generates the initial test pattern t for p, which has unspecified bits, and go to Step4.

Step4: Search a fault q in P_{init} which is satisfied with below two conditions.

(1) The test pattern for q can be generated by filling unspecified bits of t.
(2) The path of q produces most additional paths by crossing the path p than any other paths in P_{init}.

If there is a path in P_{init}, which are satisfied with these conditions, go to Step 5. Otherwise go to Step 7.

Step5: Generate a new test pattern t' for q by filling unspecified bits of t. Remove q from P_{init}, define a new test pattern t' as new t, and go to step 6.

Step6: If P_{init} is not empty, return to Step 4. Otherwise go to Step 7.

Step7: Output the generated test pattern t to T_{fin}, and go to Step 8.

Step8: If P_{init} is not empty, return to Step1. Otherwise this procedure is finished.

IV. Experimental Results and Discussions

We implemented the proposed procedure of dynamic test compaction using C programming language on a PC (Pentium III Xeon 2GHz, 4GB memory) and applied it to full scan version of ISCAS'89 benchmark circuits. We assume single path delay fault, and refer to untestable paths as non-robust untestable paths in the rest of this paper. However, discussions in this paper are valid for any other sensitizing conditions of path delay faults. We constructed a given fault list such that one of the longest potentially testable paths through each line of the circuit are included. In the construction of a given fault list, we use the partial path sensitization method to identify untestable paths. Note that the length of a path is determined by the number of logic gates on the path. In our experiments, we compared with uncompacted test sets, compacted test sets generated by static compaction procedure described in [15] and compacted test sets generated by the proposed dynamic compaction procedure.

Table 1 shows statistics of each circuit in terms of testable paths and selected paths. The columns of Table 1 give the circuit name, the total number of logical paths i.e. path delay faults, the number of testable paths which can be calculated by ATPG for all logical paths, and the number of selected paths and the number of testable paths out of the selected paths. The given fault list consists of the selected paths. In the selected paths some untestable paths existed except for s35932 because of the incompleteness of the partial path sensitization method in path selection. However most of potentially testable paths were testable.

Table 2 gives the number of generated test patterns. The first column shows circuit names. The second column shows the results of test generation without test compaction where each test pattern is generated for an undetected fault in the fault list and fault simulation is performed for the generated test pattern after random-filling for unspecified bits. The third column shows the result of static test compaction by using the procedure introduced in [15]. The last column shows the results of the proposed test compaction procedure. Table 2 shows that the dynamic test compaction could generate about 6 times smaller test pattern sets than uncompacted test pattern sets on the average. In addition, compared with the test pattern sets generated by static test compaction, the size of test pattern sets generated by dynamic test compaction is about 2/3. This result strongly suggests that the proposed dynamic test compaction procedure generates very small test sets.

Table 3 shows the number of path delay faults which are certainly detected by ATPG. In this result, accidentally detected path delay faults by random filling for remaining unspecified bits are not included. In Table 3, the second column and third column and fifth column show the number of certainly detected path delay faults by ATPG when using generated uncompacted or compacted test patterns. The forth column and the last column show the percentage of increased path delay faults that detected by the compact test patterns generated compared with the uncompacted test patterns. Table 3 shows that the test patterns generated by static compaction can certainly detect about 20% of more path delay faults than that by uncompacted test patterns, and the test patterns generated by dynamic compaction can certainly detect about 26% of more path delay faults than uncompacted test patterns. Therefore the generated test patterns can detect many untargeted path delay faults. In addition, the generated test pattern by dynamic compaction can detect more path delay faults than that by static compaction. Hence we should generate test patterns using dynamic compaction method.

Table 4 shows the coverage of untargeted path delay faults in 10000 longest testable paths. This experiment brings out how generated test sets cover structurally longest paths. In this experiment, we calculated the number of certainly detected paths by ATPG for an uncompacted test set, a compacted test set generated by static compaction and a compacted test set generated by dynamic compaction. Accidentally detected path delay faults by random filling for unspecified bits are not included in these results. In Table 4, the second column shows the number of untargeted path delay fault in 10000 longest testable paths. The third, fifth and seventh columns show the number of detected path delay faults in the untargeted path delay faults for each test pattern. The forth, sixth and eighth columns show the percentage of detected path delay faults for the number of untargeted paths delay faults in 10000 longest testable paths. From Table 4, we can observe that the generated compacted test patterns by dynamic compaction could certainly detect about 10% more path delay faults compared with the

Table 1: The number of selected paths and testable paths.

circuit	#total paths	#testable paths	#selected paths	#testable paths in selected paths
s5378	27,084	21,928	4,170	4,133
s9234	489,708	59,854	5,193	5,159
s13207	2,690,738	476,145	8,792	8,723
s15850	329,476,092	10,782,994	10,027	9,950
s35932	394,282	58,657	28,549	28,549
s38417	2,783,158	1,138,194	28,713	27,496
s38584	2,161,446	334,927	30,891	30,730

Table 2: The size of test pattern sets.

circuit	uncompacted tests	compacted tests (static compation)	compacted tests (dynamic compation)
s5378	920	254	173
s9234	1287	429	296
s13207	1441	511	384
s15850	1855	922	456
s35932	257	32	24
s38417	6799	833	596
s38584	2803	522	347

uncompacted test patterns. Although the generated compacted test patterns by static compaction could certainly detect many untargeted path delay faults, the number of detected paths are less than that by dynamic compaction From this result, since the compacted test patterns can detect many longer paths, even if critical paths are distributed by process variation or electrical noise, real critical path delay faults would detect by the compact test patterns. Therefore, to generate the compacted test patterns by the proposed method is to generate high quality compact test patterns for recent DSM circuits.

Table 5 shows the number of detected faults and fault efficiency of each test pattern set. In this experiment, unspecified bits of each test pattern are filled with 0 or 1 at random. Therefore detected faults include in faults accidentally detected by random-filling. From Table 5, we can observe that uncompacted test pattern sets have higher fault efficiency than the other compacted test sets. Since the number of uncompacted test pattern sets is larger than that of compacted test sets and uncompacted test sets would have many unspecified bits, by filling unspecified bits with 0 or 1 at random, accidentally detected faults would be increase.

Table 3: Certainly detected path delay faults.

circuit	uncompacted tests	compated tests			
		static compaction		dynamic compaction	
	#detected faults	#detected faults	%increase	#detected faults	%increase
s5378	7252	9345	28.86%	10110	39.41%
s9234	10804	14209	31.52%	15136	40.10%
s13207	80001	81961	2.45%	82068	2.58%
s15850	365360	450920	23.42%	483689	32.39%
s35932	34719	37027	6.65%	36307	4.57%
s38417	104912	140404	33.83%	154017	46.81%
s38584	82144	91158	10.97%	92989	13.20%
average			19.67%		25.58%

Table 4: Coverage for 10000 longest testable paths by certainly detected path delay faults.

circuit	untergeted path delay faults	uncompacted tests		compacted tests			
				static compaction		dynamic compaction	
		#detected untergeted	%coverage	#detected untergeted	%coverage	#detected untergeted	%coverage
s5378	7672	663	8.64%	1736	22.63%	2043	26.63%
s9234	9463	1217	12.86%	1499	15.84%	1890	19.97%
s13207	9205	4199	45.62%	4209	45.73%	4186	45.48%
s15850	9734	2752	28.27%	3470	35.65%	3684	37.85%
s35932	4324	1522	35.20%	2163	50.02%	2222	51.39%
s38417	8413	3079	36.60%	4122	49.00%	4356	51.78%
s38584	9563	2717	28.41%	2847	29.77%	2669	27.91%
average			27.94%		35.52%		37.29%

Table 5: Fault efficiency of each test pattern set.

circuit	uncompacted tests		compated tests			
			static compaction		dynamic compaction	
	#detected faults	fault efficiency	#detected faults	fault efficiency	#detected faults	fault efficiency
s5378	17240	78.62%	15992	72.93%	15436	70.39%
s9234	31174	52.08%	27509	45.96%	27118	45.31%
s13207	167612	35.20%	146771	30.82%	138209	29.03%
s15850	1537904	14.26%	1411797	13.09%	1293744	12.00%
s35932	57904	98.72%	46005	78.43%	44172	75.31%
s38417	405046	35.59%	346901	30.48%	332759	29.24%
s38584	180556	53.91%	167841	50.11%	152199	45.44%
average		52.63%		45.98%		43.82%

On the other hand, compacted test pattern sets would have a small number of unspecified bits in itself. Even if certainly detected faults are increased by the proposed dynamic compaction method, the total number of detected faults is not increased so much. However these accidentally detected faults cannot guarantee the quality of generated test pattern, since they may not be detected by another random-filling for unspecified bits. To guarantee the quality of generated test pattern set, many faults should be detected certainly. Therefore generated test pattern sets by the proposed dynamic compaction have high quality to guarantee the circuit operation.

V. Conclusions

In this paper, we proposed a dynamic compaction procedure to test paths with cross points simultaneously so as to accidentally detect many faults which may not be included in the target fault list. The proposed procedure for path delay faults brought improvement of test quality in spite of reduction of test patterns. Experimental results showed that the proposed procedure could generate a compact two-pattern test set and it could detect many untargeted path delay faults efficiently. Our future work is to improve the heuristics algorism to detect more certainly detected paths.

Acknowledgements

This work was supported by the New Energy and Industrial Technology Development Organization (NEDO).

References

[1] G. L. Smith, “Model for delay faults based upon paths,” International Test Conf., pp.342-349, 1985.
[2] Z.Barzilai and B.K.Rosen, “Comparison of AC Self-testing Procedures,” International Test Conf., pp.89-91, 1983.
[3] W.-N. Li, S. M. Reddy, S. K. Sahni, “On Path Selection in Combinational Logic Circuits,” IEEE Trans. on CAD., vol.8, no.1, pp.56-63, 1989.
[4] A. Murakami, S. Kajihara, T. Sasao, I. Pomeranz, and S. M. Reddy, “Selection of Potentially Testable Path Delay Faults for Test Generation,” International Test Conf., pp. 376-384, 2000.
[5] M. Sharma and J. H. Patel, “Finding a Small Set of Longest Testable Paths that Cover Every Gate,” International Test Conf., pp.974-982, Oct. 2002.
[6] Y. Shao, S. M. Reddy, I. Pomeranz, S. Kajihara, “On Selecting Paths to Test in Scan Designs,” Journal of Electronic Testing Theory and Applications, volume 19, pp.447-456, August 2003.
[7] W. Qiu and D. M. H. Walker, “An Efficient Algorithm for Finding the K Longest Testable Paths Through Each Gate in a Combinational Circuit,” International Test Conf., pp.592-601, Sept. 2003.
[8] S. Tragoudas, S. Padmanaban, “A Critical Path Selection Method for Delay Testing,” International Test Conf., pp. 232-241, Oct. 2004.
[9] L.-C. Chen, S. K. Gupta and M. A. Breuer, “High Quality Robust Tests for Path Delay Faults”, in Proc. VLSI Test Symp., pp.88-93, April 1997.
[10] K.-T. Cheng, S. Dey, M. Rodgers, K. Roy. “Test Challenges for Deep Sub-Micron Technologies,” Design Automation Conf., pp.142-149, June 2000.
[11] J.-J. Liou, A. Krstic, Y.-M. Jiang and K.-T. Cheng, “Path Selection and Pattern Generation for Dynamic Timing Analysis Considering Power Supply Noise Effects”, International Conf. on Computer-Aided Design, pp.493-496, Nov. 2000.
[12] J.-J. Liou, A. Krstic, L.-C. Wang, K.-T. Cheng. “False-Path-Aware Statistical Timing Analysis and Efficient Path Selection for Delay Testing and Timing Validation. Design Automation Conf., pp.566-569, 2002.
[13] I. Pomeranz and S. M. Reddy, “Test Enrichment for Path Delay Faults Using Multiple Sets of Target Faults”, Conf. on Design Automation and Test in Europe, pp.722-729, March 2002.
[14] I. Pomeranz and S. M. Reddy, “A Postprocessing Procedure of Test Enrichment for Path Delay Faults”, Asian Test Symposium, pp.448-453, Nov. 2004.
[15] S. Kajihara, M. Fukunaga, X. Wen, T. Maeda, S. Hamada, Y. Sato,"Path Delay Test Compaction with Process Variation Tolerance," Design Automation Conference, pp. 845-850, June 2005.
[16] P. Goel and B. C. Rosales, “Test Generation & Dynamic Compaction of Tests”, in Digest of Papers 1979 Test Conf. , pp.189-192, Oct. 1979.
[17] I. Hamzaoglu, J.H. Patel, “Compact two-pattern test set generation for combinational and full scan circuits,” International Test Conf., pp.944-953, Oct. 1998.
[18] 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. Computer-Aided Design of Integrated Circuits and Systems, vol.14, no.12, pp.1496-1504, Dec. 1995.
[19] I. Hamzaoglu and J. H. Patel, “Test Set Compaction Algorithms for Combinational Circuits,” Intl. Conf. on Computer-Aided Design, pp. 283-289, Oct. 1998.
[20] S. Bose, P. Agrawal, V. Agrawal, “Generation of compact delay tests by multiple path activation,” International Test Conf., pp.714-723, Oct. 1993.
[21] J. Saxena; D.K.Pradhan, “A method to derive compact test sets for path delay faults in combinational circuits,” International Test Conf., pp. 724-733, Oct. 1993.
[22] M. Abramovici, M. A. Breuer, A. D. Friedman, Digital Systems Testing and Testable Design, Piscataway, NewJersey: IEEE Press, 1990.