Adaptive Test Optimization through Real Time Learning of Test Effectiveness

Baris Arslan[1,2] and Alex Orailoglu[1]
{barslan, alex}@cs.ucsd.edu

Computer Science and Engineering[1]
University of California, San Diego
La Jolla, CA 92093

Qualcomm Incorporated[2]
5775 Morehouse Drive
San Diego, CA 92121

***Abstract*—Production test suites include a large number of redundant test patterns due to the inclusion of multiple test types with overlapping defect detection and the use of simple fault models for test generation. Identification and elimination of ineffective test patterns promises a significant reduction in test cost. This paper proposes a test framework that learns, without extensive data collection and at no additional test time, the effectiveness of individual test patterns during production testing by getting defect detection feedback from a dynamic test flow. The proposed technique is further capable of adapting to changes in the underlying defect mechanisms by tracking the defect detection trend of test patterns.**

I. INTRODUCTION

The number of test patterns and individual test pattern length continues to increase as the size of integrated circuits (IC) gets larger, boosting test time and consequently test cost. In addition to ever increasing circuit sizes, the need for new test types constitutes one of the main drivers of test cost increase. As the effectiveness of traditional test types such as stuck-at and IDDQ diminishes in DSM defect detection, the research community and industry alike are shifting their focus to new test types. Various new fault models such as N-detect [1] and gate-exhaustive [2], numerous flavors of delay test methods such as timing-aware [3] and faster-than-at-speed [4,5] and defect-oriented test methods have been proposed to keep defective parts per million (DPPM) within acceptable limits. As manufacturing cost continues to shrink due to higher levels of integration and process scaling and as the test cost continues to rise due to the increase in circuit complexity and test types, test continues to increase its share of final production cost. Increasingly, semiconductor manufacturers are squeezed by the simultaneous demands of acceptable test cost and satisfactory DPPM levels.

In structural digital logic testing, scan compression techniques have received a tremendous amount of attention in the last decade to reduce test cost. Numerous combinational and sequential scan compression methods have been proposed [6], commercialized by EDA vendors [7] and commonly used in industry ICs. Static and dynamic test cube compaction techniques [8] have been a standard component of ATPG tools for a long time and new techniques [9] continue to emerge at a slow pace. The test community has started to focus on test concurrency as a new dimension in test cost reduction, aiming at lowering the average test cost per chip [10]. Although compression levels in excess of hundredfold have been attained in recent years, the current level of compression fails to tame the ever increasing test cost, necessitating more advanced test cost reduction techniques as stated in the International Technology Roadmap for Semiconductors (ITRS) [10].

As previously reported by industry and academic researchers [11, 12], a significant number of redundant patterns exist in the test sets, resulting in a substantial increase in test time yet at no concomitant defect coverage benefit. The use of simplistic fault models such as stuck-at for test generation due to challenges in modeling and generating targeted tests for all possible defects is one of the reasons for the redundancy in the test sets. Inclusion of various test types such as stuck-at and IDDQ in the test set is another main driver of test pattern redundancy. Correlation models among different test types are typically lacking, resulting in a tremendous amount of overlap between the tests. For example, silicon defects detected by a particular stuck-at pattern may be alternatively detected by a combination of IDDQ tests, other stuck-at tests or a functional test. New test types such as N-detect and gate-exhaustive tests further exacerbate the issue by systematically adding redundancy into the test set in order to increase the probability of detection of actual silicon defects. Numerous experiments (DOE) have been reported in the past to analyze the effectiveness of various test types [13,14]. These experiments collect defect detection data for each test type on a sample of chips and results are typically reported in the format depicted in Fig. 1 [13]. The focus of these experiments predominantly has been the identification of the unique defects detected by each test type; the converse problem of the enormous defect detection overlap between test types has not as of yet received commensurate attention. It is reported in [11] that redundant patterns constitute 70% to 90% of the test set, based on studies from major semiconductor companies. As the effectiveness of individual test types diminishes, necessitating in turn the addition of new test types, it can be expected that the redundancy in test sets will grow further. Test optimization by identifying and eliminating redundant test patterns promises a tremendous opportunity in test cost reduction.

978-3-9810801-7-9/DATE11/©2011 EDAA

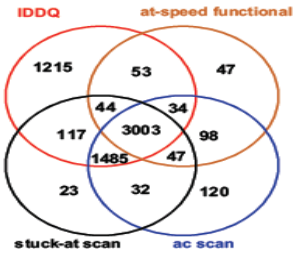

Fig. 1. Test type effectiveness

The goal of testing is to provide coverage of real silicon defects despite simple fault models being used in test pattern generation. Since modeling and generating targeted tests efficiently for all defects is not feasible, we argue in this paper that actual defects should be empowered to decide on the optimal test pattern mixture, pruning ineffective test patterns in the process. Essentially, the silicon defect detection behavior of individual test patterns, not the test sets of different types, should be analyzed by collecting data during production testing to identify the defect coverage overlaps among all test patterns and subsequently to determine the optimal test mixture. To be able to make statistically relevant decisions, test pattern effectiveness analysis should be performed over a large set of chips. The data collection should not result in test time increase and the data should be maintained incrementally in a compact form. Additionally, defect mechanisms can change over the lifecycle of the product, resulting in the optimal test set for the current lot failing to be even a satisfactory test set for another lot of devices; therefore, the optimal test set should dynamically adapt to the defect mechanism shifts [12]. These goals can only be achieved by making real time analysis and decisions during production testing.

In this paper, we propose a technique to learn the optimal test mixture dynamically during production test without incurring additional test cost. The test cost is kept at least constant by preserving the stop-at-first failure mechanism and actually even improved as the ordering of test patterns based on their effectiveness reduces the average test time for defect detection. Additionally, the proposed method monitors the defect behavior changes and adapts to the shifts in the defect mechanisms.

Section II reviews the previously published work in this area. Dynamic learning of individual test vector effectiveness is presented in Section III. Ineffective test vector elimination and adaptivity to defect behavior changes are presented in Section IV. Section V discusses the experimental results in detail and conclusions are drawn in Section VI.

II. Previous Work

Traditionally test flows are static, applying test patterns in a predefined order, with the test application stopping at the first failing test, thus reaping test time reductions for defective devices. DOEs for test effectiveness analysis override stop-at-first-failure behavior, collecting pass/fail data for the full test set. Due to the large sample size required by individual test effectiveness analysis, stop-at-first-failure override can be prohibitively costly. Furthermore, defect mechanisms can change throughout the product life cycle, requiring adaptivity to the changes.

A number of earlier pieces of work attack the optimal test pattern ordering problem by using complete pass/fail information for all patterns from a sample of devices. An efficient heuristic to determine the pattern order to reduce the average test time for defective devices is proposed in [15]. The effect of the ordering of test types on test cost is analyzed in [16] by utilizing SEMATECH data. A method that enables the prediction of the DPPM level for a test set of multiple test types is proposed in [19]. The resulting DPPM model can potentially be utilized to find a cost effective test set. These methods require costly pass/fail information collection for the full test set and, most importantly, they are not adaptive, instead presuming that the initial order remains effective throughout.

Initial research in adaptive test development primarily exists in the parametric test domain [12, 17, 18]. Since parametric tests return a measurement, not only a pass or fail information as in digital tests, these methods perform correlation analysis based on parametric measurements from a sample of devices for each lot and adaptively select optimal parametric tests per lot. A method that adaptively changes test content based on failure mechanisms [20] was recently proposed. It requires defect diagnosis throughout the production lifecycle to understand defect types and change test patterns accordingly.

III. Test Effectiveness Learning Through Dynamic Test Flow

In order to learn the effectiveness of individual test patterns and adapt to the defect behavior shifts, we propose an interactive learning framework that utilizes real time defect detection information during production testing. To identify the optimal test set, the proposed method dynamically alters the test flow during test application to maximize the information available to a subsequent learning step while still stopping at first failure. Essentially, a dynamic test flow and a policy that decides on the order of test vectors work together in an interactive loop as depicted in Fig. 2. The order of test patterns is decided by the policy and the dynamic test flow applies the vectors in this particular order and feeds back the detection information to the policy. The policy updates the pattern order based on the feedback and the process is repeated, learning in an interactive loop and converging to the optimal test pattern order. The test flow stops at first failure and the data is maintained incrementally in a compact form. The benefit of this interactive learning process is twofold. Firstly, the test patterns are ordered based on their effectiveness, reducing the average test time for defective chips by detecting the failures early in test application. Secondly and most importantly, ineffective tests are pinpointed by sieving them to the end of test pattern sequence, introducing the possibility of eliminating these patterns as the

test flow converges, thus substantially reducing test cost for both defective and defect-free devices.

The learning framework relies on defect detection effectiveness information for each pattern to determine the pattern order. This particular information is captured by keeping track of past defect detection data by assigning scores to the patterns and the test pattern order is determined to maximize the information available despite the information-gathering limitations of stopping at first failure. The details of scoring methods are presented in Section III.A and the test effectiveness learning algorithm is discussed in Section III.B.

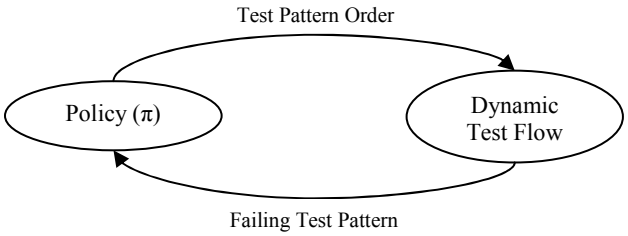

Fig. 2. Interactive learning loop

A. Test Pattern Defect Detection Information

The policy establishes a test pattern order by sorting the patterns based on the scores that are collected by each pattern. The first scoring method we propose, referred to as *detection count scoring*, maintains defect detection count for each pattern irrespective of the position in the test flow by assigning one additional credit to the test pattern that catches the current defective chip under test. However, since the ultimate goal is the optimization of total test time, the run time of each test should be taken into account, giving more credit to shorter tests. The proposed scoring method normalizes the scores of the patterns with respect to their test times, assigning weighted scores to the patterns based on their run time.

If a pattern located late in the current order detects a defect that was missed by all vectors before it in the test flow, it is highly indicative of the fact that this particular pattern may be targeting a unique and hard-to-detect defect. To take advantage of this observation, the second scoring method we propose, referred to as *detection time scoring*, takes the position of the vectors into account by assigning the total test time elapsed up to the failing pattern as the additional score to this particular failing vector.

Given that the *i*th test pattern, π_i, in the current test order detects the defective device and that the test time of this particular pattern is $testtime(\pi_i)$, the score of the pattern is updated as in equations (1) and (2) for detection count and detection time scoring, respectively.

$$score(\pi_i) += \frac{1}{testtime(\pi_i)} \quad (1)$$

$$score(\pi_i) += \frac{\sum_{j=1}^{i} testtime(\pi_j)}{testtime(\pi_i)} \quad (2)$$

B. Test Effectiveness Identification

The proposed learning framework constitutes an incremental optimization method, striving to converge to the optimal order. It is well known that greedy decision based incremental optimization methods frequently display convergence to local optimal solutions. Furthermore, the local optimal solution converged to may display significant sensitivity to the initial order of the vectors. Although the *detection time scoring* method is designed to make more drastic changes in the test order, the advantage some vectors receive due to their initial position in the test flow may prove difficult to overcome. Non-greedy decisions in addition to greedy ones are typically used in incremental optimization problems to surmount the stickiness of the local optimal solution.

We adopt an approach, known as ε-greedy [21], wherein non-greedy decisions are executed with a probability of ε, with greedy decisions resorted to otherwise. In the proposed method, completely random test pattern orders are used initially and the scores are collected based on the scoring methods described above. Later, the ε-greedy phase starts and random pattern orders are selected with a probability of ε; otherwise, the patterns are greedily sorted based on the current scores. Upon termination of the ε-greedy phase, the test pattern order decisions are confined to being solely made greedily based on the scores. It is worth stressing that the proposed technique continues to collect scores regardless of whether it is a greedy or a non-greedy order throughout the production life cycle and always stops at the first failure.

ε-greedy Pattern Effectiveness Learning: For a random ordering and *ε*-greedy phase sample sizes of N_{ran} and N_{ε}, respectively, with a non-greedy order selection probability of *ε*, the following steps are executed:

1. Sort the test patterns in random order for the first N_{ran} failing devices
2. For the next N_{ε} failing devices, either sort the test patterns randomly with a probability of ε, else sort them based on scores with a probability of 1- ε
3. Sort the test patterns based on the scores for all subsequent devices

The effectiveness of the various techniques proposed in this section is experimentally evaluated in section V and their performance compared.

IV. Adaptive Test Pattern Optimization

Learning of test effectiveness through the dynamic test flow presented in Section III not only reduces the average time to screen out defective devices by ordering the vectors based on their effectiveness but also enables the identification of ineffective vectors. Criteria based on defect detection information are needed to decide which vectors are redundant and eliminate these particular vectors during production testing, thus providing an opportunity for substantial test cost reduction. However, the underlying defect mechanisms may

change throughout the production life cycle [12], making test vector dropping a risky proposition. Therefore, the proposed method should also detect the defect behavior changes and take appropriate actions to keep test escape within the target levels.

The proposed method iteratively learns to converge to the optimal order; yet the optimality of this order is predicated on past detection behavior. If and when the defect mechanism starts to shift, the order will start to change so as to converge to the new optimal test order. Based on this observation, we propose the use of the correlation of the different pattern orders as a method to monitor defect behavior changes and, additionally, to decide when the order can be deemed to have converged portending the pattern dropping phase.

The test pattern order correlation metric used in our method is presented in Section IV.A. The utilization of the correlation to decide on test pattern ineffectiveness and defect behavior change is discussed in Sections IV.B.

A. Test Pattern Order Correlation

Since only the vector that detects the current faulty chip may move at consecutive test pattern orders in the proposed learning framework, the correlation between two consecutive orders will not vary significantly throughout the test application, providing inadequate information. Instead we compute the correlation between the pattern orders at the beginning and the end of a moving window of length, W_{coef}, to sharpen the information content. This method allows us to track the correlation changes between the test pattern orders within a specified time frame.

Spearman's rank correlation coefficient as defined in (3) is used in the proposed method to measure the correlation of two different test pattern orders, wherein n is the number of patterns and d_j is the difference of the ranks of the *jth* pattern in these two different test pattern orders. A rank correlation coefficient of 1 indicates full correlation while -1 indicates perfect inverse correlation.

$$\rho_i = 1 - \frac{6\sum d_j^2}{n\left(n^2 - 1\right)} \tag{3}$$

Rank correlation coefficient, ρi, for the ith failing device: Given that π^i is the order of test patterns for the *ith* failing device and W_{coef} is the length of the correlation distance, ρ_i is the correlation coefficient computed by (3) between the pattern orders, π^i and $\pi^{i\text{-}W\,coef}$.

B. Ineffective Test Pattern Elimination and Adaptivity to Defect Mechanism Shifts

The rank correlation coefficient, ρ_i, is monitored throughout the production life cycle and compared against the two different threshold values, τ_{conv} and τ_{shift}, to decide when the pattern order converges and when the defect behavior shifts after each failing chip.

When the pattern order converges as the rank correlation coefficient, ρ_i, surpasses the convergence threshold, τ_{conv}, the ineffective vector elimination phase is initiated. The ineffectiveness of vectors is determined by checking their defect detection history for the last D_{inef} devices. A *last detection time* (*LDT*) in addition to the score is maintained for each vector in the proposed method. A global counter (T_{cur}) is used which is increased by one for every defective device encountered during production test. When a test pattern detects a defective device, the last detection time of this particular vector is updated by the value of the global counter. When the convergence condition (i.e., $\rho_i > \tau_{conv}$) is satisfied, the vectors that have not detected any defects for a specified amount of time are dropped.

Test pattern elimination criterion: A particular pattern is eliminated if the following condition is satisfied.

$$(\rho_i > \tau_{conv}) \text{ and } ((T_{cur} - LDT) > D_{inef}) \tag{4}$$

The rank correlation coefficient is continuously monitored. If the correlation coefficient falls below the defect behavior shift threshold (τ_{shift}) subsequent to a vector elimination, it is interpreted as a change in the underlying defect mechanism and all eliminated vectors are inserted to the tail end of the test flow. The resumption of test effectiveness learning delivers elimination of ineffective test vectors once again when the test pattern elimination criterion is satisfied.

Defect mechanism shift detection criterion: All previously eliminated vectors are inserted back to the test flow if the following condition is satisfied.

$$(\rho_i < \tau_{shift}) \tag{5}$$

Through the utilization of the techniques presented in this section, the proposed method continues to learn throughout the production life cycle by maintaining a small amount of data and the production test pattern set dynamically shrinks to eliminate ineffective vectors and expands to adapt to the shifts in underlying defect mechanism.

V. Experimental Results

As discussed in earlier sections, the proposed framework explores the overlap of the patterns from multiple test types in actual defect detection. Due to the absence of actual defect detection data, in our experimental setup, we use a sample set of most likely bridging faults as surrogates and stuck-at patterns as the test set. The comparison of various techniques we have proposed and the confirmation of the use of correlation for test pattern order convergence constitute the focus of the experiments. Since a single test and surrogate type are used in our experiment, the actual test cost reduction may vary based on real defect behavior and test types used in the production. Consequently, we focus on the comparison of the efficiency of various configurations instead of the level of reduction in test cost.

An industrial design, consisting of 48,500 flip flops and 1.1 million gates, is used in the experiments. 4,000 stuck-at scan vectors are generated by a commercial ATPG tool. The coupling report is utilized to select the sample set of most likely bridging candidates that are injected to the design as defects during the test effectiveness learning process.

In the first set of experiments, the effect of ordering in the average defect detection time is analyzed. 4 different configurations as outlined below are compared against the static test flow.

Config1: Detection count scoring is used. Order is consistently greedy based on the scores (N_{ran} =0, N_{ε} =0).
Config2: Detection time scoring is used. Order is consistently greedy based on the scores (N_{ran} =0, N_{ε} =0).
Config3: Detection count scoring is used. *ε-greedy* approach is applied with the following parameters:

N_{ran} =10,000 , N_{ε} =10,000 and ε=10%

Config4: Detection time scoring is used. *ε-greedy* approach is applied with the following parameters:

N_{ran} =10,000 , N_{ε} =10,000 and ε=10%

Static: Traditional static test flow with no pattern ordering.

200,000 faults that are randomly selected from the sample set of bridging faults are injected to the design one at a time. The experiments start with a random test order. Since the proposed optimization framework aims at lowering the average test time to detect the defects, the trailing average of test time for the last 10,000 faults at each point during the testing is reported in Fig. 3. Since each pattern has the identical length in our experiments, the test time reported is proportional to the number of patterns applied. As can be seen, the test time for all 4 configurations quickly improves as the test pattern effectiveness is learned. The test time reduction trends for all configurations track each other quite closely, gradually converging to a stable level and significantly outperforming the static flow.

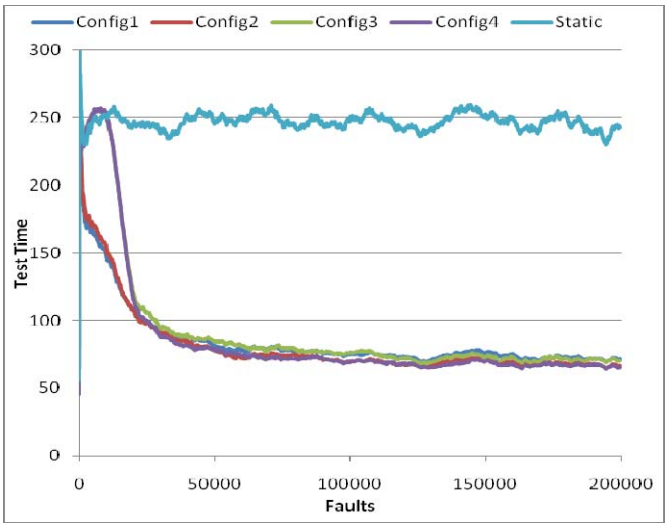

Fig. 3: Average test time as test effectiveness is learned

TABLE I. TEST TIME IMPROVEMENT OVER STATIC TEST FLOW

Method (Test Time)	*Config1* (71.69)	*Config2* (66.56)	*Config3* (70.69)	*Config4* (65.44)
Static (243.721)	3.4X	3.66X	3.45X	3.72X

The average test time for the last 10,000 faults and the improvement over the static test flow can be observed in Table 1. A close examination of the results shows that *detection time scoring* (*Config2* and *Config4*) delivers better results than *detection count scoring* (*Config1* and *Config3*). Additionally, the *ε-greedy* method slightly exceeds the greedy version. The *ε-greedy* flow with *detection time scoring* (*Config4*) delivers the highest reduction in test time.

Since the experiment indicates that *ε-greedy* with *detection time scoring* delivers the best results, the subsequent experimental results will only be reported for this configuration. The test pattern order rank correlation coefficient for this particular experiment is depicted in Fig. 4. The correlation distance, $W_{coef,}$ is set to 1000 when calculating the correlation. As can be seen, after the initial *ε-greedy* phase, the correlation between the pattern orders increases quickly and reaches up to the full correlation level.

In the next set of experiments, the ineffective test vector elimination flow is evaluated. The convergence threshold, $\tau_{conv,}$ for the rank correlation coefficient, $\rho_{i,}$ is set to 0.9999 and the ineffectiveness limit, $D_{inef,}$ is set to 100,000. Fig. 4 depicts the change in the number of vectors throughout the testing. Once the test order converges and the ineffectiveness limit is reached, dynamic vector elimination is initiated and a steady drop in the number of vectors is observed. The number of vectors stabilizes around 1000 patterns, delivering a more than fourfold reduction from the original 4000 patterns in our particular experimental setup.

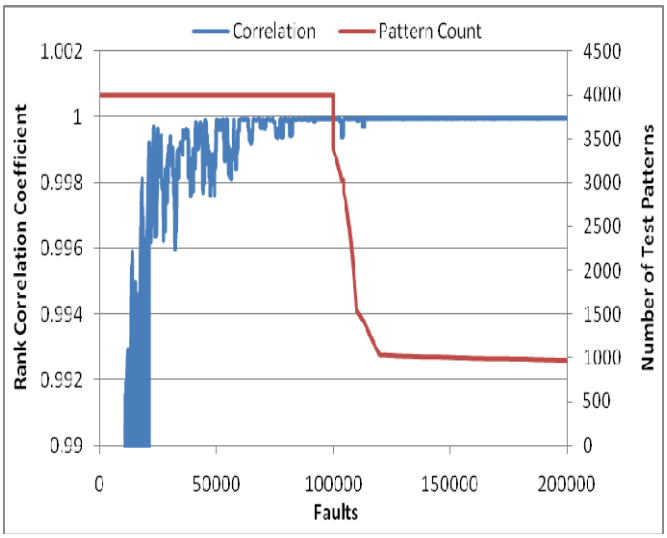

Fig. 4. Test pattern order convergence

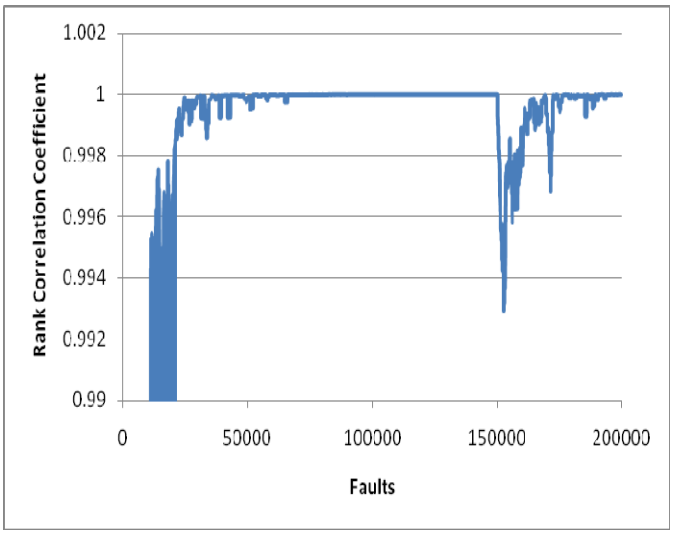

Fig. 5. Effect of defect behavior change

In the final set of experiments, the effect of defect behavior shift on the correlation is analyzed. The sample set of bridging faults is divided into two groups. Initially, faults from the first group are injected to the design and the test pattern effectiveness is learned based on this set of faults. Subsequently, fault injection from the second bridging fault set is used to model the change in the defect mechanism. The correlation coefficient throughout this process is plotted in Fig. 5, wherein the fault injection from the second set starts after the initial 150,000 faults. The rapid drop in the correlation when the defects start to change can be clearly seen, confirming the efficacy of our use of the correlation coefficient change to detect and adapt to the defect behavior shifts. After the rapid drop in the correlation, the pattern order again starts to converge for the new set of defects.

VI. CONCLUSIONS

Ever increasing test cost due to the increase in circuit complexity and additional new test types necessitates advanced test cost reduction techniques, going beyond the current state of the art methods. In this paper, we propose an adaptive test framework that learns the effectiveness of the individual test patterns, ordering the vectors based on their effectiveness in defect detection and eliminating the redundant vectors to reduce the test cost significantly.

In the proposed framework, a policy that decides on test pattern order interacts with a dynamic flow that applies the vectors in the order provided by the policy. An algorithm that utilizes the defect detection feedback provided by the dynamic flow to establish the optimal test order is presented. Rank correlation of the test pattern orders throughout production testing is monitored to identify when the test pattern order converges, leading to the elimination of ineffective vectors. Additionally, the variations in the rank correlation of test pattern orders are used as a trigger to identify the shifts in the defect mechanisms. The stop-at-first-fail mechanism is preserved and only a small amount of data is maintained, accentuating the practicality of the proposed approach. Experiments in an industry design show the effectiveness of the proposed algorithms.

The proposed method not only reduces the average test time to detect the defective devices by ordering vectors based on their effectiveness but it also significantly reduces the test time for all devices by exposing and eliminating the ineffective vectors. The ever increasing test cost problem can be thus significantly ameliorated, reducing test's overall contribution to final product cost and leading to an improved competitive position in the marketplace.

REFERENCES

[1] S. C. Ma, P. Franco and E. J. McCluskey, "An experimental chip to evaluate test techniques experiment results", Proc. Intl. Test Conf., 1995, pp. 663-672.
[2] K. Y. Cho, S. Mitra and E. J. McCluskey, "Gate Exhaustive Testing", Proc. Intl. Test Conf., 2005, pp. 1-7.
[3] X. Lin, K. H. Tsai, C. Wang, M. Kassab, J. Rajski, T. Kobayashi, R. Klingenberg, Y. Sato, S. Hamada and T. Aikyo, "Timing-Aware ATPG for High Quality At-speed Testing of Small Delay Defects" , Proc. Asian Test Symp., 2006, pp. 139 – 146.
[4] A. Uzzaman, M. Tegethoff, B. Li, K. McCauley, S. Hamada and Y. Sato, "Not all Delay Tests Are the Same - SDQL Model Shows True-Time", Proc. Asian Test Symp., 2006, pp. 147-152.
[5] P. Gillis, K. McCauley, F. Woytowich and A. Ferko, "Low Overhead Delay Testing of ASICs", Proc. Intl. Test Conf., 2004, pp. 534-542.
[6] N. A. Touba, "Survey of Test Vector Compression Techniques", IEEE Design & Test of Computers, Vol. 23, No. 4, 2006, pp. 294 – 303.
[7] J. Rajski, J. Tyszer, M. Kassab and N. Mukherjee, "Embedded Deterministic Test", IEEE Trans. on Computer-Aided Design, Vol. 23, No. 5, May 2004, pp. 776-792.
[8] M. Abramovici, M. A. Breuer and A. D. Friedman, "Digital Systems Testing & Testable Design", Wiley-IEEE Press, 1994.
[9] J. Geuzebroek, E. J. Marinissen, A. Majhi, A. Glowatz and F. Hapke, "Embedded multi-detect ATPG and Its Effect on the Detection of Unmodeled Defects", Proc. Intl. Test Conf., 2007, pp. 1-10.
[10] http://www.itrs.net/
[11] F.-F. Ferhani, N. R. Saxena, E. J. McCluskey and P. Nigh, "How Many Test Patterns are Useless?", Proc. VLSI Test Symp., 2008, pp. 23 – 28.
[12] R. Madge, B. Benware, R. Turakhia, R.Daasch, C. Schuermyer and J. Ruffler, "In Search of the Optimum Test Set – Adaptive test methods for maximum defect Coverage and Lowest Test Cost", Proc. Intl. Test Conf., 2004, pp. 203-212.
[13] P. Maxwell, "The Design, Implementation and Analysis of Test Experiments", Proc. Intl. Test Conf., 2006, pp. 1-9.
[14] P. Nigh, W. Needham, K. Butler, P. Maxwell and R. Aitken, "An Experimental Study Comparing the Relative Effectiveness of Functional, Scan, IDDQ and Delay-Fault Testing", Proc. VLSI Test Symp., 1997, pp. 459-464.
[15] W. Jiang and B. Vinnakota, "Defect-Oriented Test Scheduling", IEEE Trans. on VLSI Systems, Vol. 9, No. 3, June 2001, pp. 427-438.
[16] K. M. Butler and J. Saxena, "An empirical study on the effects of test type ordering on overall test efficiency", Proc. Intl. Test Conf., 2000, pp. 408 – 416
[17] S. Benner and O. Boroffice, "Optimal Production Test Times Through Adaptive Test Programming", Proc. Intl. Test Conf., 2001, pp. 908-915.
[18] M. Chen and A. Orailoglu, "Test Cost Minimization through Adaptive Test Development", Proc. Intl. Conf. on Computer Design, 2008, pp. 234-239.
[19] K. M. Butler, J. M. Carulli and J. Saxena, "Modeling Test Escape Rates As a Function of Multiple Coverages", Proc. Intl. Test Conf., 2008, pp. 1-9
[20] X. Yu, Y.-T. Lin, W.-C. Tam, O. Poku and R.D. Blanton, "Controlling DPPM through Volume Diagnosis", Proc. VLSI Test Symp., 2009, pp. 134-139.
[21] R. S. Sutton and A.G. Barto, "Reinforcement Learning: An Introduction", The MIT Press, 1998.