ISSN 2229-5518

Enhanced Mode of Extended Set of Target Fault Techniques in Single Stuck-at Fault for Fault Coverage in Benchmark Circuits

P. Amutha, C.Arunprasath

Abstract— Considering the full scan benchmark circuit, in which the undetectable single stuck-at faults, tends to cluster in certain areas. This indicates that certain areas may remain uncovered by a test set for single stuck-at faults. The extension to the set of target faults aimed at providing a better coverage of the circuit in the presence of undetectable single stuck-at fault. The extended set of target faults consists of double stuck-at faults that include an undetectable fault as one of their components. The other component is a detectable fault adjacent to the undetectable fault. Test sets that contain several different tests for each fault (n-detection test sets) are expected to increase the likelihood of detecting defects associated with the sites of target faults. This phenomenon is discerned from the gate level description of the circuit, and it is independent of layout parameters. In addition, the clustering is based on the gate level, and remains valid for any layout of the circuit. The fault simulation and test generation for the extended set of target faults is simulated using modelsim along with that the test set compaction is achieved by reseeding method.

Index Terms— Benchmark circuits, fault simulation, stuck-at faults, test quality, unresolved faults.

1 INTRODUCTION

FAULT models used as targets for test generation are expected to guide the generation of tests. Thus, a test set generated for single stuck-at faults is expected to detect defects associated with the sites of stuck-at faults. Test sets that contain several different tests for each fault (n-detection test sets) are expected to increase the likelihood of detecting defects associated with the sites of target faults. When a single stuck-at fault is undetectable, it leaves an uncovered site in the circuit. As we demonstrate later, in benchmark circuits, undetectable single stuck-at faults tend to cluster in certain areas. This implies that certain areas of the circuit remain uncovered, or less covered than other areas, by a test set for single stuck-at faults. We demonstrate that undetectable single stuck-at faults in full-scan benchmark circuits tend to cluster in certain areas of the circuit. We consider the set F1 of uncollapsed single stuck-at faults. This allows us to define clustering, based on structural adjacencies between faults in the circuit, without the need to account for undetectable faults that are missing due to fault collapsing. The fault simulation and test generation experiment described in Section III to full-scan ISCAS-89 benchmark circuits. We defined sets of double faults such that $|F2| \approx p|F1|$, for $p = 1, 2, 4$, and 8. We include in T1, every random test that detects a new fault from F1, when it is simulated. This test set does not detect all the detectable single stuck-at faults in the benchmark circuits considered. After targeting double faults and extending the test set into a new test set T2, we perform fault simulation of T2 to check whether any additional single faults are detected. This phenomenon is discerned from the gate level description of the circuit, and it is independent of layout parameters. Specifically, undetectable single stuck-at faults are introduced by logic synthesis. In addition, our definition of clustering is based on the gate level, and remains valid for any layout of the circuit. To obtain a better estimate of coverage for the circuit in the presence of undetectable faults, and provide a target for improving this coverage, we propose to consider double stuck at faults that include an undetectable fault as one of their components. The other component is a detectable fault adjacent to the undetectable fault. The motivation for considering such double faults is two fold.

2 DETECTION OF A DOUBLE STUCK AT FAULT

The detection of a double stuck-at fault, which includes an undetectable stuck-at fault fi and an adjacent detectable fault fj, provides indirect coverage for the site of fi. Although fi is not detected when it is present alone since it is detected when another, adjacent fault is present. This improves the defect coverage around the site of fi. For example, bridging faults are often not targeted directly by tests used for manufacturing testing due to the difficulty in extracting and generating tests for potential bridges and the larger test set sizes needed. In such cases, one depends on the accidental detection of bridges by tests generated for single stuck-at faults. For bridges occurring in parts of the circuit with undetectable faults, the probability of accidental detection can be improved by adding tests for double stuck-at faults as suggested in this paper.

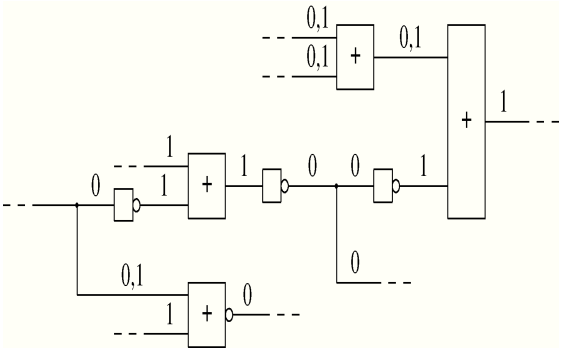

Fig.1. Sub circuit-1

Based on above, if an undetectable fault fi occurs in the circuit without being detected and a second fault fj occurs, a test set that detects fj when it is present alone may not detect the double fault that consists of fi and fj. Considering the double fault explicitly it is possible to ensure that fj will be detected when fi is present. Test sets that detect single stuck-at faults are known to detect large percentages of multiple stuck-at faults consequently, the requirement to detect double stuck-at faults is based on undetectable single stuck-at faults need not add a significant number of tests to a test set for single stuck at faults. Nevertheless, the added tests can be important in covering defects in the parts of the circuit with undetectable single stuck-at faults. It should be noted in this regard that even if a test set detects a fault fj , it is not guaranteed to detect a double fault that consists of fj and a second fault, fi. Such double faults will be targeted here when fi is an undetectable fault and f j is an adjacent detectable fault.

2.1 Clustering of Faults

In this section, we demonstrate that undetectable single stuck-at faults in full-scan benchmark circuits tend to cluster in certain areas of the circuit. We consider the set F1 of uncollapsed single stuck-at faults. This allows us to define clustering, based on structural adjacencies between faults in the circuit, without the need to account for undetectable faults that are missing due to fault collapsing. Let T1 be a given test set that detects all the detectable faults in F1. Suppose that T1 detects a subset D1 of F1. The set U1 = F1 − D1 consists of the undetectable faults in F1. We say that two faults fi and fj are adjacent, if one of the following conditions is satisfied. Let gi be the site of fi and let gj be the site of fj . For a gate G, gi is the input of G and gj is the output of G, For a gate G, gi and gj are inputs of G. gi is a fan-out stem and gj is one of its fan-out branches, or vice versa.For a fan-out stem g, gi and gj are fan-out branches of g. We apply the adjacency relation to pairs of faults in U1 in order to partition U1 into subsets S0, S1, Let U1 = {f0, f1, ..., fm−1}. Initially, we set Si = {fi} for $0 \le i < m$. We then repeat the following process in order to merge pairs of subsets that contain adjacent faults until no additional merging is possible. For every pair of subsets, Si1 and Si2 such that i1 < i2, we check whether Si1 and Si2 contain faults fi1 and fi2, respectively, such that fi1 and fi2 are adjacent. If so, we add the faults from Si2 to Si1, and remove Si2. We computed subsets of adjacent undetectable faults for

TABLE 1

CLUSTRING OF UNDETECTABLE FAULTS

Circuits	Faults	Undet	Subsets	Subset size	
				Ave	Max
S1432	2820	26	11	2.36	6
S5378	10 590	120	16	7.50	30
S9234	18 468	118	52	21.50	46
S13207	26 358	298	46	6.48	42
S15850	31 694	789	76	10.38	73

full-scan ISCAS-89 benchmark circuits. The results are shown in Table I. Under column “Faults” we show the number of uncollapsed single stuck-at faults. Under column “Undet” we show the number of undetectable faults. Under column “Subsets” we show the number of subsets of adjacent undetectable faults. Under column “Subset Size” we show the average and maximum size of a subset of adjacent undetectable faults. From Table I, it can be seen that benchmark circuits have large subsets of adjacent undetectable faults, similar to the one shown in Fig. 1 based on s5378. This is the motivation for attempting to increase the coverage of subcircuits that contain undetectable faults by considering double faults.

3 EXTENDED SET OF TARGET FAULTS

Under this extended set, we define an extended set of target stuck-at faults that consists of double stuck-at faults. The goal of the extension is to provide a target for improving the coverage for sites of undetectable single stuck-at faults. We describe a particular way of selecting the double faults. Other approaches can be used instead to define a larger or smaller subset of double faults. As before, we consider the set F1 of uncollapsed single stuck-at faults, and a test set T1 that detects all the detectable faults in F1. We denote by D1 the subset of F1 that T1 detects. The set U1 = F1 − D1 consists of the undetectable faults in F1. We provide a target for additional coverage for the sites of the faults in U1 by using a set of double stuck-at faults, denoted by F2. To define the set of double faults F2, we use pairs of single stuck-at faults consisting of undetectable faults and detectable faults that are adjacent to them. By using detectable faults that are adjacent to undetectable faults we improve the coverage of areas of the circuit that contain undetectable faults. We avoid undetectable double faults as described later. We consider the faults in U1 one at a time. For every fi ∈ U1, we add double faults to F2 as follows. Let fi ∈ U1 be the fault gi stuck-at ai. We first mark gi and the lines that are adjacent to gi. We use two variables, adj (gj) and adj2 (gj) for every line gj. Initially, we set adj (gj) = 0 and adj2 (gj) = 0 for every line gj. We set adj

(gi) = 1. For every line gj, if gj is adjacent to gi, we set

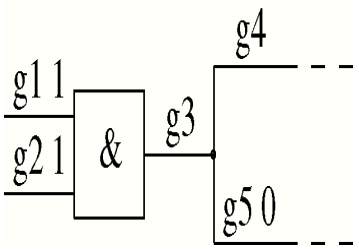

Fig. 2. Sub circuit 2.

adj2 (gj) = 1. We use fi and the lines with adj2 (gj) = 1 to add faults to F2 as follows. For every fault fj ∈ D1, if fj is the fault gj stuck-at aj and adj2 (gj) = 1, we add the fault (fi, fj) or (fj, fi) to F2. We use (fi, fj) if i < j, or (fj, fi) if j < i. If the number of faults added to F2 based on fi is smaller than a constant N, we mark additional lines that are adjacent to the lines already marked. We then add additional faults based on fi using the newly marked lines. This is done as follows. For every line gj, if adj2 (gj) = 1, we set adj (gj) = 1 and adj2 (gj) = 0. This causes all the lines that were already used to define double faults based on fi to have adj (gj) = 1. For every line gj such that adj (gj) = 0, if gj is adjacent to a line gk such that adj (gk) = 1, we set adj2 (gj) = 1. We obtain a new set of lines with adj2(gj) = 1, based on which we add double faults to F2. We repeat this process until the number of faults added to F2 based on fi reaches N.

An undetectable fault gi stuck-at ai is marked with the value ai next to the line name. Our goal is to add N = 5 faults based on the undetectable fault g1 stuck-at 1. The first pass of marking adjacent lines results in adj2 (gj) = 1 for j = 2 and 3. Based on these lines we add to F2 the faults (g1/1, g2/0), (g1/1, g3/0) and (g1/1, g3/1), where g/a is the fault g stuck-at a. We do not add the fault (g1/1, g2/1) since both g1 stuck-at 1 and g2 stuck-at 1 are undetectable. Since the number of faults added to F2 is smaller than N = 5, we set adj (gj) = 1 for j = 2 and 3, and mark lines that are adjacent to them. We obtain adj2 (gj) = 1 for j = 4 and 5. Based on these lines we can add to F2 the faults (g1/1, g4/0), (g1/1, g4/1), and (g1/1, g5/1). When the number of faults added to F2 reaches N = 5, we stop adding faults based on g1 stuck-at 1. To avoid adding undetectable faults to F2, we extend the process as follows. Before starting to add faults based on fi ∈ U1, we find the forward implications of setting gi to the value ai. In the circuit with these implications, we trace the circuit backward from the outputs, and mark the lines that have x-paths to the outputs. An x-path is a path that has unspecified (x) values on all its lines. Suppose that the implications of gi = ai include a value aj on a line gj . Let fj be the fault gj stuck-at aj . The double fault that consists of fi and fj is undetectable, since fj does not affect the value of line gj in the faulty circuit that contains fi, and fi is undetectable. We do not add the fault with components fi and fj to F2. For example, in the circuit of Fig. 3, if g1 stuck-at 0 is undetectable, the double faults (g1/0, g5/0), (g1/0, g6/0), and (g1/0, g7/0) are undetectable. If suppose that line gj carries an unspecified value

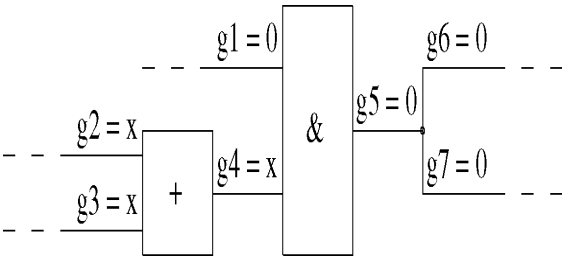

Fig. 3. Sub circuit 3.

when gi = ai is implied, but gj does not have an x-path to an output. This implies that the value of line gj cannot affect the output values in the presence of fi. Since fi is undetectable, the double fault consisting of fi and fj is undetectable.

4 EXPERIMENTAL EVALUATIONS

We applied the fault simulation and test generation experimet to full-scan ISCAS-89 benchmark circuits. We defined sets of double faults such that $|F2| \approx p|F1|$, for p = 1, 2, 4, and 8. The variable p under experimental forms was used for keeping the number of double faults manageable. Since we include in F2 approximately N double faults for every single fault in U1, we obtain $|F2| \approx N|U1|$. To obtain $|F2| \approx N|U1| \approx p|F1|$, we used $N = p \cdot |F1| / |U1|$. We increase the number of faults in F2 in steps by increasing p, and use the test set T2 computed for the previous value as a starting point for the generation of additional tests. We used a simulation-based test generation process for double stuck-at faults. The test generation process is outlined next. Test generation is carried out for every fault (fi, fj) ∈ U2. If a test t is obtained for (fi, fj), t is added to the test set T2. All the faults in U2 are then simulated under t with fault dropping. A test t for a fault (fi, fj) ∈ U2 is computed as follows. We note that one of the components of (fi, fj) is an undetectable fault, and the other is a detectable fault. Let the detectable fault be fi and let the undetectable fault be fj. For fi, the test set T1 contains a test that detects it. Let the test be t. The simulation-based process we use modifies t into a test for the double fault (fi, fj) by complementing the bits of t one at a time. For a circuit with n inputs, let t = t(0)t(1)...t(n−1). Let I = {0, 1, ..., n−1}. In a step of the test generation process, we select an index k ∈ I randomly and remove it from I. We then compute the test $\hat{t} = t(0)...t(k-1)t(k)t(k+1)...t(n-1)$ by complementing the value of input k. We simulate fi and (fi, fj) under $\hat{t}$. If fi or (fi, fj) is detected, we set $t = \hat{t}$ to accept the complemented value of input k. Otherwise, input k retains its previous value in t. If (fi, fj) is detected by t, we stop the modification of t and accept the test. Otherwise, we continue until I = ∅ . We start a new iteration by setting I = {0, 1, ..., n − 1} and repeating the process. This is done up to three times. The current test set is T1 when p = 1, or T2 generated for a lower value of p when p > 1. For example, for s38584, 142 tests in T1 leave 489 undetectable double faults for p = 1.

TABLE II

BRIDGING FAULT SIMULATION

Circuit	Test Set	Tests	Bridge
S1423	T1	26	83.53
S5378	T1	100	89.76
S5378	T2,p=1	101	89.90
S9234	T1	111	81.09
S9234	T2,p=1	113	81.53
S9234	T2,p=2	115	81.88
S9234	T2,p=4	132	83.33
S9234	T2,P=8	143	83.59
S13207	T1	235	87.97
S13207	T2,p=1	236	89.08
S15850	T1	97	86.39
S15850	T2,p=1	103	89.52
S15850	T2,p=2	104	89.72
S15850	T2,p=3	105	89.74

Test generation adds seven tests to T2 for a total of 155 tests, and detects 28 double faults. With p = 2, test generation adds three tests to T2 for a total of 158 tests, and detects 15 double faults. To illustrate that better coverage of the circuit is obtained due to the tests added to T2, we simulated nonfeedback fourway bridging faults under T1, and under the test sets T2 obtained with the various values of p. Simulation of bridging faults was used earlier to demonstrate the effectiveness of n-detection test sets for single stuck-at faults. A four-way bridging fault g/a/h is defined for a pair of lines g and h and a value a ∈ {0, 1}. In the presence of the fault, the value a on h dominates the value of g. The fault is detected by a test that sets h = a and detects the stuck-at a fault on g. The model is referred to as four-way since it associates four faults with every pair of lines, differing in the dominating line and value. For every line g and every value a, we select ten four-way bridging faults randomly by selecting h randomly ten times without repetition. The results of bridging fault simulation are shown in Table III. The first row for every circuit corresponds to T1. Additional rows correspond to T2 with various values of p. The type of the test set is shown under column “Test Set.” Under column “Tests” we show the number of tests in the test set. Under column “Bridge” we show the four-way bridging fault coverage. From Table III, it can be seen that adding tests for double stuck-at faults increases the four-way bridging fault coverage. Thus, the additional tests cover defects that are not covered by the single stuck-at test set.

If two patterns have no conflicting values in any bit position, we can say that the two patterns are compatible. If a certain position of a pattern is 1, and the corresponding position of another pattern is 0, and vice versa, that the two patterns are conflicting, in other words are incompatible. Combining the compatible test patterns will not ifluence the test coverage of F1 uncollapsed single stuck-at faults.

5 THE PRINCIPLE OF RESEEDING

In BIST, the reseeding is a widely-used method to test data compression. As CUT, we suppose a sequential circuit consisting of a combinational part and of n flip-flops, which form a scan chain of equal length. The TPG (Test Pattern Generator) circuit consists of a LFSR with n (n<m) flip-flop cells and an on-chip or off-chip ROM (Read Only Memory) used for storing seeds.

5.1 The method of test pattern grouping and Combination

Usually, for all single stuck-at faults in a CUT, only about 1%~10% of bits in test patterns that generated from an ATPG tool need to specify logical value, while the others are in ‘x’ form. Just because test patterns have the characteristics of low specified bit density and arbitrary 0 and 1 distribution of ‘x’ bit, we find that a lot of test patterns are compatible.

P1: 0 x 1 1 x 1 1
P2: x 0 x 1 0 1 x
P3: 1 x 0 0 x x 1
P4: x 0 x x 1 x x
P5: 0 x x x 1 1 0
P6: x 0 x x x x 1
P7: 0 1 0 1 1 x x

The example of test patterns

The whole test pattern set. Pattern p1 and p2 in Figure2 is an example of compatible patterns. Taking the test pattern set in figure2 for example, all test patterns can be transformed into a graph with 7 vertices, as shown in figure3, then through our-grouping-combination algorithm, 3 test pattern groups can be acquired, namely , {p1,p2,p6}, {p3, p4},and {p5,p7}. Therefore, pattern p1, p2 and p6 can merge into 0011011; pattern p3 and p4 can merge into 10001x1; pattern p5 and p7 can merge into 0101110 LFSR reseeding, the k value should be the minimum. It is known to all, graph-coloring problem is NP-complete [8], so we present a new heuristic algorithm based on Brelaz algorithm. The computational complexity of this algorithm is 2 □(V) , where the variable V is the number of patterns. After applying the heuristic algorithm, the whole test pattern set can be divided into multiple clusters, in each of which test patterns can be combined into one pattern, so we can encode multiple patterns by one LFSR seed. The algorithm for test patterns grouping and combination is explained as follows: create a conflict graph G = (V, E) for test patterns; for (every vertex v){ saturationDeg(v) = 0; uncoloredDeg(v) = deg(v); } place colors c1,c2,…,ck in an order; while (not all vertices in G are colored){ if (all uncolored vertices have the same combine the patterns that correspond to

the same color and calculate LFSR seeds.

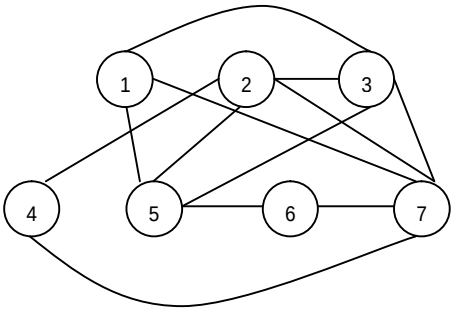

Fig.3. The graph of test patterns

5.2 Experimental results analysis

Experiments were performed on several full-scan versions of the largest ISCAS89 benchmark circuits. We used a LFSR, described with the verilog hardware description language, to generate 10000 pseudorandom patterns, which were able to detect the majority of single stuck-at faults, in each circuit. In order to attain 100% test coverage, we used ATPG tool to generate deterministic test patterns for the rest random-pattern-resistant faults. Our algorithm can successfully group and combine the compatible patterns and calculate corresponding seeds.In, one pattern can be encoded with one seed, in our method, one seed can encode multiple patterns. Table1 shows the quantity of the deterministic test patterns needed for the random-pattern-resistant faults and of the seeds which can be calculated using our method. We can draw a conclusion through comparison that our method has about 30% reduction of the seed number. The number of such faults varies with the circuit and with p. Overall; test generation for the faults in U2 adds tests to T1 in order to detect some of these faults. Although the number of tests is typically small, these tests are important due to the need to provide better coverage for areas of the circuit that may otherwise have reduced coverage. To illustrate that better coverage of the circuit is obtained due to the tests added to T2, we simulated nonfeedback fourway bridging faults [20], [21] under T1, and under the test sets T2 obtained with the various values of p. Simulation of bridging.

Table III.
Experimental results for ISCAS89

Circuit Name	Number of test patterns	Number of seeds after merging	Decreasing percentage
S1423	21	15	28.6%
S1488	18	12	33.3%
S1494	19	13	31.6%
S5378	43	30	30.2%
S9234	78	55	29.4%
S13207	69	50	27.5%
S15850	39	29	25.6%
S38584	57	36	36.8%

6 UNRESOLVED FAULTS

A test generation procedure may not be able to determine for every fault whether it is detectable or undetectable. Such a fault is said to be unresolved. The procedures described in the previous sections can treat unresolved faults as undetectable, and provide additional coverage for them. A possible byproduct of obtaining additional coverage for double faults based on an unresolved fault fi is that a test for fi would be found. We applied the fault simulation and test generation experiment to full-scan ITC-99 benchmark circuits with the same parameters as in Section IV, and with the following changes. To obtain the test set T1, we perform fault nsimulation with fault dropping of F1 under 100 000 random tests. We include in T1, every random test that detects a new fault from F1, when it is simulated. This test set does not detect all the detectable single stuck-at faults in the benchmark circuits considered. After targeting double faults and extending the test set into a new test set T2, we perform fault simulation of T2 to check whether any additional single faults are detected. As p is increased, we only define double faults based on single faults that are still undetected. It can be seen that the additional tests generated for covering sites of undetected single stuck-at faults also help detect additional detectable single faults. This is in addition to providing a better coverage for sites of faults that remain undetected. . Let the detectable fault be fi and let the undetectable fault be fj. For fi, the test set T1 contains a test that detects it. Let the test be t. The simulation-based process we use modifies t into a test for the double fault (fi, fj) by complementing the bits of t one at a time. For a circuit with n inputs, let t = t(0)t(1)...t(n−1). Let I = {0, 1, ..., n−1}. In a step of the test generation process, we select an index k ∈ I randomly and remove it from I. We then compute the test ˆt = t(0)...t(k − 1)t(k)t(k + 1)...t(n − 1) by complementing the value of input k. We simulate fi and (fi, fj) under ˆt . If fi or (fi, fj) is detected, we set t = ˆt to accept the complemented value of input k. Otherwise, input k retains its previous value in t.

7 CONCLUSION

We indicated that undetectable single stuck-at faults in full-scan benchmark circuits are clustering in certain areas. We introduced an extended set of target faults based on double stuck-at faults, whose goal was to provide a target for improving the coverage of these areas. We demonstrated the enhanced mode of extended set of target fault techniques in single stuck at fault are generally provided with the compaction of test set by using the reseeding method.We presented experimental results of fault simulation and test generation in order to demonstrate the extent to which the coverage of areas with undetectable faults can be achieved that our enhanced reseeding method can significantly increase the ratio of test data compression.

REFERENCE

[1] S. M. Reddy, I. Pomeranz, and S. Kajihara, "Compact test sets for high defect coverage," IEEE Trans. Comput.-Aided Design, vol. 16, no. 8, pp. 923–930, Aug. 1997.
2003, pp. 1031–1040..
[2] P. Goel and B. C. Rosales, "Test generation and dynamic compaction of tests," in Proc. Test Conf., 1979, pp. 189–192.
[3] S. Kajihara, T. Sumioka, and K. Kinoshita, "Test generation for multiple faults based on parallel vector pair analysis," in Proc. Int. Conf. Comput.- Aided Design, 1993, pp.
[4] J.-S. Chang and C.-S. Lin, "Test set compaction for combinational circuits," in Proc. Asian Test Symp., 1992, pp. 20–25.
[5] H. Cox and J. Rajski, "A method of fault analysis for test generation and fault diagnosis," IEEE Trans. Comput.-Aided Design, vol. 7, no. 7, pp. 813–833, Jul. 1988.
[6] 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 Design, vol. 14, no. 12, pp. 1496– 1504, Dec. 1995.
[7] I. Hamazaoglu and J. H. Patel, "Test set compaction algorithms for combinational circuits," in Proc. Int. Conf. Comput.-Aided Design, 1998,
[8] S. C. Ma, P. Franco, and E. J. McCluskey, "An experimental chip to evaluate test techniques experiment results," in Proc. Int. Test Conf., 1995, pp. 663–672.
[9] M. Abramovici, M. A. Breuer, and A. D. Friedman, "Testing for single stuck faults," Digital Systems Testing and Testable Design. Piscataway,
NJ: IEEE, 1995, ch. 6, pp. 181–281...
[10] H. Tang, G. Chen, S. M. Reddy, C. Wang, J. Rajski, and I. Pomeranz, "Defect aware test patterns," in Proc. Design Autom. Test Eur. Conf., 2005, pp. 450–455.
[11] I. Pomeranz and S. M. Reddy, "Forming N-detection test sets without test generation," in Proc. ACM Trans. Design Autom., vol. 12. Apr. 2007, pp. 1–18.
[12] M. Abramovici and M. A. Breuer, "Multiple fault diagnosis in combinational
circuits based on an effect-cause analysis," IEEE Trans. Comput., vol. C-29, no. 6, pp. 451–460, Jun. 1980.
[13] Pomeranz, L. N. Reddy, and S. M. Reddy, "COMPACTEST: A method to generate compact test sets for combinational circuits," in Proc. Int. Test Conf., Oct. 1991, pp. 194–203.