In-Field Structural Methods for End-to-End Automotive Digital Diagnosis

Von der Fakultät für Informatik, Elektrotechnik und Informationstechnik der Universität Stuttgart zur Erlangung der Würde eines Doktors der Naturwissenschaften (Dr. rer. nat.) genehmigte Abhandlung

Vorgelegt von

Alejandro Cook

aus Barquisimeto / Venezuela

Hauptberichter:	Prof. Dr. rer. nat. H.-J. Wunderlich
Mitberichter:	Prof. Dr.-Ing. Jürgen Teich
Tag der mündlichen Prüfung:	16.07.2014

Institut fuȑ Technische Informatik der Universittät Stuttgart

2014

to my wife Luisa

to my parents

Acknowledgments

Many people have contributed in several ways to the successful conclusion of this work. I am glad to have the opportunity to thank those without whose support and encouragement this dissertation would not have been possible.

I thank Ulrich Abelein for his continuous support throughout project DIANA. I thank Prof. Hans-Joachim Wunderlich for the opportunity to work at his department, and for all the constructive feedback he provided me during this dissertation. My appreciation also goes to Rafał Baranowski, to my dad, and to my brother for proofreading this manuscript.

I am thankful to my colleagues at the University of Stuttgart with whom I had the pleasure to work for the last 5 years. A big thanks goes to the DIANA team: Dominik Ull, Laura Rodríguez Gómez, Melanie Elm and Stefan Holst. I am also very grateful to Rafał Baranowski, Claus Braun, Atefe Dalirsani, Nadereh Hatami, Michael Imhof, Anusha Kakarala, Michael Kochte, Chang Liu, Abdullah Mumtaz, Eric Schneider, and Alexander Schöll, for many insightful discussions. I owe a debt of gratitude to Mirjam Breitling, Helmut Häfner, and Lothar Hellmeier for their outstanding organizational and technical expertise.

To my wife Luisa Aristizábal I am more grateful than these lines could describe. Her love, patience, and encouragement have been the source of my energy during our adventures in Germany. I thank my parents for their unconditional support in all my endeavors, no matter how big or how small. I also thank my brother, who, as the perfectionist that he is, triggered my overachieving instincts at an early age.

Last but not least, I thank my dear friends Rubén Cervantes, Diana García and Cary Brown for all the laughs we once shared in Stuttgart.

Stuttgart, June 2014

Alejandro Cook

Summary

The automotive domain has strongly relied on recent advances in semiconductor technology in order to offer customers a huge amount of appealing features of overwhelming complexity. As traditional functional tests are no longer sufficient to fulfill automotive diagnostic requirements, the analysis of automotive semiconductor failures has become a major quality concern. Semiconductor structural test solutions are already key technologies for the successful manufacturing of any integrated circuit. However, these techniques place stringent constraints on the test application process, which cannot be easily enforced outside the manufacturing environment to achieve a suitable automotive diagnostic solution.

This dissertation captures the requirements of a suitable automotive structural diagnostic approach for random digital circuits and provides cost-efficient solutions for the corresponding technical challenges. For this purpose, state-of-the-art methodologies for manufacturing test are enhanced in order to support the failure analysis process in the automotive domain throughout the complete system life-cycle.

The first contribution of this work is a conceptual architecture for the reuse of available on-chip test infrastructure necessary to conduct autonomous structural tests at system level. The presented methodology makes use of any available system resources in order to gain access to the same on-chip *design-for-test* structures originally devised to improve the effectiveness and efficiency of manufacturing tests.

The second contribution is a logic diagnostic procedure that analyzes highly compacted structural test responses and yet achieves excellent diagnostic accuracy. The presented algorithm is able to diagnose single and multiple arbitrary faults without the need for exhaustive simulation. Experimental results show the developed technique outperforms diagnostic solutions in the literature both in terms of accuracy and test response compaction ratio.

Finally, a cost-efficient on-chip diagnostic architecture is presented, which enables the autonomous collection of compacted test responses during built-in self-test (BIST). This architecture is fully compatible with traditional BIST procedures typically used for manufacturing test. Therefore, the proposed architecture may be used in combination with any technique for cost-effective autonomous on-chip test pattern generation. Thus, it efficiently provides accurate diagnostic capabilities while retaining the same

defect coverage in the generated pattern sequence.

The methods and algorithms in this dissertation enable the introduction of structural test and diagnostic solutions into the failure analysis process of the automotive industry. Experimental results show the performance of the presented methodology is comparable to that of state-of-the-art solutions employed for semiconductor manufacturing.

Zusammenfassung

Die Fahrzeugtechnik nutzt die aktuellen Entwicklungen der Halbleiterindustrie, um Kunden eine enorme Anzahl ansprechender, komplexer Produkteigenschaften zu bieten. Da traditionelle, funktionale Testverfahren nicht mehr ausreichen, um den diagnostischen Anforderungen im Automobilbereich gerecht zu werden, wird die Analyse von Halbleiterdefekten zu einem wichtigen Anspekt der Qualitätssicherung. Strukturelle Testverfahren stellen bereits eine Schlüsseltechnologie für die erfolgreiche Herstellung integrierter Schaltungen dar. Da dieses Vorgehen jedoch strenge Vorgaben bezüglich der Testanwendung mit sich bringt, kann es nicht problemlos außerhalb der Chipherstellung zum Erreichen einer angemessenen diagnostischen Lösung im Automobilbereich eingesetzt werden.

Diese Dissertation beschreibt die Anforderungen eines geeigneten strukturellen Ansatzes für die Diagnose jeglicher digitaler Schaltungen, und bietet kostengünstige Lösungen für dazugehörige technischen Herausforderungen. Zu diesem Zweck werden herkömmliche Testmethoden aus dem Bereich der Herstellung derart angepasst, dass sie die Fehleranalyse über den gesamten Lebenszyklus des Systems unterstützen.

Der erste Beitrag dieser Arbeit ist ein Ansatz, der die verfügbaren Hardwareressourcen im System wiederverwendet, um die autonome Durchführung struktureller Tests zu ermöglichen. Die vorgestellte Testarchitektur greift hierzu auf existierende Test- und Diagnoseinfrastruktur des Chips zu, welche ursprünglich zur Verbesserung des Herstellungstests konzipiert wurde.

Der zweite Beitrag ist ein Verfahren zur Logik-Diagnose, welches trotz stark kompaktierter Testantworten dennoch eine hervorragende diagnostische Auflösung erreicht. Das vorgestellte Algorithmus ist in der Lage, sowohl Einzel- als auch Mehrfachfehler ohne vollständige Fehlersimulation zu diagnostizieren. Experimentelle Ergebnisse zeigen, dass die entwickelte Methode frühere Diagnosemethoden aus der Literatur bezüglich Genauigkeit und Kompaktierung von Testantworten übertrifft.

Schließlich wird eine kostengünstige Diagnosearchitektur vorgestellt, welche das autonome Sammeln kompaktierter Testantworten bei der Durchführung eingebauter Selbsttests (Built-in Self-Test, BIST) ermöglicht. Diese Architektur ist vollständig kompatibel zu traditionellen Selbstestverfahren, wie sie typischerweise beim Herstellungstest eingesetzt werden. Der vorgeschlagene Ansatz kann mit jeder Methode zur kosten-

effektiven, autonomen On-Chip-Erzeugung von Testmustern kombiniert werden. So ermöglicht diese Methode hochgenaue diagnostische Fähigkeiten, während die Defektabdeckung der erzeugten Testmustersequenz erhalten bleibt.

Die Methoden und Algorithmen in dieser Dissertation ermöglichen die Einführung struktureller Test- und Diagnoseverfahren bei der Fehleranalyse im Bereich der Fahrzeugtechnik. Experimentelle Ergebnisse belegen, dass die Leistungsfähigkeit der vorgestellten Methoden mit in der Chipherstellung eingesetzten Verfahren vergleichbar ist.

Contents

1. Introduction

Automotive systems nowadays feature increasingly complex electric and electronic components. This trend is in strong contrast with the traditional perspective, where automotive vehicles are categorized, for the most part, as mechanical systems [Furst10]. Such a dramatic shift has driven the role of semiconductors into the spotlight of the automotive industry. In fact, most of the industry's innovations in the last decade have been enabled, either directly or indirectly, by the use of semiconductors [Abelein12]. Today a premium vehicle contains over 100 distributed electronic control units (ECUs) [Kim11] and, as semiconductor technology provides higher integration levels, even today's most demanding features will become common practice in later vehicle generations.

As semiconductors play a central role in the automotive industry, especially for the processing of digital information, very tight quality standards are usually enforced during chip manufacturing. However, the integrity of an automotive system can still be affected by subtle issues, which are not detected during manufacturing tests (test escapes), latent hardware faults which become active after system assembly, harsh environmental conditions, or degradation [Chen13, Sun13].

The automotive industry traditionally relies on *functional tests* to detect any hardware issue in the field. This kind of tests verifies the integrity of the system by comparing the system's behavior to its specification. This is achieved, for example, by means of concurrent test routines and power-up tests that evaluate functional system properties, like plausibility checks or field bus interconnection checks. Unfortunately, these techniques may not suffice to account for complex semiconductor failure behavior, since they do not fully cover the complete system functionality and only verify a few corner-cases in the specification.

For instance, in a distributed automotive subsystem, consisting of several interacting ECUs, functional tests may help to disclose many system malfunctions. However, their structural fault coverage is hard to measure, and usually amounts roughly to 50% [Maxwell00]. As a consequence, automotive semiconductor failure analysis is today a very elaborate and time-consuming process.

Structural tests identify faulty components in a device according to its internal building blocks and their interconnection. For example, they may check for unintended connections or *bridges* between neighboring signals in the circuit layout. In contrast to functional tests, which can only provide a pass/fail test outcome, structural tests offer the opportunity to distinguish fault locations. These diagnostic capabilities can complement traditional functional testing in order to meet the high quality demands of the automotive domain.

This dissertation presents structural techniques for the diagnosis of complex embedded systems, like those typically found in the automotive industry. These methods enable the application of structural tests in an assembled system and analyze the obtained failure response data to locate faulty semiconductor components. The presented methodology assists semiconductor failure analysis activities for continuous improvement of system quality, which, for the automotive domain, range from workshop tests, where the lowest replaceable unit is identified, to detailed semiconductor analysis, where diagnostic results direct physical failure analysis (PFA) to the most likely locations in the chip responsible for the observed erroneous behavior.

This chapter will first describe the challenges of a suitable structural diagnostic strategy. Then, semiconductor failures are analyzed in the context of automotive systems. After this, the benefits of structural test in the automotive failure analysis flow are detailed. In the following section, typical application scenarios, so-called diagnostic use-cases, are presented, which reflect elicited industrial requirements. Finally, this chapter provides a short overview of the rest of this dissertation.

1.1. Challenges in Automotive Semiconductor Test and Diagnosis

The goal of structural diagnostic approach for semiconductors is to enable the adoption of corrective measures that improve overall system quality and reduce service and repair costs in case of semiconductor failures. For this purpose structural test and diagnosis are required throughout the complete system life-cycle. Structural tests capture relevant information about the hardware integrity of a device, while structural diagnosis carries out the corresponding analysis in order to identify one or more faulty hardware components. Although semiconductor test and diagnosis are already mature, indispensable disciplines for the successful realization of any semiconductor device, available structural test and diagnostic methods need to be extended in order to account for the following domain properties.

Firstly, the capabilities of a structural test and diagnostic solution strongly depend on the resources available to collect structural information from a system component under test. Thus, the semiconductor test application process needs to support the characteristics of the testing environment, which defines the availability of test equipment, its usability, and the amount of time that can be devoted to test.

Secondly, diagnostic results need to support a wide range of diagnostic objectives. In this regard, the identification of a faulty component at the right granularity level is one key aspect to enable efficient corrective and preventive quality measures. For example, in some situations it may be sufficient to identify a faulty board, while in others it may be required to distinguish faulty chip locations in an integrated circuit (IC).

The resulting requirements can only be fulfilled if systems are equipped with structural self-test and diagnostic capabilities. Systems-on-chip (SOCs) commonly used today integrate lots of different specialized cores containing logic, memory, analog, and mixed-signal resources. This dissertation focuses on an efficient test/diagnostic solution for logic cores. In comparison to memory cores, logic cores do not have a regular structure, which makes their test and diagnostic procedures much more complex.

The main technical challenges for a suitable solution for structural diagnosis are: (1) the efficient access to on-chip test infrastructure, (2) the diagnosis of logic cores with highly compacted test responses and (3) the design of a cost-effective on-chip architecture for built-in diagnosis.

- *Access to on-chip test infrastructure*
  ICs are equipped with design-for-test (DfT) infrastructure to improve the effectiveness and efficiency of manufacturing tests. These internal structures are driven and controlled by expensive automated test equipment (ATE) especially designed for the semiconductor industry. In order to gain access to DfT infrastructure for system diagnosis, efficient approaches are necessary to make use of

any available system functionality to manage the activation and application of structural tests.

- *Diagnosis of logic cores with highly compacted test responses*
  The full diagnostic potential of structural diagnosis cannot be fully exploited with a simple pass/fail test criterion. For this reason, *logic diagnosis* is performed to analyze the collected response data of a faulty IC and locate and characterize the source of the erroneous behavior in the chip structure. Generally, the test response data gathered from the ATE suffice to accurately characterize arbitrary defect behavior [Holst12]. Unfortunately, the total amount of information in the response data of a single IC may surpass the available storage capacity of the entire system. Consequently, the compaction of test response data becomes mandatory and, therefore, logic diagnosis algorithms need to be extended accordingly so that they still provide enough insight into the problems affecting the manufacturing process.
- *Design of a cost-efficient on-chip architecture for built-in diagnosis*
  The efficient on-chip generation of diagnostic data calls for a specialized test architecture to (a) generate suitable diagnostic stimuli and (b) compact and store obtained test responses. The hardware resources required for both test and diagnosis must be carefully optimized so that they make the least impact on the overall system cost.

Finally, safety and security play an important role for automotive systems. Safety requirements must be fulfilled in order to minimize the risk of passenger injury, while security guidelines need to be in place to prevent the tampering with the ECU, and protect sensitive information belonging to the companies involved in the failure analysis process. Safety and security considerations are out of the scope of this dissertation

1.2. Semiconductor Failures in the Automotive Domain

The introduction of semiconductors in passenger vehicles started already more than two decades ago. As Figure 1.1 shows, in those early days only mature semiconductor technologies found their way into automotive systems. As market trends demand a staggering number of features in current vehicle generations, today's automotive systems make use of the latest manufacturing technology, which, as described in the

International Technology Roadmap for Semiconductors (ITRS) [ITR11], is more likely to exhibit operational malfunctions.

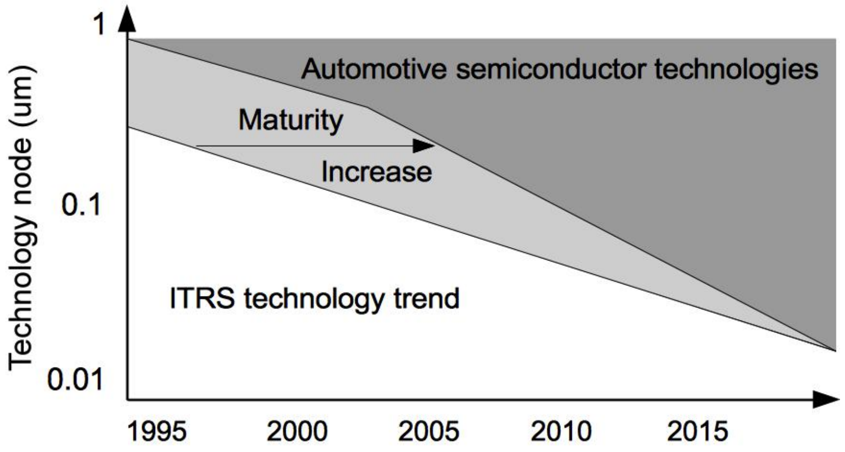


Figure 1.1.: Automotive semiconductor technology trend [Abelein13]

Malfunctions in ICs are due to physical imperfections or *defects* in the semiconductor fabrication process. Semiconductor test and diagnosis identify, locate and characterize the root cause of failures in manufactured ICs in order to help analyze the production process and adopt suitable measures to improve quality and reduce costs. This process is called *yield learning* and is performed throughout the complete chip manufacturing cycle, i.e. from first silicon to volume production. Consequently, semiconductor test and diagnosis play a key role for the successful realization of any IC and amount to the largest part of total manufacturing costs.

The automotive market segment must, therefore, account for two conflicting goals: On the one hand, stringent quality requirements demand that almost no faulty ICs be shipped and integrated into a system. On the other hand, as the automotive industry is very sensitive to costs, the price of a chip must be kept sufficiently low. A suitable compromise in this situation is very hard to achieve and, as a result, some marginal integrated circuits, which are prone to malfunctions in the field, may be inadvertently shipped to customers. In the next subsections, the most common causes of such semiconductor failures are summarized.

1.2.1. Hazard Rate

The hazard rate measures the rate at which a product is likely to fail over time. This rate has already been extensively studied in the context of reliability engineering [Yang07]. Figure 1.2 shows the typical course of failures in time (FIT) a product line suffers during its complete lifetime. The figure shows three distinct regions: For $t < t_1$ a relatively large number of early failures are expected and their failure rate monotonically decreases. This is mostly caused by marginal devices, which cannot stand operationl stress. During $t_1 < t < t_2$ the failure rate remains more or less constant, while for $t > t_2$ the product suffers from wearout mechanisms and failure rate grows steadily.

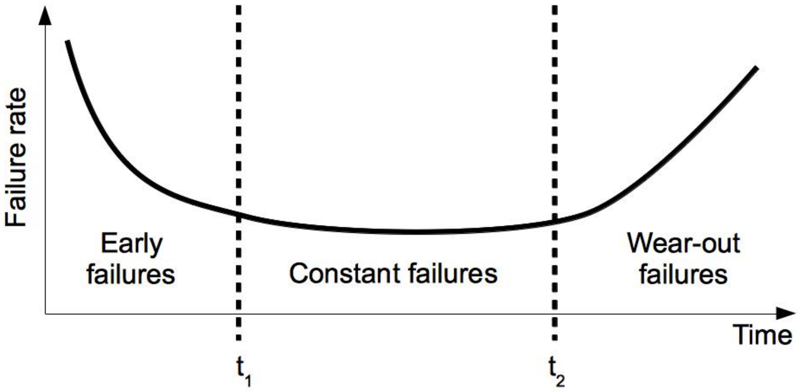


Figure 1.2.: The “bathtub” curve function [Yang07]

In the case of ICs, the failure rate can be attributed either to *random* or *systematic* defects. Random defects are the result of spurious effects during manufacturing, which cannot be precisely modeled or quantified beforehand. In contrast, systematic defects can be influenced by tuning the fabrication process or by making changes in the device design. In this case, product quality [Williams81] can be improved by adjusting the manufacturing test procedure according to the observed failure mechanism. Therefore, a suitable end-to-end semiconductor test and diagnostic strategy spanning the whole system’s lifecycle can provide valuable insight to optimize chip production.

Early life failures are usually due either to test escapes or latent defects. On the one hand, test escapes are defects for which no suitable manufacturing test was applied and no erroneous response was observed. On the other hand, latent defects only become active after some service time. Traditionally, these defects are identified in *burn-in* tests [McPherson06], where the chip is exercised under stress, like elevated temperature, increased voltage, high humidity, etc., for a long period of time. Burn-in tests, however, are very expensive to conduct, and are not guaranteed to uncover all latent defects. In any case, manufacturing variations in deep sub-micron technologies pose additional challenges when identifying an optimal test strategy.

Wearout failures are due to transistor degradation effects a chip experiences after prolonged operation periods. Different wearout mechanisms, like electromigration, oxide wearout and breakdown, hot carrier injection (HCI) and negative bias temperature instability (NBTI) have received increased attention in the context of reliability analysis [Segura05].

1.3. Failure Analysis in the Automotive Domain

Traditionally, automotive failure analysis is conducted by means of functional tests performed at different stages during the lifetime of a vehicle. The diagnostic capabilities of such tests are limited, as they can only perform data measurements, conduct plausibility checks and, to some degree, verify the physical connections between ECU components. As the amount of semiconductor devices in the car increases, these tests are no longer sufficient to provide a reasonable estimation of the integrity of semiconductor devices.

Functional tests are nowadays carried out in the field during the regular operation of a vehicle. If a problem is detected, a corresponding diagnostic trouble code [J19] is stored in the ECU memory and reported to the driver with a visual signal on the dashboard. The analysis of the reported failure begins at the workshop, where error information gathered during the operation of the system is collected. Then, the analysis and processing of this information takes place in different companies with different diagnostic strategies.

For repair purposes in the workshop, it is only necessary to identify the faulty ECU, which is replaced on the spot. The failure information is analyzed by the original equipment manufacturer (OEM) and the affected ECUs are sent to the first suppliers in the chain or *tier 1 suppliers*, who conduct another battery of tests in order to identify the root cause of the problem. As illustrated in Figure 1.3(a) this procedure may be repeated by several companies in the supply chain until, in case of a semiconductor failure, the problem is reported to the semiconductor provider. For subsequent analysis at the logic level, the chip has to be removed from the ECU and tested in isolation.

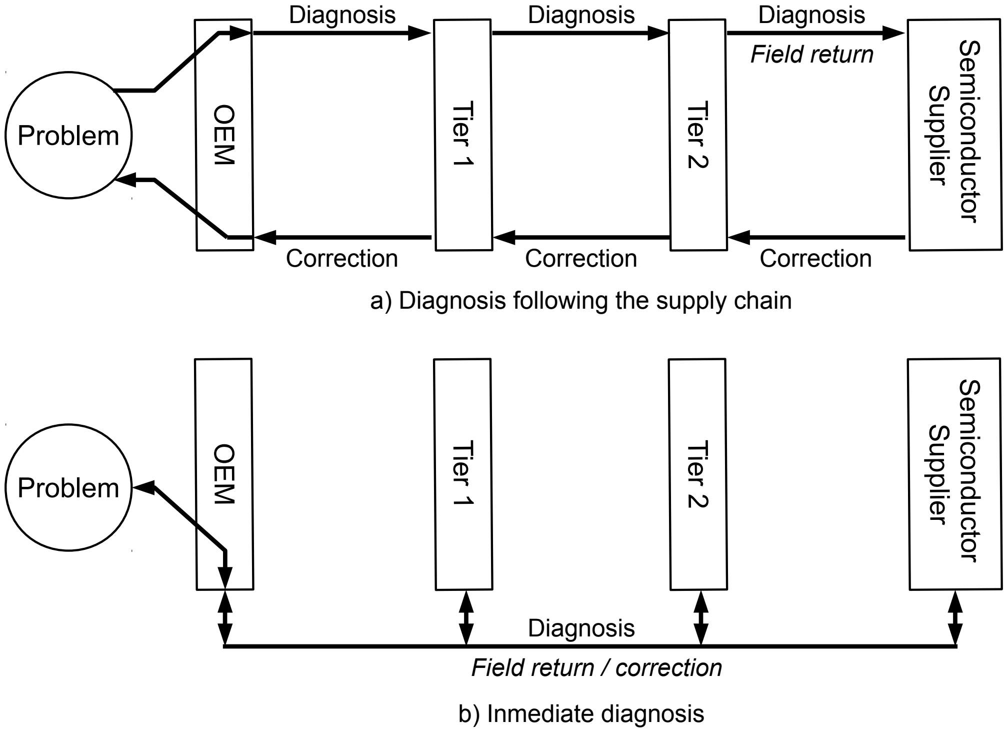


Figure 1.3.: Automotive semiconductor diagnosis

This traditional semiconductor diagnostic flow has two main drawbacks: firstly, it requires a given supplier in the chain to wait for the diagnostic information from the previous supplier before performing its own diagnostic procedure. Secondly, for detailed semiconductor diagnosis a chip has to be removed from its enclosing system. The desoldering process exposes the ECU to heat and physical stress, which may render the IC useless for further examination. Additionally, even when a chip is successfully detached from its board, subsequent structural tests may not reveal the presence of a fault. This is known as the "no trouble found (NTF)" syndrome [Conroy05], which is due to the fact that test conditions at the chip manufacturer are different from those of the chip's operating environment in the ECU.

In the proposed diagnostic flow shown in Figure 1.3(b), ICs are equipped with DfT infrastructure, whose access mechanisms are made available to the OEM. This makes it possible to perform structural diagnosis early in the failure analysis flow and gather detailed diagnostic information from chips under operating conditions very similar to those they experience during regular operation.

Structural diagnosis is the process of identifying a faulty component in a vehicle. For a hardware component under diagnosis C_{ud}, a structural test set $\mathsf{T} = [t_1, t_2, \cdots, t_n]$ comprising n tests is applied, and the corresponding test responses $\mathsf{C}_{\mathsf{ud}}(\mathsf{T}) = [r_1, r_2, \cdots, r_m]$ are gathered from the vehicle. A structural model C of each hardware component describes its internal elements and their interconnections. The same test set T is applied to this component representation and fault-free responses $\mathsf{C}(\mathsf{T})$ are obtained. Structural diagnosis analyzes $\mathsf{C}_{\mathsf{ud}}(\mathsf{T})$ and $\mathsf{C}(\mathsf{T})$ and identifies a faulty hardware element.

In some diagnostic situations it is sufficient to identify a defective ECU. As structural tests are performed separately for each ECU, a straight-forward comparison between the expected test responses $\mathsf{C}(\mathsf{T})$ and the observed responses $\mathsf{C}_{\mathsf{ud}}(\mathsf{T})$ can already achieve this goal. However, for other use-cases, where the defect location in the chip has to be derived, a more detailed analysis of the structural differences between C and C_{ud} is mandatory to find the root cause of the observed erroneous behavior.

Structural diagnosis offers benefits for the OEM, the chip manufacturer as well as for suppliers in-between. Structural tests help OEMs assess the hardware integrity of the ICs in the vehicle. Such tests are generated according to a fault model and, therefore, OEMs can precisely quantify test effectiveness. This capability has already become a requirement in the prevailing automotive safety standard for passenger vehicles [ISOc, IEC].

The application of structural tests automatically enables component diagnosis at system and ECU level. That is, they unequivocally identify a single faulty chip so that its enclosing ECU can be replaced and further analyzed. Finally, and maybe most importantly, the gathered diagnostic information reduces the number of NTF cases, since the test application is more likely to capture the observed failure mechanism. Moreover, a systematic semiconductor problem can be more efficiently handled once the collected diagnostic information is delivered to the chip manufacturer, who analyzes any systematic yield problem and adopts any suitable countermeasures during chip fabrication and testing. In this way, the time and effort devoted to failure analysis in the traditional diagnostic approach is greatly shortened, the number of future faulty vehicles is drastically reduced, and product quality is improved.

Similarly, chip manufacturers can factor in the diagnostic data collected in the field and improve their existing diagnostic procedures. Such detailed structural diagnostic information of all faulty chips is extremely valuable as the chip manufacturer optimizes yield [Keim06, Blanton12].

Finally, all the suppliers benefit from the developed flow, since a faulty chip is uniquely identified, which again speeds up diagnostic activities by identifying hardware problems and distinguishing faulty chips as early as possible.

1.4. Diagnostic Use-Cases

The various requirements and constraints at each step of the automotive failure analysis process have been categorized into four diagnostic use-cases, namely, *semiconductor failure analysis*, *workshop test*, *power-up/down test*, *online test*. They reflect current industrial needs in terms of diagnostic resolution, test requirements and application time, available test equipment, and safety/security.

1.4.1. Workshop Test

In workshop test it is most important to identify a faulty ECU to be replaced. The objective is to fix any malfunction as soon as possible so that vehicle operation is no longer compromised. As the performance of a workshop is directly visible to the end user, the effectiveness and efficiency of workshop tests play an important role for customer perception. For example, several visits to the workshop to correct the same problem can severely damage customer loyalty.

The application of structural tests, or the collection of structural test results stored in the vehicle during its operation, can greatly improve workshop performance. As structural tests can unequivocally identify a faulty hardware component, they can efficiently distinguish which ECUs need to be replaced to restore correct system operation. Furthermore, structural tests in the workshop can also reduce service and repair costs, as fault-free units are rarely discarded.

During workshop test there is no direct access to the pins of a hardware component, so tests have to be performed with the tools and interfaces commonly used in the automotive domain. Also, the test time per ECU has to be within the range of minutes. Since workshop test is usually performed by third-party companies, warranty and security limitations have to be enforced. This means that no sensitive information should be disclosed for the application of the test and the state of the ECU after the test has to comply with the corresponding warranty agreement.

1.4.2. Semiconductor Failure Analysis

Semiconductor failure analysis is performed for all field returns after a faulty ECU is identified. This includes, but is not limited to, the faulty ECUs identified during workshop test. All field returns are analyzed further in order to establish the nature of the observed problem and to determine which company needs to conduct subsequent failure analysis.

As a consequence, this use-case spans a wide array of activities carried out by the OEM, as well as tier 1 and 2 suppliers, in order to better understand hardware issues and take appropriate preventive actions in the production of the next batch of vehicles. However, as the faulty ECU needs to go through several test procedures, an important constraint in this use-case is that no applied test should render the ECU unsuitable for additional testing. This implies, for example, that a faulty chip cannot be removed from its board to be tested separately.

Structural tests help all the involved parties identify a faulty chip and accelerate system and board diagnosis. Furthermore, the collection of structural diagnostic information of each integrated circuit in the ECU provides semiconductor manufacturers with valuable information for chip quality improvement. For this purpose, the collected diagnostic data should convey sufficient information to enable the detailed analysis of the most likely faulty locations in the chip. This diagnostic step requires only the handover of the gathered diagnostic data between the company applying the test and the semiconductor provider.

In this use-case the test application process is simpler than that of workshop test, since dedicated test interfaces may be available in the ECU and generic automotive test equipment may be used. However, the required diagnostic granularity in this use-case makes the analysis of the collected data much more elaborate.

Finally, there are no stringent requirements concerning the maximum amount of time devoted to test and diagnosis.

1.4.3. Power-Up/Down Test

Power-up and power-down tests take place before system initialization when power is first applied to the ECU and before system shut down, respectively. The application of structural tests during ECU power-up/down serves two different purposes. On the one hand, they identify a faulty IC and direct the driver to the workshop, where the corresponding ECU can be replaced and further analyzed. On the other hand, they gather diagnostic information under realistic operating conditions. This offers the opportunity to accurately capture the underlying defect mechanism affecting chips and enables semiconductor manufacturers to optimize the analysis of any potential systematic fabrication problem.

The diagnostic benefits of power-up/down tests bring new challenges: As tests are conducted autonomously and any produced diagnostic data has to be stored on-chip, the diagnostic procedure has to be optimized to reduce storage costs. Consequently, any suitable diagnostic procedure has to support larger compaction ratios than those commonly used during traditional semiconductor failure analysis during IC manufacturing.

In order to ensure the safe application of power-up/down tests, additional constraints have to be met so that they do not interfere with the functional behavior of the system. As a rule of thumb in the industry, it is estimated that the time budget for power-up test is roughly 5 ms, while for power-down it amounts to 5 seconds. Furthermore, no external equipment is available during test.

1.4.4. Online Test

Online structural tests are performed during the regular operation of the system. Therefore, as they may interfere with the system's functional tasks, their execution alongside regular applications has to be thoroughly analyzed. As a consequence, online structural tests are subject to stringent safety constraints.

The main purpose of online structural tests is to bring the system to a safe state in case of any malfunction. The required diagnostic granularity is, therefore, the ECU level. However, a more detailed diagnostic outcome is also desirable to support subsequent diagnostic activities in the workshop or during semiconductor failure analysis. As tests are conducted during vehicle operation, online tests offer a unique opportunity to capture in-field errors and further alleviate the NTF problem.

Figure 1.4 depicts the application of structural tests for the diagnostic use-cases in chronological order. Note how structural tests and functional tasks are interleaved during online test.

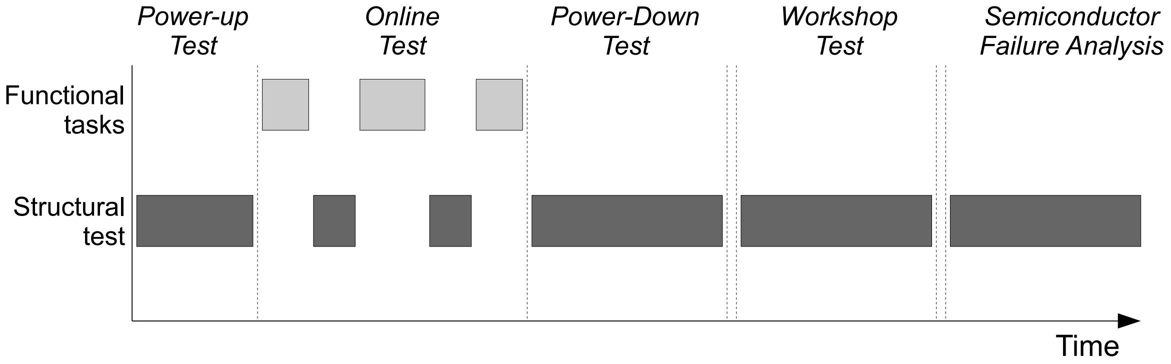


Figure 1.4.: Diagnostic use-cases and test application

1.5. Organization and Contributions

This chapter has so far described semiconductor failure analysis problems in the automotive domain. The challenges tackled in this dissertation have been identified and the benefits of a structural diagnostic solution have been analyzed in the context of the automotive industry. Industrial requirements for a comprehensive diagnostic solution have been explained and categorized into diagnostic use-cases.

In the remainder of this dissertation, structural diagnostic methods and algorithms are presented that offer end-to-end diagnostic capabilities throughout the complete failure analysis process of complex systems.

The rest of this dissertation is organized as follows: the next chapter presents the foundations of structural test and diagnosis for random digital logic. The information described in that section comprises the fundamental concepts and techniques for the development of a suitable semiconductor diagnostic strategy.

Chapter 3 presents relevant related work in the area of system test and diagnosis of complex electronic systems.

Chapters 4 to 6 describe the developed structural diagnostic solutions for the analysis of semiconductor failures in complex embedded systems. The applicability to the presented diagnostic use-cases in the automotive domain is discussed in these chapters where appropriate.

Chapter 4 details a conceptual system architecture for efficient access to on-chip test infrastructure during system-level test. The components and operation of this architecture are explained and an industrial case-study is presented, where a viable diagnostic solution is realized.

Chapter 5 presents an approach to logic diagnosis using highly compacted test signatures. The performance of the developed diagnostic algorithm can be tuned to the number of faults and the volume of test response data expected in various diagnostic applications.

Chapter 6 describes a novel architecture for autonomous test and diagnosis of logic circuits. The presented diagnostic solution strongly reduces the amount of test data to be stored on-chip, both for the generation of test patterns and for the compaction of test responses.

Chapter 7 analyzes the performance of the presented diagnostic solutions. The algorithms of chapter 5 are evaluated by means of the achieved diagnostic accuracy, while the diagnostic architecture of chapter 6 is evaluated in terms both of diagnostic accuracy and hardware costs.

Finally, chapter 8 provides a summary of this work and a few concluding remarks about the contribution of this dissertation, which may pave the way for further research in this field.

2. Structural Test and Diagnosis

Structural test and diagnosis play a fundamental role in chip manufacturing. Therefore, a lot of research effort has been already devoted to realize effective and cost-efficient test and diagnostic solutions. In the following sections, the fundamental ideas behind structural testing are briefly presented. These concepts and techniques provide the foundations for the contributions of this dissertation.

2.1. Defects and Faults

A *defect* is any distortion or undesired physical property in a manufactured silicon chip. A defect may have a fixed location in the chip (*defect site*) or it may be distributed among multiple defect sites. The focus of this dissertation is on defects in a defined location in the chip. Such defects are also referred to as *spot defects*. The nature of a defect is tightly coupled to the underlying *defect mechanism*, that is, the physical process that originates defects in the manufacturing process. Understanding the characteristics of defect mechanisms affecting chip fabrication is the most important goal semiconductor manufacturers pursue during failure analysis, as this information allows continuous improvement of yield and product quality.

Faults are defect representations that capture the most important defect properties and observable behavior. A fault models an alteration in the function of a circuit without regard to the physical characteristics of the chip. Therefore, faults offer a useful abstraction for the efficient development of test and diagnostic algorithms without considering exact physical device properties.

2.2. Fault Modeling

A fault model specifies the characteristics of the faults assumed to reflect the behavior of a physical problem in the chip. One of the most widely used fault models is the *stuck-at* model [Eldred59], where a faulty signal has its value fixed either to logic 1 (*stuck-at-1*) or to logic 0 (*stuck-at-0*). Stuck-at faults are *static* faults. That is, their behavior is fully specified by the current state of the circuit. Conversely, *dynamic* faults are also influenced by any circuit state that took place in the past. For example *gross delay faults* [Krstic98], also known as *transition delay faults*, assume that the delay of a gate is large enough so that a transition in its output is not observed in any of the circuit's outputs. For such a fault to occur, the logic state of the gate's output has to change its value in two consecutive clock cycles. Other less restrictive models account for the fact that the delay of a single gate may not be enough to make every propagation path fail. These faults are known as *small delay faults* [Park88] and their importance is increasing due to narrow timing margins and variations in newer manufacturing technologies.

More complex fault models have also been devised to resemble defects more closely. Several fault models describe *bridging faults* [Wunderlich10]. They model the various effects the unintended connection of signal lines may have on one another. Similarly, *crosstalk faults* describe capacitive couplings between neighboring lines, which are prone to experiencing additional delay or unexpected glitches [Chen97].

In order to analyze complex defect mechanisms, the *conditional stuck-at (CSA)* fault model [Holst12] offers a powerful generalization method to describe arbitrary faults. The representation of a CSA is:

$$\textit{stuck-at}[condition]$$

A fault in this model behaves as a stuck-at fault when its activation condition evaluates to true. Alternatively, the corresponding signal line remains fault-free if the activation condition is not satisfied. That is, depending on the activation condition, a line may behave as a stuck-at-1 or stuck-at-0 fault for some patterns, or as a fault-free line for other patterns that would detect an *unconditional* stuck-at fault on this line. For instance, *stuck-at-0-*$v[v]$ is a stuck-at-0 on signal line v, *stuck-at-1-*$v[\bar{v}]$ a stuck-at-1 on v, while *stuck-at-0-*$v[\overline{v_0} \wedge v_1]$ describes a slow-to-rise transition delay fault. Similarly, CSA descriptions of more complex faults can also be easily derived [Holst12].

2.3. Scan Design

The testability of a logic core is determined by two main metrics: *controllability* and *observability*. Scan design [Eichelberger77] is one fundamental technique to improve the testability of any logic design and make structural tests more effective and efficient. As Figure 2.1 shows, the idea behind scan design is to introduce a piece of design-for-test (DfT) infrastructure to connect internal state elements, like latches or flip-flops, into one or more shift registers or *scan chains*, which can be controlled and observed externally without regard to the their surrounding logic resources.

During structural tests any desired logic value can be shifted into a scan chain. Then, one or a few *capture* clock cycles are applied to the circuit in regular mode and, finally, the captured logic values can be shifted out of the scan chain so that the test result can be evaluated.

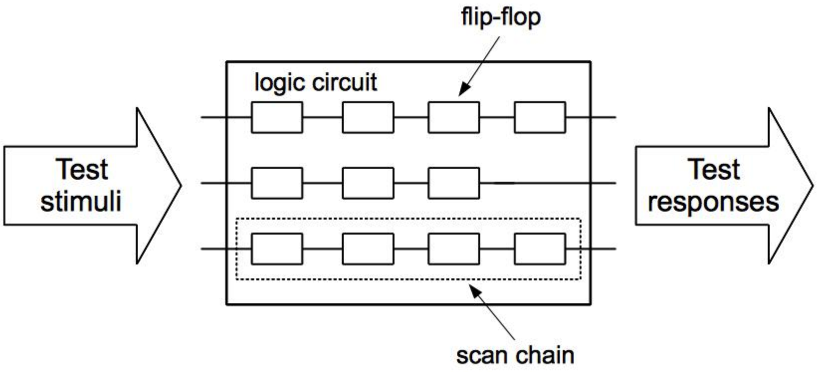


Figure 2.1.: Principle of scan design

The inputs of scannable state elements are known as pseudo-primary inputs (PPIs) and their outputs as pseudo-primary outputs (PPOs). If each state element in a circuit is part of a scan chain, the circuit is said to be a *full scan design*. The benefit of full scan is that, as shown in Figure 2.2, the circuit can be represented with an equivalent combinational circuit where the pseudo-primary inputs and pseudo-primary outputs can be regarded as regular inputs and outputs, respectively. Furthermore, as long as the added DfT infrastructure is fault-free, any test and diagnostic algorithm for combinational circuits may also be used for sequential circuits without modification.

In the remainder of this dissertation, circuits are assumed to be full scan designs.

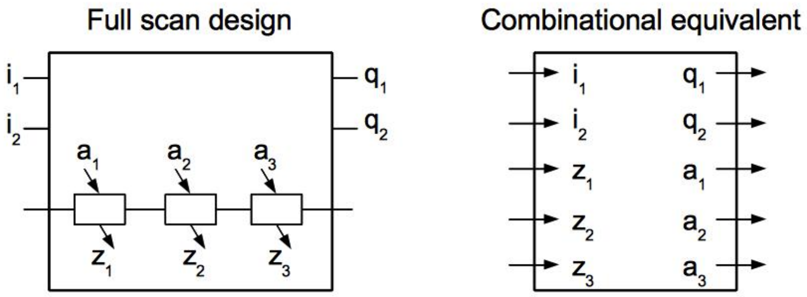


Figure 2.2.: Full scan design and equivalent combinational representation

2.4. Access to On-Chip Test Infrastructure

The design of today's complex systems-on-chip (SOCs) involves the integration of specialized intellectual property (IP) blocks into a single chip. These IP blocks are designed and verified once and reused for several designs. SOC designers usually acquire off-the-shelf IP cores in order to realize increasingly complex functionality in a relatively short amount of time. However, special DfT resources are needed so that these cores can also be tested efficiently once integrated into a chip. The conceptual architecture for modular core test of SOCs is shown in Figure 2.3 [Zorian98]. It comprises three main components:

- *Test source and sink*. The *test source* generates test patterns to be applied to the logic core, while the *test sink* receives the test responses and compares them to their expected values. The test source and sink may be located either on-chip or off-chip, within the system or in a remote location.
- *Test access mechanism (TAM)*. The TAM transports test data. It provides the communication means between the embedded core and the *test source* and *test sink*.
- *Core test wrapper*. The core test wrapper provides an interface that enables the communication between the embedded core and its environment. It isolates a core so that it can be tested on its own, without regard to the rest of the on-chip resources. Finally, the test core wrapper also connects the embedded core to the available TAM.

The challenges to architect an efficient modular test solution for SOCs have originated important standardization efforts [JTA, IEE, Stollon11] and sparked research methodologies for test scheduling optimization [Wang06a]. In the following subsections, the

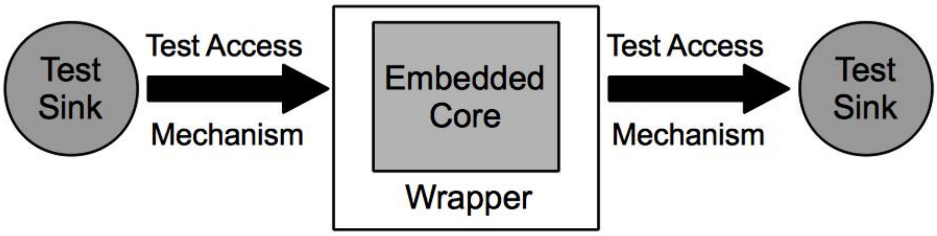


Figure 2.3.: Conceptual SOC test architecture [Zorian98]

most relevant standardization efforts are summarized, which offer potential to ease test integration.

2.4.1. IEEE 1149.1 Standard

The IEEE 1149.1 standard [JTA] or *joint test action group (JTAG)* was originally devised for testing the interconnection between several integrated circuits (ICs) on a printed circuit board (PCB). This technique, also known as *boundary scan*, connects all chip's input/output (IO) pins into a single shift register, so they can be easily controlled and observed. Since its conception, the IEEE 1149.1 has also been widely used as a generic interface to gain access to internal circuit structures for debug and test.

The minimal realization of this interface consists of four mandatory IO pins, namely, *test clock (TCK)*, *test mode select (TMS)*, *test data input (TDI)* and *test data output (TDO)*. As Figure 2.4 shows, the IEEE 1149.1 standard includes a *test access port (TAP)* controller, which implements a standardized finite state machine in order to access the boundary scan cells or other internal registers that realize additional built-in features in the chip. The *instruction register* configures the state of the device to perform the desired operation. The provided functionality is further controlled with the corresponding *data register*.

User-defined registers offer additional services not covered by the standard, while the *bypass register* connects the TDO to the TDI pin through a one-bit shift register in order to provide faster access to other daisy-chained devices connected to the same IEEE 1149.1 interface.

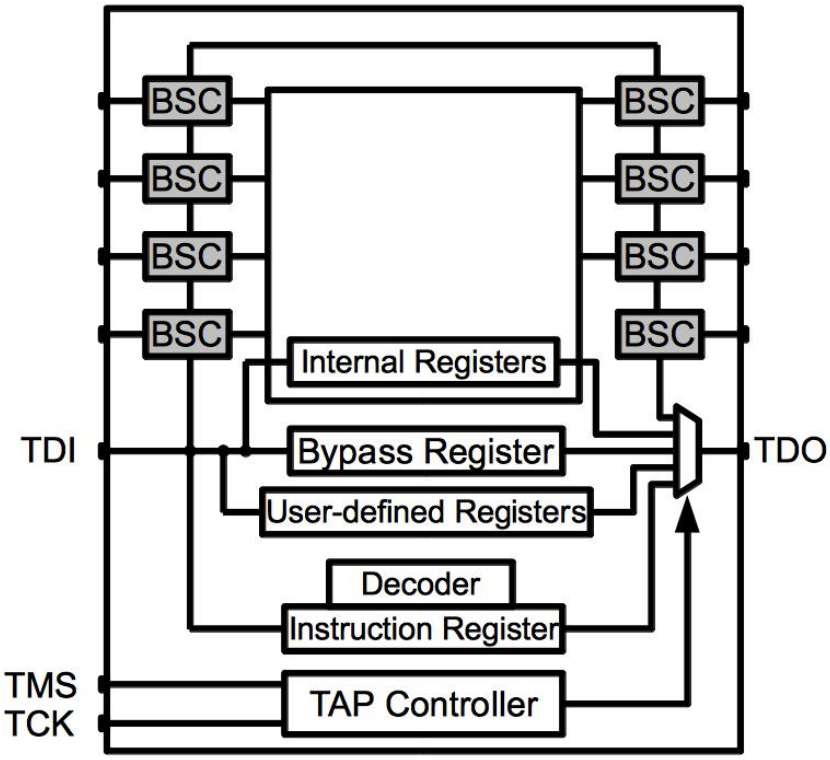


Figure 2.4.: IEEE 1149.1 architecture [JTA]

2.4.2. IEEE P1687

The complexity of today's systems-on-chip has lead to the extensive use of on-chip instrumentation for test, debug, diagnosis, power management, and on-chip measurements. The upcoming IEEE P1687 standard [Stollon11], also known as *internal joint test action group (IJTAG)*, aims at providing a common description of on-chip embedded features as well as standardized communication protocols to interact with embedded instruments. The IEEE P1687 proposal is an extension of the IEEE 1149.1 standard to make use of the TAP controller to manage the operation of on-chip instruments.

The complexity in the access to embedded on-chip instruments that comply with the IEEE P1687 has justified recent research efforts for instrument access verification [Zadegan11, Baranowski12], optimization [Larsson12, Baranowski13a] and protection [Baranowski13b].

2.4.3. IEEE 1500 Standard

Complex integrated circuits place additional requirements on the test application process well beyond the capabilities of the IEEE 1149.1 standard. Such designs make use of hierarchical IP blocks from different core providers, which require very efficient test access mechanisms for test integration. As Figure 2.5 shows, the IEEE 1500 standard [IEE] provides a test wrapper for the IO terminals of each core and exposes an interface with which the test application can be controlled. Such common interface for core-based test specifies the interaction between logic cores in order to facilitate test reuse.

The IEEE 1500 wrappers consists of a mandatory *wrapper serial port (WSP)* and an optional parallel TAM. The WSP consists of the *wrapper serial input (WSI)* and the *wrapper serial output (WSO)* and several *wrapper serial control* signals. Similar to the IEEE 1149.1 standard, the IEEE 1500 wrapper defines a *wrapper instruction register (WIR)*, which determines the instruction to be executed in the core. This register can select predefined functionality, for example, to bypass the entire core or to configure wrapper boundary cells into a shift register. However, the WSP does not make use of a TAP controller and the input signals of the wrapper's interface are applied directly to the core. The IEEE 1500 test wrapper also supports an optional parallel TAM in order to increase the efficiency of SOC testing.

The IEEE 1500 standard defines the interface between the core wrapper and the TAM, while the design of the TAM itself is left open for SOC integrators.

2.5. Logic Built-In Self-Test

Logic built-in self-test (BIST), or LBIST, integrates DfT resources for autonomous test pattern generation and on-chip response compaction [Wunderlich98].

The most widely used LBIST architecture is shown in Figure 2.6, commonly referred to as the *STUMPS* architecture (self-testing unit using MISR and parallel sequence generator) [Bardell82]. A test pattern generator (TPG) produces inexpensive pseudo-random test patterns (PRPs). As most designs cannot be sufficiently tested with random patterns, most logic BIST solutions employ one of two approaches. The design can be modified by means of *test point insertion* so that PRPs are more likely to exercise a

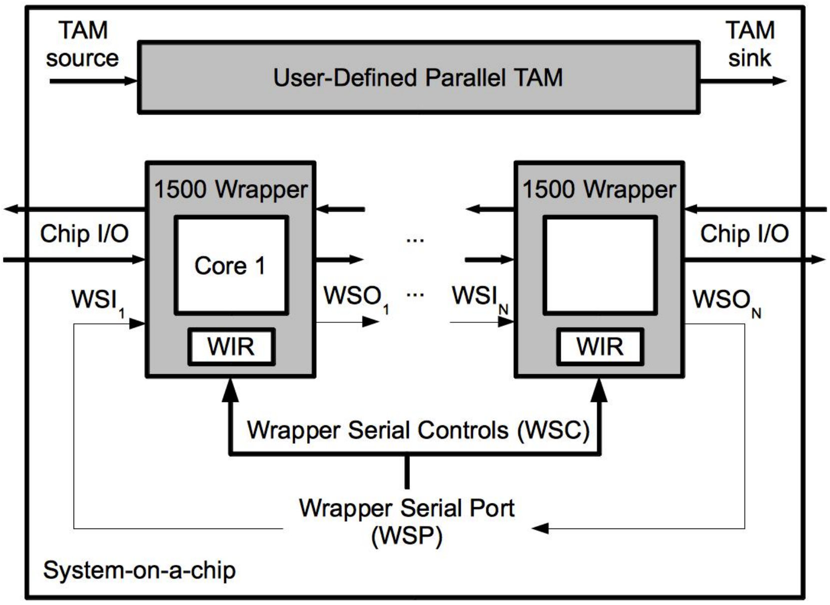


Figure 2.5.: Overview of the IEEE 1500 standard [IEE]

larger portion of the chip. Alternatively, deterministic patterns can be generated for the remaining undetected faults after the application of inexpensive patterns. These patterns are encoded and stored on-chip, and later reconstructed for test application. This test application procedure is known as *mixed-mode BIST* [Hakmi07, Hakmi09, Hellebrand92, Hellebrand98, Könemann91, Könemann01, Rajski04, Volkerink03].

The test patterns produced by the TPG are usually fed into parallel scan chains by means of a *phase shifter*. The phase shifter removes linear dependencies in the produced test sequence, which negatively affect achievable fault coverage [Rajski00].

After the application of one or more capture cycles, test responses in the scan chains are shifted out and compacted in space (*space compactor*) [Mitra04] and then compacted in time by means of a multiple-input signature register (MISR). During MISR compaction, the following test pattern is simultaneously shifted into the scan chains.

A logic built-in self-test (LBIST) test session usually compacts the whole test application into a single test *signature*, which is later compared to the expected good signature. As test results are accumulated over time, any source of undefined values ("*X*" values) may compromise the test outcome. The most common sources for "*X*" values are, for example, analog blocks, memories and non-scannable storage elements, combinational feedback loops, asynchronous set/reset signals, tristate buses, false paths, among others [Wang06a]. Therefore, any LBIST solution requires careful consideration to mask out any undefined value before it propagates to the test signature [Hetherington99, Tang06, Czysz10].

A LBIST controller manages the test application process by generating clocks, scan enable signals and control signals for the TPG and MISR.

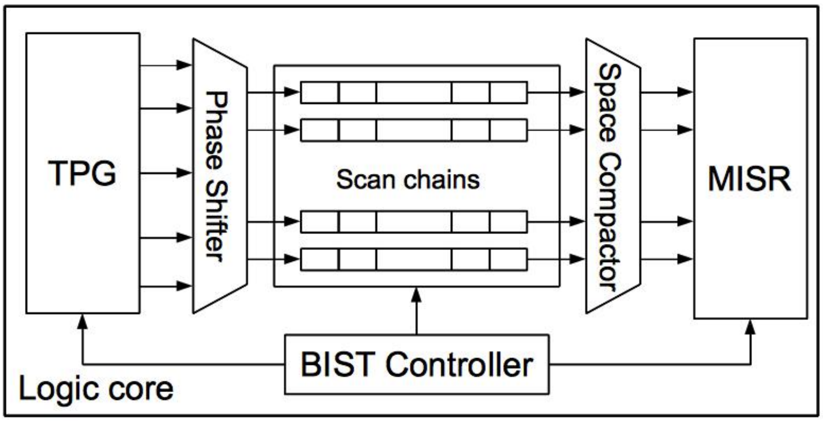


Figure 2.6.: STUMPS architecture

As LBIST solutions can apply structural tests without the need of external equipment, they are well-suited for safety-critical or mission-critical operations, like those usually found in the automotive, telecommunications, networking and health care domains, where autonomous in-system self-tests are advantageous to improve system reliability and conduct remote diagnostics [Vo06, Wang06a, Qian09].

In the next sub-sections, the most fundamental LBIST techniques for test pattern generation and response compaction are analyzed in some detail.

2.5.1. Test Pattern Generation

Test pattern generation for LBIST is most commonly accomplished by means of *linear feedback shift registers (LFSRs)*. LFSRs are one of the fundamental building blocks for *pseudo-random testing*, *pseudo-exhaustive testing*, and on-chip deterministic pattern decoding [Rajski04].

LFSRs consist of D flip-flops and a linear feedback network [Wang06a]. Figure 2.7 shows the *standard LFSR*, where the XOR gates are placed in the external feedback path. Conversely, the *modular LFSR*, shown in Figure 2.8, has XOR gates placed between two adjacent flip-flops.

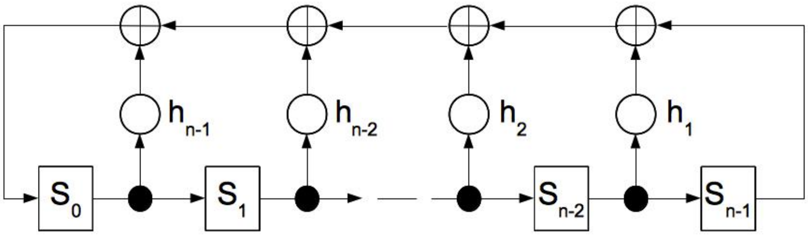


Figure 2.7.: n-stage standard LFSR

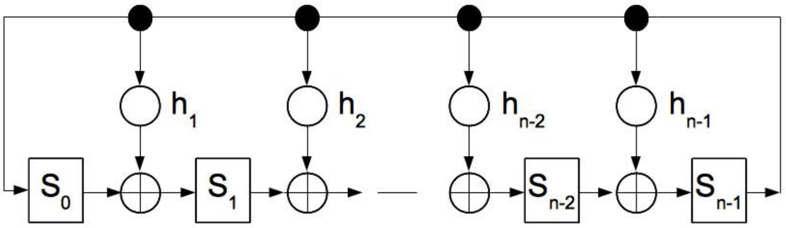


Figure 2.8.: n-stage modular LFSR

The structure of an n-stage LFSR is characterized by a *generator polynomial* $f(x)$ of degree n.

$$f(x) = 1 + h_1 x + h_2 x^2 + \cdots + h_{n-1} x^{n-1} + x^n$$

The coefficients $h_i \in \{0, 1\}$ of $f(x)$ specify the presence or absence of a feedback path between flip-flop i and flip-flop $i - 1$.

The operation of both standard LFSRs and modular LFSR can be described by a Galois field of order 2 (GF(2)). The state of an LFSR in the next clock cycle $S(t + 1)$ can be expressed in terms of the current LFSR state $S(t)$. For a standard LFSR this transition function is [Bushnell00]:

$$S(t+1) = T_{Standard} \cdot S(t)$$

where $T_{Standard}$ is:

$$T_{Standard} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ 1 & h_1 & h_2 & \cdots & h_{n-2} & h_{n-1} \end{bmatrix} \tag{2.1}$$

Similarly, for a modular LFSR the transition function is [Bushnell00]:

$$S(t+1) = T_{Modular} \cdot S(t)$$

and $T_{Modular}$ is:

$$T_{Modular} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & h_1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & h_{n-2} \\ 0 & 0 & 0 & \cdots & 1 & h_{n-1} \end{bmatrix} \tag{2.2}$$

A *primitive polynomial* is a generator polynomial, which produces all possible states of an *n*-stage LFSR with the exception of the all-zero state. Such an LFSR with a primitive polynomial reaches $2^n - 1$ states and is known as *maximum-length LFSR*. A primitive polynomial $p(x)$ divides the polynomial $1 + x^k$ for $k = 2^n - 1$ but not for any other value smaller than k [Wang06a].

Pseudo-Random Testing

Inexpensive pseudo-random patterns can be generated on chip and applied to the circuit-under-test (CUT) [Bardell87]. Test pattern generators for pseudo-random testing are usually based on maximum-length LFSRs. They can produce test sequences with uniformly distributed 1's and 0's. That is, the probability of generating either a 1 or a 0 in a test pattern is in both cases 0.5. Pseudo-random patterns may not achieve maximum fault coverage since they are unlikely to generate a test for *random-pattern resistant faults*. For example, a ten-input AND-gate requires all inputs to be simultaneously 1 in order to test for a stuck-at-0 at the gate output. Such a test pattern has low occurrence probability with a pseudo-random test pattern generator.

In order to improve fault coverage in pseudo-random testing, a *weighted LFSR* can be employed. A weighted LFSR extends an LFSR with combinational logic to bias the probability of producing a 1 or a 0. This can be easily achieved with the circuit of Figure 2.9, where k LFSR outputs feed a k-input AND-gate, producing a 1 with a probability of $(0.5)^k$. The desired output probability is chosen with the help of the *weight* *select* and *inversion* signals. Good results have been reported for output probability of a multiple of 0.25 [Chin84] and 0.125 [Wunderlich87].

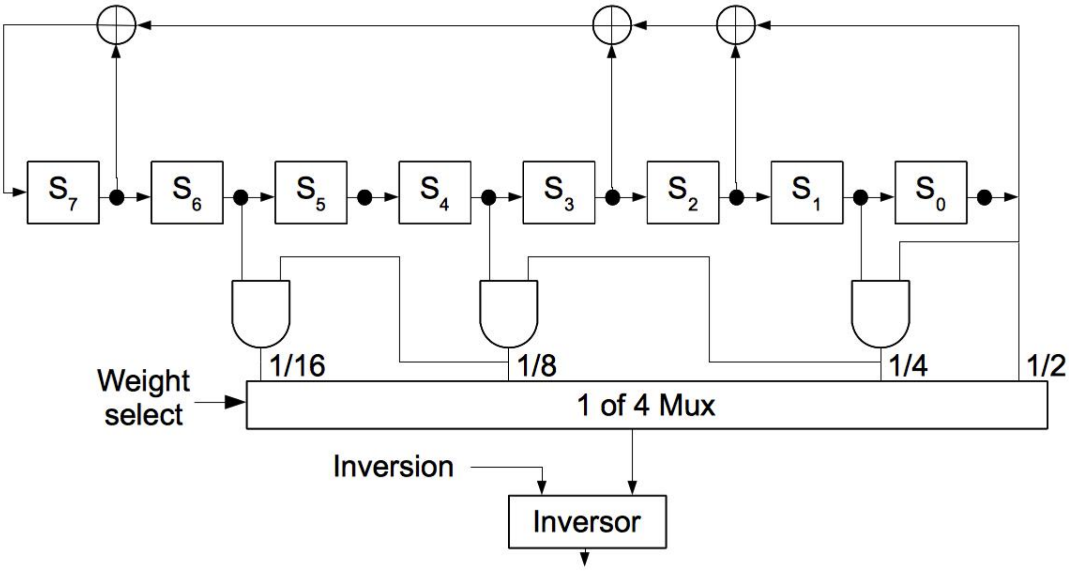


Figure 2.9.: Weighted pseudo-random pattern generator [Bushnell00]

Exhaustive Testing

Exhaustive testing applies all 2^n different test patterns to an n-input combinational circuit [Barzilai83]. For this purpose, an n-stage maximum length LFSR can be used. Exhaustive testing achieves improved *defect coverage* since it relaxes the fault model assumptions for test generation. However, for large circuits the application of such a large number of test patterns becomes impractical [Bushnell00].

In order to reduce the size of the pattern set, *pseudo-exhaustive testing* [McCluskey84, Wang86, Hellebrand90] applies patterns to test the output functions of the circuit exhaustively. As each circuit's output depends only on a subset of the circuit's inputs, the number of required patterns is drastically reduced in comparison to regular exhaustive testing. For each output $o \in O$ the test set T_o enumerates all possible input combinations in the *input cone* $I(o)$ of o. That is, all inputs for which there is a topological circuit path to o. Therefore, the number of patterns to test the function of output o is $2^{|I(o)|}$. If the circuit's outputs are tested sequentially, the total number of patterns in the pattern set is $\sum\limits_{o \in O} 2^{|I(o)|}$.

More recently, *partial pseudo-exhaustive test (P-PET)* [Mumtaz11] has been proposed to optimize pseudo-exhaustive test generation. In P-PET, only outputs with input cone $|I(o)| < b$ are tested exhaustively, while the rest of the input cones are filled with pseudo-random patterns. To generate a P-PET sequence for a given bound b, a programmable LFSR [Hellebrand92] switches between several generator polynomials stored on chip, each one exercising a subset of inputs $I_b = \left\{ I(o) \middle| \; |I(o)| \leq b \right\}$, so that each input set in I_b receives all possible input combinations.

Mixed-Mode BIST

Non-exhaustive test pattern generation may suffer from insufficient fault coverage due to random-pattern resistant faults. In order to enhance fault coverage, test points can be inserted into the design [Krishnamurthy87, Touba96, Iyengar89]. They consist of control and observation points, which improve the fault detection capabilities of pseudo-random and pseudo-exhaustive pattern sequences. However, design modifications are sometimes not possible like, for example, for hard cores or macros, or have undesirable consequences like, hardware overhead and timing penalties.

Mixed-mode BIST is another approach to improve fault coverage that does not require any modifications to the target circuit. This technique generates deterministic patterns for random-pattern resistant faults and encodes them for efficient on-chip application [Hakmi07,Hakmi09,Hellebrand92,Hellebrand98,Könemann91,Könemann01,Rajski04,Volkerink03]. The decoding of the deterministic patterns for test application usually takes advantage of the available LFSR for pseudo-random or pseudo-exhaustive pattern generation. The deterministic test data are stored in the *seed memory*, which may be located either on-chip [Hakmi07, Hakmi09, Hellebrand92, Hellebrand98] or off-chip [Könemann91, Könemann01, Rajski04, Volkerink03].

2.5.2. Test Response Compaction

Test responses in a LBIST session account for such a large amount of information that they cannot be efficiently stored on-chip. Therefore, the output test data volume needs to be reduced at the expense of some information loss. This procedure is known as *test response compaction*. Usually, the responses of the complete test set are compacted into a single *signature* that represents a statistical property of the CUT. In order to identify a faulty chip, the produced test signature is compared to the known good signature.

Any technique used for test response compaction needs to ensure with sufficient confidence that the faulty and fault-free signatures are different. Otherwise, a faulty circuit can no longer be distinguished. In this case, the faulty test response is considered an *alias* of the correct response.

Signature Analysis

Signature analysis is a technique for test response compaction based on *cyclic redundancy checking* [Peterson72]. The technique is realized using an LFSR to compact the output values in the scan chains [Wang06a]. In this subsection, the most common hardware structures for signature analysis are introduced. Then, the *aliasing probability* for these structures is derived. That is, the probability that a fault produces an error in the output response sequence, but the resulting signature is indistinguishable from the correct signature.

- Serial-input signature register (SISR)

  A single circuit output or scan chain can be compacted by means of an LFSR with an additional XOR input. Figure 2.10 shows such a n-stage *serial-input signature register (SISR)*. The circuit's output response $M = \{m_0, m_1, m_2, \cdots, m_{L-1}\}$ consists of L bits comprising the complete BIST session.

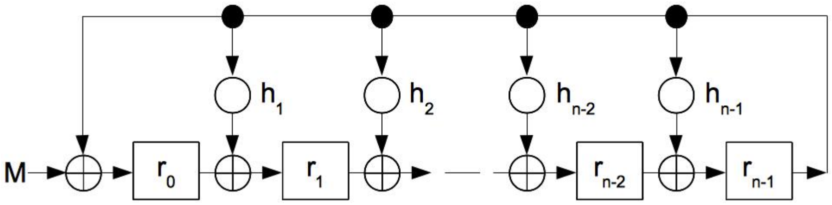


Figure 2.10.: n-stage single-input signature register (SISR)

  M is represented by the polynomial:

$$M(x) = m_0 + m_1 x + m_2 x^2 + \cdots + m_{L-1} x^{L-1}$$

  After the L-bit output response is shifted in and compacted, the state of the SISR

$R = \{r_0, r_1, r_2, \cdots, r_{n-1}\}$ is given by:

$$r(x) = r_0 + r_1x + r_2x^2 + \cdots + r_{n-1}x^{n-1}$$

The SISR compactor performs polynomial division between $M(x)$ and $f(x)$, where $f(x)$ is the SISR characteristic polynomial. This operation is expressed by the formula:

$$M(x) = q(x)f(x) + r(x)$$

where the final SISR state or signature is the *polynomial remainder* $r(x)$ if the SISR is implemented as a modular LFSR. For the standard LFSR, the polynomial remainder can be identified with a different state assignment [Bushnell00].

- Multiple-input signature register (MISR)

In order to compact multiple outputs or scan chains simultaneously, a multiple-input signature register (MISR) can be used. A n-stage MISR compacts up to n output sequences of length L. Figure 2.11 shows such a MISR based on a modular LFSR, where the feedback signals and the corresponding output sequences are XOR-ed in each clock cycle.

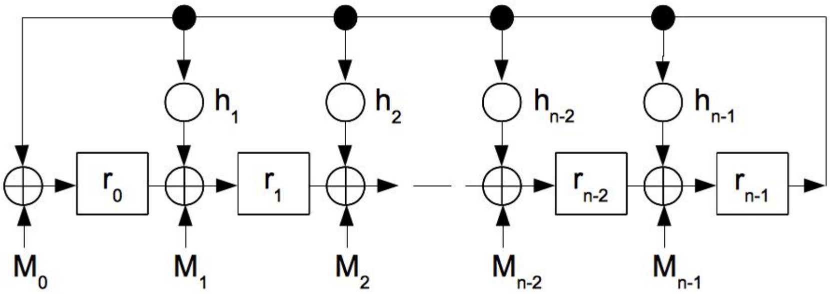


Figure 2.11.: n-stage multiple-input signature register (MISR)

An n-input MISR can be modeled as a single input SISR, whose *effective input sequence* $M_{eff}(x)$ is given by the expression:

$$M_{eff}(x) = M_0(x) + xM_1(x) + x^2M_2(x) + \cdots + x^{n-2}M_{n-2}(x) + x^{n-1}M_{n-1}(x)$$

As a consequence, MISR compaction also performs polynomial division between $M_{eff}(x)$ and the characteristic polynomial $f(x)$:

$$M_{eff}(x) = q(x)f(x) + r(x)$$

- Aliasing probability

  The probability that the correct output response and any faulty output response produce the same signature after compaction is known as the *aliasing probability*. The bounds on the aliasing probability P_A of an n-stage LFSR compactor can be calculated in terms of the error probability p of a single circuit output [Williams88]. If $0 < p \leq 0.5$ then $p^n \leq P_A < (1-p)^n$. Alternatively, if $0.5 \leq p \leq 1$, then $(1-p)^n \leq P_A \leq p^n$. Therefore, if the error probabilities are equally likely, the aliasing probability amounts to $P_A = 2^{-n}$, without regard to the initial state of the LFSR.

  In the more general case when the error probability $p \neq 0.5$, the aliasing probability P_A also converges to 2^{-n}. However, primitive characteristic polynomials reach this bound for shorter input sequences than non-primitive ones and offer the better choice for response compaction [Bushnell00].

  In case the circuit's outputs o_j for $j = \{1, 2, 3, \cdots, n\}$ in an n-stage MISR have unique error probabilities, the aliasing probability is also $P_A = 2^{-n}$, as long as the outputs o_j are independent and their error probabilities p_j are not correlated [Williams88].

  Under the same assumptions of statistically independent error probabilities, an intuitive way to determine the aliasing probability of an n-stage MISR is to consider the ratio between the total number of response sequences generating the fault-free n-bit signature, out of which one is the correct test response, and the total amount of response sequences. For a test sequence of length L and m circuit outputs or scan chains, where $L > n \geq m \geq 2$, the aliasing probability is [Wang06a]:

  $$P_A = \frac{2^{mL-n} - 1}{2^{mL} - 1}$$

  For $L \gg n$, $P_A \approx 2^{-n}$.

  The assumption that the outputs are independent usually does not hold in practice. However, output dependencies have only little impact on signature analysis [Bushnell00].

2.6. Logic Diagnosis

A comprehensive review of logic diagnosis for combinational circuits was presented in [Holst12]. In the following, only those approaches are discussed which are most relevant for the remaining chapters of this work, namely, the *inject-and-validate* and the *back-tracing* diagnostic paradigms.

2.6.1. Inject-and-Validate

The inject-and-validate paradigm relies on fault simulation to assess the degree to which a fault can *explain* the observed defect behavior. In contrast to prior work [Richman85], which make use of a *fault dictionary* to account for all possible defect mechanisms, inject-and-validate approaches do not assume that a fault model can perfectly describe the underlying failure behavior, but rather analyze the similarities between fault simulation and the observed test responses in order to infer the most likely fault locations or *candidates*.

One of the first such diagnostic approaches [Waicukauski89] identifies the stuck-at faults in the circuit, which are able to explain the biggest number of failing patterns. That is, for every fault $f \in \mathsf{F}$, where F is the set of stuck-at faults in circuit C, each test pattern $t \in \mathsf{T}$ is simulated and compared to the observed circuit response C_{ud}. Let $\mathrm{expl}(\mathsf{T}, f)$ be the subset of test patterns for which f explains the observed faulty response $\mathsf{C}^f(t)$:

$$\mathrm{expl}(\mathsf{T}, f) = \left\{ t \in \mathsf{T} \,\middle|\, \mathsf{C}^f(t) = \mathsf{C}_{\mathsf{ud}}(t) \neq \mathsf{C}(t) \right\}$$

The most likely fault candidates are:

$$\left\{ f \in \mathsf{F} \,\middle|\, |\,\mathrm{expl}(\mathsf{T}, f)\,| = \max\{\,|\,\mathrm{expl}(\mathsf{T}, g)\,| \,\middle|\, g \in \mathsf{F}\} \right\}$$

Later extensions of this diagnostic approach include the single location at a time (SLAT) method [Bartenstein01, Huisman04]. A test pattern $t \in \mathsf{T}$ has the SLAT property if there exists a fault $f \in \mathsf{F}$ that produces exactly the same test response as that of the circuit-under diagnosis C_{ud}: $\exists f \in \mathsf{F}$ with $\mathrm{expl}(\{t\}, f) \neq \emptyset$. The result of SLAT diagnosis is a set of *multiplets*, where a multiplet is a minimal subset of faults M able to account for all observed SLAT patterns and each fault $f \in \mathsf{M}$ explains at least one SLAT pattern.

Closely related to the SLAT property, the work in [Huang97] introduced the idea of *curable vectors* for design debug. A curable vector is a failing response that can be corrected by flipping a single signal line. This idea was first utilized for logic diagnosis roughly at the same time the SLAT approach was first presented [Huang01].

Several diagnostic approaches have extended the SLAT approach in order to take advantage of any diagnostic information in non-SLAT patterns [Lavo02, Wang06b, Zou06b, Liu07a]. They introduce new matching and scoring methods to improve diagnostic results. Diagnostic fault models have also been considered to account for complex failure mechanisms which are partly or totally explained by the composition of several simple faults [Millman90, Venkataraman00]. The SLAT technique has also been used to identify initial suspect lines, which are then analyzed according to specific fault models [Zou06a, Liu07b].

Most recently, the work in [Holst12] presents a generalization of the SLAT paradigm, which is able to extract failure information from both SLAT and non-SLAT patterns, even when test responses are highly compacted. Some of the main ideas behind these algorithms have also been used to deal with multiple interacting faults [Tang10, Ye11].

2.6.2. Back-Tracing

Back-tracing algorithms traverse the circuit's structure from outputs to inputs in order to reason about the circuits' internal logic state when a failing pattern is applied. This process *infers* the values of internal signals and identifies fault-free circuit locations, fault propagation paths, and fault candidates.

The most basic back-tracing approach, also called *structural pruning*, identifies all possible fault locations inside any topological circuit path that leads to a faulty output [Waicukauski89]. This technique is able to exclude a large number of circuit locations that cannot be responsible for the observed erroneous test response. However, as this procedure may still report many possible fault locations, it is typically used only as an initial diagnostic step to reduce the number of locations to be considered by more elaborate diagnostic algorithms.

Logic reasoning [Abramovici80] finds a set of internal signal assignments that *justify* a faulty circuit output. In this approach, each faulty output is analyzed and internal signals in its transitive fan-in are inferred by means of logical implication. For each gate in the logical topological path from outputs to inputs, a search tree accounts for all possible consistent inputs assignments. For example, in order to justify a logic zero at the output of an 2-input OR-gate, a single input vector [0,0] needs to be considered. Conversely, to justify a logical one, three input vectors have to be included in the search tree ([0,1],[1,0] and[1,1]). The final outcome of this technique is a set of suspect lines, which cannot be proven to be fault-free. The same procedure has also been extended to account for multiple faults [Cox88].

Critical path tracing was first introduced as an efficient alternative to fault simulation [Abramovici83]. It traces the faulty circuit's outputs back to any possible locations in the chip but, contrary to logic reasoning, it considers the sensitized logic paths to the observed errors. Critical path tracing has been used for the diagnosis of delay faults, where a six-value logic algebra is employed to represent steady signal values, signal transitions and hazards [Girard92b, Girard92a]. This concept has been refined [Rousset07a] and extended for the diagnosis of bridging faults [Rousset07b,Wang06a] and crosstalk-induced delay faults [Mehta06] Additional timing information can also be considered to prune the final fault candidate list [Yang06].

Back-tracing approaches become less effective under the presence of multiple interacting faults. *Multiple fault masking and reinforcement effects* [Ye10] are observed when an output affected by a fault depends on the presence of other faults. This challenge can be addressed by means of fault simulation to validate justification decisions [Ye10]. Then, fault locations are scored to find a set of fault candidates that better characterizes all faulty patterns in the test set. Alternatively, the effects of multiple faults can be assessed by means of "X"-value propagation [Desineni06,Yu08,Yu10].

2.7. Summary and Outlook

This chapter described basic concepts for the testing and diagnosis of random logic circuits. The terminology used throughout this dissertation has been explained and the relevant building blocks of a generic test architecture have been detailed: Relevant standardized interfaces for test application have been covered and the most common pieces of test infrastructure for test pattern generation and test response compaction have been analyzed, where special attention have been devoted to linear feedback shift registers (LFSRs).

Similarly, the most prominent solutions for logic diagnosis have been presented, namely, the *inject-and-validate* and *back-tracing* approaches have been discussed.

In the following chapter, these concepts are further analyzed to account for related work pertaining to autonomous test and diagnostic solutions of complex systems. In the later chapters the ideas presented in this chapter are revisited to describe a novel diagnostic solution for the automotive domain.

3. State-Of-The-Art in In-Field Test and Diagnosis

This chapter presents state-of-the-art approaches to identify a faulty component in complex electronic systems. The first two sections cover diagnostic techniques to identify a faulty integrated circuit (IC) in the system, while the third section presents diagnostic techniques at lower abstraction levels for detailed failure analysis of semiconductor failures.

The first section deals with functional diagnosis, where operational characteristics of a system are annotated and analyzed in order to infer a faulty IC.

The second section describes structural diagnostic approaches. For this purpose, the benefits and shortcomings of *software-based self-test (SBST)* are analyzed in the context of complex embedded systems. Then, some previous research efforts on design-for-test (DfT) infrastructure reuse are surveyed, which enable the application of structural test with little intervention of external test equipment. Similarly, some prior works are accounted for, which apply scan-based tests during in-field system test and maintenance. These methods make use of available on-chip DfT structures in order to activate and conduct structural tests during system-level test and diagnosis.

Finally, the last section deals with logic diagnostic procedures for ICs equipped with built-in test features. These diagnostic techniques analyze compacted test responses in order to identify a faulty location in the IC and characterize the observed defect behavior.

3.1. System-Level Functional Diagnosis

Functional diagnosis of complex electronic systems has been a hot research topic in the last two decades. The main objective has been to identify the faulty *least replaceable unit* in the system. The work in [Fenton01] provides a comprehensive review of the most prominent methodologies in this field.

Early functional approaches rely on expert knowledge for a particular application domain in order to diagnose a complex system. Rule-based techniques [Rich91] represent the experience of skilled diagnosticians and formulate diagnostic rules in the form of "IF-THEN" statements. The observed faulty test responses or *symptoms* are then analyzed to identify a faulty IC. Similarly, fault decision trees [Roberts87] systematically document the diagnostic process: the symptoms at the top of the tree branch out a decision tree, that consists of other symptoms, decisions, actions and, finally, repair recommendations.

The main disadvantage of both rule-based approaches and fault decision trees is the acquisition and maintenance of the knowledge required for decision making, especially for large systems where the number of failure scenarios can be extremely high. In order to overcome this limitation, model-based diagnostic methods [Darwiche00] rely on a system model to characterize the functional behavior of the system. Detailed models have been used for the diagnosis of embedded systems [Isermann05]. However, complex models like, for example, that of a microprocessor, are difficult to generate and their simulation may require very large computational effort.

Other model-based approaches simplify the system model and deal with qualitative diagnostic information [Struss03, Manley02, O'Farrill05]. They make use of Bayesian reasoning to find the most likely faulty component that can be responsible for the observed symptoms [Manley02, Barford04, O'Farrill05]. However, such qualitative model-based reasoning has the disadvantage that the underlying failure mechanisms are usually unknown and are difficult to represent in the model. Moreover, *a priori* failure probabilities for Bayesian reasoning are hard to estimate. This limitation is circumvented in [Zhang10] by means of fault insertion [Eklow04, Huang07].

More recently, *machine learning* methods have been applied to the functional diagnosis of board failures [Zhang12, Ye12, Ye13]. These techniques make use of past diagnostic experience in order to train a classifier, which is able to identify a faulty IC in the board. For this purpose, supervised learning approaches like *artificial neural networks* [Zhang11] and *support vector machines* [Zhang12] have been used as learning algorithms. In [Ye13] the advantages of these two techniques are combined into a single classifier by means of *weighted majority voting*.

3.2. System-Level Structural Diagnosis

System-level structural diagnostic approaches apply structural tests to identify a faulty IC. Software-based approaches are described next, while scan-based approaches are summarized in the following sub-section.

3.2.1. Software-Based Self-Test (SBST)

SBST [Hellebrand96] makes use of the native instruction set of a processor in order to apply test patterns to any of its target component blocks. Although the test application process is fully functional, since the processor executes a regular piece of software to apply test patterns, the test generation process can target faults in any structural fault model. In contrast to functional tests, this makes it possible to uniquely identify a faulty component during system diagnosis.

SBST features many attractive properties for system-level test. Firstly, it does not require any additional dedicated hardware, which becomes extremely important in cost-sensitive markets. Secondly, SBST programs can be generated for low-power test [Zhou06]. Thirdly, as tests are applied in functional mode, the *overtesting* problem of traditional scan tests is effectively avoided [Yuan09]. Finally, SBST is well suited for online test application in the field [Paschalis05, Merentitis12, Eberl12].

Purely functional SBST strategies rely on fault simulation to assess and improve the quality of the generated test program. This process can be automated with the help of *evolutionary algorithms* [Corno04].

SBST structural strategies can be categorized into two groups, namely, *hierarchical methods* and *register transfer level (RTL) methods* [Psarakis10].

Hierarchical methods generate structural tests for each processor component independently. Some hierarchical methods make use of formal approaches, like *bounded* *model checking* [Gurumurthy06] or satisfiability-based *automatic test pattern generation (ATPG)* [Lingappan07]. Conversely, ATPG can be constrained to represent the functional state imposed by a partially specified test program [Chen03, Wen06]. Such a technique guarantees the produced test pattern can actually be applied by a synthesized test program.

RTL methods exploit functional information in the RTL description of a processor for the generation of a test program. The methods in [Psarakis06, Kranitis05] rely on a set of predefined tests for the most common processor blocks. Such tests are likely to achieve sufficient fault coverage at the gate level. Alternatively, the gathered information can be used to set up the constraints for ATPG [Krstic02]. The approach in [Prabhu12] uses both RT and gate level descriptions and targets hard-to-detect faults with bounded model checking.

The typical fault coverage achieved with SBST approaches, either hierarchical or RTL-based, ranges between 90% and 95% [Psarakis10].

SBST programs have been generated for safety-critical processor components, like the address calculation module [Bernardi12], and other logic blocks in the chip like on-chip caches [Di Carlo11] and system peripherals [Grosso12].

Finally, diagnostic approaches for SBST have been proposed in [Chen02] and [Bernardi08]. Both approaches generate a diagnostic test program consisting of many small tests, which is able to distinguish the highest number of fault pairs. While the procedure in [Chen02] involves manual test generation, the method in [Bernardi08] automates the transformation of existing test programs. Both techniques rely on fault dictionaries for diagnosis.

3.2.2. In-System Structural Tests

On-chip test infrastructure is extensively used during semiconductor testing. However, the access to test structures is usually prevented for any application outside of the manufacturing test process at the semiconductor company. These resources can be leveraged for self-test if the chip is designed so as to enable an on-chip processing element to gain access to available core test wrappers and manage the test application process [Kretzschmar04, Lee05, Galke06, Tuna07]. For this purpose, any available system functionality can be reused for structural testing. For example, the system bus can be used as a test access mechanism (TAM) to transfer test data from an external memory to a *micro-tester* [Lee05] or to a TAM controller [Tuna07]. These on-chip resources forward the received test data to logic cores under test, which are equipped with IEEE 1500 compliant wrappers. The micro-tester as well as the TAM controller are both a master and a slave in the system interconnection bus. The slave interface is used by an on-chip general purpose processor to configure and schedule the test of the core, while the master interface is used during test to fetch the necessary test input data, that is, a test program for the micro-tester or the raw test data in case of the TAM controller.

The authors in [Fang12] propose a hybrid structural and functional approach for the detection and reproduction of system failures. The so-called *functional scan tests* are repeatable scan tests which mimic the behavior of a circuit under functional test. For this purpose, additional DfT hardware resources are integrated into the chip, so that the generated test patterns produce similar circuit states to those the circuit produces during regular operation. Therefore, the number of “no trouble found (NTF)” occurrences can be greatly diminished.

Other approaches have studied the online application of structural tests. In [Li08] scan patterns are stored off-chip in low-cost non-volatile memories. An on-chip test controller selects one of the cores for online self-test and configures the test architecture for scan-test execution and evaluation. The work in [Bernardi11] describes a test architecture specifically designed to support the application of both regular manufacturing test and online structural tests. This test architecture is very well-suited for the online test of embedded memories. It consists of an *online test controller (OTC)* inserted between the test access port (TAP) controller and the test wrappers surrounding each core with self-test capabilities. The OTC is also connected to the system bus and can be accessed like any regular peripheral functional block. During online test, the main on-chip processor uses the system bus to configure the OTC. The OTC then activates the desired self-test via the corresponding TAM to the selected core. Test execution in this approach can be scheduled by means of a timer and exception handling.

Similarly, the work in [Dutta09] presents a built-in self-test (BIST) solution based on the STUMPS architecture to support non-destructive test sessions during regular chip operation. Several design constraints are observed during chip design in order to apply non-destructive scan tests, which are interleaved with functional applications.

BIST for System-Level Test

The reuse of memory and logic BIST for system-level test is discussed in [Pateras97]. The test strategy follows a “divide and conquer” approach, where each logic component is provided with the necessary DfT resources for BIST. The associated costs for the required infrastructure are becoming much less of an issue in VLSI systems, where embedded test is much more difficult with another test methodology. Some standardization efforts in this direction are currently underway [Conroy12].

The work in [Vo06] realizes these test concepts and provides a comprehensive structural test solution for a complex telecommunications system. The adopted DfT measures include an IEEE 1149.1 TAP controller and IEEE 1500 wrappers for test access, a custom clock control scheme for test application, CPU access to the TAP controller, scan chain organization, state dumps for diagnosis as well as logic and memory BIST.

More recently, system-level BIST is further extended to account for routing congestion and potential average and peak power problems [Qian09]. Additionally, other BIST features are presented to enable logic diagnosis by means of signature analysis, scan chain masking and single-chain diagnosis. These techniques are detailed in the next section.

3.3. Logic Diagnosis for BIST

Logic BIST diagnostic approaches for the STUMPS architecture analyze highly compacted failure information in the test responses. While for a pass/fail test outcome the complete test response data in the whole BIST session can be compacted into a single *signature*, the resulting signature is not sufficient to support logic diagnosis [McAnney87].

In order to overcome this limitation, logic BIST diagnostic approaches follow one of two possible strategies, namely, *indirect diagnosis* or *direct diagnosis*.

3.3.1. Indirect Diagnosis

In indirect diagnosis, the failing signatures are evaluated in order to compute the values captured by the scan elements in each of the failing patterns. Alternatively, the on-chip compaction infrastructure can be bypassed and the contents of the scan chains can be directly retrieved from the automated test equipment (ATE). After this step, any logic diagnostic algorithm for combinational circuits is applied. Several methods have been proposed to infer or download the values of the scan cells efficiently. In the following, the most prominent approaches are described.

Interval-based Methods

Diagnostic methods in this category partition the BIST session into shorter intervals and generate a signature for each interval [Savir98, Song99, Wohl02, Amyeen11, Jayalakshmi12]. The reasoning behind this approach is to find a few failing patterns to be further analyzed, which is difficult to do with a single signature for the whole test set. All of these techniques require that the BIST controller must be programmed to apply a specific test interval. For this reason special attention must be paid to ensure the initial state of the pattern generator as well as the total number of applied patterns can be configured by an external tester.

The technique described in [Song99] presents a procedure to identify the first failing pattern in the test set. The technique performs a binary search by adjusting the location and size of test intervals. While a binary search requires at most $log_2(|\mathsf{T}|)$ steps to identify a faulty pattern, the approach in [Wohl02] requires only two sessions to collect all diagnostic information in the test set. In this method, the BIST session is divided into small intervals with fixed size, e.g. 32 patterns, which are applied and evaluated by the tester. In the second step, only the failing sessions are repeated and the uncompacted test response is shifted out to the tester. This approach can reduce the time required for BIST diagnosis but usually requires larger tester memories. Finally, the diagnostic flow of [Amyeen11, Jayalakshmi12] uses a mixed approach with fixed windows of different sizes, where only the first patterns of a failing test interval are unloaded to the tester.

Masking-based Methods

Masking-based methods evaluate which faulty scan cells are responsible for a faulty signature. For this purpose, each time a BIST session is applied only a subset of scan elements are fed into the multiple-input signature register (MISR) [Wu99, Rajski99, Ghosh-Dastidar00, Bayraktaroglu02, Liu04]. The complete test set is applied in multiple BIST sessions and logic diagnosis proceeds without downloading any uncompacted test responses.

The technique presented in [Rajski99] makes use of a linear feedback shift register (LFSR) to realize *pseudo-random masking* of scan cells during a BIST session. This approach selects a subset of scan cells to be compacted, while other scan elements are effectively masked out. After each BIST session, the LFSR produces a new subset of scan elements to be compacted in the next test application run. The accuracy of the method depends on the number of test sessions, the number of scan cells and the number of scan cells where errors are observed. The main benefit of this approach is that it can be executed in a single ATE session or *pass*, as long as the ATE has enough memory for the final signatures of all the BIST sessions.

Deterministic masking [Ghosh-Dastidar00] can also be used when complete accuracy is preferred. In this approach a piece of partitioning logic is configured to select which subset of scan cells are allowed to reach the MISR. With this goal in mind, the scan cells are represented with a matrix, where each column represents a scan chain and each row a *scan slice*, that is, a set of scan cells which are shifted into the MISR in the same clock cycle. The subset of selected scan cells is given by three values, namely, X, Y and Z. X is a duple (x_0, x_1) that specifies the range of scan cells to select. Y and Z specify the bottom-most scan slice and the top-most scan slice counting up from the MISR, respectively. In order to find the scan elements producing errors, a binary search can be conducted. This search divides the scan cells into two groups of equal size and executes a BIST session. According to the test outcome, the size of the scan cell partition can be further refined to identify all failing scan cells. This technique is more effective than pseudo-random testing but requires interaction with the tester to configure the scan cell partitioning in each test session.

Other masking-based methods [Bayraktaroglu02, Liu04] make use of a fault model to identify fault candidates that produce a distribution of errors in the scan elements consistent with that of the faulty device.

3.3.2. Direct Diagnosis

Direct diagnosis employs diagnostic algorithms specially devised for the analysis of compacted test signatures in order to identify a faulty chip location. That is, these techniques infer potential fault candidates without sorting out the values of the erroneous scan cells. Approaches in this category usually realize an extension of the *inject-and-validate* paradigm described in section 2.6.1.

Signature-based Diagnosis

The diagnostic procedure of [Cheng07] identifies fault candidates given the observed faulty signatures and their failing patterns. The technique requires only two tester sessions: in the first session, all test responses are compacted into a single signature. The second session takes place only if the first session reveals a mismatch between the obtained and the expected signatures. In the second session, one signature is produced for each test pattern and shifted out into the tester. After the compaction of a test pattern, the MISR is reset before generating the following signature. Logic diagnosis is then performed using an extension of a compactor-independent direct diagnosis technique [Cheng04].

The diagnostic algorithm makes use of *error functions* to determine which faulty scan cells may be responsible for the observed faulty MISR signature. An error function maps each MISR bit s to a set of scan cells, which may cause an error on s. In order to compute error functions, the effects each single faulty scan cell has in the MISR are simulated during scan unload. In this process, the rest of the scan cells are assumed to be fault-free. The final state of the MISR is a so-called *error signature*, which represents the distribution of errors produced by a single faulty scan cell. This process is repeated for all the scan cells in the circuit. Finally, all the error signatures are superimposed so that all the scan cells that may cause an erroneous MISR bit can be identified.

After this step, the diagnostic procedure considers the intersection of all the error functions in order to reduce the set of scan cells to be analyzed further. Critical path back-tracing [Girard92b] is then employed to identify a set of candidate fault locations which reproduce the same error behavior calculated with the error functions. Finally, these candidate faults are simulated independently and the best candidates are selected, which are able to reproduce the largest number of observed faulty signatures.

The diagnostic results in signature-based diagnosis are very similar to those achieved without compaction at all. Signature based diagnosis incurs in a reduction of diagnostic accuracy for stuck-at faults of about 2-3%. Interestingly enough, the size of the MISR has only marginal impact on the diagnostic outcome.

XP-SISR

The *extreme parity compactor with SISR (XP-SISR)* [Elm10] is a test architecture for built-in self-diagnosis (BISD). It extends the STUMPS architecture to generate and store very small test signatures which are used for logic diagnosis later on.

Response compaction is achieved in two steps. The first stage employs extreme space compaction [Holst09a], while the second stage includes time compaction by means of a serial-input signature register (SISR). Extreme response compaction makes use of a parity tree to compact the outputs of the scan chains in the same scan slice into a single bit. Therefore, special procedures for test pattern generation need to be applied in order to ensure errors in multiple scan chains do not cancel one another out. *Pattern stripping* [Miyase04, Kochte08] and *repeated encoding* [Kochte09] are employed to overcome this issue, with negligible fault coverage loss.

Time compaction is performed independently for each test pattern. The SISR compacts the stream of parity bits generated by the space compactor into a very short signature. The size of the SISR is selected so that every faulty bit in the parity stream generates a different test signature.

The diagnostic flow in this approach is as follows: A SISR signature is generated for each test pattern and compared to the expected fault-free signature stored in a *response memory*. If the two signatures differ, the obtained signature is stored in the *fail memory* together with a pattern index to identify the failing pattern. The analysis of the signature can then proceed after downloading the contents of the *fail memory*. Logic diagnosis with the gathered signatures uses an extension of the diagnostic algorithm of [Holst09b].

The XP-SISR architecture achieves the same diagnostic outcome as that obtained without any response compaction. This approach enabled for the first time the autonomous collection and storage of diagnostic signatures without repeated test sessions.

3.4. Summary and Outlook

Several system-level test and diagnostic solutions have been proposed in the research community to evaluate the integrity of complex embedded systems. The approaches presented in this chapter have been devised with several different purposes in mind and none of them can handle by itself the diagnostic requirements of the automotive domain.

Functional test approaches may identify a faulty component. However, if successful, they only support a diagnostic outcome at system-level, since the faulty location in the identified faulty IC cannot be accurately determined.

Software-based structural solutions are tailored to target specific structural faults and can be seamlessly integrated into the operation of an assembled system. However, the generation of a suitable test program cannot be easily generalized for any random logic circuit. Additionally, the diagnostic capabilities of such a software-based approach have not yet been proven effective for complex failure mechanisms.

The application of scan-based tests can reuse well-known diagnostic techniques envisioned for manufacturing test. Unfortunately, these techniques require specialized tester equipment, which is not available during most of the failure analysis process in the automotive domain. Moreover, the access mechanism and the control of the on-chip test infrastructure remains a challenge in deeply embedded systems.

Finally, the most attractive diagnostic solution available in the literature is the XP-SISR architecture. However, this approach requires dedicated test infrastructure and pattern generation procedures, which are not compatible with the traditional self-test architectures widely used today.

In the rest of this dissertation efficient structural diagnostic methods are presented to address the specific requirements of the automotive domain. As the developed diagnostic solutions are fully compatible with the traditional STUMPS architecture, they support the failure analysis process during a vehicle's complete lifetime. Additionally, they only require highly compacted test signatures in order to provide accurate logic diagnosis results.

4. Efficient Access to On-Chip Test Infrastructure

The execution of structural tests during system-level diagnosis can be achieved by reusing test procedures traditionally applied during manufacturing test. This strategy requires access to the relevant design-for-test (DfT) infrastructure on-chip. While built-in self-test (BIST) methods can greatly simplify this task, they still need an activation command to initiate test application. Similarly, scan-based tests controlled by a functional component in the system can also be exploited. In any case, the adoption of semiconductor test solutions for system diagnosis is not straightforward, as these tests require external equipment, which may not be available, as well as physical access to certain pins, which may not be accessible after system assembly.

In this chapter a conceptual test solution is presented for the system-level access to on-chip test infrastructure [Cook12c]. This approach complements available on-chip architectures for core-based test (section 3.2.2) and provides a reuse methodology for non-intrusive system test under functional and cost constraints. At the end of this chapter a case study is presented for an industrial electronic control unit (ECU).

4.1. Non-Intrusive Automotive Structural Test

The autonomous application of system-level structural tests needs to manage the execution of one or several tests at a time. This is achieved by leveraging the functionality of the enclosing system in order to mimic the manufacturing test environment and provide a suitable test harness, which can be used to isolate the device-under-test, apply a suitable test procedure and store the obtained test responses.

4.1.1. Test Access

Before any structural tests are applied, the state of the system must be switched from an operational mode to a hybrid mode where some components remain fully functional while others are driven into *test mode* and do not conform to the functional specification of the system. In this mode, the capabilities provided by the target device-under-diagnosis (DUD) and (most likely) by the components comprising the test harness are no longer available.

In order to gain access to the DfT resources of the DUD, a known fault-free unit or *test controller* may employ the DUD's available test interface. This goal can be achieved without a major cost penalty if an available functional interface in the test controller is exploited for the application of structural tests. For example, if the test controller is a programmable unit, i.e. a microcontroller, the test interface can be driven by means of its input/output (IO) subsystem. Figure 4.1 compares the test setup for manufacturing test with that of in-system test. In Figure 4.1.a the DfT infrastructure of the target DUD is accessed directly from the automated test equipment (ATE), while in Figure 4.1.b this is achieved with a test interface to the test controller alongside any other functional interface in the system.

The cost of the additional test interface on the board may have a great impact on the overall hardware cost of the system. In order to decrease the cost of a suitable system-level solution, the test interface and one of the functional interfaces of the DUD can be merged into a single interface between the test controller and the DUD. This is shown in Figure 4.1.c. Such hardware modification reduces the overall system cost but requires some additional considerations during the design of the DUD.

The fault-free operation of the test controller can be verified in the absence of other programmable devices by means of functional techniques like software-based self-test (SBST) (section 3.2.1).

4.1.2. Test Application Process

In the following, the general procedure for the in-system application of structural tests is explained.

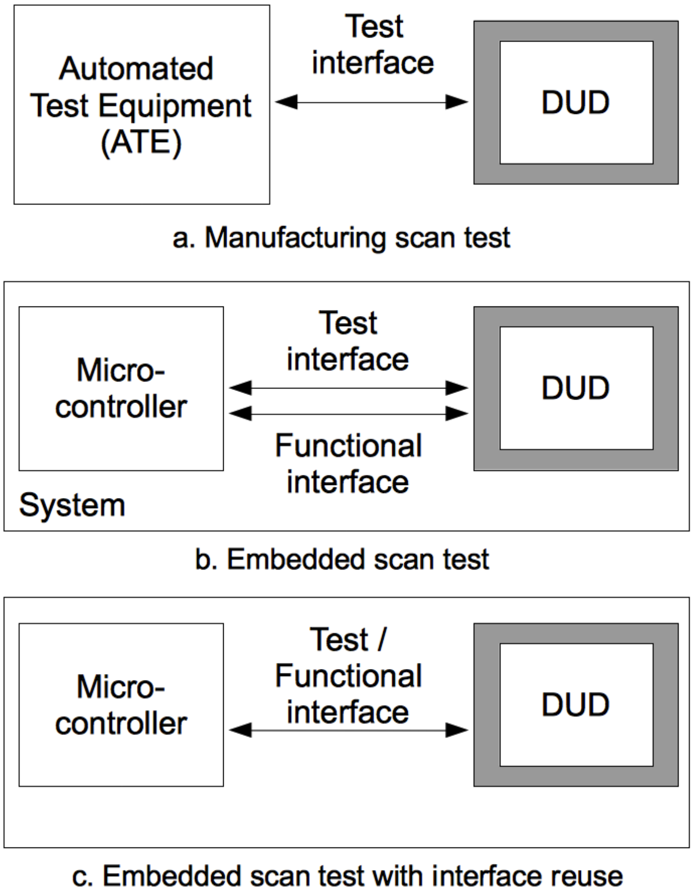


Figure 4.1.: Structural test setup for manufacturing and field test

Input Data Transfer

The test controller gathers test input data from the *test source*. Depending on the nature of the target test, the test input data can range from a simple activation command (logic and memory BIST) to a fully-specified test pattern set. The *test source* may reside locally in a memory located on the same board as the test controller, so that the latter has direct access to it, or can eventually be transmitted by means of a system bus like the controller area network (CAN) bus or FlexRay.

DUD Access

After the test controller has driven the DUD into test mode, a stream of data has to be downloaded into the chip in order to select a given structural test. For autonomous tests (BIST) an activation command may be the only piece of information necessary for test application. If scan tests are activated, however, a special operation mode is entered that enables to shift the scan data in and the test responses out. The same DUD may feature several scan configurations in order to support different ATE's with varying capabilities.

Test Application

This step is required if the test controller is responsible for controlling the test application process. Scan patterns are applied and test responses are evaluated. The test controller needs to shift one test pattern in, apply one capture clock cycle and shift the test responses out. If delay faults are targeted, two or more at-speed clock cycles must be applied before collecting the test outputs. The evaluation of the test may proceed during scan-out or, if test response compaction is used, after a given number of patterns.

Response Data Transfer

Similar to the test input data, the test outcome needs to be stored in the *test sink* for later retrieval and failure analysis. The *test sink* can be located either in an internal memory on-chip or externally in any non-volatile accessible memory. After this step, the DUD may be restarted and hence driven back to functional mode.

4.2. Case Study: Workshop Test

The feasibility of an automotive structural test solution is demonstrated on an industrial ECU designed by Robert Bosch GmbH [Ull11]. The main objective of this case study is to evaluate the capabilities of a structural test solution for workshop test under realistic industrial constraints. The application of structural tests is realized by means of extensive reuse of the system's functional capabilities and available interfaces to reach the DfT structures of the target application-specific integrated circuit (ASIC).

In this case study, a scan test pattern set for an ASIC has been reused for system-level diagnosis. The pattern test set, although generated for manufacturing test, is applied without removing the target ASIC from the assembled system. The test application process has been controlled with a portable computer connected to a functional bus (CAN). The architecture of the presented test application fulfills the requirements of the automotive diagnostic use-case *workshop test*.

4.2.1. ECU Architecture

The ECU architecture is shown in Figure 4.2. It consists of one main controller (32 bit, 150 MHz) and several ASICs (A-H). The main controller has an integrated CAN-Bus interface for communication with any other node in the CAN network. This network allows to upload temporary firmware into the controller's volatile memory and to manage its execution.

The ASICs of the ECU share a multiplexed serial peripheral interface (SPI) bus with the main controller as bus master. They are addressed by means of a chip-select (CS) signal together with a device address encoded in each SPI request frame. The serial input (SI) and serial output (SO) pins are used for data transfers.

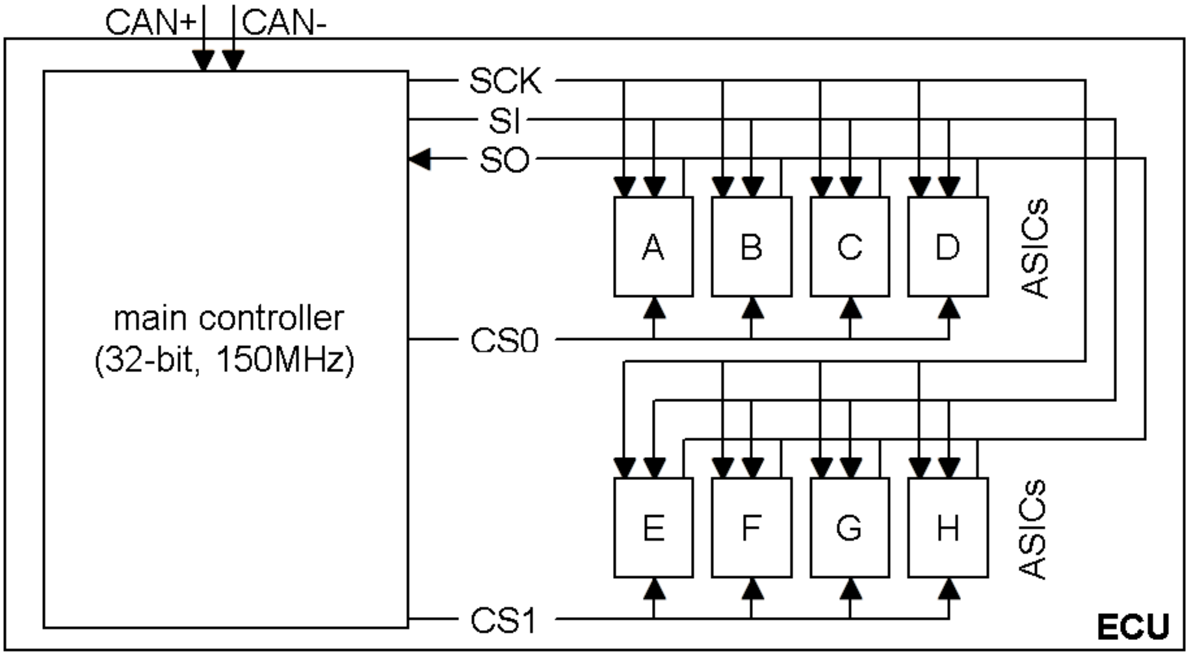


Figure 4.2.: Typical automotive ECU architecture [Ull11]

An IEEE 1149.1 compliant test access port (TAP) is present on the chip for the management of test application processes. For system-level application of structural tests, access to this test interface is mandatory. An inexpensive interface was developed as described in section 4.1.1. The JTAG test interface is combined with the functional SPI interface, thus named JTAG@SPI [Poinstingl]. The functional SPI port can be used as the first communication channel to activate the JTAG interface of ASICs. On the hardware side this requires an additional multiplexer to switch the incoming bus signals either into the TAP or the internal serial interface (Figure 4.3). For the control of this multiplexer a new internal test mode is introduced. A specific sequence of SPI commands activates the internal test mode thus switching over to the JTAG protocol. The intuitive signal mapping of SPI and JTAG protocols is shown in Table 4.1.

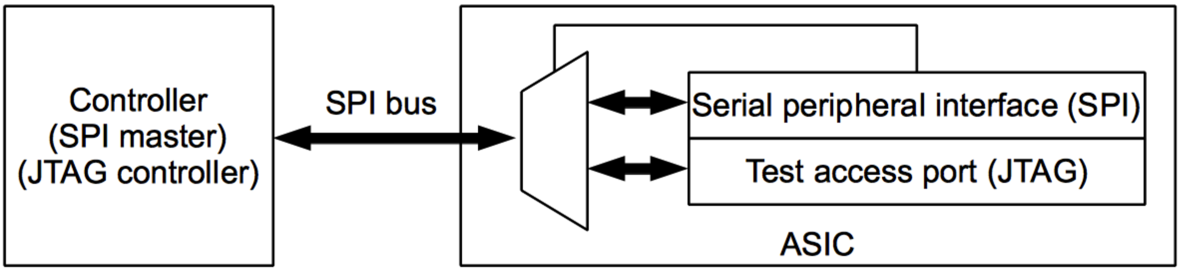


Figure 4.3.: JTAG@SPI: multiplexing test (JTAG) and functional (SPI) Protocol

JTAG signals	SPI signals
Test Clock (TCK)	Serial Clock (SCK)
Test Mode Select (TMS)	Chip Select (CS)
Test Data Input (TDI)	Serial Input (SI)
Test Data Output (TDO)	Serial Output (SO)

Table 4.1.: Signal mapping for interface reuse (JTAG@SPI)

4.2.2. Manufacturing Scan-Test

The manufacturing test of the target ASIC ensures high product quality in mass production. Numerous digital and analog, functional and structural test applications are conducted under different operating conditions, assisted by complex semiconductor testers. The tester is directly connected to the DUD, supplying it with the appropriate power sources and driving its JTAG test interface. Supported test methods in the test specification include scan-tests for stuck-at and transition delay faults as well as programmable BIST methods for internal memory blocks.

The manufacturing test setup is shown in Figure 4.4. The tester sets the test input signals {*CS0*, *SCK*, *SI*, *RST*} and evaluates the test outputs {*SO*, *PWMEn*, *IRQ*} on every clock pulse on the *ECK* pin, which is the test clock.

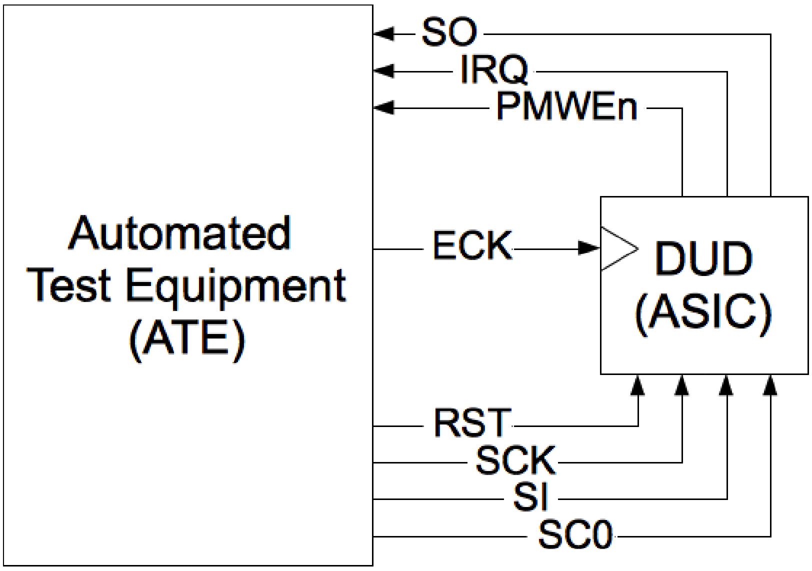


Figure 4.4.: ASIC manufacturing scan-test setup

4.2.3. Implementation Setup

The employed test setup is shown in Figure 4.5. For clarity, the rest of the ASICs are not shown. The setup makes use of the Keyword Protocol 2000 (KWP 2000) protocol over CAN [ISOa, ISOb] to exchange data between the tester and the ECUs. A laptop is employed as the client device, although any other low cost tester may be used for this purpose. The laptop serves as both *test source* and *test sink*. The test data is transferred over the CAN-Bus. This communication channel is used to upload the main controller's test software, to execute it, and to transmit test input data and test responses between the controller and laptop.

4.2.4. Test Application Process

- *Input test data transfer*: At first, the ECU's main controller has to be loaded with the test firmware and scan test data, which is provided by the tester via the CAN-Bus interface. The ECU does not hold enough volatile memory to buffer all input

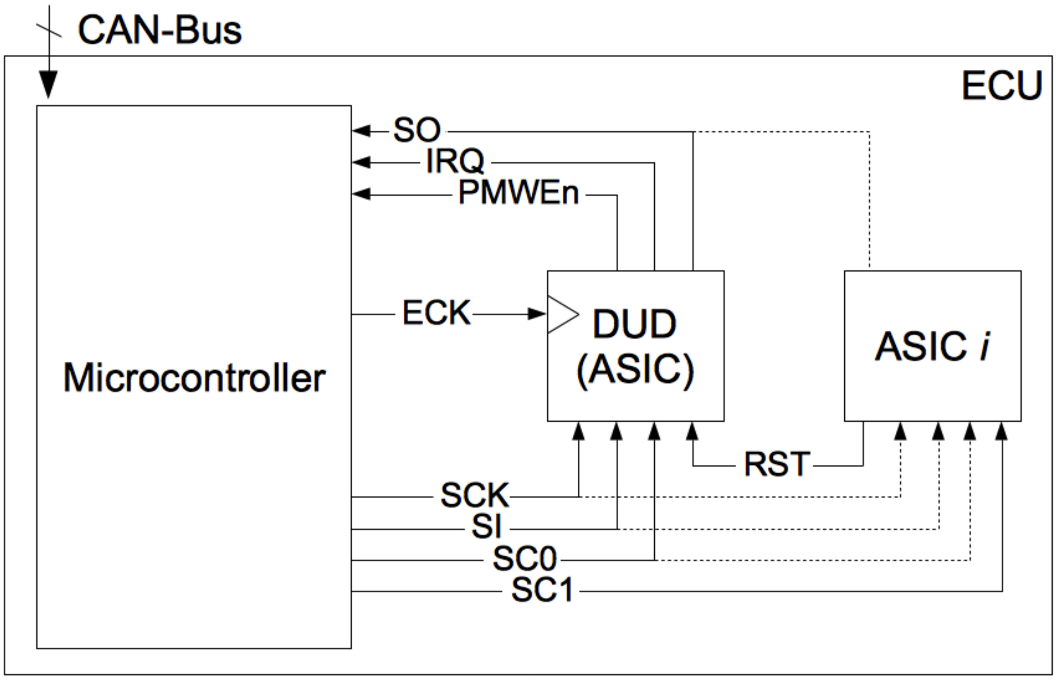


Figure 4.5.: Embedded scan test implementation

test data at once. Therefore, the test application is partitioned into sessions which are executed by means of several request messages. The provided test data from ASIC manufacturing contains approximately 150,000 test vectors and is able to detect 97% of all stuck-at faults. The test vectors contain the necessary values for the input signals as well as the expected values of the output signals.

- *DUD access*: the ASIC is brought into test mode by executing appropriate SPI commands received from the main controller. For scan tests, the test interface activation by itself is not sufficient. Further JTAG instructions are used to prepare the DUD for test by setting other internal modes and clock selection registers.
- *Test application*: Multiple request messages are transmitted over the CAN interface to the main controller in order to apply the complete test set. The test controller, in turn, sets the test inputs accordingly before pulsing the test clock signal and comparing the test outputs with the expected values. As shown in Figure 4.5, the *RST* signal is not directly controllable by the main microcontroller. In order to set this signal during test application, the main controller sends a special SPI frame to one of the other ASICs in the ECU, which causes the *RST* signal to toggle.

- *Response test data transfer*: The main controller keeps track of the test result by monitoring the test outputs. If a test output does not match its expected value, the vector index and the states of the output signals are sent to the laptop in a response message. If no error occurs, the request message is simply acknowledged. Therefore, the presented embedded scan-test solution offers the same diagnostic information as ATE-driven manufacturing test. Therefore, it enables logic diagnosis with both *inject-and-validate* and *back-tracing* approaches.

4.2.5. Results

Table 4.2 describes the performance of the test setup. As the same test pattern set is used, the fault coverage is equal for both manufacturing test and embedded system test. As the table shows the test time is much larger for embedded system test. This is due to specific architectural constraints of the ECU in consideration: As described above, the *RST* signal is controlled indirectly by means of an SPI frame. Therefore, each time a the *RST* signal needs to be toggled to produce an input pattern in the test specification, the test application process needs to be stalled for at least the duration of the SPI frame. As a result, this communication overhead greatly delays test execution. For relevant implementation details on the embedded scan test procedure, the reader is referred to [Ull11]. In the general case, however, this timing overhead can be easily avoided if the execution of embedded scan tests is considered during the design of the ECU and all necessary test pins pins are made accesible to the main controller.

Application	Test time	# of test patterns	Fault coverage
Manufacturing test	0.03 s	150,000	97% stuck-at
Embedded system test	132 s	150,000	97% stuck-at

Table 4.2.: Case study performance evaluation

4.3. Summary and Outlook

This chapter presented a conceptual test procedure for the execution and evaluation of structural tests in complex embedded systems. In the presented test strategy a test controller gains access to the on-chip DfT infrastructure of a chip and manages test execution. For this purpose, it fetches test input data from a *test source*, activates a given test and stores the test results in a *test sink*.

The presented case study shows how structural test techniques originally devised for manufacturing test can be applied during system-level diagnosis in the workshop. In the presented solution an existing test pattern set with 97% fault coverage is applied with the help of an inexpensive portable computer. The test outcome can be used to correctly identify a faulty integrated circuit (IC) in the system and to speed up semiconductor failure analysis in the automotive industry.

5. Logic Diagnosis with Compacted Test Responses

Logic diagnosis for built-in self-test (BIST) enables the cost-effective analysis of semiconductor problems affecting the manufacturing process. This chapter presents a novel direct diagnostic algorithm targeting arbitrary faults in logic cores, which analyzes highly compacted response data usually collected during BIST. This new approach makes use of the linear properties of on-chip response compactors to evaluate test *signatures* and identify one or multiple faulty locations in the structure of the device [Cook11a, Cook14]. Contrary to state-of-the art approches, the presented algorithms support superior test compaction ratios without any restriction on the test infrastructure of the chid. Additionally, highly compacted signatures are utilized for the first time to diagnose multiple faults.

5.1. Logic Diagnosis and Failure Analysis

Logic diagnosis is a fundamental step in the semiconductor failure analysis process. It analyzes faulty test responses collected from a device and estimates the most likely locations in the chip affected by one or several faults. Figure 5.10 shows the role logic diagnosis plays within a generic failure analysis flow for BIST. Test patterns are first applied to a device-under-diagnosis (DUD) and test responses are compacted into *signatures*. Although it is possible to compact the whole test pattern set into a single signature, this signature would not provide enough diagnostic information. Therefore, several intermediate test signatures are usually generated.

During logic diagnosis fault candidates are evaluated by means of fault simulation. Each candidate produces a set of faulty intermediate signatures, which are compared with the obtained signatures from the DUD. Due to response compaction, the evalu-

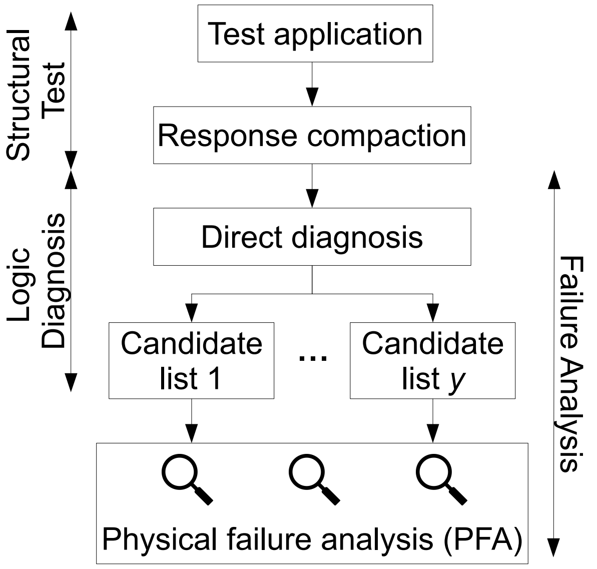


Figure 5.1.: Diagnostic flow and failure analysis

ation of the diagnostic information captured in the faulty signature cannot be easily performed by state-of-the-art diagnostic algorithms. To achieve this goal, an effective and efficient methodology for direct diagnosis is explained in the following sections.

Direct diagnosis generates one or several fault candidate lists. If the chip is affected by a single fault, a single candidate list is sufficient to characterize the error observations. However, in the presence of multiple faults, additional lists are beneficial to account for each failure mechanism, which needs to be considered during later stages of the failure analysis process. Each list contains fault candidates, that are ranked according to the degree to which they are able to reproduce a given subset of observed erroneous signatures.

After candidate ranking, a handful of candidates is selected from each fault list and analyzed further during physical failure analysis (PFA). During this step, the locations of the candidates are inspected so that defects in the chip can be discovered. PFA is a time-consuming process that cannot be easily automated. Therefore, the performance of logic diagnosis algorithms becomes extremely important in order to optimize the search for defective chip areas.

5.2. Direct Diagnosis of Arbitrary Faults with Test Signatures

Techniques for logic diagnosis with compacted test responses usually follow the *inject-and-validate* approach since *back-tracing* algorithms become less effective [Holst12]. State-of-the-art solutions for signature-based diagnosis have so far relied on a fault model to calculate faulty signatures and compare them with the obtained signatures from the DUD [Cheng07, Elm10]. Unfortunately, these approaches are only feasible for defect mechanisms which manifest themselves exactly like one of the faults in the assumed fault model. As a consequence, a large number of signatures may not provide any useful diagnostic information. Therefore, these direct diagnostic methods yield sub-optimal results for arbitrary failures.

In order to analyze arbitrary defect mechanisms, the *conditional stuck-at* fault model is employed [Holst09a]. The goal of the presented direct diagnostic methodology is to analyze the activation conditions of one or more faults which may produce a faulty signature that exactly matches that obtained from the DUD. This approach takes full advantage of any diagnostic information conveyed in the observed signatures, which becomes especially advantageous for BIST, as the diagnostic properties of pseudo-random or pseudo-exhaustive test sets can be fully exploited.

The algorithms presented in this chapter avoid the exponential computational effort to simulate every possible set of activation conditions in the fault set. Instead, the linear properties of a multiple-input signature register (MISR) compactor are utilized to analyze the combination of a fixed number of faulty responses obtained from the simulation of single stuck-at faults. These responses are superimposed and compared to the observed faulty signatures recovered from the DUD.

A conditional stuck-at fault is said to *explain* an observed signature if it is able to reproduce the exact same signature with a given allocation of activation conditions. The number of signatures a fault is able to explain is the major performance metric for the fault's ranking.

As chips may be affected by several defects [Stapper80, Koren98], a fault may explain a signature either by itself or in combination with other faults. Therefore, depending on the underlying failure mechanism, the procedure to explain erroneous signatures may require the consideration of multiple conditional stuck-at faults, which could be activated arbitrarily in the test patterns comprising the observed signature. This poses a tradeoff in terms of the accuracy of the diagnostic outcome and the amount of information necessary for diagnosis. On the one hand, multiple patterns can be compacted into a single signature under the assumption that only a single fault is active. As a consequence, maximum test compaction ratio is achieved. On the other hand, on-chip test response compaction can be adjusted to generate a test signature for every pattern in the test set. Thus, a larger number of active faults can be diagnosed.

In the next section a procedure to generate fault candidate lists is detailed, which is suitable for any number of faults and any response compaction scheme. Then, logic diagnosis for the two mentioned corner cases is covered in detail. The straightforward combination of these two methodologies can account for any desired compromise between response compaction ratio and the maximum number of faults in the DUD.

5.3. Generation of Fault Candidate Lists

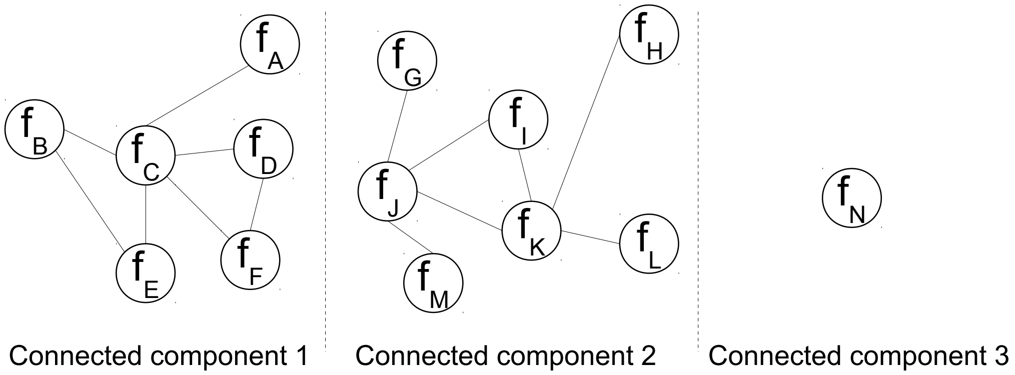
In order to account for all the failure mechanisms distinguished by the pattern set T, the fault set F is partitioned into several candidate lists according to the following heuristics:

- Two faults belong to the same candidate list if they explain at least one common signature
- A given fault can only belong to a single candidate list

To identify candidate lists, faults are modeled with an undirected graph $G = (N, L)$, where the set of nodes N contains the candidate faults which explain at least one signature. Therefore, N is a subset of the fault universe F. For each candidate fault $f \in N$ let $X(f)$ be the set of signatures $\widetilde{S}$ that f explains. As detailed later in this chapter, the set $X(f)$ is updated as faulty signatures are explained. The edge set L contains an edge $l = (f_i, f_j)$ if f_i and f_j explain at least one common signatures, i. e. $X(f_i) \cap X(f_j) \neq \emptyset$.

With this graph representation the candidate lists can be efficiently identified. As shown in Figure 5.2, each connected component of G defines a fault candidate list.

In order to evaluate candidate faults within a list, faults are ranked according to their *evidence*. Faults with higher rank are considered the most likely culprits of the defective



Candidate fault list 1: [f_A, f_B, f_C, f_D, f_E, f_F]
Candidate fault list 2: [f_G, f_H, f_I, f_J, f_K, f_L, f_M]
Candidate fault list 3: [f_N]

Figure 5.2.: Connected components and fault candidate lists

behavior. These faults are analyzed further during PFA.

The evidence of fault f is a tuple $evidence(f) = (\sigma(f), \iota(f))$ where $\sigma, \iota \in \mathbb{N}_0$. Parameter $\sigma(f)$ quantifies how many signatures f is able to explain, while parameter $\iota(f)$ describes how often f can be detected by the test pattern set.

Consequently, the $\sigma(f)$ parameter is defined as:

$$\sigma(f) = |X(f)| \tag{5.1}$$

The parameter $\iota(f)$ quantifies the number of patterns $p_i \in \mathsf{T}$ that detect f:

$$\iota(f) := |\{p_i \,|\, p_i \in \mathsf{T} \text{ detects } f\}| \tag{5.2}$$

In order to assess the likelihood of a fault, the following heuristics are employed: Candidate faults f_i in the same list are ordered according to decreasing values of $\sigma(f_i)$ and each fault f_i is assigned a rank, which is its position in the resulting ordered list. If $\sigma(f_1) = \sigma(f_2)$ for any two faults f_1 and f_2, they are ranked by increasing values of $\iota(f_i)$.

The described ranking procedure can be formalized in the following way:

$$rank(f_1) > rank(f_2) \iff \begin{cases} \sigma(f_1) > \sigma(f_2) \text{ or} \\ \sigma(f_1) = \sigma(f_2) \wedge \iota(f_1) < \iota(f_2) \end{cases} \tag{5.3}$$

5.4. Logic Diagnosis for Stringent Storage and Tester Bandwidth Requirements

In this section a diagnostic procedure is detailed, which drastically reduces the size of diagnostic information necessary to locate a single fault in the device-under-diagnosis.

With limited tester storage and bandwidth, the number of intermediate signatures has to be minimized. Therefore, a subsequence of pattern responses from the test set is compacted into a single signature. As a result, only the aggregated effect of several patterns is available for diagnostic purposes.

To enable diagnosis on such highly compacted test responses, a preprocessing step is necessary to extract diagnostic information from each pattern comprising the signature. The main problem at hand is then to identify, for every candidate fault in the fault model, a sequence of likely faulty and fault-free test responses, whose combined effect matches the observed faulty signature.

For each test pattern subsequence, the faulty behavior is analyzed without restricting assumptions as long as it is caused by a single defective location. In particular, faults with unknown activation conditions are covered as well. Finally, the analysis of all subsequences produces a single candidate list that accounts for a single culprit.

5.4.1. Diagnosis with Intermediate Signatures

For the sake of simplicity, assume a signature-based compaction scheme as shown in Figure 5.3. Note that, in case the number of scan chains exceeds the size of the MISR, this architecture may be extended to include any linear space compactor. Assume further that m is the maximum length of the scan chains in the STUMPS scheme. Therefore, the responses of each test pattern are compacted in m clock cycles.

Let n be less than or equal to the length of the MISR. In order to reduce the number of intermediate signatures, the set of test patterns T is partitioned into $h = \lceil \frac{|\mathsf{T}|}{n} \rceil$ blocks B. Each block contains n patterns at most. The patterns of each block are compacted into a single signature.

The state transition function of the MISR is $L \in \{T_{Standard}, T_{Modular}\}$ as described in section 2.5.1.

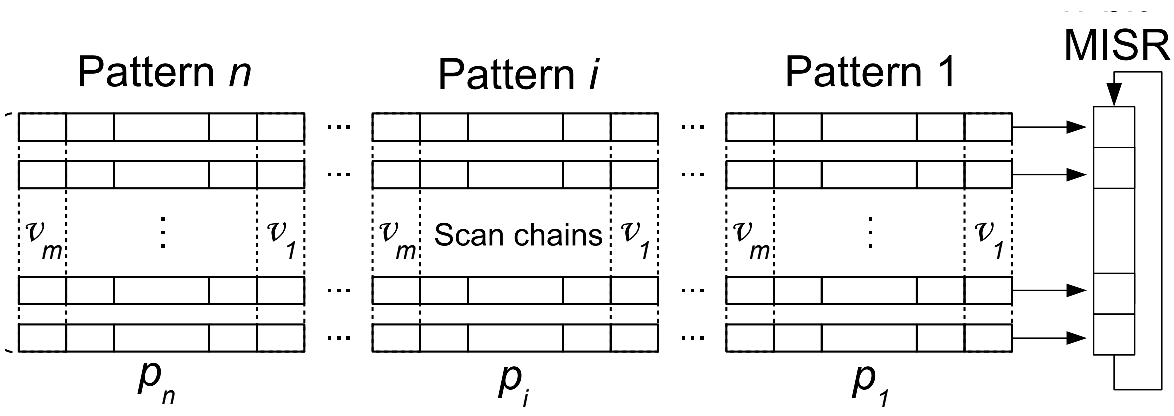


Figure 5.3.: Signature-based compaction scheme

The matrix $H = L^m$ describes the autonomous function of the MISR after *m* cycles. Each block of test patterns $p_1, p_2, \cdots, p_n$ is represented by the list:

$$B = (p_1, \cdots, p_n) \tag{5.4}$$

Assume the DUD's fault-free output response for pattern p_i is $V := [v_1, v_2, \cdots, v_m]$, where each column vector v_t represents the outputs of the n scan chains in scan slice t. If the MISR is initialized with the all-zero state, the corresponding fault-free signature S_i is described by the equation:

$$S_i := \sum_{k=1}^{m} L^{\,m-k}\, v_k \tag{5.5}$$

Each pattern block B results in a fault-free signature S_B. Applying linear superposition, the final signature after applying all patterns in *B* is captured by the equation:

$$S_B = \sum_{i=1}^{n} H^{\,n-i}\, S_i \tag{5.6}$$

After each block *B*, the MISR is reset to the all-zero state. S_B can be transferred into the chip during the test application process for immediate comparison with the obtained signatures. Alternatively, the obtained signature can be downloaded to the tester where it is compared to S_B.

Block *B* may contain some test patterns that detect the fault *stuck-at-0-x*, that is, a stuck-at-0 on location *x* (or *stuck-at-1-x*, a stuck-at-1 on *x*) and, according to the conditional stuck-at (CSA) fault model, depending on a given condition *cond*, these patterns may also detect *stuck-at-0-x*$[cond]$ (or *stuck-at-1-x*$[cond]$). Hence, the signatures for the *unconditional* faults *stuck-at-0-x* (or *stuck-at-1-x*) have to be determined and a pattern sequence has to be selected which fits *stuck-at-0-x*$[cond]$ (or *stuck-at-1-x*$[cond]$).

Let *f* be such an *unconditional* stuck-at fault, and let S_i^f be the signature of pattern p_i in the presence of *f* if the MISR starts in the all-zero state. Error vectors are defined as:

$$E_i^f := S_i \oplus S_i^f \tag{5.7}$$

Note that $|E_i^f| \neq 0$ if and only if pattern p_i detects *f* in its signature.

Now assume, the real fault $\widetilde{f}$ is a conditional stuck-at fault. In this case, either $E_i^{\widetilde{f}} = E_i^f$, if in pattern p_i the activation condition is true, or $|E_i^{\widetilde{f}}| = 0$ otherwise.

The activation conditions $c_1, c_2, \cdots, c_n \in \{0, 1\}$ that describe the the fault's activation for consecutive patterns $p_1, p_2, \cdots, p_n$ are defined as:

$$c_i = \begin{cases} 1 \text{ if } E_i^{\widetilde{f}} = E_i^f \\ 0 \text{ otherwise} \end{cases} \tag{5.8}$$

The activation conditions are found by solving a system of linear equations. Let

$$r_i^f := H^{n-i} E_i^f \tag{5.9}$$

Now, variables $c_1, c_2, \cdots, c_{n-1}, c_n$ have to be determined so that:

$$\begin{bmatrix} r_1^f & r_2^f & \cdots & r_{n-1}^f & r_n^f \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n-1} \\ c_n \end{bmatrix} = S_B \oplus \widetilde{S}_B \tag{5.10}$$

The matrix $[r_1^f \; r_2^f \; \cdots \; r_n^f]$ can be precomputed for any *unconditional* stuck-at fault *f*, the correct signature S_B after pattern block *B* can be precomputed as well, and $\widetilde{S}_B$ is the observed faulty signature. Hence, equation (5.10) contains at least *n* equations with *n* unknowns. If equation (5.10) is solvable for a conditional stuck-at fault *f*, $[c_1, c_2, \cdots, c_{n-1}, c_n]$ describes the activation conditions for the patterns $[p_1, p_2, \cdots, p_n]$, otherwise the fault location of *f* cannot be the single culprit.

The presented approach only requires the solution of a system of linear equations after the fault simulation step usually employed for logic diagnosis. These calculations may be performed as soon as faulty test signaures are collected from the device and transferred to the semiconductor manufacturer, where logic diagnosis is conducted and faulty chip locations are analyzed.

Example

Figure 5.4 shows a piece of circuitry the response of which is compacted by a four-bit MISR with generator polynomial $x^4 + x^3 + 1$.

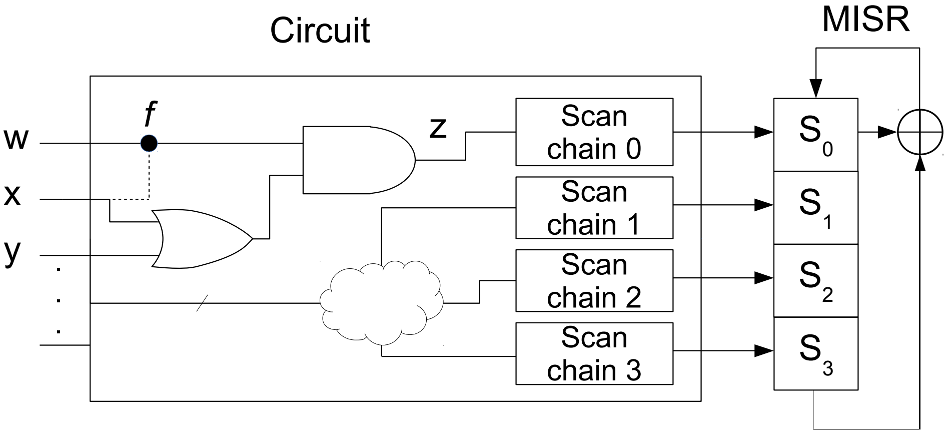


Figure 5.4.: Diagnostic example for limited tester bandwidth and storage

Assume there is a wired-and fault between signal lines *w* and *x*. This failure behavior can be modeled as the conditional stuck-at fault $\widetilde{f}$ = *stuck-at-0-w*$[x = 0]$. The signature block has a size of four patterns and the observed response from the fail memory equals $\widetilde{S}_B = [0\ 0\ 0\ 1]^t$.

The patterns in this pattern block are:

$$B = (p_4, p_3, p_2, p_1) = \left(\begin{bmatrix} 1 \\ 1 \\ 0 \\ \vdots \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 1 \\ \vdots \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 1 \\ \vdots \end{bmatrix} \right)$$

For the sake of simplicity, consider only the candidate fault f = *stuck-at-0-w*. The signatures S_i and S_i^f are found by simulating the fault-free and faulty circuit response.

$$S_1 = \begin{bmatrix}1\\1\\0\\0\end{bmatrix} S_2 = \begin{bmatrix}1\\0\\1\\1\end{bmatrix} S_3 = \begin{bmatrix}0\\0\\0\\1\end{bmatrix} S_4 = \begin{bmatrix}1\\0\\1\\0\end{bmatrix}$$

$$S_1^f = \begin{bmatrix}0\\0\\1\\0\end{bmatrix} S_2^f = \begin{bmatrix}1\\0\\0\\1\end{bmatrix} S_3^f = \begin{bmatrix}0\\1\\0\\0\end{bmatrix} S_4^f = \begin{bmatrix}0\\1\\1\\0\end{bmatrix}$$

For simplification, assume each scan chain contains a single scan cell. therefore, $m = 1$ and

$$L = H = \begin{bmatrix}0 & 1 & 0 & 0\\0 & 0 & 1 & 0\\0 & 0 & 0 & 1\\1 & 0 & 0 & 1\end{bmatrix} \tag{5.11}$$

By applying equations (5.7) and (5.6) error signatures and the fault-free block signature are calculated, respectively.

$$E_1^f = \begin{bmatrix}1\\1\\1\\0\end{bmatrix} E_2^f = \begin{bmatrix}0\\0\\1\\0\end{bmatrix} E_3^f = \begin{bmatrix}0\\1\\0\\1\end{bmatrix} E_4^f = \begin{bmatrix}1\\1\\0\\0\end{bmatrix} ; S_B = \begin{bmatrix}1\\0\\0\\1\end{bmatrix}$$

Now, equation (5.9) gives:

$$r_1^f = \begin{bmatrix}0\\1\\0\\1\end{bmatrix} r_2^f = \begin{bmatrix}1\\0\\0\\0\end{bmatrix} r_3^f = \begin{bmatrix}1\\0\\1\\1\end{bmatrix} r_4^f = \begin{bmatrix}1\\1\\0\\0\end{bmatrix}$$

From equation (5.10), a system of linear equations is derived:

$$\begin{aligned} c_2 \oplus c_3 \oplus c_4 &= 1 \\ c_1 \oplus c_4 &= 0 \\ c_3 &= 0 \\ c_1 \oplus c_3 &= 0 \end{aligned}$$

The equation system has the solution: $c_1 = 0,\ c_2 = 1,\ c_3 = 0,\ c_4 = 1$. This means that f can exactly match the observed signature produced by the real fault $\widetilde{f}$ if and only if f is activated only in patterns 2 and 4. A similar procedure is applied to every fault in the circuit.

Fault Ranking

If the observed faulty signatures are the result of a single fault, the fault graph G of section 5.3 contains a single connected component. Therefore, a single fault candidate list is generated.

For each pattern block $B = (p_1, p_2, \cdots, p_n)$, let F_B be the set of faults that can explain the observed faulty behavior in the observed signature $\widetilde{S}_B$, that is:

$$\mathsf{F}_B := \{f | \text{ equation (5.10) is solvable for} f\}.$$

The method described in section 5.4.1 provides for each block B a set of fault candidates. The set of signatures a fault explains $X(f)$ can be calculated for all observed faulty signatures:

$$X(f) = \{\widetilde{S}_B | f \in \mathsf{F}_B\}$$

To exemplify this, assume a test set was divided into four pattern blocks, B_0 provided a correct signature, and the three remaining sets $\widetilde{B}$ had the following characteristics:

$$\widetilde{B}_1 = \{f_1\} \tag{5.12}$$
$$\widetilde{B}_2 = \{f_2, f_1, f_3\} \tag{5.13}$$
$$\widetilde{B}_3 = \{f_2\}. \tag{5.14}$$

Additionally, f'_1 is detected in 10 patterns of the whole test set, while f'_2 and f'_3 in 20 and 5 patterns, respectively.

Hence, $evidence(f_1) = (2, 10)$, $evidence(f_2) = (2, 20)$ and $evidence(f_3) = (1, 5)$. Both f_1 and f_2 explain two faulty signatures. However, as f'_2 is detected by a larger number of patterns in $\{B_0, B_1, B_2, B_3\}$ than f'_1, f_1 is regarded as the most likely candidate at the top of the fault list.

5.5. Logic Diagnosis for Increased Defect Density

In this section a procedure is presented for the diagnosis of multiple arbitrary faults. The basic idea of this procedure is the introduction of a *disturbance function*. If two faults are in different regions of a circuit, they do not affect each other and they may be analyzed using superposition. However, if the observation outputs of these faults overlap, the net effect in the observed signature is not anymore the combination of the errors produced independently by the two faults. The disturbance function describes this difference.

In the rest of this section it is assumed that one test signature is generated for each pattern in the test set. However, the same principles can be applied for signatures comprising multiple patterns.

5.5.1. The Disturbance Function for Fault Pairs

As long as there is a pattern in the test set which activates only a given fault, any diagnostic algorithm for single faults can also handle multiple faults reasonably well [Wang06b]. Therefore, the diagnostic challenge, especially when response compaction is employed, is to identify those faults which only become visible when other faults are activated in the same test pattern(s).

Figure 5.5 shows the behavior multiple faults in different fanout-free regions (FFRs) of the chip may have in a test pattern. In the top half of Figure 5.5 faults f_1 and f_2 have *disjoint observation outputs*, while in the bottom half faults f_3 and f_4 interact with one another and exhibit *overlapping observation outputs* that result from multiple fault masking and reinforcement effects [Ye10].

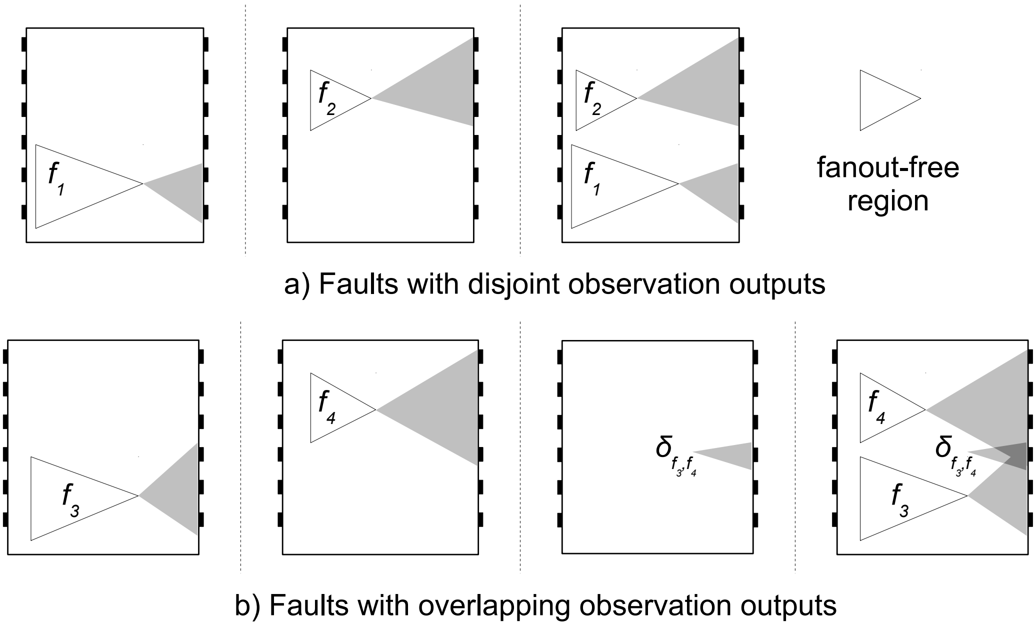

f1
f2
f2
f1
fanout-free region
a) Faults with disjoint observation outputs
f3
f4
δ f3,f4
f4
δ f3,f4
f3
b) Faults with overlapping observation outputs

Figure 5.5.: Disturbance function of a fault pair

Faults within a FFR may be subject to stronger interdependencies. For example, as shown in Figure 5.6, due to fault reconvergencies, a stuck-at-0 fault f may *dominate* another fault g so that the effect of g never becomes visible. Alternatively, the effects of f or g may be identical to that of a single fault h at their fanout node.

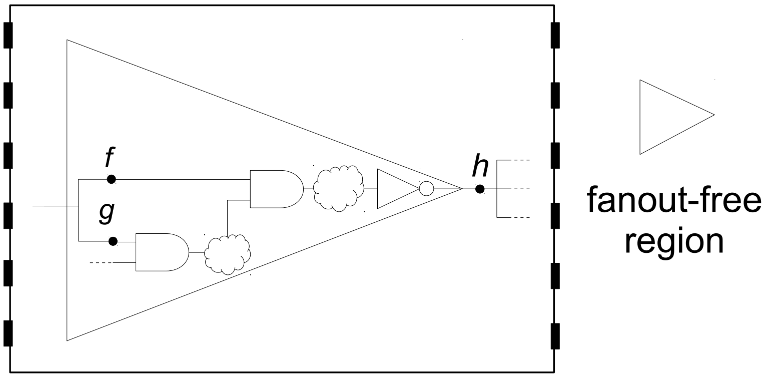


Figure 5.6.: Fault in a fanout-free region

For a combinational circuit V with inputs $I := (i_1, i_2, ..., i_p)$ let the circuit's fault-free output be $V(i_1, i_2, ..., i_p)$, and $V^f(i_1, i_2, ..., i_p)$ the faulty circuit function in the presence of fault f. The error function introduced by f is simply $D^f(i_1, i_2, ..., i_p) := V(i_1, i_2, ..., i_p) \oplus V^f(i_1, i_2, ..., i_p)$.

For independent faults f_1 and f_2, shown at the top of Figure 5.5, the error function

D^{f_1,f_2} introduced by the fault pair is

$$D^{f_1,f_2} := D^{f_1} \oplus D^{f_2} = D^{f_1} \vee D^{f_2}$$

since the output variables that equal 1 are disjoint.

For faults f_3 and f_4 with overlapping output outputs, shown at the bottom of Figure 5.5, a disturbance function δ_{f_3,f_4} is defined as:

$$\delta_{f_3,f_4} := D^{f_3,f_4} \oplus (D^{f_3} \oplus D^{f_4}) \tag{5.15}$$

The complexity of the disturbance function δ_{f_x,f_y} for any fault pair f_x and f_y will be used later on as a measure for selecting f_x and f_y as fault candidates.

The pairwise definition of a disturbance function describes whether the effect of the pair of conditional faults f_x, f_y can be constructed by the linear superposition of the respective single fault effects. If the observation outputs of f_x and f_y are disjoint, this is surely true as $\delta_{f_x,f_y} = 0$. The definition is extended to a set of faults F by

$$\delta_{\mathsf{F}} := D^{\mathsf{F}} \oplus \bigoplus_{f \in \mathsf{F}} (D^f) \tag{5.16}$$

5.5.2. Direct Diagnosis of Multiple Faults

State-of-the-art methods for direct diagnosis [Cheng04, Elm10] are extensions of the single location at a time (SLAT) approach [Huisman04]. Therefore, when multiple faults are present in the device, many signatures may not provide any diagnostic information as they cannot be explained by any single fault. The goal of the presented methodology for direct diagnosis is to explain the largest number of faulty signatures, both SLAT and non-SLAT, and take full advantage of the diagnostic properties of the test pattern set.

If a set $\widetilde{\mathsf{F}}$ of faults has pairwise disjoint observation output, it can be seen from Figure 5.5 that $\delta_{\widetilde{\mathsf{F}}}$ in formula (5.16) will be constant 0. Since linear superposition can be applied in this case, errors in the observed signature are the aggregation of the independent signature errors produced by each fault in $\widetilde{\mathsf{F}}$. As the activation conditions are not known beforehand, the signature is explained by solving a linear system of equations. Conversely, if the active faults have overlapping observation outputs, the disturbance function has to be taken into consideration. This is achieved by compacting the disturbance function δ into a signature error. With this additional construct, the observed faulty signature can also be represented as a linear combination of independent error signatures. This approach is an extension of the idea of *partially curable vectors* presented in [Huang97] for semiconductor debug and later in [Huang01] for logic diagnosis.

As discussed in the previous section, faults within the same fanout region demand additional considerations, since, for example, their failure behavior may not be distinguished from that of a dominating fault at the output of their fanout node. Therefore, two diagnostic scenarios can be identified depending on particular application requirements.

In *coarse-grained diagnosis* the goal is to identify the fanout node of a faulty gate. For example, in Figure 5.6 it suffices to identify fault candidate h for the successful diagnosis of either fault f or fault g. This is performed solely by solving systems of linear equations and it does not require the simultaneous activation of multiple faults during simulation. As this procedure is computationally efficient, is allows the statistical analysis of a large number of defective chips and provides accelerated feedback on the manufacturing process.

In *fine-grained diagnosis* the goal is to identify a few fault candidates which accurately represent the exact location of defects. In comparison to coarse-grained diagnosis, this procedures is is computationally more expensive but it directly optimizes PFA as fewer areas in the chip have to be inspected.

In the next sub-sections, the procedure to explain observed faulty signatures and select fault candidates is explained. The last sub-sections present a complete diagnostic algorithm for multiple faults, both for coarse-grained and fine-grained diagnosis, together with a short illustrative example in each case.

5.5.3. Explaining Non-SLAT Signatures

As in section 5.4.1, the fault-free output response $V := [v_1, v_2, \cdots, v_m]$ of a given pattern p_i is compacted by means of a MISR. For clarity, the resulting fault-free signature

S_i is referenced in the rest of this section without the pattern subscript. Therefore, the fault-free signature is:

$$S := \sum_{k=1}^{m} L^{m-k} v_k \qquad (5.17)$$

Assume that a maximum number of q faults are present in the circuit. The observed signature produced by faults $\widetilde{f}_1, \cdots, \widetilde{f}_q$ is $S^{\widetilde{f}_1, \cdots, \widetilde{f}_q}$.

The response error of a fault f is $D^f := [d_1^f, d_2^f, \cdots, d_m^f]$. As already defined above, D^f is the difference between the fault-free response and the faulty response under the presence of fault f:

$$D^f := V^f \oplus V$$

Similarly, the signature error E^f of fault f is defined as

$$E^f := S^f \oplus S \qquad (5.18)$$

where S^f if the faulty signature produced by fault f and S is the fault-free signature.

The relation between D^f and E^f is given by

$$E^f := \sum_{i=1}^{m} L^{m-i} d_i^f \qquad (5.19)$$

In order to explain an observed signature error $E^{\widetilde{f}_1, \cdots, \widetilde{f}_q}$ due to the activation of q faults, the effects of several candidate faults are modeled with a system of linear equations.

Let the *candidate fault set* FC $= \{f_1, \cdots, f_k\}$ contain the best candidate faults identified so far according to the ranking of section 5.3. This set may be initialized with the fault candidates obtained with any state-of-the-art approach for the diagnosis of single faults [Cheng07,Elm10]. As explained in the next sections, this set is updated as more erroneous signatures are explained.

An observed signature is explained if the following system of linear equations has a solution:

$$E^{\widetilde{f_1},\cdots,\widetilde{f_q}} = \begin{bmatrix} E^{f_1} & E^{f_2} & \cdots & E^{f_k} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_k \end{bmatrix} \tag{5.20}$$

where the constants $c_1, c_2, \cdots, c_k \in \{0, 1\}$ represent the activation conditions for the faults $f_1, f_2, \cdots, f_k$, respectively, according to the conditional fault model. Evidently, the interdependencies between $\widetilde{f_1}, \cdots, \widetilde{f_q}$ determine which combination (if any) of faults is able to explain an observed signature.

In the next sub-sections, sufficient conditions are presented which guarantee a solution for equation (5.20).

Faults with disjoint observation outputs

The real response error is

$$D^{\widetilde{f_1},\cdots,\widetilde{f_q}} := c_1 D^{\widetilde{f_1}} \oplus c_2 D^{\widetilde{f_2}} \oplus \cdots \oplus c_q D^{\widetilde{f_q}} \tag{5.21}$$

After applying linear superposition the error response is

$$E^{\widetilde{f_1},\cdots,\widetilde{f_q}} := c_1 E^{\widetilde{f_1}} \oplus c_2 E^{\widetilde{f_2}} \oplus \cdots \oplus c_q E^{\widetilde{f_q}} \tag{5.22}$$

Now, if the candidate set FC contains for each active real fault $\widetilde{f_i}$ a representative candidate f_i producing the same signature as $\widetilde{f_i}$, we can substitute the right hand side of equation 5.22 and find a solution for:

$$E^{\widetilde{f_1},\cdots,\widetilde{f_q}} := c_1 E^{f_1} \oplus \cdots \oplus c_q E^{f_q} \oplus c_{q+1} E^{f_{q+1}} \cdots \oplus c_k E^{f_k} \tag{5.23}$$

where $c_i = 0$ for $i > q$.

Faults with overlapping observation outputs

The output response error can be expressed as the combination of q and a disturbance function δ:

$$D^{\widetilde{f_1},\cdots,\widetilde{f_q}} := c_1 D^{\widetilde{f_1}} \oplus c_2 D^{\widetilde{f_2}} \oplus \cdots \oplus c_q D^{\widetilde{f_q}} \oplus \delta \tag{5.24}$$

The disturbance output can be compacted like any response. If the candidate fault set FC contains for each active real fault $\widetilde{f_i}$ a representative fault f_i, the following equivalent system of equations has a solution:

$$E^{\widetilde{f_1},\cdots,\widetilde{f_q}} := c_1 E^{f_1} \oplus c_2 E^{f_2} \oplus \cdots \oplus c_{k-1} E^{k-1} \oplus c_\delta E^{\delta} \tag{5.25}$$

What remains is to identify the disturbance function δ and the disturbance signature E^{δ}. Without any restriction on E^{δ}, equation (5.25) would be solvable for *any* candidate set and would not provide any useful diagnostic information. As the solution of equation (5.25) should serve to identify the combination of likely fault candidates $f_1, f_2, \cdots, f_k$, δ must be restricted with a reasonable heuristic.

As the disturbance function should be as simple as possible, δ is selected so as to represent the effects of a single line flip in the circuit. Hence, an auxiliary fault location $f_\delta \in \mathsf{F} \setminus \mathsf{FC}$ is evaluated as an approximation of δ. The test response of f_δ is compacted and equation (5.25) is solved for $E^{\delta} := E^{f_\delta}$.

5.5.4. Diagnostic Procedure

In the previous sub-section, a methodology was detailed to explain faulty signatures with the linear combination of several conditional stuck-at faults. In order to create candidate fault lists, the set of explained signatures $X(f)$ has to be carefully selected for each fault $f \in \mathsf{F}$ in the circuit. In this process, special attention has to be devoted to generate enough connected components in the node graph $G = (N, L)$ as to identify all the faults affecting the DUD.

In the next sections two diagnostic procedures are presented, which incrementally update $X(f)$ as faulty signatures are explained. These procedures are devised for different diagnostic goals. In the next section, the diagnostic algorithm is optimized to yield coarse-grained diagnostic resolution with reduced computational overhead.

In the following section, the diagnostic algorithm is optimized fine-grained diagnosis, although at the expense of additional computational processing.

Coarse-Grained Diagnostic Procedure

The goal of the coarse-grained diagnostic procedure is to identify the fanout node of each faulty gate as efficiently as possible.

As shown in Algorithm 1, observed signatures $\widetilde{S}$ are first explained using a SLAT-based approach. In lines 4-13 each fault $f_i \in \mathsf{F}$ is simulated and, if the simulated signature S^{f_i} matches $\widetilde{S}$, the observed signature is added to $X(f_i)$, i.e. the set of signatures explained by f_i. Next, the diagnostic graph G is updated.

The function UPDATE_GRAPH described in Algorithm 2 checks if the input fault f explains the same signature as any other fault $f_j \in \mathsf{F}$ and produces the corresponding edge (f, f_j) in L. In fact, this function can be made much more efficient as it is only necessary to evaluate if fault f belongs to any of the existing connected components in G. For the sake of simplicity, these implementation details are omitted in the rest of this chapter.

The unexplained faulty signatures are further analyzed in lines 14-26 by considering the effects of multiple faults. This step is performed iteratively. In each iteration the nodes and edges of the graph G are modified if equation (5.20) can be solved. The number of iterations Q is a user-defined parameter that accounts for the computational effort of the algorithm. Experimental results show that for $Q \leq 2$ most faulty signatures can already be explained.

For each unexplained signature $\widetilde{S}$, let fault set FC' contain the best-ranked $k-1$ faults from the current fault lists so that $E^{f_1}, E^{f_2}, \cdots, E^{f_{k-1}}$ are different and non-zero (line 16). k is a user-defined parameter that fixes the maximum number of free variables in equation (5.20). In order to assure that, even when all faults are active, equation (5.20) has a solution, k must be greater than the number of expected faults in the circuit. Note that if $r < k$ faults are active, equation (5.20) may still have a solution with $c_i = 0$ for $r < i \leq k$. However, k should not be set too high, as equation (5.20) may become underdetermined. Additional insight into the value of k is given in section 5.5.5.

In line 19 each fault $f_i \in \mathsf{F} \setminus \mathsf{FC}'$ is selected and equation (5.20) is solved for $\mathsf{FC} =$

$f_i \bigcup \mathsf{FC}'$. A solution of equation (5.20) defines a set of active faults $A_{\widetilde{S}}$:

$$A_{\widetilde{S}} = \{f_k | c_k = 1 \text{ in equation 5.20 }\} \tag{5.26}$$

If $f_i \in A_{\widetilde{S}}$, $\widetilde{S}$ is included in $X(f_i)$ and the graph's edges are updated. As new faults may be added to N when a signature is explained, new connected components may be generated in G. As a result, the candidate fault set FC may contain other candidates in the next iterations of the algorithm so that additional faulty signatures can be explained.

Finally, after Q iterations, a fault candidate list is generated for each connected component in G. These candidate fault lists are ranked, and those faults at the top of each list are selected as the most likely diagnostic candidates (line 27).

Note that solving equation (5.20) allows the identification of faults for which no SLAT pattern is present in the test set. For example, Figures 5.7 and 5.8 show a circuit affected by two faults a and b. As Figure 5.7 shows, fault a can directly explain the signatures corresponding to SLAT patterns 1 and 2, while b by itself cannot explain any SLAT pattern. Moreover, fault f_δ explains only one of the two signatures explained by fault a and affects only circuit output o. Consequently, one connected component in the fault graph G is generated, which includes both faults a and f_δ.

As Figure 5.8 shows, masking and reinforcement effects between faults a and b are observed for pattern 3 only in output o. This non-SLAT pattern can be explained if FC contains a, b, as well as fault f_δ, that corresponds to the disturbance function. Thus, a new connected component in G can be created for b and, therefore, b can also be identified as fault candidate.

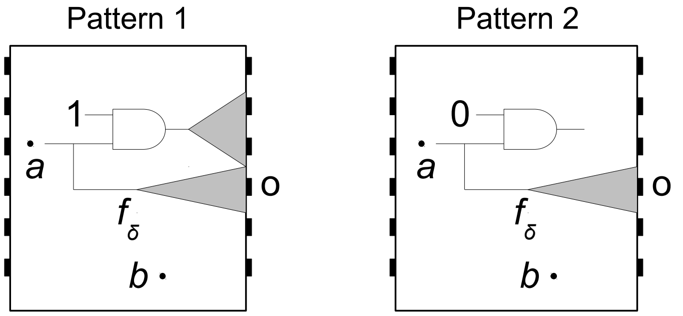


Figure 5.7.: SLAT patterns

In this diagnostic procedure the fault signatures can be calculated on-the-fly or pre-

Algorithm 1 Coarse-Grained Diagnostic Procedure

```
Input: Observed signatures $OBS$
$N = \emptyset$
$L = \emptyset$
$SLAT = \emptyset$
for all $\widetilde{S} \in OBS$ do
    for all $f_i \in$ F do
        Find $S^{f_i}$
        if $S^{f_i} == \widetilde{S}$ then
            $SLAT = \widetilde{S} \cup SLAT$
            $X(f_i) = \widetilde{S} \cup X(f_i)$
            UPDATE_GRAPH($f_i$, $G$)
        end if
    end for
end for
for $1 \rightarrow Q$ do
    for all $\widetilde{S} \in OBS \setminus SLAT$ do
        FC′ := best-ranked $k - 1$ candidates from connected components of $G$
        for all $f_i \in$ F $\setminus$ FC′ do
            FC := FC′ $\cup f_i$
            Solve equation (5.20)
            if equation (5.20) has a solution and $f_i$ is active then
                $X(f_i) = \widetilde{S} \cup X(f_i)$
                UPDATE_GRAPH($f_i$, $G$)
            end if
        end for
    end for
end for
Fault Ranking
```

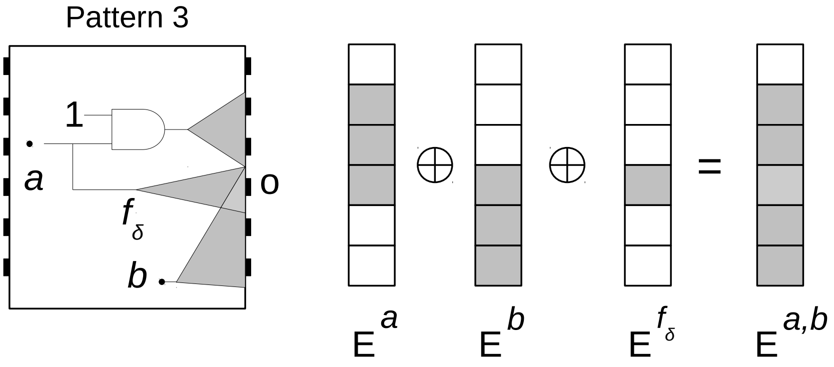


Figure 5.8.: Non-SLAT pattern

Algorithm 2 Construction of diagnostic graph

```
Input: Fault $f$
Input: Diagnostic graph $G = (N, L)$
function UPDATE_GRAPH($f$, $G$)
    for all $f_j \in N$ do
        if $X(f) \cap X(f_j) \neq \emptyset$ then
            $L = (f, f_j) \cup L$
        end if
    end for
    $N = f \cup N$
end function
```

computed and stored beforehand. In this case, the only computational effort amounts to the solution of linear systems of equations.

Fine-grained Diagnosis Procedure

The procedure for coarse-grained diagnosis has the disadvantage that, in the worst case, all faulty locations in the same fanout-free region have to analyzed. The goal of fine-grained diagnosis is to identify a reduced number of defect locations to be inspected. This endeavor justifies the expense of additional computational effort in the diagnostic procedure.

Figure 5.9 illustrates the additional challenges in the fine-grained diagnosis of multiple faults. The figure shows two faults in different FFRs of the chip. Assume any input of gate *G1* is logic zero and the input of the inversor is logic one in all the patterns for which the condition *cond* is true.

As the effects a has on the circuit's outputs need to be reinforced by fault b, fault a can only be propagated to the output of gate *G2* if fault b is active. Otherwise fault a is masked out. As a consequence, fault a, by itself, does not produce any error and the linear combination of a and b cannot explain the faulty signature produced when all outputs are faulty. However, the disturbance function δ produced by the fanout node of a, together with b, may explain the observed signature, as it is independent of the activation of b.

This observation is exploited to identify fault candidates more accurately: The propagation of a fault to its fanout node is analyzed in more detail to identify faulty chip

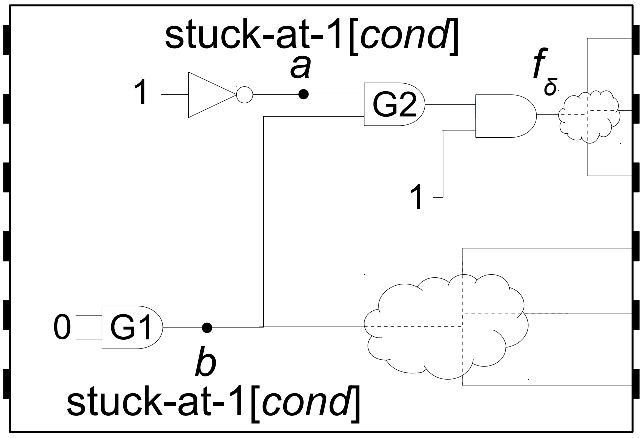


Figure 5.9.: Diagnostic example of fine-grained diagnosis

locations within a FFR.

Let $fo(f)$ be the fanout node of fault f and fault set $FFR(f)$ all the faults in the same region of f.

As shown in Algorithm 3, the procedure for fine-grained diagnosis is similar to its coarse-grained counterpart. However, in lines 21 to 27, if $f_i \in \mathsf{F} \setminus \mathsf{FC}'$ explains a non-SLAT signature, faults in the same fanout-free region of f_i are evaluated to determine if they can also explain the observed signature. For this purpose, these faults $f_r \in FFR(f_i)$ are simulated under the effects of the other active faults in $A_{\widetilde{S}}$. The circuit's response $V^{A_{\widetilde{S}} \setminus f_i \cup f_r}$ is evaluated explicitly in order to assess whether both f_i and f_r can explain this signature.

To accomplish this task it suffices to test whether the fault f_r is propagated to the fanout node of f_i in this test pattern (line 23). If this is the case, the nodes and edges of the graph node G are updated accordingly.

Note that in line 22 all active faults are injected simultaneously and simulated to determine if other faults within a FFR are also able to explain the observed signature. However, the required computational effort does not grow exponentially, as fault simulation is performed only if equation (5.20) has a solution.

Algorithm 3 Fine-Grained Diagnostic Procedure

```
Input: Observed signatures $OBS$
$N = \emptyset$
$L = \emptyset$
$SLAT = \emptyset$
for all $\widetilde{S} \in OBS$ do
    for all $f_i \in \mathsf{F}$ do
        Find $S^{f_i}$
        if $S^{f_i} == \widetilde{S}$ then
            $SLAT = \widetilde{S} \cup SLAT$
            $X(f_i) = \widetilde{S} \cup X(f_i)$
            UPDATE_GRAPH($f_i$, $G$)
        end if
    end for
end for
for $1 \rightarrow Q$ do
    for all $\widetilde{S} \in OBS \setminus SLAT$ do
        $\mathsf{FC}'$ := select best-ranked $k-1$ candidates from connected components
        for all $f_i \in \mathsf{F} \setminus \mathsf{FC}'$ do
            $\mathsf{FC} := \mathsf{FC}' \cup f_i$
            Solve equation (5.20)
            if equation (5.20) has a solution and $f_i$ is active then
                for all $f_r \in FFR(f_i)$ do
                    Find $V^{A_{\widetilde{S}} \setminus f_i \cup f_r}$
                    if $f_r$ propagates to $fo(f_i)$ then
                        $X(f_r) = \widetilde{S} \cup X(f_r)$
                        UPDATE_GRAPH($f_r$, $G$)
                    end if
                end for
            end if
        end for
    end for
end for
Fault Ranking
```

5.5.5. Aliasing Probability

In this section the probability is estimated that, although equation (5.20) has a solution, the corresponding fault candidate set FC leads to erroneous diagnostic results. In this analysis the choice of k to solve equation (5.20) plays an important role. Equation (5.20) describes a system of linear equations with k variables and n equations, where n

is the size of the MISR. If $k = n$, equation (5.20) is guaranteed to have a solution if the error responses $E^1, E^2, \cdots, E^k$ of the faults in the candidate set $\mathsf{FC} = \{f_1, f_2, \cdots, f_k\}$ are linearly independent. Therefore, equation (5.20) is solvable for *any* such combination of candidates, although the active faults $A_{\widetilde{S}} \subset \mathsf{FC}$ do not reproduce the observed failure behavior $\widetilde{V}$ at the outputs of the DUD. Therefore, they are not good candidates for logic diagnosis. This situation is known as *aliasing* and it must be kept reasonably low.

The aliasing probability P_A is defined as follows:

$$\text{P}(\widetilde{V} \neq V^{A_{\widetilde{S}}} |\text{ equation (5.20) is solvable})$$

The maximum number of error signatures spanned by the linear combination of $E^{f_1}, E^{f_2}, \cdots, E^{f_k}$ error responses is at most $2^k - 1 < 2^n$. To estimate the aliasing probability, the total number of DUD responses producing the observed signature is compared with the total number of possible output signature responses.

For an output sequence of length l and under regular BIST assumptions that the faults are equally probable, the aliasing probability P_A is

$$P_A = \frac{(2^{l \cdot m + k - n} - 1)}{(2^{l \cdot m} - 1)} \approx 2^{k-n} \text{ for } k < n \tag{5.27}$$

As equation 5.27 shows, any given number of faults in the candidate set k, the MISR length n can be adjusted to make the aliasing probability sufficiently low. Conversely, for a fixed MISR length, k may be selected in order to reduce the aliasing probability.

5.6. Automotive Use-Case: Semiconductor Failure Analysis

This chapter has presented a generic logic diagnosis methodology which requires only compacted test signatures for accurate diagnostic results. This diagnostic strategy can be directly applied for the analysis of field returns in the automotive domain. Thus, the diagnostic requirements of the use-case *Semiconductor Failure Analysis* can be fulfilled.

Figure 5.10 shows a generalized approach for field return analysis. In the first step, BIST is applied during system-level tests and compacted test signatures are collected from the faulty integrated circuit (IC) (1). The regular automotive failure analysis procedures are conducted and the faulty board is relayed across the supplier chain (2). The test signatures are then transferred to the chip manufacturer, who conducts logic diagnosis (3). Logic diagnosis can be immediately performed using the gathered diagnostic data without damaging the electronic control unit (ECU) or having to remove it from the vehicle. Finally (4), the chip manufacturer confirms diagnostic results by means of additional structural tests or PFA. The developed algorithms offer a complementary tool which allows the immediate analysis of all field returns in order to maximize the success of PFA.

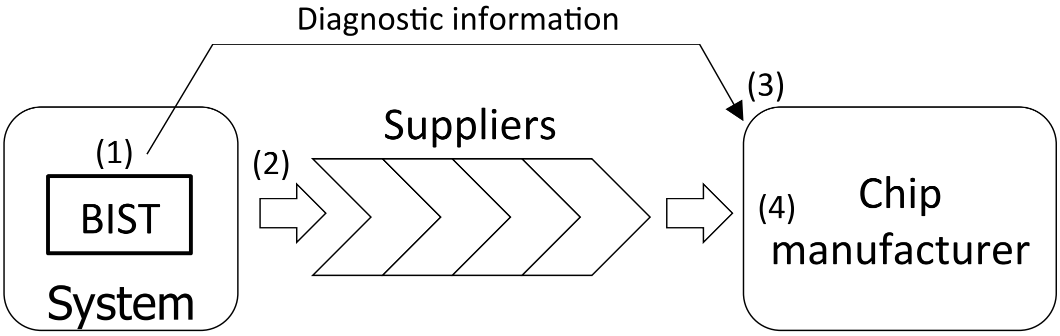


Figure 5.10.: Field return analysis

The procedure to collect test signatures from a DUD is illustrated in Figure 5.11. Any simple tester, like a portable computer or laptop, serves as *test source/sink*. The tester is connected to the DUD by means of an available test interface, like the IEEE 1149.1 standard. After gaining access to the DUD, the tester configures the application of BIST. Test patterns can be generated on-chip or transferred from the tester to the DUD. Then, the tester manages the test application process so that intermediate signatures can be collected.

5.7. Summary and Outlook

The presented direct diagnosis solution for BIST offers the opportunity to identify defective locations in a faulty device using only highly compacted test responses. The diagnostic algorithm takes advantage of the linear properties of a MISR to evaluate how closely one or a few faulty locations are able to reproduce the faulty behavior

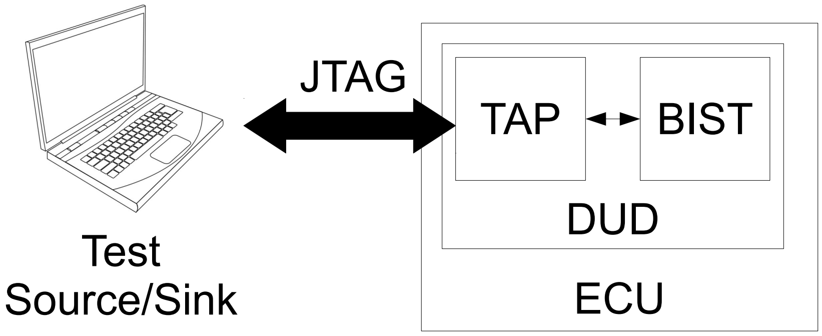


Figure 5.11.: Diagnostic architecture for field return analysis

observed in the collected responses from the device under diagnosis.

The diagnostic procedure can be adjusted to strike the right compromise between response compaction and the maximum number of faults present in the device. On the one hand, logic diagnosis can be optimized to evaluate the faulty signature produced by several test patterns under the influence of a single faulty location. On the other hand, several arbitrary faults can be diagnosed simultaneously at the expense of larger number or intermediate signatures.

In any case, the effects one or several conditional faults may have on an observed signature is determined by solving systems of linear equations. This makes it possible to identify faults which are able to explain an observed faulty signature, either by themselves or in combination with other faults. The most likely faults are identified by ordering the fault set according to the extent to which they explain the complete test set.

The developed diagnostic algorithm may be directly applied for the analysis of semiconductor field returns. It fits the traditional automotive diagnostic flow and fulfills the accuracy and resolution requirements for the automotive diagnostic use-case *Semiconductor Failure Analysis*.

6. Test Architecture for Built-In Diagnosis

This chapter describes a novel test architecture for the on-chip storage of diagnostic data [Cook11a, Cook12b, Cook12a]. This architecture collects test signatures during built-in self-test (BIST) so that they can be later used for logic diagnosis. Thus, failure analysis can be performed with test responses gathered directly from the device-under-diagnosis (DUD) without the need of specialized external test equipment. For this purpose, the traditional STUMPS architecture is extended in order to store intermediate faulty signatures. The diagnostic architecture is designed to reduce hardware costs which arise from the introduction of additional on-chip memories to generate diagnostic data.

6.1. Generic Diagnostic Architecture

During traditional manufacturing test, dedicated tester channels are employed to transfer test stimuli to the device-under-test and to gather test responses. As there is no such external test *source* or *sink* during BIST, an on-chip diagnostic architecture must employ available resources within the system to produce input patterns and store test signatures. For this reason, the amount of information required to generate and store diagnostic data may have a strong impact on the overall cost of a diagnostic solution.

A suitable diagnostic architecture must fulfill two main requirements: Firstly, test responses have to be compacted as much as possible, while still supporting accurate logic diagnosis. Secondly, the test architecture shall not constrain the procedure for test pattern generation, especially if long inexpensive pattern sequences are used as part of a mixed-mode test strategy. In the rest of this chapter an optimized diagnostic architecture is presented to meet these goals.

Figure 6.1 depicts a generalization of the STUMPS architecture to collect and store diagnostic data without an external tester. It is based on the architecture of [Elm10], where short faulty signatures are stored on-chip so that they can later be retrieved for closer examination. However, in [Elm10] a parity tree and special procedures for test pattern generation are enforced, whereas the proposed architecture is fully compatible with the traditional STUMPS architecture and achieves higher compaction ratios without compromising diagnostic performance.

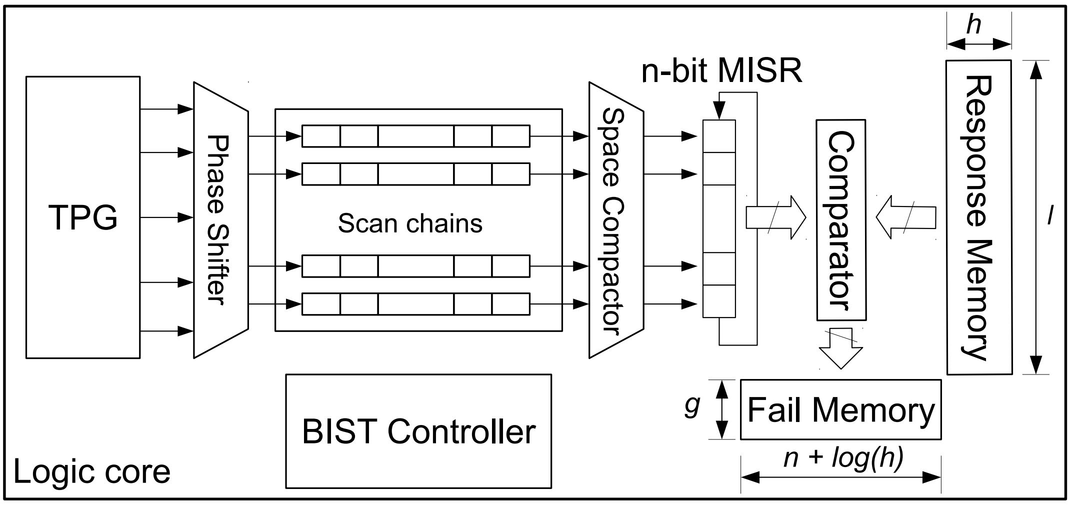


Figure 6.1.: Generic diagnostic architecture

The input side of the STUMPS architecture does not require any modification. Therefore, any technique for test pattern generation shall be used to achieve sufficient fault coverage.

On the output side, an n-bit multiple-input signature register (MISR) is fed by a space compactor succeeding the scan chains. Any approach shall be used to make sure no "*X*" value corrupts the MISR signature [Hetherington99, Tang06, Czysz10].

In order to gather sufficient diagnostic information, the test set is partitioned into h intermediate pattern sequences. The expected intermediate responses are stored in a *response memory*. The response memory contains h entries, each with size $l < n$. After an intermediate test signature is obtained during test application, it is compared to the expected test signature in the response memory. If the two signatures differ, the obtained signature is stored in the *fail memory* along with its intermediate signature index, thus resulting in a fail memory width of $n + \log h$ bits. The state of the MISR is reset after each intermediate test response is generated. In order to analyze only the first failing signatures in the test application process, the depth of the fail memory is limited to g entries. As shown in [Elm10], g can be kept relatively small (50 entries) and yet achieve excellent diagnostic performance. Moreover, as the fail memory is much smaller than the response memory, its size is not critical for the overall hardware costs.

6.2. Efficient Storage of Response Data

In this section the width of the response memory l is minimized in order to reduce the hardware overhead of the diagnostic architecture. The error propagation properties of the MISR are utilized to achieve this goal with only marginal fault coverage loss.

The purpose of an entry in the response memory is simply to evaluate if the produced signature should be stored in the fail memory. As this evaluation can be performed with very little failure information, the size of the expected signatures can be greatly reduced. With this idea in mind, a shadow MISR is introduced. At the end of each session, the obtained signature is transferred to the shadow MISR, which is then allowed to run free until the MISR has compacted the first pattern of the next session. It is sufficient to observe only a few bits $l < n$ in the shadow MISR in order to detect any deviation from the expected signature. Thus, the width of the response memory can be set without regard to the number of bits in the original MISR. In the following, the reasoning behind this observation is explained in detail.

Assume, as sketched in Figure 6.2, that the output response of the scan chains at time t is d^t, and the MISR compacts d^t in each time step. If an error $e = (e_0, \cdots, e_{n-1})$ occurs at time $t = 0$, the value $d^1 \oplus e$ enters the MISR instead of d^1, and the sequence $L \cdot e, L^2 \cdot e, L^3 \cdot e, \cdots$ describes the differences of the following states from the respective fault free states. This shows that the error propagation evolves like the state sequence of an autonomous linear feedback shift register (LFSR) with feedback polynomial $f(x)$ and initial state e. It is known from LFSR theory that the output sequence observed at a flip-flop x_i is a pseudo-random sequence if $f(x)$ is a primitive polynomial. This means that the probability of detecting an error e at the output of a flip-flop x_i is $\frac{1}{2}$. Accordingly, observing l state bits increases the probability of error detection to $1 - 2^{-l}$.

Although the probability of error detection is already very close to 1.0 for small values

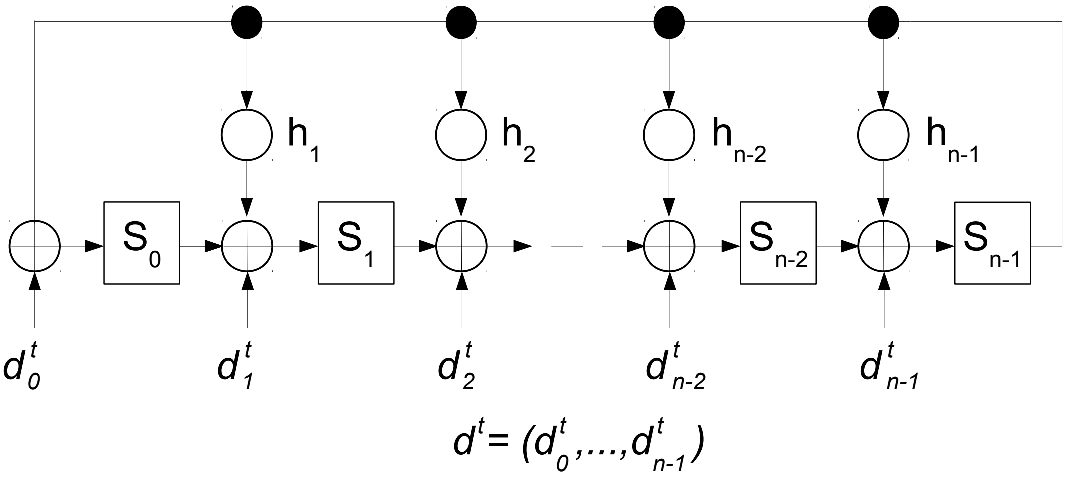


Figure 6.2.: Error propagation in MISR

of l, errors can still be masked in a worst-case scenario. For instance, if state bits $S_{n-l}, \cdots, S_n$ are observed and error e first occurs in S_0, at least $n - l$ clocks are needed to observe this error. However, as described above, the shadow MISR runs as long as the first pattern of the next session is completely compacted. Consequently, the architecture provides enough time for error propagation and it is sufficient to observe l bits of the shadow MISR. Figure 6.3 shows the extended diagnostic architecture.

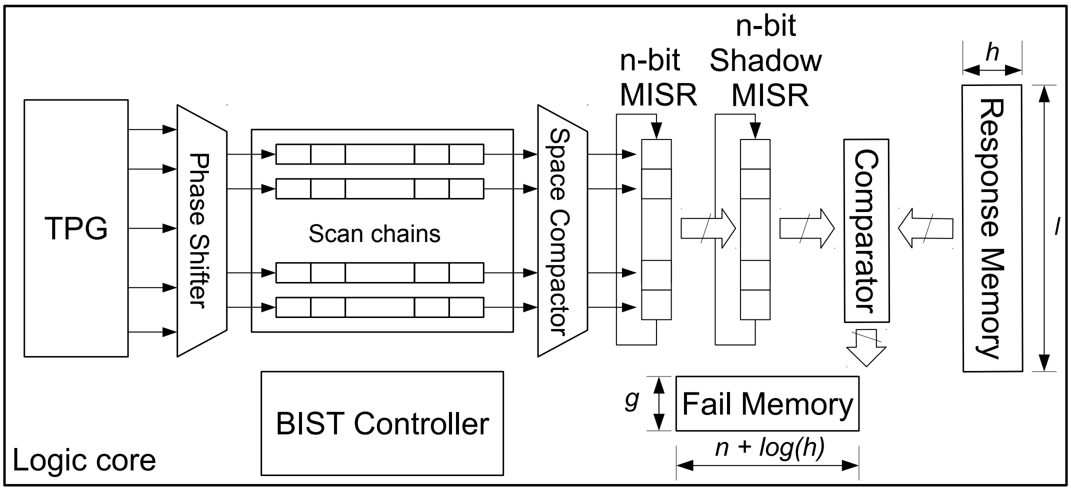


Figure 6.3.: Diagnostic architecture with shadow MISR

6.3. Diagnostic Architecture for Mixed-Mode BIST

The straightforward combination of any mixed-mode test pattern generation scheme with the diagnostic architecture of the previous section drives the storage requirements of the response data beyond acceptable limits, as some diagnostic information needs to be stored for a potentially large number of intermediate signatures. To retain the advantages of mixed-mode BIST, the diagnostic architecture presented in this section addresses BIST sessions containing long test pattern sequences.

The benefit of pseudo-random patterns in mixed-mode BIST is twofold. On the one hand, they reduce the storage requirements for deterministic test data and, on the other hand, the coverage of non-target faults improves test quality. Clearly, the key challenge is keeping the size of the response memory within a viable range without dropping important information on non-target faults in the pseudo-random sequence. The main idea to achieve this goal is to identify regions of the pseudo-random test sequence that provide essential diagnostic information. For this purpose, the detection profile of the target fault set F is used as a heuristic measure. *Strong* windows are determined, so that each target fault is covered by at least one *strong* window. Strong windows are utilized to generate faulty signatures suitable for accurate logic diagnosis. Therefore, strong windows contain a fixed number of patterns. As a result, the number of strong windows must be minimized in order to reduce the hardware cost of the response memory. This minimization procedure consists of a test generation phase followed by a partitioning phase. After the faulty test signatures are gathered, logic diagnosis is performed.

In the next subsections these steps are detailed in the context of mixed-mode BIST with pseudo-random test patterns (PRPs). However, the same approach can be used for pseudo-exhaustive and partial pseudo-exhaustive test (P-PET).

6.3.1. Test Generation

To generate a mixed-mode test with r PRPs, the fault set F is simulated against the patterns produced by the on-chip pattern generator of the target BIST architecture. The generated pseudo-random test is denoted by $\mathsf{T}_{rand}(r)$, and the subset of detected faults by $\mathsf{F}_{rand}(r)$. Then deterministic patterns are produced by means of automatic test pattern generation (ATPG) in order to account for the remaining hard-to-detect faults $\mathsf{F}_{hard}(r)$. The obtained deterministic test set is denoted by $\mathsf{T}_{det}(r)$.

6.3.2. Partitioning

The algorithm to identify *strong* diagnostic windows proceeds as follows.

- In a first step, the *essential* random patterns are extracted from $\mathsf{T}_{rand}(r)$. A subset $\mathsf{E}_{rand}(r)$ of $\mathsf{T}_{rand}(r)$ is called essential, if the patterns in $\mathsf{E}_{rand}(r)$ detect all faults in $F_{rand}(r)$ and $\mathsf{E}_{rand}(r)$ is a set of minimum cardinality. As computing $\mathsf{E}_{rand}(r)$ corresponds to solving a complex covering problem, the set of essential random patterns is approximated during fault simulation by selecting only those patterns of $\mathsf{T}_{hard}(r)$ which detect a least one new fault.
- Each essential pattern in $\mathsf{E}_{rand}(r)$ defines a strong diagnostic window of length $w = 2^k$. To simplify control, strong windows are positioned around the essential patterns, so that the starting index of a window is a multiple of 2^k (see Figure 6.4). That is, if t_j is an essential pattern, then the window around t_j starts with pattern t_i, where $i = 2^k \cdot (j\ div\ 2^k)$, and t_j has position $j\ mod\ 2^k$ in the window. In order to diagnose single faults, as discussed in section 5.4, the number of test patterns 2^k comprising a response signature may not be larger than the size of the MISR.

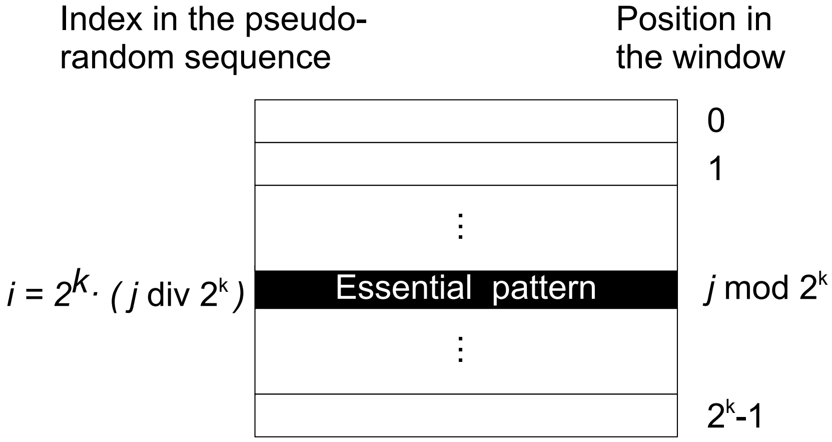


Figure 6.4.: Strong diagnostic window

- To keep as much diagnostic information as possible on non-target faults, all random patterns between two strong diagnostic windows are treated as one *weak* diagnostic window. Thus, the size of a weak window is a multiple of 2^k, and a signature is generated only at the end of a weak diagnostic window.

- Finally, the deterministic patterns in $\mathsf{T}_{det}(r)$ are divided into diagnostic windows of size 2^k, which are also referred to as strong windows. Overall, the test is then structured as shown in Figure 6.5.

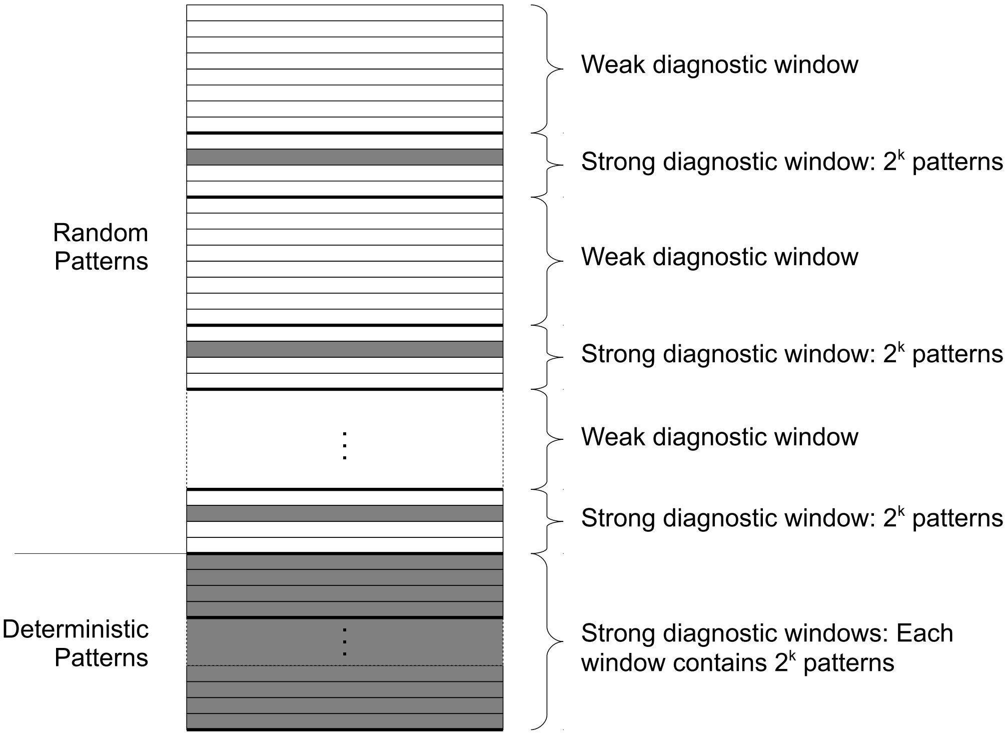

Weak diagnostic window
Strong diagnostic window: 2^k patterns
Random Patterns
Weak diagnostic window
Strong diagnostic window: 2^k patterns
Weak diagnostic window
Strong diagnostic window: 2^k patterns
Deterministic Patterns
Strong diagnostic windows: Each window contains 2^k patterns

Figure 6.5.: Aligned diagnostic windows

To minimize the number of strong windows, the number of essential random and deterministic patterns can be further reduced by reverse-order fault simulation of the patterns in $\mathsf{E}_{rand}(r) \bigcup \mathsf{T}_{det}(r)$ against F. Deleting unnecessary patterns provides reduced sets $\mathsf{E}^*(r)$ and $\mathsf{T}^*(r)$ as the basis for strong windows. The overall partitioning flow then proceeds as shown in Figure 6.6.

The size of the response memory is determined not only by the number of strong windows, but also by their distribution in the pseudo-random sequence. As weak windows can have varying size, additional control information is necessary to identify the beginning of weak and strong windows. If the number of strong diagnostic windows is large, this information is provided by tagging all patterns in the pseudo-random sequence with index $0 \ mod \ 2^k$ appropriately with one extra bit. Alternatively, if only few

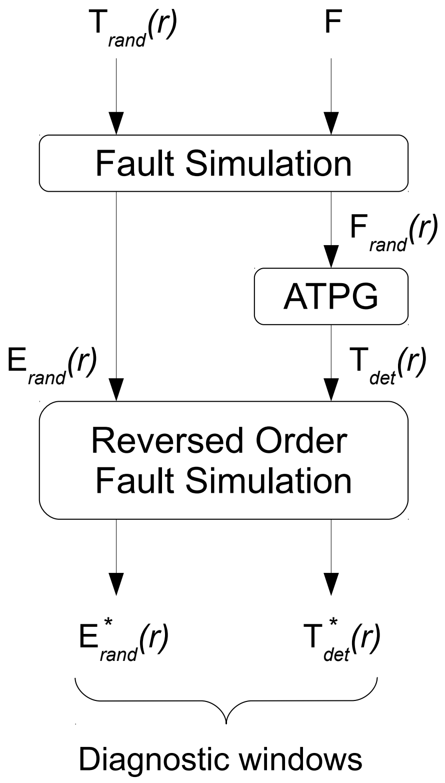


Figure 6.6.: Partitioning the test set into diagnostic windows

strong diagnostic windows are needed, the index of the starting pattern $(div\ 2^k)$ can be stored with the compacted signature in the response memory.

6.3.3. Diagnosis

If faulty signatures are observed at the end of both weak and strong diagnostic windows, then the diagnosis procedure of section 5.4 is applied to the strong windows only. Fault simulation of the weak windows is used to validate the results. If several candidate fault locations can explain the faulty behavior with the same evidence, then the reference signatures of weak windows are analyzed in more detail to improve diagnostic resolution. If a faulty signature appears only at the end of a weak window, then the direct diagnosis procedure described in [Holst12] can be applied (or any other direct diagnosis procedure that works on larger windows).

6.4. Automotive Use-Case: Power-Up/Down Test

The diagnostic architecture presented in this section enables the autonomous collection of faulty test signatures which are later analyzed during logic diagnosis. This diagnostic approach can be executed during system-level test without the need of external equipment or direct access to internal components of the system. Therefore, diagnostic tests can be conducted during a vehicle's inactivity periods like, for example, power-up and power-down.

Figure 6.7 shows a generic automotive diagnostic flow suitable for the automotive use-case *Power-Up/Down Test*. Diagnostic signatures are collected during BIST before the vehicle enters operational mode or right before system shutdown. As BIST is applied under similar conditions the DUD endures during regular operation, the number of "no trouble found (NTF)" occurrences can be greatly diminished. The diagnostic signatures are stored in non-volatile memory and are later retrieved in the workshop for subsequent analysis. At this point logic diagnosis is performed in order to enhance the traditional diagnostic failure analysis flow as described in section 5.6.

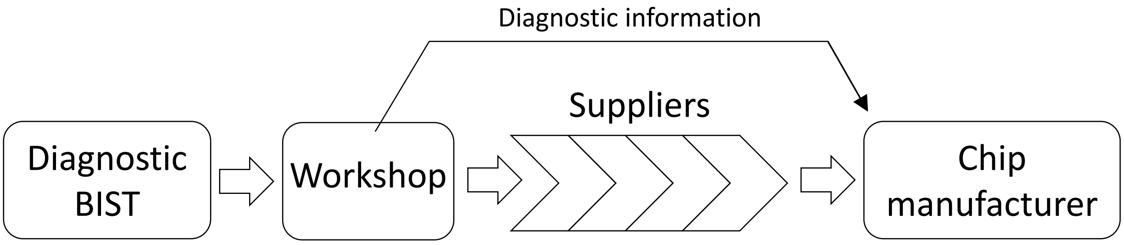


Figure 6.7.: Automotive diagnostic flow

An on-chip diagnostic architecture suitable for automotive diagnosis is shown in Figure 6.8. Both the *test source* and the *test sink* are realized by means of non-volatile on-chip memories. The *test source* comprises both the test data required for the application of deterministic patterns in a mixed-mode BIST session and the response data to identify faulty test signatures. The *test sink* holds the obtained faulty signatures from the test application process.

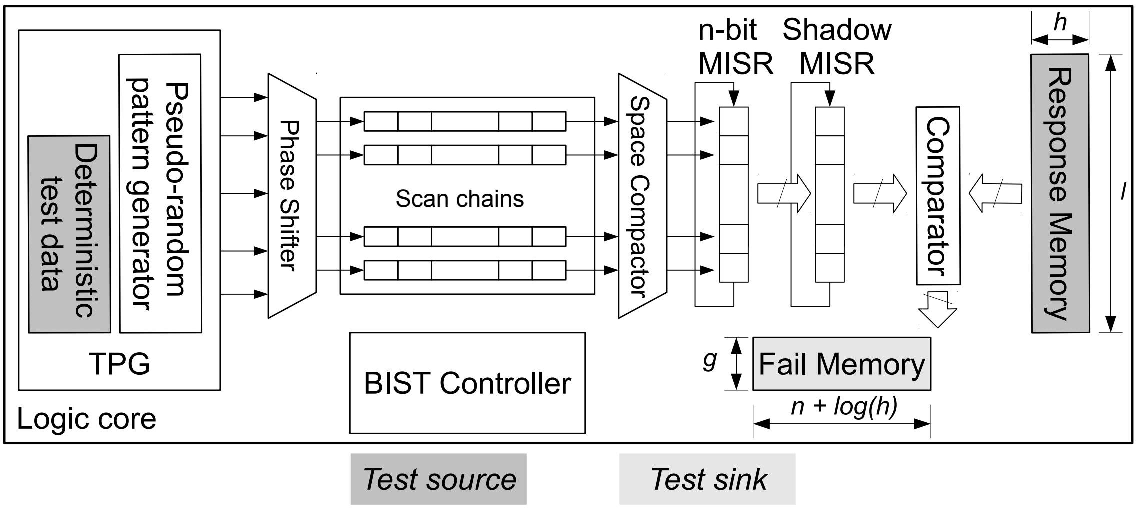


Figure 6.8.: On-chip test architecture for built-in diagnosis

6.4.1. BIST Application under Functional Constraints

As described in chapter 4, the application of structural tests may take advantage of any available system resource. Similarly, the diagnostic BIST architecture introduced in this chapter may also exploit existing resources as long as they are not presently required for any functional task during the operation of the system. For this purpose, some system components must remain operational while the described diagnostic BIST procedure is applied to other components. As a consequence, any potential dependencies between diagnostic and functional tasks must be carefully analyzed. This poses additional design challenges in order to make sure the application of BIST does not have any negative effect on the system's performance. This is especially important, for example, for automotive system-wide communication, as changing schedules for diagnosis execution during runtime is prohibited.

The system-level integration of BIST under functional constraints is a complex optimization problem. The challenge is to find a suitable system implementation that not only fulfills the usual specification requirements like, for example, bus performance, local schedules of electronic control units (ECUs), available memory, and cost, but also achieves optimal diagnostic performance. This can only be achieved with a holistic system model that captures both functional properties and the characteristics of the BIST architecture. The most suitable compromises with regard to functional and non-functional system properties as well as to diagnostic capabilities can then be evaluated by means of design state exploration.

A suitable model of the developed diagnostic architecture is shown in Figure 6.9. The *test source* containing the *deterministic test data* and the *response data* may be a local memory in the ECU. This is the case for *ECU 1* in Figure 6.9. Alternatively, the *test source* may be centralized in another system component like a gateway. For example, *ECU 2* may use an available controller area network (CAN) network as a test access mechanism (TAM) to gather the necessary test data. Similarly, a local memory may serve as *test sink* to hold the fail data. The *test sink* may also be located in any other non-volatile memory in the system.

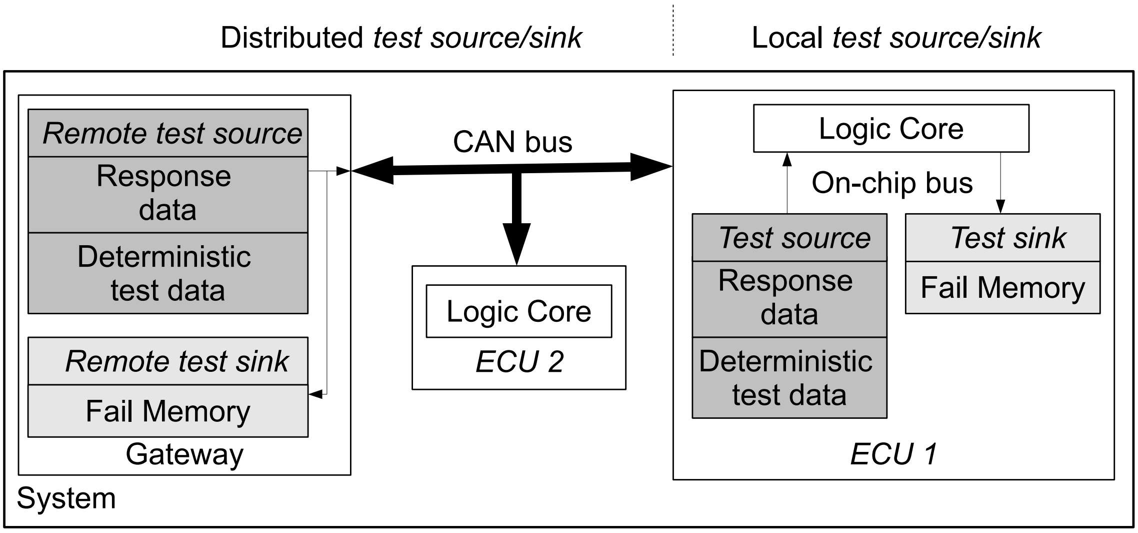


Figure 6.9.: System-level diagnostic architectural model

The work in [Abelein14] makes use of the described diagnostic infrastructure of Figure 6.9 in order to architect the integration of automotive structural diagnostic capabilities during early stages of system design. For this purpose, the design space of an automotive electric/electronic architecture is explored in order to find feasible, optimal implementations, which satisfy traditional design objectives like performance and cost, and integrate the newly introduced structural diagnostic capabilities.

A feasible implementation is a valid solution of the design space exploration problem. It consists of an *allocation*, a *binding* and a *routing*. The allocation describes the subset of resources used in the implementation. The binding is a set of mappings that assign tasks to resources, while the routing describes the set of resources used in the communication of a message between any two resources.

In [Abelein14] functional applications are modeled using *mandatory* functional tasks, while BIST procedures are modeled using *optional* diagnostic tasks. The performance of diagnostic tasks is then quantified in order to evaluate optimal implementations. Since in [Abelein14] diagnositc tasks are scheduled for execution during operational *shut-off* of the vehicle's ECUs, the first optimization objective is to minimize the *shut-off time*. This ensures that the time interval between operational shut-off and real system power-down is kept within a few seconds. The second optimization objective is to maximize the *test quality*, which is a measure of the diagnostic capabilities of the system. This optimization goal quantifies the average stuck-at fault coverage achieved for all the integrated circuits in the ECUs of a given implementation. Finally, a *cost* optimization objective is minimized, which accounts for both the regular cost of the implementation's resources and the additional cost for the storage of diagnostic data.

As presented in [Abelein14], the characteristics of the allocation, binding, and routing of a feasible implementation can be encoded with binary variables and linear constraints. The resulting multi-objective discrete optimization problem can be solved with SAT-decoding [Lukasiewycz07].

In the case study presented in [Abelein14] four control-centric applications with 45 functional tasks and 41 messages have to be implemented using at most 15 ECUs. For each ECU, 36 mixed-mode BIST profiles are available. BIST profiles feature different properties in terms of the number of applied test patterns, the size of the response and deterministic test data, and the achieved fault coverage. At most one BIST profile is selected for each ECU. The design exploration procedure identifies several optimal implementations featuring different design tradeoffs between shut-off time, test quality, and cost. This methodology enables the system designer to choose the best alternative for his or her goals. The same methodology can also be used for the integration of other structural test procedures like, for example, software-based self-test (SBST) [Reimann14].

6.5. Summary and Outlook

This chapter presented a diagnostic BIST architecture which collects and stores intermediate test signatures on-chip. These signatures are later analyzed to conduct logic diagnosis. The diagnostic architecture is fully compatible with the traditional STUMPS architecture. Therefore, any test procedure for on-chip pattern generation, space compaction and "*X*"-masking can be used without modification.

The diagnostic architecture has been optimized for low hardware overhead. It holds highly compacted expected signatures (*response memory*), which are compared to the obtained intermediate signatures during test. When the observed test response does not match its expected value, the obtained signature is stored in non-volatile memory (*fail memory*) for later evaluation.

Signatures are composed of several test patterns comprising a test *window*. In order to support mixed-mode BIST with arbitrarily large pseudo-random or pseudo-exhaustive test pattern sequences, *strong* pattern windows are identified, which hold essential diagnostic information. Each such window contains a fixed number of patterns and is evaluated with the diagnostic approach of section 5.4. Contiguous test patterns that are not included in any strong window are compacted into a single *weak* signature. Weak signatures are used to improve defect coverage and refine diagnostic accuracy.

The diagnostic solution presented in this chapter fulfills the requirements of the automotive diagnostic use-case *Power-Up/Down Test*. Moreover, if diagnostic features are considered during system design, BIST execution may proceed under functional constraints to make use of any resources available in the system.

7. Results

The results evaluation in this chapter analyzes the performance of the presented approach to logic diagnosis using highly compacted test responses. The major goal pursued in this chapter is to assess the capabilities of the developed diagnostic procedures in order to make semiconductor failure analysis more accurate and efficient in complex embedded systems.

In the next section, the procedure to conduct logic diagnosis is explained and a set of target benchmark circuits are presented. After this, the performance of the diagnostic algorithms of chapter 5 is analyzed. For this purpose, both single and multiple faults are evaluated.

The last part of this chapter evaluates the hardware costs and diagnostic capabilities of the on-chip architecture of chapter 6. Hardware costs are quantified by the amount of information stored in the embedded memories required for the generation of input test patterns and intermediate test response signatures.

7.1. Diagnostic Setup

The general evaluation procedure in all diagnostic experiments consists of the following steps.

- Fault injection
  Once one or a few conditional stuck-at (CSA) faults are selected for injection, a suitable model C_{ud} of the device-under-diagnosis (DUD) is constructed. This model is simulated for every test pattern in the test set. These test responses are then compacted into test signatures.
- Fault candidate evaluation
  Fault simulation is performed with the fault-free circuit model C. One of the diag-

nostic algorithms of chapter 5 is then applied to identify which fault candidates are able to explain the test responses produced by C_{ud}.

- Fault ranking
  Fault candidates are ordered according to the extent to which they match the observed erroneous behavior. In this step, one or several ranked lists are created where the faults at the top of the lists are considered the most likely candidates that in fact represent a faulty chip location.

In all experiments, the size of the multiple-input signature register (MISR) is set to $n = 32$. Space compaction has also been employed in order to account for any number of scan chains. The space compactor has been designed to detect errors in any one, or any two, or any odd number of scan chains in the same scan-out cycle [Mitra04].

The test application process follows a mixed-mode BIST approach: inexpensive test patterns are generated on-chip and applied to the DUD. For the hard-to-detect stuck-at faults, deterministic (ATPG) patterns are produced. They may be applied either directly from a low cost tester, or encoded on-chip and reconstructed for test application.

7.1.1. Benchmark Circuits

The circuits investigated in this dissertations are industrial designs provided by NXP Semiconductors N.V.. These circuits are gate-level netlists. Scan chain configurations have been produced with a commercial tool. No layout or timing information has been considered.

Table 7.1 shows the characteristics of the target circuits. The first column shows the circuit name, the second and third columns show the number of gates and the total number of pseudo-primary inputs (PPIs) and pseudo-primary outputs (PPOs), respectively. The fourth column shows the scan chain count, while the fifth column presents the length of the longest scan chain. The last column shows the number of collapsed stuck-at faults in the circuit.

Circuit	# of gates	# PPIs and PPOs	# of scan chains	Longest scan chain	# of stuck-at faults
p45k	38811	2250	333	97	71848
p100k	84356	5829	270	53	162129
p141k	152808	10502	264	45	283548
p239k	224597	18495	260	61	455992
p259k	298796	18495	360	61	607536
p267k	239687	16621	260	62	366871
p269k	239771	16621	360	62	371209
p279k	257736	17835	385	59	493844
p286k	332726	17835	385	60	648044
p295k	249747	18521	330	62	472124
p330k	312666	17468	320	64	540758

Table 7.1.: Benchmark circuit characteristics

7.2. Logic Diagnosis with Compacted Test Responses

In this section, the performance of the diagnostic algorithms of chapter 5 are evaluated. Diagnostic performance is characterized with the *diagnostic accuracy*. The diagnostic accuracy is a measure of the percentage of faulty locations which are correctly identified. In order to account only for those faults, which may produce an observable deviation from the expected test response, diagnostic accuracy is defined as the ratio between correctly diagnosed faults and the total number of *detected* faults.

In the next section, several test pattern responses are compacted into a few intermediate test signatures under the premise that the circuit is affected by a single fault. This approach is beneficial when maximum response compaction is desired. Later, the diagnosis of multiple faults is evaluated. For this scenario one signature is produced for each test pattern.

7.2.1. Single Faults

For the diagnosis of single faults with highly compacted test responses, 32 test patterns are compacted into a single MISR signature. The diagnostic algorithm of section 5.4 is used to identify a single fault location. A test pattern set has been generated for mixed-mode BIST with 10000 pseudo-random test patterns (PRPs). The number of patterns in the test set and the achieved fault coverage are presented in Table 7.2.

Circuit	Fault coverage	# of test patterns
p45k	99.69 %	11746
p100k	99.55 %	10386
p141k	98.85 %	10670
p239k	98.83 %	10571
p259k	99.10 %	10734
p267k	99.58 %	10641
p269k	99.59 %	10642
p279k	97.87 %	10920
p286k	98.33 %	11311
p295k	99.14 %	12086
p330k	98.93 %	15514

Table 7.2.: Test pattern set characteristics

In this diagnostic setup a fault is said to be correctly diagnosed if it is identified as one of the top five candidates in the resulting ranked candidate list.

In order to evaluate the achievable diagnostic accuracy of the proposed compaction method, a total of 400 faults: 100 stuck-at faults, 100 crosstalk faults, 100 transition delay and 100 wired-and faults were randomly and uniformly injected into each circuit. For comparison, diagnostic accuracy is also calculated with test responses without compaction [Holst09b]. In both diagnostic scenarios it is assumed that all faulty test responses produced during test application are available for diagnosis[1].

Figure 7.1 shows the diagnostic accuracy for single faults, which accounts for all 400 faults injected into each circuit. Tables A.1 and A.2 of Appendix A show the detailed diagnostic accuracy both for the procedure of section 5.4 based on systems of linear equations (SLE) and for the approach of [Holst09b], respectively. In the general case, as pointed out in [Cheng07], diagnostic accuracy with compacted test signatures drops only a few percentage points in comparison with diagnosis without compaction at all. For circuit *p295k* the reduction in diagnostic accuracy amounts roughly to 8,75 %. However, the diagnostic results without compaction are overly optimistic, as only a few uncompacted failing responses are usually downloaded from the tester to conduct logic diagnosis.

Figure 7.2 shows the diagnostic accuracy achieved with the method of section 5.4 for

[1] For diagnosis without compaction only a fixed number of faulty responses are usually stored in the tester for subsequent analysis. Therefore, the diagnostic accuracy in this case is optimistic.

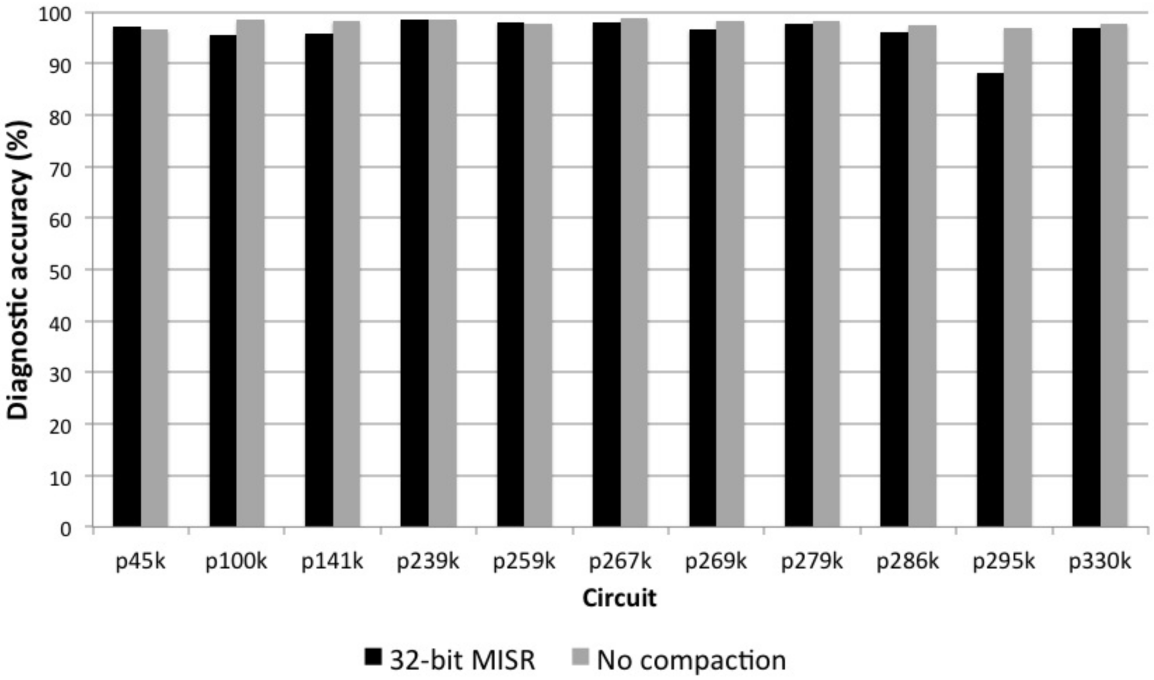


Figure 7.1.: Average diagnostic accuracy of single faults

all fault models in consideration. As the figure shows, this procedure is efficient not only for stuck-at faults, but also for *non-target faults*. Non-target faults are those faults which were not explicitly targeted during test pattern generation. Therefore, their activation conditions in the test sequence is not considered *a priori*.

Figure 7.3 shows the average diagnostic runtime of a single fault for each of the considered circuits. Logic diagnosis was conducted with a pool of heterogeneous computers, including compute servers and personal workstations. A typical computer has the following characteristics: Quad-core Intel Core i7 processor with an operating frequency of 3.4 GHz and 16 GB of RAM.

For the largest circuit the average runtime is less than 100 seconds, while for the smallest one the average runtime is approximately 9 seconds.

The results presented in this section show that the diagnostic algorithm of section 5.4 achieves excellent accuracy for single faults with reasonable computational demands. As this diagnostic procedure makes use of intermediate signatures comprising several test patterns, it allows logic diagnosis with larger response compaction ratios than those typically supported by state-of-the-art solutions. For example, in comparison with [Cheng07], the compaction ratio is increased by a factor of 32. Similarly, a

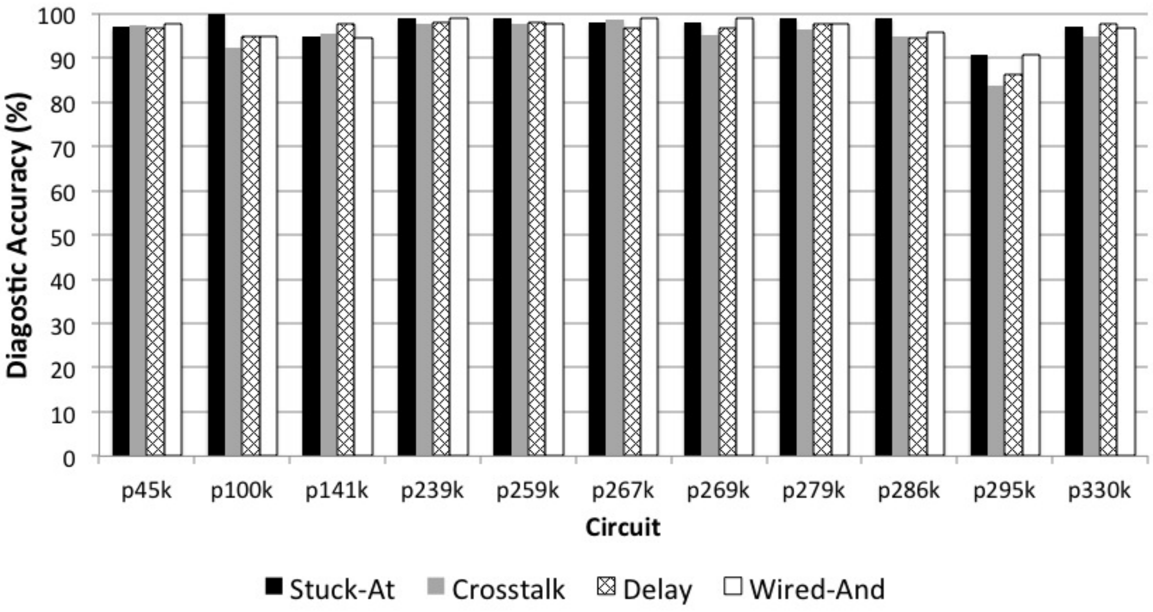


Figure 7.2.: Detailed diagnostic accuracy of single faults

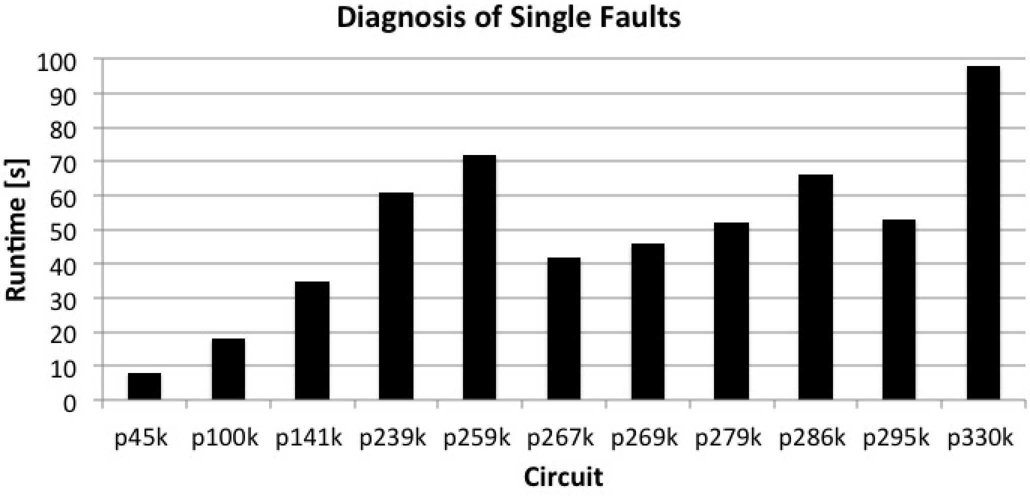


Figure 7.3.: Average diagnostic runtime for single faults

fivefold compaction ratio improvement is achieved over [Elm12] without the need for special design-for-test (DfT) resources and dedicated ATPG procedures.

7.2.2. Multiple Faults

In order to analyze the diagnostic capabilities of the procedures described in section 5.5, a total of 600 diagnostic experiments have been conducted for each circuit. In these experiments it is assumed that each test pattern in the test set is compacted into a single signature. As in the previous section, it is further assumed that all erroneous test responses are available for diagnosis.

For each diagnostic experiment, 5 faults are activated with a probability p ranging from 0.01 to 1.0. In order to evaluate test responses under the influence of multiple active faults, either all 5 faults or none of them are are activated in a given pattern. This way, patterns explained by a single fault are less frequent and logic diagnosis becomes more elaborate. Additionally, in order to consider masking and reinforcement effects between injected faults, two sets of experiments are performed: in the *uniform* diagnostic setup, the location of the faults are chosen randomly, while in the *topological* setup, faults are injected within the input or output cone of a randomly chosen location in the circuit.

Figure 7.4 shows the percentage of explained patterns in the test set for both diagnostic setups. As the figure shows, almost all faulty signatures provide some useful diagnostic information with the novel approach based on SLE, while any SLAT-based method can only account for roughly 60 % of the faulty test responses. In the next sections, diagnostic accuracy is detailed in the presence of multiple faults.

Coarse-grained Diagnosis

The diagnostic algorithm of section 5.5.4 is executed for two iterations ($Q = 2$) and 10 faults in the candidate set ($k = 10$). A fault is said to be correctly diagnosed if a fault candidate in its fanout node appears within the top five most likely candidates in any of the candidate lists.

Figure 7.5 compares the average diagnostic accuracy over all activation probabilities achieved both with the coarse-grained diagnostic method of section 5.5.4 and with a SLAT-based algorithm. For each circuit, four values are provided for the diagnostic accuracy:

- uniform-LSE: diagnostic accuracy for the uniform diagnostic setup with the pro-

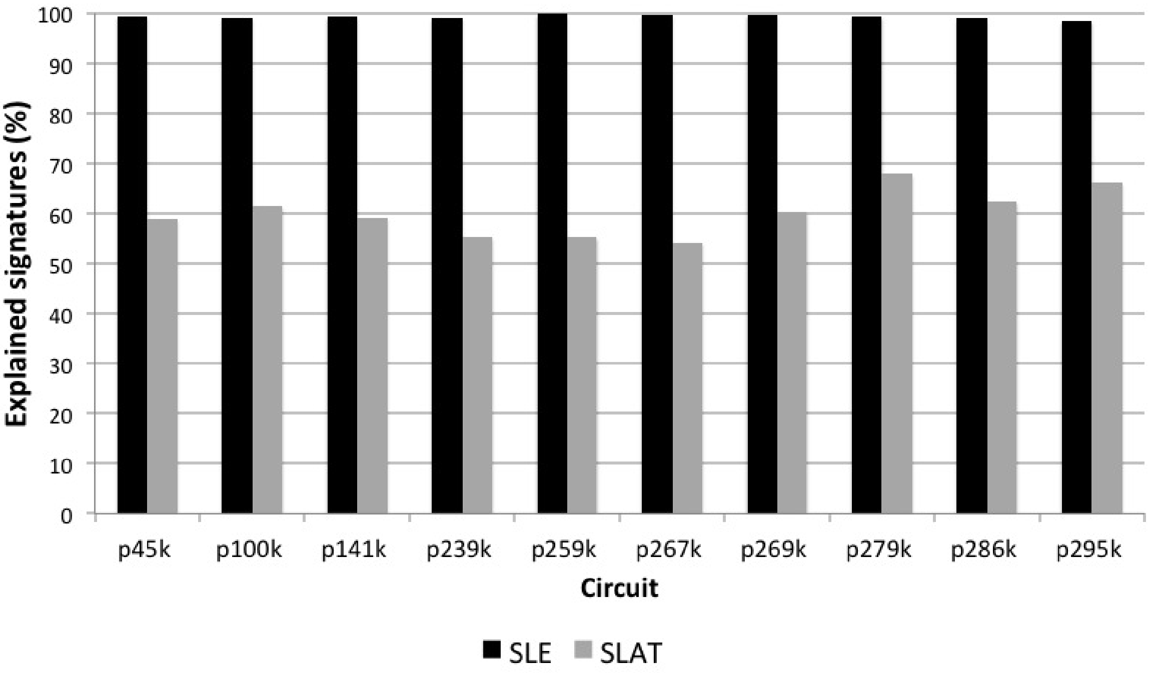


Figure 7.4.: Diagnostic experiments for multiple faults: Explained signatures in the test pattern set

posed algorithm using systems of linear equations (SLE).

- uniform-SLAT: diagnostic accuracy for the uniform diagnostic setup with an algorithm based on the single location at a time (SLAT) paradigm.
- topological-LSE: diagnostic accuracy for the topological diagnostic setup with the proposed algorithm using systems of linear equations (SLE).
- topological-SLAT: diagnostic accuracy for the topological diagnostic setup with an algorithm based on the SLAT paradigm.

These results show that the presented signature-based approach achieves an average accuracy improvement of roughly 12 % when compared to state-of-the-art SLAT-based solutions.

Figures 7.6 and 7.7 below give a more detailed insight and show the diagnostic accuracy of the novel methodology based on SLE for activation probabilities of 0.05 and 1.0 separately. Table A.2 shows detailed results for all considered activation probabilities.

It can be seen that the presented approach is efficient for both low activation probability ($p = 0.01$) and deterministic activation ($p = 1.0$). In both cases it outperforms SLAT

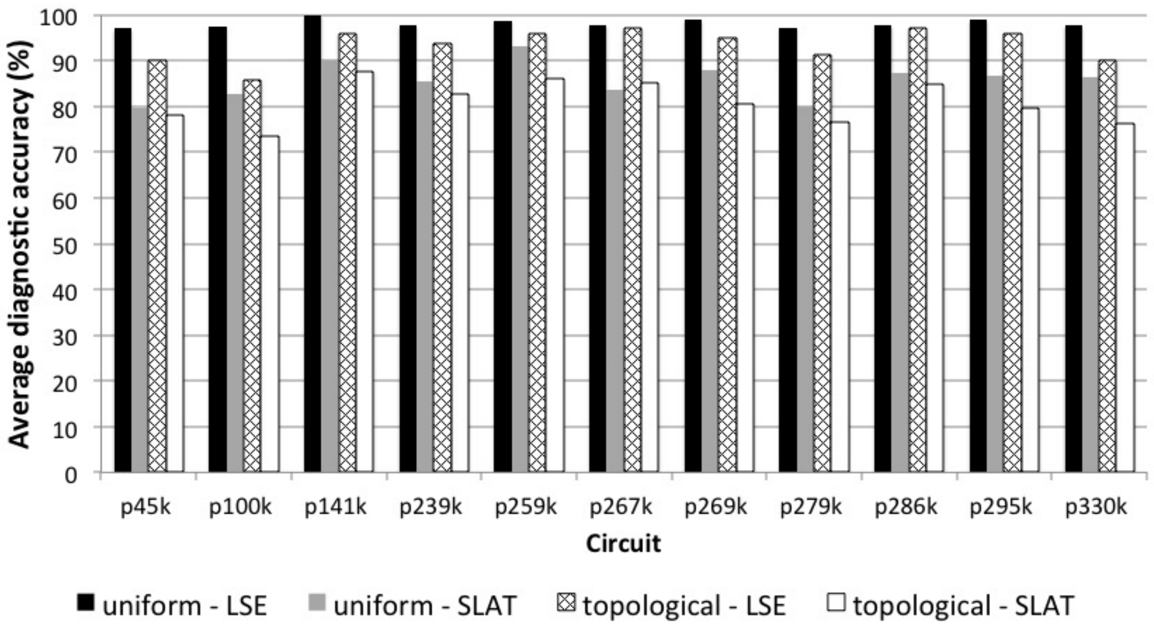


Figure 7.5.: Average diagnostic accuracy of multiple faults: Coarse-grained diagnosis

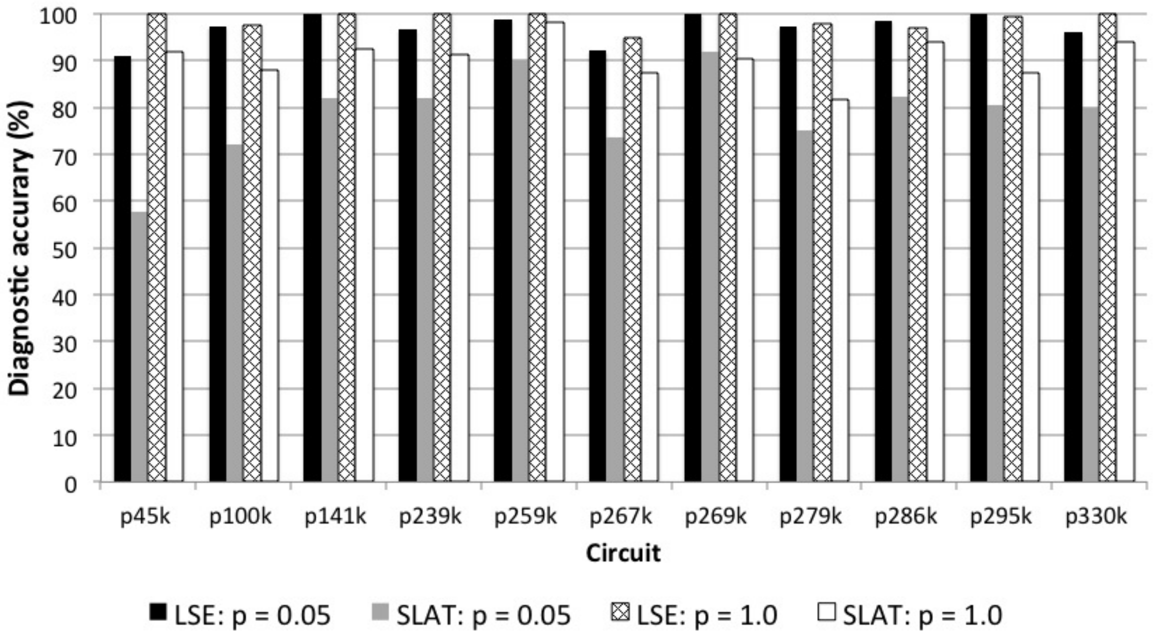


Figure 7.6.: Coarse diagnostic accuracy of multiple faults for $p = 0.05$ and $p = 1.0$: Uniform setup

significantly. The figure shows as well that the performance of the SLAT approach improves for a larger activation probability. This is due to the fact that the more often faults are activated, the more likely it is that at least one SLAT pattern exists for each

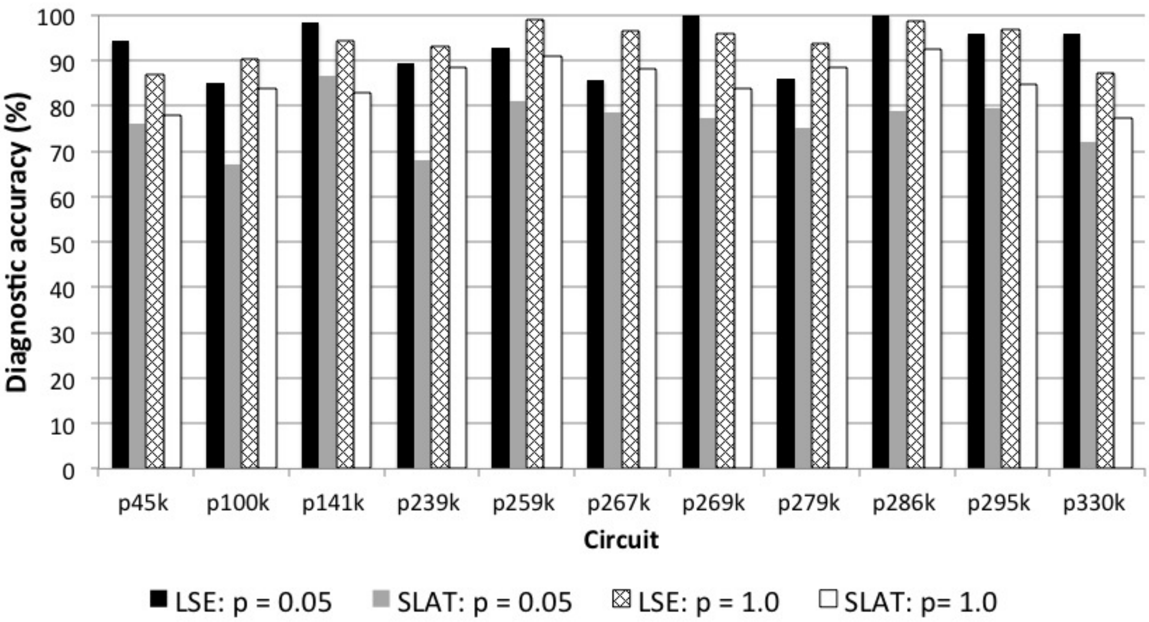


Figure 7.7.: Coarse diagnostic accuracy of multiple faults for $p = 0.05$ and $p = 1.0$: Topological setup

fault and, therefore, it can be easily diagnosed. Unsurprisingly, the topological setup is the most difficult to diagnose. In particular, the performance of the SLAT-based approach is lowest for this setup with $p = 0.05$. The accuracy in this case is improved on average by 16 %.

The greatest improvement in the uniform setup is achieved for circuit p45k where the the coarse-grained diagnostic algorithm outperforms SLAT by a margin of 33.1 %. For the topological setup, the greatest improvement is achieved for circuit p279k and amounts to 22.5 %.

Figure 7.8 shows the average diagnostic runtime for each circuit. As the figure shows, the time required for the diagnosis of multiple faults ranges from roughly 12 seconds in circuit p45k to approximately 189 seconds for circuit 330k. The average runtime amounts to 74 seconds.

Fine-grained Diagnosis

The diagnostic algorithm of section 5.5.4 is employed to identify the exact location of the injected faults. An injected fault is said to be correctly diagnosed if it appears

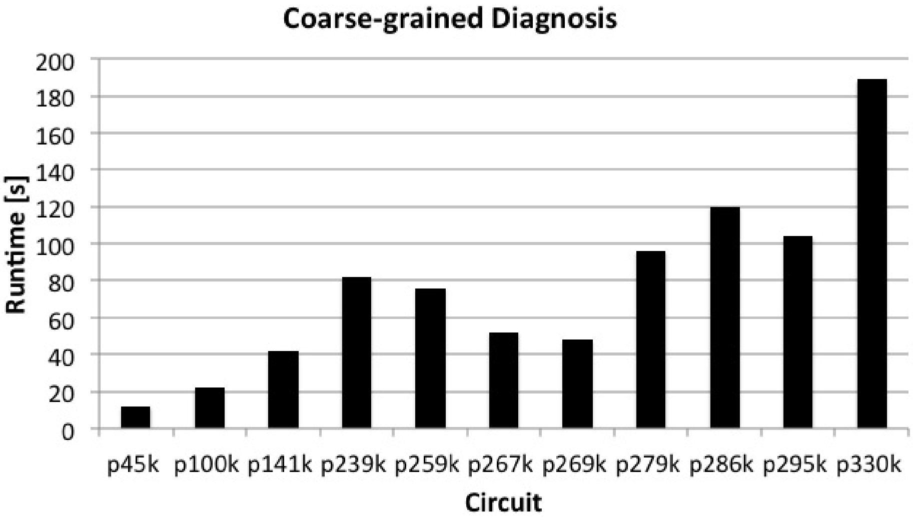


Figure 7.8.: Average runtime of coarse-grained diagnosis

within the top five most likely fault candidates in any of the candidate lists. Like in the previous section the diagnostic algorithm is executed with $Q = 2$ and $k = 10$.

Figure 7.9 shows the diagnostic accuracy achieved with systems of linear equations and with a SLAT-based solution. These experiments make use of the same fault set employed in the previous section to evaluate coarse-grained diagnostic results. As the figure shows, for both the uniform and topological setups, the fine-grained diagnostic procedure continues to achieve an average improvement of roughly 12 % in diagnostic accuracy as its coarse-grained counterpart. The exact diagnostic performances are presented in tables A.9 and A.10. However, as the most likely fault locations are directly identified, instead of faulty fanout-free regions, subsequent failure analysis can proceed with a reduced number of suspect defect sites in the chip.

Figures 7.10 and 7.11 show the diagnostic accuracy of the fine-grained diagnostic procedure with activation probabilities $p = 0.05$ and $p = 1.0$ for both the uniform and topological setups, respectively. As with coarse-grained diagnosis, the fine-grained diagnostic algorithm efficiently handles low activation probabilities as well as deterministic behavior. As expected, in comparison with figures 7.6 and 7.7, fine-grained diagnosis exhibits similar behavior: For the most difficult diagnostic scenario, namely, the topological setup with $p = 0.05$, the average improvement in diagnostic accuracy over SLAT is roughly 15 %.

In the uniform setup the greatest accuracy improvement of 30.9 % is achieved for

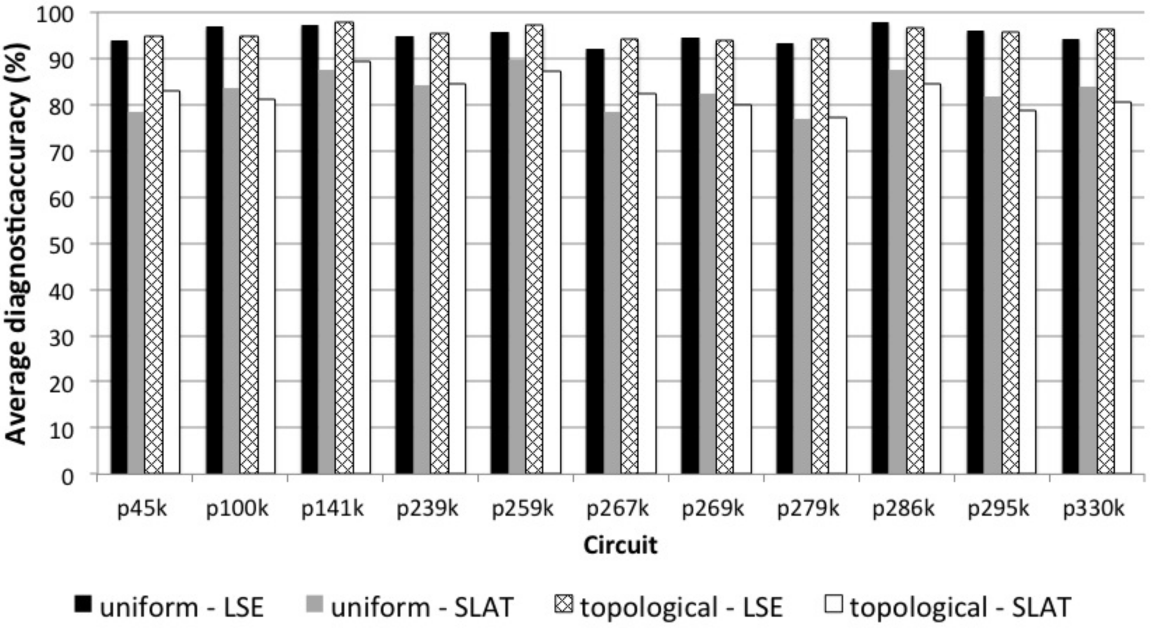


Figure 7.9.: Average diagnostic accuracy of multiple faults: Fine-grained diagnosis

circuit p45k, while in the topological setup, the greatest improvement is achieved for circuit p286k and amounts to 21.0 %.

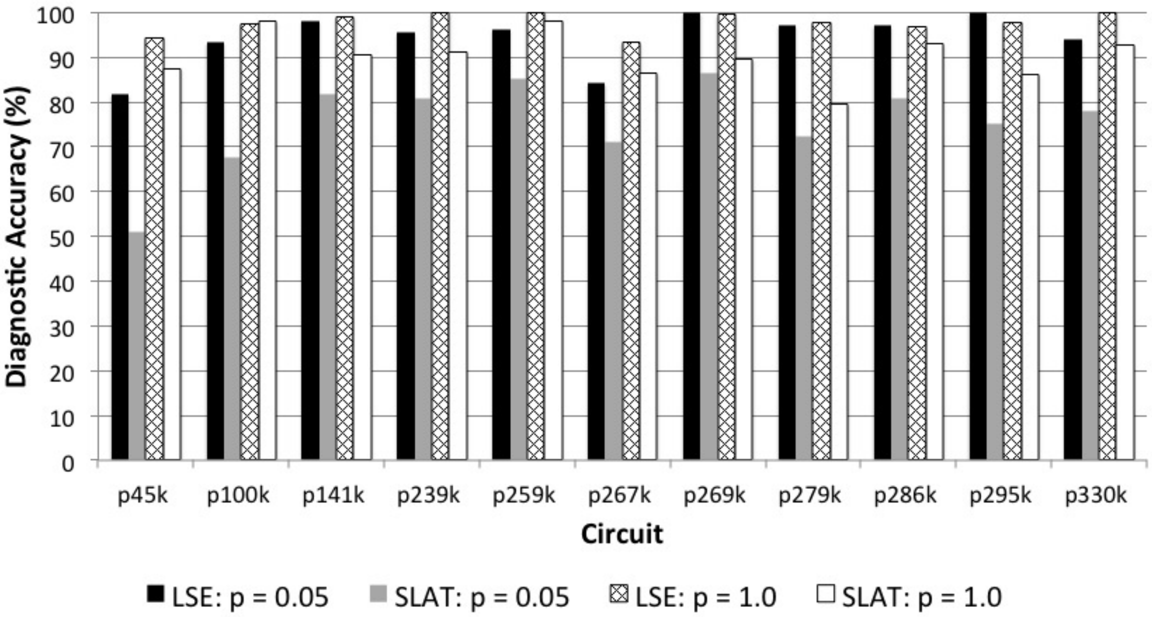


Figure 7.10.: Diagnostic accuracy of multiple faults for $p = 0.05$ and $p = 1.0$: Uniform setup

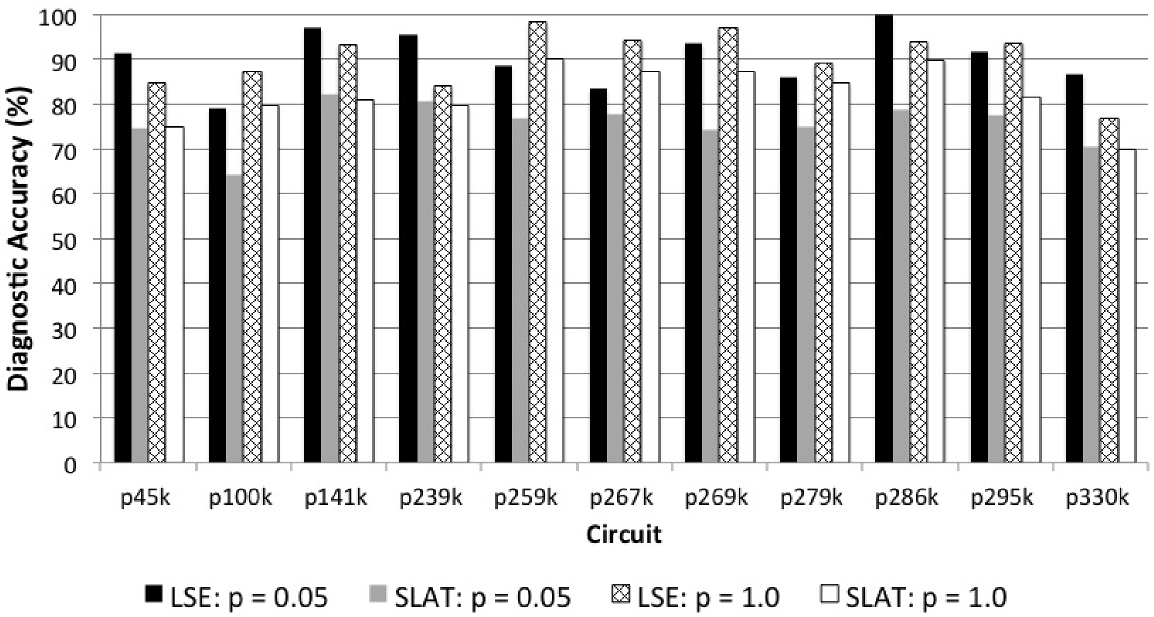


Figure 7.11.: Diagnostic accuracy of multiple faults for $p = 0.05$ and $p = 1.0$: Topological setup

Figures 7.12 and 7.13 compare the diagnostic accuracy of the coarse-grained and fine-grained diagnostic procedure for the uniform and diagnostic setups, respectively. For the uniform setup, shown in Figure 7.12, the accuracy is slightly higher for coarse-grained diagnosis. That is, in a few cases the corresponding fanout node of a faulty gate can be correctly identified, while the actual fault location is not regarded as a likely fault candidate. Interestingly enough, this situation is reversed for the topological setup. This can be explained by the interaction between the injected faults in both configurations: In the uniform setup faults are likely to have independent fanout nodes. Therefore, the coarse-grained diagnostic procedure can achieve excellent diagnostic accuracy, since it only needs to identify any fault candidate within a fanout-free region (FFR) for successful diagnosis. Conversely, in the topological setup, faults are more likely to interact with one another and share a common fanout node. In this case, the fine-grained diagnostic procedure is better suited to distinguish all fault candidates and yield better accuracy.

Finally, Figure 7.14 shows the average runtime of the fine-grained diagnostic procedure. On average the computational time for fine-grained diagnosis was approximately 160 seconds, roughly twice as computationally expensive as its coarse-grained counterpart. However, these runtimes are well within the range of the typical compu-

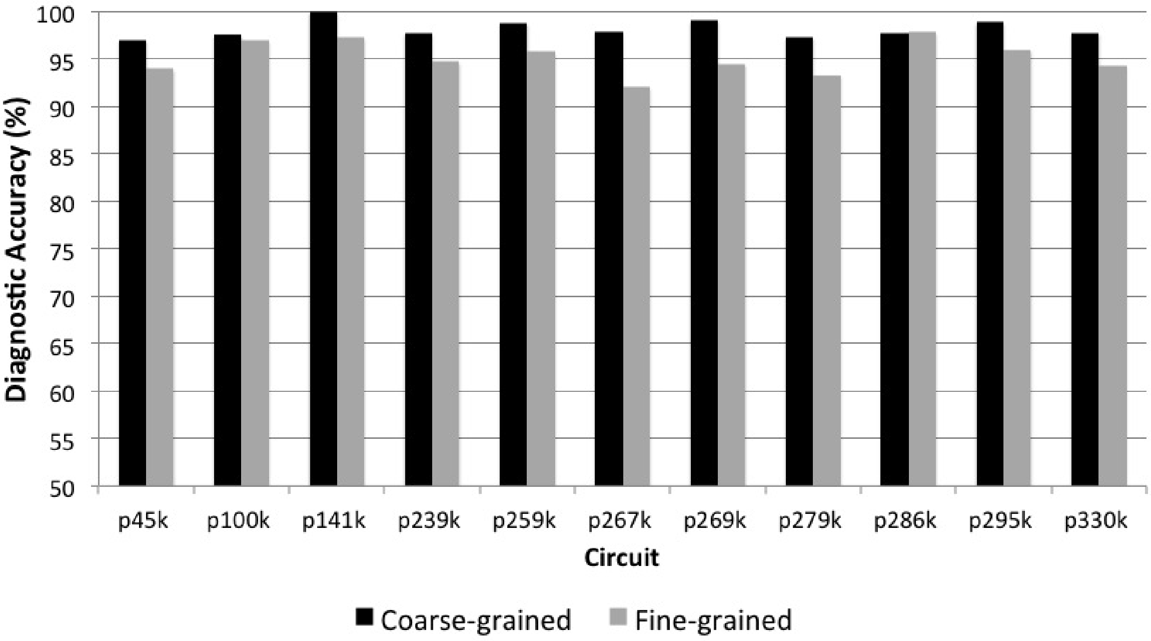


Figure 7.12.: Comparison between coarse-grained and fine-grained diagnostic accuracy: Uniform setup

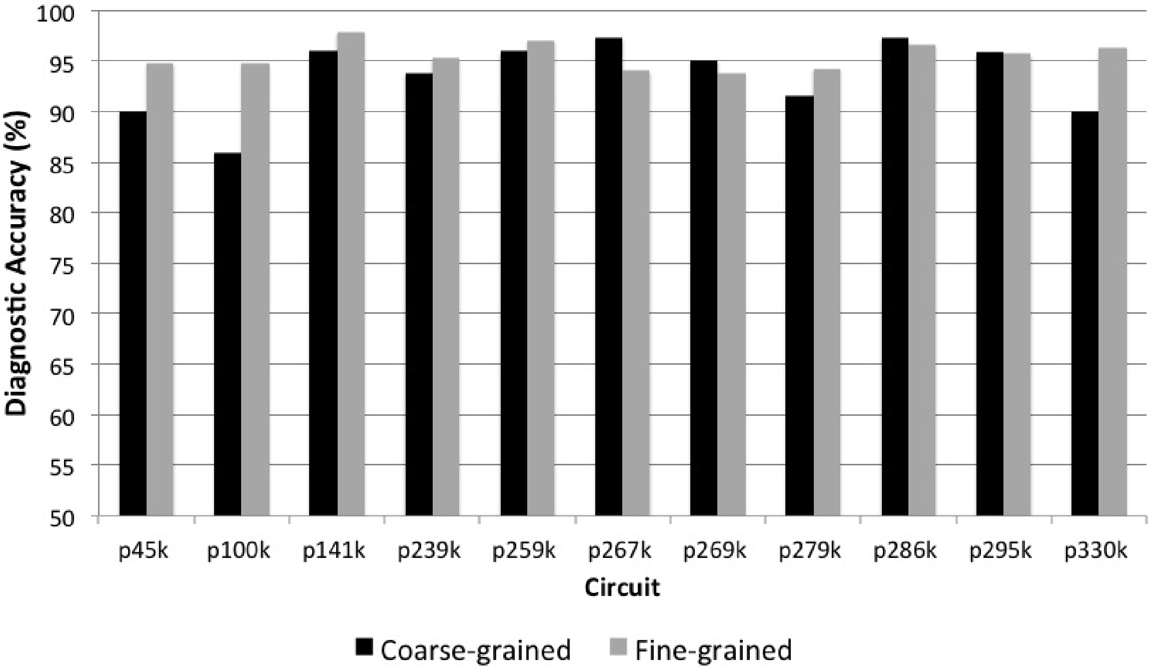


Figure 7.13.: Comparison between coarse-grained and fine-grained diagnostic accuracy: Topological setup

tational effort for volume diagnosis [Blanton12].

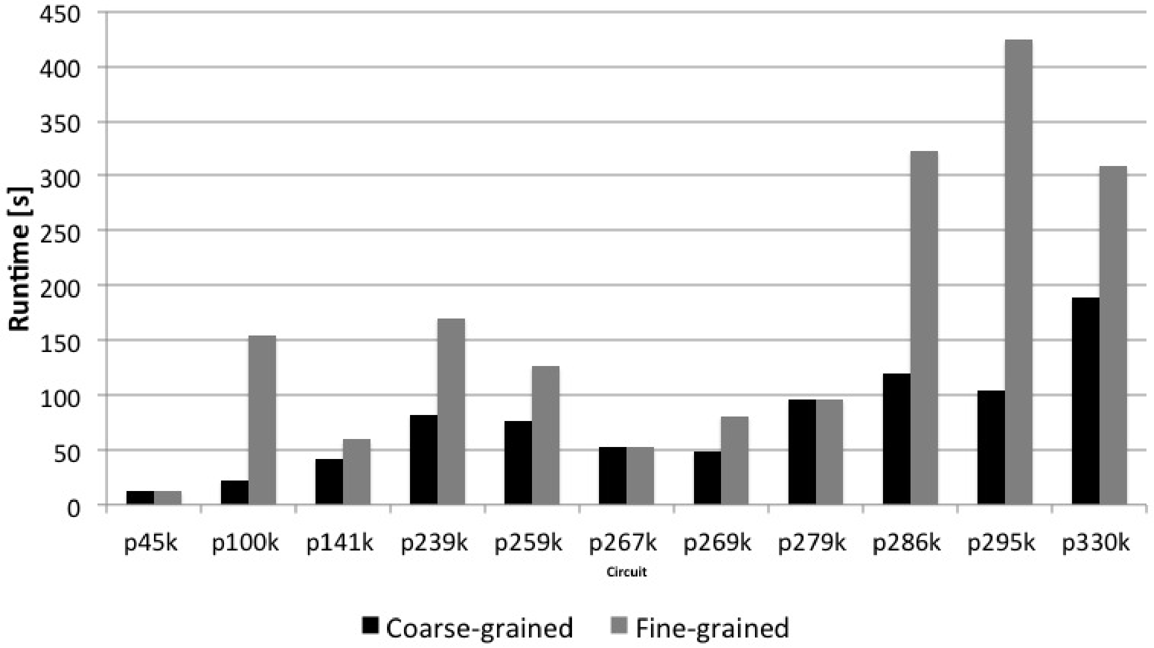


Figure 7.14.: Comparison between coarse-grained and fine-grained diagnostic runtime

7.3. Test Architecture for Built-In Diagnosis

The diagnostic architecture of chapter 6 has been evaluated for the industrial benchmark circuits of section 7.1.1. In all experiments reported next, strong diagnostic windows always contain 32 patterns ($k = 5$). The number of observed bits in the MISR signature l is set to 8, while the depth of the fail memory g in Figure 6.3 has been fixed to 100.

A mixed-mode approach for test pattern generation has been analyzed for 4096 and 100000 PRPs, as well as for partial pseudo-exhaustive test (P-PET) with a maximum input cone size b of 24 (see section 2.5.1). Table 7.3 shows the achieved fault coverage and compares the number of deterministic patterns applied for each mixed-mode test sequence.

Circuit	Fault coverage	# of deterministic patterns		
		4096 PRPs	100000 PRPs	P-PET
p45k	99.70 %	1940	303	16
p100k	99.56 %	414	118	14
p141k	98.85 %	704	494	219
p239k	98.84 %	618	431	180
p259k	99.19 %	830	523	214
p267k	99.60 %	678	578	434
p269k	99.58 %	693	533	433
p279k	97.89 %	917	668	343
p286k	98.34 %	1511	935	549
p295k	99.15 %	2553	748	662
p330k	98.95 %	5587	5191	4490

Table 7.3.: Fault coverage and number of deterministic patterns for mixed-mode BIST

7.4. Hardware Costs

Figure 7.15 compares the number of specified bits in the deterministic part of the mixed-mode test sequence. The number of specified bits is used as an estimate for the *seed* memory for test pattern decoding. As the figure shows, the on-chip storage requirements for deterministic pattern generation can be greatly reduced as more inexpensive test patterns are applied.

Table 7.4 presents the amount of information required for the storage of response signature data when P-PET is used, while table 7.5 presents the same information for 4096 and 10000 PRPs.

As eight bits of a response signature are observed ($l = 8$), each strong or weak window requires 1 byte of storage. In the case of P-PET, the generation of strong and weak windows is controlled with a window index that uniquely identifies an aligned 32-pattern block as described in section 6.3.2. The amount of control data shown in the fourth column of table 7.4 is calculated as follows:

$$\left\lceil log_2 \left(\frac{\text{\# P-PET patterns}}{32} \right) \right\rceil \cdot \text{\# strong windows}$$

For 100000 PRPs the control data amounts to a fixed cost of 391 Bytes. This cost stems from a 1-bit tag required for each window in the pattern set to identify strong windows.

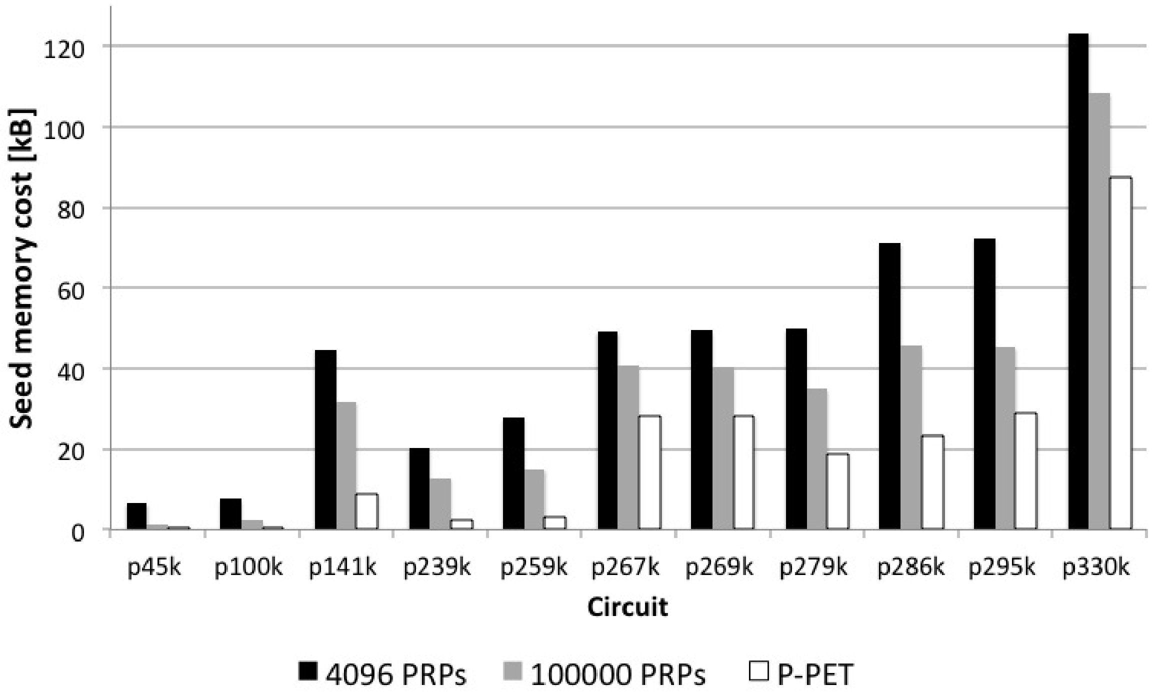


Figure 7.15.: Seed memory costs for mixed-mode BIST

The use of 4096 PRPs does not require any control overhead to manage the application of strong and weak windows. This is due to the fact that, with such a short sequence of inexpensive patterns, all pattern windows contain at least one essential pattern and they are all treated as strong windows. Therefore, the total cost in the second column of table 7.5 is calculated with the expression:

$$\left\lceil \frac{4096 + \#\text{ deterministic patterns}}{32} \right\rceil$$

Figure 7.16 shows the total diagnostic cost for P-PET, 4096 PRPs, and 100000 PRPs. The total cost is the sum of the seed memory cost in Figure 7.15 and the total cost for the generation of response signature data shown in the last column of tables 7.4 and 7.5. As the figure shows, the use of 100000 PRPs results in lower overall costs compared to 4096 PRPs. This is due to the decreased cost for the seed memory. For circuits p141k and p330k, P-PET has the lowest overall cost. Although the overall cost for P-PET is higher for the other circuits, the reduction in seed memory can compensate the growing response memory to a large extent and keep it in a acceptable range. As detailed in the next section, the use of P-PET guarantees better defect coverage and

Circuit	# of strong windows	# of weak windows	Control data [Byte]	Total cost [Byte]
p45k	1844	1029	4841	7714
p100k	2001	1239	5253	8493
p141k	3560	2847	9790	16197
p239k	3928	3046	10802	17776
p259k	4816	3547	6923	15286
p267k	3731	2801	10261	1693
p269k	3746	2795	10302	16843
p279k	5179	3794	14243	23216
p286k	8185	5996	24555	38736
p295k	7730	5397	21258	34385
p330k	3250	3949	9344	16343

Table 7.4.: Control and response data for P-PET

	4096 PRPs	100000 PRPs			
Circuit	Total cost [Byte]	# of strong windows	# of weak windows	Control data [Byte]	Total cost [Byte]
p45k	189	2785	504	391	3680
p100k	141	1469	400	391	2140
p141k	150	1260	526	391	2177
p239k	148	1405	524	391	2320
p259k	154	1694	541	391	2626
p267k	150	1325	496	391	2212
p269k	150	1350	477	391	2218
p279k	157	1776	593	391	2760
p286k	176	2182	517	391	3090
p295k	208	2575	415	391	3381
p330k	303	1215	508	391	2114

Table 7.5.: Control and response data for pseudo random patterns (PRPs)

diagnostic performance.

7.4.1. Diagnostic Accuracy

In order to analyze achievable diagnostic accuracy, the fault set of section 7.2.1 comprising 100 stuck-at faults, 100 crosstalk faults, 100 transition delay, and 100 wired-AND faults, is diagnosed solely with the data in the fail memory. The diagnostic ac-

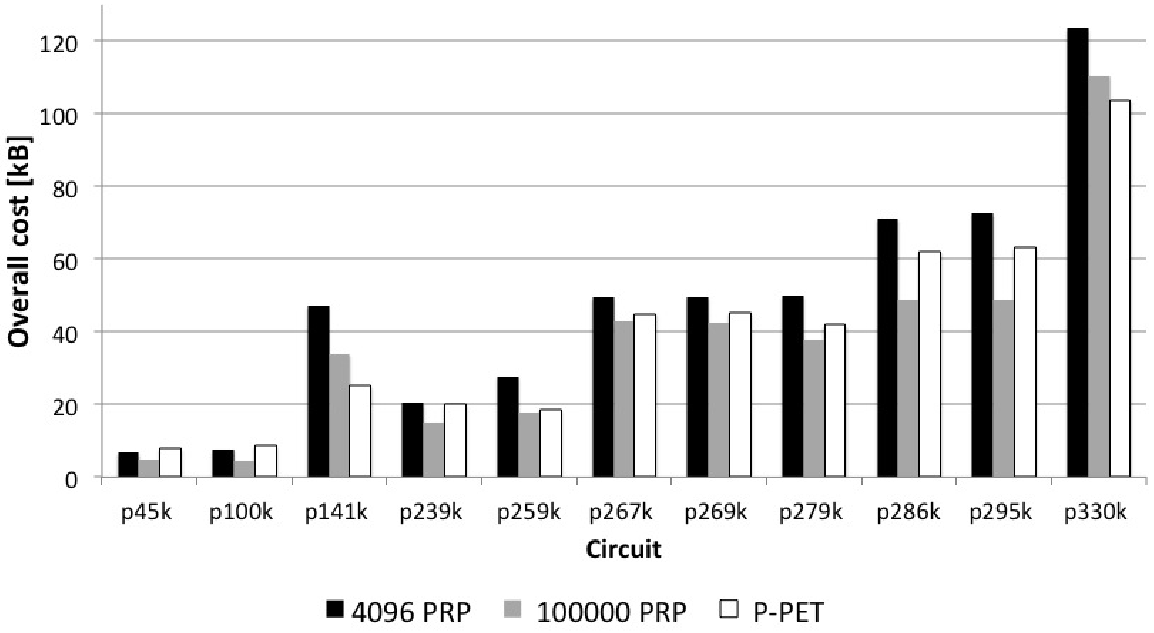


Figure 7.16.: Diagnostic architecture: Overall costs

curacy is defined as the ratio between the number of correctly diagnosed faults and the number of *injected* faults. A fault is considered as correctly diagnosed, if it is one of the top 5 fault candidates in the ranked list after the responses in the fail memory have been analyzed.

Figure 7.17 shows the average diagnostic accuracy. As more patterns are applied, better diagnostic accuracy can be achieved: the use of P-PET exhibits the best performance in almost all cases, while 100000 PRPs achieve better diagnostic results than 4096 PRPs. Results for each fault model, as well as the contribution of weak windows to the diagnostic accuracy are provided in tables A.12 to A.14 of of Appendix A.

In order to evaluate the diagnostic accuracy of non-target faults, Figure 7.18 analyzes the crosstalk fault model for PRPs and P-PET. The results for the rest of the non-target faults show similar performance. As expected, the use of more test patterns brings about higher accuracy. In most cases, the diagnostic accuracy achieved with P-PET is the highest. Interestingly enough, as some deterministic patterns may be beneficial to distinguish certain fauls, the achieved diagnostic accuracy with 10000 PRPs may be slightly larger than that achieved with P-PET for a few circuits.

Figure 7.19 illustrates the performance of strong and weak windows in the detection of non-target faults. The figure shows that weak windows take advantage of the de-

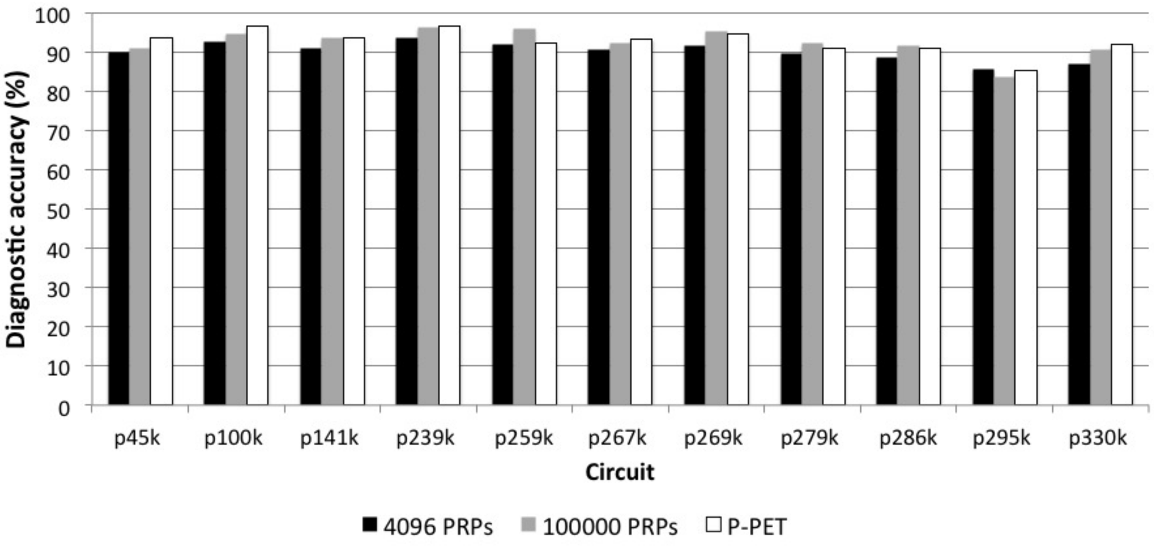


Figure 7.17.: Diagnostic accuracy with pseudo-random patterns (PRPs) and P-PET

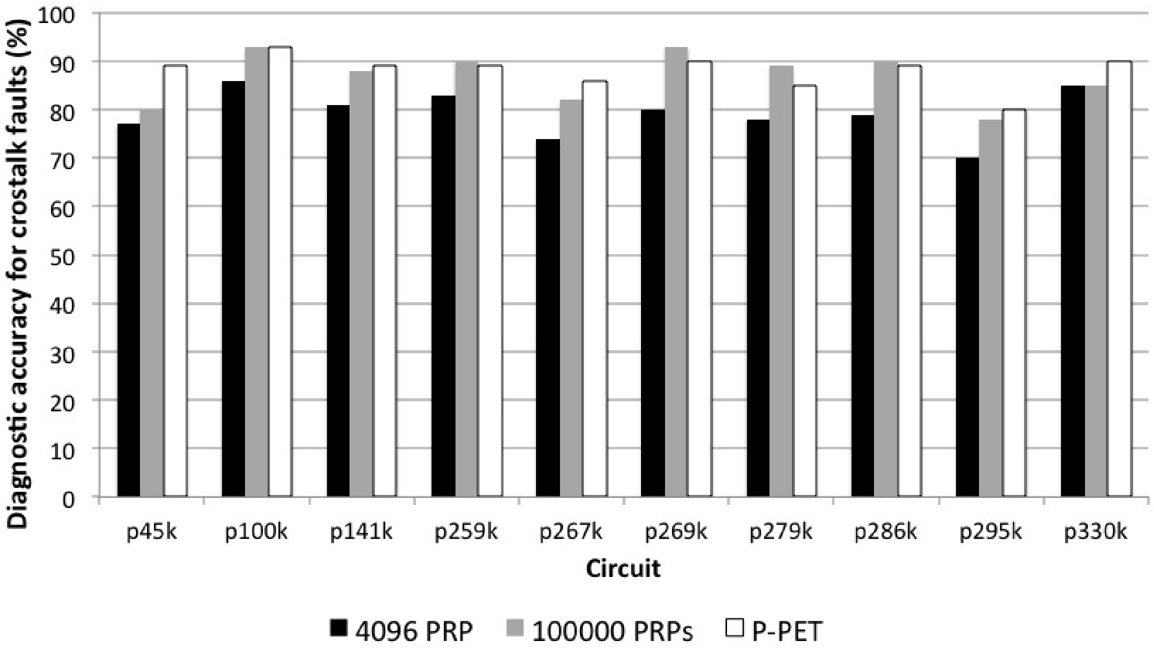


Figure 7.18.: Diagnostic accuracy for crosstalk faults with pseudo-random patterns (PRPs) and PPET

fect coverage of the test pattern set. As expected, P-PET detects the most non-target faults and the shortest sequence of 4096 PRPs provides the least coverage. Table A.15 quantifies the contribution of weak windows to the detection of non-target faults.

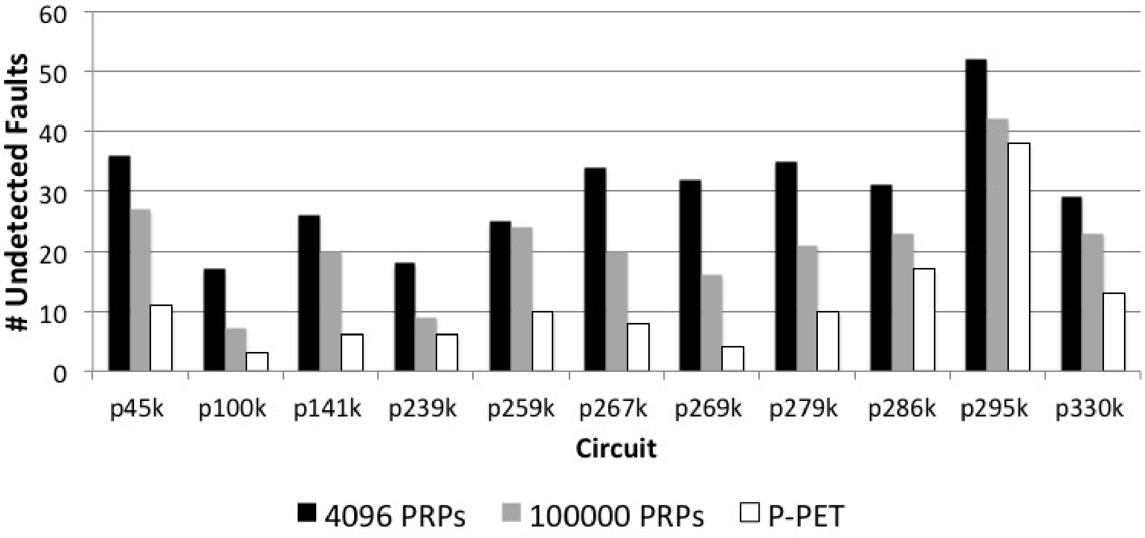


Figure 7.19.: Diagnostic architecture: Undetected faults with P-PET and pseudo-random patterns (PRPs)

7.5. Summary and Outlook

This chapter evaluated the performance of the algorithms for logic diagnosis developed in this dissertation. A wide array of spot defect mechanisms were simulated and analyzed using industrial circuits.

The results in this section shed light into the diagnostic performance when only compacted test responses are considered during logic diagnosis. In this evaluation, the diagnostic problem is tackled by solving systems of linear equations.

Diagnostic results show that several test patterns can be compacted into a single signature and yet obtain excellent diagnostic accuracy. Under the assumption that a circuit is affected by a single fault, diagnosis based on linear equations can achieve almost the same performance as state-of-the-art solutions without test response compaction. The developed diagnostic approach can handle larger compaction ratios than those supported in the literature for compacted test responses.

The diagnosis of multiple arbitrary faults can also be improved by solving systems of linear equations. Experimental results show an average improvement of 12 % with respect to state-of-the-art solutions. Moreover, the number of test responses which convey useful diagnostic information can be drastically increased from 60 % in traditional approaches to roughly 99 %. Consequently, the diagnostic properties of the test pattern set can be better exploited.

This chapter also evaluated the architecture for built-in diagnosis presented in chapter 6. The hardware costs and the diagnostic performance of this architecture were analyzed for industrial circuits under several mixed-mode BIST profiles.

The presented diagnostic architecture is able to collect response signatures during autonomous test at a reasonable hardware cost. Experimental results show the the use of a larger sequence of inexpensive patterns greatly reduces the amount of information required for input test pattern generation while demanding very little hardware overhead to support logic diagnosis. Similarly, the proposed architecture takes full advantage of the defect coverage in the generated pattern sequence.

Finally, the information provided by the diagnostic architecture provides an average diagnostic accuracy of over 90 %. The diagnostic accuracy grows when more inexpensive patterns are applied.

8. Conclusions and Future Work

The failure analysis process in the automotive domain has so far relied on functional tests to discover malfunctions and diagnose the root cause of any semiconductor problem in the field. However, as electronic automotive systems become extremely complex, product quality can only be guaranteed with the introduction of structural tests, which quantify test effectiveness and enable efficient structural diagnosis.

Structural diagnosis identifies faulty components in the system. Therefore, it offers the opportunity to enhance the automotive failure analysis process and take timely corrective actions to improve product quality. With this goal in mind, the diagnostic use-cases elicited in this dissertation reflect realistic diagnostic scenarios with specialized characteristics and requirements in terms of test access, diagnostic capabilities and costs.

The access to on-chip test infrastructure is tackled in this dissertation with the help of available functional system resources. They serve as a regular piece of semiconductor test equipment to bring a device-under-diagnosis (DUD) to a special test mode, apply structural tests and recover test responses. As presented in an industrial case study, the cost-effective reuse of design-for-test resources in the DUD can be realized by combining test and functional interfaces at the system level for the application of structural tests.

The presented diagnostic procedures for arbitrary faults make use of highly compacted test signatures in order to identify one or a few faulty locations in the chip. The developed diagnostic algorithms rely on the conditional stuck-at (CSA) model in order to construct systems of linear equations, which reflect faults and activation conditions that reproduce the observed faulty signatures collected from the DUD. As a consequence, almost all failing signatures provide some relevant diagnostic information. This diagnostic technique can be exploited to achieve excellent response compaction ratios without negatively affecting diagnostic results. On the one hand, under the single fault assumption, the diagnostic procedures in this dissertation offer an improvement of at least one order of magnitude in compaction ratio with respect to state-of-the-art solutions. On the other hand, they improve the diagnosis of multiple faults by over 12% with regard to traditional approaches for compacted test responses.

This dissertation has presented for the first time a diagnostic on-chip test architecture to collect structural diagnostic data during built-in self-test (BIST), which is also fully compatible with the traditional STUMPS architecture. Diagnostic costs in terms of area overhead are reduced by storing only very short fault-free reference signatures. Similarly, mixed-mode BIST is also supported in order to achieve sufficient fault and defect coverage with reasonable diagnostic costs: Only essential patterns in the test set demand dedicated on-chip resources.

The diagnostic methods presented in this dissertation offer a unified solution to overcome the challenges of the largest part of the identified diagnostic use-cases. Specifically, the requirements in *Workshop Test*, *Semiconductor Failure Analysis* and *Power-Up/Down Test* are fully satisfied.

Workshop test can benefit from the presented diagnostic methods, since they enable the application of structural tests using traditional equipment and interfaces usually employed in the automotive domain. Moreover, diagnostic information from an integrated circuit (IC) can be collected in the workshop without direct pin access.

Semiconductor failure analysis of field returns can be made more accurate and efficient, as logic diagnosis of arbitrary faults can be performed without removing an IC from its board. This can drastically improve the time required to deal with semiconductor issues during semiconductor manufacturing and test, without disturbing the traditional and well-established diagnostic flow in the automotive domain.

Finally, valuable structural diagnostic information can be gathered in a BIST session executed during power-up/down test. The collection of such information proceeds autonomously without the intervention of any external test equipment. Therefore, it executes structural tests under very similar conditions the IC experiences during regular vehicle operation. Consequently, the number of “no trouble found (NTF)” occurrences can be greatly diminished.

The methods presented in this dissertation lay the fundamental technical aspects for the adoption of structural diagnostic approaches in the automotive domain. They complement traditional functional approaches to overcome the diagnostic challenges arising from the pervasive use of semiconductors to realize complex features in today's and future automotive vehicles.

8.1. Further Challenges in the Automotive Domain

Structural diagnostic methods offer accurate and efficient tools for the analysis of failing vehicles. This process is able to provide valuable diagnostic insight throughout the complete vehicle life-cycle, from early prototyping to service and maintenance. As briefly discussed in the introductory chapter, this knowledge gathered by means of the introduced structural diagnostic methods in this dissertation can be exploited in order to improve both design and test procedures of manufactured ICs. For this purpose, massive amounts of information need to be analyzed to find suitable correlations between manufacturing test results and system-level structural tests. Data mining solutions offer a promising approach for the analysis of large data sets and predict semiconductor performance in the field [Chen13, Xue13].

Another major concern in the automotive domain is safety. While the introduction of structural methods can certainly improve the confidence about the integrity of electronic devices, the test application itself has an impact on safety, as it may interfere with the regular operation of a vehicle.

As long as the system does not perform any functional task, the application of structural tests does not impose a major safety issue. However, for the application of online structural tests, safety must be explicitly considered to make sure the system's performance still meets its specification. With this idea in mind, the design methodology described in section 6.4.1 to apply BIST before system shut-down may be extended for *partial networking* [Schmutzler12], where some electronic control units (ECUs) can be tested when they are not required for any functional task.

Another promising structural test strategy is the interleaving of software-based self-test (SBST) with functional applications. As the execution of a piece of software can be easily be handled at system-level, SBST routines can be seamlessly integrated into the behavior of the system [Eberl12, Reimann14]. However, to exploit all the benefits of a structural diagnostic solution, the diagnostic capabilities of SBST [Chen02, Bernardi08] must still be enhanced to support online execution and to deal with arbitrary failure mechanisms.

8.2. Further Research Opportunities

Although the diagnostic methods presented in this dissertation have been devised to overcome diagnostic challenges in the automotive domain, the resulting hardware architecture and diagnostic algorithms are generic enough to be applied outside this application domain.

The developed diagnostic architecture and its algorithms for compacted test responses can be directly employed in manufacturing test to enhance today's procedures for volume diagnosis [Elm12]. In particular, the consideration of more specialized fault models, timing simulation and layout information can serve to optimize the accuracy and resolution of the presented diagnostic approaches.

Similarly, the diagnostic insight made available by the developed diagnostic solutions has ignited research efforts in the context of robust systems [Ernst04, Das06, Sylvester06, Nicolaidis07, Mitra10, Gupta13], where a simple pass/fail criterion may not suffice to assess the integrity of a produced IC. In this regard, further diagnostic analysis is desirable to optimize yield by distinguishing between permanent, intermittent and transient faults [Cook11b, Cook13].

Bibliography

[Abelein12] U. Abelein, H. Lochner, D. Hahn, and S. Straube. Complexity, Quality and Robustness - the Challenges of Tomorrow's Automotive Electronics. In *Proc. ACM/IEEE Design, Automation and Test in Europe Conference & Exhibition (DATE'12)*, Dresden, Mar. 2012, pages 870 –871.

[Abelein13] U. Abelein. Keynote: Quality and Innovation - The Drivers for Modern Automotive Electronics. In *edaWokshop*, Dresden, May 2013.

[Abelein14] U. Abelein, A. Cook, P. Engelke, M. Glaß, F. Reimann, L. Rodríguez Gómez, T. Russ, J. Teich, D. Ull, and H.-J. Wunderlich. Non-Intrusive Integration of Advanced Diagnosis Features in Automotive E/E-Architectures. In *Proc. ACM/IEEE Design, Automation and Test in Europe Conference & Exhibition (DATE'14)*, Dresden, Mar. 2014.

[Abramovici80] M. Abramovici and M. A. Breuer. Multiple Fault Diagnosis in Combinational Circuits Based on an Effect-Cause Analysis. *IEEE Trans. on Computers (TC)*, 100(6):451–460, 1980.

[Abramovici83] M. Abramovici, P. R. Menon, and D. T. Miller. Critical Path Tracing - An Alternative to Fault Simulation. In *Proc. ACM/EDAC/IEEE Design Automation Conference (DAC'83)*, June 1983, pages 214–220.

[Amyeen11] M. Amyeen, A. Jayalakshmi, S. Venkataraman, S. V. Pathy, and E. C. Tan. Logic BIST Silicon Debug and Volume Diagnosis Methodology. In *Proc. IEEE International Test Conference (ITC'11)*, Anaheim, Sept. 2011, pages 1–10.

[Baranowski12] R. Baranowski, M. A. Kochte, and H.-J. Wunderlich. Modeling, Verification and Pattern Generation for Reconfigurable Scan Networks. In *Proc. IEEE International Test Conference (ITC'12)*, Anaheim, Sept. 2012, pages 1–9.

[Baranowski13a] R. Baranowski, M. A. Kochte, and H.-J. Wunderlich. Scan Pattern Retargeting and Merging with Reduced Access Time. In *Proc. IEEE European Test Symposium (ETS'13)*, Avignon, May 2013, pages 39–45.

[Baranowski13b] R. Baranowski, M. A. Kochte, and H.-J. Wunderlich. Securing Access to Reconfigurable Scan Networks. In *Proc. IEEE Asian Test Symposium (ATS'13)*, Taipei, Nov. 2013, pages 295–300.

[Bardell82] P. H. Bardell and W. H. McAnney. Self-Testing of Multichip Logic Modules. In *Proc. IEEE International Test Conference (ITC'82)*, Philadelphia, Nov. 1982, pages 200–204.

[Bardell87] P. Bardell, W. McAnney, and J. Savir. *Built-In Test for VLSI: Pseudorandom Techniques*. John Wiley and Sons, 1987. ISBN 978-0471624639.

[Barford04] L. Barford, V. Kanevsky, and L. Kamas. Bayesian Fault Diagnosis in Large-Scale Measurement Systems. In *Proc. IEEE Instrumentation and Measurement Technology Conference (IMTC)*, volume 2, 2004, pages 1234–1239.

[Bartenstein01] T. Bartenstein, D. Heaberlin, L. Huisman, and D. Sliwinski. Diagnosing Combinational Logic Designs Using the Single Location At-A-Time (SLAT) Paradigm. In *Proc. IEEE International Test Conference (ITC'01)*, Baltimore, Oct. 2001, pages 287–296.

[Barzilai83] Z. Barzilai, D. Coppersmith, and A. L. Rosenberg. Exhaustive Generation of Bit Patterns with Applications to VLSI Self-Testing. *IEEE Trans. on Computers (TC)*, 100(2):190–194, 1983.

[Bayraktaroglu02] I. Bayraktaroglu and A. Orailoglu. Gate Level Fault Diagnosis in Scan-Based BIST. In *Proc. ACM/IEEE Design, Automation and Test in Europe Conference & Exhibition (DATE'02)*, Paris, Mar. 2002, pages 376–381.

[Bernardi08] P. Bernardi, E. E. S. Sánchez, M. Schillaci, G. Squillero, and M. S. Reorda. An Effective Technique for the Automatic Generation of Diagnosis-Oriented Programs for Processor Cores. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 27(3):570–574, 2008.

[Bernardi11] P. Bernardi and M. Sonza Reorda. A New Architecture to Cross-Fertilize On-line and Manufacturing Testing. In *Proc. IEEE Asian Test Symposium (ATS'11)*, New Delhi, Nov. 2011, pages 142–147.

[Bernardi12] P. Bernardi, L. Ciganda, M. de Carvalho, M. Grosso, J. Lagos-Benites, E. Sánchez, M. S. Reorda, and O. Ballan. On-Line Software-Based Self-Test of the Address Calculation Unit in RISC Processors. In *Proc. IEEE European Test Symposium (ETS'12)*, Annecy, May 2012, pages 1–6.

[Blanton12] R. D. Blanton, W. C. Tam, X. Yu, J. E. Nelson, and O. Poku. Yield Learning Through Physically Aware Diagnosis of IC-Failure Populations. *IEEE Design & Test of Computers (D&T)*, 29(1):36–47, 2012.

[Bushnell00] M. Bushnell and V. D. Agrawal. *Essentials of Electronic Testing for Digital, Memory, and Mixed-Signal VLSI Circuits*. Springer, 2000. ISBN 978-0792379911.

[Chen97] W. Chen, S. K. Gupta, and M. A. Breuer. Analytic Models for Crosstalk Delay and Pulse Analysis Under Non-Ideal Inputs. In *Proc. IEEE International Test Conference (ITC'97)*, Washington, Nov. 1997, pages 809–818.

[Chen02] L. Chen and S. Dey. Software-Based Diagnosis for Processors. In *Proc. ACM/EDAC/IEEE Design Automation Conference (DAC'02)*, New Orleans, June 2002, pages 259–262.

[Chen03] L. Chen, S. Ravi, A. Raghunathan, and S. Dey. A Scalable Software-Based Self-Test Methodology for Programmable Processors. In *Proc. IEEE/EDAC/ACM Design Automation Conference (DAC'03)*, Anaheim, June 2003, pages 548–553.

[Chen13] H. H. Chen, R. Hsu, P. Yang, and J. Shyr. Predicting System-Level

Test and In-Field Customer Failures Using Data Mining. In *Proc. IEEE International Test Conference (ITC'13)*, Anaheim, Sept. 2013, pages 1–10.

[Cheng04] W.-T. Cheng, K.-H. Tsai, Y. Huang, N. Tamarapalli, and J. Rajski. Compactor Independent Direct Diagnosis. In *Proc. IEEE Asian Test Symposium (ATS'14)*, Kenting, Nov. 2004, pages 204–209.

[Cheng07] W.-T. Cheng, M. Sharma, T. Rinderknecht, L. Lai, and C. Hill. Signature Based Diagnosis for Logic BIST. In *IEEE International Test Conference (ITC'07)*, Santa Clara, Oct. 2007, pages 1–9.

[Chin84] C. K. Chin and E. J. McCluskey. *Weighted Pattern Generation for Built-In Self-Test*. Center for Reliable Computing, Computer Systems Laboratory, Depts. of Electrical Engineering and Computer Science, Stanford University, 1984.

[Conroy05] Z. Conroy, G. Richmond, X. Gu, and B. Eklow. A Practical Perspective on Reducing ASIC NTFs. In *Proc. IEEE International Test Conference (ITC'05)*, Austin, Nov. 2005, pages 7–17.

[Conroy12] Z. Conroy, J. Grealish, H. Miles, A. Suto, A. Crouch, and S. Meyers. Board Assisted-BIST: Long and Short Term Solutions for Test-point Erosion - Reaching into the DFx Toolbox. In *Proc. IEEE International Test Conference (ITC'12)*, Anaheim, Sept. 2012, pages 1–10.

[Cook11a] A. Cook, M. Elm, H.-J. Wunderlich, and U. Abelein. Structural In-Field Diagnosis for Random Logic Circuits. In *Proc. IEEE European Test Symposium (ETS'11)*, Trondheim, May 2011, pages 111–116.

[Cook11b] A. Cook, S. Hellebrand, T. Indlekofer, and H.-J. Wunderlich. Diagnostic Test of Robust Circuits. In *Proc. IEEE Asian Test Symposium (ATS'11)*, New Delhi, Nov. 2011, pages 285–290.

[Cook12a] A. Cook, S. Hellebrand, M. E. Imhof, A. Mumtaz, and H.-J. Wunderlich. Built-in Self-Diagnosis Targeting Arbitrary Defects with Partial Pseudo-Exhaustive Test. In *Proc. IEEE Latin American Test Workshop (LATW'12)*, Quito, April 2012, pages 1–4.

[Cook12b] A. Cook, S. Hellebrand, and H.-J. Wunderlich. Built-in Self-

Diagnosis Exploiting Strong Diagnostic Windows in Mixed-Mode Test. In *Proc. IEEE European Test Symposium (ETS'12)*, Annecy, May 2012, pages 146–151.

[Cook12c] A. Cook, D. Ull, M. Elm, H.-J. Wunderlich, H. Randoll, and S. Döhren. Reuse of Structural Volume Test Methods for In-System Testing of Automotive ASICs. In *Proc. IEEE Asian Test Symposium (ATS'12)*, Niigata, Nov. 2012, pages 214–219.

[Cook13] A. Cook, L. Rodríguez Gómez, S. Hellebrand, T. Indlekofer, and H.-J. Wunderlich. Adaptive Test and Diagnosis of Intermittent Faults. In *Latin American Test Workshop (LATW'13)*, Córdoba, April 2013.

[Cook14] A. Cook and H.-J. Wunderlich. Diagnosis of Multiple Faults with Highly Compacted Test Responses. In *Proc. IEEE European Test Symposium (ETS'14)*, Paderborn, May 2014.

[Corno04] F. Corno, E. Sanchez, M. Sonza Reorda, and G. Squillero. Automatic Test Program Generation: A Case Study. *IEEE Design & Test of Computers (D&T)*, 21(2):102–109, 2004.

[Cox88] H. Cox and J. Rajski. A Method of fault Analysis for Test Generation and Fault Diagnosis. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 7(7):813–833, 1988.

[Czysz10] D. Czysz, G. Mrugalski, N. Mukherjee, J. Rajski, and J. Tyszer. On Compaction Utilizing Inter and Intra-Correlation of Unknown States. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 29(1):117–126, 2010.

[Darwiche00] A. Darwiche. Model-Based Diagnosis Under Real-World Constraints. *AI Magazine*, 21(2):57, 2000.

[Das06] S. Das, D. Roberts, S. Lee, S. Pant, D. Blaauw, T. Austin, K. Flautner, and T. Mudge. A Self-Tuning DVS Processor Using Delay-Error Detection and Correction. *IEEE Journal of Solid-State Circuits*, 41(4):792–804, 2006.

[Desineni06] R. Desineni, O. Poku, and R. D. Blanton. A Logic Diagnosis Methodology for Improved Localization and Extraction of Accurate Defect Behavior. In *Proc. IEEE International Test Conference (ITC'06)*, Santa Clara, Oct. 2006, pages 1–10.

[Di Carlo11] S. Di Carlo, P. Prinetto, and A. Savino. Software-Based Self-Test of Set-Associative Cache Memories. *IEEE Trans. on Computers (TC)*, 60(7):1030–1044, 2011.

[Dutta09] A. Dutta, M. Shah, G. Swathi, and R. A. Parekhji. Design Techniques and Tradeoffs in Implementing Non-Destructive Field Test Using Logic BIST Self-Test. In *Proc. IEEE International On-Line Testing Symposium (IOLTS'09)*, Sesimbra-Lisbon, June 2009, pages 237–242.

[Eberl12] M. Eberl, M. Glaß, J. Teich, and U. Abelein. Considering Diagnosis Functionality During Automatic System-Level Design of Automotive Networks. In *Proc. ACM/EDAC/IEEE Design Automation Conference (DAC'12)*, San Francisco, June 2012, pages 205–213.

[Eichelberger77] E. B. Eichelberger and T. W. Williams. A Logic Design Structure for LSI Testability. In *Proc. ACM/EDAC/IEEE Design Automation Conference (DAC'77)*, New Orleans, June 1977, pages 462–468.

[Eklow04] B. Eklow, A. Hosseini, C. Khuong, S. Pullela, T. Vo, and H. Chau. Simulation Based System Level Fault Insertion Using Co-Verification Tools. In *Proc. IEEE International Test Conference (ITC'04)*, Charlotte, Oct. 2004, pages 704–710.

[Eldred59] R. D. Eldred. Test Routines Based on Symbolic Logical Statements. *Journal of the ACM (JACM)*, 6(1):33–37, 1959.

[Elm10] M. Elm and H.-J. Wunderlich. BISD: Scan-Based Built-In Self-Diagnosis. In *Proc. ACM/IEEE Design, Automation and Test in Europe Conference & Exhibition (DATE'10)*, Dresden, Mar. 2010, pages 1243–1248.

[Elm12] M. Elm. *Embedded Hardware Structures for Efficient Volume and In-Field Diagnosis of Random Logic Circuits*. Ph.D. thesis, Universität Stuttgart, Institut für Technische Informatik, Stuttgart, 2012.

[Ernst04] D. Ernst, S. Das, S. Lee, D. Blaauw, T. Austin, T. Mudge, N. S.

Kim, and K. Flautner. Razor: Circuit-Level Correction of Timing Errors for Low-Power Operation. *IEEE Micro*, 24(6):10–20, 2004.

[Fang12] H. Fang, K. Chakrabarty, Z. Wang, and X. Gu. Reproduction and Detection of Board-Level Functional Failure. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 31(4):630–643, April 2012.

[Fenton01] W. G. Fenton, M. McGinnity, and L. P. Maguire. Fault Diagnosis of Electronic Systems Using Intelligent Techniques: A Review. *IEEE Trans. on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 31(3):269–281, 2001.

[Furst10] S. Furst. Challenges In The Design Of Automotive Software. In *Proc. ACM/IEEE Design, Automation and Test in Europe Conference & Exhibition (DATE'10)*, Dresden, Mar. 2010, pages 256–258.

[Galke06] C. Galke, R. Kothe, S. Schultke, K. Winkler, J. Honko, and H. T. Vierhaus. Embedded Scan Test with Diagnostic Features for Self-Testing SoCs. In *Proc. IEEE International On-Line Testing Symposium (IOLTS'06)*, Como, July 2006, pages 181–182.

[Ghosh-Dastidar00] J. Ghosh-Dastidar and N. A. Touba. A Rapid and Scalable Diagnosis Scheme for BIST Environments with a Large Number of Scan Chains. In *Proc. IEEE VLSI Test Symposium (VTS'00)*, Montreal, April 2000, pages 79–85.

[Girard92a] P. Girard, C. Landrault, and S. Pravossoudovitch. A Novel Approach to Delay-Fault Diagnosis. In *Proc. ACM/EDAC/IEEE Design Automation Conference (DAC'92)*, Anaheim, June 1992, pages 357–360.

[Girard92b] P. Girard, C. Landrault, and S. Pravossoudovitch. Delay-Fault Diagnosis by Critical-Path Tracing. *IEEE Design & Test of Computers (D&T)*, 9(4):27–32, 1992.

[Grosso12] M. Grosso, W. P. Holguin, E. Sánchez, M. S. Reorda, A. Tonda, and J. V. Medina. Software-Based Testing for System Peripherals. *Journal of Electronic Testing (JETTA)*, 28(2):189–200, 2012.

[Gupta13] P. Gupta, Y. Agarwal, L. Dolecek, N. Dutt, R. K. Gupta, R. Kumar,

S. Mitra, A. Nicolau, T. S. Rosing, M. B. Srivastava, et al. Underdesigned and Opportunistic Computing in Presence of Hardware Variability. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 32(1):8–23, 2013.

[Gurumurthy06] S. Gurumurthy, S. Vasudevan, and J. A. Abraham. Automatic Generation of Instruction Sequences Targeting Hard-to-Detect Structural Faults in a Processor. In *Proc. IEEE International Test Conference (ITC'06)*, Santa Clara, Oct. 2006, pages 1–9.

[Hakmi07] A.-W. Hakmi, H.-J. Wunderlich, C. G. Zoellin, A. Glowatz, F. Hapke, J. Schloeffel, and L. Souef. Programmable Deterministic Built-In Self-Test. In *Proc. IEEE International Test Conference (ITC'07)*, Santa Clara, Oct. 2007, pages 1–9.

[Hakmi09] A.-W. Hakmi, S. Holst, H.-J. Wunderlich, J. Schloffel, F. Hapke, and A. Glowatz. Restrict Encoding for Mixed-Mode BIST. In *Proc. IEEE VLSI Test Symposium (VTS'09)*, Santa Cruz, May 2009, pages 179–184.

[Hellebrand90] S. Hellebrand, H.-J. Wunderlich, and O. F. Haberl. Generating Pseudo-Exhaustive Vectors for External Testing. In *Proc. IEEE International Test Conference (ITC'90)*, Washington, Sept. 1990, pages 670–679.

[Hellebrand92] S. Hellebrand, S. Tarnick, B. Courtois, and J. Rajski. Generation of Vector Patterns Through Reseeding of Multipe-Polynominal Linear Feedback Shift Registers. In *Proc. IEEE International Test Conference (ITC'92)*, Baltimore, Sept 1992, pages 120–129.

[Hellebrand96] S. Hellebrand, H. Wunderlich, and A. Hertwig. Mixed-mode bist using embedded processors. In *Proc. IEEE International Test Conference (ITC'96),*, Washington, Oct. 1996, pages 195–204.

[Hellebrand98] S. Hellebrand, H.-J. Wunderlich, and A. Hertwig. Mixed-Mode BIST Using Embedded Processors. *Journal of Electronic Testing*, 12(1-2):127–138, 1998.

[Hetherington99] G. Hetherington, T. Fryars, N. Tamarapalli, M. Kassab, A. Hassan, and J. Rajski. Logic BIST for Large Industrial Designs: Real Issues and Case Studies. In *Proc. IEEE International Test Conference (ITC'99)*, Atlantic City, Sept. 1999, pages 358–367.

[Holst09a] S. Holst and H.-J. Wunderlich. A Diagnosis Algorithm for Extreme Space Compaction. In *Proc. ACM/IEEE Design, Automation and Test in Europe Conference & Exhibition (DATE'09)*, Nice, April 2009, pages 1355–1360.

[Holst09b] S. Holst and H.-J. Wunderlich. Adaptive Debug and Diagnosis Without Fault Dictionaries. *Journal of Electronic Testing (JETTA)*, 25(4-5):259–268, 2009.

[Holst12] S. Holst. *Efficient Location–Based Logic Diagnosis of Digital Circuits*. Ph.D. thesis, Universität Stuttgart, Institut für Technische Informatik, Stuttgart, 2012.

[Huang97] S.-Y. Huang, K.-T. Cheng, K.-C. Chen, and D. I. Cheng. Error-Tracer: A Fault Simulation-Based Approach to Design Error Diagnosis. In *Proc. IEEE International Test Conference (ITC'97)*, Washington, Nov. 1997, pages 974–981.

[Huang01] S.-Y. Huang. On Improving the Accuracy of Multiple Defect Diagnosis. In *Proc. IEEE VLSI Test Symposium (VTS'01)*, Marina Del Rey, April 2001, pages 34–39.

[Huang07] B. Huang, M. Rodríguez, M. Li, and C. Smidts. On the Development of Fault Injection Profiles. In *Proc. Annual Reliability and Maintainability Symposium (RAMS'07)*, Orlando, Jan. 2007, pages 226–231.

[Huisman04] L. M. Huisman. Diagnosing Arbitrary Defects in Logic Designs Using Single Location At A Time (SLAT). *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 23(1):91–101, 2004.

[IEC] IEC61500. Functional Safety of Electrical/Electronic/Programmable (E/E/PE) Safety Related Systems – Part 1.

[IEE] IEEE Standard 1500-2005. IEEE Standard Testability Method For Embedded Core-based Integrated Circuits.

[Isermann05] R. Isermann. Model-Based Fault-Detection and Diagnosis–Status and Applications. *Annual Reviews in Control*, 29(1):71–85, 2005.

[ISOa] ISO 14230-3: Road vehicles - Diagnostic systems - Keyword Protocol 2000 - Part 3: Application layer.

[ISOb] ISO15765-3: Roadvehicles – Diagnostics on Controller Area Networks (CAN) - Part 3: Implementation of unified diagnostic services (UDS on CAN).

[ISOc] ISO26262. Road vehicles – Functional safety.

[ITR11] International Technology Roadmap for Semiconductors, 2011.

[Iyengar89] V. S. Iyengar and D. Brand. Synthesis of Pseudo-Random Pattern Testable Designs. In *Proc. IEEE International Test Conference (ITC'89)*, Washington, Aug. 1989, pages 501–508.

[J19] Standard J/1979. SAE International.

[Jayalakshmi12] A. Jayalakshmi and T. E. Cheong. A Methodology for LBIST Logic Diagnosis in High Volume Manufacturing. In *Proc. Asia Symposium on Quality Electronic Design (ASQED'12)*, Penang, July 2012, pages 249–253.

[JTA] IEEE Standard 1149.1-1990. IEEE Standard Test Access Port And Boundary-Scan Architecture.

[Keim06] M. Keim, N. Tamarapalli, H. Tang, M. Sharma, J. Rajski, C. Schuermyer, and B. Benware. A Rapid Yield Learning Flow Based on Production Integrated Layout-Aware Diagnosis. In *Prc. IEEE International Test Conference (ITC'06)*, Santa Clara, Oct. 2006, pages 1–10.

[Kim11] S. Kim, E. Lee, M. Choi, H. Jeong, and S. Seo. Design Optimization Of Vehicle Control Networks. *IEEE Trans. on Vehicular Technology*, 60(7):3002 –3016, Sept. 2011.

[Kochte08] M. A. Kochte, C. G. Zoellin, M. E. Imhof, and H.-J. Wunderlich. Test Set Stripping Limiting the Maximum Number of Specified Bits. In *Proc. IEEE International Symposium on Electronic Design, Test and Applications (DELTA'08)*, Hong Kong, Jan. 2008, pages

581–586.

[Kochte09] M. A. Kochte, S. Holst, M. Elm, and H.-J. Wunderlich. Test Encoding for Extreme Response Compaction. In *Proc. IEEE European Test Symposium (ETS'09)*, Sevilla, May 2009, pages 155–160.

[Könemann91] B. Könemann. LFSR-Coded Test Patterns for Scan Designs. In *Proc. IEE European Test Conference*, Munich, 1991, pages 237–242.

[Könemann01] B. Könemann, C. Barnhart, B. Keller, T. Snethen, O. Farnsworth, and D. Wheater. A SmartBIST Variant with Guaranteed Encoding. In *Proc. IEEE Asian Test Symposium (ATS'01)*, Kyoto, Nov. 2001, pages 325–330.

[Koren98] I. Koren and Z. Koren. Defect Tolerance in VLSI Circuits: Techniques and Yield Analysis. *Proceedings of the IEEE*, 86(9):1819–1838, 1998.

[Kranitis05] N. Kranitis, A. Paschalis, D. Gizopoulos, and G. Xenoulis. Software-Based Self-Testing of Embedded Processors. *IEEE Trans. on Computers (TC)*, 54(4):461–475, 2005.

[Kretzschmar04] C. Kretzschmar, C. Galke, and H. T. Vierhaus. A Hierarchical Self Test Scheme for SoCs. In *Proc. IEEE International On-Line Testing Symposium (IOLTS'04)*, Funchal, July 2004, pages 37–42.

[Krishnamurthy87] B. Krishnamurthy. A Dynamic Programming Approach to the Test Point Insertion Problem. In *Proc. ACM/EDAC/IEEE Design Automation Conference (DAC'87)*, Anaheim, June 1987, pages 695–705.

[Krstic98] A. Krstic and K.-T. Cheng. *Delay Fault Testing for VLSI Circuits*. Springer, 1998. ISBN 978-1461375616.

[Krstic02] A. Krstic, W.-C. Lai, K.-T. Cheng, L. Chen, and S. Dey. Embedded Software-Based Self-Test for Programmable Core-Based Designs. *IEEE Design & Test of Computers (D&T)*, 19(4):18–27, 2002.

[Larsson12] E. Larsson and F. G. Zadegan. Accessing Embedded DfT Instruments with IEEE P1687. In *Proc. IEEE Asian Test Symposium*

(ATS'12), Niigata, Nov. 2012, pages 71–76.

[Lavo02] D. B. Lavo, I. Hartanto, and T. Larrabee. Multiplets, Models, and the Search for Meaning: Improving Per-Test Fault Diagnosis. In *Proc. IEEE International Test Conference (ITC'02)*, Baltimore, Oct. 2002, pages 250–259.

[Lee05] K.-J. Lee, C.-Y. Chu, and Y.-T. Hong. An Embedded Processor Based SOC Test Platform. In *Proc. IEEE International Symposium on Circuits and Systems (ISCAS'05)*, Kobe, May 2005, pages 2983–2986.

[Li08] Y. Li, S. Makar, and S. Mitra. CASP: Concurrent Autonomous Chip Self-Test Using Stored Test Patterns. In *Proc. ACM/IEEE Design, Automation and Test in Europe Conference & Exhibition (DATE'08)*, Munich, Mar. 2008, pages 885–890.

[Lingappan07] L. Lingappan and N. K. Jha. Satisfiability-Based Automatic Test Program Generation and Design for Testability for Microprocessors. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, 15(5):518–530, 2007.

[Liu04] C. Liu and K. Chakrabarty. Compact Dictionaries for Fault Diagnosis in Scan-BIST. *IEEE Trans. on Computers (TC)*, 53(6):775–780, 2004.

[Liu07a] C. Liu. Improve the Quality of Per-Test Fault Diagnosis Using Output Information. *Journal of Electronic Testing (JETTA)*, 23(1):11–24, 2007.

[Liu07b] C. Liu, W. Zou, S. M. Reddy, W.-T. Cheng, M. Sharma, and H. Tang. Interconnect Open Defect Diagnosis with Minimal Physical Information. In *Proc. IEEE International Test Conference (ITC'07)*, Santa Clara, Oct. 2007, pages 1–10.

[Lukasiewycz07] M. Lukasiewycz, M. Glaß, C. Haubelt, and J. Teich. Sat-Decoding in Evolutionary Algorithms for Discrete Constrained Optimization Problems. In *IEEE Congress on Evolutionary Computation (CEC'07)*, IEEE, Singapore, 2007, pages 935–942.

[Manley02] D. Manley and B. Eklow. A Model Based Automated Debug Process. In *Proc. IEEE Board Test Workshop (BTW'02)*, Baltimore, Oct. 2002, pages 1–7.

[Maxwell00] P. Maxwell, I. Hartanto, and L. Bentz. Comparing Functional and Structural Tests. In *Proc. IEEE International Test Conference (ITC'00)*, Atlantic City, Oct. 2000, pages 400–407.

[McAnney87] W. McAnney and J. Savir. There is Information in Faulty Signatures. In *Proc. International Test Conference (ITC'87)*, 1987, pages 630–636.

[McCluskey84] E. J. McCluskey. Verification Testing—A pseudoexhaustive Test Technique. *IEEE Trans. on Computers (TC)*, 100(6):541–546, 1984.

[McPherson06] J. W. McPherson. Reliability Challenges for 45nm and Beyond. In *Proc. ACM/EDAC/IEEE Design Automation Conference (DAC'06)*, New York, June 2006, pages 176–181.

[Mehta06] V. J. Mehta, M. Marek-Sadowska, K.-H. Tsai, and J. Rajski. Timing Defect Diagnosis in Presence of Crosstalk for Nanometer Technology. In *Proc. IEEE International Test Conference (ITC'06)*, Santa Clara, Oct. 2006, pages 1–10.

[Merentitis12] A. Merentitis, N. Kranitis, A. Paschalis, and D. Gizopoulos. Low Energy Online Self-Test of Embedded Processors in Dependable WSN Nodes. *IEEE Trans. on Dependable and Secure Computing (TDSC)*, 9(1):86–100, 2012.

[Millman90] S. D. Millman, E. J. McCluskey, and J. M. Acken. Diagnosing CMOS Bridging Faults with Stuck-At Fault Dictionaries. In *Proc. IEEE International Test Conference (ITC'90)*, Washington, Sept. 1990, pages 860–870.

[Mitra04] S. Mitra and K. S. Kim. X-Compact: an efficient response compaction technique. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 23(3):421–432, 2004.

[Mitra10] S. Mitra. Robust System Design. In *International Conference on VLSI Design (VLSID'10)*, Bangalore, Jan. 2010, pages 434–439.

[Miyase04] K. Miyase and S. Kajihara. XID: Don't Care Identification of Test Patterns for Combinational Circuits. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 23(2):321–326, 2004.

[Mumtaz11] A. Mumtaz, M. E. Imhof, and H. Wunderlich. P-PET: Partial Pseudo-Exhaustive Test for High Defect Coverage. In *Proc. IEEE International Test Conference (ITC'11)*, Anaheim, Sept. 2011, pages 1–8.

[Nicolaidis07] M. Nicolaidis. GRAAL: A New Fault Tolerant Design Paradigm for Mitigating the Flaws of Deep Nanometric Technologies. In *Proc. IEEE International Test Conference (ITC'07)*, Santa Clara, Oct. 2007, pages 1–10.

[O'Farrill05] C. O'Farrill, M. Moakil-Chbany, and B. Eklow. Optimized Reasoning-Based Diagnosis for Non-Random, Board-Level, Production Defects. In *Proc. IEEE International Test Conference (ITC'05)*, Austin, Nov. 2005, pages 7–17.

[Park88] E. S. Park, M. R. Mercer, and T. W. Williams. Statistical Delay Fault Coverage and Defect Level for Delay Faults. In *Proc. IEEE International Test Conference (ITC'88)*, Washington, Sept. 1988, pages 492–499.

[Paschalis05] A. Paschalis and D. Gizopoulos. Effective Software-Based Self-Test Strategies for On-Line Periodic Testing of Embedded Processors. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 24(1):88–99, 2005.

[Pateras97] S. Pateras and P. McHugh. BIST: A Test & Diagnosis Methodology for Complex, High Reliability Electronics Systems. In *Proc. IEEE AUTOTESTCON*, Anaheim, Sept. 1997, pages 398–402.

[Peterson72] W. W. Peterson and E. J. Weldon. *Error-correcting codes*. The MIT Press, 1972. ISBN 978-0262160391.

[Poinstingl] P. Poinstingl, H. Randoll, C. Knaupp, T. Braun, T. Wieja, S. Wirth, S. Döhren, and K. R. Offenlegungsschrift DE 10 2010 002 460 A1 2011.09.01: Verfahren zum Testen eines integrierten

Schaltkreises.

[Prabhu12] M. Prabhu and J. A. Abraham. Functional Test Generation for Hard to Detect Stuck-At Faults Using RTL Model Checking. In *Proc. IEEE European Test Symposium (ETS'12)*, Annecy, May 2012, pages 1–6.

[Psarakis06] M. Psarakis, D. Gizopoulos, M. Hatzimihail, A. Paschalis, A. Raghunathan, and S. Ravi. Systematic Software-Based Self-Test for Pipelined Processors. In *Proc. ACM/EDAC/IEEE Design Automation Conference (DAC'06)*, New York, June 2006, pages 393–398.

[Psarakis10] M. Psarakis, D. Gizopoulos, E. Sanchez, and M. Sonza Reorda. Microprocessor Software-Based Self-Testing. *IEEE Design & Test of Computers (D&T)*, 27(3):4–19, 2010.

[Qian09] J. Qian, X. Wang, Q. Yang, F. Zhuang, J. Jia, X. Li, Y. Zuo, J. Mekkoth, J. Liu, and H.-J. Chao. Logic BIST Architecture for System-Level Test and Diagnosis. In *Proc. IEEE Asian Test Symposium (ATS'09)*, Taichung, Nov. 2009, pages 21–26.

[Rajski99] J. Rajski and J. Tyszer. Diagnosis of Scan Cells in BIST Environment. *IEEE Trans. on Computers (TC)*, 48(7):724–731, 1999.

[Rajski00] J. Rajski, N. Tamarapalli, and J. Tyszer. Automated Synthesis of Phase Shifters for Built-In Self-Test Applications. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 19(10):1175–1188, 2000.

[Rajski04] J. Rajski, J. Tyszer, M. Kassab, and N. Mukherjee. Embedded Deterministic Test. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 23(5):776–792, 2004.

[Reimann14] F. Reimann, M. Glaß, J. Teich, A. Cook, L. Rodríguez Gómez, D. Ull, H.-J. Wunderlich, U. Abelein, and P. Engelke. Advanced Diagnosis: SBST and BIST Integration in Automotive E/E Architectures. In *Proc. IEEE/ACM Design Automation Conference (DAC'14)*, San Francisco, June 2014.

[Rich91] E. Rich and K. Knight. *Artificial Intelligence*. Artificial Intelligence

Series. McGraw-Hill, 1991. ISBN 9780070522633.

[Richman85] J. Richman and K. Bowden. Modern Fault Dictionary. In *Proc. IEEE International Test Conference (ITC'85)*, Philadelphia, Nov. 1985, pages 696–702.

[Roberts87] N. H. Roberts and W. E. Vesely. *Fault Tree Handbook*. Nuclear Regulatory Commission, 1987. ISBN 978-0160055829.

[Rousset07a] A. Rousset, A. Bosio, P. Girard, C. Landrault, S. Pravossoudovitch, and A. Virazel. DERRIC: A Tool for Unified Logic Diagnosis. In *Proc. IEEE European Test Symposium (ETS'07)*, Freiburg, May 2007, pages 13–20.

[Rousset07b] A. Rousset, A. Bosio, P. Girard, C. Landrault, S. Pravossoudovitch, and A. Virazel. Fast Bridging Fault Diagnosis using Logic Information. In *Proc. IEEE Asian Test Symposium (ATS'07)*, Beijing, Oct. 2007, pages 33–38.

[Savir98] J. Savir. Salvaging Test Windows in BIST Diagnostics. *IEEE Trans. on Computers (TC)*, 47(4):486–491, 1998.

[Schmutzler12] C. Schmutzler, M. Simons, and J. Becker. On Demand Dependent Deactivation of Automotive ECUs. In *Proc. ACM/IEEE Design, Automation and Test in Europe Conference & Exhibition (DATE'12)*, Dresden, 2012, pages 69–74.

[Segura05] J. Segura and C. F. Hawkins. *"CMOS Electronics: How It Works, How It Fails"*. John Wiley & Sons, Inc., 2005. ISBN 978-0471476696.

[Song99] P. Song, F. Motika, D. Knebel, R. Rizzolo, M. Kusko, J. Lee, and M. McManus. Diagnostic Techniques for the IBM S/390 600 MHz G5 Microprocessor. In *Proc. IEEE International Test Conference (ITC'99)*, Atlantic City, Sept. 1999, pages 1073–1082.

[Stapper80] C. H. Stapper, A. N. McLaren, and M. Dreckmann. Yield Model for Productivity Optimization of VLSI Memory Chips with Redundancy and Partially Good Product. *IBM Journal of Research and Development*, 24(3):398–409, 1980.

[Stollon11] N. Stollon. IEEE P1687–IJTAG. In *On-Chip Instrumentation*. Springer, 2011.

[Struss03] P. Struss and C. Price. Model-Based Systems in the Automotive Industry. *AI Magazine*, 24(4):17, 2003.

[Sun13] Z. Sun, L. Jiang, Q. Xu, Z. Zhang, Z. Wang, and X. Gu. AgentDiag: An Agent-Assisted Diagnostic Framework for Board-Level Functional Failures. In *Proc. IEEE International Test Conference (ITC'13)*, Anaheim, Sept. 2013, pages 1–8.

[Sylvester06] D. Sylvester, D. Blaauw, and E. Karl. ElastIC: An Adaptive Self-Healing Architecture for Unpredictable Silicon. *IEEE Design & Test of Computers*, 23(6):484–490, 2006.

[Tang06] Y. Tang, H.-J. Wunderlich, P. Engelke, I. Polian, B. Becker, J. Schloffel, F. Hapke, and M. Wittke. X-Masking During Logic BIST and its Impact on Defect Coverage. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, 14(2):193–202, 2006.

[Tang10] X. Tang, W.-T. Cheng, R. Guo, and S. M. Reddy. Diagnosis of Multiple Physical Defects Using Logic Fault Models. In *Proc. IEEE Asian Test Symposium (ATS'10)*, Shanghai, Dec. 2010, pages 94–99.

[Touba96] N. A. Touba and E. J. McCluskey. Test Point Insertion Based on Path Tracing. In *Proc. IEEE VLSI Test Symposium (VTS'96)*, Princeton, May 1996, pages 2–8.

[Tuna07] M. Tuna, M. Benabdenbi, and A. Greiner. At-Speed Testing of Core-Based System-on-Chip Using an Embedded Micro-Tester. In *Proc. IEEE VLSI Test Symposium (VTS)*, Berkeley, May 2007, pages 447–454.

[Ull11] D. Ull. *Strukturelle Feldtests bei komplexen ASICs*. Master's thesis, Universität Stuttgart, Institut für Technische Informatik, Stuttgart, 2011.

[Venkataraman00] S. Venkataraman and S. B. Drummonds. POIROT: A Logic Fault Diagnosis Tool and its Applications. In *Proc. IEEE International Test Conference (ITC'00)*, Atlantic City, Oct. 2000, pages 253–262.

[Vo06] T. Vo, Z. Wang, T. Eaton, P. Ghosh, H. Li, Y. Lee, W. Wang, H. Jun, R. Fang, D. Singletary, and X. Gu. Design for Board and System Level Structural Test and Diagnosis. In *Proc. IEEE International Test Conference (ITC'06)*, Santa Clara, Oct. 2006, pages 1–10.

[Volkerink03] E. H. Volkerink and S. Mitra. Efficient Seed Utilization for Reseeding Based Compression. In *Proc. IEEE VLSI Test Symposium (VTS'03)*, Napa Valley, May 2003, pages 232–237.

[Waicukauski89] J. A. Waicukauski and E. Lindbloom. Failure Diagnosis of Structured VLSI. *IEEE Design & Test of Computers (D&T)*, 6(4):49–60, 1989.

[Wang86] L.-T. Wang and E. J. McCluskey. Condensed Linear Feedback Shift Register (LFSR) Testing—A Pseudoexhaustive Test Technique. *IEEE Trans. on Computers (TC)*, 100(4):367–370, 1986.

[Wang06a] L. Wang, S. K. Gupta, and M. A. Breuer. Diagnosis of Delay Faults due to Resistive Bridges, Delay Variations and Defects. In *Proc. IEEE Asian Test Symposium (ATS'06)*, Calcutta, Dec. 2006, pages 215–224.

[Wang06b] Z. Wang, M. Marek-Sadowska, K.-H. Tsai, and J. Rajski. Analysis and methodology for multiple-fault diagnosis. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 25(3):558–575, 2006.

[Wen06] C.-P. Wen, L.-C. Wang, and K.-T. Cheng. Simulation-Based Functional Test Generation for Embedded Processors. *IEEE Trans. on Computers (TC)*, 55(11):1335–1343, 2006.

[Williams81] T. W. Williams and N. Brown. Defect Level as a Function of Fault Coverage. *IEEE Trans. on Computers (TC)*, 100(12):987–988, 1981.

[Williams88] T. W. Williams, W. Daehn, M. Gruetzner, and C. W. Starke. Bounds and Analysis of Aliasing Errors in Linear Feedback Shift Registers. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 7(1):75–83, 1988.

[Wohl02] P. Wohl, J. A. Waicukauski, S. Patel, and G. Maston. Effective

Diagnostics Through Interval Unloads in a BIST Environment. In *Proc. IEEE/ACM Design Automation Conference (DAC'02)*, New Orleans, June 2002, pages 249–254.

[Wu99] Y. Wu and S. M. Adham. Scan-Based BIST Fault Diagnosis. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 18(2):203–211, 1999.

[Wunderlich87] H.-J. Wunderlich. Self Test Using Unequiprobable Random Patterns. In *Proc. International Symposium on Fault-Tolerant Computing (FTCS'87)*, Pittsburgh, July 1987, pages 258–263.

[Wunderlich98] H.-J. Wunderlich. BIST for systems-on-a-chip. *INTEGRATION, the VLSI Journal*, 26(1-2):55–78, 1998.

[Wunderlich10] H.-J. Wunderlich, editor. *Models in Hardware Testing: Lecture Notes of the Forum in Honor of Christian Landrault*, volume 43. Springer-Verlag Heidelberg, 2010. ISBN 978-9400730939.

[Xue13] Y. Xue, O. Poku, and R. D. Blanton. PADRE: Physically-Aware Diagnostic Resolution Enhancement. In *Proc. IEEE International Test Conference (ITC'13)*, Anaheim, Sept. 2013, pages 1–10.

[Yang06] K. Yang and K.-T. Cheng. Timing-Reasoning-Based Delay Fault Diagnosis. In *Proc. ACM/IEEE Design, Automation and Test in Europe Conference & Exhibition (DATE'06)*, Munich, Mar. 2006, pages 418–423.

[Yang07] G. Yang. *Life Cycle Reliability Engineering*. Wiley, Hoboken, N.J., 2007. ISBN 978-0-471-71529-0.

[Ye10] J. Ye, Y. Hu, and X. Li. Diagnosis of Multiple Arbitrary Faults with Mask and Reinforcement Effect. In *Proc. ACM/IEEE Design, Automation and Test in Europe Conference & Exhibition (DATE'10)*, Dresden, Mar. 2010, pages 885–890.

[Ye11] J. Ye, Y. Hu, and X. Li. On Diagnosis of Multiple Faults Using Compacted Responses. In *Proc. ACM/IEEE Design, Automation and Test in Europe Conference & Exhibition (DATE'11)*, Grenoble, Mar. 2011, pages 1–6.

[Ye12] F. Ye, Z. Zhang, K. Chakrabarty, and X. Gu. Adaptive Board-Level Functional Fault Diagnosis Using Decision Trees. In *Proc. IEEE Asian Test Symposium (ATS'12)*, Niigata, Nov. 2012, pages 202–207.

[Ye13] F. Ye, Z. Zhang, K. Chakrabarty, and X. Gu. Board-Level Functional Fault Diagnosis Using Artificial Neural Networks, Support-Vector Machines, and Weighted-Majority Voting. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 32(5):723–736, 2013.

[Yu08] X. Yu and R. D. Blanton. An Effective and Flexible Multiple Defect Diagnosis Methodology using Error Propagation Analysis. In *Proc. IEEE International Test Conference (ITC'08)*, Santa Clara, Oct. 2008, pages 1–9.

[Yu10] X. Yu and R. D. Blanton. Diagnosis of Integrated Circuits with Multiple Defects of Arbitrary Characteristics. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 29(6):977–987, 2010.

[Yuan09] F. Yuan and Q. Xu. On Systematic Illegal State Identification for Pseudo-Functional Testing. In *Proc. ACM/EDAC/IEEE Design Automation Conference (DAC'09)*, San Francisco, July 2009, pages 702–707.

[Zadegan11] F. G. Zadegan, U. Ingelsson, G. Carlsson, and E. Larsson. Design Automation for IEEE P1687. In *Proc. ACM/IEEE Design, Automation and Test in Europe Conference & Exhibition (DATE'11)*, Grenoble, Mar. 2011, pages 1–6.

[Zhang10] Z. Zhang, Z. Wang, X. Gu, and K. Chakrabarty. Board-level fault diagnosis using Bayesian inference. In *Proc. IEEE VLSI Test Symposium (VTS'10)*, Santa Cruz, 2010, pages 244–249.

[Zhang11] Z. Zhang, K. Chakrabarty, Z. Wang, Z. Wang, and X. Gu. Smart Diagnosis: Efficient Board-Level Diagnosis and Repair using Artificial Neural Networks. In *Proc. IEEE International Test Conference (ITC'11)*, Anaheim, Sept. 2011, pages 1–9.

[Zhang12] Z. Zhang, X. Gu, Y. Xie, Z. Wang, Z. Wang, and K. Chakrabarty. Diagnostic System Based on Support-Vector Machines for Board-Level Functional Diagnosis. In *Proc. IEEE European Test Symposium (ETS'12)*, Annecy, May 2012, pages 1–6.

[Zhou06] J. Zhou and H.-J. Wunderlich. Software-Based Self-Test of Processors under Power Constraints. In *Proc. ACM/IEEE Design, Automation and Test in Europe Conference & Exhibition (DATE'06)*, Munich, Mar. 2006, pages 430–435.

[Zorian98] Y. Zorian, E. J. Marinissen, and S. Dey. Testing Embedded-Core Based System Chips. In *Proc. IEEE International Test Conference (ITC'98)*, Washington, Oct. 1998, pages 130–143.

[Zou06a] W. Zou, W.-T. Cheng, and S. M. Reddy. Interconnect Open Defect Diagnosis with Physical Information. In *Proc. IEEE Asian Test Symposium (ATS'06)*, Calcutta, Dec. 2006, pages 203–209.

[Zou06b] W. Zou, W.-T. Cheng, S. M. Reddy, and H. Tang. On Methods to Improve Location Based Logic Diagnosis. In *Proc. International Conference on VLSI Design (VLSID'06)*, Hyderabad, Jan. 2006, pages 181–187.

A. Additional Result Tables

A.1. Diagnosis of Single Faults

Table A.1 shows the average diagnostic accuracy achieved with the approach presented in section 7.1. For these experiments 32 test patterns are compacted into one 32-bit MISR signature. In each experiment a single stuck-at, crosstalk, transition delay, or wired-and fault was injected into the circuit.

Diagnostic accuracy is calculated with the following equation:

$$accuracy = \frac{\#\ correctly\ classified\ faults}{\#\ detected\ fauls}$$

Circuit	Stuck	Crosstalk	Transition Delay	Wired-And
p45k	97.00 %	97.40 %	96.70 %	97.80 %
p100k	100.00 %	92.39 %	94.79 %	94.85 %
p141k	94.95 %	95.40 %	97.83 %	94.57 %
p239k	98.99 %	97.73 %	97.92 %	98.97 %
p259k	99.00 %	97.59 %	97.89 %	97.80 %
p267k	98.00 %	98.68 %	96.70 %	98.89 %
p269k	97.98 %	95.06 %	96.70 %	96.74 %
p279k	98.96 %	96.43 %	97.80 %	97.83 %
p286k	98.96 %	95.00 %	94.44 %	95.65 %
p295k	90.72 %	83.78 %	86.30 %	90.80 %
p330k	96.98 %	94,94 %	97,75 %	96.63 %

Table A.1.: Detailed diagnostic accuracy for single faults: Approach of section 7.1.

Table A.2 shows the average diagnostic accuracy achieved with a state-of-the-art diagnostic algorithm [Holst09b]. For these experiments test results were not compacted. As these tables show, the diagnostic accuracy achieved with both methods is comparable. As pointed out in [Cheng07] the diagnostic accuracy without test response compaction is slightly higher than when compaction is employed.

Circuit	Stuck	Crosstalk	Delay	Wired-And
p45k	98.00 %	92.21 %	97.80 %	97.80 %
p100k	100.00 %	97.83 %	96.88 %	98.97 %
p141k	100.00 %	96.55 %	98.91 %	97.83 %
p239k	97.98 %	97.73 %	98.96 %	98.97 %
p259k	100.00 %	97.59 %	96.84 %	96.70 %
p267k	100.00 %	98.68 %	96.70 %	100.00 %
p269k	98.99 %	96.30 %	97.80 %	100.00 %
p279k	98.96 %	97.62 %	97.80 %	98.91 %
p286k	100.00 %	93.75 %	98.89 %	96.74 %
p295k	96.91 %	95.95 %	98.63 %	96.55 %
p330k	98.98 %	93.67 %	98.88 %	98.88 %

Table A.2.: Detailed diagnostic accuracy for single faults: Approach of [Holst09b]

Table A.3 shows the number of undetected faults in the experiments of tables A.1 and A.2. It can be seen that the stuck-at fault model is detected the most, while the crosstalk fault model features the most undetected faults. However, in both cases diagnostic accuracy is well over 90 %.

Circuit	Stuck	Cross	Delay	Wired-And
p45k	0	23	9	9
p100k	1	8	4	3
p141k	1	13	8	8
p239k	1	12	4	3
p259k	0	17	5	9
p267k	0	24	9	10
p269k	1	19	9	8
p279k	4	16	9	8
p286k	4	20	10	8
p295k	3	26	27	13
p330k	2	21	12	11

Table A.3.: Number of undetected faults in the experiments of tables A.1 and A.2.

A.2. Diagnosis of Multiple Faults

Table A.4 shows the percentage of patterns that can be explained by the consideration of single faults (denoted by “SLAT” in the table) and by the procedure of section 5.5.4

based on systems of linear equations (SLE). Both the *uniform* and *topological* setups are presented in the table. As the table shows, any SLAT-based diagnostic procedure can only account for roughly 60 % of the observed faulty responses. Conversely, the proposed approach is able to analyze almost all faulty patterns.

Circuit	*uniform* setup		*topological* setup	
	% SLAT	% SLE	% SLAT	% SLE
p45k	53.13 %	99.99 %	64.35 %	98.84 %
p100k	66.16 %	100.00 %	56.61 %	98.27 %
p141k	58.11 %	99.99 %	60.10 %	98.59 %
p239k	51.37 %	99.91 %	59.24 %	98.04 %
p259k	61.44 %	99.99 %	49.18 %	99.61 %
p267k	59.41 %	99.98 %	49.07 %	99.33 %
p269k	59.90 %	99.99 %	60.88 %	99.47 %
p279k	64.34 %	99.89 %	71.22 %	98.83 %
p286k	67.83 %	99.97 %	56.90 %	97.77 %
p295k	63.27 %	98.01 %	68.86 %	98.92 %
p330k	66.83 %	99.94 %	62.98 %	96.81 %

Table A.4.: Percentage of explained faulty signatures when the circuit is affected by multiple faults.

Table A.5 shows the coarse-grained diagnostic accuracy for different activation probabilities. In these experiments 5 faults were activated simultaneously with probability p in each test pattern. The table compares the diagnostic accuracy achieved using SLAT-based diagnosis and with the developed technique based on SLEs (Algorithm 1). As the table shows the presented diagnostic method yields excellent results for the complete range of activation probabilities, both for the *uniform* and *topological* setup. The improvement over SLAT is greater for lower activation probability.

Circuit	p	*uniform* setup		*topological* setup	
		% SLE	% SLAT	% SLE	% SLAT
p45k	0.05	90.91	57.73	94.37	76.06
p45k	0.08	98.44	81.25	89.74	73.08
p45k	0.1	94.44	64.81	91.38	84.48
p45k	0.3	97.85	86.56	89.13	82.61
p45k	1.0	100.00	91.95	87.00	78.00
p100k	0.05	97.33	72.00	85.07	67.16
p100k	0.08	97.78	84.44	85.32	73.02

p100k	0.1	95.65	76.09	81.25	69.64
p100k	0.3	100.00	86.79	85.92	70.42
p100k	1.0	97.41	87.93	90.32	83.87
p141k	0.05	100.00	81.82	98.51	86.57
p141k	0.08	100.00	91.43	96.30	92.59
p141k	0.1	100.00	91.30	100.00	90.59
p141k	0.3	100.00	93.62	92.00	87.00
p141k	1.0	100.00	92.31	94.29	82.86
p239k	0.05	96.59	81.82	89.29	67.86
p239k	0.08	97.70	80.46	98.84	91.86
p239k	0.1	94.94	82.28	93.75	71.25
p239k	0.3	100.00	94.17	91.21	82.42
p239k	1.0	100.00	91.23	93.10	88.51
p259k	0.05	98.77	90.12	92.75	81.16
p259k	0.08	98.95	90.53	100.00	88.64
p259k	0.1	98.90	95.60	90.83	79.82
p259k	0.3	100.00	95.28	98.08	96.15
p259k	1.0	100.00	98.18	99.11	91.07
p267k	0.05	92.11	73.68	85.71	78.57
p267k	0.08	100.00	87.50	100.00	90.24
p267k	0.1	100.00	83.33	93.62	89.36
p267k	0.3	100.00	85.56	100.00	84.42
p267k	1.0	94.95	87.23	96.43	88.10
p269k	0.05	100.00	91.89	100.00	77.42
p269k	0.08	98.21	89.29	96.05	76.32
p269k	0.1	98.11	86.79	90.91	72.73
p269k	0.3	100.00	86.30	97.18	88.73
p269k	1.0	100.00	90.43	95.96	83.84
p279k	0.05	97.22	75.00	86.11	75.00
p279k	0.08	89.74	79.49	90.20	68.63
p279k	0.1	100.00	87.80	98.39	77.42
p279k	0.3	98.73	78.48	85.71	67.53
p279k	1.0	97.96	81.63	93.59	88.46
p286k	0.05	98.53	82.35	100.00	78.85
p286k	0.08	98.41	92.06	100.00	86.67

p286k	0.1	98.48	90.91	92.86	80.00
p286k	0.3	100.00	86.11	95.24	84.52
p286k	1.0	96.97	93.94	98.77	92.59
p295k	0.05	100.00	80.56	95.92	79.59
p295k	0.08	100.00	66.67	91.18	67.65
p295k	0.1	100.00	96.77	93.88	77.55
p295k	0.3	98.57	87.14	98.15	85.19
p295k	1.0	99.20	87.23	96.70	84.62
p330k	0.05	96.00	80.00	96.00	72.00
p330k	0.08	97.33	84.00	88.37	76.74
p330k	0.1	97.47	78.48	92.19	75.00
p330k	0.3	96.51	91.86	88.89	81.94
p330k	1.0	100.00	94.06	87.34	77.22

Table A.5.: Coarse-grained diagnostic results for multiple faults.

Tables A.6 and A.7 compare the average coarse-grained diagnostic accuracy of SLAT-based diagnosis with that of the developed method using systems of linear equations (SLE). Table A.6 presents the diagnostic accuracy for the *uniform* setup, while table A.7 does the same for the *topological* setup. The tables show that, on average, the accuracy is in both cases 12 % higher, when the presented method is employed.

Circuit	SLAT	SLE	Δ
p45k	79.53 %	97.03 %	+ 17.50 %
p100k	82.67 %	97.52 %	+ 14.85 %
p141k	90.16 %	100.00 %	+ 9.84 %
p239k	85.48 %	97.75 %	+ 12.27 %
p259k	93.04 %	98.79 %	+ 5.75 %
p267k	83.57 %	97.84 %	+ 14.27 %
p269k	87.89 %	99.07 %	+ 11.18 %
p279k	79.87 %	97.32 %	+ 17.45 %
p286k	87.25 %	97.79 %	+ 10.54 %
p295k	86.67 %	98.92 %	+ 12.25 %
p330k	86.35 %	97.77 %	+ 11.42 %

Table A.6.: Comparison of coarse-grained diagnostic results: *Uniform* setup.

Table A.8 shows the fine-grain diagnostic accuracy achieved with Algorithm 3 of section 5.5.4. The accuracy for different activation probabilities p is shown in the table. As

Circuit	SLAT	SLE	Δ
p45k	78.16 %	90.05 %	+ 11.89 %
p100k	73.64 %	85.94 %	+ 12.30 %
p141k	87.73 %	96.06 %	+ 8.33 %
p239k	82.84 %	93.82 %	+ 10.98 %
p259k	86.02 %	95.98 %	+ 9.96 %
p267k	85.28 %	97.32 %	+ 12.04 %
p269k	80.58 %	95.07 %	+ 14.49 %
p279k	76.60 %	91.49 %	+ 14.89 %
p286k	84.93 %	97.26 %	+ 12.33 %
p295k	79.66 %	95.86 %	+ 16.20 %
p330k	76.35 %	89.97 %	+ 13.62 %

Table A.7.: Comparison of coarse-grained diagnostic results: *Topological* setup.

in table A.5, the developed approach improves the diagnostic accuracy over the complete range of activation probabilities. Consequently, this approach is also effective to diagnose faults within the same fanout region.

Circuit	p	*uniform* setup		*topological* setup	
		% SLE	% SLAT	% SLE	% SLAT
p45k	0.05	81.82	50.91	91.55	74.65
p45k	0.08	93.75	79.69	87.18	71.79
p45k	0.1	87.04	61.11	87.93	75.86
p45k	0.3	96.77	84.41	86.17	77.66
p45k	1.0	94.32	87.50	84.75	75.00
p100k	0.05	93.33	67.67	79.10	64.18
p100k	0.08	97.78	82.22	77.78	66.67
p100k	0.1	95.65	73.91	79.02	66.07
p100k	0.3	98.11	83.96	76.39	62.50
p100k	1.0	97.41	87.93	87.23	79.79
p141k	0.05	98.18	81.82	97.01	82.09
p141k	0.08	100.00	91.43	87.04	81.48
p141k	0.1	100.00	91.30	97.65	85.00
p141k	0.3	98.94	91.49	89.00	86.00
p141k	1.0	99.04	90.38	93.33	80.95
p239k	0.05	95.45	80.68	83.93	64.29
p239k	0.08	94.25	79.31	91.86	89.53

p239k	0.1	93.67	79.75	84.38	66.25
p239k	0.3	99.03	93.20	84.78	72.83
p239k	1.0	100.00	91.23	84.09	79.55
p259k	0.05	96.30	85.19	88.41	76.81
p259k	0.08	97.89	89.47	96.88	86.46
p259k	0.1	96.70	93.41	87.16	75.23
p259k	0.3	99.06	94.81	98.08	95.19
p259k	1.0	100.00	98.18	98.21	90.18
p267k	0.05	84.21	71.05	83.33	77.78
p267k	0.08	95.83	83.33	95.24	80.95
p267k	0.1	95.49	81.94	93.62	87.23
p267k	0.3	97.78	82.22	97.73	83.12
p267k	1.0	93.35	86.44	94.38	87.08
p269k	0.05	100.00	86.49	93.55	74.19
p269k	0.08	96.67	81.67	93.51	74.03
p269k	0.1	96.23	84.91	83.64	69.09
p269k	0.3	100.00	83.56	97.18	87.32
p269k	1.0	99.47	89.63	91.09	81.19
p279k	0.05	97.22	72.22	86.11	75.00
p279k	0.08	89.74	79.49	80.88	64.71
p279k	0.1	100.00	87.80	95.16	74.19
p279k	0.3	98.73	78.48	85.71	66.23
p279k	1.0	97.70	79.59	89.10	84.62
p286k	0.05	97.06	80.88	100.00	78.85
p286k	0.08	95.24	88.89	100.00	86.67
p286k	0.1	96.97	88.64	91.43	77.14
p286k	0.3	100.00	86.11	95.24	84.52
p286k	1.0	96.72	92.93	93.83	89.81
p295k	0.05	100.00	75.00	91.84	77.55
p295k	0.08	100.00	66.67	85.29	67.65
p295k	0.1	100.00	90.32	91.84	73.47
p295k	0.3	92.14	84.29	98.15	85.19
p295k	1.0	97.87	86.17	93.55	81.72
p330k	0.05	94.00	78.00	86.67	70.67
p330k	0.08	96.00	78.67	79.31	70.11

p330k	0.1	96.20	74.68	82.81	68.75
p330k	0.3	95.35	89.53	81.94	76.39
p330k	1.0	100.00	92.57	77.08	70.24

Table A.8.: Fine-grained diagnostic results for multiple faults.

Tables A.9 and A.10 compare the average fine-grained diagnostic accuracy of SLAT-based diagnosis with that of the presented the methodology in this dissertation, both *uniform* and *topological* setups, respectively. The tables show almost the same improvement of 12 % over SLAT-based diagnosis shown in tables A.9 and A.10.

Circuit	SLAT	SLE	Δ
p45k	78.37 %	93.99 %	+ 15.62 %
p100k	83.61 %	96.96 %	+ 13.35 %
p141k	87.37 %	97.31 %	+ 9.94 %
p239k	84.24 %	94.75 %	+ 10.51 %
p259k	89.70 %	95.74 %	+ 6.04 %
p267k	78.32 %	92.07 %	+ 13.75 %
p269k	82.47 %	94.52 %	+ 12.05 %
p279k	77.03 %	93.31 %	+ 16.28 %
p286k	87.38 %	97.86 %	+ 10.48 %
p295k	81.79 %	95.98 %	+ 14.19 %
p330k	83.94 %	94.27 %	+ 10.33 %

Table A.9.: Comparison of fine-grained diagnostic results: *Uniform* setup.

Circuit	SLAT	SLE	Δ
p45k	82.98 %	94.81 %	+ 11.83 %
p100k	81.25 %	94.79 %	+ 13.54 %
p141k	89.43 %	97.93 %	+ 8.50 %
p239k	84.52 %	95.27 %	+ 10.75 %
p259k	87.25 %	97.09 %	+ 9.84 %
p267k	82.20 %	94.07 %	+ 11.87 %
p269k	80.05 %	93.80 %	+ 13.75 %
p279k	77.30 %	94.25 %	+ 16.95 %
p286k	84.36 %	96.67 %	+ 12.31 %
p295k	78.81 %	95.70 %	+ 16.89 %
p330k	80.40 %	96.28 %	+ 15.88 %

Table A.10.: Comparison of fine-grained diagnostic results: *Topological* setup.

A.3. Built-In Diagnosis

Table A.11 shows the number of specified bits in the deterministic patterns of three mixed-mode BIST profiles. The first two profiles apply 4096 and 100000 pseudo-random patterns (PRPs), while the last profile makes use of partially pseudo-exhaustive (P-PET) patterns.

Circuit	# of specified bits [Bit]		
	4096 PRPs	100000 PRPs	P-PET
p45k	52068	9064	1011
p100k	60539	18843	1574
p141k	375597	254410	71078
p239k	162221	102537	18238
p259k	221598	121104	26772
p267k	393979	325800	225732
p269k	396025	322150	227091
p279k	397743	279820	149583
p286k	568528	365475	186941
p295k	579146	362839	230737
p330k	986122	866846	699460

Table A.11.: Mixed-mode BIST: Seed memory cost.

Tables A.12, A.13 and A.14 show the diagnostic accuracy of the three mixed-mode BIST profiles with 4096, 100000 PRPs and P-PET patterns, respectively. For these experiments 32 test patterns are compacted into a single MISR signature. A single stuck-at, crosstalk, transition delay, or wired-and fault is injected in each experiment.

Diagnostic accuracy is calculated with the following equation:

$$accuracy = \frac{\#\ correctly\ classified\ faults}{\#\ injected\ fauls}$$

As the tables show, excellent diagnostic accuracy can still be achieved despite the heavy compaction of test responses. When larger inexpensive test sequences are used, weak windows have an influence on the achieved diagnostic accuracy.

As table A.13 shows diagnostic accuracy for 100000 PRPs with is higher than that achieved with 4096 PRPs. This is due to the fact that longer pattern sequences may contain additional patterns which are able to distinguish some fault pairs and lead to

Circuit	Stuck-At	Crosstalk	Transition Delay	Wired-And
p45k	99 %	77 %	93 %	91 %
p100k	99 %	86 %	90 %	96 %
p141k	98 %	81 %	91 %	95 %
p239k	98 %	87 %	92 %	99 %
p259k	99 %	83 %	93 %	93 %
p267k	99 %	74 %	88 %	93 %
p269k	99 %	80 %	89 %	96 %
p279k	95 %	78 %	87 %	95 %
p286k	96 %	79 %	89 %	93 %
p295k	95 %	70 %	69 %	88 %
p330k	98 %	85 %	86 %	91 %

Table A.12.: Diagnostic accuracy for mixed-mode BIST: 4096 pseudo-random patterns (PRPs).

better diagnostic results.

Circuit	Stuck -At	Crosstalk	Transition Delay	Wired -And	Improvement weak window
p45k	99 %	80 %	94 %	91 %	0 %
p100k	98 %	93 %	92 %	96 %	1 %
p141k	98 %	88 %	93 %	95 %	4 %
p239k	98 %	91 %	97 %	99 %	4 %
p259k	100 %	90 %	97 %	97 %	0 %
p267k	99 %	82 %	95 %	93 %	6 %
p269k	98 %	93 %	94 %	96 %	4 %
p279k	95 %	89 %	90 %	95 %	3 %
p286k	94 %	90 %	86 %	97 %	0 %
p295k	94 %	78 %	74 %	88 %	4 %
p330k	96 %	85 %	90 %	91 %	1 %

Table A.13.: Diagnostic accuracy for mixed-mode BIST: 100000 pseudo-random patterns (PRPs).

Circuit	Stuck -At	Crosstalk	Transition Delay	Wired -And	Improvement weak window
p45k	99 %	89 %	96 %	91 %	6 %
p100k	98 %	93 %	95 %	100 %	3 %
p141k	99 %	89 %	91 %	95 %	6 %
p239k	98 %	95 %	93 %	100 %	3 %
p259k	95 %	89 %	95 %	90 %	6 %
p267k	99 %	86 %	93 %	95 %	0 %
p269k	98 %	90 %	92 %	99 %	3 %
p279k	95 %	85 %	90 %	94 %	0 %
p286k	93 %	89 %	87 %	95 %	3 %
p295k	95 %	80 %	80 %	86 %	2 %
p330k	95 %	90 %	89 %	94 %	2 %

Table A.14.: Diagnostic accuracy for mixed-mode BIST: P-PET.

Table A.15 shows the number of undetected faults in the three considered mixed-mode BIST profiles. As the table shows, the number of undetected faults can be greatly reduced when a larger number of inexpensive patterns are applied.

Circuit	4096 PRPs	100000 PRPs		P-PET	
	Strong	Strong	Strong + weak	Strong	Strong + weak
p45k	36	31	27	25	11
p100k	17	10	7	10	3
p141k	26	23	20	15	6
p239k	18	11	9	10	6
p259k	25	14	14	21	10
p267k	34	23	20	20	8
p269k	32	16	16	14	4
p279k	35	21	21	18	10
p286k	31	23	23	22	17
p295k	52	42	42	38	38
p330k	29	26	23	21	13

Table A.15.: Mixed-mode BIST: Undetected faults with 4096 PRPs, 100000 PRPs and P-PET.

Glossary

ASIC application-specific integrated circuit.

ATE automated test equipment.

ATPG automatic test pattern generation.

BISD built-in self-diagnosis.

BIST built-in self-test.

CAN controller area network.

CS chip-select.

CSA conditional stuck-at.

CUT circuit-under-test.

DfT design-for-test.

DUD device-under-diagnosis.

ECU electronic control unit.

FFR fanout-free region.

FIT failures in time.

GF(2) Galois field of order 2.

HCI hot carrier injection.

IC integrated circuit.

IJTAG internal joint test action group.

IO input/output.

IP intellectual property.

ITRS International Technology Roadmap for Semiconductors.

JTAG joint test action group.

KWP 2000 Keyword Protocol 2000.

LBIST logic built-in self-test.

LFSR linear feedback shift register.

MISR multiple-input signature register.

NBTI negative bias temperature instability.

NTF no trouble found.

OEM original equipment manufacturer.

OTC online test controller.

P-PET partial pseudo-exhaustive test.

PCB printed circuit board.

PFA physical failure analysis.

PPI pseudo-primary input.

PPO pseudo-primary output.

PRP pseudo-random test pattern.

RTL register transfer level.

SBST software-based self-test.

SISR serial-input signature register.

SLAT single location at a time.

SLE systems of linear equations.

SOC system-on-chip.

SPI serial peripheral interface.

STUMPS self-testing unit using MISR and parallel sequence generator.

TAM test access mechanism.

TAP test access port.

TCK test clock.

TDI test data input.

TDO test data output.

TMS test mode select.

TPG test pattern generator.

WIR wrapper instruction register.

WSI wrapper serial input.

WSO wrapper serial output.

WSP wrapper serial port.

XP-SISR extreme parity compactor with SISR.

Index

Publications of the Author

1. Felix Reimann, Michael Glaß, Jürgen Teich, Alejandro Cook, Laura Rodríguez Gómez, Dominik Ull, Hans-Joachim Wunderlich, Ulrich Abelein, and Piet Engelke. Advanced Diagnosis: SBST and BIST Integration in Automotive E/E Architectures. In *Proc. IEEE/ACM Design Automation Conference (DAC'14)*, San Francisco, June 2014.

2. Ulrich Abelein, Alejandro Cook, Piet Engelke, Michael Glaß, Felix Reimann, Laura Rodríguez Gómez, Thomas Russ, Jürgen Teich, Dominik Ull and Hans-Joachim Wunderlich. Non-Intrusive Integration of Advanced Diagnosis Features in Automotive E/E-Architectures. In *Proc. ACM/IEEE Design, Automation and Test in Europe Conference & Exhibition (DATE'14)*, Dresden, Mar. 2014.

3. Alejandro Cook and Hans-Joachim Wunderlich. Diagnosis of Multiple Faults with Highly Compacted Test Responses. In *Proc. IEEE European Test Symposium (ETS'14)*, Paderborn, May 2014.

4. Rafał Baranowski, Alejandro Cook, Michael Imhof, Chang Liu and Hans-Joachim Wunderlich. Synthesis of Workload Monitors for On-Line Stress Prediction. In *Proc. IEEE Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS'13)*, New York, Oct. 2013, pages 137-142.

5. Alejandro Cook, Laura Rodriguez Gomez, Sybille Hellebrand, Thomas Indlekofer and Hans-Joachim Wunderlich. Adaptive Test and Diagnosis of Intermittent Faults. In *IEEE Latin American Test Workshop (LATW'13)*, Córdoba, April 2013.

6. Alejandro Cook, Dominik Ull, Melanie Elm, Hans-Joachim Wunderlich, Helmut Randoll and Stefan Döhren. Reuse of Structural Volume Test Methods for In-System Testing of Automotive ASICs. In *Proc. IEEE Asian Test Symposium (ATS'12)*, Niigata, Nov. 2012, pages 214–219.

7. Alejandro Cook, Sybille Hellebrand and Hans-Joachim Wunderlich. Built-In Self-

Diagnosis Exploiting Strong Diagnostic Windows in Mixed-Mode Test. In *Proc. IEEE European Test Symposium (ETS'12)*, Annecy, May 2012, pages 146–151.

8. Alejandro Cook, Sybille Hellebrand, Michael Imhof, Abdullah Mumtaz and Hans-Joachim Wunderlich. Built-in Self-Diagnosis Targeting Arbitrary Defects with Partial Pseudo-Exhaustive Test. In *Proc. IEEE Latin American Test Workshop (LATW'12)*, Quito, April 2012, pages 1–4.

9. Alejandro Cook, Sybille Hellebrand, Thomas Indlekofer and Hans-Joachim Wunderlich. Diagnostic Test of Robust Circuits. In *Proc. IEEE Asian Test Symposium (ATS'11)*, New Delhi, Nov. 2011, pages 285–290.

10. Alejandro Cook, Melanie Elm, Hans-Joachim Wunderlich and Ulrich Abelein. Structural In-Field Diagnosis for Random Logic Circuits. In *Proc. IEEE European Test Symposium (ETS'11)*, Trondheim, May 2011, pages 111–116.

Curriculum Vitae of the Author

Alejandro Cook received his Bachelor's degree in Electronic Engineering from the UNEXPO University in Barquisimeto, Venezuela in 2006. In 2009 he received his Master's degree in Information Technology from the University of Stuttgart. He then joined the Institute of Computer Architecture and Computer Engineering at this university.

Over the past 6 years the author has been as research assistant under the supervision of Prof. Dr. rer. nat. habil. Hans–Joachim Wunderlich. He has worked on the research project DIANA: "End–to–End Diagnostic Capabilities for Automotive Electronics Systems" sponsored by the German ministry of education and research (BMBF) in cooperation with several industrial automotive partners including Audi AG, Infineon Technologies AG and Continental AG. Additionally, he was involved in project "RM-BIST: Reliability Monitoring and Managing Built-In Self-Test", supported by the German Research Foundation (DFG).

The author's research interests include, but are not limited to, built-in self-test, logic diagnosis, software-based self-test, structural system test, test of robust circuits, and machine learning. In the scope of these topics he has supervised several students in seminars and Master's theses.

Declaration

All the work contained within this thesis, except where otherwise acknowledged, was solely the effort of the author. At no stage was any collaboration entered into with any other party.

____

Alejandro Cook