Politecnico di Torino

ScuDo
Scuola di Dottorato ~ Doctoral School
WHAT YOU ARE, TAKES YOU FAR

Doctoral Dissertation
Doctoral Program in Computer Engineering (36^{th}cycle)

New techniques for quality and reliability enhancement in electronic systems

By

Nunzio Mirabella

******

Supervisor(s):
Prof. Matteo Sonza Reorda, Supervisor
Dr. Michelangelo Grosso, Co-Supervisor
Dr. Ioannis Deretzis, Co-Supervisor

Prof. Alberto Bosio , Referee, Ecole Centrale de Lyon - Institute of Nanotechnology
Prof. Celia López Ongil, Referee, University Carlos III de Madrid
Prof. Cecilia Metra, University of Bologna
Prof. Maurizio Rebaudengo, Politecnico di Torino
Prof. Paolo Rech, University of Trento

Politecnico di Torino
2024

Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Nunzio Mirabella
2024

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Graduate School of Politecnico di Torino (ScuDo).

I would like to dedicate this thesis to my loving parents.

Acknowledgements

I would like to express my deepest gratitude to the people who have supported me throughout my PhD journey.

First and foremost, I would like to thank my responsible Ernesto for his unwavering support and guidance. His patience and dedication have been invaluable to me.

I would also like to extend my heartfelt thanks to my boss Francesco Pappalardo (Fosco) for his presence and encouragement.

I am also grateful to my ex responsible Giovanna Franchino for her support during my first year of my PhD path.

I am also very grateful to my co-supervisors Ioannis Deretzis and Antonino La Magna because their suggestions and presence have been really important during my PhD path.

I would like to express my gratitude to Riccardo Cantoro for his support, suggestions, and for attending a conference in my absence.

I would also like to thank Andrea Floridia for teaching me so much and for his unwavering support. Without his advise and chats many things would have been much more difficult.

I would also like to extend my gratitude to my co-supervisor Michelangelo Grosso for his invaluable help with all the research works we have done together, his crucial advice, and corrections throughout my entire PhD journey. His expertise and support have been instrumental in my success, and I am truly grateful for his contributions. Thank you, Michelangelo, for all that you have done for me.

I would like to extend my really heartfelt thanks Salvatore Rinaudo. Thank you for permitting me to start this journey and always being there for me and for your valuable suggestions throughout my entire PhD.

I would like to add my sincere appreciation to my Professor and Supervisor Matteo Sonza Reorda for his commendable professionalism, his invaluable suggestions, and his unwavering support throughout my PhD journey. His expertise and guidance have been instrumental in my academic success, and I am truly grateful for all that he has done for me. Moreover, I would like to thank him for teaching me a lot about the topics presented in this thesis, but also on a personal level. Thank you, Professor Sonza Reorda, for everything.

I could not have made it this far without the support of these amazing individuals. Thank you all from the bottom of my heart.

Preface

The field of microelectronics has been growing at an exponential rate over the past few decades. The development of smaller, faster, and more efficient electronic devices has revolutionized the way we live, work, and communicate. From smartphones and laptops to medical devices and automotive systems, microelectronics has become an integral part of our daily lives.

As an employee for STMicroelectronics, I was involved in an industrial PhD program and, as a PhD student in collaboration with STMicroelectronics and CNR (Consiglio Nazionale delle Ricerche), I have had the opportunity to work within one of the leading companies in the microelectronics industry that produce their devices with the most modern technologies.

STMicroelectronics is a global semiconductor company that designs, develops, and manufactures a wide range of microelectronics products, including sensors, power management solutions, and microcontrollers. The Institute for Microelectronics and Microsystems of the National Research Council (CNR-IMM) is an public research institution that works on microelectronics systems, starting from the materials and processes, up to devices, circuits and complex architectures. Through my work with STMicroelectronics, I have been able to access valuable data from the industry and use it to develop new technologies and improve existing ones. My position has allowed me to work on cutting-edge research projects that have the potential to make a significant impact on the microelectronics industry. One of the main areas of focus for my research has been the development of new processes for microelectronics testing, especially concerning the reliability of digital systems and the testing of these devices.

During the PhD experience and by working in STMicroelectronics, I have been able to gain a deeper understanding of the challenges and opportunities in this field and develop useful solutions to address them.

Abstract

In the field of integrated circuit (IC) testing, defects detection is one of the most important aspects that require a detailed investigation, to ensure the reliability and functionality of the final manufactured product. Defects can occur anytime in any step of the manufacturing process, caused by several factors, such as manufacturing issues, environmental conditions, and aging effects. Fault models aim at representing specific defects inside a circuit. The study of fault models is essential in digital testing since it helps in identifying and detecting defects. However, different fault models have different strengths and weaknesses, and selecting the best fault model for a particular circuit can be challenging. Sometimes, common fault models are not sufficient to ensure a high defect coverage and for this reason it is important to use a mixed approaches or multiple models to cover as much as possible every type of defects that can occur in a digital circuit. Particularly, the main focus of this work has been on how it is possible to increase the defect coverage of a digital circuit adopting different approaches and obtaining a decreased part-per-million (PPM) escape rate.

The work that has been carried out and reported in this thesis is the result of an industrial PhD activity, made in collaboration with the Institute for Microelectronics and Microsystems of the National Research Council (CNR-IMM) and, as an employee of STMicroelectronics, focused on the development of solutions to improve the quality and the reliability of electronic devices.

The thesis structure begins with chapter 1, that presents an introduction about testing basics; in chapter 2, different fault models have been explored for testing low-power Static Random Access Memories (SRAMs). A significant portion of this chapter has been devoted to the study of the reliability of digital systems, especially with respect to the use of fault models in digital testing. Within this context, novel approaches that allow to test different types of defects (e.g., resistive defects) have

been proposed. The detection of such a kind of faults (e.g., defects into memory cells) requires the development of particular test methods. Furthermore, a particular issue that may affect low-power SRAMs inside digital designs have been introduced, more specifically inside the array of memory cells. The effects of specific resistive defects inside the low-power 6T-SRAM (six transistors SRAM) cell with back-bias circuitry has been evaluated, especially when the entire device works and switches through particular modes (from low-power mode to normal mode). Then, specific fault models have been created and the respective tests to detect them. Hence, the evaluation of these particular defects inside the low-power SRAMs has been faced, obtaining, as a result, the introduction of suitable fault models able to identify the defects and which tests are most suitable. Among the variety of fault models that can be used to target the many possible defects in a circuit, stuck-at and delay (transition and path delay) fault models have been used for many years. Only in the last years, Cell-Aware Testing (CAT) has been introduced as a different approach that aims to improve the detection of internal defects within standard cells, previously seen as black-boxes, only. In fact, the adoption of CAT has become an option for an increasing number of semiconductor companies. Typically, CAT is adopted in the context of scan chain tests, and patterns are generated with an Automatic Test Pattern Generation (ATPG) tool. Past studies have extensively shown the capability of CAT to identify the microchips' physical defects that would otherwise remain undetected using solely traditional fault models. However, due to the higher number of patterns generated, an improper CAT-related ATPG flow can lead to a longer test application time. CAT can obviously be used to improve the efficiency and effectiveness of digital testing, which is essential in ensuring the reliability and safety of electronic circuits. The approach can also be used in a mixed way with other solutions to reduce the cost and time required for testing, which is critical in the semiconductor industry.

As known, ranges of defects can be modeled following an electrical analysis of the device or its cells. Having performed this operation on memory devices, I then applied the same mechanism to the cell-aware testing. This was done in order to replicate the same operation with commercial tools within the cells. In fact, in chapter 3, digital circuits (including benchmarks and actual designs) have been used to compare coverage and test size using different fault models in a mixed approach. In fact, in this work different ATPG flows have been compared. For each flow the cell-aware approach has been used, mixed with more common fault

models, to identify which flow is able to guarantee a more useful and efficient yield in terms of fault coverage and number of patterns. To improve and guarantee the validity of the method, different patterns coming from certain fault models have been fault simulated using fault lists coming from different fault models. Basically, different pattern sets coming from certain fault models have been extracted, then fault simulations have been performed for each fault list with all the pattern sets previously generated. As a result, the combination and the mixing of multiple fault models may allow to achieve the best trade-off between pattern count and fault coverage. To evaluate the proposed approach, several experiments were conducted on benchmark circuits and case studies. The results showed that in some cases, the proposed approaches outperform usual methods (using common fault models) in terms of pattern count and fault coverage.

Contents

List of Figures

List of Tables

List of Acronyms

ASIC Application Specific Integrated Circuit.

ATPG Automatic Test Pattern Generation.

BIST Built-in Self-test.

CAT Cell-Aware Testing.

dDRDF dynamic Deceptive Read Disturb Fault.

dDRF dynamic Data Retention Fault.

DfT Design for Testability.

dIRDF dynamic Incorrect Read Disturb Fault.

dIRF dynamic Incorrect Read Fault.

DPMM Defective Parts Per Million.

DRAM Dynamic Random-Access Memory.

dRDF dynamic Read Destructive Fault.

DRF Data Retention Fault.

IC Integrated Circuit.

IRF Incorrect Read Fault.

ME March Element.

NVRAM Non-volatile Random-Access Memory.

PCB Printed Circuit Board.

RDF Read Destructive Fault.

ROM Read-only Memory.

RWM Read-write Memory.

SAF Stuck-At Fault.

SLT System-Level Test.

SNM Static Noise Margin.

SoC System on Chip.

SRAM Static Random Access Memory.

TDF Transition Delay Fault.

VLSI Very-Large-Scale of Integration.

Chapter 1

Introduction

Digital systems are increasingly becoming an integral part of our modern world, being used in a wide range of applications, from consumer electronics to critical infrastructures. As the complexity and functionalities of these systems increases, so does the need for reliable digital circuits. Electronics employed in modern safety-critical systems require severe qualification during the manufacturing process and in the field, to prevent Fault effects from manifesting themselves as critical failures during mission operations [2]. Nowadays, the technology has reached a very high-quality level, guaranteeing increasingly low defect rates in digital circuits, but the further the technology goes toward more and more complicated nodes in the nanometer range, the greater the chance of a defect occurring during the manufacturing process. For this reason, it is important to ensure a proper coverage of any form of failure of the entire system or at most to guarantee the functionality of the device without compromising the time it takes for the test to ensure its function. The main challenge in this context is that traditional fault models are not sufficient anymore to guarantee the required quality levels for chips utilized in mission-critical applications. The research community and industry have been investigating new test approaches such as device-aware test [3], Cell-aware test [4], path-delay test [5][6], and even test methodologies based on the analysis of manufacturing data to move the scope from 0 *PPM* (Parts Per Million) to 0 *PPB* (Parts Per Billion). The research activity that described in this thesis report is the result of an industrial PhD activity, made in collaboration with the Institute for Microelectronics and Microsystems of the National Research Council (CNR-IMM) and, as an employee of the company,

STMicroelectronics, focused on the development of solutions to improve quality and reliability of electronic devices for industry.

In particular, the general area that I faced during the whole 3-years activity can be split in two main topics, which are introduced in the next two chapters:

- to model physical defects and evaluate the effectiveness of different fault models for testing low-power Static random-access memories (SRAMs), especially what concerns particular issues inside the memory cells cluster; in this case, some specific physical defects were analyzed and after evaluating the effect of these defects, faults have been modeled and implemented into the system model.
  After having evaluated all their behaviors, test programs have been provided to detect the faults.

- to analyze Automatic Test Pattern Generation (ATPG) algorithms that use the cell-aware testing (CAT) approach to target together both common and cell-aware fault models and obtain the best trade-off between fault coverage and testing time; in this case, several flows were provided and evaluated to identify the best flow to detect faults belonging to common fault models and the cell-aware fault model. Furthermore, to evaluate and understand which flow may get more advantages in terms of fault coverage, test pattern sets have been fault simulated with fault lists according to common fault models, including CAT and delay fault models; then, they have been compared with each other.

The final purpose of this work is to evaluate different test methodologies in order to guarantee defect-oriented tests [7] in such a way to allow the detection of new types of defects. In other words, to increase the quality of the test and eventually improve of the chips manufacturing process.

1.1 Reliability of digital systems

To introduce the main topics presented in this thesis it is important to start from the roots talking about digital circuits, which are the main subject of this work. As

known, a digital system can be defined as an interconnection of logic gates made of active and passive components that implement logic elements, such as AND, OR, NOT, and memory operations with flip-flops and registers. All these elements process only sets of discrete and finite-valued electrical signals. On the contrary, an analog circuit which is made up of electrical components such as resistors, capacitors, and transistors, processes electrical signals of continuous values either in the form of currents or voltages. In this work, only digital circuits will be treated.

With the introduction of new *Very-Large-Scale Integration* (VLSI) technologies, especially very low nanometer technologies, the semiconductor industry has increased their ability to answer to the performance-capacity demands from consumers. In fact, semiconductor test costs have been growing as well as the complexity of the systems. Hence it is important that the reliability of digital circuits must grow to ensure more and more the correct functioning of the system, and prevent costly and potentially dangerous failures. This includes the ability of the circuits to function correctly in the presence of various types of conditions, depending on the purposes of the system to be tested.

1.1.1 Basics about testing

In general, the flow of engineering activities includes designing, manufacturing and testing. In modern technologies, testing activities are also considered in the design stage to avoid failures of the system. This means that the testing process is integral to both design and manufacturing activities and cannot be seen as a standalone activity. Obviously, the test activities are done as fast as possible and in an cost-effective way, in order to save time and costs. Certainly, the modern electronics industry tests chips before they are mounted on a board, testing the board before system assembly and finally testing the system is a must for the business of

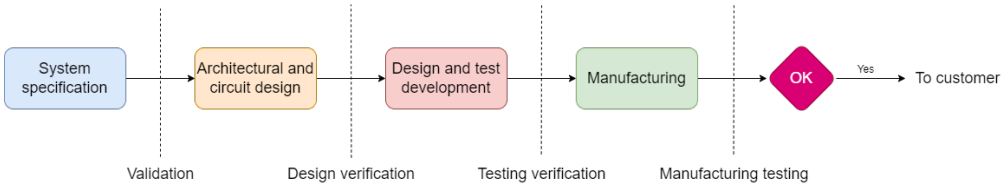

Fig. 1.1 Manufacturing process of a device.

the companies. In fact, as an example, if a chip fault is not caught by chip testing and then a failure analysis is needed due to the test escape, finding the fault may costs about 10 times as much at the printed circuit board (PCB) level than the chip level. Similarly, if a board fault is not caught by PCB testing, finding the fault costs 10 times both at the system level and at the board level. Overall, this means that a fault that is not caught at the chip level may cost 100 times as much at the system level and diagnosis is needed: it aims at identifying what does not work correctly within a product narrowing down the possible locations of the defects.

In fig.1.1 a manufacturing process of a general device is showed. Briefly, after the design specification and validation, the architectural writing of the design is performed as well as the design verification. Commonly, a design verification process is needed to be sure that the device is capable of operating properly and confirms that a design meets its specified requirements and sticking to design guidelines that commonly follow the testing steps. Then, the design and the test environment is set in order to perform the manufacturing after the validation of the test itself: aiming at executing a final verification of design correctness and compliance with specifications defining the exact operating limits for the circuit (in terms of temperature, voltage, etc.). At the end of the flow, after the final test where it is guaranteed the correct behavior of produced devices, the final products are delivered to the customer. Hence, what is testing? Testing is a set of activities designed to ensure that a circuit that has been manufactured complies with the parametric (voltage, resistance, current, capacitance, etc.), timing and functional specifications of the design. From the point of view of the manufactured products, testing verifies that the device is defect-free. Whereas, from the point of view of digital circuits, digital testing is performed on the manufactured integrated circuit (IC) using test patterns that are generated to verify that the product is fault-free. Testing obviously encompasses design verification and diagnosis (fault location for the purpose of making repairs). There are two aspects to test. One is testing the design, or carrying out design verification to make sure the design is correct and conforms to requirements. Design verification also let us know where we are in the development cycle and how stable the design is. The other aspect of the test is the testing for physical failures, making sure nothing is broken and there is no defect stemming from manufacturing. A significant portion of the development cycle time is spent on testing the product design, and that is becoming extremely expensive. It is important to notice that in this work, the design validation

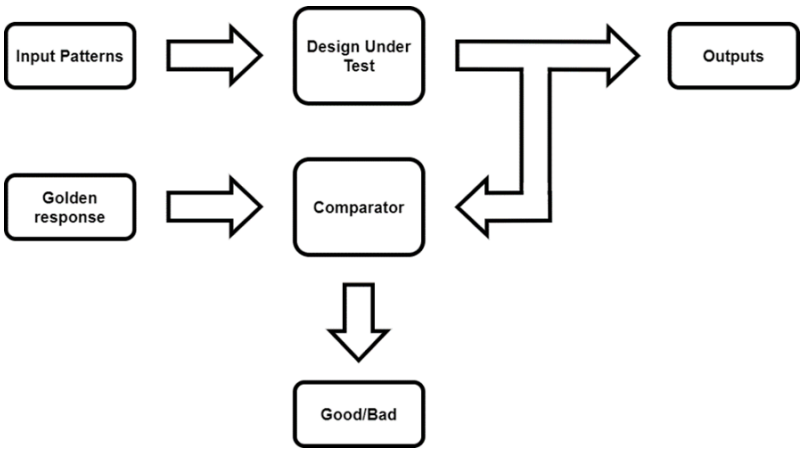

Fig. 1.2 Common test flow.

phase will not be discussed. Testing of digital systems is crucial to ensure the correct functionalities of the devices.

For this reason, *Design-for-Testability* (DfT), an IC design techniques used for increasing and adding testability features to a product design, is used to solve the testing problems and is now widely used. In other words, design for testability is basically how to make each unit in the system testable.

Basically, the device testing flow is depicted in Fig.1.2. As shown, input patterns are generated for the design under test. After the testing phase all the outputs are collected and then imported into a comparator where the golden and observed responses are compared. Through comparison we can discriminate good devices from bad ones. This flow is the base of the testing when any device deviates from its own usual functionality. Certainly, different test flows may be applied at any abstraction level of the circuit, from the system level to the chip level, depending on the scenario. In this way, DfT is crucial to make consistent each part of the design to be tested, allowing the highest defect coverage possible on the device. For this reason, the right test environment is crucial as well, because the appropriately generated stimuli by software can be processed so that the *Automatic Test Equipment* (ATE) can perform the actual test on the chip.

In this thesis not only the testing of designs will be treated but also the testing of memories where a more in depth explanation will be shown in chapter 2.

1.1.2 Fault models

A digital circuit whose functionalities deviate from its intended design is said to be *defective*. If the circuit has an *error* and the output of the circuit is wrong because of the defect it means that a *failure* is observed. When we talk about defects from a higher level of abstraction in terms of circuit function, we refer to them as *faults* [8]. Several types of defects could appear anytime during the manufacturing process or during the operational phase. There could be several reasons why defects may appear during the manufacturing process, in the material or in the package. Furthermore, testing may be very tricky. This is because it results hard to stimulate the system in order to excite the defect and propagate this to the outputs. In order to do this, it is important to find a model that behaves as the specific defect possibly present in the system. For this reason, fault models are introduced that are a representation of the effects of defects on the chip behavior. Fault models aim at supporting the identification of specific defects inside a circuit and they could be used to simulate the misbehavior of the circuit, hence helping the test engineer to develop stimuli able to detect the fault.

More specifically, fault models permit to:

- have a list of elements that could otherwise be uncountable (defects could arise anytime without being in any list)
- have the capability to evaluate the effectiveness of a test (which coverage of the list of faults could be obtained)
- have the possibility to treat the testing problem in a smarter way (using automation, software etc.)

The study of fault models is essential in digital testing since it helps in identifying and detecting faults. However, different fault models have different strengths and weaknesses, and selecting the best fault model (or fault model mix) for a particular circuit and a particular technology can be challenging. There are several fault models, the most common are stuck-at faults, bridging faults, and delay faults. The corresponding defects can occur due to various reasons, such as manufacturing defects, aging, and environmental factors. Sometimes, these models are not sufficient

to ensure a high defect coverage and for this reason it is important to use mixed approach or multiple models to cover as much as possible every type of defect that can occur in a digital circuit. Particularly, the main focus of this work has been on how it is possible to increase the defect coverage of a digital circuit adopting different approaches and obtaining a decreased part-per-million (PPM) defect rate in order to reduce the escape rate of digital devices [9].

1.1.3 Fault simulation and Automatic Test Pattern Generation

To ensure the reliability of digital circuits, various fault models can be used to test these circuits. These fault models are designed to simulate different types of misbehaviour that can occur in digital circuits and to evaluate the ability of the test stimuli to detect these faults. Common fault models used in digital systems include the stuck-at fault model, the bridging fault model, and the delay fault models. Fault detection is the process to determine if a system contains a fault. Fault detection is performed by applying a sequence of test inputs (*stimuli*) and observing the outputs, which are then compared to the expected (fault free, or golden) outputs. Testing can be performed without knowledge about the structure, only by knowing the function of the circuit. Fault simulation identifies all faults that are detected by a given test input. In fault simulation, a list of faults is kept until a fault is detected, then it is marked as detected. It consists of simulating a circuit in the presence of faults. Comparing the fault simulation results with the data given by the fault-free simulation of the same circuit simulated with the same applied test, the faults detected by the test can be determined.

Specific software (Automatic Test Pattern Generator, or ATPG) generates the proper input stimuli able to excite each fault and propagate its effects to some visible location. There could be different types of ATPG tools, such as topological (where they require the knowledge of the structure of the circuit) and functional (where they require the knowledge of the function of the circuit).

The ATPG tool, at the end of the process provides the achieved percentage of fault coverage. The resulting percentage will be compared by the test engineers with the target specified by the technology, pushing them to find further solutions to

possibly further increase it. The patterns used to stimulate the faults are collected and used in the ATE to test the device.

1.2 Thesis contributions

This thesis is organized into four chapters to make the explanation more understandable and introduce each argument in a comprehensive way.

Chapter 1 has already been introduced presenting the fundamentals of the testing and an overview of the topics of this work. In chapter 2 the fundamentals of static random access memories (SRAMs) will be presented as well as the state-of-the-art of SRAMs.

In particular, in chapter 2 different fault models will be explored for testing low-power static random access memories (SRAMs). Moreover, a significant portion of the second chapter will be devoted to the study of the reliability of these digital systems, especially with respect to the use of fault models in digital testing. Then, within this context, novel approaches that allow us to test different types of defects (e.g., resistive defects) will be proposed. The detection of such kinds of faults (e.g., defects in memory cells) requires the development of particular test methods to detect them. Furthermore, a particular issue that may affect low-power SRAMs inside digital designs will be introduced, more specifically inside the array of memory cells. The effects of specific resistive defects inside the low-power 6T-SRAM cell with back-bias circuitry will be evaluated, especially when the entire device works and switches through particular modes (from low-power mode to normal mode). Then, specific fault models will be created, together with the respective tests to detect it. Hence, the evaluation of these particular defects inside the low-power SRAMs has been faced, obtaining, as a result, the introduction of suitable fault models able to identify the defects and which tests are adaptable.

In chapter 3, digital circuits (including benchmarks and actual designs) will be used to evaluate which pattern set has the best fault coverage when using different fault models in a mixed approach. In other words, an investigation of the application of test patterns generated with stuck-at and delay fault models will be compared with others, in terms of fault coverage, pattern count and test generation time. In particular,

it is well known that among the variety of fault models that can be used to target the many possible defects in a circuit, common fault models such as stuck-at and delay fault models are been usually used. In the last years, *Cell-Aware Testing* (CAT) has been introduced as a different approach that aims to improve the detection of internal defects within standard cells, previously seen as black-boxes, only. In fact, the adoption of CAT has become an option for an increasing number of semiconductor companies. CAT is usually adopted in the context of scan chain tests, and patterns are generated with an Automatic Test Pattern Generation (ATPG) tool. Moreover, past studies have extensively shown the capability of CAT to identify the microchips' physical defects that would otherwise remain undetected using solely the traditional fault models. However, due to the higher number of patterns generated, an improper CAT-related ATPG flow can lead to a significantly longer test application time. The CAT approach can be obviously used to improve the efficiency and effectiveness of digital testing, which is essential in ensuring the reliability and safety of electronic circuits. The approach can also be used in a mixed way with other models to reduce the cost and time required for testing, which is critical in the semiconductor industry. In fact, in this work different ATPG flows have been compared. For each flow the cell-aware faults have been used, elaborated with other common fault models, to identify which flow is able to guarantee the most efficient test solution in terms of fault coverage, number of patterns, and ease of integration into the existing test flow. To improve and guarantee the validity of the method, different patterns coming from different fault models have been fault simulated using fault lists belonging to different fault models. Basically, different pattern sets belonging to certain fault models have been extracted, then fault simulations have been performed for each fault list with all the pattern sets previously extracted. If a set of stimuli can detect more faults not only for a specific model but also for other ones, we can reduce the number of patterns that have a direct impact on the test time. It means that, if the defect coverage could be increased and test time decreased, overall, the escape rate of tested devices could be decreased as well. As a result, the combination and the mixing of multiple fault models may allow to achieve the best trade-off between pattern count and fault coverage. To evaluate the proposed approach, several experiments were conducted on benchmark circuits and case studies. The results have showed that in some of them, the proposed approaches outperformed usual methods (using common fault models) in terms of pattern count and fault coverage.

The main contribution in this thesis is basically the improvement of testing techniques for digital circuits belonging to microelectronics companies (e.g., STMicroelectronics) in this work to guarantee a lower escape rate using the most modern techniques and approaches that companies can usually use. In particular, using and resorting to novel and modern fault models such as cell-aware and path-delay fault models, a way has been found to increase the fault coverage of the system and improve the defect rate of the manufactured devices, overall increasing the quality of the company products.

The final part concludes this work drawing some conclusions about what has been done.

Chapter 2

Testing low-power SRAMs

In this chapter of the thesis, an overview about **low-power SRAMs** testing will be introduced as well as an introduction to industrial issues that could arise during the manufacturing phases of these devices, with particular regard to defects into the memory cell. In this work different types of defects will be discussed, in particular what concerns possible *resistive defects* inside the SRAM cell and which kind of effects they produce. Different fault models tailored to mimic the effect of the defect will be discussed since the respective defect causes different effect on the device, with particular emphasis when it goes from a stand-by low-power state to the normal operating state. Hence, different types of resistive defects that may occur inside low-power SRAM cells will be introduced, more precisely focusing on those that impact the device operation. Notwithstanding the continuous evolution of SRAM device integration, manufacturing processes continue to be very sensitive to production faults, giving rise to defects that can be modeled as resistances, especially for devices designed to work in low-power modes.

In the next subsections, among the different resistive defects that may occur in a low-power SRAM, three main types of resistive defects will be introduced. They may impair the device functionalities in subtle ways, depending on the defect characteristics and values that may not be directly or easily detectable by traditional test methods. Regarding the contribution of this work, we analyzed each defect in terms of the possible effects inside the SRAM cell depending on their resistance value and its impact in terms of power consumption, and provided guidelines for selecting the best test methods.

2.1 Introduction

Nowadays, semiconductor memories are widely used to store programs and huge volumes of data. Commonly, memories are widely used for the testing inside many test-vehicle chips. For this reason it is usually needed to know every time the causes of random failures when they occur, that should be one of the crucial requirements for the testing. The increasing demand for low-power technologies used for most modern circuits requires more sophisticated systems than ever in terms of power consumption and reliability. Transistors MOS-channels with smaller sizes in modern node technologies involve circuits that are subject to more leakage currents than ever, especially when several types of defects lead to certain malfunction of the system. For this reason, these systems implement specific methods in order to reduce power consumption in logic and memory systems [10]. Particular methods for detecting defects are implemented as well. For example, in the context of the Internet of Things, devices may be required to stand in idle mode until scheduled events or environmental changes arise. Within these periods of time, it is crucial to keep the leakage current to a minimum, especially concerning SRAM cells, which may be part of large arrays and based on smaller technology nodes. Due to the large area that is usually occupied by SRAMs and their high level of integration, memories are critical from the quality point of view as well. For these reasons, manufacturing tests need to be very accurate, to detect any kind of defects inside the system, and these tests must be as fast as possible in order to contain costs. The situation is worse when the system is affected by defects that become evident under particular conditions only, e.g., when the memory changes its status or operation mode as well as when the power supply is reduced to the minimum allowed by the specification of the circuit. Usually, these types of defects occur inside the memory cells during the manufacturing process. They could be associated with parasitic capacitance or resistance between the routing nets and the contacts of the layout. As an example, faulty via (layers interconnections) [11] [12] may be the cause of misbehaviors inside the cell. Defects in the silicon die such as imperfections on gate oxide that lead to time-dependent dielectric break-down [13] are another actual problem. Depending on the manufacturing process node, circuits subjected to high electrical stress can be prone to imperfections in the system such as other kinds of manufacturing imperfections that could also cause unwanted resistive connections

inside the SRAM cell. Nevertheless, particular attention should be given to power consumption in such low-power SRAMs that need to be made through techniques that allow them to have a lower and lower current consumption. Additionally, defects on low-power structures involve misbehaviors, which can hardly be detected by usual tests algorithms to worsen situation (e.g., March tests [14]). Depending on the type of defect and its value, e.g., resistive defect resistance value, the system undergoes different effects.

The work done for this chapter analyzes the impact of such resistive defects on the behavior of low-power 6T-SRAM (SRAM made up of six transistors) cells and evaluates the effectiveness of different test methods. This study specifically considers the effects of such defects when the back-bias technique [15] is employed to reduce leakage, also evaluating the impact of current consumption on the memory cell. In brief, our theoretical and experimental analysis provides overall evidence that some types of effects caused by resistive defects in the memory cell can produce different types of misbehaviors inside the system under specific conditions and resistance values. To analyze the impact of the different defects and the effectiveness of the different test solutions, an accurate simulation model of a low-power SRAM cell to evaluate and detail the effects of the different resistive defects possibly affecting it has been used. Furthermore, we have evaluated the ability of the different test methods proposed in the literature in detecting these defects. To summarize, this study provides an overview of all these defects, analyzing their effects providing the most accurate fault models to be used for testing and how they can be tested, giving useful guidelines to the test engineer in selecting the best test solution(s). More specifically, this study investigates resistive defects inside the whole low-power SRAM cell as well, hence including defects in symmetric positions occurring inside low-power SRAM cells [16] [17]. At the same time, the study investigates the behaviour of the memory-cell in terms of power consumption and static noise margin (SNM): the effect of each resistive defect injected in the cell is evaluated and then the analysis of the power consumption and SNM is performed for the assessment of non-functional faults inside the system.

This chapter of the thesis is organized as follows: in section 2.2, we introduce some background about memory, testing algorithms, power consumption, and static noise margin analysis; section 2.3 describes the impact of the analyzed resistive

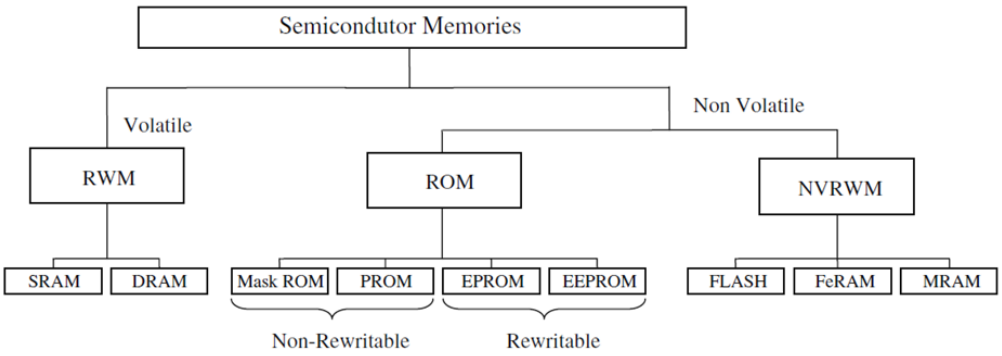

Fig. 2.1 Memories classification (from [1]).

defects on the cell functionalities, specifically referring to a 160 nm **low-power 6T-SRAM**, exploiting the back-bias technique and evaluating the behavior of each defect when its size changes (we considered the full range of resistive values for each considered defect inside the cell). In section 2.4, we introduce the possible tests considered for our analysis; in section 2.5, we report the results obtained from electrical simulations, not only in terms of testing and current consumption, but also in terms of their impact on the SNM and a discussion is presented as well; at last, in section 2.6, we draw some conclusions.

2.2 Background and motivations

2.2.1 Memories

Memories may be involved in several reliability and testing issues. It is known that memories are designed to reach high storage density but at the same time to guarantee high speed of processing when the system requires access. One of the main problems that could occur in a memory is when a manufacturing defect arises in the device. Obviously, most of them may be difficult to detect. For this reason, the main challenge for SRAMs is to find fault models and test solutions that could have a minimal impact on the system and at the same time are suitable for targeting defects that might not be easily detected by common fault models. Anyway, the more the continuous miniaturization of technology nodes goes on, the more traditional test solutions could not be sufficient to spot all the possible defects in the system.

Memories classifications are depicted in the Fig. 2.1. Read-only memories (ROMs) are non-volatile memories. Read-write memories (RWMs) are volatile memories allowing data to be stored and retrieved at very high speed. Among RWM devices, SRAMs and Dynamic random-access memories (DRAMs) are the most popular. An SRAM cell retains its data as long as the power supply of the whole memory is turned on. If the power supply is temporarily or permanently turned off, SRAM loses its data. Compared to an SRAM, a DRAM is smaller instead, since each memory cell is commonly made with one transistor and one capacitance (while the SRAM is made up of transistors only). In this case, it requires a refreshment process to keep the data value. Obviously, this process makes the memory slower and more power consuming with respect to an SRAM. In fact, an SRAM is used in systems where a high access speed is required. On the other hand, a lower cost-per-byte makes a DRAM more attractive whenever a large amount of storage is required. In the Non-volatile Random-Access Memory (NVRAM), data can be read and written as desired, like RWMs, but NVRAMs maintain their content even without power supply, like ROMs. To sum up, SRAMs are low power and very fast, but more expensive than DRAMs and harder to make in large sizes. DRAMs need more power and are slower than SRAMs, but much smaller in terms of size. In this work we consider just the volatile low-power SRAMs.

2.2.2 Memory testing

Memory testing plays an important role in modern technologies. The design of memory often requires using the maximum storage density in the minimum area. So, the more the technology evolves, the more the data storage and complexity increases, thus making the appearance of manufacturing defects inside the system more likely. Previous studies considered the different defects that may affect each cell in an SRAM [1]. In particular for this study, a special attention is given to resistive defects inside the cell that should be detected by properly setting the adopted test techniques. SRAMs are commonly subject to defects or variations during the manufacturing process. The testing of these devices is often subject to multiple challenges, using common fault models and test processes. Previous studies explain and discuss the taxonomy of static fault models and dedicated March test solutions [18]: a sequence

of operations on the whole memory, each performing on each cell the same set of read/write operations.

Nowadays, most test solutions may cover most of the faults in memories, but they are not sufficient to cover certain type of faults that are protagonists for the latest technologies (nanometer) that shrank the SRAM designs. New types of faults (*dynamic faults*) [19] [20] [21] could arise when more than one read or write operation is performed, hence an elaborate combination of operations is needed to detect them. Typically, the faults we are referring to are modeled as electrical malfunctions due to resistive defects (opens and bridges) or parameter mismatches and may occur in the various blocks of an SRAM such as the core-cell array, the address decoders, the pre-charge circuits, the sense amplifiers, and the write drivers. A comprehensive SRAM test must guarantee the correct functionality of each cell of the memory (ability to store and maintain a data) and the corresponding addressing, write, and read operations in the worst conditions with regard to temperature interval, voltage constraints, and timing requirements. It must also ensure pattern and voltage sensitivity immunity, where the voltage sensitivity is a undesired behavior caused by relevant variations in the voltage supply.

2.2.3 Low-power 6T-SRAM structure

The 6T-SRAM cell considered in this chapter is depicted in Figure 2.2. It could be considered addressable by the line or in blocks. It is made up of two inverters (composed of transistors M_1–M_2 and M_3–M_4, respectively) and two pass-transistors, M_5 and M_6, that enable the functionalities of the cell. When writing/reading operations are performed, the WL signal is high, allowing the *BL* and *BLB* signals to be connected through M_5 and M_6 to the S and SB nodes, respectively.

If a "0" writing operation is performed in the cell, the bit lines (*BL* and *BLB*) must be connected to ground (*GND*) and the power supply (VDD), respectively. Then, the WL signal activates the pass-transistors M_5 and M_6 allowing one to move *S* and *SB* node to *GND* and VDD, respectively; moreover, if a "1" writing operation is performed in the cell (see Figure 2.3), *BL* and *BLB* are charged to the VDD and *GND* value, respectively, thus writing a "1" and a "0" in the *S* and *SB* node (they move to $VDD/2$ value), respectively using the positive feedback. If no defects occurred

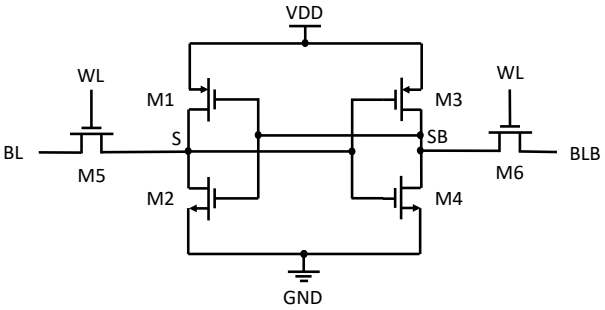

Fig. 2.2 6T-SRAM cell (from [1]).

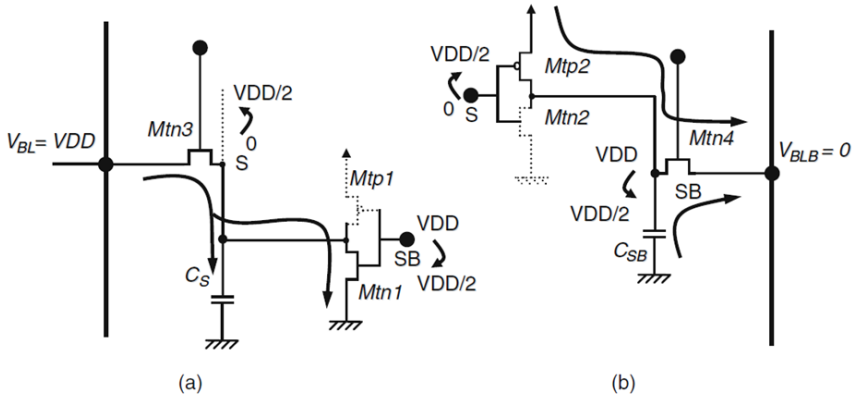

Fig. 2.3 Writing operation of a SRAM cell for *BL* (a) and *BLB* (b) lines (from [1]).

inside the cell, it should stay in the same state as long as a new operation occurs on it.

If a reading operation is performed in the cell (see Figure 2.4), the *BL* and *BLB* lines must firstly be pre-charged at a VDD value, using the respective pre-charge circuitry. Then, the WL signal activates M_5 and M_6, allowing one to connect the S and SB nodes at the *BL* and *BLB* lines, thus creating a voltage difference detected by a sense amplifier. Usually, just a few hundred millivolts differential voltage is sufficient to operate a correct read operation. Certainly, the data of the cell must remain stable and keep its logic state during readout. Typically, the inverters are designed in a way that PMOS and NMOS transistors match with the inverter threshold at $VDD/2$. The access transistors are usually designed two or three times wider than the NMOS transistors of the inverters.

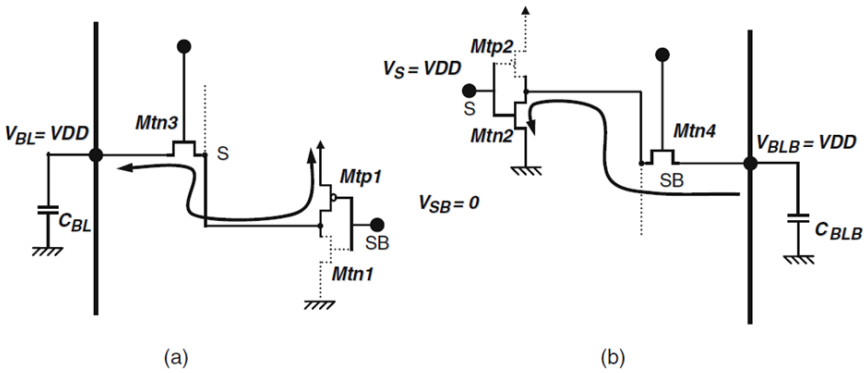

Fig. 2.4 Reading operation of a SRAM cell for *BL* (a) and *BLB* (b) lines (from [1]).

The organization of the storage cells is commonly done in a matrix. Figure 2.5 explains such an organization. The cell matrix has $2M$ rows and $2N$ columns, for a total storage capacity of $2Mx2N$ bits, where M and N are the number of bits used to specify the row address and the column address, respectively. Each cell in the array is connected to one of the $2M$ row lines, universally called word lines, and to one of the $2N$ column lines, commonly called bit lines, or also digit lines. A particular cell can be accessed for a read or write operation by selecting its word line and its bit line.

For the purpose of this chapter, we considered a low-power SRAM memory cell only, using the back-bias technique [22] [23], a widely used solution that allows the system to reduce the leakage current during the idle periods (see figure 2.6). This system reduces the rail-to-rail voltage by increasing the voltage of the (virtual) ground node $VGND$. When the control signal PDM (Power Down Mode) is activated, the cell switches from Normal Mode (NM) to Low-Power Mode (LPM), thus activating the back-bias circuit, during which neither writing nor reading operations can be performed in the cell due to the fact that the threshold of the MOS transistors changes, hence the inverters do not work. The back-bias circuitry is usually inserted in clusters of cells to decrease the current consumption of each cell and thus of the whole arrays. The time that the system requires to switch these operational modes could be measured in milliseconds, that is, the time used to allow the system to switch from normal mode to low-power mode, which is orders of magnitude larger than read/write operations.

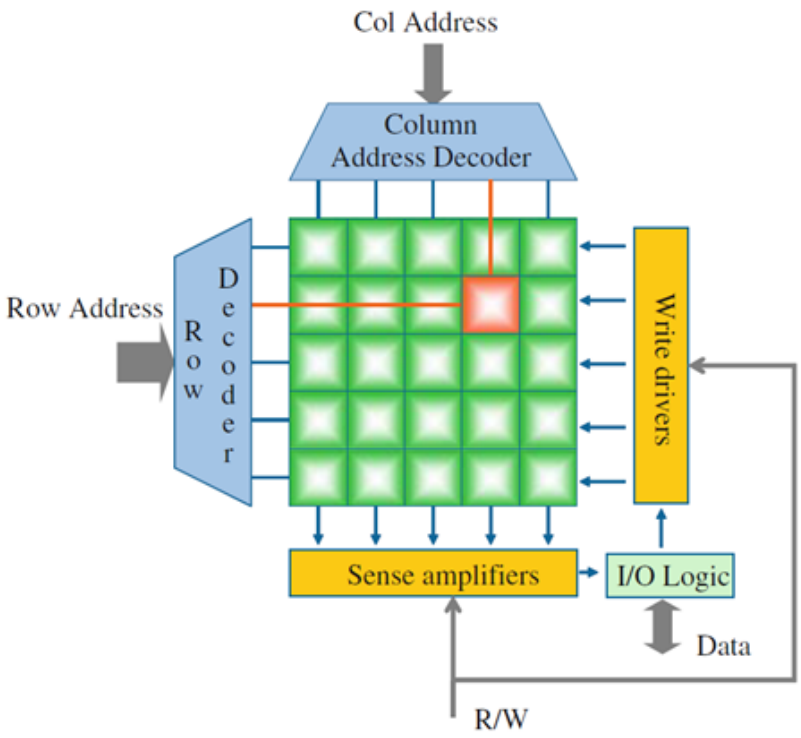

Fig. 2.5 SRAM architecture (from [1]).

2.2.4 Resistive defects and fault models

Depending on to the number of operations (m) that are needed in the sequence of testing, the fault may be classified as Static if $m \leq 1$, or Dynamic if $m > 1$. Hence, faults affecting memories can be classified in the following two categories:

- static faults: when at most one operation (read/write) is required ($m \leq 1$) such as Stuck-At Faults (SAFs, $m = 0$).
  In particular, if the faulty cell is stuck-at zero, whatever is the operation performed on the cell, its content remains at logic '0'. To sensitize this fault, it is needed to set at logic '1' its state. At the end of this operation, the cell will not swap its state from '0' to '1' as expected.
  It may be considered the same type of fault using transition fault model, or rather, when a memory core-cell fails to undergo a transition ($0 \rightarrow 1$) when it is written ($m = 1$). Other static faults are, for example, the various types of Coupling Faults (CFs) [18].

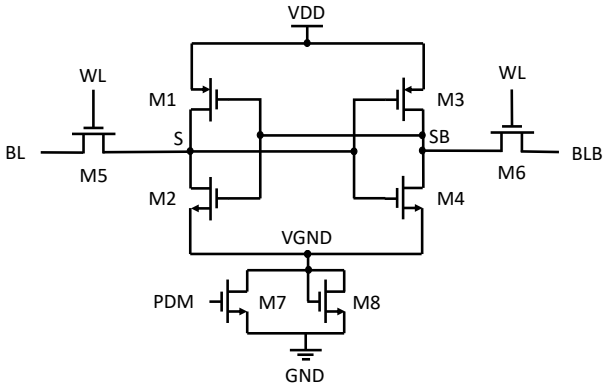

Fig. 2.6 SRAM cell with back-bias circuit.

- dynamic faults: when more than one operation (read/write) in sequence is required ($m > 1$).
  As will be shown further in this chapter, all parts of the memory may be subject to these faulty behaviors.
  For example, the Address Decoder Open Faults (ADOFs) are related to dynamic faults in the address decoder, dynamic Read Destructive Faults (dRDFs) are dynamic faults linked to failures in the core-cell [24], Un-Restored Destructive Write Faults (URDWFs) are dynamic faults due to failures either in the pre-charge circuit or in the write driver [25] [26] [27]. The Dynamic Fault Set (DFS) is infinite as the number of possible operations is not limited.

In the latest SRAM technologies, dynamic faults appear as the predominant root cause of errors requiring new test solutions. The memory cell can be affected by several defects. On a circuit model, some of them can be modeled as resistive-bridging [28] and resistive-opens defects [29][30][31]:

- resistive-bridging defects create an unwanted current path between two nodes in the cell, which are not intended to be connected.
- resistive-open defects increase the resistance of existing paths inside the cell.

Both resistive-bridging and resistive-open defects may force the cell to misbehave when the corresponding resistance holds specific ranges of values. The functional

model of a defect is referred to as a fault. A wide body of literature describes the different types of faults that may occur in a SRAM cell [18]. Among them, the following are the most used:

- stuck-at Fault (SAF), in which the logic value of a cell is always either “0” or “1”.
- transition delay Fault (TDF), when a cell is unable to change its state (0→1 or 1→0) when a write operation is made.
- incorrect Read Fault (IRF), that is when a read operation performed on the core-cell returns an incorrect logic value, and the correct value is still stored in the core-cell.
- dynamic Incorrect Read Fault (dIRF): A core-cell is said to have an IRF if a read operation performed on the core-cell returns an incorrect logic value, and the correct value is still stored in the core-cell.
- data Retention Fault (DRF) [32] when a memory cell loses its previously stored logic value after a certain period of time during which it has not been accessed.
- dynamic Data Retention Fault (dDRF) [24], a DRF that occurs when a memory cell loses its previously stored logic value after at least two read or write operations are performed on other cells.
- Read Destructive Fault (RDF) [33], that occurs when a write operation immediately followed by a read operation performed on the cell changes the logic state of this cell and returns an incorrect value on the output.
- dynamic Read Destructive Fault (dRDF), that occurs when a read operation performed on the cell changes the data in the cell itself and returns an incorrect value on the output.
- dynamic Deceptive Read Disturb Fault (dDRDF) [34] when a write operation immediately followed by a read operation changes the logical value stored in the memory cell, but returns the expected output.

- dynamic Incorrect Read Disturb Fault (dIRDF) when a write operation immediately followed by a read operation does not change the logical value stored in the memory cell, but returns an incorrect output.

2.2.5 Test algorithms

Test algorithms are particular flows used for memory testing. To detect faults in memory, specific sequences of write and read operations known as March tests are commonly used [35]. Often, the application of such tests exploits embedded built-in self-test (BIST) logic to increase test quality (e.g., with higher frequency operation and taking into account array scrambling) and lower costs. A March test is a test algorithm composed of a sequence of March Elements . Each March Element (ME) is a sequence of memory operations applied sequentially on a memory cell before proceeding to the next one. Usually, each value is compared with the respective expected one. The order to move from a certain address to another one is called Address Order (AO). The AO targets the ME direction order (it can be done in either one of two address orders): an increasing ($\Uparrow$) address order or a decreasing ($\Downarrow$) address order which is the opposite of the $\Uparrow$ address order. March tests are nowadays the dominant type of memory tests used in the industry and with several approaches and algorithms. Despite the high fault coverage achieved by March tests with respect to the set of static faults, non-classical faults that are based on a more physical view of the memory are not covered by March tests. Independently of the specific tests approaches, it is important to validate the system through fault simulations to compute the Fault Coverage of a test sequence every time a new defect is discovered, and the corresponding fault model is defined.

In Fig.2.7 the architecture of a typical memory fault simulator is shown. It requires three main inputs: the fault list, the applied March test, and the memory model. The results of the fault simulation are the coverage report and the fault dictionary that contains the set of test symptoms associated with each modeled fault.

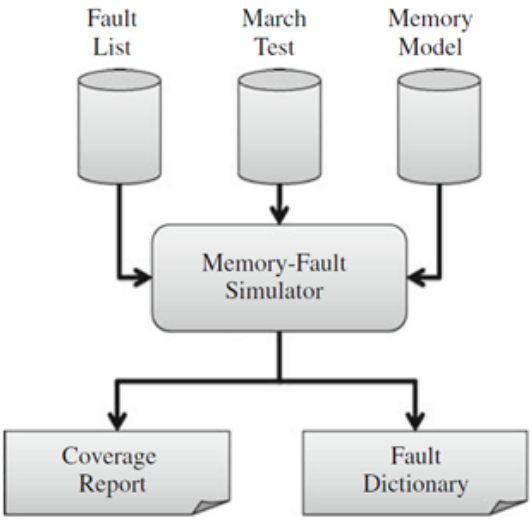

Fig. 2.7 Memory fault simulator chart (from [1]).

2.2.6 Power consumption

Regarding low-power SRAMs, it is crucial to consider the power consumption of the system. We focused on the pattern of the current of the SRAM cell when the device functionality deviates from the standard operation, especially when the system alternates between different operational modes in the presence of defects inside the cell.

In this current chapter, we will also discuss the impact of specific resistive defects in terms of power consumption under particular conditions and analyze the range in which we have an effect of the defect. In order to guarantee a lower power condition and current consumption of the system, the back-bias technique is implemented in the cell. For example, in the case of a defect if it occurs, e.g., when a resistive defect creates a new unwanted path (resistive-bridge), the system undergoes a larger current consumption that may exceed the device specifications. In addition, a value difference with respect to the nominal current can also be used to detect the fault by means of quiescent current tests. An actual situation is when a resistive path close to the power supply (resistive-open) increases the current on that, thus becoming detectable by tests.

2.2.7 Static Noise Margin

The Static noise margin (SNM) [36][37][38][39] measures the stability of the cell and it is defined as the minimum noise voltage present at each of the cell storage nodes (*S* and *SB*) necessary to flip the state of the cell. It could be split into three types: hold noise margin, write noise margin, and read noise margin. For this study, we considered only the hold noise margin, in order to check the reliability of the cell. We analyzed and measured it during each steady-state data simulation. We will consider the impact on SNM not only when the cell does not work because of the defects, but also when the defective cell is working, as well. Indeed, when a defective cell continues working despite the presence of the defect, further analysis could be necessary to better understand if the reliability of the system could be compromised.

In Figure 2.8, we report the typical and symmetrical behavior of the SNM for a fault-free cell. The SNM can be obtained by varying the V1 and V2 voltages that represent the S and SB node voltage values, respectively. The two signals are represented at the same step along the two axes, until the two curves are adjacent. The following flow is implemented to achieve a butterfly-line graph, which is representative of the status of the static noise in the circuit:

- considering INV-$M_{3,4}$ we perform a direct current (DC) analysis on the *S* node, and we look at the output of the inverter.
- then, we draw the butterfly graph representing the *S* node (V_1) axe in SB (V_2) node axis and vice versa for the *SB* node, to achieve the correct symmetric picture.

2.3 Effects of resistive defects in the Low-Power SRAM Cell

This first part of the study focuses on the effect of some resistive defects that may occur in a low-power SRAM cell. These defects may not only influence the functional behavior of the memory, but impact power consumption as well. In this chapter, we will focus not only on functional effects, but we will analyze the impact

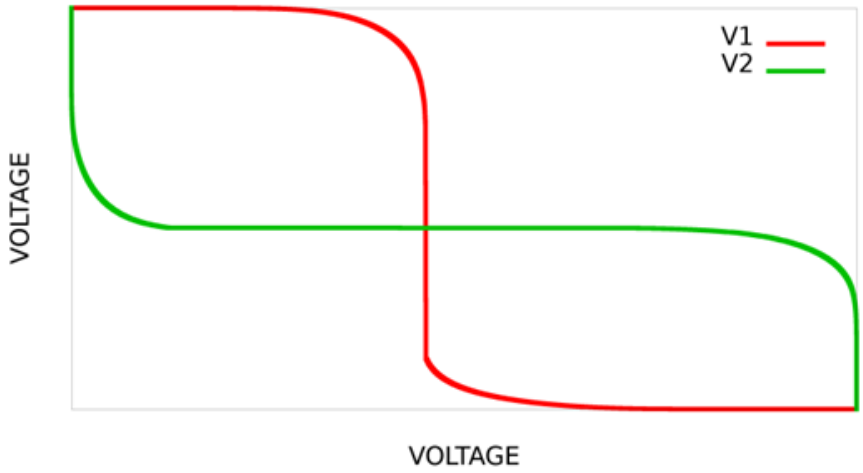

Fig. 2.8 Typical behavior of SNM for a symmetric SRAM cell.

of the main defect we considered inside the cell, evaluating the effect on power and SNM, as well. When one of such defects occurs, the cell may perform differently from the desired behavior, depending on the defect characteristics.

In this section, we analyze the impact of each defect. Previous works discussed about resistive defects [29]. Figure 2.9 shows some of the possible resistive defects that may arise in the cell; first, we are going to analyze three different main resistive defects (R_1, R_2, and R_3) inside a single inverter of the low-power 6T-SRAM cell and then we will discuss the other ones, which can be related to the former ones considering the symmetry of the design. Different colors are used in Figure 2.9 to identify defects playing corresponding roles in the different inverters composing the cell. Other resistive defects could occur in other parts of the SRAM cell whose effect is translated in fault models introduced in the previous sections but they are not presented in this study.

2.3.1 Resistive path on the cell transistor gate (R_1)

The first resistance we consider in this study corresponds to a *resistive-bridge* defect inside the cell. Concerning the R_1 resistance, it creates a resistive path connection between VDD and the gate of M_3. This link could produce, under particular conditions, a failure in the reading and writing operations performed in the cell. Depending on the R_1 value, the typical function of the inverter INV-$M_{3,4}$ may

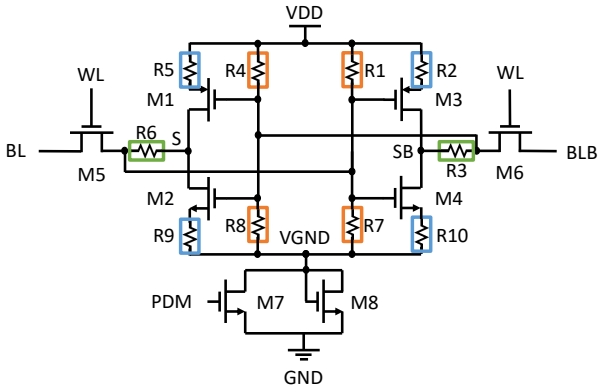

Fig. 2.9 The resistive defects considered in this study.

be compromised when the cell is written with a "0". We identified three different ranges in which we have different behaviors of the cell with this defect. In the first range of resistance values ($0 \div R'_X$), it does not allow the cell to be written since the defect is considered as a short circuit that does not allow any operation. On the other hand, the third range (above R''_X) has no functional effect on the cell, no matter whether we are in the NM or in the LPM. In the middle range (between R'_X and R''_X), the resistive-bridging defect will have no effect on NM, but when the memory switches to LPM, R_1 could cause a change in the data stored in the cell when the cell switches again to NM, thus allowing the reading of the stored data. Indeed, during LPM, the *VGND* node voltage value increases as well as the *S* and *SB* node voltage value changing the threshold of the inverters, so when the cell returns to NM, the stored data flip. We define this type of fault as a Low-Power Retention fault (LPRF), as it behaves as a DRF, but it becomes active only when passing through LPM. To summarize, assuming the cell is written with a "0", the behavior of the system with a resistance of an increasing value is the following:

- when the resistance value is very low (in the $0 \div R'_X$ range) the cell cannot be written and is isolated from the system so it does not work at all, because the S and SB nodes cannot keep their value. In this case, we have a stuck-at fault.
- when the resistance value is in the $R'_X \div R''_X$ range, the cell keeps working on until it switches to LPM and then switches again to NM. In this case, the cell

cannot preserve its state and flips its value, preserving the faulty value in the achieved NM state. In this case, we have an LPRF.

- when the resistance value is above R''_X, the cell works correctly in any case because we have a situation very similar to a fault-free SRAM cell.

This resistive-bridge defect has a certain impact on power consumption. In fact, the presence of a resistive path between the two focused *S* and *SB* nodes leads to quite a large current as an effect of the defect in the gate of M_3 keeping the functionality of the transitor out of the correct usage. In particular, the unwanted path causes an increase in the absorbed current in the cell that maintains its higher value until either the bit-flip (after switching from LPM to NM) or the changing of the data stored involves a decrease in the non-expected value within the specification of the system. Indeed, this current, starting from *VDD* and passing through the gate of M_3 and then through the *S* node and the M_2 channel as far as the *GND* node, modifies the expected working behavior of the entire system. In terms of SNM, considering only the middle range, we observe an impact on the noise margin that is very small with respect to the correct graph depicted in fig.2.8.

2.3.2 Resistive open on the cell transistor source terminals (R_2)

The other resistance we consider in this study corresponds to a resistive-open defect like R_2, inside the cell.

In this case, the behavior of the system is not compromised as long as the resistance value is not high enough to produce a failure, where the resistance may be considered almost an open circuit, thus not allowing the cell to handle read and write operations. Considering all the ranges of values that R_2 may have, causing a fault in the cell, this defect can cause misbehavior in two cases:

- when the cell is written and then we quickly perform a read operation.
- when the cell is written, then we switch the system to LPM and after switching again to NM we perform a read operation.

The effects of the fault can be analyzed by referring to three ranges of values for the size of the resistive defect R_2. Assuming the cell is written with a “0” and

considering R_2 as resistive-open defect injected inside the cell, there are the following cases:

- when the resistance is within the range $0 \div R'_Y$ the system works correctly, even if we perform a read operation either in NM or after a transition between LPM and NM.
- when the resistance is within the range $R'_Y \div R''_Y$ the cell undergoes a failure when a read after write (RAW) operation is performed [24] [40]. This effect can be modeled as a dRDF.
- when the resistance is above R''_Y, a failure is visible if we perform a quick RAW (the cell is read right after a write action is performed), even if we keep the cell either in NM or passing through LPM (the associated model is dRDF).

We observe no effect on current consumption for this defect, but we do see an impact on SNM, if we consider the symmetry of the cell. This effect may compromise the quality of the cell, thus increasing the defectivity of the entire system.

2.3.3 Resistive open between the inverters (R_3)

The last resistance we consider in this study corresponds to a resistive-open defect modeled by R_3 that may occur inside the low-power SRAM cell. It creates a resistive path between M_6 and the SB node, thus involving a degradation of the voltage value at the SB node when the cell is written with a "1". The failure of the cell has only two different ranges of values compared with the previous defects. Indeed, assuming in this case a "1" is written in the cell, the following behaviors can be observed:

- when the resistance is within the range $0 \div R'_Z$ the system works correctly, even if we perform a read operation either in only the NM or after switching from LPM to NM.
- when the resistance is above R'_Z the cell undergoes a failure when a read operation is performed right after a write operation (RAW) either when the cell is kept in NM or when it passes through LPM (the associated model is dRDF).

There is no impact on power consumption in this case and a minimal impact on the SNM that may not compromise the reliability of the cell.

2.3.4 Discussion: symmetry of the cell

The study has focused so far on the three main resistive defects that we considered in our analysis, but we have already pointed out the symmetry of the low-power SRAM cell used for our purposes. These other resistive defects may have an impact on the system in similar ways to what we focused on before. Considering the effect of the resistive defects considered so far (R_1, R_2, and R_3), we have the same effects with the other resistive defects (from R_4 to R_{10}). In particular, when a "0" is written in the cell, based on the symmetry of the cell, we can state that:

- we have the same effect of R_1 considering the INV-$M_{1,2}$ with the R_8 resistance.
- we have the same effect of R_2 considering the INV-$M_{1,2}$ with the R_9 resistance.
- the R_3 resistance corresponds to R_6 when considering the INV-$M_{1,2}$.

Considering the effect of the other three resistances (R_4, R_5, and R_6), we have the same effect with the other symmetric resistive defects as explained in the following. When a "1" is written in the cell, based on the symmetry of the cell, we can state that:

- we have the same effect of R_1 considering the INV-$M_{1,2}$ with R_4 or R_7.
- we have the same effect of R_2 considering the INV-$M_{1,2}$ either with R_5 or R_{10}.
- the R_6 resistance corresponds to R_3 when considering the INV-$M_{3,4}$.

Moreover, regarding SNM, we notice that the behavior of the cell is symmetric in all cases except for the resistive defects on the pull-down of the case of the inverter. The reason we obtain this difference lies in the reduction of the rail-to-rail value of the supply voltage from the ground, thus increasing the threshold voltage that, in this case, may compromise the correct functionalities of the inverter inside the cell. Table 2.1 summarizes the correspondence of each resistive defect with each other when a "0" or a "1" is written in the cell.

Table 2.1 Defect correspondences due to the symmetry of the cell.

Resistance	**Write "0"**	**Write "1"**
R_1	R_8	N.A.
R_2	R_9	N.A.
R_3	N.A.	R_6
R_4	N.A.	R_7
R_5	N.A.	R_{10}
R_6	R_3	N.A.

2.4 Tests for resistive defects in the low-power SRAM cell

In this section, we summarize our analysis of the capability of different test solutions to detect the considered faults caused by resistive defects, considering the different values of each one and the related fault model associated with these defects. To better explain our analysis, it is necessary to list the tests we considered for our purposes:

- **March test**; A test algorithm that is used for RAM and consists of a sequence of so-called March elements (a write and read action with increasing and decreasing addresses for each cell) that are performed on the device under test (DUT) [41].
- **Low-power retention test (LPR)**; It corresponds to a few steps that are performed in the cell. First, a “0” or a “1” is written into the cell, then it is switched into LPM, then it is turned back to NM, and finally the cell value stored is read.
  To decrease the test time for our analysis, we considered only a read action right after switching to NM in the measure of nanoseconds instead of milliseconds. This is because we avoid the time for the entire device to switch in LPM. This test does not cover a read action enacted after a long time.
- **IDDQ test**; It corresponds to the measurement of the current during SRAM operations. It works on measuring the supply current in the quiescent state (when the circuit is not switching and inputs are held at static values).

Usually, there is no static current path between the power supply and ground, except for a small amount of leakage. Even if it could be a simple way to identify defects in a circuit, it is anyway time consuming and an expensive test to be done compared to the most common test techniques.

- **Read Equivalent Stress test (RES)** [42][43][44][45]. It is an alternative methodology that can be employed, which does not require current measurements or passages to LPM.
  This test methodology is based on the Read Equivalent Stress method. The RES test involves repeated reading operations, which cause stress on the faulty cell. These operations are performed not on the faulty cell but on the other cells in the same row.
  To implement this test solution, we consider a row of cells in the same word line. Firstly, an operation is performed inside the faulty cell, then the other cells are selected. The WL signal continues acting on the other cells but has an impact on the faulty cell because of the stress created by the indirect read action on the same word line.
  Obviously, the most stressed cell will be anyway the first cells of the rows, starting from each way the test is performed (see Figure 2.10). Furthermore, when the system does not select the cell, the pre-charge circuit stays in the active status and continues charging the bit lines at the VDD value.
  On the other side, when the cell is selected, the pre-charge circuit switches off and an operation can be performed on the cell. It is possible to use a built-in self-test (BIST) to perform this type of test.
  Providing that the BIST engine can apply these stimuli, i.e., execute a long enough uninterrupted sequence of selective read operations on the cells of the same row, this method allows an easier and faster way to apply and test than the tests based on Low-power retention or IDDQ.
  To extend the effectiveness of the technique for our purposes, this test must be performed at the minimum VDD value admissible by the specifications of the system and the technology.

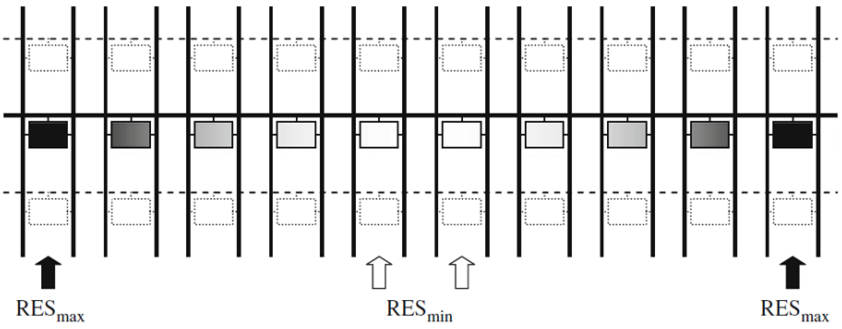

Fig. 2.10 Most stressed cells in a Row using the RES test.

2.4.1 R_1 case

In this paragraph, we analyze the two main resistive-bridging defects R_1 and R_8 that are involved in the test when the cell is written as a "0", but (as anticipated before) we would find the same results with R_4 and R_7 when the cell is written as a "1" due to the symmetry of the cell. The faults caused by the R_1 and R_8 defects can be tested through different types of techniques. We analyze the test options in each range case, starting from the minimum to the maximum resistance values. Due to the misbehavior caused by R_1 (and R_8 for symmetry) in the range $0 \div R'_X$, any test writing and reading a "0" for R_1 and R_8 can detect it. Therefore, a basic March test is sufficient. For the defects within the $R'_X \div R''_X$ range, a March test is not suitable. However, it is possible to resort to other kinds of tests, such as:

- Low-power retention (LPR) test.
- IDDQ test; If a defect in this range occurs inside the cell, a higher current consumption by the whole system is detectable in NM; indeed, we have a higher power consumption when we have already written the cell. Then, after switching to LPM and returning to NM, the current consumption decreases, remaining within the specification limits.
- Read Equivalent Stress (RES) test in a particular sub-range.

Above R''_X (i.e., when the defect approximates an open circuit) no fault is present.

2.4.2 R_2 case

In this sub-section, we will analyze the two resistive-open defects, R_2 and R_9 (for simmetry), that are involved in the test when the cell is written with a "0". We would find the same results with R_5 and R_{10} when the cell is written with a "1". When considering the fault corresponding to R_2 and R_9 in the range $0 \div R'_Y$ (where R'_Y is a relatively small value, depending on the specific cell), there is no functional effect and so no fault to detect. For R_2 and R_9 within the $R'_Y \div R''_Y$ range, a March RAW (read after write) test is sufficient to detect a bit-flip in the cell without passing through LPM. For R_2 and R_9 above the R''_Y value, it is possible to resort to the following types of tests:

- LPR test: write the cell with "0", then isolate the cell through WL signal by acting on M_5 and M_6, enter LPM, then return to NM, and at last read the cell value.
- A March RAW (read after write) test without passing through LPM.

2.4.3 R_3 case

In this paragraph, we analyze the last two resistive-open defects R_3 and R_6 that are involved in the test when the cell is written with a "1" or a "0", respectively. The failure of the cell has two different ranges of values with respect to the previous defect. Indeed, the following behaviors can be observed:

- when the resistance is within the range $0 \div R'_Z$, the system works correctly, even if we perform a read operation either in the NM or after coming back from the LPM.
- when the resistance is above R'_Z, the cell undergoes failure in the case of a read operation protocol after a write operation. This failure can occur either if the cell is in NM or passes through LPM, so a March RAW is sufficient to detect the fault.

2.5 Experimental results

This section presents the experimental results based on the main simulation of a 6T SRAM cell in 160nm STMicroelectronics technology, addressing the defects discussed in the previous sections. For our purpose, we used Cadence Virtuoso™ for the schematics and the ELDO™ SPICE simulator for simulations and analysis. Figure 2.11 shows a complete low-power SRAM sub-system featuring all the components of a single cell that we used for our simulations. This system is made up of three main components:

- a 6T-SRAM cell with INV-$M_{3,4}$, INV-$M_{1,2}$ and two pass-transistors, M_5 and M_6, through which bit-lines can access the cell when the WL signal is high.
- a back-bias circuit, increasing the $VGND$ value to reduce leakage currents when the cell switches from Normal-Mode (NM) to Low-Power-Mode (LPM).
- a pre-charge circuit (M_7, M_8, M_9), which charges the bit-lines to VDD when the cell is not selected for any operation. It is driven by the *PCON* signal that works either before an operation when the cell is selected or when the word line is activated, and other cells are selected for any operation.

To implement our analysis, we properly drive the WL signal to write the cell, the *PDM* signal in order to enable/disable the LPM in the cell, the *PCON* signal to enable/disable the pre-charge circuit, the *BLCON* signal to enable/disable the bit-lines, and thus the column of the cells (we consider them with high-impedance end). The experimental results of our analysis for the three main resistive defects (R_1, R_2, and R_3) are illustrated in the following sub-sections for each of the considered defects.

2.5.1 R_1 case

Figure 2.12(a) shows the results of the simulation of an LPR test in the $R'_X \div R''_X$ range, which detects a bit-flip after switching the mode of the cell and reading the data stored with R_1 injected in the cell. For the sake of comparison, Figure 2.12(b) depicts the behavior of an LPR test of a fault-free SRAM cell. Looking at the figure,

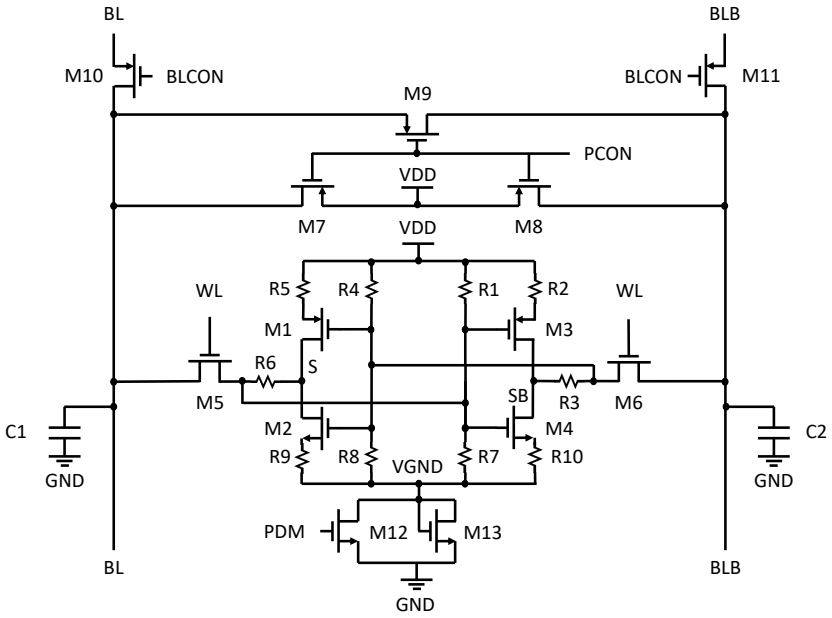

Fig. 2.11 6T-SRAM with pre-charge circuit.

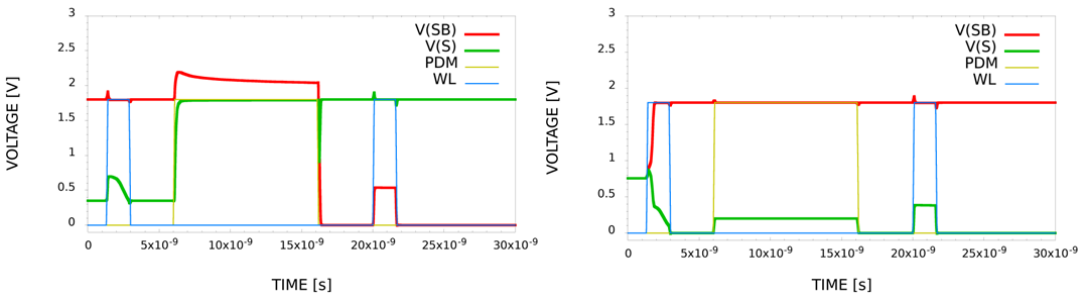

Fig. 2.12 (a) Simulation of an LPR test with a bit-flip due to R_1; (b) simulation of an LPR test without defects.

we can see that after exiting LPM, the SB voltage node sharply decreases causing a failure in the cell behavior and allowing defect detection when the reading action is performed in the cell. The LPM is usually in the millisecond range at least, but it has been shortened for clarity.

In Figure 2.13, we considered a resistive value in the sub-range of $R'_X \div R''_X$, in which we simulated the effect of the RES test during which we have a bit-flip after several indirect read operations. To summarize, among all the tests we performed in this range, the considered defect is detected by the RES technique, as well.

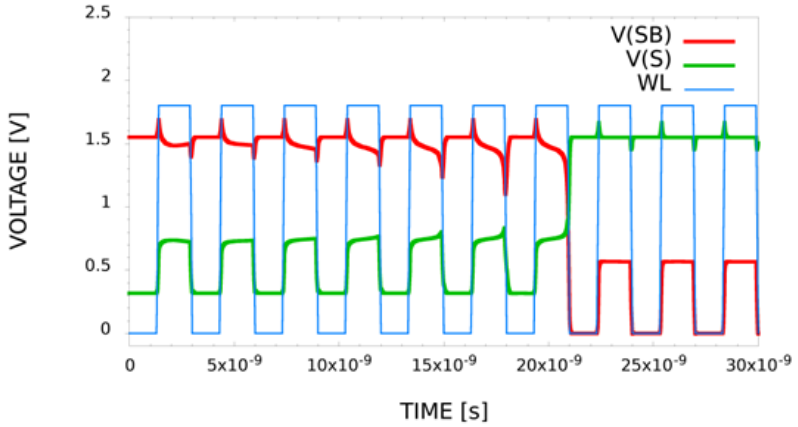

Fig. 2.13 Simulation of a RES test with a bit-flip due to R_1 (allowing the fault detection) after several indirect read operations.

This type of simulation was performed using the lowest power supply voltage that allows the system to keep working. The reason behind this choice is that when we consider the sub-range and perform reading operations in the cell, this continues working without any effect. For this reason, and in agreement with the capability of the tester to work with lower power supply voltage, we consider this solution more affordable than the previous one, not only in terms of time but also in terms of costs. The only drawback is the perfect knowledge of the layout of the row of the cell required because we need to know it in order to perform writing/reading actions in the other cell of the same row of the defective one. However, if this test was to be performed with the lowest power supply value that the system allows, the effect could be visible in very few reading operations. This mechanism is useful because, as we explained in the previous chapters, the device that embeds the SRAM requires milliseconds of time to pass from NM to LPM. With this type of test, we may detect that particular defect in a shorter time. This particular range could also have an impact on the IDDQ test and the SNM as well. In particular, the IDDQ technique may detect a higher current consumption before any kind of operation may be performed inside the cell. Looking at Figure 2.14, we can see the trend of the current during the LPR test simulation with a decreasing R_1 defect inserted in the cell. In particular, the green line depicts the current consumption of an LPR test with a fault-free cell. We can see that the quiescent current maintains its low value during the operation inside the cell. The red line depicts the current consumption of an LPR test in the sub-range of $R'_X \div R''_X$. In this case, we can see that the quiescent

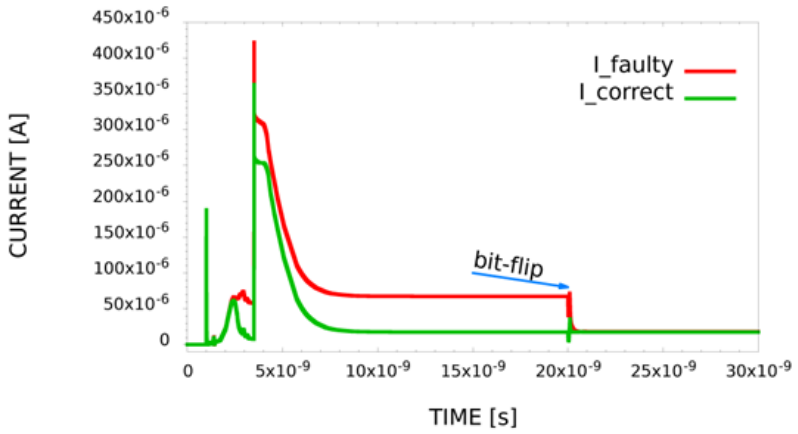

Fig. 2.14 Effect of the R_1 defect on the absorbed supply current.

current is higher than the previous one, that is up to 20% more than the previous one. As the arrow indicates on the figure, the more R_1 is increased, the lower the effect on the power consumption we have inside the cell until we have no impact when the R_1 value is below R'_X (as the arrow indicates in the figure when we have the bit-flip).

Regarding SNM, Figure 16 shows a simulation that represents the characteristics of SNM when R_1 is injected in the cell. Considering R_1 in the $R'_X \div R''_X$ value range inserted in the cell, the red line depicts the IN/OUT characteristic of the INV-$M_{3,4}$ in node SB depending on the S node as a source. Vice-versa, the green line depicts the IN/OUT characteristic of the INV-$M_{1,2}$ in node *S* depending on the SB node. By inverting the axes, the butterfly-line graph is produced. As we can see, the defect inside the cell impacts the SNM, reducing the stability of the cell that is up to 30% more affected than the typical one.

According to our simulations, for the considered cells, the values for R'_X and R''_X are $20k\Omega$ and $35k\Omega$, respectively. Table 2.2 summarizes the defect effects and the fault detection capabilities of each test in each range of values. The RES test detects the fault only in a limited sub-range of $R'_X \div R''_X$ (around $30k\Omega$).

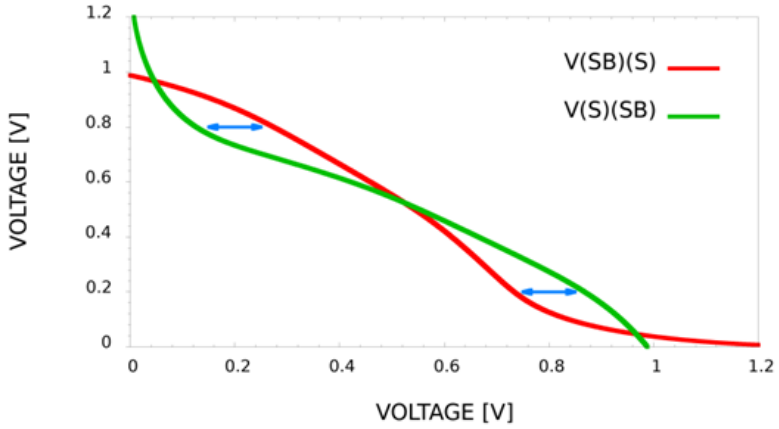

Fig. 2.15 Effect of the R_1 defect on SNM.

Table 2.2 Effectiveness of the different test methods with respect to the R_1 defect.

	$< R'_X$	$R'_X \div R''_X$	$> R''_X$
LPR TEST	Detected	Detected	No Effect
March TEST	Detected	Undetected	No Effect
IDDQ TEST	N.A.	Detected	No Effect
RES TEST	N.A.	Detected	No Effect

2.5.2 R_2 case

Figure 2.16 shows the results of the simulation of an LPR test considering the R_2 defect above the R''_Y value. The simulation detects a bit-flip after reading the stored data when the cell switches from LPM to NM.

In this case, it is useless to use the IDDQ technique due to the impossibility to detect any effect on the power consumption of the cell. For this reason, we did not consider the IDDQ test for this resistive defect. In this case, we have no effect on SNM for R_2, but when considering the R_{10} resistance in the $R'_Y \div R''_Y$ range and looking at Figure 2.17, we can see a large impact on the SNM that may affect the stability of the cell by 40% more than the typical one. This situation is caused by the MOS resistance (M_{13}) in the VGND node that interferes with the threshold of the inverter and compromises the static noise margin of the cell.

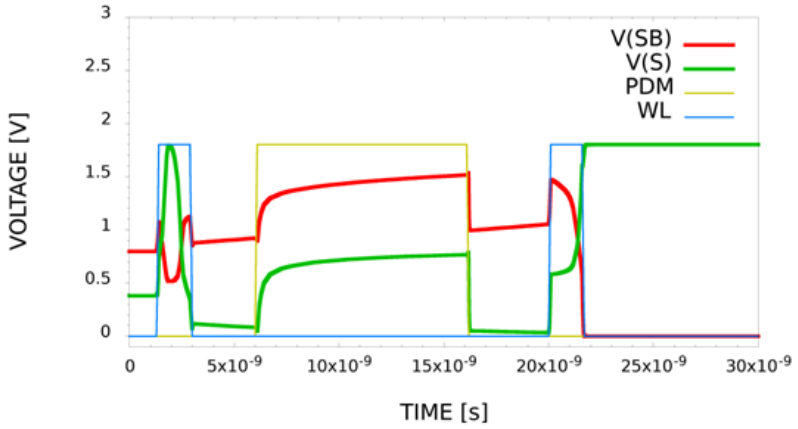

Fig. 2.16 Simulation of an LPR test with bit-flip because of R_2 effect.

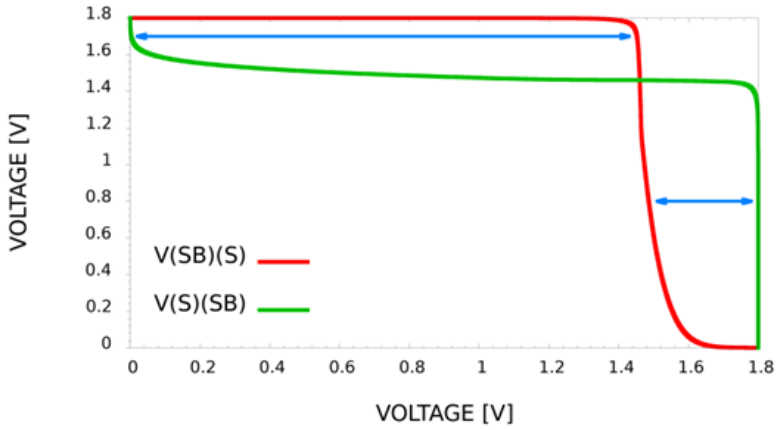

Fig. 2.17 Effect of the R_{10} defect on SNM.

According to our simulations, for the considered cells, the values for R'_Y and R''_Y are $3M\Omega$ and $15M\Omega$, respectively. The effect of every test when the defect size belongs to each range is summarized in the following Table 2.3.

2.5.3 R_3 case

Figure 2.18 shows the results of the simulation of an LPR test considering the R_3 defect. The simulation shows that the LPR test detects a bit-flip after reading the stored data when the cell switches from LPM to NM. Even in this case, the IDDQ technique is useless due to the impossibility to detect any kind of effect in the power consumption of the entire cell with the considered R_3 resistance.

Table 2.3 Effectiveness of the different test methods with respect to the R_2 defect.

	$< R'_Y$	$R'_Y \div R''_Y$	$> R''_Y$
LPR TEST	No Effect	Undetected	Detected
March TEST	No Effect	Detected	Detected

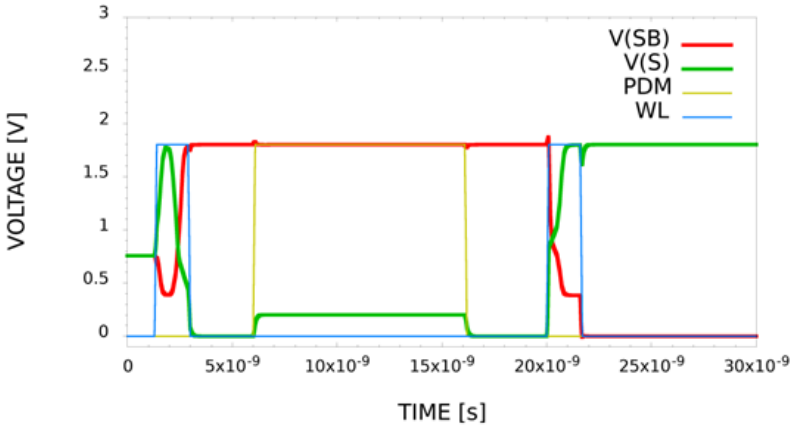

Fig. 2.18 Simulation of an LPR test with a bit-flip due to R_3.

Figure 2.19 shows an SNM margin analysis considering an R_3 value above R'_Z implemented in the cell where we can see a not-so-explicit impact on the stability of the cell, except for being 15% more affected than the typical SNM of the faulty-free cell.

According to our simulations, for the considered cell, the value for R'_Z is $30k\Omega$. The effect of every test when the defect size belongs to each range is summarized in Table 2.4.

2.5.4 Discussion

We have seen in the previous sections that, for each case, the defects that may occur in the circuit may produce different effects and may be detected by different tests. The best test strategy must be evaluated case by case considering the technology used (hence, the likelihood of the different defects, and the range of their values), the product quality objectives, and the maximum costs one can afford. To summarize the conclusions of our analysis, we can state that traditional retention tests generally require more time than a March test applied via BIST. Similarly, any test based on

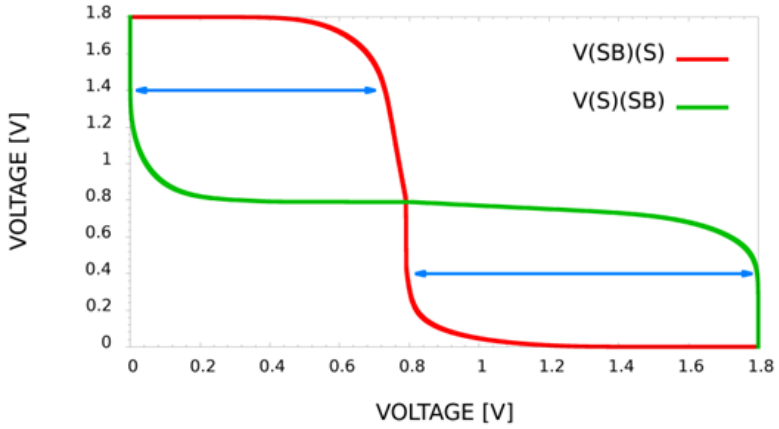

Fig. 2.19 Effect of the R_3 defect on SNM.

Table 2.4 Effectiveness of the different test methods with respect to R_3 defect.

	$< R'_Z$	$> R'_Z$
LPR TEST	No Effect	Detected
March TEST	No Effect	Detected

moving the memory to LPM and back or on the IDDQ current measure may be relatively long and expensive. Other solutions able to detect the addressed effects more quickly are thus highly welcome. The performed analysis allows one to evaluate the pros and cons of each case and choose the best test solution.

To conclude, in Tables 2.5,2.6,2.7 we can see all the possible cases we discussed in this chapter considering the real values analyzed through the SRAM model, allowing one to identify the test able to spot a defect in the cell.

2.6 Conclusion

This chapter provides a comprehensive overview, supported by experimental results, on the possible effects of some resistive defects affecting a 160 nm low-power SRAM cell, which uses the back-bias technique to reduce the leakage currents when entering the low-power mode. The analysis focuses on several resistive defects provoking a bit-flip of the cell when moving to LPM, taking into account the symmetry of the entire SRAM cell. Furthermore, the chapter reports an evaluation

Table 2.5 Summary of the effectiveness of the different test methods with respect to R_1.

	$<25k\Omega$	$25k\Omega \div 35k\Omega$	$>35k\Omega$
LPR TEST	Detected	Detected	Detected
March TEST	Detected	Undetected	Detected
IDDQ TEST	N.A.	Detected	No Effect
RES TEST	N.A.	Detected	No Effect

Table 2.6 Summary of the effectiveness of the different test methods with respect to R_2.

	$<3M\Omega$	$3M\Omega \div 15M\Omega$	$>15M\Omega$
LPR TEST	No Effect	Undetected	Detected
March TEST	No Effect	Detected	Detected
IDDQ TEST	N.A.	N.A.	N.A.
RES TEST	N.A.	N.A.	N.A.

of the effects of each defect and the associated fault models in order to assess the effectiveness and advantages/limitations of different test methods. The analysis can be used by test engineers to select the test solution(s) more effectively to be used for each product. The analysis methodology considering the low-power architecture can also be extended to more advanced technology nodes.

Future works will include:

- the evaluation of the most likely values for such defects through low-level failure analysis of the specific technology.
- the exploration of possible BIST solutions to implement the considered tests.

Table 2.7 Summary of the effectiveness of the different test methods with respect to R_3.

	$<3M\Omega$	$>3M\Omega$
LPR TEST	No Effect	Detected
March TEST	No Effect	Detected
IDDQ TEST	N.A.	N.A.
RES TEST	N.A.	N.A.

Chapter 3

Cell-Aware Testing

This chapter introduces the last main topic of the thesis presented in the introduction in chapter 1. It will be introduced:

- the adoption of Cell-Aware Testing (CAT) [2] for the purposes of the work that it has been done, to target together both common and cell-aware fault models and obtain the best trade off between fault coverage and testing time; several flows will be considered and evaluated to identify the best flow to detect faults coming from common fault models and cell-aware fault model [46];
- the evaluation of which flow may be more effective in terms of Fault coverage using the obtained test patterns and other sets targeting other fault models that will be compared and fault simulated with fault lists belonging to common, CAT and delay fault models [47].

In the field of integrated circuit (IC) testing, the detection of defects is crucial to ensure the reliability of the final product. CAT has become an option for an increasing number of semiconductor companies. CAT has been introduced as a different approach that aims to improve the detection of internal defects within standard cells: it involves using specific patterns to detect intra-cells faults. Typically, CAT is adopted in the context of scan chain tests, and patterns are generated with an Automatic Test Pattern Generation (ATPG) tool. Moreover, past studies have extensively shown the capability of CAT to identify some physical defects of microchips that would otherwise remain undetected using traditional fault models, only.

However, due to the higher number of patterns generated, an improper CAT-related ATPG flow can lead to a longer test application time. This means higher costs for semiconductor companies, thus reducing the advantages of CAT. Nevertheless the CAT effectiveness, it may be not capable to detect defects that arise under certain conditions where System-Level Test (SLT) may be more effective [48] [49]. Among the variety of fault models that can be used to target the many possible defects in a circuit, delay faults (transition and path delay) have been used for many years. Both delay and cell-aware faults can be caused by several factors, such as manufacturing defects, environmental conditions, and aging effects. It appears evident that a range of defects can be modeled following an electrical analysis of the device or its cells. Therefore, to understand the functioning of certain defects and to detect them, a low-level, thus electrical, analysis is required. Having performed this operation on memory devices, I then applied the same mechanism to the CAT. This was done in order to replicate the same operation with commercial tools within the cells. The aim of the work that will be introduced in this chapter can be split in two main contributions:

- the first contribution consists in a overview of different ATPG flows supporting CAT, showing advantages and disadvantages of each approach. Each flow will be evaluated together with traditional and cell-aware fault models in terms of achievable fault coverage and pattern count.

- the second contribution consists in processing the generated test patterns to evaluate the impact that each considered fault model has with other fault lists. In particular, an investigation about the application of the generated test patterns with the transition and path delay fault models in comparison with others developed with the cell-aware approach will be presented, in terms of fault coverage, pattern count and test generation time.
  The study shows that the combination of the path delay fault model and cell-aware testing can lead to improved fault coverage and lower costs. The experimental results will be presented over a wide range of open-source benchmarks and on a RISC-V design using a proprietary industrial technology library.

3.1 Introduction

The continuous miniaturization of technology nodes, combined with the ever-increasing complexity of *Systems-On-Chips* (SoCs), especially for *Application-Specific Integrated Circuits* (ASICs), can contribute to an increase in defectiveness of silicon products. Furthermore, nowadays SoCs products are made more than ever difficult to test, turning harder to achieve low *Defective Parts Per Million* (DPMM) levels, having gained prominence due to their customized functionality and high-performance capabilities. However, the escalating complexity of ASIC designs poses significant challenges in ensuring their dependable and error-free operation. Consequently, thorough testing of ASICs becomes indispensable to identify potential faults that could undermine their functionality. Customers, on their side, ask for higher quality levels, especially in safety-critical and mission-critical applications.As a consequence, semiconductor companies constantly improve manufacturing processes to systematically increase yield. At the same time, they try to reduce the defect level for integrated circuit (IC) designs, regardless of the technology node used. To this purpose, new testing techniques are needed to achieve the least defect rate. It is quite well known that physical defects can occur at any time during the manufacturing process of semiconductor devices. For this reason, *Very-Large-Scale Integration* (VLSI) IC testing plays a key role in detecting possible defects. Basically, *Automatic Test Pattern Generation* (ATPG) tools use fault models to represent faults into a circuit and create patterns to be used by product engineers on automatic test equipment (ATE). Although common fault models such as stuck-at faults (SAFs), bridging faults (BFs) and transition delay faults (TDFs) can detect a high percentage of defects within a circuit, they view each standard-cell as a black box by considering only the inputs and outputs of each cell. As a result, these fault models might not consider some type of physical defects that could possibly occur in the ICs production. ICs require higher quality tests to be applied at the end/during the manufacturing process, evaluating not only defect-oriented tests [7] but also physical-aware tests [50].

Cell-aware testing (CAT) [4] [51] [52], unlike other fault modeling approaches, can model a considerable number of intra-cell defects. These cell-aware faults can be then tested with suitable test patterns generated by an ATPG tool, increasing the overall quality of the manufacturing process by reducing the number of test

escapes. As shown in [53][54], the adoption of CAT (supported by proper ATPG tools) produces a higher number of patterns compared with the other fault models. This in turn causes an increase in testing time on the ATE. To reduce this negative side effect, it is important to find ways to reduce the number of generated patterns and obtain the best trade-off between coverage of all faults, including traditional fault models (stuck-at and transition delay faults) and application time. For this reason, a good strategy for test engineers is to find the most appropriate test development flow implementing CAT without compromising fault coverage of traditional fault models while maximizing at the same time the coverage of cell-aware (CA) faults addressed by CAT. Moreover, another aspects of ASIC testing is the accurate recognition and detection of path delay faults [55], as it directly impacts the overall system performance. Path delay faults [56] [57] are particularly relevant in digital circuits, as they can lead to timing violations, signal integrity issues, and, ultimately, functional failures. Delay fault models [58] can allow the detection of potential delay defects into designs when a transition signal (slow-to-rise and rise-to-fall) is run into a specific gate and exceeds the clock period. The path delay fault model focuses on testing, identifying, and analyzing the delays encountered by signals as they traverse various paths within a circuit. By considering the individual delays at each stage and the cumulative effect, the path delay fault model provides valuable insights into potential timing issues.

3.1.1 First contribution - CAT flows comparison

Considering the above-mentioned limitations of CAT, the first goal of the work done in this chapter is to propose a comparative overview of different ATPG flows that include CAT and the evaluation of these approaches in terms of fault coverage and pattern count. Previous works [59][54] have demonstrated the use of certain ATPG flows supporting CAT. The purpose for the next sections is not to prove the effectiveness of CAT or to propose new algorithms. These were addressed in previous research works that have presented results comparing different failure models [53], devising different methodologies [60][61][62] or improving the characterization of libraries [63]. Differently, in these sections a comparison between different flows (all supported by commercial tools) is performed, to allow test engineers to fully assess their advantages and disadvantages. These flows are evaluated in terms of

the coverage obtained and the number of patterns (that ultimately contribute to the testing time).

The experimental results were obtained using commercial tools, a proprietary technology that lends itself well to the use of CAT and using open-source benchmarks, all of them with the same scan chain architecture, to make the experiments as design-independent as possible. Then, we evaluate the approach of cell-aware testing on path delay faults, to enhance the effectiveness of path delay fault detection.

3.1.2 Second contribution - comparison of different fault models

The second purpose of the work presented in this chapter aims to give a comparative evaluation of the effectiveness of test stimuli generated targeting one fault model out of those discussed before when they are evaluated with respect to a different fault model. In this way we pave the way towards a more effective test pattern generation in terms of achieved defect coverage and pattern count. Through an in-depth examination of the path delay fault model and the potential of cell-aware testing the methodology of the test may be optimized for obtaining efficient defect coverage in ICs. In fact, CAT, if properly used, can improve fault coverage and pinpoint specific faults associated with path delays. By directly targeting the cells that contribute significantly to path delays, cell-aware testing offers a more refined and efficient testing methodology. Linking the cell-aware testing pattern set with the path delay fault model pattern set we ensure a comprehensive test coverage for critical timing issues. This integration allows designers and test engineers to focus their efforts on specific areas of concern, optimizing test resources and reducing overall test time.

By addressing path delays, researchers and industry professionals can strive towards the production of ASICs with improved quality and performance. In particular, among the fault models that can detect physical defects in a circuit, cell-aware testing (CAT) can model several intra-cell defects. The cell-aware faults, if detected, can increase the overall quality of the manufacturing process by reducing the number of test escapes.

3.2 Background and motivations

In this section the *state-of-the-art* of cell-aware testing and delay fault models will be presented to introduce the main body of the work done. At the same time the case studies used for the experiments will be introduced together with each design detail.

3.2.1 Cell-aware testing (CAT)

Physical defects may occur during the fabrication process of semiconductor devices. To detect such defects, common fault models have been proposed [64] and used for the generation of test patterns such as stuck-at (SA) [65], bridge [66][67][68], transition (TR) [69][70], as well as timing-aware [71], layout-aware [72], N-Detect [73] and gate-exhaustive [74]. More and more frequently, customers ask companies to increase the efficiency of such fault models to reduce tests escape and hence, too many defective parts from their suppliers. In fact, it can happen that cell-internal defects may remain undetected when using traditional ATPG tools with traditional fault models. For this reason other types of fault models based on SPICE netlists with parasitic have been done [4].

Cell-aware testing is a methodology increasingly adopted for modelling and testing intra-cell defects such as short circuits, open circuits and transistor defects that may be neglected by traditional inter-cell fault models. It is based on a post-layout transistor-level netlist including parasitic objects, resulting in a defect-based ATPG approach, which can be applied to large, state-of-the-art designs.

The CAT methodology consists of a preliminary phase aiming at the cell-aware fault models generation. This is a one-time task performed for each cell of the technology library. This step requires the SPICE netlist, the technology library, and the timing information data (liberty file). This step starts with the layout extraction phase, then an analog fault simulation is performed, followed by a synthesis phase to create the Cell-Aware library models to be used during the IC-level ATPG [75] [76] [77]. During the layout extraction, the tool extracts from the cell structure (i.e., a transistor-level netlist with layout information) a list of possible defects. The tool reads the layout data of the selected library cell and creates a SPICE transistor netlist

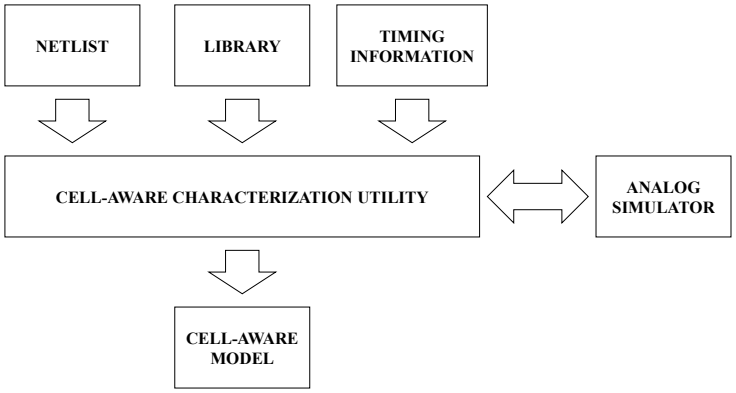

Fig. 3.1 Cell-aware characterization flow.

in *detailed standard parasitic format* (DSPF) including the parasitic elements (such as resistors and capacitors) which is stored in a file. Then, for each of the previously extracted defects, an analog simulation is performed to determine the complete set of cell-input combinations that detect the intra-cell defect and the list of Cell Aware (CA) faults. It starts with the extraction of the target defects from the DSPF SPICE netlist. The resulting considered defects are stored in the cell-dependent defects file. The analog fault simulation exhaustively (i.e., considering all possible input stimuli) compares the outputs of the defect-free transistor-level netlist against the outputs of each defective netlist (i.e., the netlist with the defect).

The CAT methodology differentiates the following different defect types:

- open: any cell-internal open defect, such as an open in poly, metal, diffusion, or vias. In the SPICE netlist, these defects are represented as resistors with very high resistance values.
- bridge: any cell-internal bridge defect such as bridges between adjacent objects in the same layer or different layers in a standard-cell layout. In the SPICE netlist, these defects are represented as capacitors with resistors in parallel with them.
- transistor open: any cell-internal transistor defect that will switch a transistor partially on with a certain resistive value.

- transistor short: any cell-internal defect that will switch a transistor partially off with a certain resistive value.
- port bridge: a bridge between a port and any other port of the cell.
- port open: a disconnected port with the rest of the system.

The performed simulations correspond to analog transient analysis simulations, which determine the voltage on the cell output at a calculated strobe time. Both a static (one time frame) and a dynamic (two time frame) analog simulation is performed. A defect is labelled as detected when the defective cell output voltage deviates from the defect-free voltage by more than a specified percentage of the supply voltage for at least one input combination. A defect is labelled as *static-CA* fault (CAT-STAT) when there is an output difference for at least one vector in a specified time instant. Defects requiring a pair of vectors to be detected are labeled as *dynamic-CA* faults (CAT-DYN). This analysis requires comparing at regular intervals the output voltages of the defect-free and defective cell. When the defective cell output voltage deviates from the defect-free only for a given number of strobe instants, the defect is labelled as dynamic. All analog simulations, including the creation of the stimuli, are fully automated by the CAT library view generation tool. For the static analysis, exhaustive one-time frame stimuli are analyzed. For the delay analysis, robust two-time frame stimuli are analyzed by default, but also exhaustive two time frame stimuli can be analyzed on user request. The result of this process is a defect matrix for the library cell. Each matrix contains the input values detecting a particular defect and the classification of the defect as static or dynamic. The third step is the cell-aware fault model synthesis. During this step, the defect matrix is optimized in order to reduce the number of test conditions required for detecting the cell defects. In fact, the purpose of the step is to identify and store the cell input conditions that are useful in detecting the defects inside each cell. At the end, an exhaustive defect matrix in order to generate the corresponding CAT library view that is created and stored in the CAT model file. For each detected cell-internal defect, the CAT model file contains one or more alternative test conditions for detecting the corresponding defect. This ensures that the subsequent CAT-ATPG still has the freedom to choose between all alternative test conditions for detecting a certain cell-internal defect, while maintaining a very compact test pattern set for a complete

design. The outcome of the process described previously is then integrated in the standard test flow where the ATPG tool uses all the traditional and CAT fault models to cover all the defects inside an IC [7], generating the scan vectors able to detect all the considered faults. Figure 3.1 summarizes the flow used for the characterization of an industrial library.

3.2.2 Delay fault models

A delay defect in a circuit causes the cumulative delay of a path to exceed some specified duration. The combinational logic path begins at a primary input or a clocked flip-flop output, contains a connected chain of gates, and ends at a primary output or a clocked flip-flop input. The propagation delay is the time that a signal event takes to traverse the path. Both switching delays of devices and transport delays of interconnects on the path contribute to the propagation delay. Common fault models, such as stuck-at, assume a permanent change in the data over time, so they can be defined as infinite delay faults. On the contrary, fault models based on the timing delay can be defined. In particular, the most common delay fault models include the Transition delay faults (TDFs), if we refer to each gate of the circuit, and the path delay faults (PDFs), if we refer to each path of the circuit. It means that each gate in a path can meet its timing specification, but the propagation delay through a path (a given sequence of gates) might exceed a specified value.

The disadvantage of the path delay model is the high number of possible paths in a circuit. To summarize, timing faults are detected with the delay fault model. Minor timing problems in the system, for instance if a signal is slightly delayed, may not affect the system at all. However, longer delays through gates can make the system fail to meet its timing specification. Therefore, delay testing is required. Considering the most common fault models, the transition delay fault model may be considered as a special case of the Stuck-at fault (SAF) model, in which when a gate of a cell is considered, it fails when a transition is performed in the cell delaying the transition on its output. To test a TDF, a stimulus (slow-to-rise and rise-to-fall vector) must be applied to a circuit input (input port or scan flip-flop), propagated through the target cell and then to an observable output. The corresponding faults can be caused by numerous factors such as manufacturing defects, aging, and environmental factors.

The fault detected in the cell in a circuit can cause a different behavior than expected, leading to incorrect results or circuit malfunction.

While the transition delay fault model focuses on the delay of a specific gate, the path delay fault model captures small extra delays that have cumulative effect along a path that may result in faulty behavior of the circuit. A path delay fault is activated when the delay along a single path exceeds the clock period of the circuit and hence may produce a failure. A delay fault is considered detected by applying two vectors, one initialization vector to set the fault location, and in the consecutive clock cycle one transition vector or propagation vector is applied to activate the fault at a primary output. The initialization vector sets the condition in order to test a slow-to-rise or a slow-to-fall signal and the propagation vector propagates the fault effect to an output. The delay faults can be classified into sequential/combinational robust, when there is a test that guarantees to detect a target delay fault of a sequential/combinational circuit in the presence of arbitrary delays in the circuit, and non-robust, when there is a test that guarantees to detect a path-delay fault, when no other path-delay fault is present [78].

3.2.3 Case studies

This subsection presents the designs used for the purpose of this part of the work.

The open-source benchmarks presented in [79] were synthesized with STMicroelectronics' proprietary 130nm HCMOS technology for power applications. We also considered the PULPino 32-bit RISC-V by ETH Zurich [80] used for the comparison of each fault model in terms of path delay faults including the other models as well. About five hundred standard cells were characterized for CAT models. For sake of simplicity and conciseness, only fifteen out of twenty-two benchmarks were considered for the ATPG flows of the first contribution. For the second contribution the RISC-V and the rest of the open-source benchmarks have been used as well for the comparative evaluation of test stimuli of one fault model with respect to a different fault model. All the experiments were performed resorting to a commercial CAT-ATPG tool. All the standard-cell characterizations were performed by a commercial cell-aware tool. Whereas, for the extraction of the paths, a STA commercial

tool was used. It is important to consider that for the experiments a single scan chain was used for each design.

Considering other works that used open-source design to analyze path delay faults [81][82], this choice stems from the purpose of this work, i.e., comparing different ATPG flows and the efficiency of a fault model with respect to the other ones or rather targeting different defect-oriented fault models, to observe the testing behavior through simple open architectures. Starting from these designs and approaches it is possible to observe the benefits and drawbacks of each flow in terms of coverage and testing time. In table 3.1 the main figures related to the adopted benchmark circuit designs are collected. For each column, the number of sequential (flip-flops) and combinational cells (logic-gates), the number of inputs and outputs (PIs and POs), and the number of faults collected for each fault model are defined: path-delay faults (PDFs), stuck-at faults (SAFs), transition-delay faults (TDFs), static-CA faults (CAT-STATs), and dynamic-CA faults (CAT-DYNs). It is worth noting that for some designs the cell-aware characterization tool generated more dynamic-CA (CAT-DYN) faults than static-CA (CAT-STA) faults. For instance, in the B14 design the synthesis tool inferred a considerable number of XOR-cells. For this kind of cell, the produced defect matrix has more CAT-DYNs than CAT-STAs. Therefore, the overall number of CAT-DYNs is higher than CAT-STAs.

The number of extracted PDFs is a result of all detected paths by the ATPG tool, excluding invalid and untestable paths. In the next sections the results obtained from the methodology used is presented. They are even included in the larger designs, RISC-V included as the fault models comparison has been done resorting to a larger designs set. It is important to notice that for the first contribution about the comparison between the different ATPG flow supporting CAT, only from B01 to B15 are intended to be analyzed; while the rest of the designs has been used for the obtained results of the second contribution about the comparison between different pattern sets fault simulated with different fault lists belonging to different fault models.

Table 3.1 Benchmarks details.

Designs	Flip-Flops	Logic Gates	PIs	POs	PDFs	SAFs	TDFs	STAT-CATs	DYN-CATs
B01	5	29	4	2	37	270	230	492	397
B02	4	14	3	1	27	158	124	282	216
B03	30	62	6	4	682	868	678	1,842	1,721
B04	66	146	13	8	932	2,062	1,656	5,029	4,448
B05	34	265	3	36	1,342	2,408	2,194	5,050	4,986
B06	9	26	4	6	104	318	254	544	392
B07	45	151	3	8	1,752	1,760	1,480	4,001	4,158
B08	21	78	11	4	345	852	716	1,717	1,528
B09	28	74	3	1	749	898	720	1,881	1,838
B10	17	83	13	6	468	846	734	1,632	1,350
B11	30	164	9	6	1,218	1,734	1,544	4,191	4,653
B12	121	469	7	6	2,954	5,018	4,282	11,033	9,844
B13	51	141	12	10	618	1,708	1,392	3,398	2,854
B14	215	2,977	34	54	10,064	27,408	26,004	78,237	87,402
B15	417	2,560	38	70	20,551	27,510	24,802	73,014	72,757
B17	1,316	9,208	42	98	7,895	-	81,656	-	222,566
B18	3,062	30,805	41	24	18,363	-	237,320	-	735,970
B19	6,126	62,007	50	31	30,000	-	477,616	-	1,486,346
B20	430	7,060	37	23	2,571	-	55,556	-	216,062
B21	430	7,159	37	23	2,571	-	55,772	-	214,248
B22	645	10,694	37	23	3,857	-	83,696	-	324,465
RISC-V	2,329	29,036	222	268	5,054	-	195,812	-	376,720

3.3 Methodology used for the comparison between different fault models

This section presents the methodology used for the analysis for each fault model considered in this work.

The chart in figure 3.2 represents the summary of the considered flow. The flow is composed of three parts, depending on which fault model is considered (path-delay, transition delay and cell-aware fault model). Starting from the left-top side of the figure, the static-time analysis (STA) tool extracts the path delay faults of the design (the critical and less critical ones in terms of the length of each path and minimum timing slack between the starting and ending point, considering fixed the period of the clock), hence they are included in a path delay fault list and used in the ATPG tool.

The path delay faults are extracted considering all the slack ranges in which the design works, where the clock period is previously defined. Once the list is generated and imported into the ATPG tool, the pattern set is created to detect all possible path

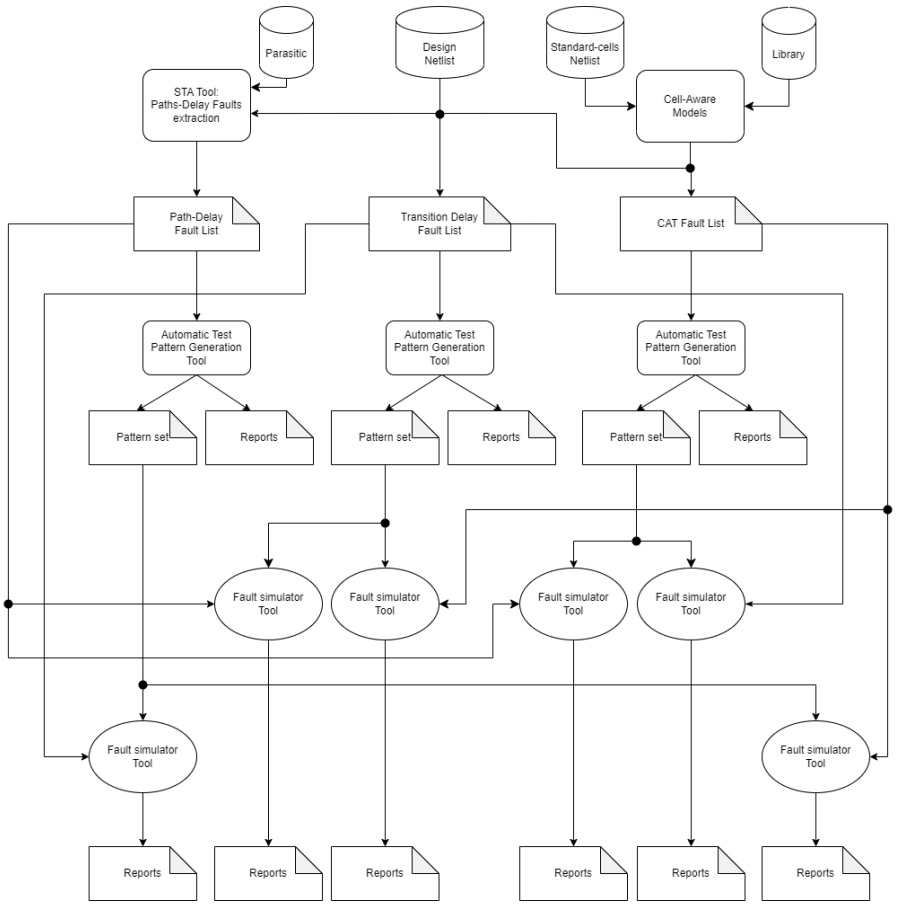

Fig. 3.2 Methodology flow.

delay faults. At the end of this flow, the final reports are provided containing fault coverage and pattern count information. The same design netlist place in top-center side of the figure is used to run an ATPG to generate stimuli detecting as many as possible transition delay faults in the design where the reports and the pattern set containing all the generated patterns are collected to be used in the second phase of the flow. In the right-top side of the figure, the CAT flow is depicted. The flow starts with the characterization of each cell in the library, using the layout SPICE netlist of each standard-cell and their own electrical-level models. Then, the cell-aware fault list containing all the static and dynamic CAT faults of the circuit is collected by the ATPG tool through which the final reports and the pattern set file are collected. The three final reports reporting the data regarded the ATPG flow are showed in the middle side of the figure 3.2. Considering the middle and bottom side of the figure, fault simulator tool is used for fault simulation. This process is used to analyze the

coverage obtained using the patterns generated targeting the different fault models when considering the fault lists of the other fault models.

In particular, following the flow depicted in the figure, using the path delay fault list, two fault simulations are performed resorting to the transition delay pattern set processed by the ATPG for the first one, and dynamic CAT pattern set previously processed by the ATPG tool for the second one. At the end of the flow the reports are reported to the bottom side of the methodology flow figure. The same process is depicted for the other fault models: this means that, first for the transition delay patterns and then for the dynamic CAT ones, two additional fault simulations are performed in each case using the different fault lists depicted in the figure. In particular, for the transition delay fault list two fault simulations are performed: the path-delay pattern set for the first one; the dynamic CAT pattern set for the second one. About the dynamic CAT fault list, other two fault simulations are performed resorting to the path-delay pattern set for the first one, and the transition delay pattern set for the second one. All the six final reports are depicted in the bottom side of the figure. The purpose of using these mixed approaches is to evaluate the fault coverage obtained by the pattern set generated targeting different fault models. In this way we will show that in some cases the patterns generated for a different fault model may achieve relevant fault coverage figures also for the other ones.

3.4 Possible ATPG flows using the CAT approach

In this section the different ATPG flows considered for the comparative analysis of the first contribution introduced in this chapter are presented. The considered flows were divided into two sets of four flows each, considering SAFs and static-CA faults for the first set (ATPG-FLW$_1$ to ATPG-FLW$_4$) and TDFs and dynamic-CA faults for the second set (ATPG-FLW$_5$ to ATPG-FLW$_8$). Since the ATPG tool differentiates the two types of cell-aware faults (static and dynamic), the SAFs flow is matched with the corresponding static-CA faults and the TDFs flow is matched with the corresponding dynamic-CA faults. Furthermore, each flow for a given fault model (e.g., SAFs) has its corresponding identical flow for the different one (e.g., TDFs). It is worth noting that the fault coverage for traditional fault models is ensured by the ATPG tool. In other words, the fault coverage for SAF/TDF is always kept at the

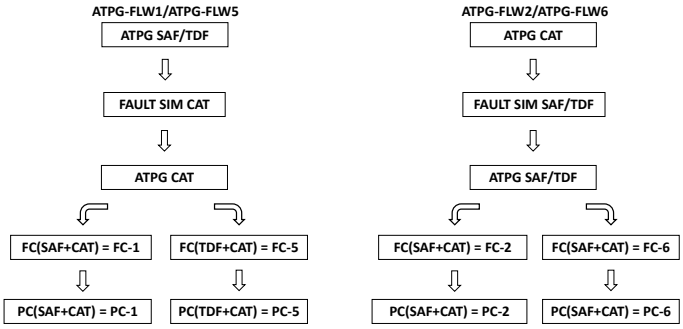

Fig. 3.3 ATPG-FLW$_1$ (ATPG-FLW$_5$) and ATPG-FLW$_2$ (ATPG-FLW$_6$).

maximum. The following subsections detail the eight flows presented above. For every flow once the CA models for the library cells have been defined, the design gate-level netlist (already including the scan-related logic) and libraries are imported into the ATPG tool.

3.4.1 SAFs/TDFs ATPG flows incremental with CA faults

Two sets of flows are presented in this subsection for the SAF and static-CA fault models for ATPG-FLW$_1$ and ATPG-FLW$_2$ and for the TDF and dynamic-CA fault models for ATPG-FLW$_5$ and ATPG-FLW$_6$. The ATPG-FLW$_1$ (and ATPG-FLW$_5$, respectively) flow implements a typical ATPG run with a SAF (TDF) model fault list that incrementally add a CAT-type flow. The ATPG-FLW$_2$ (ATPG-FLW$_6$) flow exploits the opposite approach, i.e., it implements a cell-aware ATPG run and incrementally adds an ATPG flow with a fault list of the SAF (TDF) fault model. These flows exploit the ATPG tool to work in an incremental way.

The ATPG-FLW$_1$ (ATPG-FLW$_5$) procedure is as follows:

- the ATPG step is run on the SAFs (TDFs) and suitable patterns are generated to detect them;
- the patterns generated for the SAFs (TDFs) are then fault simulated with the static (dynamic) CAT faults;

- a new ATPG step is then executed to generate further patterns to detect the still undetected CAT faults;
- finally, fault coverage (FC) and pattern count (PC) figures are collected

The ATPG-FLW$_2$ (ATPG-FLW$_6$) is reversed with respect to traditional and CA faults. In other words, the ATPG run starts with the CA faults, then the generated patterns are obtained. A fault simulation of the generated patterns with the SAF (TDF) faults is performed. Finally, the incremental ATPG run is performed to cover all the undetected SAF (TDF) faults. The pattern count and the coverage percentage are therefore collected. Figure 3.3 summarizes the representation of the above-explained flows.

3.4.2 ATPG flow with both SAF/TDF and CAT lists

The other two sets of flows are presented in this subsection for the SAF and static-CA fault models for ATPG-FLW$_3$ and ATPG-FLW$_4$ and for the TDF and dynamic-CA fault models for ATPG-FLW$_7$ and ATPG-FLW$_8$, respectively. These flows exploit the use of the ATPG run for creating patterns to find the fault coverage of the considered design. Unlike the previously introduced flows, in ATPG-FLW$_3$ (ATPG-FLW$_7$) the fault list of SAF (TDF) model and the fault list of static (dynamic) CA fault model are merged before the ATPG is executed. In the ATPG-FLW$_4$ (ATPG-FLW$_8$) flow, on the other hand, the ATPG flow is split by fault models: one traditional ATPG flow is performed on a fault list with SAF (TDF) and another separate for static (dynamic) CA faults. The obtained fault lists are then merged (preserving fault detection status) within a single flow, resulting in the summary results of the two flows considered. The collected results consider the total patterns and coverage obtained from the two separate flow.

The ATPG-FLW$_3$ (ATPG-FLW$_7$) is as follows:

- once the fault model is defined, all SAFs (TDFs) and static (dynamic) CA faults of the current design are added.

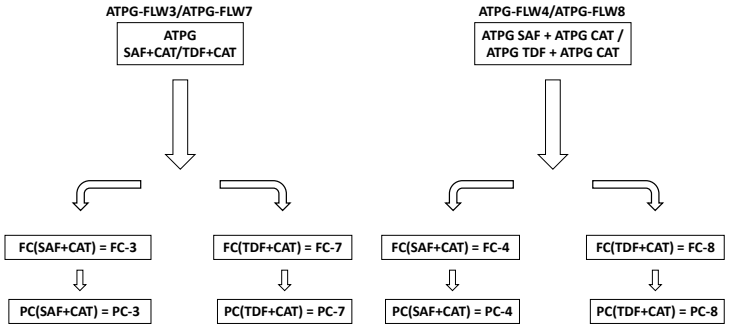

Fig. 3.4 ATPG-FLW$_3$ (ATPG-FLW$_7$) and ATPG-FLW$_4$ (ATPG-FLW$_8$).

- the ATPG is executed and all patterns created are saved, including considering the coverage obtained for all faults of both the traditional and cell-aware fault model.

The ATPG-FLW$_4$ (ATPG-FLW$_8$) consists of two different ATPG runs separately. In practice, the ATPG for SAF (TDF) fault lists and the ATPG for static (dynamic) CA fault lists are run in parallel. Then, the two runs are implemented in a single flow in the ATPG tool to collect the overall fault coverage and pattern counts derived from the separate usage of the two procedures. Figure 3.4 summarizes the explained flows.

3.5 Experimental results about CAT flows comparison

To summarize the specification of the first contribution, different ATPG flows has been considered for the comparative analysis discussed in the previous sections; the aim of the study is to overview different ATPG flows supporting CAT, showing advantages and disadvantages of each approach. Each flow is evaluated together with traditional and cell-aware fault models in terms of achievable fault coverage and pattern count. In particular, this section discusses the results of the first contribution of the chapter about the comparison of these flows for the considered designs, as

shown in table 3.2, where the fault coverage (FC) and the number of patterns (PC) obtained for each design are presented for each considered flow.

Table 3.2 Considered ATPG flows results.

Designs	**ATPG-FLW$_1$**		**ATPG-FLW$_2$**		**ATPG-FLW$_3$**		**ATPG-FLW$_4$**		**ATPG-FLW$_5$**		**ATPG-FLW$_6$**		**ATPG-FLW$_7$**		**ATPG-FLW$_8$**	
	FC-1 [%]	**PC-1**	**FC-2 [%]**	**PC-2**	**FC-3 [%]**	**PC-3**	**FC-4 [%]**	**PC-4**	**FC-5 [%]**	**PC-5**	**FC-6 [%]**	**PC-6**	**FC-7 [%]**	**PC-7**	**FC-8 [%]**	**PC-8**
B01	98.69	24	98.69	26	98.69	21	98.69	45	75.12	18	67.46	22	67.46	21	67.46	36
B02	97.27	14	97.95	15	97.95	15	97.95	26	88.24	11	77.65	13	77.65	15	77.65	23
B03	98.82	38	98.82	49	98.82	46	98.82	74	93.46	37	59.44	70	59.44	65	59.44	96
B04	98.52	60	98.52	79	98.52	77	98.52	112	76.01	54	69.91	70	69.96	68	69.91	99
B05	72.03	87	71.71	134	71.52	165	71.71	192	66.95	104	57.24	147	57.97	163	57.24	222
B06	98.03	17	98.38	26	98.38	23	98.38	40	73.84	15	61.76	21	61.76	18	61.76	33
B07	97.74	77	97.74	114	97.74	111	97.74	163	91.55	102	73.80	145	73.87	158	73.80	200
B08	99.03	57	99.03	82	99.03	77	99.03	127	88.01	67	71.84	92	71.84	98	71.84	135
B09	99.03	50	99.03	63	99.03	52	99.03	89	92.65	53	74.00	94	74.04	90	74.00	125
B10	98.75	59	98.75	73	98.75	67	98.75	116	74.57	52	64.97	72	64.73	76	64.97	101
B11	97.47	106	97.47	169	97.47	213	97.47	232	89.62	123	74.89	202	74.89	272	74.87	278
B12	99.14	165	99.14	225	99.14	207	99.14	311	89.81	334	77.90	516	78.10	546	77.89	716
B13	97.20	56	97.20	87	97.20	83	97.20	119	80.10	78	64.74	111	64.72	103	64.74	150
B14	95.95	698	96.09	792	96.09	795	96.09	1,207	95.91	913	87.63	1,553	87.72	1,576	87.62	2,077
B15	97.47	558	97.41	1,668	97.39	1,940	97.41	1,977	92.17	709	83.99	1,601	84.18	1,550	83.98	2,028

3.5.1 Results

By comparing all the methodologies presented and dividing them by type (SAF/static-CAT and TDF/dynamic-CAT) from the results obtained it can be seen that:

- ATPG-FLW$_1$ and ATPG-FLW$_5$ produce on average up to 50% less patterns than the other approaches and thus resulting in a reduced test application time; they would be the best choice for CAT implementation in the test development flow.
- ATPG-FLW$_4$ and ATPG-FLW$_8$ yield the worst results, due to the higher number of patterns and a lower coverage value regarding the ATPG TDF/dynamic-CAT (ATPG-FLW$_8$).

In particular, from ATPG-FLW$_1$ to ATPG-FLW$_4$ similar fault coverage values were obtained. Especially for B14 and B15, by comparing the number of patterns among the various flows, there is a considerable reduction in pattern count compared to the other flows. For ATPG-FLW$_4$ to ATPG-FLW$_8$, only the first flow gives a higher fault coverage than the others, with a lower number of patterns as well. Overall, it can be observed that ATPG-FLW$_1$ and ATPG-FLW$_5$ produce the best results. At the same time, they have up to 50% more pattern generation run-time than the other flows and up to 30% more patterns to detecting CA faults, unlike the other flows where they have up to 50% more patterns to detect CA faults. Experiments were done using the tools CMGen (for cell-aware fault model generation) and TestMAX™ ATPG (for pattern generation and fault simulation) by Synopsys.

3.5.2 Conclusions of the first contribution

About the ATPG flow presented in the previous subsections, a comparative overview of different test flows for CAT implementation was conducted resorting to a set of open-source benchmarks. The advantages and disadvantages related to the use of CAT methodology considering different test flows were shown. The results obtained showed that the flows ATPG-FLW$_1$ and ATPG-FLW$_5$ (i.e., incremental CA-ATPG starting from a set of patterns for traditional fault models) are the most suitable in terms of achievable fault coverage and number of total patterns.

Future work will involve the adoption of such approaches within more complex designs. It is important to notice that these flows using CAT approach can be used for other types of fault model as they has been introduced in the previous sections.

3.6 Experimental results about the comparison of different fault models

The purpose of the work includes a more effective test pattern generation in terms of achieved defect coverage and pattern count. In this section the results of the second contribution about the comparison of each fault model with different fault lists in terms of fault coverage and pattern count are reported and discussed. In particular, the details about the effectiveness of each fault model to target faults coming from different fault lists will be showed. Simulation and test generation time are considered negligible due to the fact that each flow requires no more than 10 minutes to be processed on a server with 16 CPU cores Intel® Xeon e-2000 series and 40*Gb* of RAM.

3.6.1 Results

In Table 3.3 the data regarding the patterns generated targeting the path delay fault model when simulated with respect to the different fault lists are shown both for the ITC'99 benchmark and the RISC-V designs. We denoted with FC^{i}_{j} the fault coverage figure obtained with the test patterns generated by targeting the fault model "i" with respect to the fault list for fault model "j". For example, FC^{TDF}_{PDF} is the fault coverage obtained by fault simulating the patterns generated targeting the TDFs with respect to the fault list of PDFs.

Starting from the left side of the table, in the first column all the considered designs are listed; the number of path delay faults are included in the second column. The other three columns represent the fault coverage figures achieved by the patterns targeting the 3 fault models: the first one relates to the patterns generated by the ATPG targeting the path delay fault model (FC^{PDF}_{PDF}); the second one is obtained by the fault simulation of the path delay fault list with the pattern set obtained by an

Table 3.3 Effects of patterns targeting path delay faults on different fault models.

Designs	**Faults**	**Fault Coverage [%]**			**Pattern Count**
	PDFs	**FC$^{PDF}_{PDF}$**	**FC$^{CAT}_{PDF}$**	**FC$^{TDF}_{PDF}$**	**PC^{PDF}**
B01	37	54.05	54.05	**59.46**	12
B02	27	**51.85**	40.74	**51.85**	7
B03	682	**53.08**	28.01	29.77	57
B04	932	**43.45**	27.04	22.42	60
B05	1,342	**18.85**	12.37	11.70	73
B06	104	**33.65**	33.65	33.65	9
B07	1,752	**15.47**	14.50	13.81	48
B08	345	**27.54**	18.26	23.48	27
B09	749	12.42	**18.42**	16.69	23
B10	468	21.58	20.94	**21.79**	20
B11	1,218	**7.06**	5.67	4.68	25
B12	2,954	**33.07**	20.07	22.51	292
B13	618	**39.48**	31.07	33.82	45
B14	10,064	**7.24**	3.34	3.44	232
B15	20,551	**3.15**	2.40	1.89	217
B17	7,895	**34.05**	17.18	16.81	405
B18	18,363	**26.60**	13.17	13.17	424
B19	30,000	**31.90**	15.01	13.94	536
B20	2,571	**9.37**	1.71	1.52	111
B21	2,571	**7.90**	1.94	2.14	86
B22	3,857	**8.06**	2.46	2.00	145
RISC-V	5,054	**12.96**	0	0	129

ATPG flow targeting the dynamic CAT faults (FC$^{CAT}_{PDF}$); the third one is obtained by the fault simulation of the path delay fault list with the transition delay pattern set (FC$^{TDF}_{PDF}$). The last column shows the pattern count (PC) for the considered fault model which, for the sake of showing all the data listed, is included in each table of this work and in order: PC^{PDF} is the number of patterns that refer to FC$^{PDF}_{PDF}$; PC^{CAT} refers to FC$^{CAT}_{CAT}$; PC^{TDF} refers to FC$^{TDF}_{TDF}$.

It can be noted that FC$^{PDF}_{PDF}$ has the overall best results in terms of coverage and pattern count, except for B01, B09 and B10 design, due to the fact that the ATPG tool, after the pattern generation, classified many faults as undetectable and ATPG untestable, hence decreasing the fault coverage of the design; whereas FC$^{CAT}_{PDF}$ has the best results in B09 but using more patterns with respect to the pattern sets for the other fault models (analogous situation for B06 but with the same coverage for all the fault models but with more patterns used). FC$^{TDF}_{PDF}$ has the best results in B01 and B10 where the reason may be inferred from the number of patterns Involved in

the fault simulation. The data regarding RISC-V require particular attention. In this case only the ATPG flow FC_{PDF}^{PDF} has the best performances, able to detect a very low set of the considered path delay faults in the design. With the other considered fault models used for the comparison with the path delay fault list, even though the pattern count is much higher than the PC^{PDF}, the results are 0% for both considered fault coverages, not allowing any path delay fault detection. From the analysis done for this work, only the specific path delay ATPG flow can detect some paths, which still are very few because most of them are classified either undetectable or ATPG untestable by the ATPG tool.

Table 3.4 shows the results obtained by using the dynamic CAT fault list. As shown in the previous table, the designs and the number of faults are included in the first and second column from the left. The fault coverages (the other three columns) show the data collected by the ATPG and the fault simulator tool, according with the methodology explained in the previous section. In the last column, the number of patterns for the respective fault model is shown. It can be noticed that FC_{CAT}^{PDF} has the worst overall fault coverage excluding B01, B03 and B04 that do not have a large number of gates and they may be considered negligible with respect to the other circuits.

In those cases, the PDFs pattern set has been capable of detecting most of the dynamic CAT faults of the design with a comparable number of patterns implemented in the fault simulator. FC_{CAT}^{TDF} has the best performance in B02, B08, B11, B13 and B17, but also in these cases the increased number of patterns used (especially for the bigger designs) affects the test time. The data regarding RISC-V show a slightly different situation: FC_{CAT}^{PDF} has a very low fault coverage because of the number of patterns (PC^{PDF}) that are capable to cover most of the faults in the design. The other fault models have comparable performances. Table 3.5 shows the data gathered from the transition delay fault model. The design list and the transition delay faults (TDFs) are included in the first and second column. FC_{TDF}^{TDF} is the fault coverage obtained by the TDF-oriented ATPG flow; FC_{TDF}^{PDF} and FC_{TDF}^{CAT} are the fault coverage values obtained by the respective fault simulations and the pattern count is shown in the last column of the table. Also in this case, FC_{TDF}^{PDF} has the worst performance in terms of fault coverage, considering the difference between the pattern count of each considered fault model, excluding B01 where the number of patterns used is

Table 3.4 Effects of patterns targeting dynamic CAT faults on different fault models.

Designs	**Faults**	**Fault Coverage [%]**			**Pattern Count**
	Dyn-CATs	**FC$_{CAT}^{CAT}$**	**FC$_{CAT}^{PDF}$**	**FC$_{CAT}^{TDF}$**	**PC^{CAT}**
B01	397	79.85	**90.68**	76.32	20
B02	216	92.13	87.04	**93.06**	13
B03	1,721	94.13	**94.60**	94.36	46
B04	4,448	80.64	**89.95**	77.90	56
B05	4,986	**69.53**	57.10	68.75	109
B06	392	**81.38**	79.34	76.02	15
B07	4,158	**94.13**	83.31	91.75	105
B08	1,528	90.31	72.05	**90.90**	62
B09	1,838	**93.96**	90.86	92.38	58
B10	1,350	**79.41**	75.04	75.19	49
B11	4,653	91.51	71.33	**91.66**	116
B12	9,844	**90.61**	86.64	90.09	347
B13	2,854	80.87	76.94	**83.50**	60
B14	87,402	**96.39**	54.91	94.17	853
B15	72,757	**94.64**	50.24	94.14	747
B17	222,566	91.88	60.85	**92.45**	1,108
B18	735,970	**83.34**	55.71	83.15	1,371
B19	1,486,346	**82.31**	55.80	82.15	1,530
B20	216,062	**97.51**	28.70	94.31	1,565
B21	214,248	**97.54**	19.94	94.18	1,569
B22	324,465	**97.64**	39.74	94.87	1,677
RISC-V	376,720	**96.11**	17.28	94.99	2,282

higher than the transition delay fault model. FC$_{TDF}^{TDF}$ has the best performance of all the designs.

The data regarding RISC-V shows that: FC$_{TDF}^{PDF}$ has low fault coverage because of the low number in PC^{PDF} as already shown and it is not capable to cover most of the faults in the design. Similar conditions hold true for FC$_{TDF}^{CAT}$, where most of the faults of the design were classified as "not detected" by the fault simulator. Overall, considering the benchmark designs and the data from the previous tables, it can be noticed that:

- The patterns generated targeting the path delay fault model, compared with the other ones, have the best performances when fault simulated on their own fault lists, excluding some cases for the small designs.
- The patterns generated targeting the dynamic CAT fault model have the best performance when a CAT-ATPG is performed, and a fault simulation is run

Table 3.5 Effects of patterns targeting transition delay faults on different fault models.

Designs	Faults	Fault Coverage [%]			Pattern Count
	TDFs	FC_{TDF}^{TDF}	FC_{TDF}^{PDF}	FC_{TDF}^{CAT}	PC^{TDF}
B01	230	79.85	**80.00**	70.43	16
B02	124	**92.13**	77.42	77.42	11
B03	678	**94.13**	87.61	86.73	32
B04	1,656	**80.64**	68.06	60.00	30
B05	2,194	**69.53**	47.93	58.85	78
B06	254	**81.38**	67.32	64.17	12
B07	1,480	**94.13**	65.99	81.23	62
B08	716	**90.31**	57.40	74.44	54
B09	720	**93.96**	82.92	82.50	33
B10	734	**79.41**	62.94	66.89	32
B11	1,544	**91.51**	53.70	76.43	96
B12	4,282	**90.61**	76.09	82.44	263
B13	1,392	**80.87**	63.51	66.16	61
B14	26,004	**96.39**	44.17	88.68	722
B15	24,802	**94.64**	34.26	80.56	589
B17	81,656	**86.05**	41.08	80.55	979
B18	237,320	**79.20**	38.47	76.03	1,121
B19	477,616	**78.89**	39.67	75.44	1,164
B20	55,556	**93.33**	21.40	91.93	986
B21	55,772	**93.99**	12.42	92.62	1,022
B22	83,696	**93.61**	30.17	92.51	1,080
RISC-V	195,812	**95.70**	13.93	14.36	2,205

(FC_{CAT}^{TDF}); the path delay pattern set has the worst performance in terms of fault coverage.

- The patterns generated targeting the transition delay fault model used with the TDF ATPG tool has the best performances in terms of coverage, followed by the fault simulation for the other ones.

Considering the overall data for RISC-V design, from the previous tables, it can be observed that:

- The patterns generated targeting the path delay fault model have the best performance when a PDF ATPG run is performed (FC_{PDF}^{PDF}); the same patterns cannot detect any fault considering the other models.
- Considering the patterns generated targeting the dynamic CAT fault model, they have the best performance when the respective fault list is used in ATPG

(FC_{CAT}^{CAT}); whereas, from the comparison with the other fault models, FC_{CAT}^{PDF} has the worst performance, while FC_{CAT}^{TDF} have similar results.

- The patterns generated targeting the transition delay fault model have the best performance only resorting to an TDF ATPG run (FC_{TDF}^{TDF}), while for the other fault models the achieved fault coverages figures are not comparable with the first one.

3.6.2 Conclusions of the second contribution

To sum up the work we have just presented, different fault models and designs have been used to generate different sets of defect-oriented test patterns. We then evaluated the fault coverages (FCs) obtained by running the fault simulation of the generated patterns with different fault lists stemming from different fault models. The obtained results showed that the behavior of the different circuits are rather different. Clearly, the FC obtained using the ATPG tool targeting a specific fault model is generally higher than the one obtained by using patterns generated for other fault models. However, there are cases where the patterns generated for a given fault model achieve very good fault coverage figures even when simulated with different fault lists. In some cases, as shown for CAT faults in table 3.4, it can be observed that pattern sets generated targeting one fault model (TDF) may be useful to detect other faults as well, possibly increasing the achieved fault coverage. This observation may pave the way to develop optimized test generation strategies, able to reduce the test generation time and (most importantly) to reduce the number of required test patterns while still achieving the same fault coverage.

3.7 Conclusions of the work

To sum up the work discussed in this chapter, different ATPG flows approaching cell-aware testing have been developed to evaluate the effectiveness of targeting different types of faults using mixed fault models. As a result, it has been shown that certain flows are more performing than the other ones in terms of pattern count and fault coverage. In the second introduced contribution, the above mentioned pattern sets, enriched with the added delay fault models (i.e., path-delay fault model), have

been used in a smart way to be fault simulated with other fault lists coming from different fault models. As a result, a mixed approach has been used to make the methodology suitable to target more faults than commonly detectable with common fault models.

Chapter 4

Conclusions

To conclude the PhD work that it has been carried out, it can be said that the main contribution in this thesis was the improvement of testing techniques for digital circuits manufactured by microelectronics companies to guarantee a lower escape rate using the most modern techniques and approaches that companies usually use.

In particular, this PhD thesis has presented three two topics and studies:

- the first one includes a comprehensive study on SRAM cells with particular emphasis on *resistive defects* that can arise anytime during the manufacturing process.
  More specifically, different types of defects have been discussed, in particular focusing on the effects produced by the resistive defects inside the SRAM cell; each defect has been analyzed in terms of the possible effects inside the SRAM cell depending on its resistance value and its impact in terms of power consumption, and provided guidelines for selecting the best test method.

- due to the fact that ranges of defects can be modeled through electrical analysis on the device and its cells, to understand the functioning of certain defects and to detect them, a low-level electrical analysis is required. Having performed this operation on the first topic of this work on memory devices, the same mechanism has been applied on other devices with the CAT. therefore, the second topic includes an evaluation of cell-aware testing, followed by an evaluation of ATPG flows to target together both common and cell-aware fault

models and obtain the best trade-off between fault coverage and testing time; more in detail, several flows have been considered and evaluated to identify the best flow to detect faults according to the most common fault models and the cell-aware fault model.
Furthermore, we included the usage of the obtained pattern sets from the previous study including those derived from delay fault models in order to compare and fault simulate the effectiveness in terms of fault coverage with fault lists belonging to different fault models.
In particular, an investigation about the application of the generated test patterns with the transition and path delay fault models in comparison with others developed with the cell-aware approach has been performed in terms of fault coverage, pattern count and test generation time.

In chapter 2 of this thesis, the research has focused on the effect of particular resistive defects that may produce a fault in the device, especially when the SRAM switches stand-by low-power state to the normal operating state and vice-versa. The contribution of this main research topic is the evaluation of the effect of each resistive defect in the low-power SRAM cell with back-bias circuitry. Three main resistive defects have been evaluated, including a discussion subsection where the symmetry of the cell has been used to identify symmetric resistive defects that present the same effects of the other ones. The most suitable fault model has been identified for each of the analyzed defects (depending of the resistance value) and then, the corresponding test aiming at identifying and detecting the fault has been evaluated. The research study has identified the key issues about what kind of effect is caused by changing the resistance value of the analyzed defects until the cut-off values produce a different behavior and then implementing the right test flow. This has been useful to ensure a more reliable and efficient testing of low-power SRAMs.

In chapter 3 of this thesis two main contributions have been shown. More specifically, the first contribution has included a testing methodology and ATPG flows have been presented to combine different fault models and increase the fault detection figures coming from different fault lists. As a result, the CAT approach has been shown to be highly effective in detecting as much as possible faults of the given fault list from a certain fault model. The research has also highlighted the importance of cell-aware testing in identifying defects that are not easily detectable

using traditional testing methods. By resorting to the research study, it may be possibly demonstrated that cell-aware testing can significantly improve the accuracy and reliability of digital testing.

In the last contribution of chapter 3 different fault models and designs have been used to generate different sets of defect-oriented test patterns. The obtained fault coverages derived from the fault simulation process of the generated patterns with different fault lists belonging to different fault models have been evaluated. The results have shown the different behavior of the analyzed circuits and the obtained FC using the ATPG tool targeting a specific fault model is generally higher than the one obtained by using patterns generated for other fault models. The study has also demonstrated that there are cases where the patterns generated for a given fault model achieve very good fault coverage when fault simulated with different fault lists. As a result, the presented observations may pave the way to develop optimized test generation strategies, able to reduce the test generation time and (most importantly) to reduce the number of required test patterns while still achieving the same fault coverage.

Overall, the results of this research have important implications for the design and testing. The introduced approaches can be used to develop more effective testing strategies that can help to improve the reliability and performance of digital circuits in a wide range of applications. In particular, defining the right fault model associated to the effect of a SRAM defect it can be found the right test to detect the associated defect. At the same time, not only common fault models may cover all the possible defects in a circuit, hence as a result of the remaining works, defining the right ATPG flow and mixing together in the same time the detection of more fault models with the same process, advantages can be found. For this reason, to increase the quality of the sold devices it is important to enhance the quality of the test to defect-oriented testing to spot defects that may be undetectable by common test solutions. To implement better test approaches and achieve better performances from the testing process, avoiding high value of defect rate hence decreasing test escapes.

Future works may include the implementation and validation of the introduced techniques on actual devices in order to augment the testing processes and the reliability of the microelectronics circuits manufactured by the company.

References

[1] A. Bosio, L. Dilillo, P. Girard, S. Pravossoudovitch, and Virazel A. *Advanced Test Methods for SRAMs - Effective Solutions for Dynamic Fault Detection in Nanoscaled Technologies*. Springer New York, NY, 01 2009.

[2] P. Bernardi, R. Cantoro, A. Coyette, W. Dobbeleare, M. Fieback, A. Floridia, G. Gielen, J. Gomez, M. Grosso, A. M. Guerriero, I. Guglielminetti, S. Hamdioui, G. Insinga, N. Mautone, N. Mirabella, S. Sartoni, M. Sonza Reorda, R. Ullmann, R. Vanhooren, N. Xama, and L. Wu. Recent trends and perspectives on defect-oriented testing. In *2022 IEEE 28th International Symposium on On-Line Testing and Robust System Design (IOLTS)*, pages 1–10, 2022.

[3] M. Fieback, L. Wu, G.C. Medeiros, H. Aziza, S. Rao, E.J. Marinissen, M. Taouil, and S. Hamdioui. Device-aware test: A new test approach towards dppb level. In *2019 IEEE International Test Conference (ITC)*, pages 1–10, 2019.

[4] F. Hapke, W. Redemund, A. Glowatz, J. Rajski, M. Reese, M. Hustava, M. Keim, J. Schloeffel, and A. Fast. Cell-aware test. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 33(9):1396–1409, 2014.

[5] Lin Chin Jen and S.M. Reddy. On delay fault testing in logic circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 6(5):694–703, 1987.

[6] M.H. Schulz, K. Fuchs, and F. Fink. Advanced automatic test pattern generation techniques for path delay faults. In *[1989] The Nineteenth International Symposium on Fault-Tolerant Computing. Digest of Papers*, pages 44–51, 1989.

[7] F. Hapke, W. Howell, P. Maxwell, E. Brazil, S. Venkataraman, R. Dutta, A. Glowatz, A. Fast, and J. Rajski. Defect-oriented test: Effectiveness in high volume manufacturing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 40(3):584–597, 2021.

[8] A. Avizienis, J.C. Laprie, B. Randell, and C. Landwehr. Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 1(1):11–33, 2004.

[9] C.U. Ngene. The essentials of testing digital circuits. https://cyberleninka.ru/article/n/the-essentials-of-testing-digital-circuits, 2012. Radioelectronics and Informatics.

[10] International technology roadmap of semiconductors. available online. https://www.semiconductors.org/resources/2015-international-technology-roadmap-for-semiconductors-itrs/, 2015.

[11] N. Mirabella, M. Ricci, and M. Grosso. On the test of single via related defects in digital vlsi designs. In *23rd International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS)*, pages 1–6, 2020.

[12] N. Mirabella, M. Ricci, I. Calà, R. Lanza, and M. Grosso. Testing single via related defects in digital vlsi designs. *Microelectronics Reliability*, 120:114100, 2021.

[13] Ho C. H., Kim S. Y., Panagopoulos G. D., and Roy K. Statistical tddb degradation in memory circuits: Bit-cells to arrays. *IEEE Transactions on Electron Devices*, 63(6):2384–2390, 2016.

[14] Suk and Reddy. A march test for functional faults in semiconductor random access memories. *IEEE Transactions on Computers*, C-30(12):982–985, 1981.

[15] P.K. Bikki and K. Pi. Sram cell leakage control techniques for ultra low power application: A survey. *SCIRP Circuits Syst.*, 8:114100, 2017.

[16] N. Mirabella, M. Grosso, G. Franchino, S. Rinaudo, I. Deretzis, A. La Magna, and M. Sonza Reorda. Comparing different solutions for testing resistive defects in low-power srams. In *2021 IEEE 22nd Latin American Test Symposium (LATS)*, pages 1–6, 2021.

[17] N. Mirabella, M. Grosso, G. Franchino, S. Rinaudo, I. Deretzis, A. La Magna, and M. Sonza Reorda. An experimental evaluation of resistive defects and different testing solutions in low-power back-biased sram cells. In *MDPI Electronics*, volume 11, 2022.

[18] A.J. Van de Goor. Testing semiconductor memories: Theory and practice. *COMTEX*, 1998.

[19] S. Hamdioui and A.J. Van De Goor. An experimental analysis of spot defects in srams: realistic fault models and tests. In *Proceedings of the Ninth Asian Test Symposium*, pages 131–138, 2000.

[20] M.J. Geuzebroek, J.T. van der Linden, and A.J. van de Goor. Test point insertion for compact test sets. In *Proceedings International Test Conference 2000 (IEEE Cat. No.00CH37159)*, pages 292–301, 2000.

[21] Ad.J. van de Goor and Z. Al-Ars. Functional memory faults: a formal notation and a taxonomy. In *Proceedings 18th IEEE VLSI Test Symposium*, pages 281–289, 2000.

[22] M.M. Pelicia, R.P. Coimbra, and P.B. Zanetta. A back biasing voltage generator for 28nm utbb-fdsoi rvt cmos digital circuits. In *2018 International Conference on IC Design & Technology (ICICDT)*, pages 1–4, 2018.

[23] B. C. Paz, M. Cassé, S. Barraud, G. Reimbold, M. Vinet, O. Faynot, and M. A. Pavanello. Back bias impact on effective mobility of p-type nanowire soi mosfets. In *2018 33rd Symposium on Microelectronics Technology and Devices (SBMicro)*, pages 1–4, 2018.

[24] L. Dilillo, P. Girard, S. Pravossoudovitch, A. Virazel, S. Borri, and M. Hage-Hassan. Dynamic read destructive fault in embedded-srams: analysis and march test solution. In *Proceedings. Ninth IEEE European Test Symposium, 2004. ETS 2004.*, pages 140–145, 2004.

[25] A. Ney, P. Girard, C. Landrault, S. Pravossoudovitch, A. Virazel, and M. Bastian. Un-restored destructive write faults due to resistive-open defects in the write driver of srams. In *25th IEEE VLSI Test Symposium (VTS'07)*, pages 361–368, 2007.

[26] A. Ney, P. Girard, C. Landrault, S. Pravossoudovitch, A. Virazel, and M. Bastian. Slow write driver faults in 65nm sram technology: Analysis and march test solution. In *2007 Design, Automation & Test in Europe Conference & Exhibition*, pages 1–6, 2007.

[27] A. Ney, P. Girard, C. Landrault, S. Pravossoudovitch, A. Virazel, and M. Bastian. Dynamic two-cell incorrect read fault due to resistive-open defects in the sense amplifiers of srams. In *12th IEEE European Test Symposium (ETS'07)*, pages 97–104, 2007.

[28] F. Renan Alves, L. Dilillo, A. Bosio, P. Girard, S. Pravossoudovitch, A. Virazel, and N. Badereddine. Impact of Resistive-Bridging Defects in SRAM at Different Technology Nodes. *Journal of Electronic Testing: Theory and Applications*, 28(3):317–329, June 2012.

[29] L. Dilillo, P. Girard, S. Pravossoudovitch, A. Virazel, S. Borri, and M. Hage-Hassan. Resistive-open defects in embedded sram core cells: Analysis and march test solution. *13th Asian Test Symposium*, 8:pp.266–271., 2004.

[30] E.I. Vatajelu, A. Bosio, L. Dilillo, P. Girard, A. Todri, A. Virazel, and N. Badereddine. Analyzing resistive-open defects in sram core-cell under the effect of process variability. *18th IEEE European Test Symposium*, 2013.

[31] L. Dilillo, P. Girard, S. Pravossoudovitchi, and A. Virazel. Resistive-open defect injection in sram core-cell: Analysis and comparison between 0.13 um and 90nm technologies. *42nd IEEE Design Automation Conference*, 2005.

[32] L. Dilillo, P. Girard, S. Pravossoudovitchi, and A. Virazel. Data retention fault in sram memories: Analysis and detection procedures. *23rd IEEE VLSI Test Symposium*, 2005.

[33] L. Dilillo, P. Girard, S. Pravossoudovitchi, A. Virazel, S. Borri, and Hage-Hassan M. Efficient march test procedure for dynamic read destructive faults detection in sram memories. *Electron. Test. Theory Appl.*, 21:551–561, 2005.

[34] Adams R.D. and Cooley E.S. Analysis of a deceptive read destructive memory fault model and recommended testing. In *IEEE North Atlantic Test Workshop*, pages 361–368, 1996.

[35] S. Hamdioui, Z. Al-Ars, and A.J. van de Goor. Testing static and dynamic faults in random access memories. *20th IEEE VLSI Test Symposium*, 2002.

[36] A.J. Bhavnagarwala, X. Tang, and J.D. Meindl. The impact of intrinsic device fluctuations on cmos sram cell stability. *IEEE J. Solid-State Circuits*, 36, 2001.

[37] E. Seevinck, F.J. List, and J. Lohstroh. Static-noise margin analysis of mos sram cells. *IEEE J. Solid-State Circuits*, 22, 1987.

[38] V. Singh, S.K. Singh, and R. Kapoor. Static noise margin analysis of 6t sram. *IEEE International Conference for Innovation in Technology*, 2020.

[39] R. Kolhal and V. Agarwal. A power and static noise margin analysis of different sram cells at 180nm technology. *3rd IEEE International Conference on Electronics, Communication and Aerospace Technology*, 2019.

[40] S. Hamdioui, Z. Al-Ars, and Ad.J. van de Goor. Testing static and dynamic faults in random access memories. In *Proceedings 20th IEEE VLSI Test Symposium (VTS 2002)*, pages 395–400, 2002.

[41] R. Keerthi and C.H. Chen. Stability and static noise margin analysis of low-power sram. *IEEE Instrumentation and Measurement Technology Conference*, 2008.

[42] M. Bushnell and V.D. Agrawal. *Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits*. Springer, 2002.

[43] S. Borri, M. Hage-Hassan, P. Girard, S. Pravossoudovitch, and A. Virazel. Defect-oriented dynamic fault models for em-bedded-srams. *8th IEEE European Test Workshop*, 2003.

[44] A. Ney, L. Dilillo, P. Girard, S. Pravossoudovitch, A. Virazel, M. Bastian, and V. Gouin. A new design-for-test technique for sram core-cell stability faults. *IEEE Design, Automation & Test in Europe Conference & Exhibition*, 2009.

[45] L. Dilillo, P. Rosinger, P. Girard, and B.M. Al-Hashimi. Minimizing test power in sram through reduction of pre-charge activity. *IEEE Design Automation & Test in Europe Conference & Exhibition*, 2006.

[46] N. Mirabella, A. Floridia, R. Cantoro, M. Grosso, and M. Sonza Reorda. A comparative overview of atpg flows targeting traditional and cell-aware fault models. In *2022 29th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pages 1–4, 2022.

[47] N. Mirabella, A. Floridia, R. Cantoro, M. Grosso, and M. Sonza Reorda. Targeting different defect-oriented fault models in ic testing: an experimental approach. In *2023 26th Euromicro Conference on Digital System Design (DSD)*, pages 214–219, 2023.

[48] I. Polian, J. Anders, S. Becker, P. Bernardi, K. Chakrabarty, N. ElHamawy, M. Sauer, A. Singh, M. Sonza Reorda, and S. Wagner. Exploring the mysteries of system-level test. In *2020 IEEE 29th Asian Test Symposium (ATS)*, pages 1–6, 2020.

[49] D. Appello, H. H. Chen, M. Sauer, I. Polian, P. Bernardi, and M. Sonza Reorda. System-level test: State of the art and challenges. In *2021 IEEE 27th International Symposium on On-Line Testing and Robust System Design (IOLTS)*, pages 1–7, 2021.

[50] H.M. Eissa and A. ElRahman ElMously. Physical aware design methodology for analog & mixed signal integrated circuits. In *2009 4th International Design and Test Workshop (IDT)*, pages 1–5, 2009.

[51] Zhan Gao, Santosh Malagi, Erik Jan Marinissen, Joe Swenton, Jos Huisken, and Kees Goossens. Defect-location identification for cell-aware test. In *2019 IEEE Latin American Test Symposium (LATS)*, pages 1–6, 2019.

[52] Peter Maxwell, Friedlich Hapke, and Huaxing Tang. Cell-aware diagnosis: Defective inmates exposed in their cells. In *2016 21th IEEE European Test Symposium (ETS)*, pages 1–6, 2016.

[53] F. Hapke and J. Schloeffel. Introduction to the defect-oriented cell-aware test methodology for significant reduction of dppm rates. In *2012 17th IEEE European Test Symposium (ETS)*, pages 1–6, 2012.

[54] K.S. Das and A. Zala. Optimizing cell-aware atpg pattern volume to keep test cost competitive. In *2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, pages 1–6, 2020.

[55] P. Varma. On test generation for path delay faults in asics. In *Digest of Papers. 1992 IEEE VLSI Test Symposium*, pages 19–24, 1992.

[56] L. Gordon Smith. Model for delay faults based upon paths. In *International Test Conference*, 1985.

[57] M.H. Schulz, K. Fuchs, and F. Fink. Advanced automatic test pattern generation techniques for path delay faults. In *[1989] The Nineteenth International Symposium on Fault-Tolerant Computing. Digest of Papers*, pages 44–51, 1989.

[58] Barzilai Z. and Rosen B. Comparison of ac self-testing procedures. *International Test Conference*, 1983.

[59] J. Sudbrock, J. Raik, R. Ubar, W. Kuzmicz, and W. Pleskacz. Defect-oriented test- and layout-generation for standard-cell asic designs. In *8th Euromicro Conference on Digital System Design (DSD'05)*, pages 79–82, 2005.

[60] Dong-Zhen Lee, Ying-Yen Chen, Kai-Chiang Wu, and M.C.T. Chao. Improving cell-aware test for intra-cell short defects. In *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 436–441, 2022.

[61] F. Hapke, R. Krenz-Baath, A. Glowatz, J. Schloeffel, H. Hashempour, S. Eichenberger, C. Hora, and D. Adolfsson. Defect-oriented cell-aware atpg and fault simulation for industrial cell libraries and designs. In *2009 International Test Conference*, pages 1–10, 2009.

[62] Po-Yao Chuang, Cheng-Wen Wu, and Harry H. Chen. Cell-aware test generation time reduction by using switch-level atpg. In *2017 International Test Conference in Asia (ITC-Asia)*, pages 27–32, 2017.

[63] Harry H. Chen, Simon Y-H. Chen, Po-Yao Chuang, and Cheng-Wen Wu. Efficient cell-aware fault modeling by switch-level test generation. In *2016 IEEE 25th Asian Test Symposium (ATS)*, pages 197–202, 2016.

[64] R.C. Aitken. Finding defects with fault models. In *Proceedings of 1995 IEEE International Test Conference (ITC)*, pages 498–505, 1995.

[65] K.C.Y. Mei. Bridging and stuck-at faults. *IEEE Transactions on Computers*, C-23(7):720–727, 1974.

[66] F.J. Ferguson and T. Larrabee. Test pattern generation for realistic bridge faults in cmos ics. In *1991, Proceedings. International Test Conference*, pages 492–, 1991.

[67] P. Engelke, I. Polian, J. Schloeffel, and B. Becker. Resistive bridging fault simulation of industrial circuits. In *2008 Design, Automation and Test in Europe*, pages 628–633, 2008.

[68] J. Rearick and J.H. Patel. Fast and accurate cmos bridging fault simulation. In *Proceedings of IEEE International Test Conference - (ITC)*, pages 54–62, 1993.

[69] J. A. Waicukauski, E. Lindbloom, B.K. Rosen, and V.S. Iyengar. Transition fault simulation. *IEEE Design & Test of Computers*, 4(2):32–38, 1987.

[70] H. Cox and J. Rajski. Stuck-open and transition fault testing in cmos complex gates. In *International Test Conference 1988 Proceeding at mNew Frontiers in Testing*, pages 688–694, 1988.

[71] X. Lin, K. Tsai, C. Wang, M. Kassab, J. Rajski, T. Kobayashi, R. Klingenberg, Y. Sato, S. Hamada, and T. Aikyo. Timing-aware atpg for high quality at-speed testing of small delay defects. In *2006 15th Asian Test Symposium*, pages 139–146, 2006.

[72] S.K. Goel, K. Chakrabarty, M. Yilmaz, K. Peng, and M. Tehranipoor. Circuit topology-based test pattern generation for small-delay defects. In *2010 19th IEEE Asian Test Symposium*, pages 307–312, 2010.

[73] I. Pomeranz and S.M. Reddy. On n-detection test sets and variable n-detection test sets for transition faults. In *Proceedings 17th IEEE VLSI Test Symposium (Cat. No.PR00146)*, pages 173–180, 1999.

[74] C. Kyoung Youn, S. Mitra, and E.J. McCluskey. Gate exhaustive testing. In *IEEE International Conference on Test, 2005.*, pages 7 pp.–777, 2005.

[75] Zhan Gao, Santosh Malagi, Min-Chun Hu, Joe Swenton, Rogier Baert, Jos Huisken, Bilal Chehab, Kees Goossen, and Erik Jan Marinissen. Application of cell-aware test on an advanced 3nm cmos technology library. In *2019 IEEE International Test Conference (ITC)*, pages 1–6, 2019.

[76] P. d'Hondt, A. Ladhar, P. Girard, and A. Virazel. A learning-based methodology for accelerating cell-aware model generation. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1580–1585, 2021.

[77] Francesco Lorenzelli, Zhan Gao, Joe Swenton, Santosh Malagi, and Erik Jan Marinissen. Speeding up cell-aware library characterization by preceding simulation with structural analysis. In *2021 IEEE European Test Symposium (ETS)*, pages 1–6, 2021.

[78] P.K. Lala. *Digital Circuit Testing and Testability*. The Morgan Kaufmann Series in Computer Architecture and Design Series. Academic Press, 1997.

[79] F. Corno, M. Sonza Reorda, and G. Squillero. Rt-level itc'99 benchmarks and first atpg results. *IEEE Design & Test of Computers*, 17(3):44–53, 2000.

[80] M. Gautschi, P.D. Schiavone, A. Traber, I. Loi, A. Pullini, D. Rossi, E. Flamand, F.K. Gürkaynak, and L. Benini. Near-threshold risc-v core with dsp extensions for scalable iot endpoint devices. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 25(10):2700–2713, 2017.

[81] W. Qiu and D.M.H. Walker. Testing the path delay faults of iscas85 circuit c6288. In *Proceedings. 4th International Workshop on Microprocessor Test and Verification - Common Challenges and Solutions*, pages 19–24, 2003.

[82] I. Pomeranz. Path unselection for path delay fault test generation. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 31(2):267–275, 2023.

Glossary

Automatic Test Pattern Generation

It is a method within test automation that focuses on generating input sequences, or test patterns. These patterns are essential for enabling automatic test equipment to differentiate between proper and defective behavior in digital circuits.

Cell-aware test

An advanced test methodology that targets potential manufacturing defects within the cells of an integrated circuit, enhancing the detection of cell-internal faults that traditional testing methods might miss, thereby improving the overall quality and reliability of semiconductor devices.

Delay fault model

It is a conceptual framework used in testing and verifying digital circuits, which assumes that faults within the circuit cause deviations in signal propagation times, potentially leading to incorrect outputs when the circuit does not meet its timing specifications.

Dynamic random-access memories

A common type of volatile memory that stores each bit of data in a separate capacitor within an integrated circuit, requiring periodic refreshing of the stored data to maintain integrity, and is widely used in computing devices due to its simplicity and cost-effectiveness compared to other types of random-access memory.

Fault

A Fault is a defect affecting a circuit or system.

Fault coverage

It is a metric that quantifies the percentage of detectable faults out of the total possible faults in a circuit, providing an indication of the effectiveness of the test procedures in identifying potential defects.

Path delay fault

Refers to a specific type of defect in digital circuits where a signal takes longer than expected to propagate through a combinational path, potentially leading to timing errors and to the failure of the circuit to operate correctly within its specified timing constraints.

Read-only memories

A type of non-volatile memory that is pre-programmed with data and cannot be modified, used to store firmware or software in computers and other electronic devices, ensuring that essential instructions or data are preserved even when the device is powered off.

Read-write memories

A type of computer memory that allows data to be written and read multiple times, enabling both the storage and retrieval of information as needed by the system, which is essential for tasks such as running applications and processing data in real-time.

Static noise margin

A metric used to quantify the stability of a memory cell, specifically in SRAM, representing the maximum amount of noise voltage that a cell can withstand without causing a change in its stored value, thereby serving as an indicator of the robustness of the cell against noise and disturbances.

Static random-access memories

A type of volatile memories that uses bistable latching circuitry to store each bit, providing faster access than dynamic RAMs (DRAMs) and retaining data as long as power is supplied.

Stuck-at fault

It is a common type of digital circuit fault where a gate-level signal line is erroneously fixed at a logical high (1) or low (0) value, regardless of the input to the circuit, potentially leading to incorrect circuit behavior or failure.

System-Level Test

In the context of VLSI it refers to the process of testing an integrated circuit at the final stage of production, ensuring that all components and subsystems function correctly together within the complete system under real-world operating conditions.

Transition delay fault

It occurs when a change in the logical state of a digital signal (transition) is slower than specified, causing the signal to arrive at its destination later than intended, which can lead to incorrect circuit operation if the timing requirements are strict.

Appendix

Pubblications by Nunzio Mirabella:

- Mirabella N. and Ricci M. and Grosso M., "On the test of single via related defects in digital VLSI designs," 2020 23rd International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS), Novi Sad, Serbia, 2020, pp. 1-6,
  doi: 10.1109/DDECS50862.2020.9095717.

- Mirabella N. and Ricci M. and Calà I. and Lanza R. and Grosso M., "Testing single via related defects in digital VLSI designs," Microelectronics Reliability, Volume 120, 2021, 114100, ISSN 0026-2714,
  doi: 10.1016/j.microrel.2021.114100.

- Mirabella, N. and Grosso, M. and Franchino, G. and Rinaudo, S. and Deretzis, I. and La Magna, A. and Sonza Reorda, M., "Comparing different solutions for testing resistive defects in low-power SRAMs," 2021 IEEE 22nd Latin American Test Symposium (LATS), Punta del Este, Uruguay, 2021, pp. 1-6,
  doi:10.1109/LATS53581.2021.9651760.

- Mirabella, N. and Grosso, M. and Franchino, G. and Rinaudo, S. and Deretzis, I. and La Magna, A. and Sonza Reorda, M., "An Experimental Evaluation of Resistive Defects and Different Testing Solutions in Low-Power Back-Biased SRAM Cells," Electronics 2022, 11, 203.
  doi: 10.3390/electronics11020203

- Bernardi P. and Cantoro R. and Coyette A. and Dobbeleare W. and Fieback M. and Floridia A. and Gielen G. and Gomez J. and Grosso M. and Guerriero A. M. and Guglielminetti I. and Hamdioui S. and Insinga G. and Mautone

N. and Mirabella N. and Sartoni S. and Sonza Reorda M. and Ullmann R. and Vanhooren R. and Xama N. and Wu L., "Recent Trends and Perspectives on Defect-Oriented Testing," 2022 IEEE 28th International Symposium on On-Line Testing and Robust System Design (IOLTS), Torino, Italy, 2022, pp. 1-10,
doi: 10.1109/IOLTS56730.2022.9897647.

- Mirabella N. and Floridia A. and Cantoro R. and Grosso M. and Sonza Reorda M., "A comparative overview of ATPG flows targeting traditional and cell-aware fault models," 2022 29th IEEE International Conference on Electronics, Circuits and Systems (ICECS), Glasgow, United Kingdom, 2022, pp. 1-4,
doi: 10.1109/ICECS202256217.2022.9971003.

- Mirabella N. and Floridia A. and Cantoro R. and Grosso M. and Sonza Reorda M., "Targeting different defect-oriented fault models in IC testing: an experimental approach," 2023 26th Euromicro Conference on Digital System Design (DSD), Golem, Albania, 2023, pp. 214-219,
doi: 10.1109/DSD60849.2023.00039.