

Dynamic N-Detect Patterns Based on Equivalent Faults

Paul Reuter and Yu Huang

Mentor Graphics, Inc. 300 Nickerson Rd., Marlborough, MA, 01752

Contact Author: paul_reuter@mentor.com, Phone: 508-303-5669, Fax: 508-480-0882

Abstract

This paper investigates a method of using equivalent fault metrics to optimize N-detect scan ATPG patterns. The introduction goes over the motivation for using N-detect scan patterns and why dynamic N-detect patterns are useful. The next section discusses what equivalent fault metrics are and why equivalent fault metrics might apply to dynamic N-detect scan patterns selection. We then present experimental results covering the pattern count and test effectiveness of the dynamic N-detect algorithm.

1. Introduction

Scan based ATPG is the predominant method to create manufacturing test patterns for digital integrated circuits. The goal is to produce a test pattern set that screens for manufacturing defects in order ensure that outgoing parts meet a certain quality level. The use of the single stuck-at fault model to produce scan ATPG patterns is no longer sufficient for modern VLSI designs [1]. Various methods to improve defect coverage have been proposed and used, including transition faults, path-delay, iddq faults, bridging faults, and N-detection.

N-detect scan patterns are stuck-at ATPG patterns where each fault is detected by at least N different scan patterns. N-detect scan patterns have been shown to improve defect coverage in VLSI designs by activating un-modeled defects [2][3]. Various methods are available for creating N-detect scan patterns. Because of the large size of the resulting N-detect scan patterns, various methods have also been proposed for optimizing N-detect scan patterns [4][5][6].

In this paper, we propose a method for optimizing N-detect scan patterns by dynamically applying the N-detection such that the detection number varies with different fault sites. Multiple patterns are only generated for fault sites where the N-detect patterns will be most effective. In this scenario, each fault site f

has its own detection number N_f . N_f represents the lower bound for the number of scan patterns used to detect fault f . The problem we have to solve would be to predict which fault sites will most effectively catch un-modeled defects with minimum multiple scan patterns. That is to say, we need find optimal N_f for each fault f .

2. Review of Traditional N-detect and N-detect Optimizations

Several N-detect ATPG algorithms were illustrated in [12], [3] and [7]. Although they may have differences in the implementation details, they do share some commonalities described as follows:

- (1) N-detect ATPG is an iterative procedure. For each run, a pattern set will be generated to target a set of faults.
- (2) N-detect ATPG is an incremental procedure. For each run, a newly generated pattern set will be added to the pattern list. No compaction or pattern dropping is performed.
- (3) The fault simulation is modified to drop faults from the fault list only after they are detected at least N times.

Algorithms of this type can be called traditional N-detect ATPG algorithms. In addition to the traditional N-detect ATPG algorithms, several methods have been proposed to optimize the N-detect patterns [4] [5] [6]. The objectives of these optimization techniques is to find the minimum pattern set P such that each fault F is detected at least N times if its N-detect pattern exists, or each fault F is detected as many times as possible if its N-detect pattern does not exist. These optimization techniques do not change the number of detects needed for each fault site, instead, they use static compaction and heuristics to re-order and minimize the number of generated patterns.

3. Equivalent Faults and Dynamic N-detect

In order to dynamically vary the number of N-detect patterns to create for each stuck-at fault site, we need to analyze some metric related to the complexity of the design near the fault site. Ideally, we would like to analyze physical layout information about the design in order to identify nodes in the design that are prone to certain types of defects, and then target N-detect patterns at these nodes to increase the effectiveness. Using the layout information can be compute intensive and time consuming [7].

In this paper, we propose to use equivalent faults to provide some indication of circuit complexity. Equivalent faults and collapsing equivalent faults are a common component in all ATPG programs and fault simulation programs. The algorithms and methods used are mature [8][9][10]. Equivalent faults are faults that will be detected by the same exact test. There is no way to detect one of the faults without catching them all. Collapsing these equivalent faults into a single fault site is useful for improving the performance of ATPG and fault simulation programs without changing the functionality of the program. While collapsing equivalent faults into a single fault site does not change the content of ATPG patterns or the fault coverage of a set of patterns, the number of equivalent faults collapsed into a single fault site might be a metric which can tell us something useful about the design topology.

While all faults that are collapsed into a single equivalent fault are indeed detected by the same test set, the number of equivalent faults collapsed into a single fault site does indicate a measurement of the circuit complexity represented by that fault site. Because of this, the number of equivalent faults at a fault site can be used as a metric to determine the effectiveness of N-detect patterns for detecting unmodeled defects.

Another way to look at this is to assume that we need to do N-detect pattern generation on the un-collapsed fault set. For example, if we set N to be 2, we intend to create 2 independent patterns to detect each un-collapsed fault. If we then collapse the faults and also add up the number of independent test patterns required for each equivalent fault being collapsed, we get a new number N_c which corresponds to the number of test patterns required for each collapsed fault.

The algorithm we propose is that for each stuck-at fault f , that has X_f equivalent faults, we will create $N_f =$

$(X_f + 1)$ N-detect patterns for this fault. This is shown pictorially in Figure 1.

The advantage of using equivalent fault information for determining the dynamic number of N-detect patterns for a specific fault site is that the equivalent fault number is a metric that is already computed and stored by almost all ATPG programs. There is no extra computation needed to gather this information. The disadvantage is that this metric will only be a rough approximation of the number of unmodeled defects that may be caught by applying N-detect patterns.

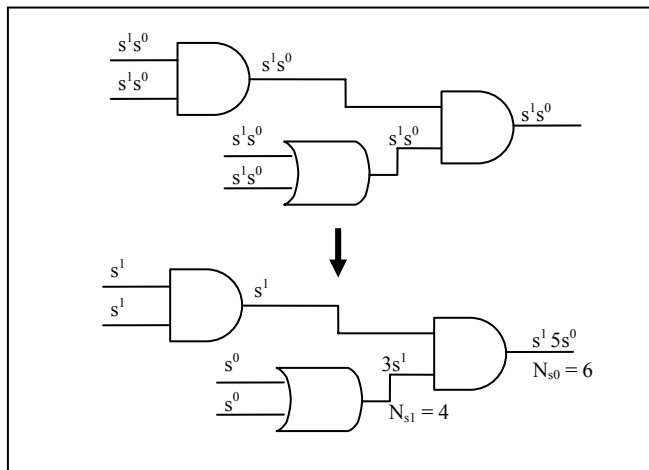


Figure 1: Equivalent Fault Collapsing

4. Experimental Results

This section presents experimental results to compare the pattern count and test effectiveness between the proposed dynamic N-detection ATPG and the traditional N-detection ATPG.

We use one industrial design to run the experiments. Some information about this design is listed in Table 1.

Table 1: Design Information

# gates	3.1 M
#PIs	93
#POs	108
# flip-flops	19192
# scan chains	20
# extracted bridging net	47388

The first 5 rows show the number of gates, the number of primary inputs, the number of primary

outputs, the number of flip-flops, and the number of scan chains respectively. In the last row of Table 1, we show the number of extracted bridging net. We use Calibre™ to extract the bridging net pairs from the layout (GDSII file) of the design. We define 5 types of bridging as follows, which are also explained in Figure 2. More details about the bridging features can be found in [11].

- (1) Type 1: Side-to-side bridge
- (2) Type 2: Corner-to-Corner bridge
- (3) Type 3: End-of-line bridge
- (4) Type 4: Via-to-via bridge
- (5) Type 5: Side-to-side bridge over wide metal in layer below

The pairs of nets as illustrated in Figure 2 have specific characteristics that make them vulnerable to bridging. In this experiment, we use bridging defects as surrogate to validate the proposed ATPG algorithm. The *static bridge fault model* is used by FastScan™ and TestKompress™ to test against potential 2-way bridge defects (net pairs) extracted from the design's layout.

This model uses a 4-way dominant fault model that works by driving one net (dominant) to a logic value and ensuring that the other net (victim) can be driven to the opposite value.

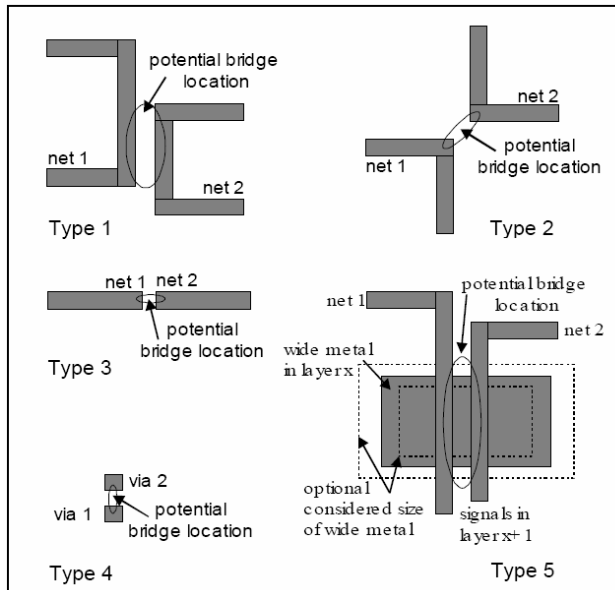


Figure 2: Extracted Bridge Features

Let sig_A and sig_B be two nets in the design. If sig_A and sig_B are bridged together, the following faulty relationships may exist:

- (1) sig_A is dominant with a value of 0 (sig_A=0; sig_B=1/0)
- (2) sig_A is dominant with a value of 1 (sig_A=1; sig_B=0/1)
- (3) sig_B is dominant with a value of 0 (sig_B=0; sig_A=1/0)
- (4) sig_B is dominant with a value of 1 (sig_B=1; sig_A=0/1)

Next we use FastScan™ or TestKompress™ to create N-detection test patterns that target traditional stuck-at faults. Then we simulate the generated patterns against the 47388 net pairs extracted from the layout. For each net pair, we use the static fault model to target all four faulty relationships. The fault coverage can be used to reflect the bridging defect coverage. The experimental results are shown in Table 2.

Table 2: Pattern Count and Fault Coverage Comparison

		# patterns	Fault Coverage (coll.)
<i>Dynamic N-detection</i>		3110	74.15%
<i>Traditional N-detection</i>	<i>N=2</i>	3023	71.42%
	<i>N=3</i>	4263	71.56%
	<i>N=4</i>	5494	71.69%

In the second row of Table 2, we show the pattern count and fault coverage of the patterns generated by the proposed dynamic N-detection algorithm. That is to say, for each targeted stuck-at fault f , if it has X_f ($X_f \geq 0$) equivalent faults, we drop the fault from the targeted fault list only if the fault has been detected ($X_f + 1$) times. From the 3rd row to the 5th row, we set $N = 2, 3, 4$ respectively and run traditional N-detection ATPG. That is to say, we drop the fault from the targeted fault list if the fault has been detected N times, no matter how many equivalent faults it has.

From the results, we can see that:

- (1) The proposed dynamic N-detection can achieve higher fault coverage for the targeted bridging pairs than the traditional N-detection ATPG. It only used 3110 patterns to detect 2.46% more faults than using 5494 tradition N-detection patterns when setting $N=4$.

- (2) When using the traditional N-detection ATPG, increasing N will generate a large number of additional patterns. However, the additional N-detection patterns may not lead to big enhancement of the fault coverage when targeting bridging defects for this particular design. Further investigation is necessary in future.

5. Future Work

Further experiments need to be done both with other industrial designs and with other ratios of the number of N-detect patterns with the number of equivalent faults. Experiments should also be done with other metrics that can be used as a measure of circuit complexity. These metrics could be used alone or in combination with equivalent fault counts to see if the dynamic N-detect patterns can be further refined. Based on these experiments, we may be able to create a computationally efficient method to create dynamic N-detect patterns that are much smaller in pattern count than traditional N-detect patterns yet may achieve a higher coverage of un-modeled defects. In other words, we hope to find the most computationally efficient method to achieve the optimal N_f for each fault site f , and then create dynamic N-detect patterns using these optimal N_f numbers.

6. Conclusions

In this paper we reviewed the motivation for using N-detect ATPG scan patterns and introduced the concept of dynamic N-detect patterns. We then introduced equivalent fault information as a metric that can be used to determine the effectiveness of N-detect patterns for detecting un-modeled defects.

We have shown that dynamic N-detection patterns based on equivalent fault counts can achieve as high or higher bridging fault coverage as traditional N-detection patterns can. This can be achieved with little extra computational effort, and this method could easily be combined with other techniques [4] used for optimizing N-detect patterns.

One limitation to this method is that it applies to N-detect stuck-at ATPG scan patterns where equivalent fault collapsing is used. This algorithm may find limited use with other fault models. For example, when creating transition fault ATPG N-detect patterns, fault collapsing does not occur. Another metric for controlling the dynamic number of detects per fault site, N_f when using other fault types would be needed. Future experiments are needed to investigate other circuit metrics.

References

- [1] E. J. McCluskey and Chao-Wen Tseng, "Stuck-fault tests vs. actual defects", *Proceedings IEEE Int. Test Conference*, 2000, pp. 336-342.
- [2] M. E. Amyeen, S. Venkataraman, A. Ojha and S. Lee, "Evaluation of the Quality of N-Detect Scan ATPG Patterns on a Processor," *Proceedings IEEE Int. Test Conference*, 2004, pp. 669-678.
- [3] B. Benware, C. Schuermyer, S. Ranganathan, R. Madge, N. Tamarapalli, K.-H. Tsai and J. Rajski, "Impact of Multiple-Detect Test Patterns on Product Quality," *Proceedings IEEE Int. Test Conference*, 2003, pp. 1031-1040.
- [4] Y. Huang, "On N-Detect Pattern Set Optimization," *Proceedings 7th International Symp. On Quality Electronic Design*, Mar. 2006, pp. 445-450.
- [5] K. R. Kantipudi and V. D. Agrawal, "On the Size and Generation of Minimal N-Detection Tests," *Proceedings 19th International Conf. VLSI Design*, 2006, pp. 425-430.
- [6] S. Lee, B. Cobb, J. Dworak, M. R. Grimaila, and M. R. Mercer, "A New ATPG Algorithm to Limit Test Set Size and Achieve Multiple Detections of all Faults," *Proceedings Design Automation and Test in Europe Conf.*, 2002, pp. 268-274.
- [7] R.D. Blanton, K.N. Dwarakanath and A.B. Shah, "Analyzing the Effectiveness of Multiple-Detect Test Sets," in Proc. of International Test Conference, pp.876-885, Oct. 2003.
- [8] A. V. S. S. Prasad, V. D. Agrawal, and M. V. Atre, "A New Algorithm for Global Fault Collapsing into Equivalence and dominance Sets", *Proceedings Int. Test Conference*, 2002, pp. 391-397.
- [9] M. Abramovici, M. A. Breuer, and A. D. Friedman, *Digital Systems Testing and Testable Design*, Piscataway, New Jersey: IEEE Press, 1990.
- [10] W.-T. Cheng, and T. J. Chakraborty, "Gentest: An Automatic Test Generation System for Sequential Circuits," *Computer*, vol. 22, no. 4, pp. 43-49, Apr. 1989.
- [11] M. Keim, N. Tamarapalli, H. Tang, M. Sharma, J. Rajski, C. Schuermyer and B. Benware, "A Rapid Yield Learning Flow Based on Production Integrated Layout-Aware Diagnosis", ITC, 2006. Paper 7.1.
- [12] M.R. Grimaila, S. Lee, J. Dworak, K. M. Butler, B. Stewart, H. Balachandran, B. Houchins, V. Mathur, J. Park, Li-C. Wang and M. R. Mercer, "REDO – Random Excitation and Deterministic Observation – First Commercial Experiment," VTS, 1999, pp. 268-274.