40th Southeastern Symposium on System Theory
University of New Orleans
New Orleans, LA, USA, March 16-18, 2008

N-Model Tests for VLSI Circuits

Nitin Yogi and Vishwani D. Agrawal
Auburn University, Department of Electrical and Computer Engineering, Auburn, AL 36849, USA
yoginit@auburn.edu, *vagrawal@eng.auburn.edu*

We define N-model tests that target detection of faults belonging to N specified fault models. We provide a method for deriving minimal tests using integer linear programming (ILP) without reducing the individual fault model coverage. Any test sequences, deterministic, random, functional, N-detect, etc., can be minimized for the given set of fault models. Stuck-at, transition, and pseudo stuck-at I_{DDQ} faults are used as illustrations. We generate tests using Mentor Graphics FastScan ATPG tool employing a single fault model at a time. A minimized test set for the three fault models is then obtained by solving the proposed combined ILP problem. For s5378 benchmark circuit we achieved about 50% reduction in the number of vectors and 10% reduction in the I_{DDQ} current measurements compared to the originally generated tests. We also propose a reduced complexity ILP approximation.

1. Introduction

Several years ago, in the published results of a Sematech study [8], four types of tests, namely, scan-based stuck-at, scan-based delay, I_{DDQ}, and functional, were examined. A general conclusion was that none of the tests could be dropped. That study has been a subject for numerous discussions [9, 10]. With advances in technology, new fault modes are continuously emerging and the gap between the age-old stuck-at fault model and "realistic defects" continues to widen. On the other hand, the need to minimize test length and test time has never been greater because of complex system-on-chip (SOC) devices. This paper addresses the problem of combining tests that target several fault models into a single compact test.

Test minimization has been a widely researched area. However, most of the published methods will be difficult to apply to multiple fault models. Integer linear programming (ILP) is an effective method of test optimization. Applications of ILP have been reported for separately optimizing vectors for detection of single stuck-at faults [1, 13, 15], N-detection of single stuck-at faults [3], and detection of transition faults [16]. To our knowledge a simultaneous ILP optimization for multiple fault models has not been attempted before.

In this paper we develop an ILP method for multiple-fault models [14]. It is recognized that the complexity of ILP would be too high even for medium size circuits. This problem is overcome by using reduced-complexity ILP variations. We show that such procedures can effectively solve the test minimization problem.

In Section 3, we define a new type of test, the *N-model test*. For these tests the target faults belong to several selected fault models. The rest of the paper develops procedures for generating and optimizing N-model tests using the existing ATPG tools.

2. Overview

Different fault models address different types of realistic defects that can occur in a VLSI chip. For example, stuck-at faults model some types of transistor short defects and transition delay faults model certain timing defects. None of the two are adequate in modeling each other's characteristics completely [16]. This characteristic holds true for other fault models too. This necessitates the use of combined test vectors for different fault models to improve the defect coverage. A simple method of achieving this is to concatenate all test sets together [16], however, that rapidly increases the number of test vectors. Hence we need a test minimization method, which caters to all the considered fault models.

To address this issue, we use an integer linear programming (ILP) problem formulation. An ILP problem [11] is an optimization problem in which we minimize or maximize an objective function for a given set of constraints. The objective function and constraints are both linear in nature. An ILP problem formulation consists of three parts: a set of variables defined as integers, a set of linear constraints on variables and

978-1-4244-1807-7/08/$25.00 ©2008 IEEE.

an objective function defined as a linear expression of those variables which needs to be minimized or maximized. On solving this problem, the solution consists of values assigned to the variables that satisfy the constraints and meet the objective function goal.

3. The N-Model Tests

Definition: For a set of N given fault models, $N \geq 1$, the N-model tests target detection of all faults in the superset of faults for all N fault models.

Typically, a set of fault models may include stuck-at, transition, path-delay, I_{DDQ}, bridging, coupling, etc. The tests may be a combination of functional, verification and random vectors, as well as those generated by targeting one or more fault models. The problem then is to reduce the test set size without affecting the coverage of the N specified fault models. Although an experimental verification is yet to come, we believe such tests will be more effective at uncovering real defects than those generated by the existing ATPG strategies. Besides, any new fault model can be included with the basic prerequisite that we have a fault simulator for it.

The difficulty with multiple fault models is that a fault simulator can deal with only one model at a time. We therefore concatenate all tests and separately simulate the entire set for each fault model, one at a time without fault dropping. We thus obtain a fault dictionary i.e. information about the faults detected by each test vector. Here we consider combinational and scan-inserted sequential circuits only. Hence the test sets are combinational in nature and their order of application is immaterial. Having obtained this fault dictionary, our aim is to select the least number of test vectors that cover all faults belonging to the considered fault models.

Different fault models may require different procedures for testing. For I_{DDQ} faults, after applying a vector, an extra step of I_{DDQ} measurement needs to be carried out. Since this I_{DDQ} measurement is an expensive and time consuming test, it is necessary that their number be kept to a minimum, sometimes even at the expense of slight increase in the total number of vectors to be applied. We formulate this problem as a "combined integer Linear programming (ILP) model" described in the next section.

4. Combined ILP Model

We represent each vector x in the initial vector set by two {0,1} integer variables, t_x and i_x. The variable t_x can take the value 0 (test number x is discarded) or 1 (test number x is selected). Similarly the variable i_x can take the value 0 (no I_{DDQ} measurement on test vector x) or value 1 (I_{DDQ} measurement is performed on test vector x). This definition implies that $i_x = 1$ variables must be a subset of $t_x = 1$ variables.

A separate set of ILP constraints is generated for non-I_{DDQ} and I_{DDQ} faults. If a non-I_{DDQ} fault f_i is detected by test numbers j, m and q, then the corresponding constraint is,

$$t_j + t_m + t_q \geq 1 \tag{1}$$

This constraint means that at least one of the three tests is required to detect the fault f_i. Notice that a test here can be a single vector or a vector pair (for transition delay faults). For an I_{DDQ} fault f_k, that is detected by vectors u, v and w, the constraint inequalities are as follows:

$$t_u + t_v + t_w \geq 1 \tag{2}$$

$$i_u + i_v + i_w \geq 1 \tag{3}$$

$$t_u \geq i_u \tag{4}$$

$$t_v \geq i_v \tag{5}$$

$$t_w \geq i_w \tag{6}$$

The last three constraints mean that I_{DDQ} measurements may be conducted for some subset of all vectors with capability of detecting I_{DDQ} faults. The objective function is,

$$\text{Minimize} \sum_{all\ vectors\ x} (\ t_x + \ W \times i_x) \tag{7}$$

Here the weight factor W specifies the importance to minimize the number of I_{DDQ} measurements. Setting $W = 0$ will minimize the vector count without any regard for I_{DDQ} measurements. Choosing a larger value of W will minimize the number of I_{DDQ} measurements but may lead to an increased number of vectors. This trade off is illustrated by the results in Section 6. The solution of ILP for the above formulated consists of {0,1} integer values assigned to variables t_x and i_x.

Though the ILP gives an optimum solution, it's time complexity is exponential and can become a bottleneck. Results in Section 6 exhibit this characteristic. A linear programming (LP) model is often used in place of

Table 1. Multiple fault model test optimization by ILP method.

Circuit	Type of vectors	No. of FastScan vectors		N-model test optimization by ILP					
		Original	FastScan reduction	$W = 0.1$		$W = 1.0$		$W = 10$	
				No. of vectors	CPU sec.	No. of vectors	CPU sec.	No. of vectors	CPU sec.
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
c1908	All	216	147	79	9.3	81	9.4	84	9.9
	I_{DDQ}	37	26	24		22		20	
c3540	All	519	404	225	5044.2*	226	5046.7*	247	5046.9*
	I_{DDQ}	53	45	40		41		37	
s1238	All	837	683	203	97.1	205	63.5	212	243.1
	I_{DDQ}	70	58	45		43		40	
s1488	All	557	451	175	76.0	176	254.0	187	694.9
	I_{DDQ}	55	45	39		38		33	
s5378	All	796	750	320	2313.9	326	5154.2*	353	5160.9*
	I_{DDQ}	71	70	78		73		64	

*Incomplete optimization due to 5,000 CPU s time limit.

integer linear programming (ILP) model to reduce the CPU time and obtain a result that may still be quite close to the optimum. The worst-case complexity of LP is polynomial-time while that of ILP is exponential. We have devised an "hybrid LP-ILP" method, outlined in Section 5, which is a variation of the published recursive-LP method of test minimization [4].

5. Hybrid LP-ILP Method

Observing the large time-complexity for some ILP runs, we developed an hybrid LP-ILP method. This method solves any ILP problem and can be effectively used for our proposed model described in Section 4.

Once we have formulated the ILP, an LP model is obtained by changing the {0,1} integer variables to real numbers in the range [0,1]. This LP is solved and variables below 0.1 are then rounded to 0 and variables above 0.9 are rounded to 1. The rounded variables are now treated as constants, constraints are updated and the LP solution and rounding processes are repeated until no more variables can be rounded to either a 0 or a 1. Then a much reduced ILP is solved. The results in Section 6 reveal the reduced time complexity of this method.

Since we round variables on both sides, i.e., to 0 and to 1, the procedure can lead to an intermediate solution that may not satisfy all remaining constraints. In such a case, the problem can be solved by the previously proposed recursive-LP algorithm [4], which guarantees a solution. We believe that, whenever a solution is possible, the hybrid LP-ILP will be faster than the recursive-LP.

6. Results

We applied our optimization techniques to two combinational and three scan inserted sequential circuits. We used three fault models: stuck-at, transition and I_{DDQ} faults. We initially generated test vectors for each fault model using the commercial ATPG tool, Mentor Graphics FastScan [7]. For I_{DDQ} faults, the pseudo stuck-at model [6] was used as supported in FastScan. Stuck-at and I_{DDQ} vectors were single pattern tests and transition delay vectors were two-pattern tests, which were considered as a single entity in the ILP model. For combinational circuits, the transition delay tests were simple two patterns tests applied one after another, however, for scan-inserted sequential circuits we obtained both launch-on-shift (LOS) and launch-on-capture (LOC) patterns [5, 12].

The results of minimization over multiple fault models using the ILP method are given in Table 1. After generating the vector sets for the fault models, they were reduced separately for their respective fault models using tools provided in FastScan. Columns 3 and 4 give the number of vectors for original and reduced vector sets. We used the test vectors generated by FastScan (column 3 of Table 1) as our initial test set. We obtained the fault dictionary information for all of

Table 2. Multiple fault model test optimization by hybrid LP-ILP method.

Circuit	Type of vectors	No. of FastScan vectors		N-model test optimization by hybrid LP-ILP					
		Original	FastScan reduction	$W = 0.1$		$W = 1.0$		$W = 10$	
				No. of vectors	CPU sec.	No. of vectors	CPU sec.	No. of vectors	CPU sec.
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
c1908	All	216	147	79	17.88	81	23.74	84	23.23
	I_{DDQ}	37	26	24		23		20	
c3540	All	519	404	225	166.6	226	188.6	248	516.23
	I_{DDQ}	53	45	41		39		34	
s1238	All	837	683	203	48.36	203	61.64	215	63.12
	I_{DDQ}	70	58	46		46		40	
s1488	All	557	451	175	38.33	176	50.33	187	59.99
	I_{DDQ}	55	45	39		39		34	
s5378	All	796	750	320	528.89	326	617.14	353	793.22
	I_{DDQ}	71	70	80		72		63	

those vectors and then used that in the formulation of ILP as mentioned in previous sections.

For our experiments we used the Sun Sparc Ultra-10 machine with 4.0 GB shared RAM. We used AMPL-CPLEX package for ILP [2]. While executing AMPL we specified a CPU time limit of 5000 seconds. The results for various circuits and their vector sets are tabulated in Table 1. FastScan separately generated vectors for each fault model. I_{DDQ} vectors and "All" vectors (including I_{DDQ} vectors) are shown in column 3. Column 4 shows the sum of numbers of vectors obtained by using the minimization tools of FastScan separately for each vector set with its corresponding fault model. Columns 5 through 10 show the results of ILP of Section 4, with increasing emphasis on I_{DDQ} by setting $W = 0.1$, 1 and 10, respectively, in the objective function 7. The trade off between the number of I_{DDQ} measurements and test length is clearly observed.

Reducing Run Times: The results of minimization by hybrid LP-ILP method are given in Table 2. The columns of the table have the same meaning as those in Table 1. The number of minimized vectors and the CPU time complexity can easily be compared between the tables. The hybrid LP-ILP method achieves close to optimum solution in terms of the number of minimized vectors, with an order of magnitude reduction in CPU time as compared to the pure ILP method.

The optimality of the solution achieved the hybrid LP-ILP method can be judged by examining the value of the objective function 7 obtained using pure LP. This value (rounded to the next higher integer) gives a lower bound on the absolute optimum solution as would be given by the ILP.

Table 3 gives a comparison between this lower bound number, pure ILP method and hybrid LP-ILP method for the combined model. Columns 1 and 2 give the circuit name and the different weights W used, respectively. Columns 3 through 5 give the lower bound and the results of the exact ILP and hybrid LP-ILP methods. In most cases we observe that the value of the result obtained by the hybrid LP-ILP method either equals or is close to the exact optimum or lower bound obtained by pure LP. The pure ILP gives non-optimum solution in cases where it had to quit due to CPU time limit. In those cases the hybrid LP-ILP method performs better.

7. Conclusion

The results show an effective reduction in both the total number of vectors and the number of I_{DDQ} measurements required as compared to the original test set. For test set minimization with multiple fault models, the problem arises because fault simulators lack the capability of simultaneously simulating more than one type of fault. The ILP technique allows us to combine the results of separate fault simulations and gives a global optimization of the test set. The ILP method also provides a useful trade off by varying the emphasis between total test length and the number of I_{DDQ} mea-

Table 3. Comparing solutions: hybrid LP-ILP lower bound, ILP optimum and hybrid LP-ILP.

Circuit	Weight W	min (vecs + $W \times I_{DDQ}$ meas.)		
		LP lower bound	ILP solution	hybrid LP-ILP
c1908	0.1	81.4	81.4	81.4
	1	102.25	103	104
	10	276.5	284	284
c3540	0.1	227.94	229*	229.1
	1	257.82	267*	265
	10	499.97	617*	588
s1238	0.1	207.47	207.5	207.6
	1	247.64	248	249
	10	606.04	612	615
s1488	0.1	178	178.9	178.9
	1	211.74	214	215
	10	506.89	517	527
s5378	0.1	326.76	327.8	328
	1	392.28	399*	398
	10	910.68	993*	983

*Incomplete optimization due to 5,000 CPU s limit.

surements. Using a hybrid LP-ILP method to solve an ILP problem significantly reduces the time complexity, while achieving an almost optimum solution.

We hope the practical benefits of the N-model tests in VLSI testing will be explored in the future.

References

[1] P. Drineas and Y. Makris, "Independent Test Sequence Compaction through Integer Programming," in *Proc. Int. Conf. Computer Design*, 2003, pp. 380–386.

[2] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming.* South San Francisco, California: The Scientific Press, 1993.

[3] K. R. Kantipudi and V. D. Agrawal, "On the Size and Generation of Minimal N-Detection Tests," in *Proc. 19th Int. Conf. VLSI Design*, Jan. 2006, pp. 425–430.

[4] K. R. Kantipudi and V. D. Agrawal, "A Reduced Complexity Algorithm for Minimizing N-Detect Tests," in *Proc. 20th International Conf. VLSI Design*, Jan. 2007, pp. 492–497.

[5] R. Madge, B. R. Benware, and W. R. Daasch, "Obtaining High Defect Coverage for Frequency-Dependent Defects in Complex ASICs," *IEEE Design & Test of Computers*, vol. 20, no. 5, pp. 46–53, Sept.-Oct. 2003.

[6] P. C. Maxwell and R. C. Aitken, "I_{DDQ} Testing as a Component of a Test Suite: The Need for Several Fault Coverage Metrics," *Journal Electronic Testing: Theory and Applications*, vol. 3, no. 4, pp. 305–316, Dec. 1992.

[7] Mentor Graphics, *FastScan and FlexTest Reference Manual*, 2004.

[8] P. Nigh, W. Needham, K. Butler, P. Maxwell, and R. Aitken, "An Experimental Study Comparing the Relative Effectiveness of Functional, Scan, I_{DDQ}, and Delay-Fault Testing," in *Proc. 15th IEEE VLSI Test Symp.*, April-May 1997, pp. 459–464.

[9] P. Nigh, W. Needham, K. Butler, P. Maxwell, R. Aitken, and W. Maly, "So What Is an Optimal Test Mix? A Discussion of the SEMATECH Methods Experiment," in *Proc. Int. Test Conf.*, Nov. 1997, pp. 1037–1038.

[10] P. Nigh, D. Vallett, A. Patel, J. Wright, F. Motika, D. Forlenza, R. Kurtulik, and W. Chong, "Failure Analysis of Timing and IDDq-only Failures from the SEMATECH Test Methods Experiment," in *Proc. Int. Test Conf.*, Sept. 1999, pp. 1152–1161.

[11] J. Noyes and E. W. Weisstein. Linear Programming. From MathWorld–A Wolfram Web Resource. *http://mathworld.wolfram.com/LinearProgramming.html.*

[12] L. Xijiiang, R. Press, J. Rajski, P. Reuter, T. Rinderknecht, B. Swanson, and N. Tamarapalli, "High-frequency, at-speed scan testing," *IEEE Design & Test of Computers*, vol. 20, no. 5, pp. 17–25, Sept-Oct. 2003.

[13] N. Yogi and V. D. Agrawal, "High-Level Test Generation for Gate-Level Fault Coverage," in *Proc. 15th IEEE North Atlantic Test Workshop*, May 2006, pp. 65–74.

[14] N. Yogi and V. D. Agrawal, "Optimizing Tests for Multiple Fault Models," in *Proc. 16th IEEE North Atlantic Test Workshop*, May 2007, pp. 57–63.

[15] N. Yogi and V. D. Agrawal, "Spectral RTL Test Generation for Microprocessors," in *Proc. 20th International Conf. VLSI Design*, Jan. 2007, pp. 473–478.

[16] N. Yogi and V. D. Agrawal, "Transition Delay Fault Testing of Microprocessors by Spectral Method," in *Proc. 39th Southwestern Symp. on System Theory*, Mar. 2007, pp. 283–287.