UNIVERSITY OF CALIFORNIA, SAN DIEGO

Adaptive Test Cost and Quality Optimization Through An Effective Yet Efficient Delivery of Chip Specific Tests

A dissertation submitted in partial satisfaction of
the requirements for the degree
Doctor of Philosophy

in

Computer Science

by

Baris Arslan

Committee in charge:

Professor Alex Orailoglu, Chair
Professor Chung-Kuan Cheng
Professor Sadik Esener
Professor Yoav Freund
Professor William E. Howden

2013

The Dissertation of Baris Arslan is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

_____

_____
Chair

University of California, San Diego

2013

DEDICATION

To *my parents*.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Chapter 3, in part, is a reprint of the material as it appears in *B. Arslan and A. Orailoglu, "Adaptive Test Optimization through Real-time Learning of Test Effectiveness", Design, Automation and Test in Europe Conference, 2011*; and in *B. Arslan and A. Orailoglu, "Aggressive Test Cost Reductions Spanning Multiple Levels of Test Flow through Continuous Test Effectiveness Assessment", submitted to IEEE Transactions on Very Large Scale Integration Systems*. The dissertation author was the primary investigator and author of these papers.

Chapter 4, in part, is a reprint of the material as it appears in *B. Arslan and A. Orailoglu, "Adaptive Test Framework for Achieving Target Test Quality at Minimal Cost", Asian Test Symposium, 2011*; and in *B. Arslan and A. Orailoglu, "Tracing the Best Test Mix through Multi-Variate Quality Tracking", VLSI Test Symposium, 2013*. The dissertation author was the primary investigator and author of these papers.

Chapter 5, in part, is a reprint of the material as it appears in *B. Arslan and A. Orailoglu, "Delay Test Quality Maximization through Process-aware Selection of Test Set Size", International Conference on Computer Design, 2010*; and in *B. Arslan and A. Orailoglu, "Full Exploitation of Process Variation Space for Continuous Delivery of Optimal Delay Test Quality", Asia and South Pacific Design Automation Conference, 2013*. The dissertation author was the primary investigator and author of these papers.

Chapter 6, in part, is a reprint of the material as it appears in *B. Arslan and A. Orailoglu, "Delay Test Resource Allocation and Scheduling for Multiple Frequency Domains", VLSI Test Symposium, 2012*; and in *B. Arslan and A. Orailoglu, "Power-Aware Delay Test Quality Optimization for Multiple Frequency Domains", submitted to IEEE*

Transactions on Computer-Aided Design of Integrated Circuits and Systems. The dissertation author was the primary investigator and author of these papers.

Chapter 7, in part, is a reprint of the material as it appears in *B. Arslan and A. Orailoglu, "CircularScan: A Scan Architecture for Test Cost Reduction", Design, Automation and Test in Europe Conference, 2004*; in *B. Arslan and A. Orailoglu, "Design Space Exploration for Aggressive Test Cost Reduction in Circular Scan Architectures", International Conference on Computer-Aided Design, 2004*; and in *B. Arslan and A. Orailoglu, "Test Cost Reduction through A Reconfigurable Scan Architecture", International Test Conference, 2004*. The dissertation author was the primary investigator and author of these papers.

VITA

2000	Bachelor of Science, Computer Engineering & Information Science Bilkent University, Ankara, Turkey
2001-2005	Teaching Assistant, Department of Computer Science and Engineering University of California, San Diego, California
2002	Master of Science, Computer Science University of California, San Diego, California
2002-2004	Research Assistant, Department of Computer Science and Engineering University of California, San Diego, California
2005-2006	DFT Engineer Intel Corporation, Chandler, Arizona
2006-2007	DFT Engineer Marvell Semiconductor Inc., Chandler, Arizona
2007-2013	Staff Engineer Qualcomm Inc., San Diego, California
2013	Doctor of Philosophy, Computer Science University of California, San Diego, California

PUBLICATIONS

B. Arslan and A. Orailoglu, "Tracing the Best Test Mix through Multi-Variate Quality Tracking", *VLSI Test Symposium,* pp. 1-6, 2013.

B. Arslan and A. Orailoglu, "Full Exploitation of Process Variation Space for Continuous Delivery of Optimal Delay Test Quality", *Asia and South Pacific Design Automation Conference,* pp. 552-557, 2013.

B. Arslan and A. Orailoglu, "Delay Test Resource Allocation and Scheduling for Multiple Frequency Domains", *VLSI Test Symposium,* pp. 114-119, 2012.

B. Arslan and A. Orailoglu, "Adaptive Test Framework for Achieving Target Test Quality at Minimal Cost", *Asian Test Symposium,* pp. 323-328, 2011.

B. Arslan and A. Orailoglu, "Adaptive Test Optimization through Real-time Learning of Test Effectiveness", *Design, Automation and Test in Europe Conference,* pp. 1430-1435, 2011.

B. Arslan and A. Orailoglu, “Delay Test Quality Maximization through Process-aware Selection of Test Set Size”, *International Conference on Computer Design*, pp. 390-395, 2010.

B. Arslan and A. Orailoglu, “CircularScan: A Scan Architecture for Test Cost Reduction”, *Design, Automation and Test in Europe Conference*, pp. 1290-1295, 2004.

B. Arslan and A. Orailoglu, “Test Cost Reduction through a Reconfigurable Scan Architecture”, *International Test Conference*, pp. 945-952, 2004.

B. Arslan, O. Sinanoglu and A. Orailoglu, “Extending the Applicability of Parallel-Serial Scan Designs”, *International Conference on Computer Design*, pp. 200-203, 2004.

B. Arslan and A. Orailoglu, “Design Space Exploration for Aggressive Test Cost Reduction in Circular Scan Architectures”, *International Conference on Computer-Aided Design*, pp. 726-731, 2004.

B. Arslan and A. Orailoglu, “Extracting Precise Diagnosis of Bridging Faults from Stuck-at Fault Information”, *Asian Test Symposium*, pp. 230-235, 2003.

B. Arslan and A. Orailoglu, "Perfect Conservation of Diagnostic Information in Aggressive Reduction of SOC Test Bandwidth and Use”, *European Test Workshop*, pp. 13-14, 2003.

B. Arslan and A. Orailoglu, "Aggressive SOC Test Response Compaction with Full Diagnostic Information Conservation”, *International Test Synthesis Workshop*, 2003.

B. Arslan and A. Orailoglu, “Fault Dictionary Size Reduction through Test Response Superposition”, *International Conference on Computer Design*, pp. 480-485, 2002.

PUBLICATIONS UNDER REVIEW

B. Arslan and A. Orailoglu, “Power-Aware Delay Test Quality Optimization for Multiple Frequency Domains”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*.

B. Arslan and A. Orailoglu, “Aggressive Test Cost Reductions Spanning Multiple Levels of Test Flow through Continuous Test Effectiveness Assessment”, *IEEE Transactions on Very Large Scale Integration Systems*.

ABSTRACT OF THE DISSERTATION

Adaptive Test Cost and Quality Optimization Through
An Effective Yet Efficient Delivery of Chip Specific Tests

by

Baris Arslan

Doctor of Philosophy in Computer Science

University of California, San Diego, 2013

Professor Alex Orailoglu, Chair

The higher levels of integration and process scaling imposes failure behaviors which are challenging to interpret, necessitating the continuous augmentation of fault models and test vectors in the hopes of taming the defect escape rate. The subsequent inflation in the number of test vectors coupled with the constant increase in the size of each test vector continuously boosts test cost. The economics of particularly the competitive consumer marketplace however require a constant vigilance at the test cost while ensuring a satisfactory test quality.

While the inclusion of new fault models helps boost test quality, the non-uniform distribution of various defect types and the defect coverage overlaps between fault models imply variable effectiveness of fault models and test vectors, resulting in the inclusion of a large number of ineffective vectors in test flow. A static derivation of test effectiveness however remains problematic in practice as it is well known that defect characteristics are prone to drifts throughout the product lifecycle. Furthermore, the increasing process variation and the integration of hundreds of domains within a chip result in increasingly distinct domains and individualized chip instances with diverse test resource requirements. The conventional test method of a static application of an identical test set to all chips consequently struggles to satisfy the demanding test cost and quality constraints in the face of the evolving defect behaviors and the increasing diversification in test resource requirements.

This thesis addresses the simultaneous necessity for satisfactory test quality and low test cost through an adaptive test cost and quality optimization framework. The proposed methodologies not only adaptively assess the effectiveness of fault models and test vectors but also evaluate the variable test resource requirements of the chips and domains based on their distinct characteristics, enabling an effective yet efficient test through the selection of the most effective vectors and a carefully crafted allocation of test resources. The proposed methodologies are tailored for a broad set of application scenarios through the consideration of different defect classes and defect characteristic drift types while incorporating the test data gathering and delivery constraints and overcoming the associated algorithmic challenges.

Chapter 1

Introduction

VLSI testing is performed subsequent to the manufacturing of the integrated circuit (IC) to screen out the defective chips caused by the imperfections in the manufacturing process, ensuring the delivery of high quality chips to the customers. The VLSI test field is however continuously challenged by aggressive advances in the IC manufacturing process. Higher levels of integration and continuous process scaling, fueled by the intense competition for market share in the semiconductor industry, enable the manufacturing of increasingly low power and high performance integrated circuits at a lower cost. Yet ambitious manufacturing technology leaps magnify the failure rates and engender failure behaviors which are challenging to interpret and test, necessitating rigorous and costly test methods.

Fault model based structural testing coupled with scan-based test architecture, which configures the flops of an IC as serially accessible shift registers, have gained wide acceptance in industry as a *de facto* test standard in the last couple of decades as a way of keeping test generation complexity and test cost of today's large System-On-A-Chip (SOC) within practical limits while delivering an acceptable test quality. Conventionally, a small set of fault models, such as stuck-at, transition and IDDQ, is targeted during the test generation process and the resulting test sets are applied identically to each chip during manufacturing testing through a static test flow. Fault coverage, such as stuck-at coverage, is used as a primary

metric to assess the quality of the test set. As numerous years of experience in structural testing have showed, fault model based structural test has proved to be very effective in screening defects in earlier process nodes.

As semiconductor manufacturing has moved to deep submicron (DSM) process nodes, subtle defects that effect the parametric attributes such as timing came into prominence as a result of shrinking geometries and tighter design margins, diminishing the effectiveness of the traditional fault models such as the stuck-at fault and IDDQ. The decline in the effectiveness of the existing test tests subsequently necessitates the continuous piling of new fault models and the corresponding test vectors in the hopes of taming the defect escape level, measured as defective parts per million (DPPM), at an increasingly higher cost. Various new fault models and test methods such as N-detect, gate-exhaustive, timing-aware delay test and faster-than-at-speed delay test have been proposed to keep DPPM within acceptable limits as discussed in Chapter 2.

Not only the sheer number of test vectors due to the inclusion of emerging test types, but also, as a result of the serial access mechanism of scan-based test architecture, the size and subsequently the cost of each individual test vector is quickly growing as the complexity and the size of SOCs increase, further boosting test cost. Test cost subsequently constitutes with each new process node an increasingly larger fraction of overall product cost as illustrated in Figure 1.1 based on the data reported by a semiconductor company [1]. Although the actual percentages may vary for different semiconductor companies and product families, the trend of the continuous increase of test cost as a fraction of overall cost as shown in this figure applies across the semiconductor industry, quickly heading to the point of the cost of test exceeding the cost of manufacturing in the near future. The results of an extensive survey reported in the International Technology Roadmap for Semiconductors (ITRS) 2011 edition [2] show that 85% of the survey participants expect the cost of test to become their biggest

concern going forward in the VLSI test field. In the face of newly emerging defect types in DSM process nodes and the continuous increase in the complexity of the chips, advance test methodologies are required to tame the rapid increase in test cost. Reductions in the number of test vectors and in the cost of each individual test vector can equivalently help to alleviate the concern of continuously increasing test cost.

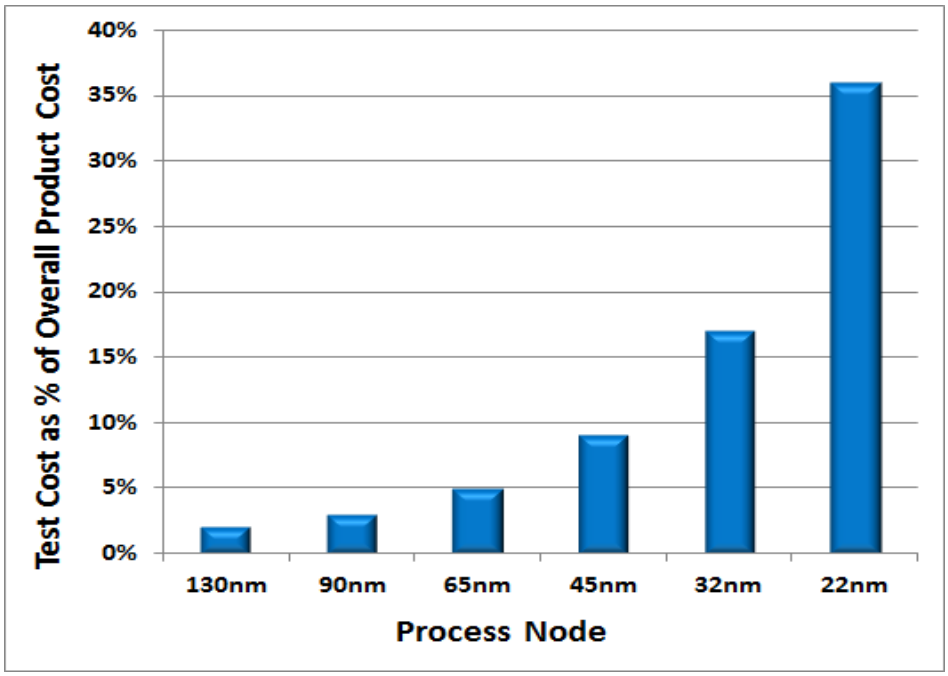


Figure 1.1 Test cost vs. overall cost

The effectiveness of fault models is typically analyzed by design of experiments (DOEs) with numerous results having been reported in the past [3], [4]. These experiments collect defective device detection data for each fault model on a sample of production chips and results are typically reported in a Venn diagram as depicted in Figure 1.2 [3]. The focus of these experiments predominantly has been the identification of the unique defects detected by each fault model, evaluating the additional DPPM improvement by a fault model. However, a defect may manifest itself in a number of distinct ways, leading to the detection of the same

defect by multiple fault models, resulting in defect coverage overlaps between fault models and the corresponding test vectors. As highlighted in Figure 1.2, a tremendous amount of defect detection overlap among fault models exists in practice, resulting in numerous ineffective test vectors. It is reported in [5] that ineffective patterns constitute 70% to 90% of the test sets, based on studies from major semiconductor companies. Furthermore, due to the non-uniform distribution of various defect types, different fault models and test vectors achieve distinct defect coverage levels as underscored by distinct numbers of defective devices detected by each fault model in Figure 1.2.

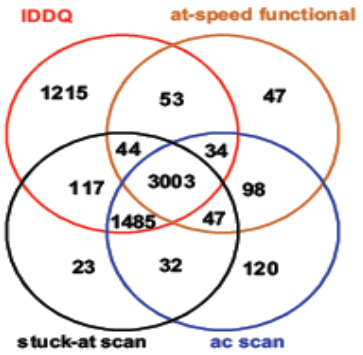


Figure 1.2 Fault model effectiveness

Consequently, while the introduction of new fault models helps boost test quality, appropriate attention needs to be paid to the effectiveness of each test vector, promoting the use of a proper mixture of test vectors from these fault models based on test vector effectiveness assessment in order to attenuate test cost. The economics of particularly the competitive consumer marketplace lend themselves to a rigorous test effectiveness assessment and a subsequent test cost and quality tradeoff analysis. As the intense competition in the consumer marketplace shrinks the profit margins, a constant vigilance at the test cost while ensuring a satisfactory test quality is required. The assessment of test effectiveness enables the exploration of test cost and quality tradeoff, resulting in the selection of a lowest cost test suite

while meeting the quality goals based on test economics. Furthermore, since the test is performed at multiple levels such as wafer-sort and package test, a test effectiveness assessment paves the way for test cost and quality optimization across multiple test levels.

Today, SOCs integrate numerous cores, ranging from multiple microprocessors, digital signal processing cores, and graphics processing units to multi-mode modems, consisting of hundreds of unique domains. The aggressive integration of a diverse set of cores in an SOC results in the cohabitation of various domains with distinct parametric attributes such as timing. Similarly, the increasing magnitude of process variation implies the individualization of each manufactured chip with distinct parametric profiles. The highlighted diversification across the domains and the chips exposes a major inefficiency of conventional test methods, namely, the application of an identical test set, selected based on fault coverage, to all chips. The detection of the subtle defects that effect parametric attributes necessitates not only the satisfaction of the logical defect defection distinct conditions but also the violation of target values such as frequency in the face of the activated defect when the tests are applied. Such parametric targets and the accompanying design margins vary across domains and chips depending on design and process parameters. The fault coverage, utilized to select the test sets, however quantifies the test quality only in terms of the satisfaction of logical fault detection conditions. Sole fault coverage based test quality assessment subsequently fails to provide an accurate picture of test quality as it overlooks the effect of the parametric attributes in defect detection. Similarly, distinct parametric profiles of chips as a result of process variation lead to an identical test with the same fault coverage delivering distinct test quality for each chip. Consequently, the conventional way of applying a fixed test set, selected by fault coverage based test quality assessment, fails to satisfy the cost and quality goals of VLSI test in the face of the evolving defect behaviors in DSM process modes and the increasing diversification within and across the chips.

The simultaneous demand of satisfactory test quality and low test cost in VLSI test evidently necessitates a paradigm shift from the conventional test techniques. New methodologies are required to assess the effectiveness of fault models and individual test vectors, resulting in the identification of a proper mixture of fault models and test vectors to deliver the desired test quality at a low cost. Furthermore, a variable allocation of the available test resources to each domain and to each individualized chip based on their distinct parametric attributes is necessary to extract the highest test quality, forgoing the long lasting practices of the solely fault coverage based test selection and the application of identical test set to each chip.

The static derivation of test effectiveness and test resource allocation in the process of test cost and quality optimization however imposes inordinate challenges in practice. The design and process parameters effect the occurrence frequency of various defect types and their distribution across chips, subsequently influencing the effectiveness of the fault models and the test vectors. The aforementioned defect characteristics (i.e. the type of defects and their distribution) may shift throughout the production life cycle due to the change in manufacturing environment and process parameters, precluding a static derivation of test effectiveness. Adaptivity is subsequently required in order to learn the effectiveness of the fault models and the test vectors and continuously explore and adjust the test quality and cost tradeoff points as the defect characteristics evolve.

Although adaptive test effectiveness assessment and variable test resource allocation can enable the exploration of the test quality and cost tradeoff and help to extract the highest test quality at a lower cost, they fall short of addressing the pressing issue of the continuously increasing individual test vector cost as a result of the serial access mechanism of scan-based test architecture. The reduction in the cost of each test vector through a test compression method can proportionally provide an immediate relief in the overall test cost and enable test

quality improvement within the test cost budget. Naturally, the importance of this particular component of test cost is also steadily growing. The cost of each test vector can be reduced in a scan-based test architecture by decreasing the number of test cycles required to apply a test vector through the serial access mechanism. The number of chip I/O pins used for test access should be limited as well due to the higher cost of testers with a large set of pins. An aggressive test compression technique in scan-based architectures that substantially reduces the number of test application cycles for each pattern while limiting chip I/O usage is subsequently necessary.

In the next section, a detailed overview of the motivation for paradigm shift in testing and the subsequent test strategy and architecture advancements needed for test cost and quality optimization are provided. Constraints and challenges are discussed in Section 1.2. A summary of the thesis is subsequently provided in Section 1.3.

1.1 Test Cost and Quality Optimization

1.1.1 Test Effectiveness and Efficiency Exploration

As the effectiveness of traditional fault models and the corresponding test sets such as stuck-at and IDDQ diminishes in DSM defect detection, the research community and industry alike are shifting their focus to new fault models. While the continuous trend of inclusion of new fault models in the test flow helps to improve test quality, it also frequently results in the inclusion of a significant number of ineffective test vectors in the test flow, reported to be in the range of 70% to 90% [5], at a high test cost yet with no concomitant defect coverage benefit.

The use of simplistic fault models such as stuck-at for test generation efficiency and also due to challenges in modeling and generating targeted tests for all possible defects is one

of the reasons for the inefficiency in the test flows. The lack of a direct correlation between fault models and the actual defects results in the generation of some test vectors with no or insignificant defect coverage. Secondly, defect occurrence rate variation across the chip as a result of design margin variations and systematic process issues implies that some test vectors are more effective than others in defect detection even within a test set generated by a single fault model. The inclusion of multiple fault models further exacerbates the level of inefficiency in the test flows. The manifestation of a defect in possibly multiple distinct ways, coupled with the use of higher level fault models, leads to the detection of the same defect with multiple test vectors from different fault models. The resulting defect coverage overlaps among the fault models and the corresponding test vectors underscores the existence of a significant level of ineffective vectors in the test flows. Furthermore, a non-uniform distribution of various defect types implies that different fault models deliver distinct defect coverage levels, leading to a variation in the defect detection effectiveness of different fault models.

The necessity of tailoring test suites and a test strategy to be *effective* in defect detection (i.e. delivering satisfactory defect escape levels) while performing the job *efficiently* within a short test time (i.e. acceptable test cost) requires an understanding of the effectiveness of test vectors in the test flow and a consequent test effectiveness & efficiency tradeoff. A test effectiveness assessment process needs to quantify the effectiveness of fault models and test vectors while considering the coverage overlaps in defect detection. As the effectiveness of individual fault models diminishes, necessitating in turn the addition of an increasingly larger number of fault models, it can be expected that the sheer number of test vectors with insignificant defect coverage will grow further. A test cost and quality optimization through the assessment of test effectiveness and a subsequent selection of the best test mixture from

various fault models and test vectors promises a tremendous opportunity in delivering an effective yet efficient test suite.

While the derivation of an effective yet efficient test suite through the identification and selection of the effective tests from a variety of fault models is certainly a desirable goal, it faces inordinate challenges in practice. The lack of a direct correlation between fault models and the actual defects, coupled with the dependency of the defect types and their distributions on the variable design and process parameters, severely limits the possibility of a pre-silicon exploration of test cost and quality tradeoff. Furthermore, even if a fault-defect coverage relationship while considering defect coverage overlaps is established based on a limited post-silicon data analysis, the defect characteristics (i.e. the type of defects and their distribution) that play an essential role in the quality of a test set may shift throughout the production life cycle. Process improvements to increase yield, process variation across lots and manufacturing environment and equipment updates are among the main causes of defect characteristics shifts. Consequently, not only test effectiveness assessment and the subsequent test cost and quality exploration should be performed based on the existing defect characteristics but it also should be adaptive as the defect characteristics evolve. This goal necessitates the collection and utilization of real time data from the production test flow and a subsequent dynamic alteration of the test flow, breaking away from the conventional static test flows. Furthermore, the data collection constraints of the production test environment due to the cost and data management challenges, particularly in external test houses, may preclude the availability of full test data. In order to promote a widespread applicability, adaptive test effectiveness assessment and the test cost-quality exploration process in order to deliver an effective yet efficient test flow should be capable of extracting the necessary information from the limited test data.

1.1.2 Variable Test Resource Allocation

While constraining distinct chip designs to an identical test resource would be a highly confounding practice, the idea of subjecting various manufacturing instances of the same design to an identical set is a well-accepted part of conventional test flows. The process variation during manufacturing however results in chips with distinct characteristics, effectively implying the individualization of each manufacturing instance of the same design. Nowhere is this individualization more pronounced than when it comes to delay effects, resulting in chips with distinct timing margins and consequently each chip's vulnerability to random delay defects exhibiting significant variations. The conventional approach of devoting an identical test resource to the highly individualized chips is subsequently starting to fray as the process variation intensifies with each new process node.

Similarly, the higher level of integration enables the coexistence of the hundreds of domains with distinct characteristics in the same chip. In particular, it is not uncommon to have various domain frequencies ranging from very low speeds to GHz levels. The resulting timing margin variations across the domains effectively imply that the domains exhibit variable vulnerability to random delay defects.

While the manifestation of distinct chip and domain characteristics impacts not only timing margins but also other parametric attributes such as power, it is particularly appropriate to focus on an examination of its influence on timing and delay test. Partly, this emphasis is due to the increasing occurrence frequency of subtle delay defects as the unceasing process scaling continues, preventing the operation of the circuit in the target frequencies. Delay testing consequently carries a big burden in delivering high quality chips to the customers, steadily increasing its share in overall test cost. Furthermore, conventional delay test flows are unable to provide a satisfactory answer as to how to effectively test for delay defects.

Since the emergence and wide acceptance of fault model based structural testing, fault coverage has been used as a primary metric to assess the quality of the test sets. As the numerous years of experience in structural testing show, fault coverage is indeed an effective metric to measure the test quality for static defects that only require logical activation and propagation of erroneous behavior. However, the increasing importance of parametric attributes such as delay throws into stark relief the limitations of fault coverage based test quality measurements. In order to detect a delay defect for example, not only is it required that a logical fault detection condition be satisfied by generating a transition at the faulty location and propagating it to an observation point, but it is furthermore imperative that the faulty behavior be propagated to an observation point within a time period determined by the target frequency. Evidently, actual signal propagation time depends on path lengths and the additional delay incurred by the delay defect. Although fault coverage considers the satisfaction of the logical detection condition, parametric attributes, such as frequency, path lengths and defect size, that have direct effect on delay test quality, go unnoticed.

Consequently, in an era of pronouncedly individualized chip instances and the hundreds of distinct domains within a chip, the conventional delay test method of applying an identical test set to each chip, determined solely based on fault coverage, fail to satisfy the cost and quality goals of VLSI test. The chips from the slower corners of the process variation space, for example, exhibit diminished timing margins in comparison to the ones from the faster corners, consequently exhibiting reduced tolerance to delay defects. Similarly, the higher frequency domains with smaller timing margins have reduced tolerance to delay defects than the slower frequency domains. Evidently, a variable test resource allocation strategy based on chip and domain characteristics is necessary to optimize test quality while keeping the test cost within acceptable levels.

The major challenges in effective test resource allocation are the estimation of test quality based on chip and domain characteristics and the algorithmic advances necessary to efficiently identify the optimal allocation of available resources in order to extract highest test quality. Furthermore, test architectures of state-of-the-art SOCs support the concurrent testing of the domains, proving an opportunity for a substantial reduction in test cost. The variable resource allocation strategy should therefore be compatible with the concurrent domain testing, identifying in the process the best schedule of the concurrently tested domains while ensuring that test power limits are not exceeded due to an aggressive exploitation of test concurrency.

1.1.3 Test Compression

The higher level of integration continuously boosts the complexity of SOCs by enabling the placement of enhanced functionality into a single chip, increasing test generation complexity as well in the process. Scan-based test architectures have emerged as a widely accepted solution to keep the test generation complexity of SOCs within practical limits. The scan-based test architecture configures the flops of a circuit as shift registers, referred to as scan chains. Consequently, the circuit can be set to any state by serially shifting in the desired values (i.e. test patterns) and the current state of the circuit (i.e. test responses) can be observed by serially shifting out the values of the flops. The enhanced controllability and observability delivered by scan-based test architectures help increase the maximally attainable levels of fault coverage. Furthermore, since all state elements can be controlled and observed, it reduces the complexity of test generation by eliminating the necessity for sequential test generation. However, as a result of the serial access mechanism, test application time is inevitably increased, resulting in elevated test cost.

While the aforementioned test effectiveness assessment and variable test resource allocation help to optimize test cost and quality through an effective test selection and a judicious distribution of test resources, the increasing cost of individual test patterns dictated by a scan-based test architecture imposes a strict limitation on the attainable level of test cost and quality optimization. In order to achieve the goal of an effective yet efficient test and to fully enjoy the benefits of scan-based designs, the high test cost of individual patterns needs to be reduced through test compression (commonly referred to as scan compression) in an era where scan-based test architectures has become a *de facto* standard.

In scan-based test architectures, the number of test application cycles for a test pattern is equal to the maximum length of the scan chains. The use of the multiple scan chains, reducing the average scan chain length, emerges as a natural solution for reducing the test application time. Nevertheless, the extra scan I/O pins required by the increased number of scan chains necessitate the use of high cost testers with a large set of pins, precluding the scalability of the utilization of an increasingly larger set of scan I/O pins.

In practice, the scan-based test patterns generally consist of a small number of specified bits, reported to be in the range of 1%-5%. Although the specified bit density of the test patterns is quite low, the traditional scan-based test architecture does not allow the exploitation of this property. The unspecified bits of the test patterns are randomly set and subsequently both the specified and the randomly set bits are serially loaded. The major challenge of test compression in scan-based test architectures is to ensure the usage of a limited set of scan I/O pins while enabling the efficient delivery of the specified bits through the use of a set of reduced length scan chains to alleviate the cost of the serial shift mechanism.

1.2 Constraints and Challenges

The delivery of the adaptive test strategy advancements needed for test cost and quality optimization in face of the increasing inefficiencies of the conventional test flow and test architecture as outlined in Section 1.1 requires an answer to the question of how to overcome a number of constraints and challenges. The continuous evolution of defect characteristics is one of the key drivers in the development of advanced test strategies, necessitating an understanding of the classification of the defects and their effect on an adaptive test strategy. The functional constraints of the test flow and test architecture determines the level of flexibility and information available to new test strategies, raising the question of how to deal with these constraints. Finally, a set of algorithmic challenges, partly driven by defect characteristics and functional constraints, needs to be addressed to achieve the goal of adaptive test cost and quality optimization.

1.2.1 Defect Clasification

The characteristics of the defect play a key role in the development of test strategies since the early days of the field of VLSI test. The defects can be random in nature or can have regularity based on design and process parameters. The defects such as a short to the ground that have dominated the defect population in earlier process nodes tend to occur randomly. Parametric defects such as delay defects however depend on design margins and process variation, showing a regularity based on design and process parameters.

Randomness or regularity of a defect type has a profound impact on the development of adaptive test cost and quality optimization strategy. The randomness of a defect type severely limits the possibility of a pre-silicon analysis in the exploration of test cost and quality tradeoff. Since the random nature of defects precludes a ranking of defects based on

their criticality, the pre-silicon test generation phase of test strategies naturally aims at providing the highest possible coverage for the target fault models at a high cost. However, despite the randomness of some defect types, not only the occurrence rates of distinct random defects in the manufactured chips differ, but also the multiple test vectors from various test types may cover the same defect, resulting in a distinct effectiveness of each test vector in defect detection. Furthermore, the defect characteristics such as the distribution of defects may shift throughout the production lifecycle due to changes in the manufacturing process and environment as discussed in Section 1.1.1, resulting in a continuous change in test vector effectiveness. Consequently, an adaptive test strategy for random defect types needs to rely on a post-silicon analysis based on the test data collected from the production test flow in order to rank the effectiveness of the test vectors in defect detection and track the subsequent changes in test vector effectiveness based on defect characteristic shifts.

The defect types that exhibit regularity based on process and design parameters however lend themselves to a meticulous analysis during the pre-silicon phase. For example, the susceptibility of each net in the chip to the delay defects can be assessed based on a timing analysis, raising the possibility of a pre-silicon evaluation of test effectiveness to optimize the test cost and quality. Regularity in a defect type subsequently promotes the integration of static information obtained by a pre-silicon analysis in the adaptive test cost and quality optimization process. While an adaptive post-silicon decision making process is still required to obtain process information from the chip under test and to track the process variation changes, a pre-silicon analysis not only significantly reduces the burden of post-silicon analysis but also increases the accuracy and efficiency of the test optimization process.

1.2.2 Functional Constraints

The development of adaptive test strategies driven by the classification of defects as discussed in Section 1.2.1 is heavily influenced by the functional properties and constraints of test flow and test architecture. Since an adaptive test methodology, particularly for random defects, requires an intensive post-silicon data collection, the data gathering constraints of test flow have a profound effect on the test strategy. Similarly, the constraints of the test delivery method dictate the limits of test cost optimization.

Conventional static test flows stop at the first failing test to reduce the cost of test for defective devices. As the testing of multiple chips in parallel becomes a more common practice, the stop-at-failure requirement can be relaxed without incurring significant additional test cost, particularly for high yield products. However, due to cost and challenges associated with collecting and managing full failure information for all failing tests throughout the product lifecycle, particularly in external test houses, it is not uncommon in the industry to bin the failing devices based on the first failing vector, resulting in the collection of only first failing test data for each defective chip. The severely limited test data availability constitutes an enormous obstacle to the post-silicon test data driven adaptive test methodology development, necessitating new techniques to boost the available test data content without incurring a significant test cost. A dynamic adjustment of test flow while still collecting only first failure data provides an opportunity to extract more information from a test flow with no increase whatsoever in test cost. Additionally, a minor expansion of the test data collection with a minimal increase in test cost can open up the possibility of the development of a more accurate and efficient adaptive test method. An adaptive test strategy subsequently needs to answer the question of how to maximize the available information through the use of a dynamic test flow or a minimal expansion of the test data collection.

In addition to test data gathering constraints, the test architecture and methodology imposes test delivery constraints, necessitating the development of test strategies to extract the highest level of test optimization under these constraints. Test architectures can provide capabilities to perform concurrent testing of domains within a chip, presenting an opportunity for a substantial test cost reduction. However, the level of attainable concurrency is constrained by not only an architectural concurrency limit but also a test power limit. Consequently, the maximization of the level of test cost optimization necessitates the development of methodologies to find the optimal schedule of the concurrently tested domains under architectural and power constraints.

As discussed in Section 1.1.3, a conventional scan-based test architecture substantially increases the delivery cost of an individual test vector due to serial access mechanism, setting a hard limit on test cost reduction. Test vectors consist of a large set of unspecified bits, resulting in the use of valuable test resources for the delivery of non-essential test bits. A possible reduction of the highlighted inefficiency in test vector delivery can have a proportional reduction on overall test cost. Subsequently, in order to alleviate a major limitation on test cost optimization, test architecture advancements are required to enable the efficient delivery of test vectors through the reduction of gratuitous test cycles.

1.2.3 Algorithmic Challenges

An adaptive test cost and quality optimization methodology development poses a number of algorithmic challenges, partially driven by the defect characterization and functional constraints. Since the defect characteristics shift throughout the production life cycle, the type of the drift has a profound effect on the adaptive test methodology, imposing a set of algorithmic challenges for test effectiveness learning and defect characteristic tracking. In the initial deployment phase of a new process node and during the product ramp-up period,

the defect characteristics are exposed to frequent and sharp drifts as a result of frequent tweaking of the manufacturing process parameters in order to increase the manufacturing yield. Once the process matures, the defect characteristics are however mostly stable or very slowly drifting, only infrequently punctuated by sharp changes, primarily from lot to lot.

An adaptive test optimization strategy, particularly for defects that are random in nature, aims at assessing the effectiveness of test vectors based on the gathered test data, extracting the underlying model of test effectiveness. The allowable change in test flow during adaptive optimization to overcome the test data gathering constraints depends considerably on the type of drift, heavily influencing the fidelity of the extracted model of test effectiveness. Mostly stable defect characteristics punctuated by infrequent sharp drifts as observed in mature processes permit a more intensive learning process and aggressive changes in test flow in the process of the extraction of test effectiveness model. Although the model may suffer inaccuracies in the earlier stages of the test effectiveness learning, long stretches of stable defect characteristics provide an opportunity to converge to a stable model of test effectiveness. However, subsequent to the infrequent sharp changes in defect characteristics, much of the previously collected information that is still relevant needs to be retained to prevent the repetition of a lengthy learning process, necessitating an incremental update of the model to reflect the changes and the preservation of the fidelity of the model in the process.

The frequent and sharp drifts in defect characteristics as observed in the product ramp-up period however do not allow the application of a lengthy learning process as a result of rapidly changing test effectiveness, necessitating a quick test effectiveness learning process at a cost of a sub-optimal test effectiveness assessment. Furthermore, the necessity for preserving the fidelity of the learned test effectiveness model precludes aggressive dynamic changes in the test flow during the learning process due to the frequent drifts in defect characteristics, requiring a relaxation of test data gathering constraints through a slightly

expanded test data collection at a minuscule increase in test cost. Consequently, the adaptive test optimization strategy needs to strike a balance between efficiency and effectiveness while ensuring the preservation of the fidelity of the learned test effectiveness model.

A fundamental component of an adaptive test optimization strategy is the method for the tracking of defect characteristic drifts. The type of drifts similarly plays a crucial role in determining the constraints of the tracking method. The stability characteristics of the underlying model of test effectiveness that is learned based on the analysis of the gathered test data is a powerful indicator for defect behavior shift for the type of drifts where infrequent sharp changes follow the long stretches of mostly stable defect characteristics. Subsequently, an efficient and accurate methodology is required to measure and track the stability of the test effectiveness, highlighting sharp fluctuations in the stability characteristics. Frequent and sharp drifts in defect characteristics on the other hand impose distinct challenges. Due to frequent changes in the stability of the underlying model of test effectiveness, a balance needs to be struck between the accuracy of the test effectiveness model and the fast adaptivity to the frequent drifts. The larger the set of test data used for the extraction of the test effectiveness model is, the slower the reaction to the frequent drifts in defect characteristics is. Conversely, the use of a smaller set of test data diminishes the accuracy of test effectiveness analysis.

In sum, adaptive test cost and quality optimization strategy not only needs to be tailored based the different classes of defects as discussed in Section 1.2.1, but also should address the functional constraints and algorithmic challenges as outlined in Sections 1.2.2 and 1.2.3, respectively.

1.3 Thesis Summary

In this thesis, we address the urgent necessity for an effective yet efficient VLSI test process in the intensely competitive semiconductor industry. An adaptive test cost and quality

optimization framework consisting of various novel techniques and test architecture advancements is proposed to meet the customer expectations in test quality while bounding the test cost within an economically acceptable level. We question the feasibility of the conventional test flows to deliver the demanding test cost and quality goals as the defect behaviors evolve and as the variation within and across chips increases with each DSM process node, advocating instead a fundamental paradigm shift in VLSI test. Namely, as discussed in Section 1.1, the conventional way of the continuous piling of test vectors from various fault models and the identical application of the resulting test set to all chips with no consideration of chip and domain characteristics fails to satisfy the evolving needs of VLSI test, resulting in an inefficient utilization of test resources and struggling to deliver the acceptable quality goals within the limited test cost budgets. The proposed methods have been tailored to address a broad set of application scenarios through the consideration of different defect classes and defect characteristic drift types while overcoming the data gathering and delivery method constraints and the algorithmic challenges as outlined in Section 1.2.

The proposed techniques capture the criticality of not only fault models and test vectors but also individual chips and frequency domains and, subsequently, exploit the variation in the criticality to deliver an effective yet efficient test through an adaptive selection of most effective vectors and a carefully crafted variable allocation of test resources. Furthermore, a novel scan-based test architecture is proposed to enable a cost-effective application of the resulting test set.

The first set of the proposed methods tackles the challenging goal of the adaptive test effectiveness assessment for particularly the defect types that are random in nature and the subsequent selection of the most efficient test set while dynamically adjusting the test effectiveness information and the resulting test set as the underlying defect characteristics shift as discussed in Section 1.1.1. Since not only the effectiveness of fault models but also the

effectiveness of the vectors within a fault model can vary due to the systematic defects and the variations in manufacturing process, an effectiveness assessment at an individual vector level is necessary for an optimal exploration of test cost and quality tradeoff. An adaptive individual test effectiveness assessment methodology that overcomes the data gathering constraints of the conventional test flow by allowing a dynamic re-ordering of test vectors in the test flow is subsequently proposed. The proposed methodology prioritizes the test vectors according to their defect detection effectiveness based on the real time feedback from the production flow. The stability characteristics of test vector order is utilized not only to check the fidelity of the test vector priority information but also to track the defect characteristic drifts, enabling adaptivity to the drifts. A number of algorithmic advances subsequently enables the exploration of test effectiveness and efficiency tradeoff not only at a single test level but also across the multiple test levels through the utilization of the vector priority information as a foundation stone.

While the effectiveness assessment at the individual vector level enables a fine-grained optimization, the intensive learning process makes it more suitable for mature processes with infrequent sharp drifts in defect characteristics. In the production ramp-up phase and at the earlier phases of each new process node, frequent and sharp drifts in defect characteristics are observed as the manufacturing process is continuously adjusted to eliminate systematic issues and subsequently improve the yield as discussed in Section 1.2. In order to effectively optimize the test cost and quality for this type of drifts, a test effectiveness assessment methodology that can quickly learn and adjust is crucial, promoting a need for a coarse-grained yet fast fault model level effectiveness assessment methodology while maintaining the fault coverage based vector order within the test set of each fault model in order to ensure the fidelity of the learned test effectiveness information. Subsequently, an adaptive methodology that represents and continuously adjusts test quality as a function of

fault coverages of multiple fault models based on the failure information from a small set of recently tested defective chips is proposed. The data gathering constraints of the test flow are slightly relaxed to obtain first failing test information per fault model at a minuscule increase in test cost. The continuously updated representation of test quality as a multi-variate function of fault coverages enables the selection of an optimal mixture from various fault models while quickly adapting to the defect characteristic shifts through the utilization of the failure data from a small sample of recently failed chips. The selection of sample size is carefully orchestrated to address both fast adaptivity and test effectiveness assessment accuracy requirements.

The second set of proposed methods addresses the question of the optimal use of test resources through a carefully crafted allocation of delay test resources based on chip and domain characteristics as discussed in Section 1.1.2. The proposed methods exploit the regularity of the delay defects to incorporate a pre-silicon test effectiveness analysis into the test cost and quality optimization framework. First, a chip-specific test resource allocation method that captures the effect of the individualization of chips on test quality through a pre-silicon statistical analysis of the test quality changes across the process variation space is proposed. An analysis through the use of the resulting statistical test quality model subsequently identifies the test resource allocation for each chip based on its position in the process variation space, enabling an adaptive selection of test resources by extracting process information from the chip under test.

The coexistence of the hundreds of domains with distinct frequencies within the same chip necessitates not only an effective allocation of delay test resources based on domain characteristics to optimize the test quality but also a careful exploitation of the concurrent test support. While concurrent test is capable of delivering substantial test cost reduction, the level of attainable concurrency is bounded by not only the architectural concurrency constraints but

also the constraints on simultaneous power utilization. A test resource allocation and domain scheduling method, that simultaneously identifies the best allocation of test resources and the schedule of which domains are to be tested in parallel while complying with architectural and power related concurrency constraints, is subsequently proposed. An optimization formulation as well as efficient algorithms based on convexity and fast concurrent test scheduling techniques through the utilization of test quality estimation based on domain characteristics are provided.

Finally, a novel scan architecture, denoted as *CircularScan*, is proposed to alleviate the test delivery constraints of a conventional scan architecture, reducing the exceedingly high test delivery cost due to the serial access mechanism through the compression of test patterns. The proposed architecture exploits the low specified bit density of scan-based test patterns, enabling a fast application of the specified bits through a circular configuration of scan chains while utilizing only a small set of scan I/O pins. The circular configuration of scan chains enables the use of the captured responses of the previously applied pattern as a template for the subsequent test pattern. A small set of scan I/O pins is utilized as an addressing mechanism to pinpoint and quickly update the positions on the template that conflict with the corresponding specified bits of the next pattern, substantially reducing the test application cycles required to apply a test pattern.

We start the presentation of proposed test cost and quality optimization framework with an overview of the related work in Chapter 2. Chapter 3 presents the adaptive individual test effectiveness assessment and the subsequent effectiveness and efficiency exploration process. The multi-variate test quality tracking as a function of fault coverages and the adaptive selection of the best test mix from multiple fault models is presented in Chapter 4. Chip-specific test resource allocation based on chip characteristics to maximize delay test quality is presented in Chapter 5. The test cost and quality optimization across multiple

frequency domains through delay test resource allocation and concurrent test scheduling based on domain characteristics is presented in Chapter 6. The *CircularScan* test architecture for test compression is presented in Chapter 7. Conclusions are drawn in Chapter 8.

Chapter 2

Previous Work

While the higher level integration and continuous process scaling enables the increasingly complex and high performance designs, the accompanying necessity for rigorous test strategies in order to achieve a satisfactory test quality at an acceptable test cost continuously challenges academia and industry alike to develop innovative test methodologies. The development of new fault models and test generation techniques has become the main vehicle in the test community to improve test quality. Stuck-at tests [6], [7], which assume that each net in the design can permanently get stuck at logic 0 or 1, and IDDQ tests [8], which evaluate whether the steady state current (i.e. quiescent current) of the chip is within the expected level, have been in production test flows for a long time.

Bridging fault model based tests [7], [9], which assume the existence of possible shorts between the nets in the design, are frequently included in test flows, albeit in a limited capacity due to the exceedingly large set of possible bridging fault candidates. Delay tests [10], which target timing related defects, have become an indispensable part of test flows in recent years. The transition fault model [10], [11], which assumes that each net in the design can have a delay defect that effects the timing of rising and falling transitions, and the path delay fault model [12], which specifies the exact paths and tests the cumulative effect of possible delay defects on these paths, are the most commonly used delay test fault models.

Due to the sheer volume of possible paths in a large circuit, only a small set of critical paths can be typically targeted by the path delay test, aiming at identifying the chips at the tail end of the process variation distribution. Transition fault test remains the main test used to target the random delay defects due to its topological coverage of the design and its compact test set size although effectiveness suffers at small delay defect detection as test generation tends to favor short activation and propagation paths.

As the delay defect occurrence frequency exacerbates with continuous process scaling, various new delay test generation techniques have been recently proposed to improve transition delay test quality. Timing-aware delay test methods [13], [14], [15], [16], [17] utilize timing information during test generation to test the faults through the longer paths, increasing small delay defect coverage. Faster-than-at-speed delay test methods [18], [19], [20] run the tests at a speed higher than the target frequency to reduce the timing margins, consequently increasing the small delay defect coverage.

Furthermore, in order to alleviate the issue of the lack of a direct correlation between faults and defects that is inherent in high level fault model usage for test generation, new test generation methods that aim at boosting the likelihood of defect detection have been proposed. N-detect test generation method [21], [22] strives to detect each target fault such as stuck-at and transition multiple times, increasing the possibility of fortuitously activating a possible defect located at the target fault location. Gate-exhaustive test generation [23], [24] aims at exhaustively covering all input states of each cell, boosting the coverage of intra-cell defects.

While the emergence of new fault models and test methods helps to boost test quality, test cost is inevitably increased in face of continuous piling of new test types. Not only the sheer number of test vectors, but also the size of each test vector is continuously growing due to the increasing size of SOCs. The taming of ever increasing test cost has become one of the

focal points of research in the test community. Reductions in the run time of each individual test vector or in total test vector count can yield corresponding reductions in test cost.

Automatic test pattern generation (ATPG) methods [6], [10], that are utilized in fault model based test generation, target a fault and generate an initial test pattern (denoted as a *test cube*) while only specifying the bits required for the fault detection, leaving the majority of the bits unspecified. A subsequent test cube compaction process [6] that aims to reduce the number of test patterns by merging test cubes has been a standard component of ATPG tools for a long time. Test cube compaction methods can be classified as static and dynamic compaction methods. Static test cube compaction [25], [26], [27] commences subsequent to the completion of test generation, merging compatible test cubes and also eliminating the redundant ones. Dynamic test cube compaction [28], [29], [30] is concurrently performed with the test generation process, enabling the generation of test cubes that are compatible with the existing ones and also paving the way for the fault simulation of the merged test cubes during test generation in order to identify the additional faults detected.

In addition to the customary test compaction methods, various techniques have been proposed to tame the test cost. Test compression techniques in scan-based test architectures (referred to as scan compression techniques hereafter) have drawn tremendous attention in the last decade and have shouldered the responsibility of delivering a major reduction in test cost. Static test optimization techniques through various approaches such as effective test generation, test selection and vector ordering continue to emerge at a steady pace. Adaptive test optimization techniques are recently gaining popularity in test cost and quality optimization. An overview of scan compression techniques is provided in the next section. Static test optimization techniques are briefly summarized in Section 2.2. Adaptive test optimization techniques are discussed in Section 2.3.

2.1 Scan Compression

Scan test compression techniques came into prominence in the last decade as scan-based test architectures have gained a wide acceptance. Numerous combinational and sequential scan compression methods have been proposed, commercialized by EDA vendors and are commonly used in industry ICs. Research in this area continues to strive for more aggressive scan compression techniques. A survey and a study of the historical evolution of scan test compression techniques can be found in [31], [32].

Scan test compression techniques can be broadly divided into 3 categories; namely, code-based, broadcast-based and linear-decompressor-based schemes [31]. Code-based schemes typically employ off-the-shelf data compression techniques to compress test data. Various compression algorithms such as statistical coding [33], Golomb code [34], Huffman code [35], [36], [37], frequency-directed run-length code [38], [39], nine-coded compression [40] and multi-dimensional pattern run-length code [41] have been explored. These methods exploit the variable occurrence frequencies of test pattern blocks to efficiently compress the test data while the decompression is being performed on chip.

Broadcast-based schemes rely on the concept of broadcasting scan-in data to a larger set of internal scan chains. A single pin is used in [42] to broadcast the test data to multiple scan chains and the remaining undetected faults are tested by reconfiguring the scan chains as a single chain. The use of two different scan chain configurations is proposed in [43] to improve on [42] and a feedback architecture is presented in [44] to eliminate the serial vectors. The methodology in [45] provides multiple broadcast configurations, wherein the configuration can be changed at each test cycle. [46] extends the concept of broadcast to the scan chain segments by constructing tree-shaped scan designs, wherein the scan data is broadcast at each tree node to the branches of the node.

Linear-decompressor-based schemes utilize combinational or sequential XOR networks to compress the scan data. Prominent examples of this widely used technique can be found in [47], [48], [49], [50], [51], [52] which use either combinational XOR or sequential LFSR- based networks to encode the test data as a linear combination of input scan data.

The scan compression scheme we propose in Chapter 7, *CircularScan*, follows a significantly different approach. The scan chains are configured in a circular form to enable the use of the captured response as a template for the next pattern. The scan inputs are used as an addressing mechanism to update the specified bits of the next test pattern that are at conflict with the template. The use of captured response as a template significantly reduces the number of specified bits that need to be updated and the proposed optimized addressing scheme provides a quite efficient method to update the conflicting bits. Although the proposed scheme is generated by analyzing a predetermined test set, it is flexible enough to support the application of any random test pattern. This property of the proposed method distinguishes it from the previously proposed test set dependent methodologies, wherein test pattern updates may necessitate reconstruction of the design or result in an inability to supply the test vectors. A number of researchers have proposed methodologies that utilize similar configurations. Random Access Scan [53], [54], [55], [56] enables the individual addressing of each scan cell, at a more fine-grained control than *CircularScan*. The major disadvantage of [53], [54], [55], [56] is the significantly increased hardware cost for the individual addressing of the scan cells.

2.2 Static Test Optimization Techniques

The continuous inflation of test vectors included in test flows as result of piling numerous test types triggered the emergence of a number of static test optimization techniques

through various approaches such as effective test generation, test selection and vector ordering.

While N-detect and gate-exhaustive test generation methods increase the possibility of fortuitous defect detection and consequently test quality, the test vector count is significantly increased. Embedded multi-detect (EMD) test generation [57], [58] has been proposed as an alternative to N-detect. EMD enables the multi-detection of faults during regular single-detect test generation by exploiting unspecified bits. Although a particular multi-detection level is not guaranteed for each fault, the maximization of multi-detect improves the test quality while preventing a significant increase in test cost.

Cell-aware test generation flow [59], [60], [61], [62], [63], [64], that has been proposed as an alternative to gate-exhaustive test generation, initially utilizes the layout information of each cell, extracting possible intra-cell defects and subsequently selecting the cell input states that are necessary to cover these particular defects. The cell-aware test generation subsequently tries to generate only the selected cell input states, reducing the test generation complexity and test set size in comparison to gate-exhaustive test generation that aims to cover all cell input states.

Delay test generation methods such as timing-aware test generation improve the quality of the transition test set by testing the faults through longer paths, albeit at a substantial increase in test cost. Delay test pattern selection methods [65], [66] start with a large set of test patterns generated through N-detect or other means and subsequently select a small subset of patterns that detects the faults through longer paths. The pattern selection methods can account for process variation as well [67], [68], aiming to increase the test quality throughout the process variation space. Similarly, path selection methods [69], [70] for path delay testing to increase the process variation space coverage have been proposed. These methods terminate pattern and path selection when a certain threshold is exceeded and the resulting test set is

applied identically to all devices. Although these techniques increases test quality, they do not consider the delay resource allocation tradeoffs among chips and frequency domains, failing to extract the highest test quality from the available resources.

A number of previously published methods attack the optimal test vector ordering problem in order to reduce the average test time to detect the defective device by using the complete test vector failure information 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 [71]. The effect of the ordering of test types on test cost is analyzed in [72]. The analysis is constrained to examining the impact of the ordering of different test types, but does not extend to the question of individual vectors. These methods require costly pass/fail information collection for the full test set and, most importantly, they fail to adapt, instead adhering to a fixed, statically determined, initial vector order throughout.

2.3 Adaptive Test Optimization Techniques

Adaptive test techniques dynamically adjust the test set during production testing, aiming at selecting the optimal test set based on the evolving defect characteristics. Earlier work in this area primarily takes place in the parametric test domain [73], [74], [75], [76], [77], [78], [79], [80]. The applicability of adaptive test to the parametric test domain is perhaps to be expected as the domain offers a measurement which enables correlation analysis. Parametric test measurements are collected from a sample of devices and a correlation analysis is performed between the tests to select the optimal test set per lot.

Adaptive test development in the structural test domain is traditionally limited as no data other than a discrete, binary pass/failure information is conventionally available. [81] proposes the identification of the failure mechanism based on diagnosis and changes the test content periodically using a test quality estimation based on the diagnosed failure

mechanisms. It requires a costly diagnostic run for each chip throughout production testing, depending heavily on the accuracy of diagnosis, and is limited to a small set of defect classifications. Adaptive delay test development has primarily focused on path delay test. Different path delay test sets based on the process parameters are generated and adaptively applied during testing in [82] and the optimal test frequency per chip for path delay test is adaptively identified in [83] to minimize yield loss. These methods help to screen out the devices that fail to meet the system frequency target due to process variations, but fall short of providing the intended coverage for random delay defects.

A significantly distinct direction for adaptive test consists of the outlier identification techniques which aim at adaptively pinpointing the boundary between the good and defective devices. They are usually applicable once again to parametric tests. Instead of making a pass/fail decision based on a predefined threshold during run time, the test data is collected and statistical post-processing techniques are used to identify the defective devices based on the test data distribution. IDDQ testing has been the primary focus [84], [85], [86], [87], [88], [89] of this set of techniques as the boundary between good and bad leakage current measurements increasingly blurs. Analog tests [90], [91], [92], [93], [94] are getting increasingly targeted by statistical methods and there have been attempts to apply them to delay testing [95], [96], [97], [98]. This group of techniques usually do not attempt to dynamically select an optimal test set but try to increase test quality for a given test set by cumulatively analyzing the test data.

Chapter 3

Adaptive Test Optimization through Test Effectiveness Assessment

While the piling of various new fault models in production test suites helps to keep the defect escape level in check, it also frequently results in the inclusion of various ineffective test patterns with no significant concomitant defect coverage benefit as discussed in Chapter 1. The use of high level fault models in test generation with indirect correlation to defects at best and the defect coverage overlaps among numerous test types constitute the primary causes of the ineffective patterns. The necessity for an effective yet efficient test suite requires an understanding of the effectiveness of tests in defect detection and a subsequent test cost and quality optimization. The challenge of test cost and quality optimization is compounded by the fact that test is performed at multiple levels such as wafer sort and package test, imposing the question of the appropriate use of test resources at not only a single test level but across multiple test levels.

The static derivation of optimized test sets through the modeling and simulation of all possible defects, subsequently leading to the elimination of ineffective vectors, imposes inordinate challenges in practice. Furthermore, the continuous shift in the underlying defect mechanism from lot to lot throughout the production life cycle [73] causes corresponding changes in test effectiveness, precluding a static derivation of an optimal test set.

The simultaneous demand for efficiency and effectiveness is addressed in this chapter through an adaptive test flow that dynamically assesses the individual test vector utility through the prioritization of vectors in terms of defect detection effectiveness based on the real time feedback from the production flow. While the prioritization of test vectors lends itself to exploitation for delivering a partitioning into the two sets of effective and ineffective vectors, this may be a somewhat circuitous route for achieving the stated aim. Yet numerous advantages recommend the use of the prioritization approach, primary among them being the challenging issue of dealing with defect characteristics shifts. The issue of tracking the appropriate test sets even when defect manifestations change necessitates a balance between the need for test set modification to reflect the new defects and the desire to retain as much of the previously collected information which may still be appropriate. A priority order which can be somewhat modified by inserting trailing vectors early back in the order enables the necessary dynamic update of the test set to be eventually reflected while largely retaining the stable information previously collected.

The proposed work knits together a number of algorithmic advances in order to deliver a variety of solutions that can be utilized in different parts of the test flow in numerous industrial settings. The earlier mentioned prioritization of test vectors constitutes a foundation stone that delivers a sharply graded and descending utility ordering of the test vectors based on the learning of defect detection effectiveness. This information can be utilized to deliver sharp reductions in test cost while delivering a user set threshold in DPPM. For industrial environments wherein DPPM levels remain nonnegotiable, the prioritization order enables a sharp reduction in overall test cost by distributing the resources between multiple test flow levels appropriately. The underlying complex tradeoff is explored through a cumulative analysis of test effectiveness & efficiency tradeoff across test levels.

Conventionally, due to test data collection restrictions of the high volume production test environment, only the first failing vector information is typically available. The proposed test effectiveness process relies solely on this limited failure information that is traditionally available. The effectiveness of the vectors is assessed by dynamically altering the order of vectors in the test flow to prioritize them based on the defect detection information collected so far. Vector priority is continuously updated throughout the production life cycle, effectively tracking the changes in underlying defect types. The proposed work addresses the issue of culling ineffective vectors even in the face of versatile defect characteristics from the typically sizable predetermined test set, thus delivering test application efficiencies. The proposed test effectiveness assessment and subsequent ineffective vector elimination process aims at attaining a similar quality as the original test set at a substantially lower test cost.

Section 3.1 presents the motivation for the proposed technique. Section 3.2 provides a brief overview of the proposed method while the dynamic learning of individual test vector effectiveness is presented in Section 3.3. Test cost optimization by utilizing the test effectiveness information and adaptivity to defect mechanism changes are presented in Section 3.4. Section 3.5 discusses the experimental results in detail and conclusions are drawn in Section 3.6.

3.1 Motivation

Although the inclusion of new test types improves the overall effectiveness of test suites in defect detection, a significant level of ineffective individual test vectors exists in the test sets [5], [73]. Test generation is typically performed at a fault model level due to challenges in modeling and generating targeted tests for all possible defects. Test generation based on the simplistic fault models such as stuck-at with no direct correlation to defects

frequently underlies the multitude of ineffective patterns in test sets. Inclusion of a variety of fault models such as stuck-at, IDDQ, transition and path delay in the test suites is another main driver of the increasingly higher level of ineffective patterns. Defect coverage correlation models among different fault models are typically lacking, resulting in a tremendous amount of defect coverage overlap between the fault models. 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 methods 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. As new fault models and the corresponding test types are continuously added to test suites to ensure that the target DPPM level is achieved, it can be expected that the inefficiency in test sets will grow further in the future.

Conventional design of experiments (DOEs) as depicted in Figure 1.2 help to evaluate the effectiveness of fault models but fall short of helping with the individual test vector effectiveness assessment. Individual vector effectiveness assessment requires that test effectiveness analysis through the collection of defective device detection information should not terminate at fault model level as in Figure 1.2 but rather needs to descend further to the individual test vector level. The challenges of the more aggressive analysis at the individual vector level have traditionally been data availability limitations and the possibility of defect mechanism changes throughout the product life cycle, necessitating continuous failure data collection and monitoring.

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. Higher level parallelism through multi-site testing weakens the applicability of the stop-at-first-failure approach yet the cost and the challenges associated with the collection and

management of full failure information, particularly in external test houses, frequently reduce the collected information to the binning of the failing devices based on the first failing vector. Consequently, in order to develop a methodology that is applicable to both conventional single site with stop-at-first-failure and multi-site wherein only first-failure information may get recorded, in this work, we assume the constrained case of limited information availability (i.e. first failing vector) and develop a test effectiveness learning based on this stringent case. Industrial practices capable of delivering increased failure information can be readily incorporated to the proposed framework, thus strengthening further the quality of the delivered results.

Figure 3.1 Test vector effectiveness

In conventional static test flows with only first failing vector data for each defective device, the available information can be depicted as in the hypothetical example in Figure 3.1.(a), wherein each bar shows the additional defective devices detected by the corresponding vector. The continuous vector prioritization based on the effectiveness in defect detection may result in a new vector order with a defect detection profile as depicted in Figure 3.1.(b). Benefits of the continuously updated prioritized test vector order include:

- Reduction in average time for defective device detection
- Increased efficiency in ineffective vector elimination
- More efficient test time/quality tradeoff in multi-level test flows
- Continuous tracking of defect type changes

Since test vectors are prioritized based on cumulative defect detection effectiveness, the average time to reach the first failing vector is also reduced, improving test time for defective devices in stop-at-first-failure test flows.

Additionally, the ineffective vectors whose contribution to defect detection is minuscule due to the defect detection overlap with other vectors are pushed to the tail of the vector priority list, introducing the possibility of eliminating a large set of vectors, subsequently reducing test time for both defective and good devices in single site and multi-site flows.

Prioritization furthermore enables a more efficient test cost optimization in multi-level test flows, wherein test cost vs. quality tradeoffs can be exploited across test levels based on test economics in order to reduce the overall test cost with no sacrifice in test quality. In a test strategy with various test levels (e.g. wafer sort and package test), typically the later a defective device is detected in the multi-level flow, the higher the cost substantially is. However, a better prioritization of vectors in defect detection effectiveness at each level enables the elimination of ineffective vectors more aggressively while incurring only a small additional defect escape rate to the next level in the test flow.

Finally, the continuous update of the test vector priority information based on the evolving defect types throughout production testing enables the tracking of defect mechanism changes. Subsequently, test cost and quality tradeoff decisions effectively follow the current defect types.

3.2 Approach

In the context of the absence of global defect detection information for all vectors in the test flow, the main challenge in our pursuit of test effectiveness assessment is how to utilize the limited available information to move towards a prioritization of test vectors based on their effectiveness in defect detection. To overcome the limited information of the stop-at-first-fail approach, the proposed method *dynamically* alters the test flow during test application to extract more information for test effectiveness assessment thus converging eventually to a prioritized vector order. Fundamentally, the defect detection information of test vectors needs to be captured efficiently while a control mechanism is driving and monitoring the process of convergence to a priority order.

The learning of a prioritized order of vectors in defect detection paves the way for a variety of aggressive test cost reduction approaches. On one hand, ineffective vectors can be eliminated from the test flow based on a user set threshold in potential DPPM impact of the vector elimination. On the other hand, a test cost and quality tradeoff can be more efficiently exploited at different levels of a multi-level test flow to optimize overall test cost with no impact on the DPPM of the original test flow.

Nowadays, modern testers provide an application programming interface (API) support (e.g. test methods in Verigy testers or Visual Basic macros in Teradyne testers), enabling the implementation of an intelligent dynamic test program. User defined routines through API calls can be used to reorder and modify test patterns on the fly and perform complex computations. They are currently used for tasks such as data collection in memory, ATPG failure analysis and adaptive test flows. The proposed dynamic test flow for test effectiveness assessment and subsequent test cost optimization can be easily implemented by leveraging the outlined capabilities of the testers.

The only information available to the test effectiveness assessment process for a failing device is the order the vectors are applied in (i.e. current vector priority) and the first failing vector. Additional data points that can be derived from this information are the position of the failing vector in the test vector order and the stability characteristics of the test vector orders (i.e. how the vector priority is changing) during the learning process. The position of the failing vector can help to infer whether the defect is a hard-to-detect one, indicating that the vector is potentially an essential vector for defect coverage and can therefore benefit from moving to an earlier position in the test flow. The changes in test vector order over time (i.e. stability characteristics) can help to decide whether the test effectiveness learning process is converging to an optimized test vector order.

The failing vector position and the stability characteristics are effectively utilized to grade the effectiveness of the vectors and evaluate whether the current vector priority reflects the current defect population. The proposed test effectiveness learning process dynamically determines the order of vectors in the test flow based on the effectiveness information derived from failing vectors and receives updated information from the dynamic test flow based on this new vector order. The correlation of test vector orders during this process is utilized to determine when the learning process converges to an optimized test vector order.

Additionally, test vector order correlation information can be utilized to identify defect mechanism shifts, quickly adapting to the changes by adding back the eliminated vectors and subsequently converging to a new vector priority. In order to leverage the effectiveness information already collected for the vectors in the test flow, the priority for these vectors is initially preserved while intermixing the previously eliminated vectors, eventually converging to a new priority order as the vectors identify their position in the new overall vector order.

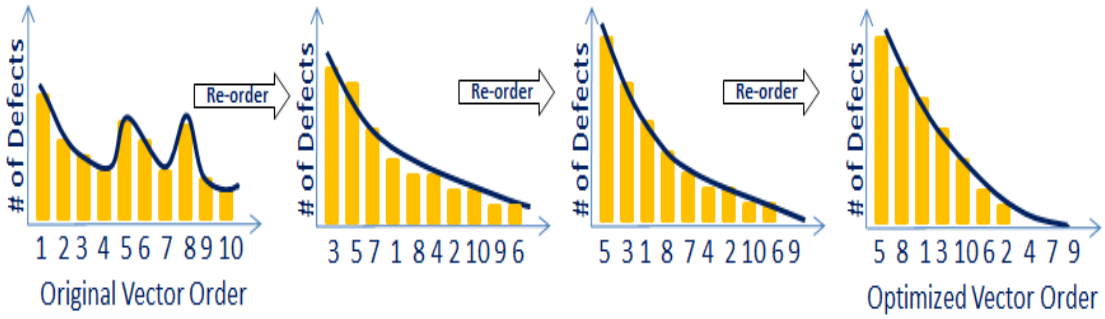


Figure 3.2 Dynamic test vector re-ordering

Upon the learning of test effectiveness, since the priority order is constructed based on vector failure information from the production flow, the failure rate for each vector is readily available from a large set of defective devices. A curve fitting to the failure rate data is utilized to access the DPPM impact of the elimination of vectors on the tail of the prioritized vector order. As depicted in Figure 3.2, the establishment of an improved priority of the vectors boosts the quantity of vectors that can be eliminated while retaining the same DPPM impact, thus delivering higher levels of test cost reduction.

In a distinctly different utilization of the test effectiveness information, test economics considerations enable the exploitation of the test cost and quality tradeoff efficiently in the earlier phases of the multi-level test flows while keeping the DPPM level delivered by the overall test flow intact. In a test flow with various test levels such as wafer-sort and package test, defect escapes from an earlier test level can be detected in a later test level, albeit at a substantially higher cost. Despite the high cost of defect escapes from a test level, if the test cost savings from a test set size reduction exceed the cost of detecting additional defect escapes at a later test level, the overall test cost of the multi-level test flow can be reduced with no impact on the final defect escape rate. A fine balance exists in this test cost and quality tradeoff to extract the highest level of test cost reduction benefit. A thorough mathematical treatment is subsequently provided in this chapter to find the optimal tradeoff

point in test cost and quality so as to maximize the overall test cost savings of multi-level test flows. Evidently, as a better prioritization of test vectors is obtained by test effectiveness assessment, a more aggressive test size reduction can be achieved while retaining the same defect escape rate to the next test level.

3.3 Test Effectiveness Assessment through Dynamic Test Flow

The proposed framework utilizes real-time defect detection effectiveness information provided by the production test flow to determine the test vector priority order. Due to the absence of full pass/fail information for all vectors, the proposed method relies on the limited information gathered so far to decide on the current vector priority order and collect new defect detection information for the resulting vector order. The vector priority is updated based on the new information and eventually converges to a particular test vector priority as this learning loop continues.

The defect detection information per vector is captured by keeping track of past defect detection data and rewarding the vectors that catch defective devices through a scoring mechanism. The stability characteristics of the test vector order are monitored by tracking the correlation of test vector orders.

The details of the scoring method for defect detection data tracking and test vector order correlation calculation are presented in Sections 3.3.1 and 3.3.2, respectively, and the test effectiveness learning algorithm that utilizes the scoring and vector order correlation is subsequently discussed in Section 3.3.3.

3.3.1 Test Vector Defect Detection Information

In pursuit of capturing the effectiveness of each vector in defect detection and the subsequent sorting based on this information, a natural approach, referred to hereafter as *position insensitive scoring,* would be to simply increment the priority score of the test vector that detected a defective device, thus maintaining a defect detection count for each vector irrespective of its position in the test flow for the failing device. As test cost is correlated directly with test time, the run time of each test should be taken into account, giving more credit to shorter tests (i.e. a vector with 100 ms of test time should not be accorded the identical score of a vector detecting the same defect in 10 ms of test time). The scores of the vectors are therefore normalized with respect to their test times, assigning weighted scores to the vectors.

Perhaps an aspect that is missed by the aforementioned *position insensitive scoring* is that the detection of a defective device by a vector that is late in the application order would hint at the possibility that this particular vector may be targeting a unique and hard-to-detect defect. Since this vector may be essential in catching hard-to-detect defects and needs to be in the effective vector list, test cost optimization can benefit from placing it in an earlier position in the test flow, helping to uncover other vectors that have defect coverage overlap by pushing them towards the tail of the vector priority list. To exploit this observation, we propose a new scoring method, referred to as *position sensitive scoring*, that 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, normalized with respect to the test time of the vector[1]. The two possible scoring methods are formalized as follows.

[1] To provide appropriate credit to the very first vector in the test flow when it fails, the test time of the failing vector is also added to the total elapsed time.

Position insensitive scoring: Given that the *i*th test vector, π_i, in the current test order detects the defective device and that the test time of this particular vector is *testtime*(π_i), the score of the vector is updated as follows:

$$score(\pi_i) += \frac{1}{testtime(\pi_i)} \qquad (3.1)$$

Position sensitive scoring: Given that the *i*th test vector, π_i, in the current test order detects the defective device, the preceding vectors in the flow are $\pi_1 \ldots \pi_{i-1}$ and the test time of the *k*th vector can be denoted as *testtime*(π_k), the score of this particular vector is updated as follows:

$$score(\pi_i) += \frac{\sum_{j=1}^{i} testtime(\pi_j)}{testtime(\pi_i)} \qquad (3.2)$$

3.3.2 Test Vector Order Correlation

The stability characteristics of the test vector orders in the process of test effectiveness assessment provide additional information available to guide and monitor the learning progress. The correlation of test vector orders during the learning process can be utilized to measure the stability of the vector orders. Since only the first failing vector information is available to the learning framework, only the score and position of this particular failing vector can change at the consecutive test vector orders. Subsequently, the correlation between two consecutive vector orders will not vary significantly throughout the test application, providing inadequate information. Instead we compute the correlation between the vector orders at the beginning and the end of a moving window of length, *W*, to sharpen the information content. This method allows us to track the correlation changes between the test vector orders within a specified time frame.

A widely used correlation metric, Spearman's rank correlation coefficient [99], as defined in Eq. (3.3) is used in the proposed method to measure the correlation of two different test vector orders, wherein n is the number of vectors and d_j is the difference in rank of the *jth* vector in these two test vector orders. A rank correlation coefficient of 1 indicates full correlation.

$$\rho_x = 1 - \frac{6\sum d_j^2}{n\left(n^2 - 1\right)} \tag{3.3}$$

Rank correlation coefficient, ρ_x, for the *xth failing device:* Given that π^x is the order of test patterns for the *xth* failing device and W is the width of the correlation window, ρ_x is the correlation coefficient computed by Eq. (3.3) between the pattern orders, π^x and $\pi^{x\text{-}W}$.

3.3.3 Test Effectiveness Identification

The proposed test effectiveness assessment process constitutes an incremental optimization method, striving to converge to the optimal vector priority order in incremental steps by making greedy decisions based on the available defective device detection information. It is well known that greedy decision based incremental optimization methods frequently display convergence to locally optimal but globally non-optimal solutions. Furthermore, the locally optimal solution converged to may display significant sensitivity to the initial order of the vectors. Test vectors of a fault model are typically ordered based on the coverage of this particular fault model and it can be expected that the same vector order serves as a good starting point in terms of defect detection effectiveness for this test set. However, since the proposed scheme, in its general form, is exploring test effectiveness across multiple fault models, the initial vector order such as the order that the test types are executed in may

govern the locally optimal solution that the learning method converges to. While the *position sensitive scoring* method is capable of making more drastic changes in the test order, nonetheless 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 avoid the locally optimal solutions.

We adopt an approach, known as ε-greedy [100], wherein non-greedy decisions are executed with a probability of ε. Based on the particularities of our problem, we divide the test effectiveness learning process into 3 phases; namely, *initialization*, *exploration* and *convergence* as depicted in Figure 3.3.

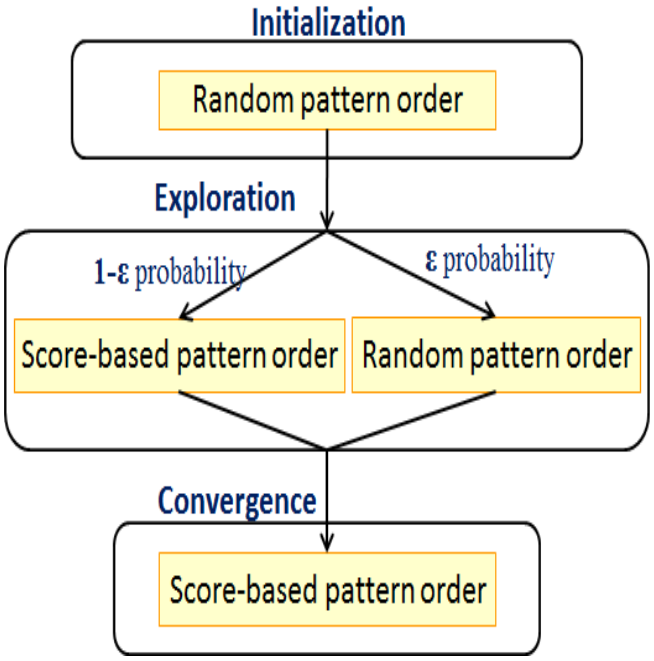


Figure 3.3 Test effectiveness learning phases

In the initialization phase, completely random vector orders are used to eliminate any bias to the initial test vector order by according equal chance to all vectors. Although the vector orders are random during this phase, the scores for each vector are collected based on

the scoring method described in Section 3.3.1 and used to create an initial order to build upon in the subsequent phases. In the exploration phase, random pattern orders are selected with a probability of ε; otherwise, the patterns are greedily sorted based on the scores collected so far. Essentially, better vector orders are obtained through the outlined scoring mechanisms; yet occasional random decisions are made to explore different vector orders to help overcome locally optimal solutions. Finally, in the convergence stage, the test vector order decisions are confined to being solely made based on the scores, gradually converging to a stable vector order.

The stability characteristics of vector orders, measured by the rank correlation coefficient as described in Section 3.3.2, are used to guide the test effectiveness learning process through the phases outlined above as depicted in Figure 3.4. As the correlation of vector orders increases, the test effectiveness learning process moves onto the next phase, eventually converging to a stable vector priority order.

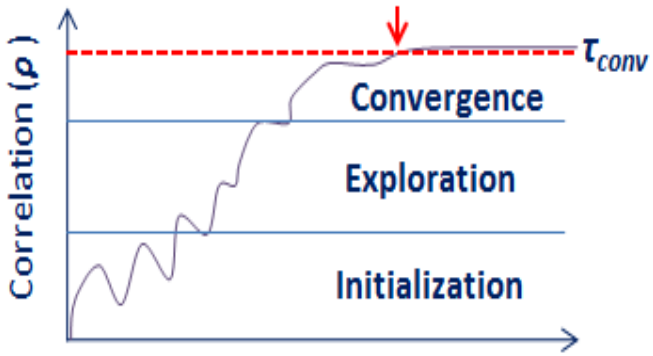


Figure 3.4 Correlation throughout the learning phases

A closer examination of this process reveals that the initialization phase is a special case of the ε-greedy exploration phase with an ε value of 100%. Similarly, the convergence phase is a special case with ε being 0%. Instead of utilizing a fixed ε between these extreme

points, reducing ε gradually in order to provide a smoother transition from initialization to convergence as depicted in Figure 3.5 can enhance the efficiency of the learning process, enabling the use of correlation as a guide for tracking the learning process. Essentially, ε is gradually reduced as the vector order correlation improves. Finally, when the convergence condition (i.e., $\rho > \tau_{conv}$) is satisfied, the test cost optimization process commences as discussed in the next section by utilizing the test effectiveness information gathered through this learning process.

ε-greedy Pattern Effectiveness Learning:

- As correlation increases:

    1. Gradually reduce random vector order selection probability of ε
    2. Sort the test vectors randomly with a probability of ε; else sort the vectors based on scores collected as a result of detecting defective devices

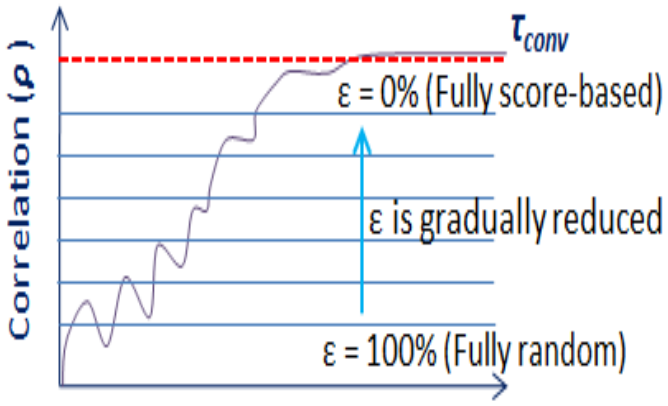


Figure 3.5 Correlation as ε is reduced

The effectiveness of the various techniques proposed in this section is experimentally evaluated in section 3.5 and their performance compared.

3.4 Adaptive Test Cost Optimization

A primary benefit of the learning of test effectiveness through the dynamic test flow as presented in Section 3.3 is the test time reduction for defective devices. Since the vectors are prioritized based on their effectiveness, the average time to reach the first failing vector for a defective device is reduced, resulting in test cost savings in stop-at-first-failure test flows. An additional and potentially more important benefit is that the prioritization of the vectors in terms of defect detection enables the identification of ineffective vectors by pushing them to the tail end of the vector list. The elimination of the ineffective vectors based on a set threshold in potential DPPM impact offers an opportunity for substantial test cost reduction. Furthermore, the test cost and quality tradeoff can be more efficiently exploited across test levels to optimize the overall test cost with no impact on the DPPM of the original test flow.

Since defect mechanisms are subject to change, test cost optimization needs to be adaptive. The proposed learning method is a continuous process, adjusting the vector priority as more devices are tested. However, the elimination of ineffective vectors introduces a new challenge. Although the test effectiveness assessment process continues for the vectors that remain in the test flow, the eliminated vectors may regain effectiveness in defect detection subsequent to a defect mechanism shift. In order to identify a new optimized test effectiveness order and also avoid potential test escapes, the defect mechanism shifts should be rapidly detected, thereupon initiating a process of reconvergence to a new optimized vector priority order.

3.4.1 Ineffective Test Vector Elimination

Upon convergence to an optimized test vector priority order based on effectiveness in detecting defective devices, the vectors with no or minimal defect detection benefit can be

easily identified. Since real-time failing vector information for defective devices is continuously fed back for test effectiveness assessment, the fallout rate for each vector (i.e. the number of failing devices divided by total devices tested) can be easily obtained, denoted as F(t) in Figure 3.6.

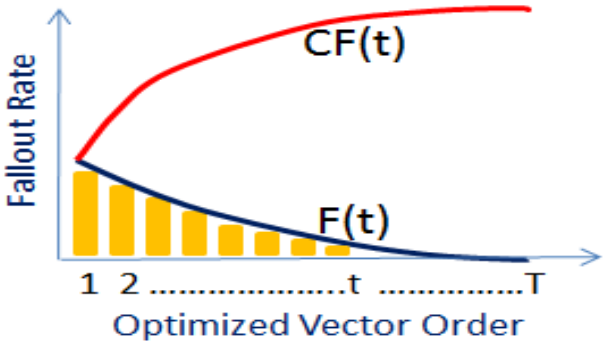


Figure 3.6 Vector failure rate and cumulative failure rate

The defective device escape rate when a set of vectors at the tail end of the vector order is eliminated can be estimated as a summation of each eliminated vector's fallout rate. Cumulative fallout starting from the beginning of the vector list can be obtained as well by adding the fallout rate of individual vectors, depicted as CF(t) in Figure 3.6. Given that T vectors exist in the test suite, the additional test escape rate can be estimated by Eq. (3.4) if the vectors at the tail end of the vector order list, starting from the t^{th} vector, are eliminated.

$$ER(t) = \sum_{k=t}^{T} F(k) = CF(T) - CF(t) \qquad (3.4)$$

The vectors on the tail end of the prioritized vector order can be eliminated based on the additional user set DPPM threshold by using the estimate given in Eq. (3.4). Since test suites include a large set of ineffective vectors, a significant amount of ineffective vectors at the tail end of the vector list can be eliminated with minimal impact on defect escape level.

3.4.2 Test Cost/Quality Tradeoff

In a distinctly different utilization of the test effectiveness information, the test cost can be sharply reduced in multi-level test flows based on test economics with no DPPM impact. Product testing includes various test levels, such as wafer-sort and package test. Devices go through these test levels, with only the passing ones moving to the next level. The ineffective vectors can be eliminated in earlier test levels based on the proposed effectiveness learning framework while retaining the full test set in the final test level to avoid any additional defect escape due to vector elimination.

If a defective device is not detected at wafer-sort but detected at package test for example, an additional cost is incurred due to the packaging and package test cost. However, if test cost savings in earlier test levels as a result of test set size reduction are higher than the cost of detecting the defect escapes in the later levels, an opportunity to reduce the overall test cost can be obtained. This tradeoff can be explored in many test levels, starting from first test level (i.e. wafer-sort) up to customer samples. Since it is easier to quantify the cost of a defect escape in a manufacturing test environment, we will focus on two earlier test levels; namely, wafer-sort and package test, in this section. The analysis can be extended to other levels, given appropriate test cost and test quality models.

The average test cost per device accepted by wafer-sort and package test levels depends on the test cost and also the yield of each test level. Since test time is spent for both good and defective device testing, the cost of testing a bad device is averaged out to good devices based on test yield. A detailed analysis of cost per accepted device, including manufacturing and test cost, can be found in [101]. We focus on test cost in this work. Given that C_{TW} and C_{TP} denote wafer-sort and package test cost, respectively, and Y_{TW} and Y_{TP}, yield for wafer and packaging, respectively, the test cost of a packaged device accepted by both test levels, TC_{Accept}, can be found by Eq. (3.5).

$$TC_{Accept} = \frac{\left(C_{TW} / Y_{TW} \right) + C_{TP}}{Y_{TP}} \qquad (3.5)$$

This cost model assumes that test time is identical for both good and defective devices. If the test flow is terminated at first failure, test time is reduced for defective devices. Assuming $E(C_{TW})$ is the expected wafer-sort test cost for defective devices that can be obtained by calculating the expectation from the fallout rate curve, the wafer-sort test cost can be determined by Eq. (3.6). Essentially, passing devices (wafer-sort yield) consume full test time but only a portion of the test set is used for the defective devices. A similar formulation can be derived for package test. Since the proposed learning method prioritizes the vectors based on their effectiveness, it reduces the expected test cost of defective devices, improving overall test cost. To simplify the presentation of the test cost and quality analysis, we will assume in subsequent analysis the identical test time for good and defective devices although this simplification underestimates a benefit of the proposed method in stop-at-first-failure test flows.

$$C_{TW} \times Y_{TW} + E(C_{TW}) \times (1 - Y_{TW}) \qquad (3.6)$$

If the test cost of wafer-sort is reduced by eliminating test vectors at the expense of test escapes, in order to model the total test cost, the additional packaging cost of the escapes needs to be considered in addition to the changes in wafer-sort test time and test yields. In our analysis, we assume that the final test is a superset of wafer-sort test, indicating that if a defective device can be detected at wafer-sort by the original test set then the final test can also detect this defective device. Package test is typically more stringent than wafer-sort so this assumption usually holds. However, if package test does not provide the same coverage as

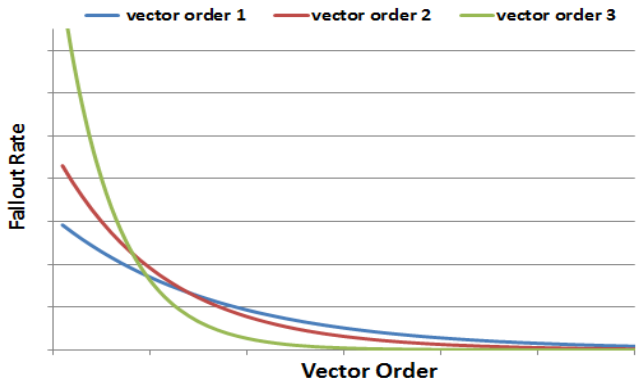
wafer-sort, a similar analysis still can be applied by including the coverage and the defect escape cost of package test in the analysis.

Assuming T vectors in the wafer-sort test set and the elimination of all vectors after the t^{th} vector in the vector order, the additional wafer-sort defect escape rate due to vector elimination can be estimated by Eq. (3.4). Test cost reduction due to vector elimination can be estimated by Eq. (3.7), wherein *TC(k)* is the cost of vector *k*.

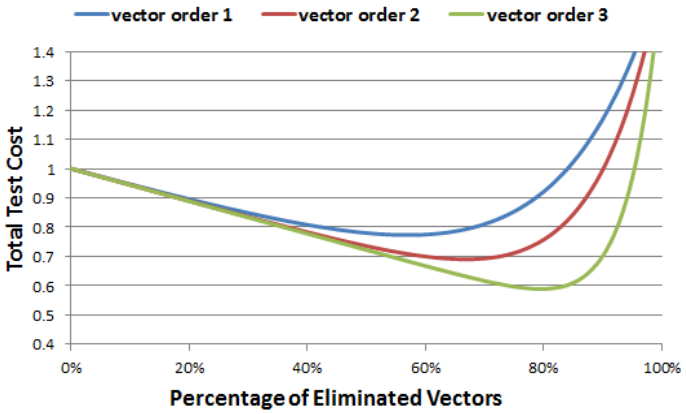
$$TR(t) = \sum_{k=1}^{t-1} TC(k) / \sum_{k=1}^{T} TC(k) \tag{3.7}$$

Although wafer-sort test cost per device is reduced by *TR(t)*, the packaging cost for each additional wafer-sort test escape, denoted as C_P, should be amortized over the accepted devices. Similarly, additional test escapes from wafer-sort as a result of vector elimination increase the observed wafer-sort test yield but reduce the package test yield. The updated cost model built upon Eq. (3.5) by including the effect of test vector elimination can be seen in Eq. (3.8). Wafer-sort test cost reduction and the accompanying additional packaging cost of the test escapes as a result of vector elimination in wafer-sort are reflected in the new model. Furthermore, wafer-sort yield is increased by the test escape rate; but since the wafer-sort test escapes will be propagated and caught in package test, the package test yield is accordingly reduced. It should be noted that the overall test yield upon completion of both wafer-sort and package test levels stays constant.

$$TC_{Accept} = \frac{\left(\frac{C_{TW} \times TR(t) + C_P \times ER(t)}{Y_{TW} + ER(t)} \right) + C_{TP}}{Y_{TP} \times \left(\frac{Y_{TW}}{Y_{TW} + ER(t)} \right)} \tag{3.8}$$



(a)



(b)

Figure 3.7 Test cost as more vectors are eliminated

As can be seen in the fallout curve in Figure 3.6, as the ineffective vectors are eliminated from the tail end of the optimized vector order, initially test escape is minimally effected while a substantial reduction in test size is obtained. However, as more vectors are eliminated, the fallout rate swiftly increases and consequently the cost of test escapes starts to offset the additional test time reduction benefit. This tradeoff is highlighted in Figure 3.7 for 3 distinct vector orders with varying fallout curves. The test cost curves in Figure 3.7(b) depict how the test cost for the accepted devices, calculated by Eq. (3.8), changes as more vectors are dropped (i.e. *t* in Eq. (3.8) gets smaller) from the tail end of the vector orders shown in Figure

3.7(a). The lowest points in test cost curves are the optimal vector elimination points to achieve the highest level of test cost reduction in the tradeoff space. The more optimized the vector order in Figure 3.7(a), the higher the corresponding achievable test cost reduction is as seen in Figure 3.7(b), highlighting the importance of the proposed framework for test effectiveness assessment based vector ordering.

3.4.3 Adaptivity to Defect Mechanism Shifts

Underlying defect mechanisms may change throughout the production life cycle, particularly for new lots [73], necessitating adaptivity to these changes. The proposed method detects defect behavior changes and takes appropriate actions to continue delivering the optimal cost test set while achieving target quality goals.

A new defect that is uniquely detected by test patterns that were found to be largely ineffective based on the previous failures is symptomatic of shifts in defect behavior. However, once the ineffective vectors are dropped, the uniquely detected defects by these particular vectors go unnoticed. Although this is a rather challenging problem, one crucial and helpful piece of information available is once again the correlation of the vector orders. As presented in Section 3.3, 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 majority, if not all, of the defective devices will still be detected, even after ineffective patterns are eliminated. However, since the current test order was constructed based on previously seen defects, 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 vector orders as a method to monitor defect behavior changes.

A correlation coefficient that falls below the defect behavior shift threshold (τ_{shift}) subsequent to a vector elimination is interpreted as a perturbation in the underlying defect

mechanism; all eliminated vectors are thereupon inserted back to the test flow. Test effectiveness learning resumes with the previously deemed ineffective test vectors included, subsequently converging once again to a new optimized order.

Defect mechanism shift detection criterion: Given that ρ is the rank correlation coefficient for the current test pattern order and the defect behavior shift threshold is τ_{shift}, all previously eliminated vectors are inserted back to the test flow if the following condition holds. Test effectiveness for the full test set needs thereupon to be fully re-learned.

$$(\rho < \tau_{shift})$$

The repetition upon the detection of a defect behavior shift of the full ε-greedy Pattern Effectiveness Assessment process, discussed in Setion 3.3, essentially resets the learning process and discards previously collected information. A case can be made though that the vectors in the optimized test flow represent the product of a laborious data collection process, albeit for a slightly shifted defect behaviour. Nonetheless, the gradualness of the defect behaviour shifts motivates the use of this order, at least as an initial order to help speed up reconvergence. The previously eliminated vectors, reinserted into the test set upon a defect behaviour shift, enjoy no such history of test effectiveness, since all information regarding them was discarded once they were eliminated. Their insertion back to the tail end of the vector order may result however in a sub-optimal solution as these vectors may not get the opportunity to move towards earlier positions in the vector order. In order to exploit the effectiveness information already collected for the vectors that are currently in the test flow and to explore the effectiveness of previously eliminated vectors in the detection of new defects, a modified version of the ε-greedy learning algorithm, wherein the previously eliminated vectors are randomly re-distributed with a probability of ε among the vectors currently in the test flow, is utilized.

Since the order of vectors currently in the test flow is initially preserved and the ε-greedy exploration is only performed for the previously eliminated vectors, this process both utilizes the already collected test effectiveness information and also provides an opportunity for the eliminated vectors to show their effectiveness with new defect characteristics.

3.5 Experimental Results

In this section, we evaluate the effectiveness of the proposed test cost optimization framework. An industry design is used in the experiments and a commercial ATPG tool is utilized to generate stuck-at and transition fault scan patterns. A randomly selected set of stuck-at and transition faults with varying ratios are injected to generate faulty design instances and the proposed method is applied on this set of patterns and faulty design instances. The experiments focus on the analysis of the effectiveness of the proposed method with different parameters.

The industrial design used in the experiments consists of approximately 1.1 million gates. Around 5,000 stuck-at and 11,000 transition fault scan vectors generated by a commercial ATPG tool are grouped into test vectors of 100 patterns each and are used in the experiments.

In the first set of experiments, the effectiveness of the proposed dynamic learning process is analyzed. Since ATPG tools are highly optimized to generate compact and efficiently ordered test sets for a fault model, it can be expected that the stuck-at and transition pattern sets generated for the experiment have been ordered based on fault coverage and that this particular pattern order can be used as an approximation of the optimal order for each fault model.

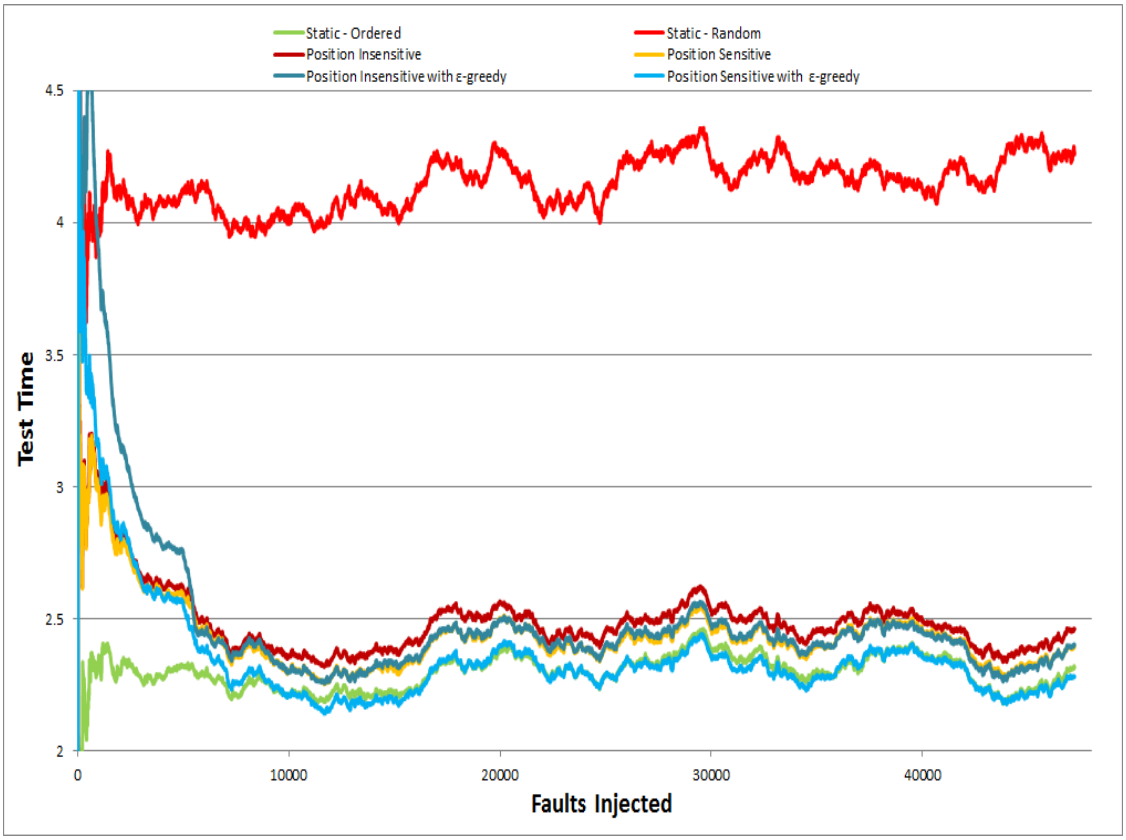


Figure 3.8 Test time change with various configurations of the proposed learning process

In this set of experiments, we focus on the stuck-at pattern set and analyze whether the dynamic learning process can approach the effectiveness of the original optimized order of the stuck-at test set. An initial random stuck-at pattern order is generated and the patterns are applied to the defective design instances created by injecting randomly selected stuck-at faults. The failing pattern information for each defective device is fed back to the proposed learning algorithm which is applied with different parameters, namely, position sensitive and insensitive scoring techniques with and without the ε-greedy learning process. 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 5,000 devices at each point during testing is reported. The average test time throughout the testing process as new faults are injected is plotted in Figure 3.8. The top and bottom curves represent the conventional static test flows

for the random pattern order and the original optimal pattern order, respectively. The average test time with the proposed learning process quickly improves on the random order and starts to approach the optimal order. Position sensitive scoring outperforms position insensitive scoring as anticipated and the ε-greedy learning process further improves effectiveness as can be seen in Figure 3.8. The ε-greedy learning process with position sensitive scoring quickly reaches the effectiveness of the optimal order and virtually matches it thereafter. This experiment shows that the proposed method is capable of learning the optimal test effectiveness. Since ε-greedy learning with position sensitive scoring delivers the best results, only the results for this particular configuration of the proposed test effectiveness assessment method are reported in the subsequent experiments.

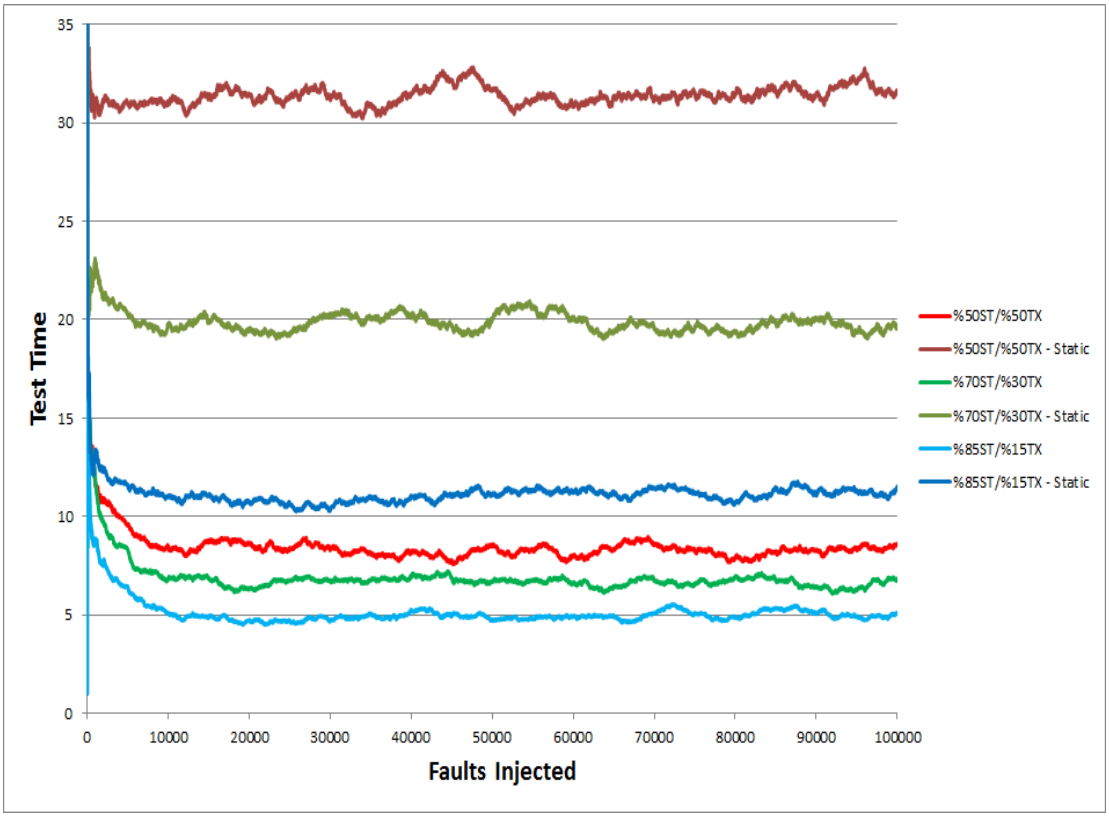


Figure 3.9 Test time change when the stuck-at tests precede the transition tests

In the second set of experiments, the effectiveness of the proposed dynamic learning method is analyzed for a test flow with multiple test types. Although ATPG tools aim at ordering patterns based on fault coverage, this optimization is at a single fault model level. Since our experimental setup consists of test sets of stuck-at and transition faults, a test flow is generated with these two test types. Initially, the test flow consists of a stuck-at test set followed by the transition test set and later the experiment is repeated with a test flow that consists of the transition test set followed by the stuck-at test set. Since the transition fault detection criteria constitute a superset of the stuck-at fault ones, in order to emulate defect coverage overlaps of test sets, the transition test set is fault-simulated for the stuck-at faults. When the transition test set is applied during testing, if the defective device has a stuck-at fault, it is checked whether it is detected by the transition test set to provide an appropriate score. Randomly selected stuck-at or transition faults are injected to the design to generate the faulty design instances in this experiment. Since the faults injected are selected from two fault models, a diverse set of stuck-at and transition fault ratios are evaluated. The 3 different configurations tried are 50% stuck-at (ST) / 50% transition (TX), 70% stuck-at (ST) / 30% transition (TX) and 85% stuck-at (ST) / 15% transition (TX).

The average test time for a test flow that consists of the stuck-at test set followed by the transition test set is depicted in Figure 3.9. In addition to the test time with the proposed test effectiveness assessment process, conventional static test flow results are provided as well. When the injected stuck-at fault percentage is higher, it is easier to detect the faulty devices; subsequently, the average test time is lower as expected. Up to a 3X reduction in average test time is achieved, depending on the ratio of fault types observed by the defective devices.

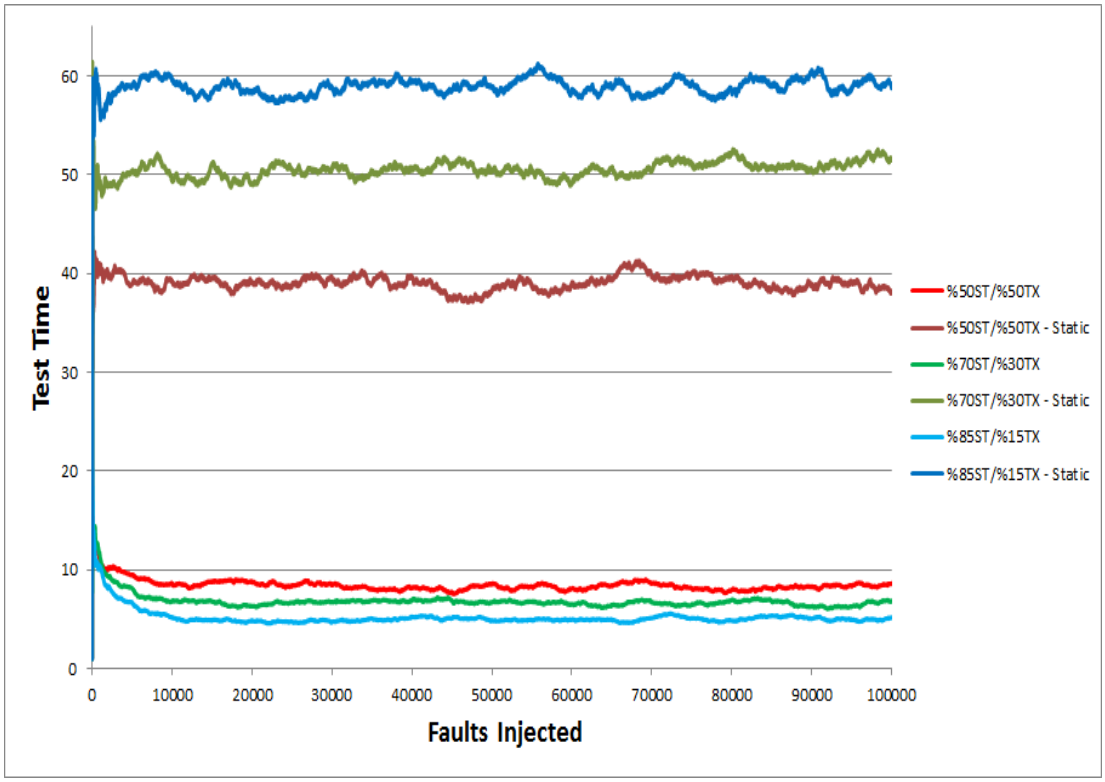


Figure 3.10 Test time change when the transition tests precede the stuck-at tests

An identical set of experiments is repeated for a test flow that applies the transition test set first, followed up by the stuck-at test set. Test time deviation as the function of the number of devices that have been tested is depicted in Figure 3.10. The proposed test effectiveness assessment method substantially outperforms the conventional static test flow. Since the transition test set is applied first, when the percentage of stuck-at faults in the defective devices increases, the performance of the conventional static test flow deteriorates as expected. The proposed method delivers over 4X test time reduction in this particular experimental setup, reaching up to a 10X level as the percentage of stuck-at defects increases.

Although the performance of conventional static test flows is swiftly effected by the order of test types, the proposed dynamic learning flow has been developed to overcome all biases that may stem from the initial test pattern order. A close examination of Figure 3.9 and

Figure 3.10 reveals that the dynamic test effectiveness assessment process delivers identical results regardless of the initial ordering of the test types as confirmed by the overlay of the curves in Figure 3.11.

In the next set of experiments, the test vector fallout rate is analyzed upon convergence of the proposed test effectiveness assessment method. A test flow of stuck-at and transition test sets is utilized. 3 distinct ratios of stuck-at and transition faults (50% ST / 50% TX, 70% ST / 30% TX and 85% ST /15% TX) are evaluated as in the previous experiment. Normalized fallout rates for these 3 different experimental setups are depicted in Figure 3.12. As the percentage of stuck-at faults increases in defective devices, it becomes easier to detect stuck-at faults, resulting in a steeper fallout curve. The 85% ST /15% TX combination has the most steep fallout curve as expected, followed up by 70% ST / 30% TX and then 50% ST / 50% TX.

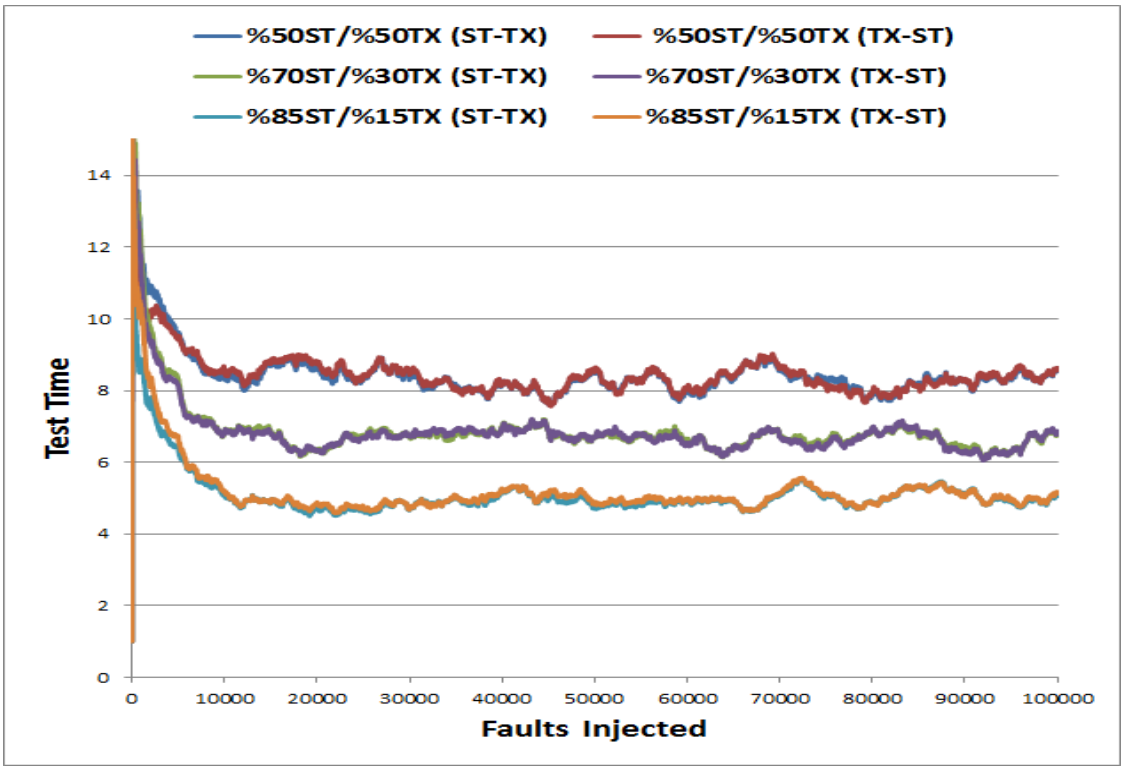


Figure 3.11 Test time for a variety of initial orderings

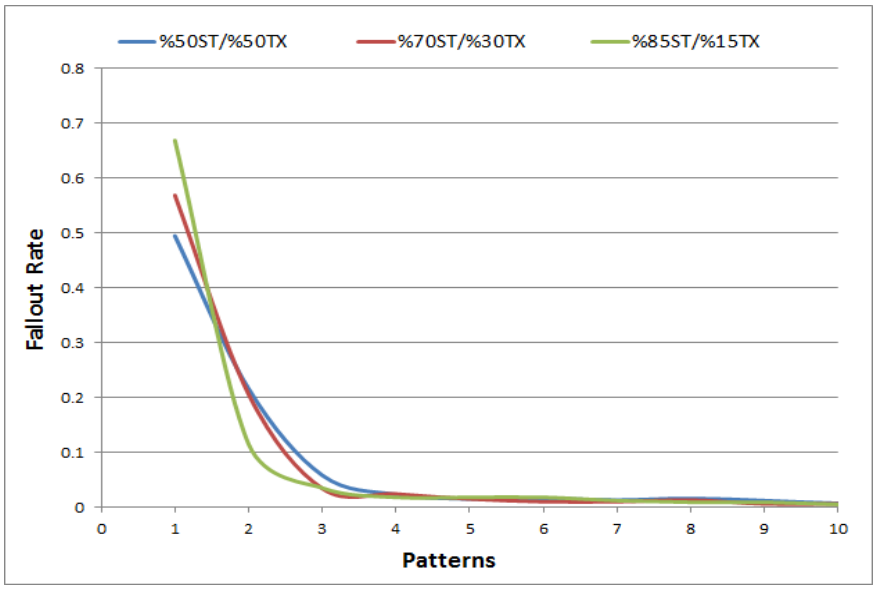


Figure 3.12 Failure rate after convergence

Test cost optimization in multi-level test flows is evaluated by utilizing the optimized test vector orders and the corresponding fallout curves obtained in this experiment as provided in Figure 3.12. This analysis focuses on two levels of test flow, namely, wafer-sort followed by package test. It is assumed that the identical test sets are applied at both levels and the test cost trade-off space is explored as more test patterns are dropped from the wafer test flow while keeping the package test flow intact, ensuring no deterioration in the overall test quality. As the size of the wafer-sort test set shrinks, more defective devices are carried to the package test, incurring additional packaging and package test costs while lowering the test cost of wafer-sort. The normalized overall test cost, obtained by the application of Eq. (3.8) on fallout rates in Figure 3.12, can be seen in Figure 3.13. The packaging cost is assumed to be 10X of the wafer-sort test cost with wafer and packaging yields of 75% and 95% used in this particular experimental setup. Figure 3.13 shows the trend of the overall test cost (wafer-sort/package test costs plus test escape induced packaging cost) as more tests are removed from the tail end of the optimized test order of the wafer-sort test suite. The overall test cost initially improves and quickly deteriorates after reaching the optimal point. Overall test cost

reductions of up to 40% are observed in this setting while still preserving the original test quality. The 85% ST /15% TX fault combination delivers the highest level of test cost reduction as expected due to its steeper fallout curve.

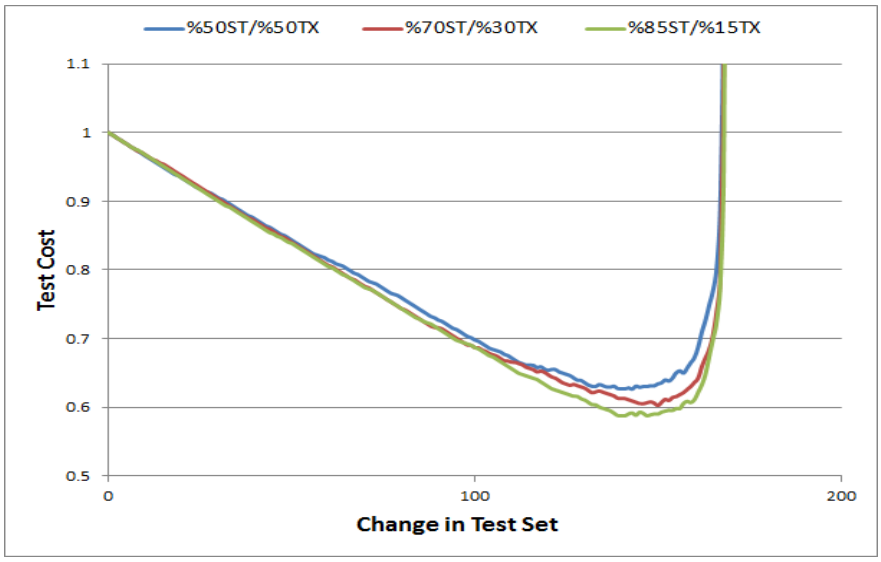


Figure 3.13 Overall test cost as wafer-sort test set shrinks

In the final set of experiments, the effect of different parameters on overall test reduction in multi-level test cost optimization is analyzed. Namely, packaging cost and wafer and packaging yields are altered in the experimental setup discussed in Figure 3.13. Overall test cost reduction as the packaging cost varies for the same setup presented in Figure 3.13 is plotted in Figure 3.14. As the cost of packaging increases, overall cost savings deteriorate as expected due to the increasing cost of defect escapes at wafer-sort. The effect of wafer yield on overall test cost reduction is provided in Figure 3.15. As wafer yield increases, the overall test cost reduction delivered improves due to a lower level of wafer-sort defect escapes for the identical test set. Although the overall test cost is affected by packaging yield, since the package yield is a mere normalization factor in Eq. (3.8), the test cost reduction provided by

the proposed method is not expected to depend on it. Figure 3.16 depicts the test cost reduction level as packaging yields change, showing no dependency on packaging yield as expected.

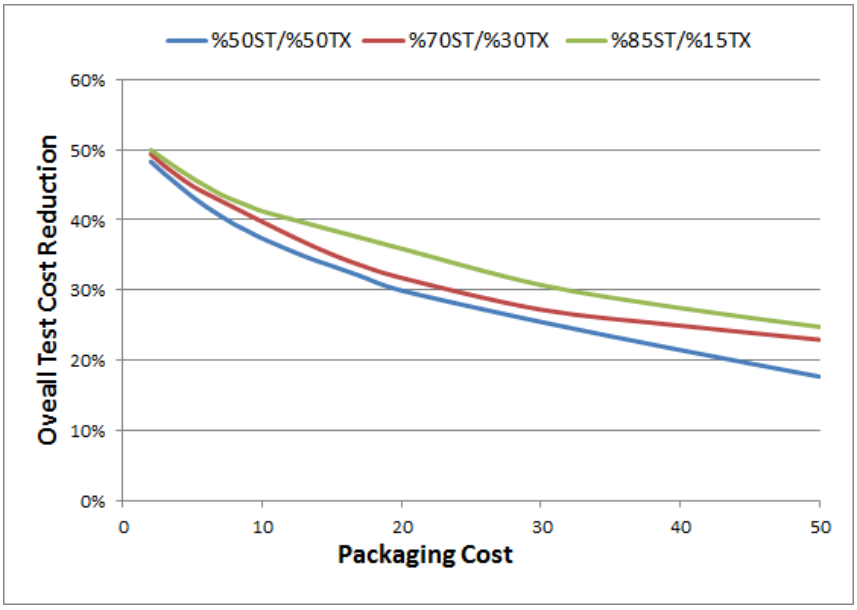


Figure 3.14 Test cost reduction vs. packaging cost

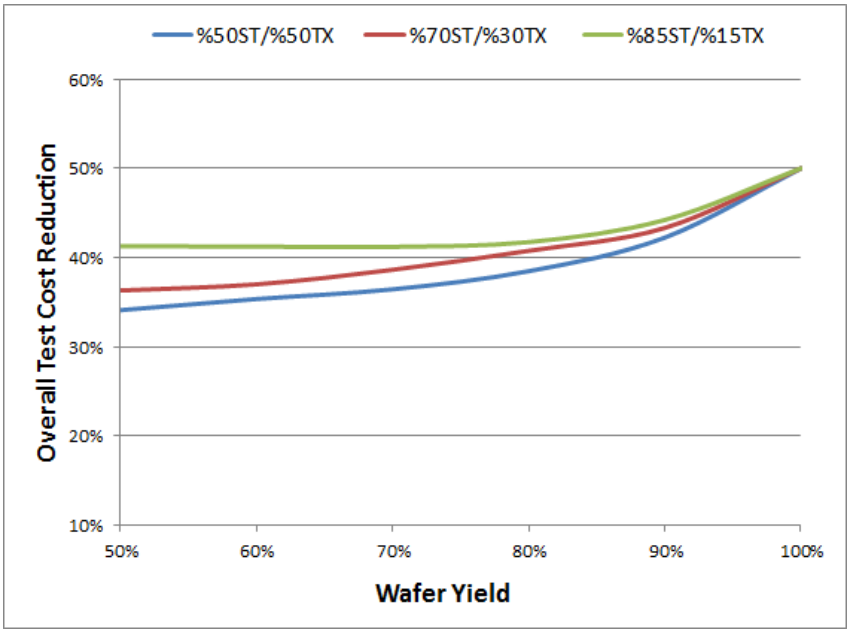


Figure 3.15 Test cost reduction vs. wafer yield

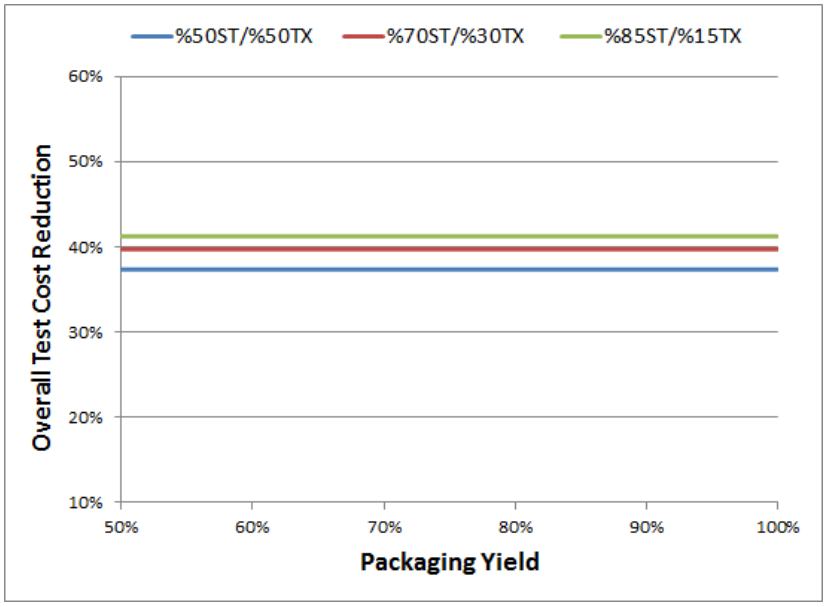


Figure 3.16 Test cost reduction vs. packaging yield

Experimental results show the effectiveness of the proposed method in learning the optimal test order. The test cost reduction levels demonstrated with two fault models on an industry design indicate that substantial test cost savings can be achieved. It can be expected that the proposed framework will deliver increasingly higher levels of cost reduction for production test flows as the number of fault models with tremendous defect coverage overlap among them increases.

3.6 Conclusions

The continuous trend of including new fault models in test flows in the hope of taming defect escape levels increases the inefficiency in test flows, raising the test cost with no commensurate defect coverage benefit. In this chapter, an adaptive method is proposed to assess the effectiveness of test vectors through the prioritization of vectors in a test vector order, enabling efficient and effective test set selection.

The proposed method learns the effectiveness of the individual test vectors by utilizing real time test failure information during production testing and dynamically altering the test flow. Despite the use of only first failing test information for each failing device, the position of the failing test and an ε-greedy process are effectively utilized to prioritize the test vectors. Test vector prioritization not only reduces the average test time to detect a faulty device, but also enables test cost optimization, particularly in multi-level test flows.

Experimental results show the effectiveness of the test vector prioritization process and illustrate the accompanying substantial test cost reduction. The ever increasing test cost problem can be thus significantly ameliorated, reducing test's overall contribution to final product cost thus leading to an improved competitive position in the marketplace.

Chapter 3, in part, is a reprint of the material as it appears in *B. Arslan and A. Orailoglu, "Adaptive Test Optimization through Real-time Learning of Test Effectiveness", Design, Automation and Test in Europe Conference, 2011*; and in *B. Arslan and A. Orailoglu, "Aggressive Test Cost Reductions Spanning Multiple Levels of Test Flow through Continuous Test Effectiveness Assessment", submitted to IEEE Transactions on Very Large Scale Integration Systems*. The dissertation author was the primary investigator and author of these papers.

Chapter 4

Tracing the Best Test Mix through Multi-Variate Test Quality Tracking

The individual test vector effectiveness assessment through the prioritization of test vectors from various fault models as presented in Chapter 3 enables the identification of a highly optimized test set through a fine-grained analysis down to a single vector level. The test effectiveness assessment process at the individual vector level naturally requires an extensive test failure data collection and an intensive learning process. The test vector level optimization is therefore more suitable for mature manufacturing processes, wherein defect characteristics very slowly change but are infrequently punctuated by rapid shifts due to lot changes, consequently allowing reaping the benefits of the laborious test effectiveness learning process for an extended duration.

The manufacturing process is however frequently adjusted at the initial deployment of a new process node and in the product ramp-up period in order to eliminate the systematic process relates issues, aiming to improve the process yield. The frequent and sharp defect characteristics changes in this phase necessitate a test cost and quality optimization technique that can quickly assess test effectiveness and rapidly adjust based on defect characteristics changes. The consequent necessity for rapid test optimization and adaptivity is addressed in this chapter through an adaptive test effectiveness assessment process at fault model level and

a subsequent optimized test suite selection. While a fault model level test effectiveness assessment may introduce a deviation from the test cost-quality optimization attainable with an individual vector level test effectiveness assessment if the defect detection effectiveness based prioritization of test vectors within a fault model substantially diverges from a fault coverage based prioritization, the higher level test effectiveness analysis at the fault model level enables quick learning and fast adaptivity.

The proposed adaptive test optimization flow aims at selecting the optimal cost-effective test mixture from various fault models to continuously deliver the target test quality by dynamically updating the test set during production testing based on defect characteristics. This goal is achieved by establishing a fault-defect coverage mapping and maintaining this mapping in the face of evolving defect characteristics. The bridge between fault and defect coverage is established by representing the failure data from a sample of devices as a function of coverages of the utilized fault models. This process cumulatively looks at the binary pass/fail data from multiple devices and effectively converts the pass/fail information from individual tests of the structural test suite to a parametric test quality data as a multi-variate function of fault coverages. The accuracy of this mapping as defects evolve is ensured by utilizing the data from a small sample of recently tested defective devices that reflects the current defect characteristics. The selection of the sample size of the recently failed devices for test quality estimation leads to a tradeoff between accuracy and adaptivity as a sample size that delivers superior accuracy during a slow drift of defect characteristics may suffer during rapid changes and conversely. A novel method is developed to adaptively adjust the sample size during production test as well so as to improve both accuracy and adaptivity.

An overview of the proposed framework is provided in Section 4.1 and the test quality estimation as a multi-variate function of fault coverages and the optimal test set identification are presented in Section 4.2. The failure data selection for accurate defect characteristics

tracking is presented in Section 4.3. Section 4.4 discusses the experimental results in detail and conclusions are drawn in Section 4.5.

4.1 Approach

The proposed adaptive test framework tracks test quality during production testing and adjusts the active test set as a varying coverage combination of different fault models in order to continuously retain the test quality at the target level while consistently expanding the minimal test cost. A possible scenario during production testing is depicted in Figure 4.1. Initial testing starts with a full suite of tests from various fault models with no heed paid to test cost. As the failure data is collected, the test quality model whose parameters are estimated during run time based on the defect characteristics is constructed and a particular test mixture (such as, for example, 98% stuck at, 85% transition, 80% IDDQ, and so on) that delivers the target test quality at minimal cost is selected at time t_1. As test application proceeds, the estimated test quality that is continuously tracked improves at time t_2 as seen in Figure 4.1. The adaptive test flow exploits this improvement and shrinks the test set (perhaps to 95% stuck at, 80% transition, 79% IDDQ), reducing the test cost, but still delivering the target test quality. As test proceeds and new devices are screened, test quality may deteriorate at time t_3. The adaptive flow detects this quality change and expands the test set based on current defect characteristics (say, to 96% stuck at, 84% transition, 82% IDDQ) to lower the defect escape rate to the target level at minimal test cost.

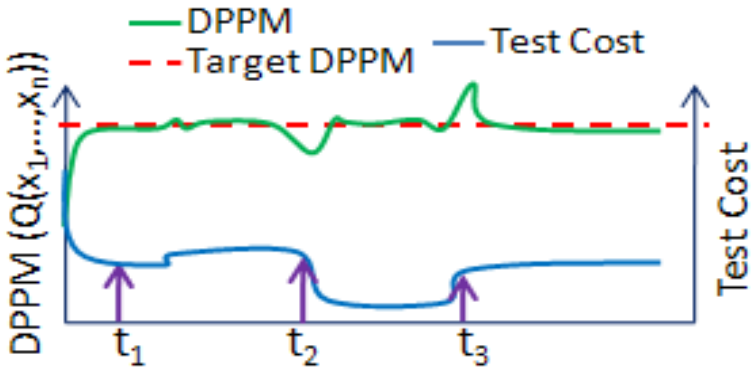


Figure 4.1 Test quality vs. test cost

The estimation of test quality based on the instantaneous defect characteristics constitutes a major challenge in the adaptive flow. In fault-model based test generation, the quality of a test set is measured in terms of the coverage of the target fault model. Fault coverage is quite a useful metric as it can be computed pre-silicon through fault simulation and guides the test generation process. However, due to the absence of a one-to-one match between faults and defects leading to frequent defect coverage overlaps of multiple fault models, fault coverage does not directly engender an accurate test quality estimation for a test set, particularly one generated through the use of multiple fault models. Since it is infeasible to model and simulate all possible defects and the distribution of various defect types depends on the current manufacturing process parameters and environment, establishing a *pre-silicon* correlation between fault coverage and defect coverage is rather challenging, if not downright impossible. The tracking of the evolving defect characteristics necessitates a continuous feedback from the devices under test. We resolve this quandary in our adaptive test framework by resorting to the actual silicon failure data in test quality estimation (DPPM) in order to explore the fault-defect coverage relationship, wherein defect detection data is represented in a multi-dimensional space of fault coverages.

Since defect characteristics may drift slowly throughout production testing, punctuated by frequent sharp perturbations, it is crucial that the failure data used for test quality estimation faithfully represent the current defect characteristics in order to track the instantaneous test quality. Since the current device under test can be expected to have characteristics similar to the recently tested devices on the same wafer/lot (i.e. close proximity and similar process parameters), a small set of recent failures is utilized in our adaptive flow to estimate the test quality. The selection of a particular sample size of the recent failures introduces a tradeoff between accuracy and adaptivity. While a larger sample size offers better estimation accuracy in a stationary environment with slow drifts in defect characteristics, a smaller size delivers fast adaptivity in a rapidly changing environment. In order to satisfy simultaneously both accuracy and adaptivity goals, a two-size approach, with a small sample size for adaptivity and a large one for estimation accuracy, is utilized and a novel adaptive sample size selection technique is proposed to identify and adaptively adjust the sample size based on the convergence of test quality estimation.

Our adaptive test framework, equipped with a failure data collection scheme that can track defect characteristics, estimates the instantaneous test quality as a function of multiple fault coverages as discussed in Section 4.2.1. Since a varying subset of the full test suite is utilized to test the devices based on the evolving defect characteristics, the representation of test quality in terms of fault coverage enables the extrapolation of the estimation beyond current active test set boundaries. Once a multi-variate test quality estimation model is obtained, the appropriate coverages for various fault models can be selected and continuously adjusted so as to deliver a consistent test quality in this dynamic environment independent of the changes in defect characteristics. However, it should be noted that not only the coverages but also the number of patterns to reach a particular fault coverage level and the cost of each

pattern for different tests should be considered during optimal cost test set selection as discussed and modeled in Section 4.2.2.

4.2 Multi-Variate Test Quality Optimization

4.2.1 Test Quality Estimation

Test quality modeling in the form of a DPPM estimate is an extensively studied area with a diverse set of distinct models having been proposed [102], [103], [104], [105], [106], [107], [108]. The accuracy and complexity of the models, taking into account parameters such as manufacturing yield and fault clustering, vary, with Williams-Brown [104] and Seth-Agrawal [102] being among the most widely used models. As yield is inversely correlated with the test escape rate captured as DPPM, it is explicitly utilized as a parameter in most of test quality models although a small number of them [103], [106] also attempt to estimate test quality without explicitly using yield as a parameter. Fortunately, yield is of paramount importance for manufacturing as it affects the bottom line of the companies, thus leading to numerous proposals for its estimation [109], [110], [111] and its rigorous tracking in the manufacturing environment.

The test quality models in the literature predominantly represent the quality estimates as a function of single fault coverage, frequently stuck-at, ignoring the joint defect coverage of the test sets of multiple fault models. Recently, [112] has extended these methods, utilizing silicon data to represent the defect coverage as a function of multiple fault coverages. Our proposal makes no attempt to define a new test quality model in this extensively-studied area, instead building upon the technique promulgated in [112]. Additionally, an accurate estimate of yield is assumed whenever needed through any of the existing techniques. It should be nevertheless noted as well that the adaptive test methodology we propose constitutes a generic

framework, enabling the use of any test quality model that the user prefers as long as said model can produce an accurate estimation model as a function of fault coverages.

Table 4.1 Defective chips detected on ATE vs. fault coverages

Transition (%) \ Stuck-at (%)	30%	65%	89%	93%	98%
30%	356	389	434	437	442
50%	442	452	466	468	471
65%	452	472	481	482	484
75%	466	481	488	489	490
83%	475	486	492	493	494
96%	488	493	497	498	499

The adaptive test flow captures the silicon failure data in a multi-dimensional table as in [112], wherein each axis represents the coverage of a fault model. Each entry in the table denotes the number of defective devices that have been detected by a test set with the corresponding coverages. Table 4.1 shows the sample data for a hypothetical device to illustrate the impact of coverage on the number of defective devices detected on the ATE and the tradeoff between the multiple fault models. The figure further shows a direct relationship between the coverage and test quality, which is observed in a real world system and ATE testing. Since the test set is divided into smaller tests in production test flows with known incremental fault coverages, this data can be efficiently collected during production testing. While the conventional test flow is content with terminating at first failure, a slight test increase needs to be incurred as the first failure for each model needs additionally to be observed. Since parallel testing of multiple chips (i.e. multi-site testing) is becoming an industry standard [2], the extra test cost is minuscule.

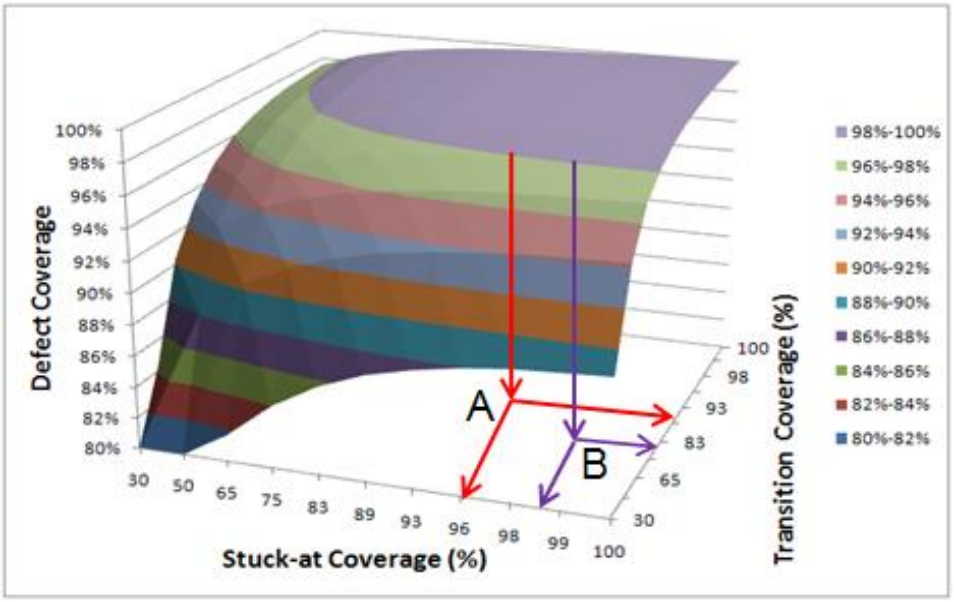


Figure 4.2 Defect coverage as a function of fault coverages

One major benefit of the representation of failure data in this form is that each entry shows the cumulative detection of the corresponding fault models with some chips detected by multiple fault models. Consequently, it implicitly takes into account the defect coverage overlaps of multiple fault models. Once the defect detection data is collected in this form, a multi-variate model can be fitted to represent the defect coverage and subsequently test quality (denoted as $Q(x_1, x_2,.., x_n)$ where x_i denotes the coverage of the *ith* fault model) in terms of multiple fault coverages. Since the failure data may not be available for the full fault coverage range of all available tests as the adaptive flow shrinks the test set whenever possible to reduce test cost, the fitted model extrapolates the test quality estimate beyond the active test set boundaries. It is worth stressing that our adaptive test flow keeps updating the model parameters as new failure data becomes available to track the failure behavior. Figure 4.2 shows the defect coverage curve for the example in Table 4.1. Multiple points with the exact same defect coverage exist in this multi-dimensional space of fault models. For example, A

and B in Figure 4.2 denote two points with the identical defect coverage, giving us an opportunity to select the one with the lowest test cost to reach the same defect coverage.

4.2.2 Optimal Test Set Selection

The modeling of test quality as a multi-variate function of fault coverages offers a quite powerful tool, paving the way for test cost optimization in a multi-dimensional space of various fault models. If the example in Figure 4.2 is revisited, points A and B can be seen to be among the numerous stuck-at/transition test combinations that deliver identical defect coverage. The selection of the minimal cost (i.e. test time) combinations depends not only on the coverages but also the number of patterns to reach these particular coverage levels and the test time of each individual pattern. For example, stuck-at faults can be expected to have a more steep coverage curve than transition faults, requiring more transition patterns than stuck-at patterns to reach the same coverage level. It is well known as well that it gets more costly to improve coverage as it reaches higher levels. The cost of a pattern of a particular test type imposes an additional constraint to be considered; a single IDDQ pattern for example consumes significantly more test time than a single stuck-at/transition pattern due to the settling time on the tester.

The formulation of the problem of selecting the appropriate coverage x_i for fault model i, where $TP_i(x_i)$ is the number of patterns to reach this particular coverage, can be easily cast mathematically as an optimization problem by incorporating TPC_i, the cost of a pattern in fault model i. Both $TP_i(x_i)$ and TPC_i are known pre-silicon. More concretely, $Q(x_1, x_2,.., x_n)$, the test quality level previously discussed, needs to satisfy the target defect escape level as follows:

$$\textit{minimize} \quad \sum_{i=1}^{n} TPC_i \times TP_i(x_i)$$

$$\textit{subject to} \quad Q(x_1, x_2, .., x_n) \leq \textit{Target Defect Escape Level}$$

This nonlinear optimization problem, wherein the objective and the constraint are convex functions, belongs to the class of convex optimization problems [113]. In convex optimization, a local minimum is unique, thus guaranteeing to be the global minimum of the problem, in turn enabling the use of effective algorithms in the search for an optimal solution. The user can select any of a set of well-developed methods such as gradient descent and interior-point methods [113] to solve this convex optimization problem efficiently.

4.3 Adaptive Sample Size Optimization

The proposed adaptive test framework aims at tracking instantaneous test quality based on the current defect behavior during production testing. In order to achieve this goal, test quality is constantly re-evaluated by using the failure information from the defective devices. Therefore, it is crucial that the failure data used for test quality estimation at any point during production testing represent the characteristics of the defect population at that particular point. In order to obtain an accurate estimate of current defect characteristics, a small set of recent failures with similar characteristics is utilized. Once a sample size is selected, as more defective devices continue to be observed during production testing, new failure data replaces the oldest data, thus evoking a sliding window that traces the failure information from the newest samples.

The size of the sliding window (i.e. sample size) is critical to the defect characteristics tracking as it offers a trade-off between estimation accuracy and adaptivity. If the defect characteristics are stable or drifting slowly at a particular moment during production testing, a

larger window size can improve the test quality estimation accuracy due to the increased sample size. However, a large window size will react slowly to possible rapid perturbations in the defect population, failing to adapt quickly. Conversely, a small window can quickly adapt to the rapid changes in the defect characteristics but the estimation accuracy suffers due to the smaller sample size. In order to effectively track the defect characteristics on both ends of this spectrum, we propose a finely tuned method, striving for accuracy and rapid adaptivity simultaneously.

The window size is adaptively selected to obtain the proper size for current defect characteristics in our adaptive test framework. Various window sizes are incrementally evaluated and the correlation among the test quality estimates monitored. When the test quality estimation converges (i.e. the variation is less than a user defined threshold of say 10 DPPM), it is selected as the target window size to be used for test quality estimation. The window size is re-evaluated and slightly adjusted for optimal tracking as notable changes are observed in the test quality estimate model during production testing.

Furthermore, in addition to the window size (w_{est}) that is adaptively selected for test quality estimation, an additional smaller window size (w_{mon}) is concurrently monitored for potential rapid perturbations in the defect characteristics as depicted in Figure 4.3. Since the smaller window will react quicker in case of rapid change, it can be used as a trigger to warn the adaptive system, enabling fast reaction and adaptivity. When a large fluctuation in test quality estimation is observed with the small window size, indicating a possible abrupt change in defect characteristics, the test quality estimation process switches to the small window for fast adaptation; the defect escape rate target is also temporarily reduced to compensate for the possible inaccuracy in the test quality estimation. As the defect characteristics stabilize subsequent to the rapid change, the test quality estimation process switches back to the larger window and re-tunes it for improved accuracy.

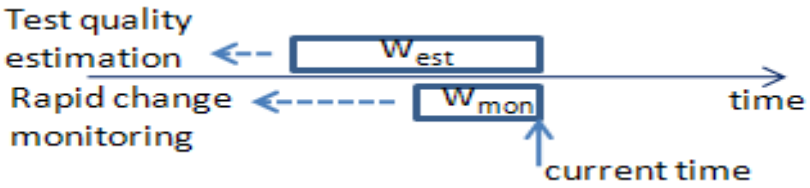


Figure 4.3 Two windows for accuracy and adaptivity

4.4 Experimental Results

The proposed adaptive test method operates during production testing, continuously monitoring and adjusting the test set as a varying combination of the different fault models based on the evolving defect characteristics throughout the product life cycle. Our experimental setup simulates the adaptive flow by injecting randomly selected faults to an industrial design of more than 1 million gates. The behavior of the adaptive flow is therein analyzed as parameters such as yield, fault type and fault clustering are modified, primarily focusing on the flow's ability to track such defect characteristics.

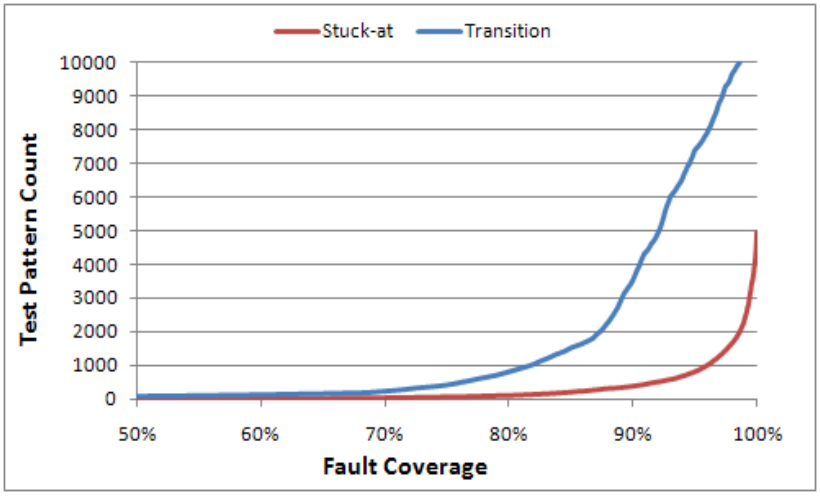


Figure 4.4 Test pattern count vs. fault coverage

The stuck-at and transition fault models and the corresponding test sets generated by a commercial tool are used in our experimental setup. Figure 4.4 shows the exponential growth in test pattern count as coverage approaches 100% for both transition and stuck-at fault models for the industrial circuit used in the experiments. The randomly selected stuck-at and transition faults are injected to the design and tested by these test sets. The transition test set is also fault simulated for the stuck-at fault set in order to consider the defect coverage overlaps of fault models in our experimental setup. Consequently, when a stuck-at fault is injected, it can be detected by both the stuck-at and the transition test sets.

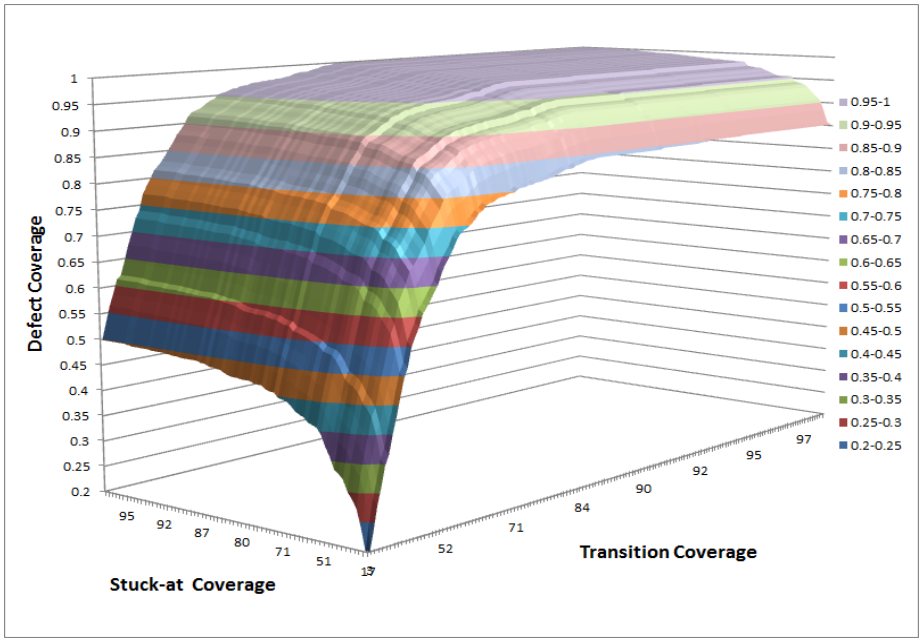


Figure 4.5 Defect coverage vs. stuck-at & transition fault coverages

In order to simulate the defect behavior change, we alter three parameters, namely, the ratio of the injected stuck-at and transition faults, the average number of faults in a defective device and the manufacturing yield. The Williams-Brown test quality estimation model is used in our experiments; of course, other models, including the Seth-Agrawal model, could be used

as well instead. The defect coverage obtained as a function of stuck-at and transition fault coverages based on the failure data during test application is utilized in the model as explained in Section 4.2. An example of defect coverage data at a particular point during our experiment for a mix of 50% stuck-at fault and 50% transition fault occurrence probability with an average of 2 faults per defective device is depicted in Figure 4.5.

100,000 fault combinations based on varying parameters are injected and the target test quality is set to a fixed DPPM in our experiments. As one of the variables of the injected fault set, the average number of faults per device is gradually changed within the range of 1 to 3 faults during test application. The test cost during this process in order to reach the target test quality for two different manufacturing yields, 80% and 95%, and two different stuck-at/transition fault occurrence probabilities, 50%/50% and 85%/15%, is reported in Figure 4.6. When the average number of faults per device increases, the same test quality can be delivered by a smaller test set as fault detection requirements ease. As can be clearly observed in Figure 4.6, the adaptive test flow successfully tracks the change in the number of faults per defective device, reducing the test cost as the number of faults increases. Similarly, an increase in the manufacturing yield improves the test quality for an identical test set. The proposed method successfully reduces the test cost for the higher yield as depicted in Figure 4.6, delivering the same test quality for 95% yield with a lower test cost than the 80% yield. Finally, the ratio of occurrence probabilities of stuck-at and transition faults has a profound effect on test quality. As seen in Figure 4.4, it is less costly to reach a higher coverage for stuck-at faults. Consequently, as the occurrence probability of stuck-at faults increases, the overall test quality can be improved in a cost effective manner by increasing the stuck-at coverage and reducing the transition coverage. Conversely, the increase in transition fault frequency necessitates the more costly increase in transition coverage. The adaptive test flow can also track this behavior as can be observed in Figure 4.6, wherein the 85%/15% stuck-at/transition fault

combination warrants a lower cost test set than the 50%/50% mix to deliver an identical test quality. The tracking of the stuck-at/transition occurrence ratio is further studied in Figure 4.7, wherein the test cost to reach the target test quality is reported as the stuck-at/transition occurrence ratio is altered while keeping other parameters constant. As expected, the test cost improves as the stuck-at occurrence probability is increased.

The experimental results highlight the success of the proposed adaptive test framework in tracking defect characteristics changes. As the stuck-at/transition fault occurrence ratio, the average number of faults per device and the manufacturing yield drift, the adaptive flow successfully uses the defect detection data in test quality estimation and reacts to all changes as expected by delivering the most cost effective test set for the test quality desired.

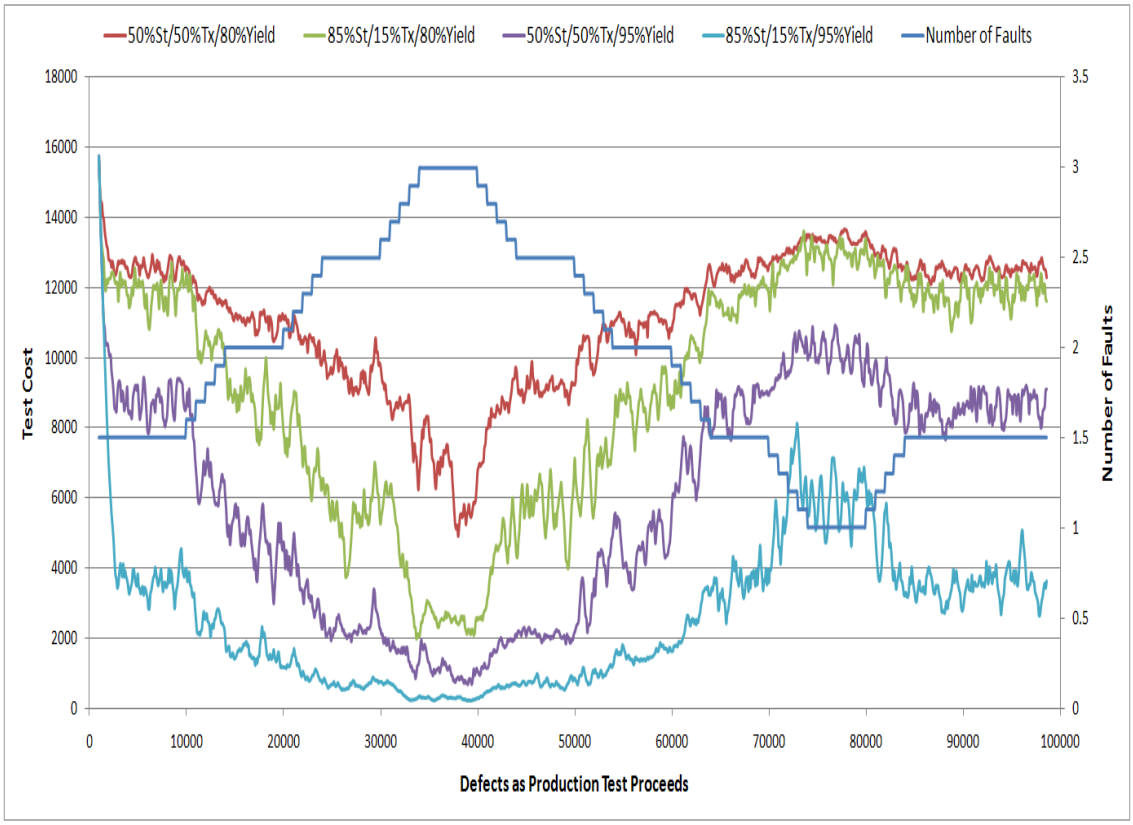


Figure 4.6 Test cost as defect characteristics evolve

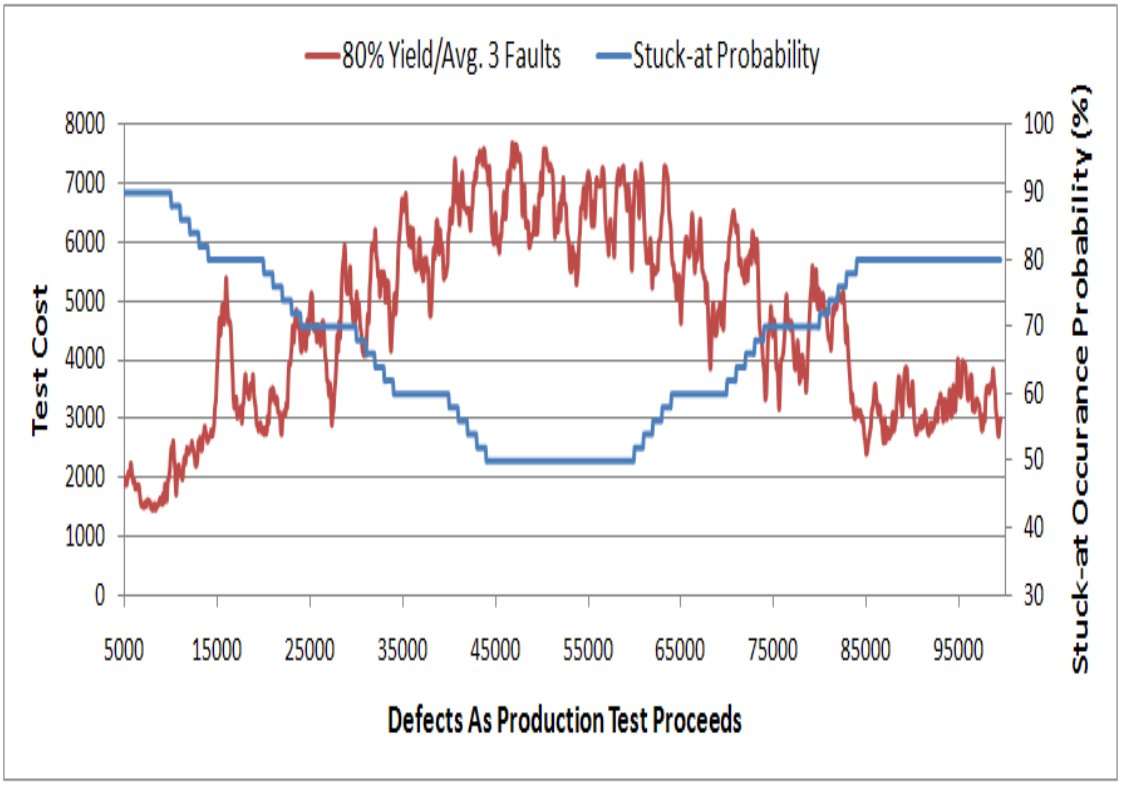


Figure 4.7 Test cost vs. stuck-at/transition fault occurrence ratio

4.5 Conclusions

The goal of delivering a target test quality within an acceptable test cost budget is continuously challenged as process scaling marches on, leading to the emergence of a number of additional fault models to meet the test quality goal while demanding the identification of a cost-effective mixture of these test types. However, the continuous drift in defect characteristics precludes the possibility of a single, unique optimal test set throughout a product life cycle, necessitating the adoption of adaptive test approaches to meet the evolving needs of this dynamic environment.

Chapter 3 presents an adaptive individual test vector effectiveness assessment method and a subsequent identification of cost-effective test set based on the test effectiveness

information. The assessment of test effectiveness at individual vector level however requires an extensive test failure data collection and learning process, resulting in an approach that is more suitable for mature manufacturing processes wherein the defect characteristics infrequently change. This chapter addresses the need for an adaptive test cost optimization method that can accurately track frequent and sharp defect characteristic shifts during the product ramp-up phase due to frequent updates in process parameters in order to increase the manufacturing yield. The proposed method represents test quality as a continuously updated multi-variate function of fault coverages by utilizing the test failure data from a small sample of recently tested chips during production testing, resulting in an optimal test mixture being identified through the selection of appropriate fault coverage for each fault model. The sample size of the chips used for test quality estimation is also adaptively adjusted for an optimal tracking of the evolving defect characteristics, aiming at both accuracy and rapid adaptivity.

Chapter 4, in part, is a reprint of the material as it appears in *B. Arslan and A. Orailoglu, "Adaptive Test Framework for Achieving Target Test Quality at Minimal Cost", Asian Test Symposium, 2011*; and in *B. Arslan and A. Orailoglu, "Tracing the Best Test Mix through Multi-Variate Quality Tracking", VLSI Test Symposium, 2013*. The dissertation author was the primary investigator and author of these papers.

Chapter 5

Chip-Specific Delay Test Resource Allocation

The increasing magnitude of process variations individualizes effectively each chip, boosting the inefficiencies in conventional delay test flows that apply an identical test set to all chips as discussed in Chapter 1. While the distinct chip characteristics impact not only delay tests but also various other tests such as IDDQ, this emphasis on delay test is partly due to its increasing importance in DSM defect detection, subsequently consuming a substantial fraction of the test cost budget, and the lack of a satisfactory answer as to how to effectively test for delay defects. Furthermore, delay tests lend themselves to an analysis based on a differentiation of individual chips since distinct tests can be envisioned as questioning diverse magnitudes of delay margins and consequent vulnerability to different magnitudes of random delay defects.

This chapter attempts to address delay test effectiveness in an era of pronouncedly individualized chip instances by addressing the optimal use of overall test delay resources so as to deliver the best possible test quality by identifying the appropriate and optimal amount of delay test resources for each chip based on its position on the process variation curve. Optimal allocation of test resources throughout the complete process variation space is achieved through a statistical analysis of the design, culminating in a model of delay test quality in

terms of process parameter variations. Subsequently, a mathematical treatment based on the delay test quality distribution follows, leading to the optimal and precise solution for the test resource allocation problem. The mathematical treatment is theoretically shown to be optimal in terms of delivering maximum test quality for a specified constant test cost. Furthermore, practical considerations such as the impact of the possible process measurement errors on the overall effectiveness of the proposed method are discussed and empirically analyzed.

The motivation for the proposed technique is presented in Section 5.1. An overview of the proposed flow, including the details of delay test quality and process variation estimation methods, is provided in Section 5.2. The theoretical framework and a corresponding solution are presented in Section 5.3. Experimental results are provided in Section 5.4 and followed by conclusions in Section 5.5.

5.1 Motivation

Timing behavior varies from chip to chip due to variations in the process parameters. The process-variation induced delay distribution of a chip, modeled as a Gaussian distribution for illustration purposes, is shown in Figure 5.1. The left side of the distribution corresponds to faster chips and the right side to slower ones. The chips with the longest path delays that are smaller than the system clock period meet the frequency target and are shipped to the customer. However, the chips whose longest path length exceeds the system clock period cannot meet customer specification and are discarded. As a chip gets closer to the system frequency, timing margins get smaller.

A delay defect alters the timing of the design. However, the impact of the identical delay defect varies from chip to chip. If a delay defect of size d exists in all chips with the timing distribution shown in Figure 5.1, the delay defect effectively shifts the distribution to

the right. As can be clearly seen in this example, the chips with the larger timing margins (i.e. faster chips) continue to meet the target speed despite this particular defect. However, the chips with the smaller margins cannot tolerate the effect of this particular defect, failing to satisfy the system frequency.

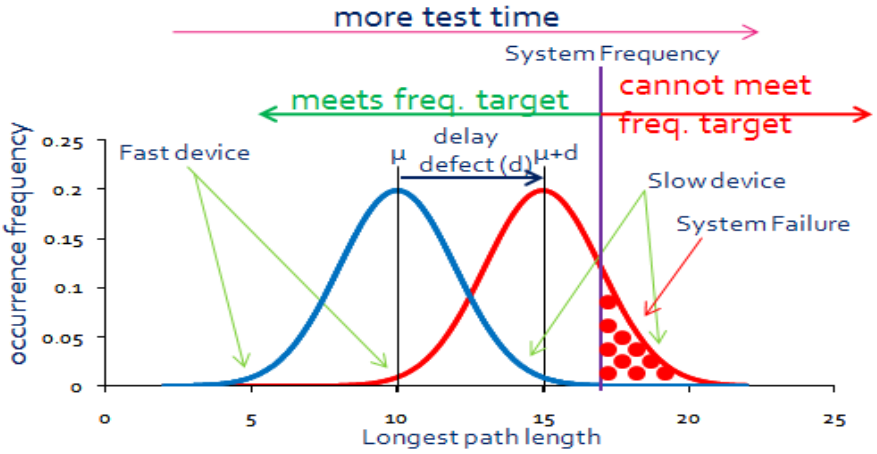


Figure 5.1 Delay defects and process variation effect

Since the effect of a particular delay defect varies based on the process variation, the set of possible random delay defects will manifest itself diversely, with slow chips delivering heightened rates of observed failures whereas fast chips may pass for all but a small proportion of the defects of significant delay size. The identical delay test set paradoxically delivers distinct defect escape rates for each set of chips at a particular point in the process variation curve and consequently distinct delay test quality ensues for different chips, depending on their process parameters. Since the defect escape rate on the slower side of the process variation space is higher than the faster side, test quality can be increased by an improved allocation of test resources (i.e. test time) based on the position of the devices in the process variation distribution. The proposed delay test scheme allocates varying test time to the devices based on process variation, providing more test time to the slower chips to

increase the overall delay test quality, consequently reducing the overall delay defect escape rate within the available delay test time budget.

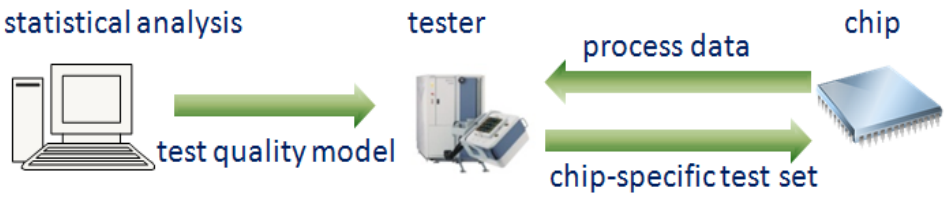


Figure 5.2 Proposed process-aware delay test flow

The proposed process-aware delay test flow as depicted in Figure 5.2 performs statistical analysis of design and test characteristics in the test development phase and provides an analytical model to determine the chip-specific test time allocation. During production testing, on-chip process monitors are utilized to extract the location of the DUT in the process variation space. The tester dynamically determines the optimal test time allocation for the DUT by providing the chip-specific process data obtained from process monitors as an input to the analytical model. On-chip process monitors such as a ring oscillator [114], on-chip path measurement [115] or other common approaches such as leakage measurement are possible candidates for process monitoring.

5.2 Delay Test Time Allocation

The proposed delay test framework aims at allocating more test time to the chips on the slower side of the process variation space in order to improve the overall delay test quality while still keeping the average test time per chip equal to the target test time budget. We utilize the points with equal occurrence probability across the peak of process variation induced delay distribution curve to retain an overall test time identical to that of the traditional

test flow which allocates equal test time to all chips. We illustrate the proposed method for the Gaussian process variation model, a commonly used process variation model. This concretization to the Gaussian model is effected at no loss in generality as the method can be straightforwardly extended to other process variation models as briefly discussed later.

The probability density function of the normal distribution as shown in Figure 5.3 is symmetric around the mean (μ) with symmetric points either side of the mean having the same occurrence probability. The proposed method allocates more test time to the devices on the slow side of the process distribution while keeping the average test time at the symmetric points equal to the test time budget. Since the symmetric points have the same occurrence frequency, the average test time per device remains the same as when applying identical tests to all devices in conventional test flows. The proposed analytical framework determines the amount of test time allocation to maximize the improvement in delay test quality. The parameters of the distribution can be obtained empirically or through a statistical timing analysis as discussed in Section 5.2.2.

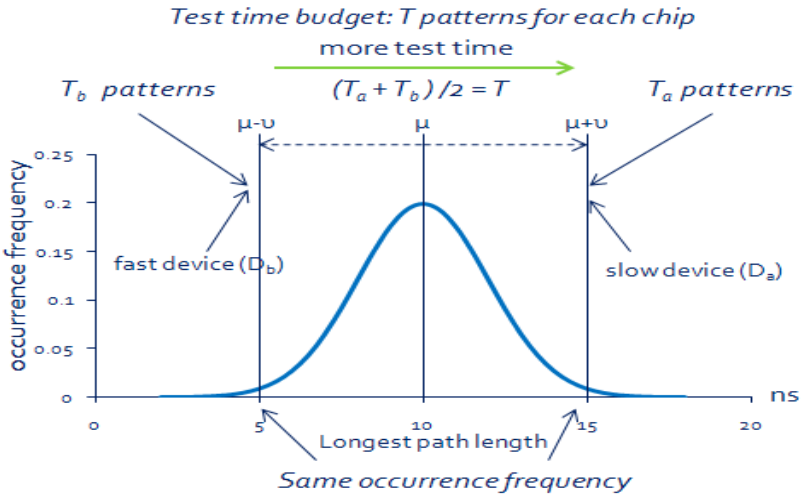


Figure 5.3 Process-aware test time allocation

After reviewing the delay test quality and the process variation induced delay distribution estimation methods in Sections 5.2.1 and 5.2.2, respectively, we proceed to present the analytical calculation of test time allocation in Section 5.3.

5.2.1 Delay Test Quality Estimate

The quality of a test set in delay defect detection is correlated with the length of the test paths. To be able to detect a delay defect, the delay of the test path with the additional delay from the defect should exceed the period of the functional clock. Therefore, the slack of the test path (T_{test}) as illustrated in Figure 5.4 determines the size of the smallest delay defect that can be detected. When the delay defect size exceeds the slack of the test path, it is detected by this particular test; otherwise, it remains undetected. However, if the size of a delay defect is smaller than the slack of the longest functional path (T_{func}) passing through the defect location, this particular defect does not result in a malfunction of the circuit at the system frequency and is therefore deemed redundant. In summary, delay defects that are larger than the slack of the longest functional path passing through the defect location, thus irredundant, but smaller than the slack of the test path, thus undetectable, constitute a set of defects that may cause failure in functional mode.

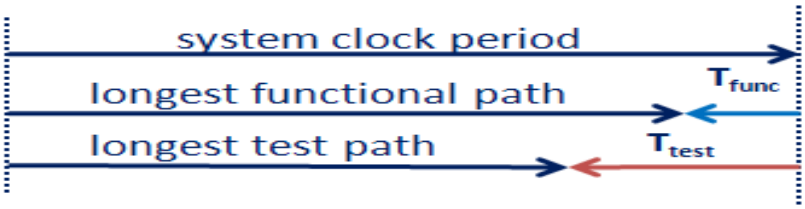


Figure 5.4 Functional and test path slacks

The distribution of the size of delay defects is not uniform. As reported in several previously published works [13], [116], [117], [118], the occurrence frequency of the delay defects of smaller size is higher than the delay defects of larger size and it can be modeled as an exponential distribution as in Eq. (5.1), wherein the parameters, a, b and λ, can be obtained through a least squares fitting of the curve to the empirically collected data [13], [116].

$$F(s) = ae^{-\lambda s} + b \qquad (5.1)$$

The statistical delay quality level (SDQL) is a delay test quality metric that incorporates the functional and test path lengths and delay defect distribution information, wherein a smaller SDQL indicates improved test quality and subsequently lower test escapes [13], [117]. SDQL divides the delay defect distribution to three regions as depicted in Figure 5.5. If the size of the delay defect fails to exceed the slack of the longest functional path (T_{func}), no failure in functional mode ensues, resulting in the redundancy of these particular defects. If the size of the delay defect is larger than the slack of the longest tested path (T_{test}), the defect is detected by the test set as the total delay of the tested path exceeds the period of the functional clock. The delay defects with a size larger than T_{func} but smaller than T_{test} are not detected by the test set and the area between these two boundaries in Figure 5.5 is denoted as $SDQL_{test}$ and is calculated as in Eq. (5.2).

$$SDQL_{test} = \int_{T_{func}}^{T_{test}} F(s)ds \qquad (5.2)$$

If a fault is not targeted by any test pattern, any defect with a size larger than T_{func} is undetected and can cause system failure. The $SDQL_{untst}$ for untested faults can be calculated as in Eq. (5.3).

$$SDQL_{untst} = \int_{T_{func}}^{\infty} F(s)ds \qquad (5.3)$$

The SDQL is calculated for all faults, including both tested and untested faults, and the total of SDQL estimations of all faults denotes the quality level of the test set.

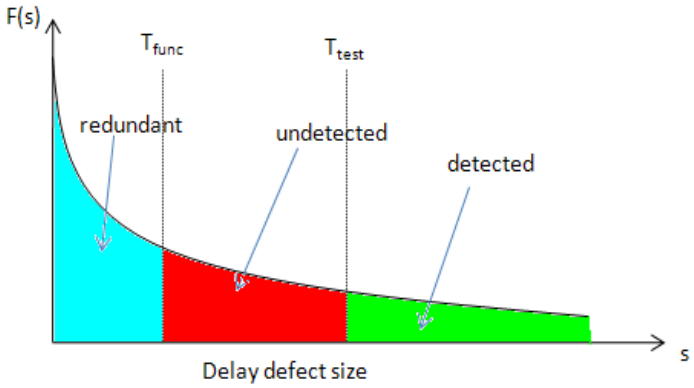


Figure 5.5 Delay defect distribution

5.2.2 Delay Variation Modeling

Statistical timing analysis in the quest to obtain an accurate modeling of process variation induced delay distribution has received enormous attention in the last decade. Pioneering work [119], [120] in the area assumes a normal process variation distribution although techniques that are considering non-Gaussian distributions [121], [122] are emerging as well. We continue the illustration of the method we propose using the Gaussian model assumption as in the previous section although any model could be utilized as long as the delay distribution can be obtained through a statistical timing analysis.

The canonical delay model is widely used in statistical timing analysis to express the delay distributions, wherein all net and gate delays are modeled in the canonical first-order form as in Eq. (5.4) [119].

$$delay = a_0 + \sum_1^n a_i \Delta X_i + a_{n+1} \Delta R_a \qquad (5.4)$$

In the canonical form, a_0 is the mean of the distribution, ΔX_i denotes the globally correlated variation of the process parameters and ΔR_a the uncorrelated variation. ΔX_i and ΔR_a are unit normal distributions (*N(0,1)*) and a_i is the sensitivity of the delay to the corresponding process parameter. The spatially correlated variation within the die also can be modeled in a similar form [120]. Statistical timing analysis uses the sum, difference, max and min operations on normally distributed delay values while traversing the circuit in a manner similar to that of static timing analysis [123]. Addition and subtraction of Gaussian distributions results in a Gaussian distribution while the maximum or minimum of Gaussian distributions does not strictly constitute a Gaussian distribution but can be approximated with a reasonable accuracy as a Gaussian distribution as shown in [119]. Given these functions, the conventional forward and backward propagation on timing graphs can be performed by using the delays in the canonical form and the slack of each node (T_{func}) in the design can be obtained as a Gaussian distribution. In our proposed analytical framework, the average of the slacks of all nodes is used as an approximation to the slack of the longest functional path, T_{func}, and is formulated as in Eq. (5.5).

$$T_{func} = f_0 + \sum_1^n f_i \Delta X_i + f_{n+1} \Delta R_a \qquad (5.5)$$

The mean and standard variation of T_{func} are provided in Eq. (5.6) and Eq. (5.7), respectively.

$$\mu_{T_{func}} = f_0 \qquad (5.6)$$

$$\sigma^2_{T_{func}} = \sum_1^{n+1} f_i^2 \qquad (5.7)$$

Similarly, the slack of the longest test path through a node (T_{test}) in the design can be estimated by performing statistical timing analysis on the paths sensitized by the test pattern.

The average of all the longest test path slacks is employed as an approximation of the test path slack, T_{test}, and represented in the form given in Eq. (5.8). The mean and standard deviation of the test path slack distribution can be calculated by using equations similar to those of the functional path slack distribution.

$$T_{test} = t_0 + \sum_1^n t_i \Delta X_i + t_{n+1} \Delta R_a \qquad (5.8)$$

5.3 Analytical Framework

While we conventionally assume a single fault coverage curve for the full batch of chips, such a curve represents an amalgamation of multiple and distinct delay test quality level (i.e. SDQL) curves depending on the characteristics of the chips. While the shape of the delay test quality curves presumably holds intact across various chip speeds, the inflection points and the points at which sharply diminishing returns are encountered vary depending on the speed. Such individualized test quality curves are particularly important in terms of delay fault manifestation. As discussed previously, the slow chips suffer actual fault manifestation from a whole slew of delay defects, while the fast chips are immune to the manifestation of a significant number of defects. As illustrated in Figure 5.6, an aggressive test regime for slow chips will help capture many more defects, while a somewhat laxer regime of pursuing delay defects for the fast chips will incur nothing but a negligible increase in terms of delay escapes. The identification of the optimal test allocation necessitates the consideration of the correlation of delay test quality with process variation and fault coverage.

We represent the overall delay quality for any symmetric point in the process variation space by utilizing the test quality and process variation models as outlined in Section 5.2. However, since the quality level is different for tested and untested faults, the fault coverage

change as the test size is adjusted needs to be modeled. Once the fault coverage is modeled, the cumulative test quality of tested and untested faults can be represented as a function of fault coverage. The lowest point of the overall delay quality curve, which can be obtained through the first-order derivative of this particular function, corresponds to the optimal test allocation point. The formal and optimal solution is outlined as follows.

Given that A and B are two symmetric points in the process variation space and T_{func} (T_{test}) are a (a') and b (b'), respectively, the corresponding SDQL estimates for the tested and untested faults can be obtained by Eq. (5.9) and Eq. (5.10), respectively. Although T_{func} and T_{test} are defined for each fault, the average values are used as explained in Section 5.2.2.

$$SDQL_{test}(D_a) = \int_a^{a'} F(s)ds \quad and \quad SDQL_{test}(D_b) = \int_b^{b'} F(s)ds \tag{5.9}$$

$$SDQL_{untst}(D_a) = \int_a^{\infty} F(s)ds \quad and \quad SDQL_{untst}(D_b) = \int_b^{\infty} F(s)ds \tag{5.10}$$

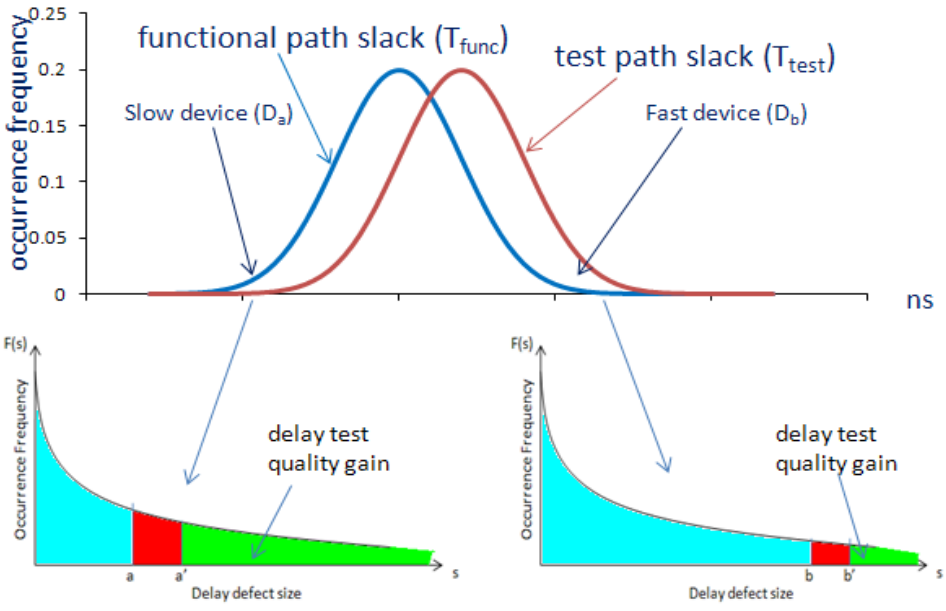


Figure 5.6 Process variation and its effect on test quality

Given that T is the average pattern budget per device, the delay test quality level at the slow process corner A and the fast process corner B can be obtained by Eq. (5.11) and Eq. (5.12), respectively, under the constraint shown in Eq. (5.13).

$$SDQL_{total}(D_a) = C(T_a) \times SDQL_{test}(D_a) + (1 - C(T_a)) \times SDQL_{untst}(D_a) \quad (5.11)$$

$$SDQL_{total}(D_b) = C(T_b) \times SDQL_{test}(D_b) + (1 - C(T_b)) \times SDQL_{untst}(D_b) \quad (5.12)$$

$$\frac{T_a + T_b}{2} = T \quad (5.13)$$

C(t) in Eq. (5.11) and Eq. (5.12) denotes the fault coverage and can be modeled as in Eq. (5.14), wherein *N* is the number of patterns in the test set. The parameter α can be obtained by performing a least squares fitting to the coverage curve of the given delay test set.

$$C(t) = \frac{1 - e^{-\alpha t}}{1 - e^{-\alpha N}} \quad (5.14)$$

Since A and B are symmetric points, the occurrence frequencies are identical. Consequently, the overall delay quality for these particular points is the average of Eq. (5.11) and Eq. (5.12) as provided in Eq. (5.15).

$$SDQL_{avg} = \frac{SDQL_{total}(D_a) + SDQL_{total}(D_b)}{2} \quad (5.15)$$

The T_a and T_b quantities that minimize Eq. (5.15) represent the optimal test pattern allocation that delivers the highest level of delay test quality by keeping the average test time per device identical to the traditional test flow. Since the values of T_a and T_b are constrained by Eq. (5.13), T_b can be substituted by 2T - T_a in Eq. (5.15). Subsequently, the optimal solution can be analytically obtained by finding the value of T_a that sets the derivative of Eq. (5.15) to zero as in Eq. (5.16). Once the optimal value of T_a is obtained, the optimal T_b can be

calculated by Eq. (5.13). It should be noted that since the equation system is solved based on process feedback from the DUT during runtime for the known symmetric points, the only variable in Eq. (5.16) is T_a.

$$\frac{dSDQL_{avg}}{dT_a} = 0 \qquad (5.16)$$

The tester obtains the process data from the on-chip process monitors and provides the process parameters as inputs to Eq. (5.16) in order to identify the chip specific test time allocation in the continuous process variation space.

The proposed test quality optimization technique can be easily extended to any process variation model. First, a delay distribution curve is obtained through statistical timing analysis or empirically for the assumed process variation model. Subsequently, the points with equal occurrence probability on the delay distribution curve that happen to be symmetric points in normal distribution, are paired in the analytical framework to optimally identify the test time allocation.

5.4 Experimental Results

The effectiveness of the proposed delay test quality maximization framework is evaluated in this section and the correlation of the test quality improvement to various parameters is studied.

A system frequency of 500 MHz (t_{sys} = 2ns) and a fault coverage curve, *C(t)*, with α equal to -0.04 are utilized in our analysis. The longest test path length is assumed to be 95% of the longest functional path length and $\mu_{T_{test}}$ and $\sigma_{T_{func}}$ are determined accordingly based on the mean and standard deviation of the functional path distribution. The delay defect size distribution, F(s), provided in [13] is utilized in our analyses. [13] sets the parameters, a, b and

λ, to 1.58 x 10^{-3}, 4.94 x 10^{-6} and -2.1, respectively, in Eq. (5.1) based on the data provided in [118].

In the first set of analyses, the delay test quality change is evaluated as more test patterns are applied as depicted in Figure 5.7. In this analysis, the results displayed are for $\pm 3\sigma$ points with standard deviation equal to 0.16; the reported values are normalized. The mean of the longest path slack distribution is set to 30% of the system clock period ($\mu_{T_{func}} = 0.3 \times t_{sys}$) in our analysis. As can be seen in the figure, a device from the fast side of the process corner quickly converges to a stable delay test quality level with additional vectors only providing minuscule improvements on quality. However, a slow device continues to benefit from additional vectors, improving delay test quality as more vectors are added.

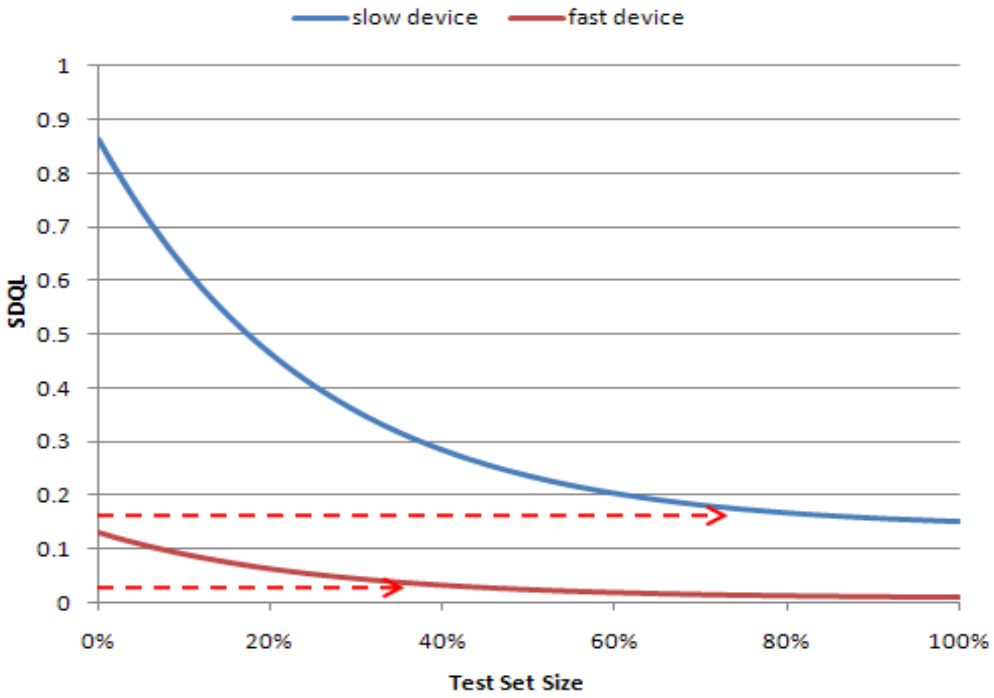


Figure 5.7 Test quality change as test time is increased

In the next set of analyses, the delay test quality improvement attained by the proposed technique is evaluated. A test time budget that would provide 80% transition fault coverage when identical patterns are applied to all devices is selected. Figure 5.8 depicts the SDQL trend for various values of standard deviation and mean of the path slack distribution. Reported SDQL values represent the average of SDQL values at $\pm 3\sigma$ points in the process variation space and they are normalized with respect to a conventional test flow that assigns equal test time to all devices. The proposed method delivers an increasingly higher level of test quality improvement at no additional test cost in comparison to the conventional flow as process variation effects and the associated standard deviation increase. Although the method is less sensitive to the mean of the process distribution, the quality improvement substantially increases as the process variation gets wider.

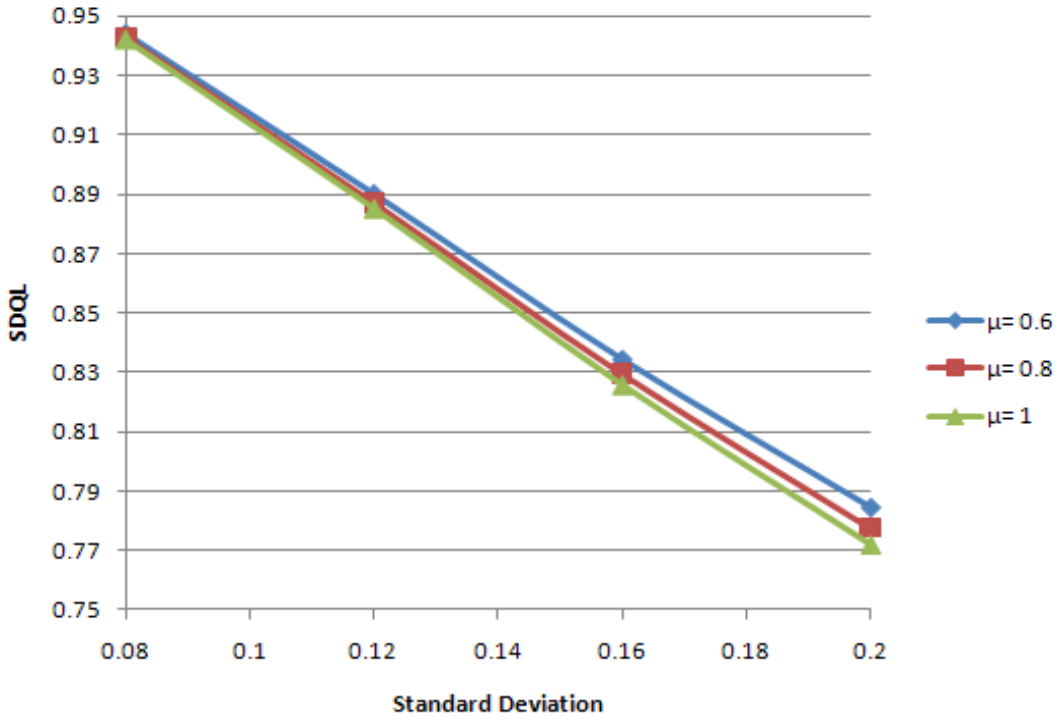


Figure 5.8 Delay test quality improvement

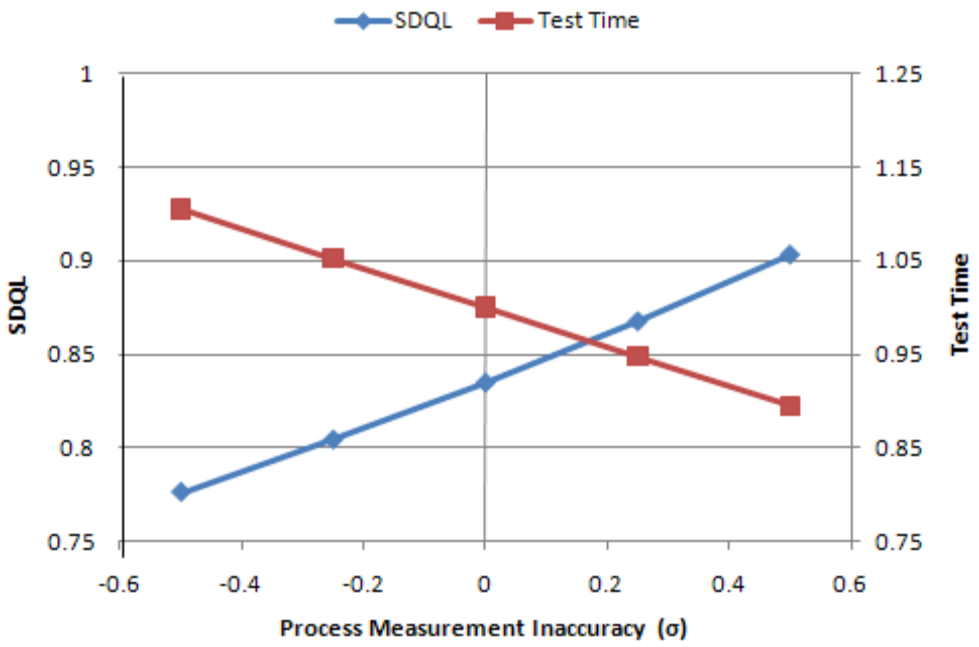


Figure 5.9 The effect of process measurement error

In the final set of analyses, we evaluate the effect of possible process measurement inaccuracies on the overall delay test quality gain delivered by the proposed method. Since the proposed method relies on the data gathered by the process monitors, any errors due to the measurement technique, inaccurate models, on-chip variation, etc. may affect the optimality of the test time allocation. If the process measurement shows that the device under test is slower than its real speed, the method may assign more test time than the allocated test time budget; however, the additional test time will further improve delay test quality. Conversely, if the device is observed to be faster than the real speed, less test time than the target test budget will be used; consequently, the delay test quality improvement over conventional flows will be limited. The SDQL and test time are depicted in Figure 5.9 for process monitor inaccuracies of up to $\pm.5\sigma$. As the process measurement error on the slower side increases, the test cost gets higher but the delay test quality further improves. Conversely, the error on the faster side

attenuates the test quality improvement but the overall test cost is reduced as well. However, even with the consideration of ambiguities introduced as a result of process monitor inaccuracies, the proposed method substantially outperforms the conventional test flow.

5.5 Conclusions

Delay test quality depends on the process parameters, resulting in the identical test set providing distinct test quality for each chip. Consequently, the application of identical tests to all devices fails to deliver efficiencies in the utilization of available test resources.

We propose a process-aware test flow and a corresponding analytical framework to adjust the test time allocation based on the process parameters in order to improve the delay test quality at no additional test cost. The proposed method utilizes statistical information and combines it with the delay test quality estimation to analytically identify the optimal test time allocation without performing any exhaustive search. Additionally, run-time test time allocation decisions are performed in the continuous process variation space, fully utilizing the available process data with no resolution loss.

The analysis provided indicates that a substantial improvement in delay test quality can be attained with no increase in overall test time. Substantial benefits continue to be delivered despite process monitor induced inaccuracies, albeit slightly dampened. The benefits are only slated to increase further as process variation effects continue to increase as a result of continuous process scaling.

Chapter 5, in part, is a reprint of the material as it appears in *B. Arslan and A. Orailoglu, "Delay Test Quality Maximization through Process-aware Selection of Test Set Size", International Conference on Computer Design, 2010*; and in *B. Arslan and A.*

Orailoglu, "Full Exploitation of Process Variation Space for Continuous Delivery of Optimal Delay Test Quality", Asia and South Pacific Design Automation Conference, 2013. The dissertation author was the primary investigator and author of these papers.

Chapter 6

Test Resource Allocation and Scheduling for Multiple Frequency Domains

While the chip-specific delay test resource allocation strategy presented in Chapter 5 addresses the emerging need for an effective delay test resource usage in an era of pronouncedly individualized chip instances as a result of the increasing process variation, the steadily growing diversity within a chip as a result of the coexistence of the hundreds of domains with distinct timing characteristics imposes another major challenge in the effective delay test resource utilization. Since the integration of numerous distinct cores in a typical state-of–the-art SOC results in hundreds of domains with a diverse set of attributes such as frequency and path lengths that directly influence the delay test quality, a carefully crafted allocation of overall delay test resources to these domains based on their distinct characteristics is necessary to extract the highest benefit from the allocated resources. For example, a faster frequency domain with narrow path slacks has reduced tolerance to delay defects than a slower domain with wider path slacks, warranting more generous test resources.

We address the question of the most effective delay test resource allocation across numerous domains in this chapter. Effective test resource allocation necessitates an estimation of test quality based on domain characteristics and algorithmic advances to efficiently identify

optimal allocation of resources in order to maximize test quality. Evidently, the effective test resource allocation process does not necessitate the use of a specific delay test quality estimation method. Any delay test quality estimation method that considers domain characteristics can be equally utilized. Two particular test quality estimation techniques, one a pre-silicon and the other one a post-silicon, are considered in this work to provide two distinct examples. The pre-silicon method utilizes the same metric, *SDQL*, used in Chapter 5 that combines delay defect size distribution information with each domain's frequency, path lengths and the fault coverage vs. test pattern count relationship to provide a simulation-based solution. The post-silicon method utilizes the failure rate of production test to guide test quality optimization. By considering numerous domain characteristics, test quality estimation delivers a metric that correlates test time to the test quality per domain, thus introducing the possibility of posing an optimization problem. To explore the delay test quality tradeoffs, we first formulate the overall optimization problem, which we subsequently simplify to a computationally tractable convex optimization problem.

While the apportionment of test time resources to frequency differentiated domains is an appealing idea that maximizes test quality within cost and opens up the possibility of enabling test quality boosting techniques, nonetheless, its application in the context of conventional test architectures gets challenged by the practical constraints thus imposed. Primary among them stand test architectural techniques enabling the exploitation of test concurrency that necessitate the identification of the schedule of which domains are tested in parallel, an approach capable of significantly reducing elapsed test time yet imposing synchronization constraints among domains simultaneously tested. Concurrency not only maximizes test time exploitation but imposes in turn a further constraint on simultaneous power utilization, which may necessitate judicious test/power tradeoffs.

The considerable mathematical and algorithmic challenges inherent in resolving the best allocation of test resources to various frequency domains are thus exacerbated by concurrent domain testing considerations under these additional synchronization and power constraints. While overall optimality can be delivered in the case of the fundamental problem definition, the scope of optimization shifts to the minimization of the deviation from the optimal solution in the latter cases. The minimization of deviation ensures negligible error introduction into the more challenging formulations, enabling an efficient exploration of the multidimensional tradeoff space of optimal test resource allocations while determining the schedule of the concurrently tested domains in the face of concurrency induced synchronization and power constraints.

Although a slew of test scheduling techniques [124], [125], [126] that focus on an effective use of test bandwidth have been previously proposed, scheduling in the context of this work fundamentally differs, aiming at the identification of which domains are to be concurrently tested. Furthermore, while the conventional test scheduling focuses on the effective use of a test bandwidth for a *predetermined* test set, the scheduling of concurrently tested domains in this work tackles a unique and challenging goal of the *simultaneous* consideration of the test time allocation (i.e. variable test set) and the domain scheduling to maximize the test quality.

Section 6.1 provides an overview of the proposed methods. Section 6.2 reviews the delay test quality estimation and the optimal test time allocation framework for sequential domain testing is provided in Section 6.3. The simultaneous test time allocation and domain scheduling with architectural and power constraints under support for concurrent domain testing is subsequently presented in Section 6.4. Section 6.5 discusses the experimental results and conclusions are drawn in Section 6.6.

6.1 Overview

As the level of integration and, consequently, design size aggressively grows, core-based test architectures are increasingly becoming the norm for testing large SOCs. In the core-based SOC test, instead of having a single test mode to test the full chip, the design is partitioned to cores and the testing of each core in isolation through core wrappers is enabled. Although numerous methods are being proposed to develop configurable core wrappers with different test channel bandwidth requirements and to achieve an efficient sharing of common test access mechanisms [124], [125], [126], a typical core-based SOC test architecture in an industry setting supports a dedicated test mode for each core with the cores being tested sequentially as depicted in Figure 6.1. In this architecture, each core utilizes the full test channel bandwidth to receive the test stimulus and output the test responses. Each core is composed of multiple frequency domains depicted as $F_{i,j}$ in the figure. An at-speed clock generation circuitry resides on chip which can be configured to generate launch and capture clocks for delay testing at various speeds based on the target frequency of the domain-under-test.

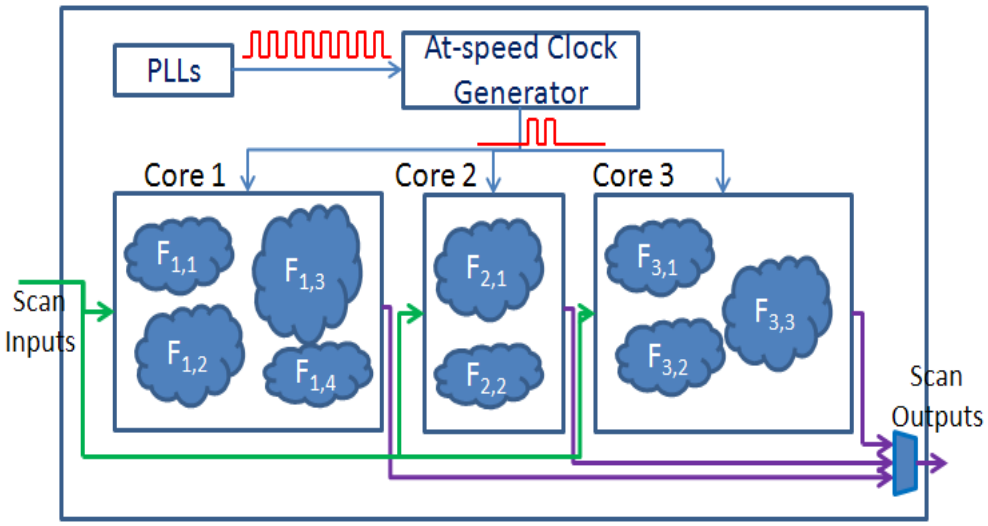


Figure 6.1 Core-based SOC test architecture

The test mode of a core configures its scan chains and routes input/output test channels to this particular core. Since each frequency domain under a core is tested in this particular core scan configuration, the maximum scan chain length and scan shift frequency, and consequently test time, for a single pattern is identical for each frequency domain of a core. However, the maximum scan chain lengths and scan shift frequency can differ across the cores.

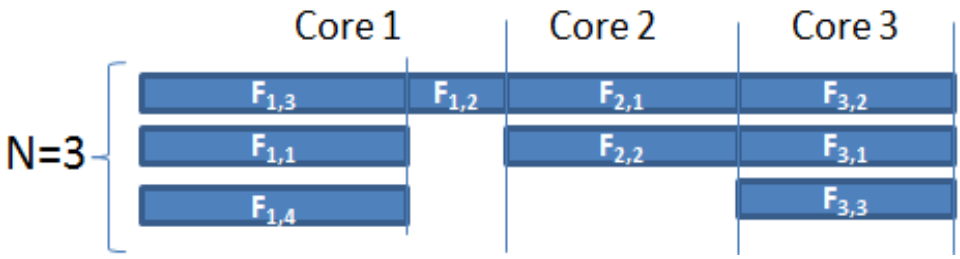


Figure 6.2 Concurrent delay testing of domains

The capabilities of the on-chip clock generation circuitry determine the level of concurrency achievable while performing parallel delay testing of various frequency domains. If at-speed launch and capture can be generated for one frequency domain at a time as in a typical system, the delay test of frequency domains is performed sequentially. If a more complex at-speed circuit enables concurrent generation and routing of *N* frequencies, up to *N* frequency domains can be tested in parallel. An architectural concurrency limit of *N* is thus imposed with the goal being to complete the testing as quickly as possible by careful scheduling of the domains to be concurrently tested.

Since each core is tested sequentially through its dedicated test mode in this particular test architecture, up to *N* frequency domains only under the currently selected core can be concurrently tested. Furthermore, the test start times of all frequency domains that are being concurrently tested are synchronized. In the test architecture depicted in Figure 6.1, 4 frequency domains reside in *Core 1*. If the architecture concurrency limit, *N*, is assumed to be

3, only 3 out of these 4 domains in *Core 1* can be concurrently tested as illustrated in an example test schedule in Figure 6.2, wherein the length of the bars represents the test time for the corresponding frequency domain. Since the architecture concurrency limit of 3 is reached, $F_{1,2}$ is subsequently tested separately from the other domains in *Core 1*. Since the number of domains in *Core 2* and *Core 3* does not exceed the architectural concurrency limit of 3 in this example, all domains in these particular cores can be concurrently tested as depicted in Figure 6.2.

In this chapter, we assume the use of the core-based SOC test architecture described above when developing our delay test quality optimization techniques. We expect that ideas and techniques proposed in the chapter can be extended to other core-based test architectures, albeit with appropriate modifications based on the particularities of the architecture.

Although the SOCs include the test features necessary to enable manufacturing testing, the efficient delay testing of hundreds of frequency domains within an acceptable test time as increasingly higher levels of integration are reached remains a challenge. Although test concurrency provides an effective method to slow down the growth of test time of SOCs, the attainable concurrency level is constrained by not only architectural limits, but also by the power constraints due to the increasing switching activity as a result of higher levels of concurrent testing. As illustrated in Figure 6.3, wherein the width and length of the bars represent the switching activity level and test time for each corresponding frequency domain, the domains that can be tested concurrently are not only determined by the test architecture concurrency limit, denoted as N, but also by the power limit as a result of the cumulative switching activity level, denoted as P. Although the architecture supports the testing of 3 cores in this example, since the cumulative power of all domains in *Core 3* exceeds the power limit, $F_{3,3}$ is tested separately than the concurrently tested domains $F_{3,2}$ and $F_{3,3}$.

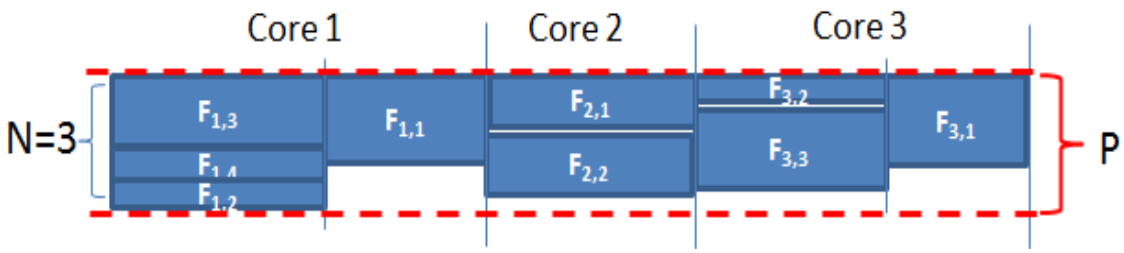


Figure 6.3 Concurrency limits by power constraints

The large set of domains in SOCs with diverse frequencies has numerous other characteristics that affect delay test quality and delay test time in addition to architectural and power related concurrency constraints. Functional and test slack distributions of each domain affect the set of delay defects that can cause a system failure and that can escape delay test, respectively. The maximum scan chain length and the scan shift speed of each core determine the test time of a single pattern for a frequency domain in this particular core. Additionally, the size and fault coverage curve as a function of the number of test patterns for each domain determines the number of faults tested and the test quality improvement trend as the number of patterns increases. Since these parameters vary for each frequency domain based on the design and the test set, we propose a methodology that considers all these parameters and analyzes the tradeoffs across the frequency domains with the goal of obtaining the lowest overall delay defect escape rate for an SOC for a given test time budget. This analysis determines the optimal allocation of the test time budget to each domain and simultaneously pinpoints the optimal scheduling of the domains to be concurrently tested under the constraints of power and test architecture imposed concurrency limits.

Although the proposed method is presented in the context of quality maximization within a test budget, the method can be utilized to efficiently boost the delay test quality further by enabling an effective use of the improved delay test generation techniques such as timing-aware or even simple *N*-detect test generation as well. Timing-aware and *N*-detect

delay test methods, for example, can generate test sets with lower defect escape rates yet the cost of the substantial growth in test set size for all domains can be prohibitive with even generously relaxed test cost constraints. The proposed methodology can be utilized to judiciously allocate test resources for timing-aware or *N*-detect testing of the critical domains, delivering test quality levels unattainable with regular transition test while avoiding cost prohibitive timing-aware/*N*-detect testing of all domains.

We start in Section 6.2 with an overview of delay test quality estimation. The proposed test quality optimization and domain scheduling method is subsequently presented in Sections 6.3 and 6.4.

6.2 Delay Test Quality Estimation

The proposed test resource allocation method necessitates the use of the delay test quality estimation techniques that capture the domain attributes contributing to the delay test quality. We discuss, in this section, two alternative delay test quality estimation techniques that can be applied at the pre-silicon and post-silicon phases as examples. Furthermore, a hybrid approach, wherein the pre-silicon method is followed by the post-silicon method to fine-tune the estimation, is briefly discussed. Of course, any metric that captures the attributes that effect delay test quality could be used equally as well.

The pre-silicon method utilizes the same metric, *Statistical Delay Quality Level* (SDQL), discussed in Section 5.2.1, that combines the defect distribution data with functional and test slacks that are obtained by using a timing simulation. SDQL is a delay test quality metric that incorporates the defect distribution data with functional and test slacks that are obtained by using a timing simulation, wherein a smaller SDQL signals a better test quality and subsequently lower test escapes.

Previously published methods and tools [13], [14], [15] exist for SDQL calculation. The normalized value of SDQL that has been calculated for a frequency domain, $F_{i,j}$, of an industrial design by a commercial ATPG tool is depicted in Figure 6.4. As more patterns are added, SDQL improves as expected. Once the incremental SDQL is calculated as a function of the additional test patterns, a curve can be fitted to represent SDQL as a function of the number of patterns for this domain, $F_{i,j}$, denoted as $Q_{i,j}(x_{i,j})$ in the figure. Our goal is the minimization of the cumulative SDQL across all frequency domains within a set test time budget.

Since the redundant defect set depends on the functional slack of the target fault location as discussed in Section 5.2.1, hence fixed for a given design, the undetected defect level (i.e. SDQL) shrinks as the detected defect level grows; consequently, the goal of SDQL minimization can be achieved by maximizing the detection level as well. The detection level and the curve fitted to it, denoted as $D_{i,j}(x_{i,j})$, can be seen in Figure 6.4 for the same domain and are inversely correlated with SDQL as expected.

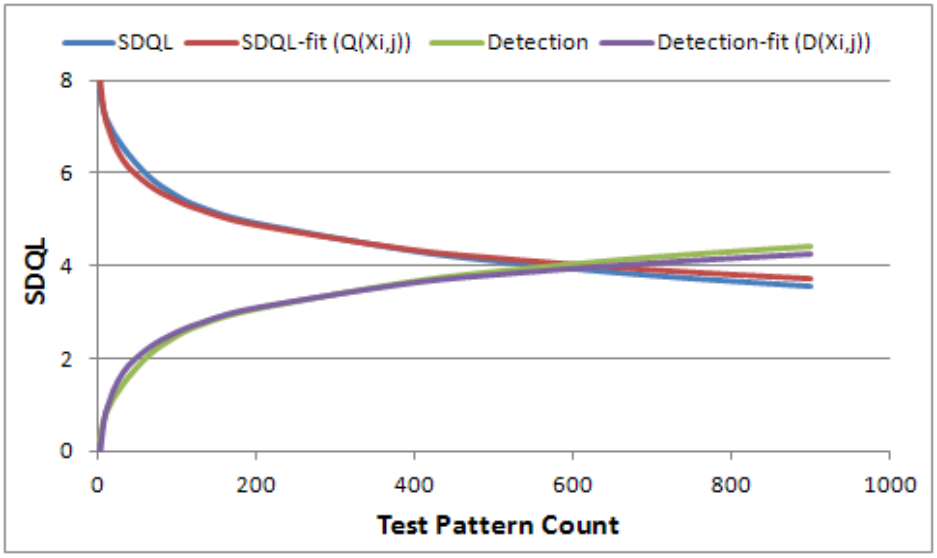


Figure 6.4 SDQL vs. test pattern count

The defect detection level corresponds to the failure rate of the delay tests on actual chips during production testing, hence an observable quantity. Consequently, delay test failure rate data can be collected during production testing per frequency domain and can be represented as a function of the number of test patterns applied, leading into a post-silicon test quality metric. Once the detection level function per domain is obtained, our test time allocation method can utilize it to maximize the cumulative detection across all domains without exceeding the test time budget.

Silicon failure data collection eliminates the need for pre-silicon SDQL estimation methods and associated tools, furthermore avoiding the potential inaccuracies in the pre-silicon estimation due to, for example, timing and delay defect distribution information. However, the test time allocation decision has to be postponed until post-silicon data collection.

Since it is desirable to perform test time allocation per domain before silicon arrival, a hybrid flow can deliver the best of both approaches. First, pre-silicon SDQL estimation can be utilized for initial test time allocation per domain. As sufficient failure information is collected during production testing after silicon arrival, the initial allocation can be fine-tuned based on actual data.

6.3 Optimal Test Time Allocation

The test quality estimated as a function of test patterns per domain in Section 6.2 implicitly captures numerous distinct characteristics of the domain such as frequency, domain size, function and test path lengths, and the fault coverage improvement as more patterns are added. It also allows us to explore test quality trade-offs among the frequency domains. For example, if it is assumed that all characteristics but the frequency of two domains are

identical, the faster domain will have tighter functional and test slacks, resulting in inferior test quality (i.e. higher SDQL) for the same number of test patterns. It is consequently beneficial to allocate more test time to the faster domain. Other domain characteristics offer tradeoffs similar to the frequency. While consideration of these characteristics helps deliver the most beneficial test allocation, one should not overlook the variable cost side of the equation, namely, the test time of each pattern, which has to be taken into account while exploring the tradeoffs throughout the numerous domains.

An SOC test architecture, depicted in Figure 6.1, that only generates at-speed launch and capture pulses at a single frequency forces the domains to be tested sequentially. In this particular architecture, the sole challenge consists of the identification of optimal test time allocation per domain as the schedule of the domains does not affect the overall test delay quality. If the test flow stops at first failure, the user may, after test time allocation, order the testing of frequency domains based on their detection level to minimize the average test time for the defective devices. We proceed in this section with the discussion of the optimal test allocation formulation for SOCs that restrict themselves to sequential testing of frequency domains only. Test quality optimization when an SOC admits concurrent domain testing necessitates the simultaneous optimization of test time allocation and test scheduling as discussed in the next section.

Once the test quality function, $Q_{i,j}(x_{i,j})$, per each frequency domain, $F_{i,j}$, is obtained, the optimal test pattern count, $x_{i,j}$, per domain to maximize test quality within a test time budget should be identified. The test time budget can be set to the total test time utilized by conventional fault coverage based delay test set selection so as to aim at maximizing test quality while using a total test time identical to conventional delay tests. Alternatively, the test time budget can be determined by product and test cost. The test time allocation can be easily cast as an optimization problem by incorporating the maximum scan chain length, $SC_{i,j}$, and

the period of scan clock, $SP_{i,j}$, for each domain. $SC_{i,j}$ and $SP_{i,j}$ are known values for each frequency domain and expected to be identical for all frequency domains of a particular core. Given c cores, f_i frequency domains in core i and test time budget of TT_{target}, the optimal test time allocation per domain can be determined as follows:

$$\textit{minimize} \quad \sum_{i=1}^{c}\sum_{j=1}^{f_i} Q_{i,j}\left(x_{i,j}\right)$$

$$\textit{subject to} \quad \sum_{i=1}^{c}\sum_{j=1}^{f_i} x_{i,j} \times SC_{i,j} \times SP_{i,j} \;\leq\; TT_{target}$$

This formulation corresponds to an integer nonlinear optimization problem as the number of patterns per domain, $x_{i,j}$, is an integer variable. However, since a test suite for an SOC consists of thousands of scan patterns, addition or removal of an individual scan pattern for a domain has a minuscule effect on test time and test quality. We therefore ease the integer restriction on $x_{i,j}$ to simplify the problem, rounding up to the closest integer after the identification of the non-integer solution. Additionally, test setup sequence overhead is ignored in the formulation as it proves to be negligible in comparison to the scan shift time.

This particular nonlinear optimization problem, wherein the objective is a convex function and the constraint is an affine function, belongs to the class of convex optimization problems [113]. In convex optimization, a local minimum is guaranteed to be the global minimum of the problem, enabling the use of effective algorithms in the search for an optimal solution. The user can select any of a set of well-developed methods such as gradient descent and interior-point methods [113] to solve this convex optimization problem efficiently.

If the detection level function, $D_{i,j}(x_{i,j})$, is available instead of SDQL, $Q_{i,j}(x_{i,j})$, the same formulation can be used by replacing $Q_{i,j}(x_{i,j})$ with $D_{i,j}(x_{i,j})$ and changing the objective to the

maximization of the detection level instead of minimizing SDQL. The detection level function is a concave function. The problem of a concave function maximization can be easily re-formulated as the equivalent problem of convex function minimization, enabling the use of the same efficient convex optimization techniques.

6.4 Test Time Allocation and Scheduling

An at-speed launch and capture clock generation circuitry that enables the concurrent testing of up to N frequency domains provides an opportunity for significant test quality improvement/cost reduction. An effective use of concurrency support necessitates a careful identification of the schedule of which domains are concurrently tested while performing test resource allocation; yet, the simultaneous power utilization of the concurrently tested domains should adhere to the imposed power limit, P, to ensure reliable test application.

The proposed methodology therefore strives to identify both the optimal allocation of test time to each domain and the test scheduling while adhering to the test time limit, TT_{target}, on the horizontal dimension and the architectural concurrency, N, and power limits, P, on the vertical dimension as illustrated in Figure 6.3. We assume throughout this work that test power dissipation (i.e. switching activity) of any single domain is below the power limit. If the test power of any single domain exceeds the overall limit, the test set of this particular domain can be re-generated by power-aware test generation techniques [127] to fit it within the power envelope. We start with addressing the test time allocation and scheduling problem with only a test time budget and architectural concurrency limit as illustrated in Figure 6.2. Subsequently, the power constraint is introduced and the complete solution is provided.

6.4.1 Test Time Allocation and Scheduling with Test Time and Concurrency Constraints

The test time allocation and scheduling problem with an overall test time budget of TT_{target} and a test architecture supporting concurrent testing of N frequency domains in a core is significantly more complicated than the test time allocation problem for sequential testing, presented in Section 6.3. We initially provide an optimization formulation for this problem and follow this with an efficient solution.

The test time allocation and scheduling problem has to consider not only the distribution of the test time budget, TT_{target}, but also determine when and where to test each frequency domain, effectively forming the test schedule. In order to create the optimization formulation for this problem, the available test resources are partitioned to horizontal and vertical channels as modeled in Figure 6.5. On the horizontal dimension, since the test architecture supports the concurrent testing of N domains, it is modeled as N distinct horizontal channels that each frequency domain can be assigned to. On the vertical dimension, for each *core i*, V_i vertical channels that are equal to the number of domains in *core i*, denoted as f_i, are available as can be seen in Figure 6.5. Each slot in the horizontal/vertical channel crosspoint can accommodate only a single domain.

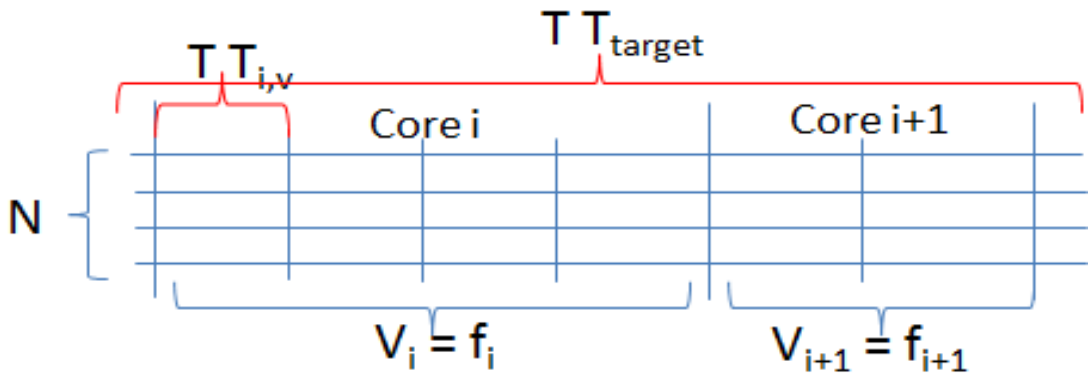


Figure 6.5 Horizontal and vertical test resources

The objective is to determine a test time allocation for each domain and assign these domains to the available slots in order to maximize test quality while only using a total test time budget TT_{target}. As an observant reader would notice, not all vertical channels need to be used. The optimization process identifies the minimum number of vertical channels needed while the rest are not allocated any test time at all. Let $x_{i,j}$, $SC_{i,j}$ and $SP_{i,j}$ denote the optimal test pattern count allocation, the maximum scan chain length and the period of scan clock for domain *j* in core *i*, respectively. In order to keep track of the location assigned to each domain, $x_{i,j,h,v}$ represents whether domain *j* in core *i* is assigned to the slot in the h'*th* horizontal and v'*th* vertical channel crosspoint. TT_{target} is the available test time budget and $TT_{i,v}$ is the test time allocated to the v'*th* vertical channel in core *i* from the available test time budget. Assuming that c cores exist, the optimal test time allocation and scheduling problem can be posed as the following optimization formulation:

$$\textit{minimize} \quad \sum_{i=1}^{c}\sum_{j=1}^{f_i} Q_{i,j}\left(x_{i,j}\right)$$

subject to

$$\sum_{h=1}^{N}\sum_{v=1}^{f_i} y_{i,j,h,v} = 1, \ \forall i \in \{1,\ldots,c\} \quad \forall j \in \{1,\ldots, f_i\}$$

$$\sum_{j=1}^{f_i} x_{i,j} \times y_{i,j,h,v} \times SC_{i,j} \times SP_{i,j} \le TT_{i,v}, \ \forall i \in \{1,\ldots,c\} \qquad \forall h \in \{1,...,N\} \forall v \in \{1,..., f_i\}$$

$$\sum_{i=1}^{c}\sum_{v=1}^{f_i} TT_{i,v} \le TT_{\text{target}}$$

$$y_{i,j,h,v} \in \{0,1\}$$

In the formulation shown above, the first constraint ensures that each domain is assigned to only a single slot. The second constraint determines the test time assigned to each

vertical channel and the third constraint keeps overall test time usage within the test time budget while pursuing the objective of defect escape (i.e. SDQL) minimization. The problem could be easily formulated as detection level, $D_{i,j}(x_{i,j})$, maximization as discussed in Section 6.3. Even if the number of patterns, $x_{i,j}$, are allowed to be non-integer, this is still a very complex mixed-integer nonlinear optimization problem, that while solvable requires extensive computational cost. A tractable solution of this problem necessitates a decomposition that exhibits the underlying properties of the space to deliver an efficient solution while minimizing any possible deviation from optimality. We propose therefore in the subsequent parts of this subsection a decomposition of this formulation into an efficient, incremental delay test quality optimization method to identify the test time allocation and scheduling solution.

While the formulation of Section 6.3 provides an optimal solution to the test effectiveness problem of various domains by equalizing the marginal utility of the final test vector in each test set, the addition of the concurrency constraints deviates the test time allocation to ensure adherence to the synchronization constraints that mark the start of each domain group, comprised of the cores to be tested concurrently. It should be evident that the minimization of the deviation of optimal test times in a domain group is a worthwhile goal. Yet not only does the minimization of deviation should be aimed at, but furthermore the inefficiencies of the fit within the synchronization constraints of the domain groups bloat the test time away from the user-specified test time parameter. Shrinking the time back to the specified test time constraint requires a revisiting of the marginal utility of the test set vectors, albeit this time no longer in the context of a single test set but rather in the context of a test domain group.

Our proposed algorithm starts off by introducing an optimal approach for test scheduling under concurrency constraints that minimizes synchronization induced test set

deviations. It then rebalances the allocated time by equalizing the utility of the vectors across domain groups, thus delivering a computationally effective approach that maximizes the utility of the allocated test resources.

In more detail, we initially disregard the concurrency consideration to perform a test resource allocation, referred to as *initial test time allocation*, that delivers the optimal delay test quality without any concurrency constraint, setting an upper bound on the attainable quality. The concurrency constraint is subsequently introduced while aiming at the minimization of the deviation from the delay test quality level obtained by the *initial test time allocation.* The minimization of the deviation from the attained delay test quality level is ensured with a two-step process. First, an optimal schedule of the concurrently tested domains is identified based on the initial test time allocation. This scheduling step, referred to as *domain scheduling*, minimizes the unutilized test time slacks that potentially exist due to an unequal allocation of the test time to the domains. Furthermore, since the *initial test time allocation* is performed to enable the full utilization of the test time budget, the unutilized time slack in the schedule can result in test budget overflows. Subsequently, a final step, referred to as *slack avoidance & final allocation*, applies the optimal test time allocation process once again yet this time on the concurrently tested domain groups determined during the domain scheduling phase, instead of on individual domains. This final step eliminates the unutilized timing slacks and keeps the test time within the budget while ensuring the delivery of the highest test quality under the concurrency consideration.

Initial test time allocation:

Subsequent to the test generation and test quality estimation for the individual frequency domains, an initial test time allocation is performed, assuming all test resources can be freely used. Since N horizontal channels exist, the total test time available is $N \times TT_{target}$,

each channel providing TT_{target} test time. Alternatively, if T test time is allocated to a domain, since N domains can be concurrently tested, this particular domain is effectively using $(1/N)\times T$ test time resources from the overall test time budget, TT_{target}. Since test time is being allocated with no restriction in this step, the optimization formulation is similar to the one in Section 6.3 while taking into account the increased total test time available due to concurrency as follows.

$$\textit{minimize} \quad \sum_{i=1}^{c}\sum_{j=1}^{f_i} Q_{i,j}\left(x_{i,j}\right)$$

subject to

$$\sum_{j=1}^{f_i} \frac{1}{N} \times x_{i,j} \times SC_{i,j} \times SP_{i,j} \le TT_i \;\; \forall i \in \{1,\ldots,c\}$$

$$x_{i,j} \times SC_{i,j} \times SP_{i,j} \le TT_i \;\; \forall i \in \{1,\ldots,c\} \forall j \in \{1,\ldots, f_i\}$$

$$\sum_{i=1}^{c} TT_i \le TT_{\text{target}}$$

TT_i denotes the total test time assigned to core *i.* As can be noted, the first constraint considers the concurrent testing of the domains by obtaining the effective test time usage which corresponds to 1/N of the total test time. The second constraint ensures that a single domain does not exceed the total test time assigned to a core. This constitutes a convex optimization problem, similar to the one in Section 6.3 with minor modifications, and can be efficiently solved.

Domain scheduling:

The initial test time allocation is followed up by the *domain scheduling* phase to determine the domains to be tested in parallel. Domain scheduling creates *domain groups* with

up to *N* domains each while minimizing the unutilized test time slacks to ensure the smallest deviation from the initial test time allocation. Optimal test scheduling of all *n* frequency domains, wherein $n = \sum_{i=1}^{c} f_i$, under the concurrency constraints of test architecture can be obtained as follows.

Domain Scheduling with Architectural Concurrency Limit:

- Order the domains based on their allocated test times.
- For each domain (starting from the first one on the list)
  - If there is no available domain group (vertical channel) or all domain groups of the corresponding core have already *N* domains, start a new domain group for this core and add this particular domain to it.
  - Otherwise, assign this domain to the group of the corresponding cores with an available slot.

The domain ordering and scheduling can be performed at *O(nlog(n))* and *O(n)* time, respectively. Consequently, the complexity of this phase is *O(nlog(n)).* The schedule however may leave unutilized test time slacks as not all domains in the same group are allocated an identical test time by the initial test time allocation. The test time slacks can subsequently result in a deviation from the test time budget although the deviation is minimized with the optimal scheduling algorithm.

An example of a domain schedule for a design with 8 frequency domains, 4 per core, and the ability to concurrently test 2 domains is shown in Figure 6.6(a). The start times of the concurrently tested domains are aligned as expected and the overall test time is minimized. However, since the allocated test times for the concurrently tested domains in this particular example are not equal during the domain scheduling, unutilized slacks abound in the schedule,

resulting in the overall test time exceeding the budget. In order to eliminate the unutilized test time slacks and keep the total test time within the overall budget, a *slack avoidance & final allocation* step follows the *domain scheduling*.

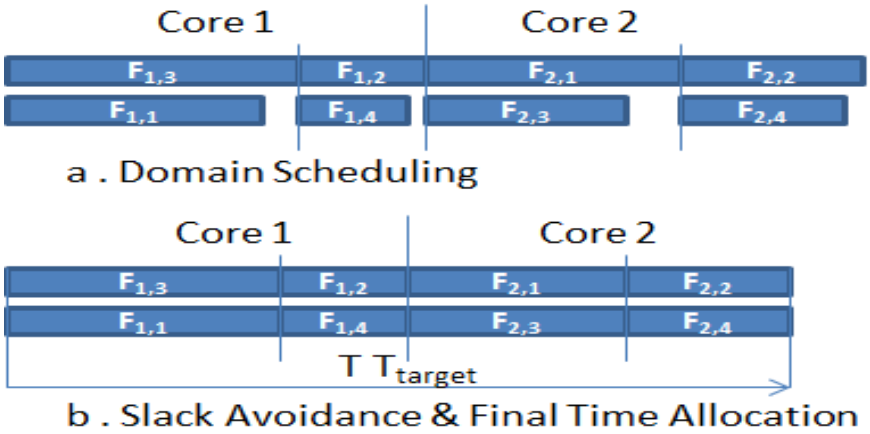


Figure 6.6 Domain scheduling and final test time

Slack avoidance & Final allocation:

This final step starts with obtaining a cumulative test quality function for each frequency domain group. For example, $F_{1,3}$ and $F_{1,1}$ form one of these domain groups in Figure 6.6(a). If the individual test sets generated in the first step can be merged, the user can simply merge the tests of the domains of each group and add the individual test quality functions to obtain the cumulative test quality function per domain group. In practice, however, since scan compression is a *de facto* standard nowadays, it may not be possible to merge the test sets due to the scan compression constraints. In that case, a new test set for each domain group is generated and the cumulative test quality for each group is estimated.

Once delay test quality functions per domain group are obtained, the same convex optimization method discussed in Section 6.3 is applied to identify the new test time allocation

for each domain group. The only change in the optimization formulation is the use of domain groups and their corresponding test quality functions instead of individual domains. The final test time allocation and domain schedule for the example can be seen in Figure 6.6(b) for the initial schedule in Figure 6.6(a). No unutilized test time slacks remain while the total test time is kept within the test time budget.

6.4.2 Test Time Allocation and Scheduling with Test Time, Concurrency and Power Constraints

In addition to the architectural concurrency limit modeled in Section 6.4.1, power also plays a crucial role in constraining the possible domain schedules. Excessive switching activity generated during testing can result in thermal and power supply conditions beyond those of the functional mode of operation. Elevated peak current draw and power dissipation due to the excessive switching activity can cause large instantaneous voltage droop and an increased temperature, increasing the circuit delays beyond those of functional mode during delay testing and subsequently resulting in unnecessary yield loss [128]. Evidently, test time allocation and domain scheduling has to consider power limits to create reliable test conditions.

As illustrated in Figure 6.3, the concurrently tested domains at any particular moment should remain within the power envelope. The power constraint, P, is modeled in this work as the switching activity level, and represented as the ratio of active circuit nodes to the total circuit nodes. Similarly, switching activity level for domain j in core i, $p_{i,j}$, is the active node level when this particular domain is singly tested. Although we utilize the widely used switching activity level in this work, any other test mode power estimation method could be easily substituted.

The test time allocation and domain scheduling process had to consider not only the test time budget and the architectural concurrency limits on the horizontal and vertical dimensions, respectively, as depicted in Figure 6.5, but also an additional power constraint on the vertical dimension. Given a power level for each domain, $p_{i,j}$, and an overall limit, P, the cumulative power level of all domains assigned to any vertical channel in Figure 6.5 is constrained to be within the overall power limit, P. Optimal test time allocation and domain scheduling with the power constraint can be posed as the following optimization problem.

minimize $\sum_{i=1}^{c}\sum_{j=1}^{f_i} Q_{i,j}\left(x_{i,j}\right)$

subject to

$$\sum_{h=1}^{N}\sum_{v=1}^{f_i} y_{i,j,h,v} = 1,\ \forall i \in \{1,\ldots,c\} \quad \forall j \in \{1,\ldots,f_i\}$$

$$\sum_{j=1}^{f_i} x_{i,j} \times y_{i,j,h,v} \times SC_{i,j} \times SP_{i,j} \le TT_{i,v},\ \forall i \in \{1,\ldots,c\} \qquad \forall h \in \{1,...,N\} \forall v \in \{1,...,f_i\}$$

$$\sum_{j=1}^{f_i}\sum_{h=1}^{N} p_{i,j} \times y_{i,j,h,v} \le P,\ \forall i \in \{1,\ldots,c\} \quad \forall v \in \{1,\ldots,f_i\}$$

$$\sum_{i=1}^{c}\sum_{v=1}^{f_i} TT_{i,v} \le TT_{\text{target}}$$

$$y_{i,j,h,v} \in \{0,1\}$$

In this optimization formulation, the second and fourth constraints limit the overall test time usage to the test time budget while the first constraint ensures that each domain is only assigned to a single available slot. The third constraint identifies the domains assigned to each vertical channel and ensures that the total power limit of these domains does not exceed the overall power budget. This particular power constraint further increases the computational

complexity of this optimization problem. As in the optimal test time allocation and scheduling problem with only an architectural concurrency constraint, we propose an efficient, incremental optimization method to identify the test resource allocation with the additional power constraint. It is even more imperative to develop an efficient methodology for the solution of this problem that minimizes the deviation from test effectiveness optimality.

As one can recollect, the solution to the multidimensional tradeoff problem outlined in the previous subsection relied on an initial partitioning of test sets into domain groups that was able to deliver a schedule that minimized test set deviations. Yet the power constraint can invalidate such optimal solutions as they can possibly contravene the power budget. Not only the power constraint can invalidate such solutions, but furthermore it questions even the initial test time allocation as stringencies of power budgets that supervene the strictness of the architectural concurrency limit may result in reduced initial test time allocations.

The proposed algorithm starts off by assessing the relative stringencies of the power and concurrency constraints, possibly resulting in reduced test times as stringent power constraints could introduce unutilized horizontal channels in addition to test time deviation induced slacks. This formulation is followed up by a test scheduling approach that again minimizes test time deviations in domain groups, yet this time under power constraints. The optimal algorithm introduced in the previous section needs to be modified to handle the twin constraints of test time deviation minimization and power constraints in each domain group. A rebalancing of the test times across domain groups eradicates the concurrency and power induced test time bloats, while equalizing the marginal utility of the vectors in each domain group.

The optimal test time allocation and domain scheduling with the total test time budget, the architectural concurrency and the power constraints is performed in three incremental steps which we define as follows. The *initial test time allocation* step considers not only the

architectural concurrency limit but also imposes a power limit during test time allocation to the domains. Similarly, *domain scheduling* does not simply assign domains to an available slot, but also considers the power limit in the vertical dimension while minimizing the deviation from the quality level obtained by the initial test time allocation. After the initial test time allocation and domain scheduling, the *slack avoidance & final allocation* step follows the same convex optimization solution presented in Section 6.4.1. The new *initial test time allocation* and *domain scheduling* steps with additional power constraint are performed as follows.

Initial test time allocation:

If a domain is allocated T test time, since up to N domains can be concurrently tested, this particular domain is effectively using $1/N \times T$ test time resource from the test time budget, TT_{target}, as previously discussed. The power constraint, however, can hinder the efficient use of all available test time. When a domain with $p_{i,j}$ power level is assigned to a vertical channel, it effectively uses $p_{i,j}$/P of the available power budget. If $p_{i,j}$/P exceeds 1/N, it endangers the full utilization of all available horizontal channels. The *initial test time allocation* leverages this observation and uses a more stringent utilization ratio based on both architectural and power limits for each domain to guide the test time allocation. $n_{i,j}$, maximum of 1/N *and* $p_{i,j}$/P as represented below, is used as a normalization factor in the test time allocation formulation.

$$n_{i,j} = \max\left(\frac{1}{N}, \frac{p_{i,j}}{P}\right)$$

$$\textit{minimize} \quad \sum_{i=1}^{c}\sum_{j=1}^{f_i} Q_{i,j}\left(x_{i,j}\right)$$

subject to

$$\sum_{j=1}^{f_i} n_{i,j} \times x_{i,j} \times SC_{i,j} \times SP_{i,j} \le TT_i \ \forall i \in \{1,\ldots,c\}$$

$$x_{i,j} \times SC_{i,j} \times SP_{i,j} \le TT_i \ \forall i \in \{1,\ldots,c\} \forall j \in \{1,\ldots, f_i\}$$

$$\sum_{i=1}^{c} TT_i \le TT_{\text{target}}$$

The first constraint uses an architectural and power based normalization factor, $n_{i,j}$, to estimate the effective test resource. This is a convex optimization problem as well, similar to the initial test time allocation formulation in section 6.4.1, and can be efficiently solved.

Domain scheduling:

Domain scheduling with the utilization of test time allocations in the previous step aims at minimizing the overall test time usage (horizontal direction) by determining which domains are tested concurrently. In addition to the architectural limit on the number of domains that can be tested concurrently, the schedule has to consider the power usage of each test, $n_{i,j}$, as represented as the width of the tests in Figure 6.3. Consequently, not only should the number of domains in any vertical channel not exceed *N*, but also the cumulative width of all tests in a vertical channel should not surpass the power limit, *P*.

This problem is a revised version of a problem known as two-dimensional strip packing [129]. In the two-dimensional strip packing problem, a set of rectangular items, similar to the delay test representations of each domain in Figure 6.3, is packed to a rectangular strip of a fixed width and an infinite length with the objective of packing all items on the strip while minimizing the length of the rectangular strip. This is known to be a NP-hard problem [129] with numerous heuristics proposed in the literature.

Our domain scheduling problem is similar to the two-dimensional strip packing with two additional constraints. First, the number of total items (i.e. domains) assigned to any vertical dimension is limited to *N*. Secondly, the start time of all domains in a vertical channel needs to be identical. Many of two-dimensional strip packing heuristics align the item start positions on the vertical dimension (referred to as levels), thus automatically satisfying the second constraint; the incorporation of the first constraint is straightforward as we illustrate later on.

A commonly used two-dimensional strip packing heuristic is the First-Fit Decreasing Length (FFDL) algorithm [129] [2]. Items are sorted in decreasing order by their length and processed in this order. Each item is placed at the first level where it fits within the width limit of the strip. The same heuristic with a limit of *N* on the number of items placed on each level is utilized in this work. The complexity of this algorithm is *O(nlog(n))*, wherein *n* is the number of items (domains). Strip packing is an active research area. Any of the more advanced heuristics could be equally adapted as well.

Domain Scheduling with First-Fit Decreasing Length:

- Order the domains based on their allocated test times.
- For each domain (starting from the first one on the list)
  - If there is a domain group (vertical channel) for the corresponding core with less than N domains assigned and with a sufficient power margin to accommodate the current domain, assign this particular domain to this domain group.
  - Otherwise, start a new group for this core and assign this domain to it.

[2] The common definition of the two-dimensional strip packing problem assumes that the infinite side of the strip is laid out on the vertical dimension; subsequently, this heuristic is commonly referred to as First-Fit Decreasing Height (FFDH) algorithm in literature.

6.4 Experimental Results

The proposed delay test quality optimization method is evaluated in this section for various frequency domain configurations. We use a frequency domain from an industrial circuit in our experiments. A commercial ATPG tool is utilized to generate transition patterns and to calculate the corresponding SDQL and defect detection level for this circuit. We set the delay defect distribution to Eq. (6.1) in our experiments, which is provided in [13] based on the data in [118].

$$F(s) = 1.58 \times 10^{-3} \times e^{-2.1s} + 4.94 \times 10^{-6} \quad (6.1)$$

We generate various configurations of the frequency domain by altering four parameters, namely, the frequency, the path lengths, the scan chain length and the scan speed. For the purposes of this experiment, the scan chain length of a domain is constrained to be one of 4 possible chain lengths. Given that the shortest scan length is *scl*, the chain lengths of the remaining configurations are set to *2*scl*, *3*scl* and *4*scl*. Similarly, the scan speed of a domain can be set to one of 4 possibilities. Given that the slowest scan speed is *ss*, the rest are set to *2*ss*, *3*ss* and *4*ss*. Scan chain length and scan speed affect the cost of each scan pattern. 4 different frequencies of 250 MHz, 500 MHz, 750 MHz and 1GHz are generated by scaling the timing information. Finally, two different path length distributions, denoted as short and long paths, are supported. The test time budget is set to the cumulative test time needed to obtain 80% of achievable coverage for each domain.

In the first set of experiments, we take a close look at the delay test quality tradeoff across the domains by focusing on simple domain pairs instead of a large set of domains in order to analyze the trend of overall test quality change as we alter the test time distribution

between a couple of domains. A sample set of 3 domain pairs is generated by changing the parameters as defined in Table 6.1. Only the chain lengths differ in the domain pair of *Config 1* while keeping everything else identical. The domain with the long scan chains subsequently consumes more test time to deliver the same quality than the domain with the short scan chains. The path lengths, in addition to the chain lengths, are also altered between domain pairs in *Config 2*, boosting the variation between the domains. Finally, the variation is further increased in *Config 3* by changing the frequency, scan chain and path lengths between the domains. The scan shift speeds are kept identical in this particular experiment. The detection level as a function of test patterns is computed for each domain by a commercial ATPG tool. The proposed delay test quality optimization technique is applied to each domain pair in these three different configurations to analyze the effect on overall test quality as the test resource allocation between the domain pairs is altered.

Table 6.1 Domain configurations

	Domain A			**Domain B**		
	Freq.	Chain Length	Path Length	Freq.	Chain Length	Path Length
Config 1	500	Short	Long	500	Long	Long
Config 2	500	Short	Long	500	Long	Short
Config 3	500	Short	Long	250	Long	Short

Figure 6.7 depicts the overall defect detection level change as more test time is allocated to *Domain A* in the domain pair while keeping the overall test time usage intact. The starting point is the original test time allocation with an equal transition fault coverage for both

Domain A and Domain B. As the allocation of the overall test time is altered by giving more weight to *Domain A*, the overall defect detection level starts to increase as can be seen in Figure 6.7. The rate of the additional quality gain however diminishes as increasingly more test resources are allocated to Domain A, eventually reaching a peak point in the overall test quality. The peak points are the optimal test allocation points for the domain pairs as can be seen in Figure 6.7. As the variation between domains increases from *Config 1* to *Config 3*, the level of test quality improvement delivered by the proposed method increases as expected. In the case of *Config 3*, a test quality improvement of almost 30% is attained by just altering the test time allocation between *Domains A and B* with no increase in the original test time at all.

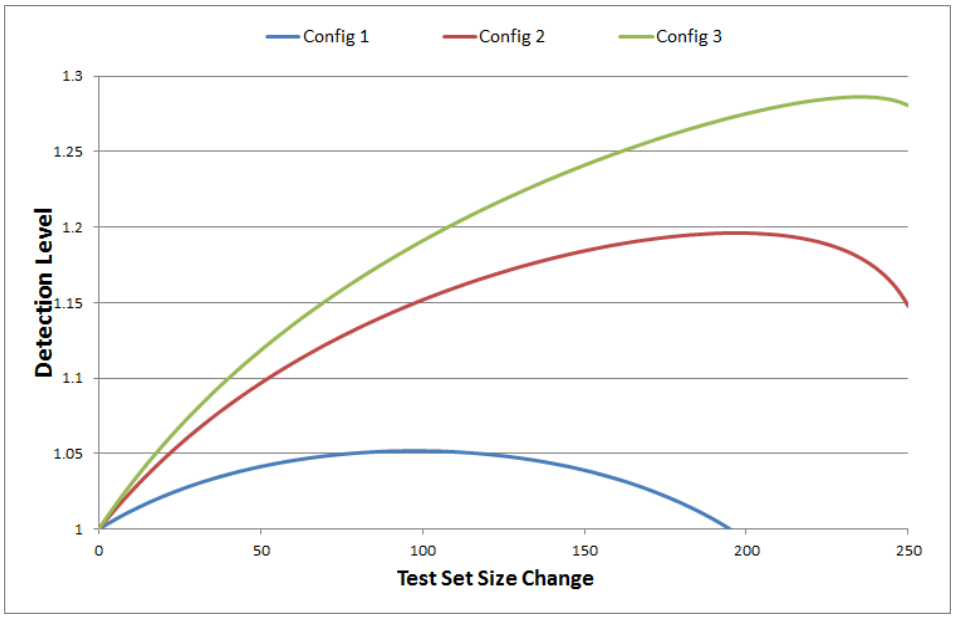


Figure 6.7 Detection level change as test time allocation altered

In the second set of experiments, 3 different SOC configurations composed of 8, 32 and 128 domains are created. The domains in these SOC configurations are obtained by changing 4 different parameters (i.e., frequency, path length, scan length and speed) as discussed above.

8 domains: 4 frequencies × 2 path lengths

32 domains: 4 frequencies × 2 path lengths × 4 scan lengths

128 domains: 4 frequencies × 2 path lengths × 4 scan lengths × 4 scan speeds.

The improvement in defect detection level for the sequential testing of all domains can be seen in Figure 6.8, denoted as "4 Frequencies". An improvement of around 15% is observed while maintaining the original test time. In an effort to increase the variation among domains and analyze the effect on the test quality improvement, the frequencies of all 500MHz and 750MHz domains in these configurations are changed to 250MHz and 1GHz, respectively. As the differences between domains increase, the test quality improvement is further boosted as expected. The improvement surpasses the 20% level as depicted in Figure 10, denoted as "2 Frequencies".

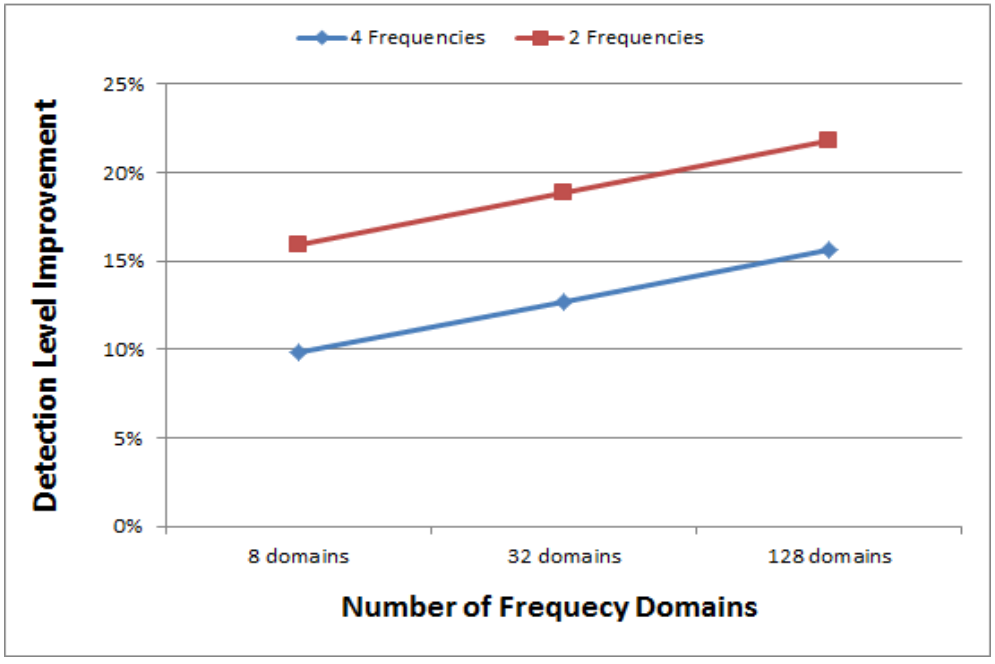


Figure 6.8 Quality improvement for various SOC configurations

In the final set of experiments, test quality improvement with the concurrent testing of domains under architectural and power constraints is evaluated. The 128-domain configuration that is divided into 16 cores is utilized. The scheduling algorithm used in the proposed method is applied to the transition test sets of the domains to obtain the test time for the conventional delay testing. Subsequently, the proposed test time allocation and scheduling method is applied to improve the quality while using test time identical to that of the conventional delay testing.

We start the evaluation with no consideration of power concerns during concurrent testing. The defect detection improvement delivered by the proposed method for the architectural concurrency limits, *N*, of 2 and 4 is depicted in Figure 6.9, denoted as "*N=2, P=N/A*" and "*N=4, P=N/A*", respectively. A substantial improvement, exceeding 20%, in the test quality is observed while consuming the identical test time to the original test set. The quality improvement is slightly lower during the concurrent testing of 4 domains as expected due to the additional constraints of an increased concurrency level.

In the remaining part of this set of experiments, the proposed method is evaluated with a power constraint, *P*, in addition to the architectural constraint, *N*, during the concurrent testing. The power consumption of each domain is assumed to be proportional to their respective frequencies and the highest power consumption among all domains is denoted as p_{max}. Initially, the overall power limit, *P*, is set to a very large number, effectively allowing the architectural constraint, *N*, being the only restriction on the attainable level of concurrency. As can be seen in Figure 6.9, the quality improvement for "*N=4, P=Large*" is identical to the results reported for "*N=4, P=N/A*" as expected, with only the architectural limit determining the quality improvement level. Subsequently, the overall power limit, *P*, is set to p_{max}, $1.5*p_{max,}$ and $2*p_{max,}$ while keeping the architectural limit, *N*, at 4. The delay test quality improvement levels with these particular constraints are reported in Figure 6.9, denoted as

"*N=4, P=p_{max}*", "*N=4, P=1.5*p_{max}*" and "*N=4, P=2*p_{max}*", respectively. A substantial level of delay test quality improvement, reaching up to %15, even with strict power limits is observed. As the power limit and the variation among domains increase, the attainable level of the delay test quality improvement delivered by the proposed method increases as expected. Evidently, the continuous integration of an increasingly larger number of domains with distinct characteristics in the state-of-the-art SOCs can be expected to boost the quality improvement level delivered by the proposed delay test resource allocation method.

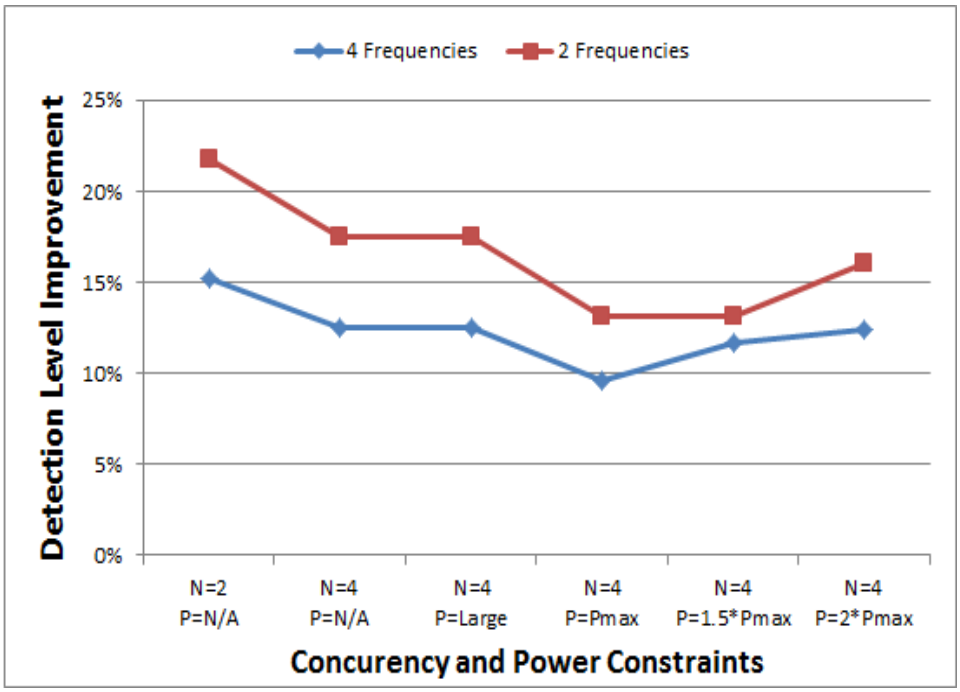


Figure 6.9 Quality improvement with concurrency

6.5 Conclusions

Today's SOCs are composed of hundreds of frequency domains with distinct characteristics that affect delay test quality. In order to extract the highest value from the

available test resources, a carefully crafted allocation of the test resources to each domain based on the domain characteristics as well as a full utilization of the concurrent test support while adhering to power limits is necessary.

A technique for the identification of the test resource allocation and the schedule of the concurrently tested domains in order to maximize overall delay test quality within the limits of available test resources and power is proposed in this chapter. The delay test quality of each domain while considering distinct domain characteristics is initially estimated. Subsequently, optimization formulations as well as efficient test time allocation and concurrent test scheduling methods based on convexity and fast scheduling algorithms are provided.

Experimental results show that the proposed method can deliver a substantial delay test quality improvement in comparison to the conventional, fault coverage driven approach while utilizing an identical test time. As the level of integration and complexity of SOCs increases, the test quality improvement delivered can only be expected to grow.

Chapter 6, in part, is a reprint of the material as it appears in *B. Arslan and A. Orailoglu, "Delay Test Resource Allocation and Scheduling for Multiple Frequency Domains", VLSI Test Symposium, 2012*; and in *B. Arslan and A. Orailoglu, "Power-Aware Delay Test Quality Optimization for Multiple Frequency Domains", submitted to IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. The dissertation author was the primary investigator and author of these papers.

Chapter 7

CircularScan: A Scan-Based Test Architecture for Test Cost Reduction

Scan-based test architectures enhance the controllability and observability of the design, thus helping to keep the test generation complexity of today's large circuits within practical limits and boosting the maximally attainable levels of fault coverage. Although the aforementioned benefits of scan-based designs promote its wide acceptance, test cost is inevitably increased as a result of the serial access mechanism. While the adaptive cost-effective test selection and the carefully crafted utilization of test resources through the methods presented from Chapter 3 to Chapter 6 help in making great strides in test cost and quality optimization, the high cost of each test pattern in scan-based designs poses a serious obstacle in pursuit of an effective yet cost efficient test methodology. In an era where scan-based test architecture has become an indispensable part of test strategies, it is essential to reduce test cost in order to fully enjoy the benefits of scan architecture and help pave the way for an effective yet efficient test methodology.

Employing multiple scan chains can reduce test application time; nevertheless, the corresponding increase in the number of scan I/O pins necessitates a higher cost automatic test equipment (ATE) with a high pin count and memory bandwidth. It is paramount therefore to ensure that the approach for test cost reduction limits the number of scan I/O pins. In practice,

even the compacted deterministic test patterns generally consist of a small number of specified bits, in the range of 1%-5% as reported in [130]. Although the specified bit density of the test patterns is quite low, the traditional scan architecture does not allow the exploitation of this property as a result of serial load of all bits of the pattern.

In this chapter, we propose a novel scan architecture, denoted as *CircularScan*, that can deliver a high test cost reduction level while utilizing a number of scan I/O pins through the exploitation of the fact that only a small number of bits is specified in each pattern. *CircularScan* enables the use of the captured response of the previous pattern as a template for the next pattern and a cost-effective addressing scheme is developed to efficiently update the template in order to obtain the next pattern. Since the specified bit density is low for test patterns, it suffices to update only a small set of bits on the template.

Section 7.1 overviews the motivation and the preliminary form of the proposed scan architecture for test cost reduction. Section 7.2 outlines the proposed addressing scheme and Section 7.3 presents the algorithm used to identify the addressing mechanism. Section 7.4 discusses the test application methodologies. Section 7.5 provides the experimental results and a brief set of conclusions is drawn in Section 7.6.

7.1 Motivation

In traditional scan architectures as depicted in Figure 7.1(a), each test pattern is serially shifted in one bit at a time per scan chain. This particular access mechanism necessitates spending one cycle for each slice of the test pattern whether it is specified or not. An increase in the scan chain count may reduce the test application cycles by shrinking the scan chain lengths, yet it comes at an increase in the scan I/O pin count, raising the cost of the ATE. Although test patterns consist of a small number of specified bits, the traditional scan

construction cannot exploit this property and the specified and unspecified bits of a test pattern are applied identically during the test application process. An architecture that enables the efficient load of essentially only the specified bits of test patterns promises quite high levels of reduction in test cost. However, a method is needed to fill the unspecified bits of the test patterns.

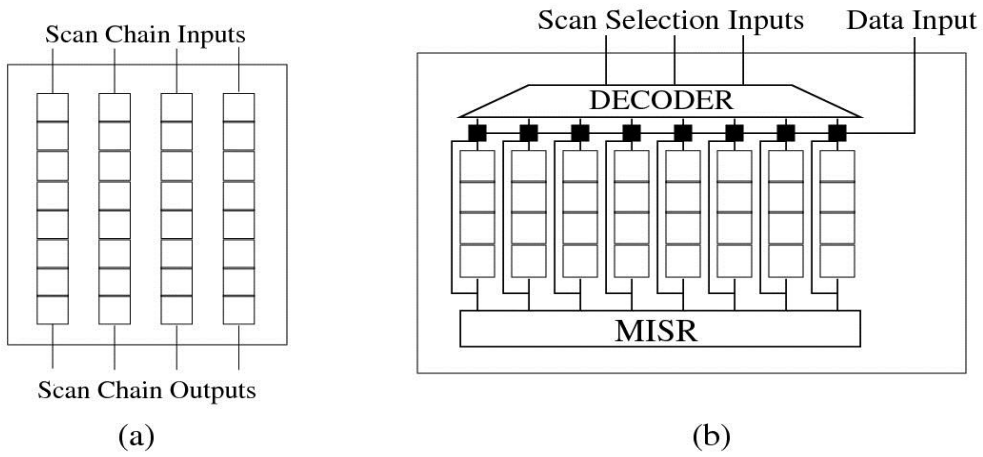


Figure 7.1 Traditional and *CircularScan* architecture

A circular configuration of the scan chain, wherein the outputs of the scan chains are connected to their own inputs, can provide a solution for filling the unspecified bits of the test patterns by enabling the use of the captured response of the previously applied pattern as a template for the subsequent pattern. While the captured response is shifted out, the data is also routed back to the scan inputs, resulting in preservation of the previous state when the shift-out is completed. Consequently, only the update of the specified bits of the test pattern that are at conflict with the template may suffice to apply the subsequent pattern. Although the captured response can be used as a template, in order to be able to apply only the specified bits of a test pattern, this configuration should be accompanied with an addressing scheme to select and enable the efficient update of the specified bit positions on the template.

Ideally, the addressing needs to be able to index all the specified bit positions that are at variance with the template when applying the current slice of the test pattern, updating all conflicting bits of the current test slice with actual logic values. Nevertheless, such an addressing scheme becomes quite complex and quite a high number of scan input pins is necessitated. The complexity and high scan input count requirement of the addressing mechanism may be reduced by allowing multiple rotations of the scan chains when applying a test pattern. Essentially, multiple rotations of the scan chains distribute the update of conflicting bits among different rotations, reducing the complexity of the addressing scheme.

A preliminary form of the proposed *CircularScan* architecture is depicted in Figure 7.1(b), in which only one conflicting bit can be updated at each cycle. All scan input pins but one in this preliminary form are used as a single addressing mechanism to select a particular scan chain and the remaining scan pin, *data input*, is utilized to load test data. Consequently, the captured test response of the previous pattern that is kept in the scan chains with the help of the circular structure is updated one bit at a time to load the specified bits of the next test pattern that are at conflict with the captured response. Although the output compactor is depicted as a MISR in this figure, numerous other output compaction techniques such as XOR based compactors could have been used as well.

Although the outlined properties of *CircularScan* promise significant savings in test cost, the capabilities of the system as presented in this preliminary form are quite restricted. The ability to update only one bit of a test slice at each cycle, due to a single addressing mechanism, limits the achievable test cost reduction levels significantly. Even if quite a small set of test slices has a large number of conflicting bits, since only one bit of a test slice can be set for each rotation of the scan chains, a high number of scan chain rotations are required to be able to apply the test pattern.

An addressing scheme which enables the simultaneous update of the multiple conflicts at a single test cycle by exploiting the possible parallelism among the updates is necessary to boost the attainable level of test cost reduction. A method for the construction of an addressing scheme that enables, in the minimal number of addressing bits, the update of *all* conflicting bits of a test slice in one cycle is proposed in this chapter. While this particular method ensures the minimal use of addressing bits under the single rotation constraint, slight relaxations of this constraint offer overall superior solutions. Subsequently, we follow up with a methodology for exploiting the tradeoff space between the simultaneous multiple bit update and the multiple rotations of the scan chains, resulting in a cost-effective addressing scheme as presented in the subsequent sections

The proposed scheme is generated by analyzing a predetermined test set. Nevertheless, the dependence on a given test set does not jeopardize the application of the test set by the proposed scheme when the original test set is modified. Although the addressing mechanism is built to generate the best application strategy for the original test set, it is flexible enough to support the application of any random test pattern. This property of the proposed scheme strictly distinguishes it from the previously published test set dependent methodologies, wherein test pattern updates may necessitate reconstruction of the design or result in an inability to supply the test vectors.

7.2 Proposed Addressing Scheme

As discussed in the previous section, the main obstacle that limits the test cost reduction level in the preliminary form of *CircularScan* architecture as depicted in Figure 7.1(b) is the use of a single addressing mechanism. Since only one bit of a test slice can be updated for each rotation of the scan chains, an exceedingly large number of rotations of the

scan chains is required, unnecessarily limiting the achievable test cost reduction levels. Furthermore, since the number of internal scan chains is exponentially increased by the use of a decoder, if the scan pin count is quite high, the number of scan chains that can be addressed may not be practical, wasting some of the scan input pins.

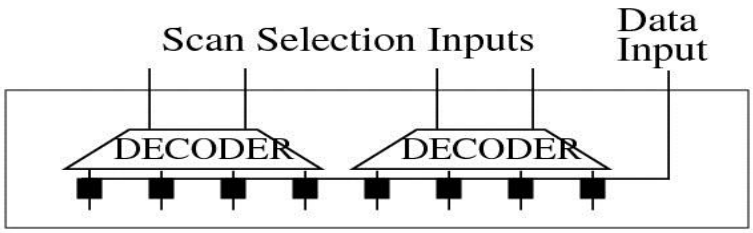


Figure 7.2 Multiple decoder addressing mechanism

Since the specified bit density even post-compaction is in the range of 1-5%, and the availability of a template further reduces the conflicting bits, the scan chains can be divided into smaller subsets, wherein only one conflicting bit exists on each test slice for each scan chain group; a decoder-based addressing mechanism can be assigned to each scan chain group, enabling the addressing of multiple bits each test cycle. A sample new configuration can be seen in Figure 7.2. If N scan chains exist, the following equation spells out the required number of scan inputs, S_{SD}, when a single decoder is employed, wherein an additional address is used to denote the possibility of no toggle.

$$S_{SD} = \lceil \log_2(N+1) \rceil \qquad (7.1)$$

If scan chains are divided into k groups and each scan chain group consists of N_i scan chains with an additional pin for no toggle, the total number of scan inputs, S_{MD}, can be determined by Eq. (7.2). Each decoder can be straightforwardly implemented by using $2^{\lceil \log_2(N_i+1) \rceil} - 1 \approx N_i$ *1-to-2* decoders, wherein an *1-to-2* decoder is implemented by 2 AND gates and an INVERTER.

$$S_{MD} = \sum_{1}^{k} \lceil \log_2 (N_i + 1) \rceil \quad wherein \ N = \sum_{1}^{k} N_i \tag{7.2}$$

Although the use of the multiple decoders evidently results in an increase in the number of scan inputs, a single rotation for each test vector suffices to apply it. For *T* test vectors, longest scan chain length of *L* and average number of rotations for the single decoder case of *R*, the test application time reduction in comparison to the traditional scan architecture for the single and multiple decoder cases can be seen in Eq. (7.3) and Eq. (7.4), to be, respectively:

$$\text{TestTime}_{SD} = T \times L \times R$$

$$\text{Reduction}_{\text{SD}} = \frac{T \times L \times R}{T \times N \times L / S_{SD}} = \frac{R \times S_{SD}}{N} \tag{7.3}$$

$$\text{TestTime}_{MD} = T \times L$$

$$\text{Reduction}_{\text{SD}} = \frac{T \times L}{T \times N \times L / S_{SD}} = \frac{S_{SD}}{N} \tag{7.4}$$

Essentially, if the reduction ratio in the rotation count exceeds the ratio of the increase in scan input pin count, the use of the multiple addressing mode delivers improved results. Nevertheless, since two scan chains are placed into different scan chain groups if they need to be updated simultaneously for even a single slice of a single test vector, it can be expected that the scan chain groups will be quite fragmented and that the increase in scan chain count may overwhelm the reduction in the rotation count, delivering essentially no improvement. Multiple scan chain rotations deliver a quite powerful tool to deal with the outlined limitation. If multiple rotations are allowed for some test vectors, the conflicts between some scan chains can be broken and larger subsets of the scan chains, reducing scan pin count, can be created

with a small increase in the rotation count. The single addressing mechanism with multiple rotations and the multiple addressing mechanism with a single rotation constitute two extremal points of a spectrum that can be more fully observed in Figure 7.3. An intermediate point, namely, larger scan chain groups with a smaller number of multiple rotations, may deliver a substantial reduction in the test cost. If *m* scan groups require S_{MD}^{m} scan input pins, calculated with an equation similar to Eq. (7.2), and the rotation count is, on average, R^{m}, Eq. (7.5) can be used to depict the reduction in the test cost in comparison to the traditional scan architecture.

$$\text{Reduction} = \frac{R^m \times S_{MD}^m}{N} \quad \textit{wherein} \quad S_{\min} \le m \le S_{\max}, 1 \le R^m \le R_{\max} \tag{7.5}$$

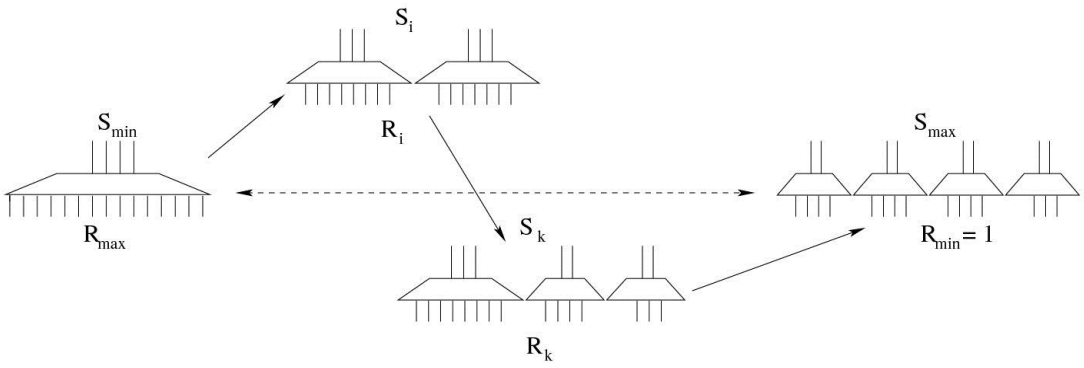


Figure 7.3 Addressing space

Since the data input pin is only used to load test data when there is a conflict between the test vector and the template, the scan architecture is changed to flip the current value in the template instead of loading from the data input, eliminating one of the scan pins. Furthermore, broadcasting is quite a powerful mechanism for the faults that can be tested under the constraints of the broadcast mode. Broadcast mode can also be used to eliminate potential X's in the captured responses due to non-scan elements. Consequently, a broadcast mode is added

to the current architecture. One of the combinations in one of the decoders is used to select the broadcast mode and the broadcast data is loaded from any one of the remaining scan inputs in the broadcast mode. The final scan architecture is depicted in Figure 7.4.

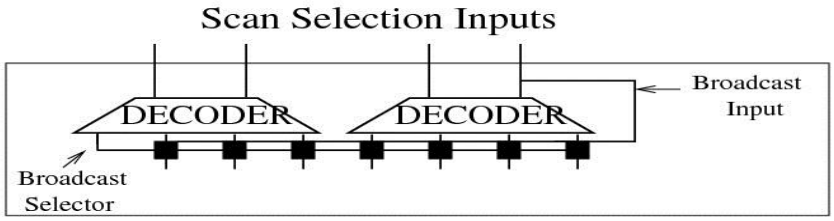


Figure 7.4 Proposed configuration

Perhaps, a brief summary of the proposed technique can be obtained by noting that the search for a cost-effective addressing scheme aims at creating larger scan groups, reducing scan input pin count, by resolving some of the conflicts with the help of multiple rotations in order to attain a higher test cost reduction level. Section 7.3 presents in detail the algorithm used in the search of this particular addressing scheme.

7.3 Addressing Space Search

The proposed algorithm analyzes test data and aims at generating scan chain groups, wherein no two scan chains require the update of the template at the same test cycle. Two scan chains are called *incompatible* if they conflict with the template for at least one test slice in the test set. They are deemed *compatible* if there is no conflict for any test slice in the test set. An *incompatibility graph* represents the conflicts between the scan chains. The nodes in the incompatibility graph denote scan chains and the edges join incompatible scan chains. A set of nodes that are mutually compatible can be controlled with a single decoder with no need for multiple rotations. Consequently, if multiple decoders with a single rotation are targeted, the nodes of the incompatibility graph need to be partitioned into groups that require the minimum

number of scan inputs, wherein nodes in each group are mutually *compatible*. This particular problem is known to be NP-complete and an $O(n^3 log(n))$ heuristic can be found in [131]. The resulting addressing scheme is one extremal point of the design space that is depicted in Figure 7.3, namely, that of multiple decoders and a single rotation.

Our aim is to identify an addressing scheme that possibly lies between the addressing mechanism that can be found by the algorithm outlined in the last paragraph and the single decoder addressing mechanism as illustrated in the preliminary form of *CircularScan* in Section 7.1. If an edge in the conflict graph is removed by allowing multiple rotations, the corresponding nodes may be embedded within the same scan group. However, multiple rotations are required for each test vector that has a conflict at a test slice for the scan chains corresponding to these particular nodes. As can be noticed, the number of extra rotations is determined by the number of test vectors that exhibit conflicts at the corresponding scan chains. In order to include that information, a *weighted incompatibility graph* is introduced. In the weighted incompatibility graph, the weight of the edges represents the number of test vectors in which the corresponding scan chains are incompatible. It should be noticed that the number of test slices at conflict within each test vector are not considered. Consequently, if the scan input count is reduced by the same amount when either of the two edges is removed from the incompatibility graph, the removal of the one with the smaller weight is desirable as it helps minimize the rotation count.

The proposed algorithm to search the possible addressing space constitutes an iterative method. At each step, a set of edges are broken in order to generate larger scan chain groups. The algorithm tries to divide the nodes into two groups such that the total weight of the edges between these two groups is maximized. However, as mentioned in the last paragraph, it is desirable to break the edges with smaller weights. In order to achieve this aim, the weights of

the edges are updated in reverse order; the highest weight is given to the previously lowest weighted edge and the lowest weight is accorded to the previously highest one.

The *Kernighan-Lin* minimum cut algorithm [132] is applied to divide the nodes into two groups such that the total weight of the edges between the groups is maximized. In order to achieve a maximal cut with the *Kernighan-Lin algorithm*, the weight of the edges is negated. When the *Kernighan-Lin* minimal cut algorithm is executed for the graph with negative weights, the result corresponds to a maximum cut in the original weighted graph with positive weights. The resulting graph is partitioned into groups that are mutually compatible and the corresponding test cost is calculated. The same procedure is repeated by the current weighted incompatibility graph until all edges are broken, corresponding to the single decoder case, and the addressing scheme that delivers the minimal cost is selected.

The run time of the *Kernighan-Lin algorithm* is known to be $O(n^2 log(n))$ [132]. The heuristic that is used for the identification of the decoder groups [131] is $O(n^3 log(n))$. In the worst case, only one of the nodes is separated from the remaining ones, requiring n iterations. Consequently, the total run time is $n(O(n^3 log(n) + n^2 log(n))) = n^4 log(n)$. The algorithm that is presented in this section will be referred to as the *Addressing Space Search* algorithm throughout the rest of this chaprter.

Addressing Space Search:

- Generate *weighted incompatibility graph*
- Identify decoder groups in the initial graph
    - Calculate the test cost and record as the best configuration
- While any edge exists in the incompatibility graph
    - Apply Kernighan-Lin algorithm to cut edges
    - Identify decoder groups in new graph and calculate test cost

- If test cost improves, record as the new best configuration

- Select the lowest cost decoder groups as the final addressing scheme

Although the proposed addressing scheme is constructed based on a given test set, it is incremental, supporting any possible change in the given test set after the generation of the addressing hardware. Since the modified or newly included test vectors are not considered during the addressing mechanism construction, the application cost for these vectors may not be minimal. Nonetheless, the structure of the proposed scheme enables the application of these test vectors with a small test cost. The estimated test cost for these vectors can be determined by the following analysis.

Assume that scan chains are partitioned into n groups and the size of each group is S_i. The test slice with the maximum number of conflicts in any of these groups determines the test application time. Let $k_{i,j}$ be the number of conflicting bits in test slice j of the scan chain group i and let $K_{i,max}$ be the maximum of the $k_{i,j}$'s for the scan chain group i. The maximum of the $K_{i,max}$'s equates to the rotation count.

For a specified bit density of p, the probability of conflict is $p/2$. The probability of having m conflicts in the j*th* test slice of the scan chain group i can be found by the following equation.

$$P\left(k_{i,j}=m\right)=\binom{S_i}{m}\left(\frac{p}{2}\right)^m\left(1-\frac{p}{2}\right)^{S_i-1} \tag{7.6}$$

If the length of the scan chains is L, the probability of $K_{i,max}$ being equal to m is given by Eq. (7.7).

$$P\left(K_{i,\max}=m\right)=\left(P\left(k_{i,j}\le m\right)\right)^L-\left(P\left(k_{i,j}\le m-1\right)\right)^L \tag{7.7}$$

Assuming a uniform distribution of the specified bits, the expected value of $K_{i,max}$ can be derived by the following equation.

$$E\left(K_{i,\max}\right)=\sum_{t=1}^{S_i} t \times P\left(K_{i,\max}=t\right) \tag{7.8}$$

Since the rotation count is the maximum of the $K_{i,max}$'s, the average rotation count, *RC*, for a new test vector is given by the following equation.

$$RC=\max_i\left\{E\left(K_{i,\max}\right)\right\} \tag{7.9}$$

The provided analysis assumes uniform distribution of the specified bits. However, since the physically neighboring scan cells are usually driven by the same logic, the specified bits tend to cluster on a few scan chains in practice. Consequently, the analytically obtained results shown above constitute an understatement of the average improvements to be expected in practice.

7.4 Test Application

The compaction algorithm employed in this work is a slightly modified version of the one presented in [48]. The algorithm selects a seed fault from the fault list and generates a test cube for this particular fault. The fault list is traversed and the test cubes of the faults that are compatible with the seed are merged with the seed cube. Whenever a fault that is not compatible with the seed is encountered, the fault counter is incremented. When the fault counter exceeds a predetermined limit or the fault list is exhausted, the unspecified bits of the current combined cube are filled by the corresponding response bits of the previous test vector and fault simulation is performed. The detected faults are dropped from the fault list and the algorithm continues by selecting a new seed fault from the fault list.

The test set is analyzed and the weighted conflict graph is generated as described in Section 7.3. The *Addressing Space Search* algorithm is performed by the weighted incompatibility graph and the addressing scheme that delivers the maximum reduction in the

traversed space is identified. Essentially, this particular algorithm partitions the scan chains into a number of groups and each scan chain group is addressed with a dedicated decoder.

Once the addressing scheme is identified, the test application process becomes quite straightforward. At each cycle, a conflicting bit for each scan chain group is updated by addressing these particular bit positions through the decoders. Since a dedicated decoder exists for each scan chain group, one bit for each group can be updated at each test application cycle. Consequently, the scan chain group that has the maximum number of conflicting bits at a test slice determines the number of rotations to be performed.

Example Assume that a scan configuration consisting of 8 scan chains is given and the length of the scan chains is 2. A sample test set with 3 test vectors is provided in Figure 7.5, wherein "1" denotes a conflicting bit position with the template and "-" denotes no conflict. In this example, test data volume for different addressing schemes will be compared. If a single decoder is utilized, 4 scan inputs, three pins for addressing and one pin for test data, are required for 8 scan chains. The maximum number of conflicting bits at any test slice are 3, 2 and 2 for the first three test vectors, in that order. Consequently, test vectors 1, 2 and 3 require 3, 2 and 2 rotations of the scan chains, respectively, and one of the conflicting bits is updated at each rotation. Since each test vector has two test slices and the number of scan inputs is 4, a full rotation takes 2 cycles and $4 \times 2 = 8$ bits of test data are needed. Consequently, a total of 7 rotations necessitates $7 \times 8 = 56$ bits of test data.

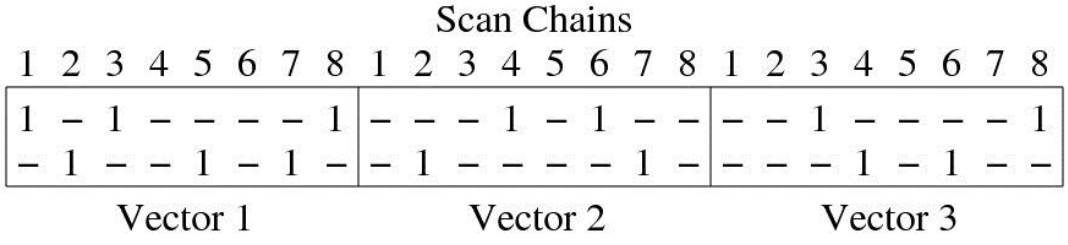


Figure 7.5 Test vectors

If only one rotation of scan chains is considered, an addressing scheme with multiple decoders that enables the update of all conflicting bits of a test slice at one test cycle is required. In order to identify this particular addressing scheme, an incompatibility graph is generated. A weighted incompatibility graph for the given test set can be seen in Figure 7.6. For instance, both scan chains 2 and 7 have conflicting bits on the second slices of the test vectors 1 and 2; consequently, an edge of weight 2 exists between the corresponding nodes of these particular scan chains in the weighted incompatibility graph. The nodes of this particular graph are partitioned into groups, wherein the nodes in each group are mutually compatible. At least 3 scan groups are required in this example to ensure resolution of all conflicts of a test slice in one cycle; {{1,2,4}, {3,5,6}, {7,8}} is one of the possible groupings that requires a minimum number of scan inputs. Three decoders, each addressing the scan chains in a particular group, are used. Since the sizes of the groups are 3, 3 and 2, a total of 6 scan pins suffices to address all scan chains as an additional address needs to be accounted for each decoder to denote the possibility of no toggle in that group, as shown in the Eq. (7.10) below.

$$S = \lceil \log_2(3+1) \rceil + \lceil \log_2(3+1) \rceil + \lceil \log_2(2+1) \rceil = 6 \qquad (7.10)$$

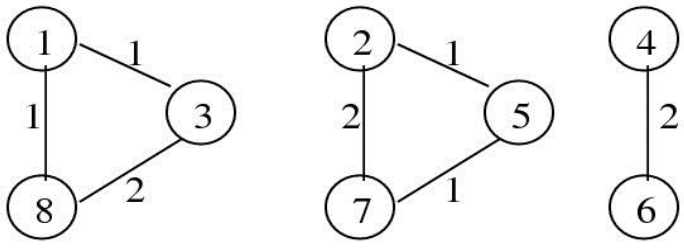


Figure 7.6 Weighted incompatibility graph

Since any two scan chains in the same addressing group can never be at conflict with the template for the same bit slice, each test vector can be applied in one rotation. For example, scan chains 1 and 2 that are part of the same group do not require a simultaneous update for any test vector. 6 scan pins and 2 test cycles for each test vector necessitate 12 bits of test data for each rotation and a total of 3 rotations to apply all test vectors results in a test data of *12 x 3 = 36* bits. This particular addressing scheme of the multiple decoders with one rotation and the single decoder addressing scheme constitute the two extremal points of the addressing space that is depicted in Figure 7.3.

In order to help break the edges that affect fewer test vectors, the weights of the edges of the incompatibility graph are adjusted, assigning the lowest weight to the highest weight edge and the highest weight to the lowest weight edge. Furthermore, in order to use the Kernighan-Lin minimal cut algorithm in the process of breaking edges with higher weights, the weights of the edges are negated, resulting in the adjusted weighted incompatibility graph in Figure 7.7(a).

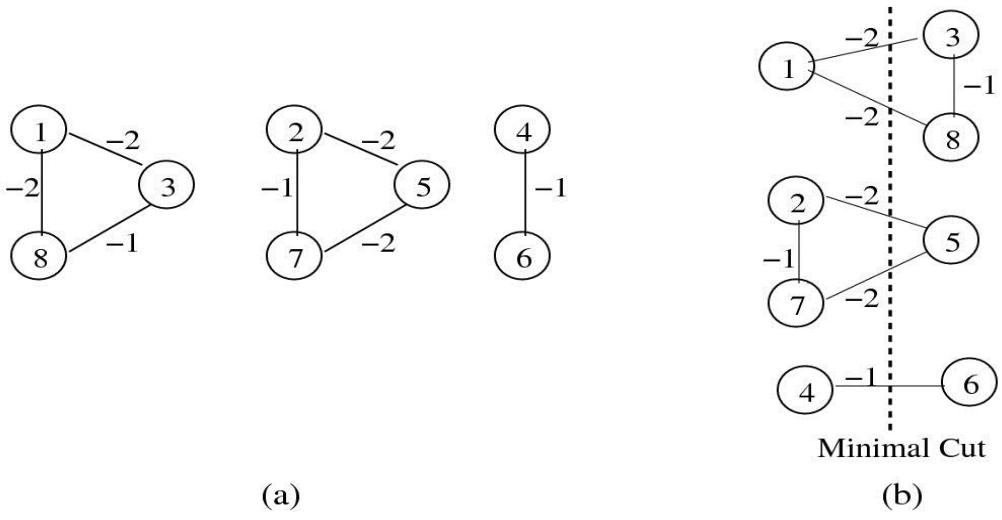


Figure 7.7 Minimal cut partitioning

If the algorithm that has been described in Section 7.3 is applied with the adjusted weighted incompatibility graph, the nodes of the graph are divided into two groups and all edges between these two node groups are broken as can be seen in Figure 7.7(b). After these particular edges are broken, only edges between nodes 3 and 8 and nodes 2 and 7 remain. So, scan chains can be partitioned into 2 groups, wherein the nodes in a group are mutually compatible. {{1,2,3,4,5}, {6,7,8}} is one of the possible partitionings of the scan chains. This particular configuration requires 2 decoders; the total number of scan inputs is 5 as can be seen in the Eq. (7.11).

$$S = \lceil \log_2(5+1) \rceil + \lceil \log_2(3+1) \rceil = 5 \qquad (7.11)$$

In this configuration, since scan chains 1 and 3 are addressed by the same decoder and the two scan chains exhibit conflicting bits on the first test slice of the test vector 1, they cannot be updated in one test cycle, necessitating 2 rotations for this particular vector. Scan chains 2 and 5 are also in the same group and have bits that are simultaneously conflicting for the second slice of the test vector 1. Nevertheless, 2 rotations of test vector 1 suffice to update these bits. Consequently, 2, 1 and 1 rotations are required to apply test vectors 1, 2 and 3, respectively. Since each rotation needs *2 x 5 = 10* bits of test data, a total of *4 x 10 = 40* bits suffices to apply the given test set. Consequently, the partitioning of the scan chains into three groups delivers the best solution for this example.

7.5 Experimental Results

The performance of the proposed method has been analyzed on the larger ISCAS89 [133] benchmark circuits. The ATALANTA test generation tool [134] and the HOPE fault simulation tool [135] have been used for the experiments.

The test set is analyzed and an incompatibility graph is generated. The *Addressing Space Search* algorithm is applied and the optimal addressing mechanism is selected. After the construction of the addressing mechanism, the test set is applied. Table 7.1 reports the obtained test cost reduction levels for the benchmark; the second column denotes the number of scan cells, the third column denotes the number of faults, the fourth column denotes the specified bit density (*p*) of the test vectors on average, the fifth column denotes the number of scan chains used in the experiments and the last column denotes the test time reduction levels with the *CircularScan* architecture in comparison to the traditional scan architecture, assuming an identical number of scan input pins.

Table 7.1 Benchmark info and test cost reduction

Circuit	FF	f	$p(\%)$	SC	$CircularScan$
s13207	669	9664	5.6	256	87.4%
s15850	597	11336	11.3	256	71.1%
s35932	1728	35110	10.3	256	95.1%
s38417	1636	31015	14.1	256	73.1%
s38584	1452	34797	7.7	256	86.8%

The experimental results in Table 7.1 show that a substantial test cost reduction, 82.7% on average, in comparison to the traditional scan architecture can be attained. Furthermore, the specified bit densities for the benchmark circuits are quite high in comparison to the specified bit densities that are observed in industrial circuits. Consequently, the results constitute strong evidence that substantially higher test cost reduction levels can be obtained for large industrial circuits with the lower specified bit densities.

Table 7.2 Comparison with test set independent methods

		TS independent					
Circuit	*CircularScan*	[42]	[44]	[34]	[40]	[41]	[51]
s13207	87.4	38.0	74.6	74.8	82.3	86.01	70.3
s15850	71.1	71.0	62.4	47.1	66.4	69.99	64.0
s35932	95.1	82.0	N/A	98.5	N/A	N/A	N/A
s38417	73.1	46.0	82.4	44.1	60.6	55.38	65.8
s38584	86.8	55.0	56.3	47.7	65.5	67.73	68.6
Avg.	82.7	58.4	68.9	62.4	68.7	69.02	67.2

Tables 7.2 and 7.3 provide a comparison to various previously published code-based, broadcast-based and linear-decompressor-based test cost reduction methodologies as reviewed in Chapter 2. The test cost reduction methodologies are divided into two groups, test set dependent, denoted as *TS dependent*, and test set independent, denoted as *TS independent*. As can be seen in Table 7.2, the proposed methodology outperforms the test set independent methodologies by quite significant margins. The examination of the test set dependent methodologies, reported in Table 7.3, shows that the proposed method delivers higher test cost reduction levels in comparison to [35], [37], [39], and slightly outperforms [48]. Nevertheless, although the test set dependent methods we compare against suffer when the test set is modified, the methodology we propose can support arbitrary possible changes in the test set. The previously published methods may require the reconfiguration of the design to be able to apply the modified test set or the modified test vectors may simply no longer be deliverable. However, the scan architecture we propose can apply any possible test vector; consequently, the modified test set can be applied with no change in the scan architecture. The substantial reduction in the test cost, accompanied by the support for the application of the modified test

set, distinguishes the proposed methodology from the previously published methods and highly increases its appeal.

Table 7.3 Comparison with test set dependent methods

		TS dependent			
Circuit	*CircularScan*	[35]	[37]	[39]	[48]
s13207	87.4	77.0	85.3	89.6	83.8
s15850	71.1	66.0	72.2	74.3	78.5
s35932	95.1	N/A	N/A	81.6	82.4
s38417	73.1	59.0	64.5	64.9	79.5
s38584	86.8	64.1	69.5	66.4	83.9
Avg.	82.7	66.5	72.9	75.4	81.7

7.5 Conclusions

Scan-based designs are widely used to reduce the high test complexity of today's large circuits by enhancing controllability and observability. Nevertheless, the traditional scan configuration significantly increases test cost.

A novel scan architecture, denoted as *CircularScan*, that enables the use of the captured response of the previously applied test vector as a template for the next pattern with the scan inputs being used as an addressing mechanism to convert the template to the subsequent pattern is proposed in this chapter. The proposed architecture enables the update of the multiple bits of the template at each test application cycle and the multiple rotations of the template are utilized in order to achieve addressing with a small set of scan input pins.

The addressing scheme of *CircularScan* is generated for a given test set; nevertheless, it is incremental, supporting possible changes to the test set after the generation of the

addressing scheme, thus supporting late test set updates. A theoretical analysis is provided to estimate the application cost of the changes to the original test set. The experimental results indicate that substantial reductions in test cost can be achieved by the proposed scheme. Furthermore, the test reduction levels obtained in the experimental results are for the benchmark circuits with higher specified bit density. This is strong indication that the proposed methodology may deliver even higher test cost reductions for large industrial circuits with the typically significantly lower specified bit densities.

Chapter 7, in part, is a reprint of the material as it appears in *B. Arslan and A. Orailoglu, "CircularScan: A Scan Architecture for Test Cost Reduction", Design, Automation and Test in Europe Conference, 2004*; in *B. Arslan and A. Orailoglu, "Design Space Exploration for Aggressive Test Cost Reduction in Circular Scan Architectures", International Conference on Computer-Aided Design, 2004*; and in *B. Arslan and A. Orailoglu, "Test Cost Reduction through A Reconfigurable Scan Architecture", International Test Conference, 2004*. The dissertation author was the primary investigator and author of these papers.

Chapter 8

Conclusions

The higher levels of integration and continuous process scaling are increasingly challenging the conventional test method of the static application of the identical test suite, generated through the use of fault models, to all chips under test. The emergence of new defect types as higher level integration and process scaling march on has resulted in the inclusion of a plethora of new fault models in test suites, at a significantly elevated cost, in the hopes of taming the defect escape level. Furthermore, the growth in the size and complexity of SOCs effectively implies a corresponding increase in the size of individual test vectors, further boosting the test cost. The economics of VLSI test on the other hand requires the delivery of an acceptable test quality at a low cost, particularly in the extremely competitive consumer marketplace.

The overlapping detection of defects by numerous fault models included in a test suite and the variable distribution of defect types however lead to a variation in the effectiveness of fault models and test vectors, subsequently resulting in a large set of ineffective test vectors in the test suite with no or insignificant defect coverage. Static derivation of the fault model and test vector effectiveness information however faces inordinate challenges in practice due to the defect characteristics shifts throughout the production life cycle as a result of the changes in manufacturing process, necessitating an adaptivity during production test to track the

fluctuations in defect characteristics. Furthermore, the integration of hundreds of domains within the chip and the increase in process variations as a result of process scaling imply the emergence of the increasingly distinct domains and individualized chip instances with variable test resource requirements, highlighting the inefficiencies in the conventional test method of the identical application of a test suite to all chips.

We address, in this thesis, the pressing necessity for an effective yet efficient test through a set of test cost and quality optimization techniques in the face of growing inefficiencies in test resource utilization. The proposed techniques not only adaptively assess the effectiveness of the test vectors and fault models but also evaluate the criticality of each chip and domain based on their distinct characteristics, subsequently enabling the optimization of test cost and quality through a selection of the most effective test vectors and a measured allocation of test resources among chips and domains. Furthermore, the ever growing cost of individual test vectors is alleviated through a novel scan compression architecture.

The proposed methods have been optimized to deliver the highest level of effectiveness and accuracy through the consideration of distinct defect classes and defect characteristic drift types while addressing the functional constraints and the algorithmic challenges. The proposed methods rely on an adaptive, post-silicon test data driven test effectiveness evaluation for the defect types that are random in nature while overcoming test data gathering constraints of test flows through dynamic test flow adjustments and a slight expansion of test data collection. The defect characteristics are accurately tracked by tailoring the methods for different drift types. The infrequent sharp drifts following the long stretches of mostly stable defect characteristics that are typically observed in mature processes are efficiently tracked through the analysis of the stability characteristics of the test effectiveness data. The defect characteristics that frequently and sharply drift particularly during product ramp-up are tracked through the utilization of a carefully selected sample of test data from the

most recent failures, addressing the necessity for both fast adaptivity and test effectiveness assessment accuracy. For the defect types such as delay defects that exhibit regularity, the proposed methods incorporate a pre-silicon analysis to boost the accuracy and efficiency of test cost and quality optimization. A pre-silicon test quality estimation model based on chip and domain characteristics is utilized to explore test cost and quality tradeoff, enabling an optimized allocation of test resources across the chips and domains while maximizing the utilization of concurrent domain testing support under architectural and test power constraints.

The first set of proposed methods that are tailored for random defect types enables an adaptive test effectiveness assessment and a subsequent identification of an optimized test set to deliver the target test quality at minimal cost through the analysis of real-time test failure information from the production test flow. An individual test vector effectiveness assessment, as presented in Chapter 3, is achieved through a dynamic ordering of the test vectors based on their effectiveness in defect detection. The test vector effectiveness ordering information is subsequently utilized to explore the test cost vs. quality tradeoff not only at a single test level but across multiple test levels, aiming at minimizing test cost yet still achieving the test quality goals. While the test effectiveness learning at the individual vector level is poised to deliver the highest level of test cost and quality optimization, it naturally requires an extensive test failure data collection process, making it more suitable for the defect characteristics that infrequently drift. In order to address the necessity for a quick test effectiveness learning process in the product ramp-up period due to the frequent defect characteristic drifts, a fault model level test effectiveness learning while retaining the order of the test vectors within the fault model is proposed as presented in Chapter 4. Test quality is represented as a continuously refreshed multi-variate function of fault coverages based on the failure data from a small set of recently tested defective chips, subsequently enabling the selection of the

optimal mixture of test vectors from various fault models while achieving quick adaptivity through the use of a small set of recent failures.

The optimization of delay test resource usage driven by the prominent impact on delay test effectiveness of the increasing chip and domain individualization constitutes the focus of the second set of proposed test methods. As a result of the individualization of chip instances due to the increasing process variation, an identical delay test set delivers a differing test quality for each individualized chip instance. A pre-silicon statistical test quality analysis is developed in Chapter 5 to capture the delay test quality variation across the process variation space, subsequently enabling an adaptive and optimized allocation of test resources to each chip based on its position in process variation space in order to extract the highest quality from the available test resources. Similarly, the increasing diversification among domains due to the integration of various cores within a chip implies variable susceptibility to delay defects among domains. An optimal test resource allocation method based on the distinct characteristics of each domain is subsequently proposed as presented in Chapter 6. The proposed method considers not only domain characteristics but also the concurrent domain testing support, simultaneously identifying the resource allocation and the schedule of the concurrently tested domains without exceeding test power limits.

Finally, a *CircularScan* test architecture is proposed in Chapter 7 to reduce the test cost in scan-based designs through the reduction of test application cycles for each test vector. The proposed architecture configures the scan chains in a circular form, enabling the use of the previously captured test response as a template for the next pattern. Scan inputs are utilized through an optimized addressing scheme to efficiently pinpoint the bits that need to be updated on the captured test response to obtain the next pattern. The use of the captured response as a template for the next pattern and the optimized addressing scheme enable the application of each test pattern quite efficiently, substantially reducing test cost.

The proposed methodologies can individually or as a combination be utilized in an industrial setting based on the user preferences. In the initial deployment of a new process node and during the product ramp-up period, the defect characteristics can show frequent, sharp changes due to the continuous update of the manufacturing process parameters in order to improve the manufacturing yield. Once the process matures, the sharp changes in defect characteristics are only infrequently observed, primarily from lot to lot only. Although either of the individual test vector level or the fault model level test effectiveness learning methods can be utilized during these phases, it is recommended that the fault model level effectiveness learning be applied during the process and product ramp-up period, benefitting from the quick adaptivity to the frequent defect characteristics shifts during this period. The individual test vector effectiveness assessment process is recommended once the process matures, reaping the benefits of fine-grained optimization for an extended period subsequent to an intensive learning process.

The delay test source allocation methods based on chip and domain characteristics can successfully coexist in an industrial setting. It is recommended that test resource allocation across domains be performed first for nominal timing values. Subsequently, the method for test resource allocation across chips can be applied to adjust each domain's previously allocated test resources based on process parameters.

The CircularScan architecture is compatible with all proposed test effectiveness assessment and resource allocation methods and it is highly recommended that *CircularScan* be implemented as a backbone of the proposed test cost and quality optimization methods. Subsequent to the test set selection based on test effectiveness learning and test resource allocation, the resulting test set can be efficiently applied to the chip under test through *CircularScan* architecture, magnifying the level of optimization achieved through the test effectiveness learning and test resource allocation methods.

The experimental results in industry and benchmark circuits evince the effectiveness of the proposed methods. The experiments for test vector effectiveness assessment and subsequent test cost optimization show that up to a 10X reduction in average test time to detect a defective device and test cost reduction levels exceeding 40% in a two-level test flow can be achieved. The defect characteristics tracking is very effective as highlighted in the experiments, wherein various parameters including the occurrence rate of the different faults and the number of faults in a defective chip are frequently changed. Test resource allocation based on chip characteristics can deliver delay test quality improvement levels reaching to 25% in experiments with no increase in test cost at all. The experiments show a substantial delay test quality improvement level, exceeding 20%, through an optimized test resource allocation across domains. *CircularScan* delivers a test cost reduction level of, on average, 82% for benchmark circuits in comparison to the traditional scan architecture. As evinced by the substantial test cost and quality optimization levels achieved by the individual application of the proposed methods, the use of a combination of the proposed methods can significantly boost the delivered test cost and quality optimization level, paving the way for an effective yet efficient test.

As the level of inefficiency in test suites continues to rise as a result of the inclusion of various new test types and the variation across chips and domains constantly grows as a result of continuous process scaling and the higher levels of integration, it can be expected that the benefits of the proposed methods are only slated to increase further. The ever increasing test cost problem can be thus significantly ameliorated while ensuring the desired test quality level, reducing test's overall contribution to final product cost and leading to an improved competitive position in the marketplace.

Bibliography

[1] S. Benner and O. Martinez, "Need for Change in Test," in *Collaborative Alliance for Semiconductor Test Workshop*, March 2011.

[2] "International Technology Roadmap for Semiconductors," 2011. [Online]. Available: http://www.itrs.net/.

[3] P. Maxwell, "The Design, Implementation and Analysis of Test Experiments," in *International Test Conference*, 2006.

[4] P. Nigh, W. Needham, K. Butler, P. Maxwel and R. Aitken, "An Experimental Study Comparing the Relative Effectiveness of Functional, Scan, IDDQ and Delay-Fault Testing," in *VLSI Test Symposium*, 1997.

[5] F.-F. Ferhani, N. R. Saxena, E. J. McCluskey and P. Nigh, "How Many Test Patterns are Useless?," in *VLSI Test Symposium*, 2008.

[6] M. Abramovici, M. A. Breuer and A. D. Friedman, Digital Systems Testing and Testable Design, Wiley-IEEE Press, 1994.

[7] K. C. Mei, "Bridging and Stuck-At Faults," *IEEE Transaction on Computers*, vol. 23, no. 7, pp. 720-727, 1974.

[8] M. Sachdev, Defect Oriented Testing for CMOS Analog and Digital Circuits, Kluwer Academic Publishers, 1998.

[9] F. Ferguson and T. Larrabee, "Test Pattern Generation for Realistic Bridge Fault in CMOS ICs," in *International Test Conference*, 1991.

[10] M. L. Bushnell and V. D. Agrawal, Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits, Springer, 2000.

[11] J. Waicukauski, E. Lindbloom, B. K. Rosen and V. Iyengar, "Transition fault simulation," *IEEE Design & Test of Computers*, vol. 4, no. 2, pp. 32-38, 1987.

[12] G. L. Smith, "Model for Delay Faults Based Upon Paths," in *International Test Conference*, 1985.

[13] Y. Sato, S. Hamada, T. Maeda, A. Takatori, Y. Nozuyama and S. Kajihara, "Invisible Delay Quality – SDQM Model Lights Up What Could Not Be Seen," in *International Test Conference*, 2005.

[14] 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," in *Asian Test Symposium*, 2006.

[15] X. Lin, M. Kassab and J. Rajski, "Test Generation for Timing-Critical Transition Faults," in *Asian Test Symposium*, 2007.

[16] C.-J. Chang and T. Kobayashi, "Test Quality Improvement with Timing-aware ATPG: Screening Small Delay Defect Case Study," in *International Test Conference*, 2008.

[17] S. Eggersgluss, M. Yilmaz and K. Chakrabarty, "Robust Timing-Aware Test Generation Using Pseudo-Boolean Optimization," in *Asian Test Symposium*, 2012.

[18] A. Uzzaman, M. Tegethoff, B. Li, K. McCauley, S. Hamad and Y. Sato, "Not all Delay Tests Are the Same - SDQL Model Shows True-Time," in *Asian Test Symposium*, 2006.

[19] S. Chakravarty, N. Devta-Prasanna, A. Gunda, J. Ma, F. Yang, H. Guo, R. Lai and D. Li, "Silicon Evaluation of Faster Than At-Speed Transition Delay Tests," in *VLSI Test Symposium*, 2012.

[20] T. Yoneda, K. Hori, I. Inoue and H. Fujiwara, "Faster-Than-At-Speed Test for Increased Test Quality and In-Field Reliability," in *International Test Conference*, 2011.

[21] I. Pomeranz and S. Reddy, "On N-detection Test Sets and Variable N-detection Test Sets for Transition Faults," in *VLSI Test Symposium*, 1999.

[22] R. D. Blanton, K. N. Dwarakanath and A. B. Shah, "Analyzing the Effectiveness of Multiple-Detect Test Sets," in *International Test Conference*, 2003.

[23] K. Y. Cho, S. Mitra and E. J. McCluskey, "Gate Exhaustive Testing," in *International Test Conference*, 2005.

[24] R. Guo, S. Mitra, E. Amyeen, J. Lee, S. Sivaraj and S. Venkataraman, "Evaluation of test metrics: stuck-at, bridge coverage estimate and gate exhaustive," in *VLSI Test Symposium*, 2006.

[25] T. M. Niermann, R. K. Roy, J. H. Patel and J. A. Abraham, "Test Compaction for Sequential Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 11, no. 2, pp. 260-267, 1992.

[26] M. S. Hsiao, E. M. Rudnick and J. H. Patel, "Fast Algorithms for Static Compaction of Sequential Circuit Test Vectors," in *VLSI Test Symposium*, 1995.

[27] I. Pomeranz and S. M. Reddy, "Procedures for Static Compaction of Test Sequences for Synchronous Sequential Circuits," *IEEE Transactions on Computers*, vol. 49, no. 6, pp. 596 - 607, 2000.

[28] P. Goel and B. C. Rosales, "Test Generation and Dynamic Compaction of Tests," in

International Test Conference, 1979.

[29] E. M. Rudnick and J. H. Patel, "Simulation-Based Techniques for Dynamic Test Sequence Compaction," in *International Conference on Computer-Aided Design*, 1996.

[30] E. M. Rudnick and J. H. Patel, "Efficient Techniques for Dynamic Test Sequence Compaction," *IEEE Transactions on Computers,* vol. 48, no. 3, pp. 323 - 330 , 1999.

[31] N. A. Touba, "Survey of Test Vector Compression Techniques," *IEEE Design and Test of Computers,* vol. 23, no. 4, p. 294–303, 2006.

[32] R. Kapur, S. Mitra and T. W. Williams, "Historical Perspective On Scan Compression," *IEEE Design and Test of Computers,* vol. 25, no. 2, pp. 114-120, 2008.

[33] A. Jas, J. Ghosh-Dastidar and N. Touba, "Scan Vector Compression/Decompression Using Statistical Coding," in *VLSI Test Symposium*, 1999.

[34] A. Chandra and K. Chakrabarty, "System-on-a-Chip Test-Data Compression and Decompression Architectures Based On Golomb Codes," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 20, no. 3, p. 355–368, 2001.

[35] A. Jas, J. Ghosh-Dastidar, M. Ng and N. A. Touba, "An Efficient Test Vector Compression Scheme Using Selective Huffman Coding," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 22, no. 6, p. 797–806, 2003.

[36] P. T. Gonciari, B. M. Al-Hashimi and N. Nicolici, "Variable-Length Input Huffman Coding for System-on-a-Chip Test," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 22, no. 6, p. 783–796, 2003.

[37] X. Kavousianos, E. Kalligeros and D. Nikolos, "Optimal Selective Huffman Coding for Test-Data Compression," *IEEE Transactions on Computers,* vol. 56, no. 8, p. 1146–1152, 2007.

[38] A. Chandra and K. Chakrabarty, "Test Data Compression and Test Resource Partitioning for System-on-a-Chip Using Frequency-Directed Run-Length (FDR) codes," *IEEE Transactions on Computers,* vol. 52, no. 8, p. 1076–1088, 2003.

[39] M. Y. Wan, Y. Ding, Y. Pan, S. Zhou and X. L. Yan, "Test Data Compression Using Extended Frequency-Directed Run Length Code Based on Compatibility," *Electronics Letters,* vol. 46, no. 6, p. 404 – 406, 2010.

[40] M. Tehranipour, M. Nourani and K. Chakrabarty, "Nine-Coded Compression Technique With Application to Reduced Pin-Count Testing and Flexible On-Chip Decompression," in *Design, Automation and Test in Europe Conference*, 2004.

[41] L.-J. Lee, W.-D. Tseng, R.-B. Lin and C.-L. Lee, "A Multi-Dimensional Pattern Run-Length Method for Test Data Compression," in *Asian Test Symposium*, 2009.

[42] I. Hamzaoglu and J. Patel, "Reducing Test Application Time for Full Scan Embedded Cores," in *International Symposium on Fault-Tolerant Computing*, 1999.

[43] A. R. Pandey and J. H. Patel, "Reconfiguration Technique for Reducing Test Time and Test Data Volume in Illinois Scan Architecture Based Designs," in *VLSI Test Symposium*, 2002.

[44] N. Oh, R. Kapur, T. W. Williams and J. Sproch, "Test Pattern Compression Using Prelude Vectors in Fan-Out Scan Chain with Feedback Architecture," in *Design, Automation and Test in Europe Conference*, 2003.

[45] N. Sitchinava, S. Samaranayake, R. Kapur, E. Gizdarski, F. Neuveux and T. W. Williams, "Changing the Scan Enable during Shift," in *VLSI Test Symposium*, 2004.

[46] K. Miyase, S. Kajihara and S. M. Reddy, "Multiple Scan Tree Design with Test Vector Modification," in *Asian Test Symposium*, 2004.

[47] I. Bayraktaroglu and A. Orailoglu, "Test Volume and Application Time Reduction Through Scan Chain Concealment," in *Design Automation Conference*, 2001.

[48] I. Bayraktaroglu and A. Orailoglu, "Concurrent Application of Compaction and Compression for Test Time and Data Volume Reduction in Scan Designs," *IEEE Transactions on Computers*, vol. 52, no. 11, p. 1480–1489, 2003.

[49] S. Mitra and K. S. Kim, "XPAND: An Efficient Test Stimulus Compression Technique," *IEEE Transactions Computers*, vol. 55, no. 2, p. 163–173, 2006.

[50] B. Koenemann, "LFSR-Coded Test Patterns for Scan Designs," in *European Test Symposium*, 1991.

[51] A. Jas, B. Pouya and N. Touba, "Virtual Scan Chains: A Means for Reducing Scan Length in Cores," in *VLSI Test Symposium*, 2000.

[52] J. Rajski, J. Tyszer, M. Kassab and N. Mukherjee, "Embedded Deterministic Test," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 5, p. 776–792, 2004.

[53] D. H. Baik, K. Saluja and S. Kajihara, "Random Access Scan: A Solution to Test Power, Test Data Volume and Test Time," in *International Conference on VLSI Design*, 2004.

[54] A. S. Mudlapur, V. D. Agrawal and A. D. Singh, "A Random Access Scans Architecture to Reduce Hardware Overhead," in *International Test Conference*, 2005.

[55] D. H. Baik and K. K. Saluja, "Progressive Random Access Scan: A Simultaneous Solution to Test Power, Test Data Volume and Test Time," in *International Test Conference*, 2005.

[56] R. Adiga, G. Arpit, V. Singh, K. K. Saluja, H. Fujiwara and A. D. Singh, "On Minimization of Test Application Time for RAS," in *International Conference on VLSI Design*, 2010.

[57] J. Geuzebroek, E. Marinissen, A. Majhi, A. Glowatz and F. Hapke, "Embedded Multi-Detect ATPG and Its Effect on the Detection of Unmodeled Defects," in *International Test Conference*, 2007.

[58] S. Eichenberger, J. Geuzebroek, C. Hora, B. Kruseman and A. Majhi, "Towards a World Without Test Escapes: The Use of Volume Diagnosis to Improve Test Quality," in *International Test Conference*, 2008.

[59] F. Hapke, R. Krenz-Baath, A. Glowatz, J. Schloeffel, H. Hashempour, S. Eichenberger, C. Hora and D. Adolfsson, "Defect-Oriented Cell-Aware ATPG and Fault Simulation for Industrial Cell Libraries and Designs," in *International Test Conference*, 2009.

[60] F. Hapke, W. Redemund, J. Schloeffel, R. Krenz-Baath, A. Glowatz, M. Wittke, H. Hashempour and S. Eichenberger, "Defect-Oriented Cell-Internal Testing," in *International Test Conference*, 2010.

[61] F. Hapke, W. Redemund, J. Schloeffel, A. Glowatz, J. Rajski, M. Reese, J. Rearick and J. Rivers, "Cell-aware Analysis for Small-delay Effects and Product ion Test Results from Different Fault Models," in *International Test Conference*, 2011.

[62] F. Hapke, J. Schloeffel, S. Eichenberger and H. Hashempour, "Gate-Exhaustive and Cell-Aware Pattern Sets for Industrial Designs," in *International Symposium on VLSI Design, Automation and Test*, 2011.

[63] F. Hapke, M. Reese, J. Rivers, A. Over, V. Ravikumar, W. Redemund, A. Glowatz, J. Schloeffel and J. Rajski, "Cell-aware Production Test Results from a 32-nm Notebook Processor," in *International Test Conference*, 2012.

[64] F. Hapke and J. Schloeffel, "Introduction to The Defect-Oriented Cell-Aware Test Methodology for Significant Reduction of DPPM Rates," in *European Test Symposium*, 2012.

[65] M. Yilmaz, K. Chakrabarty and M. Tehranipoor, "Interconnect-Aware and Layout-Oriented Test-Pattern Selection for Small-Delay Defects," in *International Test Conference*, 2008.

[66] M. Yimaz, K. Chakrabarty and M. Tehranipoor, "Test-Pattern Selection for Screening Small-Delay Defects in Very-Deep Submicrometer Integrated Circuits," *IEEE*

Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 29, no. 5, pp. 760 - 773 , 2010.

[67] K. Peng, M. Yilmaz, M. Tehranipoor and K. Chakrabarty, "High-quality Pattern Selection for Screening Small-Delay Defects Considering Process Variations and Crosstalk," in *Design, Automation and Test in Europe Conference*, 2010.

[68] K. Peng, M. Yilmaz, K. Chakrabarty and M. Tehranipoor, "Crosstalk- and Process Variations-Aware High-Quality Tests for Small-Delay Defects," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 21, no. 6, pp. 1129 - 1142, 2013.

[69] J. Xiong, V. Zolotov and C. Visweswariah, "Pre-ATPG Path Selection for Near Optimal Post-ATPG Process Space Coverage," in *International Conference on Computer-Aided Design*, 2009.

[70] L. C. Wang, J. J. Liou and K.-T. Cheng, "Critical Path Selection for Delay Fault Testing Based Upon a Statistical Timing Model," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 11, pp. 1550-1565, 2004.

[71] W. Jian and B. Vinnakota, "Defect-Oriented Test Scheduling," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 9, no. 3, pp. 427-438, 2001.

[72] K. M. Butler and J. Saxena, "An Empirical Study on the Effects of Test Type Ordering on Overall Test Efficiency," in *International Test Conference*, 2000.

[73] 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," in *International Test Conference*, 2004.

[74] S. Benner and O. Boroffice, "Optimal Production Test Times through Adaptive Test Programming," in *International Test Conference*, 2001.

[75] E. Yilmaz, S. Ozev and K. Butler, "Adaptive Test Flow for Mixed-Signal/RF Circuits Using Learned Information From Device Under Test," in *International Test Conference*, 2010.

[76] E. Yilmaz and S. Ozev, "Adaptive Test Elimination for Analog/RF Circuits," in *Design Automation Conference*, 2009.

[77] M. Chen and A. Orailoglu, "Test Cost Minimization through Adaptive Test Development," in *International Conference on Computer Design*, 2008.

[78] H.-G. Stratigopoulos and S. Mir, "Adaptive Alternate Analog Test," *IEEE Design and Test of Computers*, vol. 29, no. 4, pp. 71-79, 2012.

[79] E. Yilmaz and S. Ozev, "Adaptive Multi-Site Testing for Analog/Mixed-Signal Circuits

Incorporating Neighborhood Information," in *European Test Symposium*, 2012.

[80] E. Yilmaz, S. Ozev and K. Butler, "Per-Device Adaptive Test for Analog/RF Circuits Using Entropy-Based Process Monitoring," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 21, no. 6, pp. 1116-1128 , 2013.

[81] X. Yu, Y.-T. Lin, W.-C. Tam, O. Poku and R. Blanton, "Controlling DPPM through Volume Diagnosis," in *VLSI Test Symposium*, 2009.

[82] T. Uezono, T. Takahashi, M. Shitani, K. Hatayama, K. Masu, H. Och and T. Sato, "Path Clustering for Adaptive Test," in *VLSI Test Symposium*, 2010.

[83] J. Xiong, V. Zolotov, C. Visweswariah and P. A. Habitz, "Optimal Test Margin Computation for At-Speed Structural Test," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 9, pp. 1414-1423, 2009.

[84] W. R. Daasch and R. Madge, "Variance Reduction and Outliers: Statistical Analysis of Semiconductor Test Data," in *International Test Conference*, 2005.

[85] W. R. Daasch, J. McNames, R. Madge and K. Cota, "Neighborhood Selection for IDDQ Outlier Screening at Wafer Sort," *IEEE Design and Test of Computers*, vol. 19, no. 5, pp. 74-81, 2002.

[86] R. Madge, M. Rehani, K. Cota and W. Daasch, "Statistical Post-Processing at Wafersort - An Alternative to Burn-In and a Manufacturable Solution to Test Limit Setting for Sub-Micron Technologies," in *VLSI Test Symposium*, 2002.

[87] S. Sabade and H. Walker, "Evaluation of Statistical Outlier Rejection Methods for IDDQ Limit Setting," in *Asia and South Pacific Design Automation Conference*, 2002, 755-760 .

[88] R. Turakhia, B. Benware, R. Madge, T. Shannon and R. Daasch, "Defect Screening Using Independent Component Analysis on IDDQ," in *2005*, VLSI Test Symposium.

[89] S. Sabade and D. Walker, "IC Outlier Identification Using Multiple Test Metrics," *IEEE Design and Test of Computers*, vol. 22, no. 6, pp. 586-595, 2005.

[90] L. Fang, M. Lemnawar and Y. Xing, "Cost Effective Outliers Screening with Moving Limits and Correlation Testing for Analogue ICs," in *International Test Conference*, 2006.

[91] H.-G. Stratigopoulos and Y. Makris, "Nonlinear Decision Boundaries for Testing Analog Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 11, pp. 1760 - 1773, 2005.

[92] H.-G. Stratigopoulos and Y. Makris, "Error Moderation in Low-Cost Machine-

Learning-Based Analog/RF Testing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 27, no. 2, pp. 339-351, 2008.

[93] H.-G. Stratigopoulos, S. Mir and Y. Makris, "Enrichment of Limited Training Sets in Machine-Learning-Based Analog/RF Test," in *Design, Automation and Test in Europe Conference,* 2009.

[94] E. Yilmaz, S. Ozev and K. Butler, "Adaptive Multidimensional Outlier Analysis for Analog and Mixed Signal Circuit," in *International Test Conference*, 2011.

[95] B. Lee, L.-C. Wang and M. Abadir, "Issues on Test Optimization with Known Good Dies and Known Defective Dies - A Statistical Perspective," in *International Test Conference,* 2006.

[96] S. H. Wu, B. N. Lee, L.-C. Wang and M. S. Abadir, "Statistical Analysis and Optimization of Parametric Delay Test," in *International Test Conference*, 2007.

[97] S. Wu, D. Drmanac and L.-C. Wang, "A Study of Outlier Analysis Techniques for Delay Testing," in *International Test Conference*, 2008.

[98] D. Drmanac, B. Bolin, L.-C. Wang and M. S. Abadir, "Minimizing Outlier Delay Test Cost in the Presence of Systematic Variability," in *International Test Conference*, 2009.

[99] J. L. Myers and A. D. Well, Research Design and Statistical Analysis (2nd edition), Lawrence Erlbaum Associates, 2003.

[100] R. S. Sutton and A. Barto, Reinforcement Learning: An Introduction, The MIT Press, 1998.

[101] P. M. O'Neill, "Statistical test: A New Paradigm to Improve Test Effectiveness & Efficiency," in *International Test Conference*, 2007.

[102] V. D. Agrawal, S. C. Seth and P. Agrawal, "Fault Coverage Requirements in Production Testing of LSI Circuits," *IEEE Journal of Solid-State Circuits,* vol. 17, no. 1, pp. 57-61, 1982.

[103] S. C. Seth and V. D. Agrawal, "Characterizing the LSI Yield Equation from Wafer Test Data," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 3, no. 2, pp. 123-126, 1984.

[104] T. W. Williams and N. C. Brown, "Defect Level as a Function of Fault Coverage," *IEEE Transactions on Computers,* Vols. C-30, no. 12, pp. 987-988, 1981.

[105] E. J. McCluskey and F. Buelow, "IC Quality and Test Transparency," *IEEE Transactions on Industrial Electronics* , vol. 36, no. 2, pp. 197-202, 1989.

[106] J. T. de Sousa and V. D. Agrawal, "Reducing Complexity of Defect Level Modeling Using the Clustering Effects," in *Design, Automation and Test in Europe Conference*, 2000.

[107] J. Savir, "AC Product Defect Level and Yield Loss," *IEEE Transactions on Semiconductor Manufacturing*, vol. 3, no. 4, pp. 195-205, 1990.

[108] D. Das, S. Seth and V. D. Agrawal, "Accurate Computation of Field Reject Ratio Based on Fault Latency," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 1, no. 4, pp. 537-545, 1993.

[109] C. H. Stapper, F. M. Armstron and K. Saji, "Integrated Circuit Yield Statistics," *Proceedings of the IEEE*, vol. 71, no. 4, pp. 453-470, 1983.

[110] W. Maly, A. J. Strojwas and S. W. Director, "VLSI Yield Prediction and Estimation: A Unified Framework," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 5, no. 1, pp. 114-130, 1986.

[111] N. Kumar, K. Kennedy, K. Gildersleeve, R. Abelson, C. M. Mastrangelo and D. C. Montgomery, "A Review of Yield Modeling Techniques for Semiconductor Manufacturing," *International Journal of Production Research*, vol. 44, no. 23, p. 5019–5036, 2006.

[112] K. M. Butler, J. M. Carulli and J. Saxena, "Modeling Test Escape Rates As a Function of Multiple Coverages," in *International Test Conference*, 2008.

[113] S. Boyd and L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.

[114] Q. Liu and S. Sapatnekar, "A Framework for Scalable Postsilicon Statistical Delay Prediction Under Process Variations," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 8, pp. 1201-1212, 2009.

[115] X. Wang, M. Tehranipoor and R. Datta, "A Novel Architecture for On-Chip Path Delay Measurement," in *International Test Conference*, 2009.

[116] P. Nigh and A. Gattiker, "Test Method Evaluation Experiments & Data," in *International Test Conference*, 2000.

[117] Y. Sato, S. Hamada, T. Maeda, A. Takatori and S. Kajihara, "Evaluation of the Statistical Delay Quality Model," in *Asia and South Pacific Design Automation Conference*, 2005.

[118] S. Mitra, E. Volkerink, E. J. McCluskey and S. Eichenberger, "Delay Defect Screening Using Process Monitor Structures," in *VLSI Test Symposium*, 2004.

[119] C. Visweswariah, K. Ravindran, K. Kalafala, S. G. Walker and S. Narayan, "First-order Incremental Block-based Statistical Timing Analysis," in *Design Automation Conference*, 2004.

[120] H. Chang and S. S. Sapatnekar, "Statistical Timing Analysis Under Spatial Correlations," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 9, pp. 1467-1482, 2005.

[121] H. Chang, V. Zolotov, C. Visweswariah and S. Narayan, "Parameterized Block-Based Statistical Timing Analysis with Non-Gaussian and Nonlinear Parameters," in *Design Automation Conference*, 2005.

[122] L. Cheng, J. Xiong and L. He, "Non-Gaussian Statistical Timing Analysis Using Second-Order Polynomial Fitting," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 1, pp. 130-140, 2009.

[123] S. Sapatnekar, Timing, Kluwer Academic Publishers, 2004.

[124] Y. Zorian, E. J. Marinissen and S. Dey, "Testing Embedded Core-based System Chips," in *International Test Conference*, 1998.

[125] E. Larsson and H. Fujiwara, "System-on-Chip Test Scheduling With Reconfigurable Core Wrappers," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 14, no. 3, pp. 305-309, 2006.

[126] C. Yao, K. K. Saluja and P. Ramanathan, "Power and Thermal Constrained Test Scheduling Under Deep Submicron Technologies," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 2, pp. 317-322, 2011.

[127] X. Wen, K. Miyase, S. Kajihara, T. Suzuki, Y. Yamato, P. Girard, Y. Ohsumi and L.-T. Wang, "A Novel Scheme to Reduce Power Supply Noise for High-Quality At-Speed Scan Testing," in *International Test Conference*, 2007.

[128] J. Rearick, "Too Much Delay Fault Coverage is a Bad Thing," in *International Test Conference*, 2001.

[129] A. Lodi, S. Martello and M. Monaci, "Two-Dimensional Packing Problems: A Survey," *European Journal of Operational Research*, vol. 141, no. 2, pp. 241-252, 2002.

[130] T. Hiraide, K. O. Boateng, H. Konishi, K. Itaya, M. Emori and H. Yamanaka, "BIST-Aided Scan Test - A New Method for Test Cost Reduction," in *VLSI Test Symposium*, 2003.

[131] I. Park, S. Hong and C. Kyung, "Two Complementary Approaches for Microcode Bit Optimization," *IEEE Transactions on Computers*, vol. 43, no. 2, p. 234–239, 1994.

[132] B. W. Kernighan and S. Lin, "An Efficient Heuristic Procedure to Partition Graphs," *Bell System Technical Journal*, vol. 49, no. 2, p. 291–307, 1970.

[133] F. Brglez, D. Bryan and K. Kozminski, "Combinational Profiles of Sequential Benchmark Circuits," in *International Symposium on Circuits and Systems*, 1989.

[134] H. K. Lee and D. S. Ha, "On the Generation of Test Patterns for Combinational Circuits," Technical Report 12-93, Department of Electrical Engineering, Virginia Polytechnic Institute and State University, 1993.

[135] H. K. Lee and D. S. Ha, "An Efficient Parallel Fault Simulator," in *Design Automation Conference*, 1992.