2008

Efficient Selection of Observation Points for Functional Tests

Jian Kang
University of Nebraska – Lincoln, jkang@cse.unl.edu

Sharad C. Seth
University of Nebraska – Lincoln, seth@cse.unl.edu

Yi-Shing Chang
Intel Corporation, yi-shing.chang@intel.com

Vijay Gangaram
Intel Corporation, vijay.gangaram@intel.com

Efficient Selection of Observation Points for Functional Tests

Jian Kang and Sharad C. Seth
University of Nebraska – Lincoln
jkang, seth @ cse.unl.edu

Yi-Shing Chang and Vijay Gangaram
Intel Corporation
yi-shing.chang, vijay.gangaram@intel.com

Abstract

The fault coverage of existing functional tests can be enhanced by additional observation points. For a given set of functional tests, this paper proposes an efficient fault-dropping fault simulation method for selecting a subset of observation points at a small fraction of the cost of non-fault-dropping fault simulation. Experimental results on industrial circuits demonstrate the effectiveness of the method in achieving close to optimal results in the size of the selected subset with an order of magnitude less time, without losing achievable coverage. The technique is particularly applicable to industrial designs where fault-simulation times can be prohibitively expensive, even when only a sample of faults is simulated using distributed techniques.

1. Introduction

In high-volume manufacturing of microprocessor, a large number of validation tests are often available. These validation tests exercise the functionality of the design and can be reused in testing the fabricated chips. However the error observation at design outputs is not required during validation hence these tests may not give sufficient fault coverage.

DFT to improve observability is popular on designs that require extensive at-speed functional testing [1], to enhance the fault coverage of existing tests. Since observation point insertion has design costs, optimal selection of signals where observation points can be added is very critical. If we have an efficient method that can optimize selection of observation points to maximize fault coverage achievable with a given validation content, it will significantly reduce manufacturing test development efforts, test application time and debug efforts.

Given a large database of functional tests, there are two problems related to the tapping of it to advantage by the test engineer: (1) selecting a small subset of the available tests that provides a fault coverage close to the maximum achievable by selecting all the test sequences, and (2) as mentioned above, for the selected subset test, choosing a small number of observation points, so as to enhance the fault coverage close to the maximum achievable by scanning out all internal candidates. The complexity of the current designs in gate counts places a premium on finding computationally efficient solutions to these problems. In an earlier paper [2], we proposed a solution to the first problem, demonstrating its effectiveness on large industrial circuits. This paper focuses on the second problem, i.e. the selection of observation points.

Our problem is obviously related to partial scan and test-point insertion. A bibliographic search on these topics reveals over 150 contributions to journals and conferences. However, the bulk of this literature focuses on testability enhancements for patterns generated and applied in a non-functional mode, employing a diverse range of testability assessment measures based on: empirical or symbolic testability [3], cyclic complexity of the circuit's s-graph [4], valid-state analysis using logic simulation of random input patterns [5], and implicit exploration of the machine's state space [6]. Further, because the solutions may impact the circuit performance, many authors have considered timing-driven approaches to testability enhancement [7]. Solving this problem on large industrial circuits using the aforementioned methods causes performance degradation in coverage and the number of selected test-points. Testability measures and ATPG based approaches are test independent and their runtimes are often shorter than those of the simulation based ones. However, the number of cycles unrolled in a sequential ATPG is usually very small, typically in the ranges of 2 - 5 for an industrial circuit, due to memory limitation. Therefore, only the easy-to-detect faults are considered. Moreover, the functional constraints required for ATPG may not be available or can be incomplete, hence, many faults may be detected easily in the scan mode, but not in the functional mode or vice versa. For these reasons, ATPG based approaches provide poor quality solutions to the test-point selection problem, compared to simulation-based methods.

Only a few authors have considered enhancing the testability of a given test sequence [8-12]. Among these, Rudnick et al. [11] address the problem that most closely resembles the one considered here. They improve the testability of at-speed tests by adding probe (observation) points that are further condensed to one or two outputs via XOR trees. However, their method requires non-fault-dropping fault simulation (hereafter abbreviated to non-dropping fault simulation) to build the covering table. Other contributions limit themselves to combinational or full-scan designs. Most of these works follow a common

0-7695-3117-2/08 $25.00 © 2008 IEEE
DOI 10.1109/ISQED.2008.113

approach in the selection of observation points, which can be summarized in the following procedure:

Procedure 1 (Select observation points):

1. Identify hard-to-detect (HTD) faults
2. Trace the HTD faults while performing fault simulation of the test, identify candidate observation points that each fault propagates to, and build a covering table of faults vs. points.
3. Solve the set covering problem for the table to find the set of observation points.

The first step can be carried out by the relatively efficient fault-dropping fault simulation and the third step has a greedy, polynomial-time approximation that produces not too much larger than an optimal set cover [13]. Therefore, it is the second step, which involves the expensive non-dropping fault simulation for large industrial circuits that we target in this paper. We show that, by limiting the number of observation points a fault can propagate to, we can achieve close to optimal results at a much smaller computational effort. We limit our candidate observation points to sequential elements (flip-flop, latch) only, and the problem of bringing the selected observation points to primary outputs without causing excessive number of pins can be solved by existing DFT techniques, such as scanout [1] which uses on-chip hardware to compresses the responses to a signature, or XOR tree [11].

The rest of the paper is organized as follows. Section 2 formulates the observation point selection problem and points out the shortcomings of the existing techniques. Section 3 introduces our approach to solve the problem. Section 4 describes experimental results in support of our method, and Section 5 concludes the paper with pointers to future work.

2. Problem formulation

Fault-Simulation Model: Because the cost of fault simulation is our primary concern here, we start by assuming a general model for fault simulation that covers a range of techniques involving a combination of serial and distributed fault simulation to reduce the cost. An easy and practical way to exploit available independent computing resources is as follows:

1. Partition the set of independent tests across distributed machines.
2. Perform independent fault simulation of the assigned tests on each machine serially using identical initial fault lists.
3. Merge the results after all the machines have completed their work.

As the outer steps do not involve significant computational effort, the middle step will be assumed to determine the timing performance. The model is general in that depending upon the number of available machines it can cover the full range of purely serial fault simulation on one machine, to purely distributed fault simulation in which every machine is assigned one test. In all cases, we can assume the cost to be the maximum time it takes to perform step 2 over all the machines.

The fault simulation model is agnostic to the strategy used for dropping faults as they are simulated. For fault-coverage analysis, it is enough to drop a fault as soon as it is detected at any observation point. This is the well known fault dropping fault simulation, which we will also refer to as 1-detect fault simulation. For observation point selection using Procedure 1, however, 1-detect fault simulation would build a covering table with a lot of missing information, since after a fault is dropped we would not know whether it will get detected at additional test points later on. In order to build a covering table without any missing information, we can use the other extreme, where we never drop a fault during fault simulation. Figure 1 compares the costs of 1-detect vs. non-dropping fault simulation for an industrial-circuit block with 370K gates, simulated on a random sample of 5% faults. The figure shows the run times for four randomly picked functional tests. In all cases, the cost of non-dropping fault simulation is well over an order of magnitude larger. Given that the fault-simulation time rises super-linearly with the number of faults [14], the ratio will be even larger for the complete fault list.

On the other hand, our results show that there is a dramatic difference in the solutions of the covering tables obtained for the two types of simulation: the 1-detect leads to 10,911 observation points while non-dropping fault simulation leads to only 1,262 observation points.

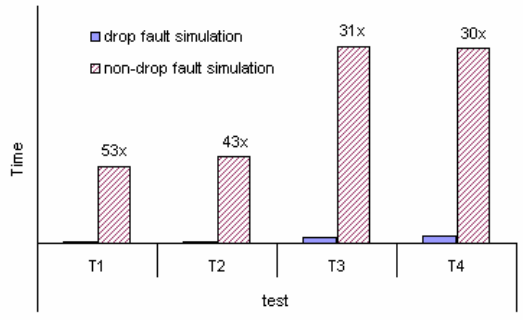

Figure 1. Non-dropping fault simulation takes much more time than dropping fault simulation

3. Proposed point selection method

The above 1-detect fault simulation and non-dropping fault simulation provides two extremes in the trade-off between time and quality. In general, we could choose to have finer-grain trade-offs by defining a generalization of the 1-detect fault simulation.

K-detect Fault Simulation: The k-detect fault simulation involves a fault-dropping strategy, in which a fault is not dropped until it has been detected at at least k observation points.

As the value of k is increased, less number of faults are dropped during simulation, thus we regain more and more of the information missing in the covering table of 1-detect fault simulation.

Figure 2 graphically illustrates the amount of information we would regain for our example industrial circuit. Through non-dropping fault simulation we can determine the number of observation points each fault can propagate to, this is shown in the figure as solid line. The figure shows this distribution for one test; the results for other tests are similar. In the figure, we see that a significant fraction (10%) of faults is detected at only one observation point for the entire test sequence. The percentage increases to 33% faults k = 10, and to 50% faults for k = 20. This means the rows corresponding to these faults already have complete information and we will not gain additional information by k-detect fault simulation at these rows for a higher k value. The gain in additional information after k-detect fault simulation is related to the area below the curve and to the right of the k value. It can be seen that as the value of k increases, the gain drops very sharply.

Table 1 shows the increase in simulation time for increasing value of k, using the same test and fault sample. The first column in the table shows increasing values of k, and the second column shows simulation time. For this test, a fault can reach a maximum of 382 observation points, so any k value larger than that is equivalent to non-drop fault simulation. In the table we use k = 383 to denote this. From the table, it can be seen that the rise in simulation time with the increase of k is not as sharp.

Note that there is a significant difference between our k-detect and the well-known multiple detect (or n-detect) technique [15]. The latter requires that a fault be detected multiple *times* to any observable points so as to increase the defect coverage. Our technique considers fault detection to multiple *points*, to get a balance between fault simulation time and the test point selection quality.

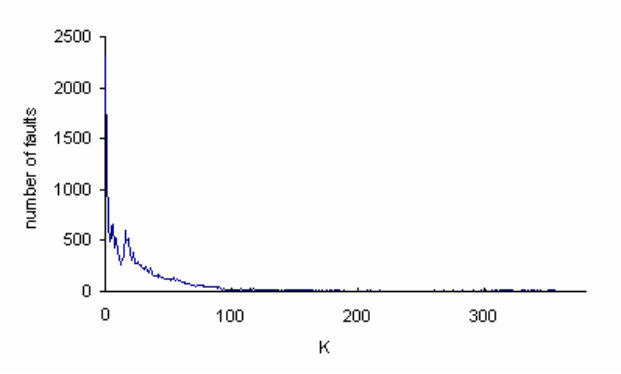

Figure 2. The number of faults detected at k points

Point Selection Method: The above data on real circuit suggests that as we use large values of k, the decrease in the number of selected observation points is going to be non-linear; it should drop faster when k is small. Adding to the trend is the fact that the density of the covering table (the fraction of 1's in the table) quickly approaches the final value with a small value of k and incremental rise in the density only yields incremental improvement in the solutions to the covering problem.

An obvious approach to implementing and testing this hypothesis would be to build the covering tables by running k-detect fault simulation multiple times, each time for a different value of k. However, there is a more efficient way: from the results of k-detect fault simulation, we already have the fault propagation information to reconstruct k-detect fault simulation for all value of $k' \leq k$ with only incremental effort. This leads us to propose the following iterative procedure for selecting observation points:

Table 1. The increase of simulation time with k for a single test

K	Fault simulation time (s)
1	611
2	1280
3	1416
4	1742
5	1469
10	2010
15	2341
20	3098
100	11785
200	22364
383	32424

Procedure 2 (Select observation points using k-detect fault simulation):

1. Select an initial value of k empirically.
2. Run k-detect fault simulation. With this run, get the relationship of number of points vs. k', for all $k' \leq k$.
3. Analyze the curve to see whether we have reached a point of diminishing return.
4. If so, solve the covering table for the current value k and use the covering set as the solution.
5. Otherwise, drop these faults whose number of detection < k (to remove redundancy), increment k by a fixed amount (an adjustable parameter of the algorithm) and go to step 2.

The decision in Step 2 as to when the point diminishing return is reached is analyzed, along with experimental results, in the next section.

4. Experiments and results

To analyze how the threshold k affects the fault simulation time and number of selected points, we adapted an existing industrial concurrent sequential circuit fault simulator in accordance with the point selection method described in the last section and tested it on an industrial design with functional tests. The modifications do not add significantly to the runtime of the fault simulator.

The design is a data path circuit block from an Intel graphics chip, with 370k gates and 675,970 collapsed stuck-at faults. Since this block is buried deep inside the circuit, we assume all faults are hard-to-detect. The 54 functional tests available for the circuit were targeted for increased coverage through additional observation points. These tests have length ranging from 1K to 10K (average 4K) clock cycles. A random 5% fault sample was chosen for fault coverage analysis. However, the results should apply to any larger fault list. The 54 tests give fault coverage of 88% when all observation points are used. After the observation points are selected for existing tests, additional tests could be used to further boost the fault coverage.

4.1. The effect of k on selection quality

In this section, we demonstrate how the chosen value of k affects the number of selected points. This is accomplished by carrying out non-dropping fault simulation for each test, merging the results, then building and solving the covering table for every value of k up to the total number of observation points, which is the largest possible value (this value plus one, which can never be reached for any fault, is equivalent to non- dropping fault simulation; it is denoted here as MAX). The result is shown in Table 2.

Table 2. As threshold k increases, the number of selected points drops non-linearly

K	# points
1	10911
2	5109
3	3214
4	2447
5	2078
10	1493
15	1333
20	1297
100	1260
200	1259
MAX	1262

In the table, the first column shows the k value from 1 to 20, then 100, 200, and MAX. The y-axis shows the number of selected observation points for each k. As can be seen, the result using dropping fault simulation (k=1) is far from optimal, resulting in the selection of nine times more points than at k=MAX. On the other hand, a very sharp drop is evident in the range when the value of k is small. This confirms our analysis in the last section. The non-monotonic decrease at high K values is believed to be caused by the randomness in the greedy algorithm.

For this example, a k value of 10 already provides close to the best possible result: k=MAX leads to 1,262 points, and k=10 leads to 1,493, only an 18% increase.

In practice, the non-dropping fault simulation, assumed in plotting Table 2, would not be practical. Instead, the test engineer might start Procedure 2 with an initial value of, say, k=20 and use the first derivative of the number of points vs. k plot (Figure 3), as an aid to decide if the point of diminishing return is reached. This is indeed the case

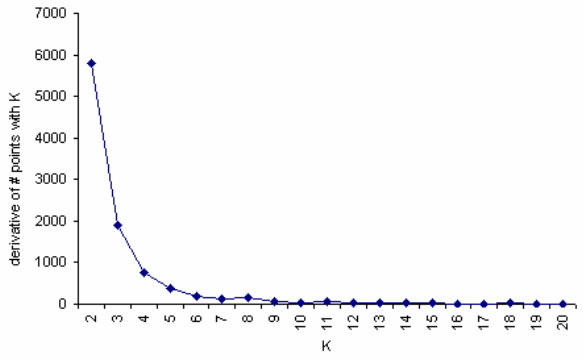

Figure 3. The point of diminishing return is reached very quickly for small values of k

here therefore the engineer will likely decide to forgo running another iteration of the steps of the procedure with a higher value of k.

In the next section we will show that even 20-detect fault simulation affords significant savings in time compared with non-dropping fault simulation.

4.2. The effect of k on simulation time

It is known that dropping (1-detect) fault simulation takes the shortest simulation time, but as we have shown in last section, the point selection provides sub-optimal results. With delayed fault dropping in k-detect fault simulation, for k > 1, the simulation time will increase. In this section we examine the effect of k on simulation time.

Table 3 lists fault simulation times for different values of k. This experiment takes a significant amount of time, especially for large values of k, hence we present the results for only selected values of k.

Table 3. Fault simulation time for all tests

K	Maximum Time (s)
1	3531
2	3851
3	5018
4	5108
5	6546
10	7015
15	7497
20	8417
100	24070
200	42999
MAX	114514

In the table, we report the maximum time over all tests in column 2, assuming distributed simulation of individual tests. It can be seen that there is a significant difference in the fault simulation time between small k values and big ones. Specifically, for k=20, which gave us close to optimal results, it takes less than an order of magnitude smaller amount of time compared with k=MAX. A detailed analysis shows that, with smaller values of k, more faults is dropped in earlier cycles, thus leading to reduced simulation time.

It is worth noting that the simulation times reported in Table 3 correspond to the best-case scenario for distributed fault simulation in which every test is simulated on a dedicated processor. Any other scenario that involved a combination of serial and distributed fault simulation would show even more dramatic differences in the maximum simulation times for k-detect fault simulation vs. non-dropping (k=MAX) fault simulation.

5. Conclusions and future work

In high volume manufacturing designs, both circuit and functional tests are of significant size. Exact method to select the smallest number of observation points requires non-fault-dropping simulation, which is computationally prohibitive. In this paper we have demonstrated that controlling the number of observation points per fault provides a good trade-off between simulation time and result quality: a small threshold value for the control parameter provides close to optimal results with an order of magnitude saving in simulation time. This is very important for real designs where time-to-market is a crucial issue.

While we have indicated the reasons behind the experimental results, future work could build on our preliminary analysis to provide a firm theoretical basis for the method. In particular, it would be helpful to develop accurate models (even if they are circuit-dependent) to predict the timing performance of k-detect fault simulation and the number of points selected by the set-cover algorithm for different values of k. Observation point selection based on k-detect fault simulation may also be combined with testability based techniques for further speed up. We are conducting additional experiments on different designs, to gain insights in furtherance of these goals.

Reference

[1] A. Carbine and D. Feltham, "Pentium(R) Pro processor design for test and debug," in International Test Conference, 1997, pp. 294-303.

[2] J. Kang, S. C. Seth, and V. Gangaram, "Efficient RTL Coverage Metric for Functional Test Selection," in VLSI Test Symposium, 2007. 25th IEEE, 2007, pp. 318-324.

[3] V. Chickermane and J. H. Patel, "An optimization based approach to the partial scan design problem," in International Test Conference, 1990, pp. 377-386.

[4] S. T. Chakradhar, A. Balakrishnan, and V. D. Agrawal, "An exact algorithm for selecting partial scan flip-flops," in Design Automation Conference, 1994, pp. 81-86.

[5] G. S. Saund, M. S. Hsiao, and J. H. Patel, "Partial scan beyond cycle cutting," in International Symposium on Fault-Tolerant Computing, 1997, pp. 320-328.

[6] M. J. Geuzebroek, J. T. van der Linden, and A. J. van de Goor, "Test point insertion that facilitates ATPG in reducing test time and data volume," in International Test Conference, 2002, pp. 138-147.

[7] J. Y. Jou and K. T. Cheng, "Timing-driven partial scan," in International Conference on Computer-Aided Design, 1991, pp. 404-407.

[8] A. J. Briers and K. A. E. Totton, "Random pattern testability by fast fault simulation," in International Test Conference, 1986.
[9] V. S. Iyengar and D. Brand, "Synthesis of pseudo-random pattern testable designs," in International Test Conference, 1989, pp. 501-508.
[10] Y. Savaria, M. Youssef, B. Kaminska, and M. Koudil, "Automatic test point insertion for pseudo-random testing," in International Symposium on Circuits and Systems, 1991, pp. 1960-1963 vol.4.
[11] E. M. Rudnick, V. Chickermane, and J. H. Patel, "An observability enhancement approach for improved testability and at-speed test," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 13, pp. 1051-1056, August 1994.
[12] N. A. Touba and E. J. McCluskey, "Test point insertion based on path tracing," in VLSI Test Symposium, 1996, pp. 2-8.
[13] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, Introduction to Algorithms: MIT Press, Cambridge, MA., 1990.
[14] P. Agrawal, V. D. Agrawal, K. T. Cheng, and R. Tutundjian, "Fault simulation in a pipelined multiprocessor system," in International Test Conference, 1989, pp. 727-734.
[15] S. C. Ma, P. Franco, and E. J. McCluskey, "An experimental chip to evaluate test techniques experiment results," in International Test Conference, 1995, pp. 663-672.