Improvement of Hardware Reliability with Aging Monitors

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

Vorgelegt von

Chang Liu

aus Beijing, China

Hauptberichter:	Prof. Dr. Hans-Joachim Wunderlich
Mitberichter:	Prof. Dr. Lorena Anghel

Tag der mündlichen Prüfung: 26. Oct. 2017

Institut für Technische Informatik
der Universität Stuttgart

2017

To my parents and Shiyi Xiao.

Contents

List of Figures

Chapter 5

Chapter 6

Chapter 7

List of Tables

Acknowledgments

I would like to express my sincere gratefulness to all the people who made this work possible.

I am truly thankful to Prof. Wunderlich, who gave me the golden opportunity to work at the Institut für Technische Informatik (ITI) and nurtured me with his invaluable patience, professional supervision and insightful ideas in countless discussions. Furthermore, I also appreciate Prof. Anghel, for the constructive feedback of my work on conferences and seminars and for accepting to be the second reviewer of my dissertation.

I would like to thank Michael Kochte for his unconditional supports and enlightening advice, which orientated me to the research life in the foreign country. He consistently shares his enthusiasm and experience and provides the helpful guidance and feedbacks that considerably contributed to this thesis and all of my research work. I am also grateful to Rafał Baranowski for his inspiring ideas and the honest suggestions.

I am more than happy to thank all the colleagues and students, who worked with me and made me such a fruitful and enjoyable ITI experience in the past years, especially: Claus Braun, Alejandro Cook, Atefe Dalirsani, Melanie Elm, Laura Rodríguez Gómez, Stefan Holst, Nadareh Hatami, Michael Imhof, Anusha Kakarala, Abdullah Mumtaz, Eric Schneider, Alexander Schöll, Dominik Ull, Marcus Wagner, Ahmed Atteya and Zahra Najafi Haghi.

Many thanks also go to our secretary Mirjam Breitling and staffs Helmut Häfner and Lothar Hellmeier for providing us the excellent administrative and technical assistance.

I can never thank my parents enough for their enduring love, continuous encouragement and wholeheartedly support. I am very grateful to Shiyi Xiao for his patience and faith in the success of my PhD. I would like to also thank my best friend Thomas Stein for making my life in Germany so colorful and unforgettable.

Stuttgart, 2017

Chang Liu

Abstract

The continued pace of technology scaling minimizes the transistor size and enlarges their density in VLSI circuits. Each new generation of devices is more susceptive to manufacturing imperfections, process variations, the operational environment, soft errors and aging degradation, which threaten the reliability of hardware components and systems. On the other hand, the extremely high demand for functional safety and system availability in critical applications increases the requirement of system reliability.

Existing techniques for reliability improvement are rather expensive because of the dedicated external test equipment and on-chip infrastructures, the area and power penalty, the increased design complexity, the extra test data volume and extended test time. More importantly, observing only degradation effects which affect the circuit behavior limits the monitoring timeliness of aging processes and the capacity to proactively perform aging countermeasures. Furthermore, a low measurement frequency restricts the prediction accuracy of imminent failures and in the worst case, may even lead to an unmonitored timing violation.

This thesis addresses the improvement of system reliability during the two most vulnerable periods of the circuit lifetime: the early life and wear-out failure periods. For aging monitoring, this work introduces a proactive monitor design to assess the deterioration stress online, which allows protecting the device before degradation takes place by aging mitigation. Another main contribution of the work is a novel monitor placement scheme that enables accurate and efficient failure predictions by frequent online delay measurement. Finally, the entire aging prediction framework is re-utilized for testing of small delay faults in the hardware, which may eventually degrade into early life failures.

Experiments on benchmark and industrial circuits confirm the reduced test latency and enhanced prediction precision by the proposed monitor design and placement. The relocated monitors show a significant reduction of measurement cost and prevent unmonitored timing failures. In addition, a high coverage of hard-to-detect small delay faults is achieved by reusing aging monitors for testing. The results demonstrate a high cost-efficiency for both aging monitoring as well as testing of small delay faults.

Zusammenfassung

Der anhaltende technologische Wandel der Skalierung führt zur Minimierung der Transistorgrößen und erhöht die Transistordichte in VLSI Schaltungen. Jede neue Generation von Geräten ist anfälliger für Unvollkommenheiten bei der Fertigung, den Prozessvariationen, die Betriebsumgebung, Soft Errors und Alterungsabbau, welche die Zuverlässigkeit von Hardwarekomponenten und Systemen bedrohen. Auf der anderen Seite erhöht die extrem hohe Nachfrage nach funktionaler Sicherheit und Systemverfügbarkeit in kritischen Anwendungen das Erfordernis der Systemzuverlässigkeit.

Bestehende Techniken zur Zuverlässigkeitsverbesserung sind ziemlich teuer wegen der dedizierten externen Testausrüstung und der On-Chip Infrastrukturen, der Flächen- und Stromstrafe, der erhöhten Designkomplexität, des zusätzlichen Testdatenvolumens und der verlängerten Testzeit. Überdies, werden die Prognosegenauigkeit der Alterungsprozesse und die Fähigkeit zur proaktiven Durchführung von Alterungsgegenmaßnahmen begrenzt, wenn ausschließlich die Verschlechterungseffekte, welche das Systemverhalten beeinflussen, beobachtet werden. Darüber hinaus beschränkt eine niedrige Messfrequenz die Vorhersagegenauigkeit von drohenden Ausfällen und kann im schlimmsten Fall sogar zu einer nicht überwachten Zeitverletzung führen.

Diese Arbeit befasst sich mit der Verbesserung der Systemzuverlässigkeit während der beiden am risikobehafteten Zeiträume der Schaltungslebensdauer: Es handelt sich dabei sowohl um Fehler, die zu frühen Lebenszeiten der Schaltung auftreten als auch um Verschleißfehlerperioden. Für die Alterungsüberwachung führt diese Arbeit ein proaktives Monitordesign ein, um den Zustand des Verschleißes online zu beurteilen, welches es ermöglicht, das Gerät vor dem alterungsbedingten Abbau durch Gegenmaßnahmen vor Alterung zu schützen. Ein weiterer Hauptbeitrag der Arbeit ist ein neuartiges Monitor-Platzierungsschema, das genaue und effiziente Fehlerprognosen durch häufige Online-Verzögerungsmessungen ermöglicht. Schließlich wird das gesamte Alterungsvorhersagesystem zum Testen von kleinen Verzögerungsfehlern in der Hardware wiederverwendet, womit schließlich zu frühe Ausfälle reduziert werden können.

Experimente auf Benchmark- und Industrieschaltungen bestätigen die reduzierte Testlatenz und die verbesserte Vorhersagegenauigkeit durch die vorgeschlagene Monitorgestaltung und -platzierung. Die verlagerten Monitore zeigen eine deutliche Redu-

zierung der Messkosten und verhindern unüberwachte Timing-Ausfälle. Auch eine hohe Abdeckung von schwer zu erkennenden kleinen Verzögerungsfehlern wird durch Wiederverwendung von Alterungsmonitoren zum Testen erreicht. Die Ergebnisse zeigen eine hohe Kosteneffizienz sowohl für die Alterungsüberwachung als auch für die Entdeckung kleiner Verzögerungsfehler.

CHAPTER 1

Introduction

The tremendous advancements in manufacturing technology enable the exponential growth in the complexity of microelectronic hardware, which has shown a close fit to Moor's Law [Moore68] for more than 40 years [ITR12]. The increasing complexity significantly improves the circuit performance, and shrinking geometries of the chips reduce the fabrication cost. As a result, the microelectronic hardware starts to touch and play an important role in nearly all aspects of lives: in the automotive, healthcare, and entertainment markets; in systems of education, research and government; in communications networks; in military activities, in space explorations or energy domains.

The continuous pace of chip miniaturization brings us into the new era of nanoscale technology. However, due to the small size and large density of transistors, the new generation of devices becomes more suspective to the manufacturing imperfections, process variations, operational environments, intermittent soft errors and aging degradation [Borka05], which reduce the hardware performance and may eventually deteriorate into a functional failure.

In a sophisticated system nowadays, the malfunctions or silent corruptions due to device errors may cause catastrophic consequences. Therefore the hardware reliability becomes a major concern of the manufacturers and primary requirement of the market. The rest of the chapter firstly provides the definition and measures of circuit *reliability*.

Then it discusses the device *failure rate* as an important influencing factor and shows how it varies at different periods over the hardware lifetime. The motivation and the principal methods for reliability improvement are illustrated in details. Finally, contributions of the work at hand are collectively listed and followed by a brief overview of the entire thesis.

1.1 Hardware Reliability

Reliability is the *ability* of a hardware component to *remain functional* (i.e., without any failure and within specified performance boundaries) under a particular operation condition. *Reliability* is quantified as the probability that the component operates without any interruption of malfunctions until a particular moment [Birol04].

Assume the *failure probability* ($F(t)$) is the probability that a hardware device fails before or at a particular time t. $F(t) = Prob\{T \leq t\}$, where T is a random variable and denotes the lifetime of a module i.e. the time until it fails. *Reliability* $R(t)$ i.e. the probability that the component survives longer than t can be formulated as:

$$R(t) = Prob\{T \geq t\} = 1 - F(t) \tag{1.1}$$

In a real system, the calculation of the system reliability is a tough task due to a large number of components and a diverse circuit structure. For simplification, *Mean Time To Failure* (MTTF) is an essential measure to represent the system reliability. MTTF is the expected value of the time to failure i.e. the average duration a system runs until a failure occurs [Shoom02], described as:

$$MTTF = E[T] = \int_0^\infty t \cdot f(t)\, dt = \int_0^\infty R(t)\, dt \tag{1.2}$$

where $f(t)$ is the *probability density function* of $F(t)$ (Failure probability).

$$f(t) = \frac{dF(t)}{dt} = -\frac{dR(t)}{dt} \tag{1.3}$$

$f(t) \geq 0$ for all $t \geq 0$ and $\int_0^\infty f(t)\, dt = 1$. MTTF is a common factor to reveal how the defects or degradation mechanisms influence the system reliability (cf. Section 2.2).

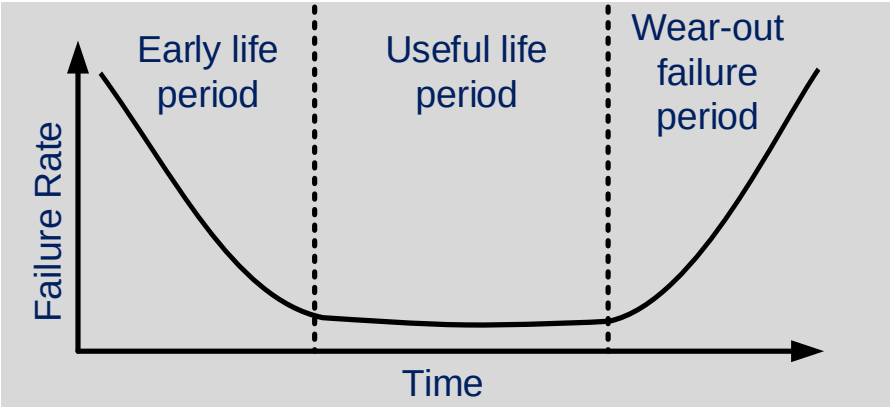
1.2 Failure Rate over Circuit Lifetime (Bathtub Curve)

The *Failure Rate* $\lambda(t)$ of an electronic equipment is another key indicator of the hardware reliability. It is defined as number of failures per time unit compared with number of surviving components and denoted as:

$$\lambda(t) = \frac{f(t) \cdot N}{(1 - F(t)) \cdot N} \tag{1.4}$$

where N is the number of components in overall equipment population and $F(t) \cdot N$ presents the number of failed components at time t, thus $(1 - F(t)) \cdot N$ refers to the number of surviving components.

The failure rate of a device varies over its lifetime. Carhart describes it as a bathtub curve shown in Fig. 1.1 composed of three distinct regions: an infant mortality region, useful life, and a wear-out region [Carha53].



▲ **Figure 1.1** — Bathtub curve [Shoom02]

Infant Mortalities (Early Life Failures) Region:

At or near the beginning of the operating period, young devices often have a high failure rate due to the performance marginality or manufacturing imperfections, such as poor insulation, weak connections, inadequate parameter matching, etc. To eliminate the early life failures, most fabricators put their chips to an initial burn-in test during the manufacturing quality control and defective devices are weeded out before the product shipment (cf. Section 4.1.2).

Useful Life Period:

During the middle period of the device operation, fewer failures occur. Those failures are mainly caused by accidental conditions (e.g. vibrations in transport, the humidity environment or ionizing radiation), unexpected operation stresses (e.g. the high temperature or supply voltage) or intermittent soft errors caused by cosmic rays or radiations. Consequently, such failures in the device useful life period are called *chance* or *random failures*. After excluding the components with early life failures, the failure rate of remaining devices exhibits a relatively constant value.

Wear-out Region:

Old components deteriorate and face a rising failure rate due to aging effects. As the transistor size scales and the current density dramatically increases, the high power density raises the local temperature and brings about hot spots. Most aging mechanisms, such as Bias Temperature Instability (BTI), Hot Carrier Injection (HCI), Electromigration (EM) or Gate Oxide Breakdown (OBD), depend highly on the temperature, power, and current density on the chip. As a result, the new technology generations will become more prone to aging-induced failures (cf. Section 2.2). To slow down the device degradation speed and cope with the rising failure rates, aging countermeasures such as the frequency or voltage scaling and on-line testing or failure prediction schemes are widely proposed (cf. Section 4.3 and 4.4).

1.3 Improve the system reliability

Though the reliability of individual hardware components decreases due to the technology scaling, consumer requirements of system reliability keep increasing. For instance, a study of US data centers announced that an average cost of an unplanned data center outage is around US$7,900 per minute in 2013, which has increased by 41% from $5,600 of 2010 [Sverd13]. Further, a survey from Gartner Research showed that more than 84% of organizations depend on systems over five years old, and more than 50% still work on hardware components over ten years old [Blome07]. On the other hand, a modern luxury automobile may have up to 100 electronic control units (ECUs) typically composed of microprocessors. The ECUs receive the electric signals transmitted by the sensors and process these data to generate control signals for the actuators [Kaise15]. A failure in such safety-critical applications (e.g. nuclear power plants, medical equipment

or transportation facilities) could be life-threatening, posing new challenges for the design and test.

For maintaining a high reliability and extending the useful life period of a hardware system, some major principles are listed as flows:

Burn-in Test

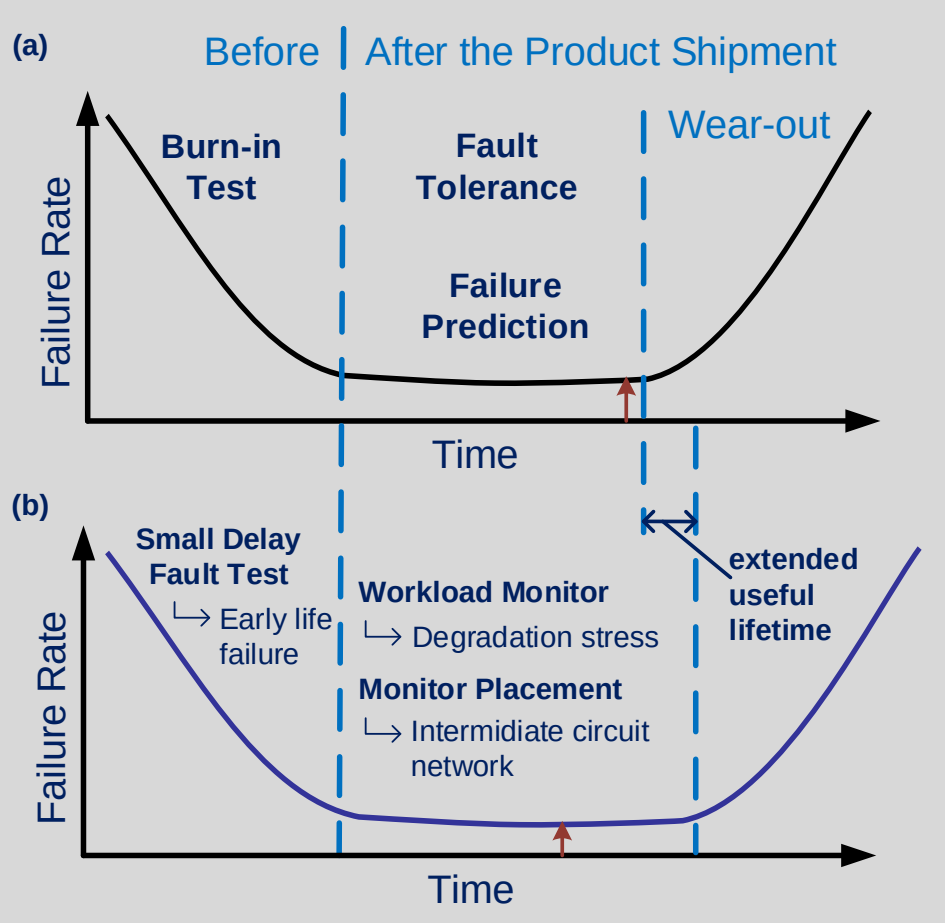
As shown in Fig. 1.2 (a), burn-in test is often conducted to identify and remove the defective or marginal chips before the product shipment. It prevents early life failures occurring in the field, ensures the product quality and diminishes the refund cost.

Burn-in test is a time compression testing procedure, which subjects hardware products to various or constant operational (e.g. high supply voltage or workload) or environmental (e.g. thermal, vibration or mechanical) stresses. Through the intense stress, failures can be exposed in hours or a few days, while under normal operating conditions they would show up in months or years. The stressed tests can reveal the design limitations, structure weaknesses, and fabrication imperfections, therefore help the hardware producers to assess the reliability of their design.

Fault Tolerance

Fault tolerance is a technique applying different *redundancies* to allow an uninterrupted system operation even at the appearance of component failures. As shown in Fig. 1.2 (a), fault tolerant techniques are performed during the useful life period (middle term of a device lifetime) after the product shipment.

Redundancy corresponds to extra resources beyond the requirement of a specified function. Error detecting and correcting codes are information redundancy methods, often utilized to overcome the errors existing in memories. Structural redundancy implements the additional hardware components in the system to detect, mask or correct errors. The triple modular scheme [vonNe56], replica units [Fujiw06] and self-checking circuitry [Ander71,Jha93] are of such kind. However, the redundant hardware is usually quite expensive regarding both power and area. The time redundancy repeats computations, therefore can potentially reduce the hardware cost. However, it induces a performance penalty. Additionally, in principle the time redundant techniques are restricted to transient fault detections unless any duplicate components are under consideration.



▲ **Figure 1.2** — Methods for reliability improvement

Failure Prediction

Failure prediction techniques are conducted also during the useful life period. The prediction approaches often measure some performance indicators on chips and estimate system healthy, then provide an alert (red arrow in Fig. 1.2 (a)) to warn and eventually prevent the imminent failure. Since random failures (cf. Section 1.2) are unpredictable, such failure prediction methods are principally limited to forecast the degradation induced failures by in-field monitoring the gradual behavior changes in the circuit. As a result, the prediction schemes are typically applied to observe the performance deterioration due to device aging. Comparing to fault-tolerant approaches, failure prediction methods are more cost-effective and can cooperate with aging countermeasures to mitigate the degradation impact and extend the system lifetime.

The work at hand focuses on using aging monitor, one common prediction technique to improve the reliability of the hardware system.

Design for Reliability

To relieve the variability impacts on the circuit performance and reliability, deterministic VLSI designs slowly change to probabilistic and statistical designs. Electronic Design Automation (EDA) tools and developing methods are required to take effects of variation into account and optimize the target system not only for its operation frequency but also for the power consumption, leakage current, and their variability distribution.

The multicore design is also a new tendency to improve hardware reliability and maintain high performance. A multicore chip can provide an identical logic throughput at a lower voltage and frequency, reducing the power consumption. Besides, multiple cores in the system can be used as the hardware redundancy, error detection and correction units, self-checking circuitry or a system reconfiguration controller to mitigate the performance degradation [Borka08].

1.4 Overview and Contributions

The bathtub curve shows two vulnerable periods of system reliability over the hardware lifetime: the early life and wear-out failure period. Standard techniques to improve the reliability during these particular time intervals are rather expensive. With technology scaling, the growing complexity continuously challenges the feasibility and efficiency of the conventional methods.

As shown in Fig. 1.2 (b), this dissertation introduces a small delay fault test method by reusing aging monitors [Liu17] to reduce early life failures (cf. Chapter 7) and presents a synthesis flow of the workload monitor [Baran13] (cf. Chapter 5) and the placement scheme of delay monitors [Liu15] (cf. Chapter 6) to forecast the wear-out degradation as early as possible. The work at hand aims to reduce the test and monitoring cost for maintaining the high reliability and prolong the system lifetime of the coming technology generations from following aspects:

- **Efficacy:** During manufacturing test phase, the monitor reuse approach (cf. Chapter 7) can be applied for small delay fault test. The experimental results show a good fault coverage even for faults with very small magnitude. Small delay

faults are an essential indicator for early life failures. Consequently, incorporating with faster-than-at-speed test, the monitor reuse method provides a potential alternative option to reduce burn-in tests for early life failure.

During the useful life period, workload monitors are introduced to directly measure the operational stress, i.e. the cause of degradation instead of the degradation effects e.g. delay or working current. Thus, the workload monitors (cf. Chapter 5) can sense aging at very early stage and perform countermeasures proactively. The device useful lifetime can be possibly extended by enabling mitigation procedures at a earlier stage. The red arrows in Fig. 1.2 (a) and (b) represent the moment for countermeasure activation.

To the same end, the monitor placement method (cf. Chapter 6) takes path sensitization information into account and selects observation points at the intermediate circuit network, therefore reduce the detection latency of delay degradation and the imminent timing failures.

- **Efficiency:** As an alternative to burn-in tests, the monitor reuse approach does not require external infrastructures to provide stress conditions for circuits under test. Due to the reuse of monitors, which have been integrated on-chip for aging prediction, the hardware overhead induced by monitor control for the testing purpose is very limited. Since aging monitors can be used to evaluate the test responses, the X-tolerant MISR, and on-chip storage for intermediate signatures in the Built-in Self Test (BIST) during the faster-than-at-speed test are not needed anymore. Similarly, when comparing to the Automatic Test Equipment (ATE), the monitor reuse method avoids the needs of the X-handling compactor and eases the requirements of the large bandwidth for test responses transmission.

  The topological analysis of the circuit structure shows that path segments are typically shared by multiple paths. The delay monitors inserted at carefully selected positions i.e. observation points can assess partial degradation of all or at least a high fraction of longer paths. In consequence, our monitor relocation method can reduce the number of monitors integrated at observation points by up to 98% compared to the end-path placement.

- **Generality:** The methods in this thesis do not limit themselves to a particular core structure or hardware system. There are no specific requirements for circuit operation conditions or applications running on the target system.

By detecting the deviated delay, the testing and monitoring schemes are independent of aging mechanisms or models and applicable to any manufacturing defects or in-field noise (e.g. the power noise, crosstalk, etc.) behaving as small delay faults. For implementation, any aging monitor design satisfying the delay detection principle mentioned in Section 4.4 can be chosen for our testing and placement methods (cf. Chapter 6 and 7).

The test scheme in Chapter 7 can cooperate with infrastructures in different testing frameworks such as the Automatic Test Equipment (ATE) and Built-in Self Test (BIST) structure. Additionally, the aging prediction methods in Chapter 5 and 6 can support any mitigation means of aging with their immediate and accurate prognostic alerts.

The remainder of the thesis is structured as follows:

Chapter 2 – *Fundamental Basics* – briefly presents the basics of fault modeling. An overview of aging mechanisms is followed by the aging-aware gate delay models. To detect the deviated delay from design specifications, delay tests are commonly utilized as a standard technique to tackle the problem.

Chapter 3 – *Formal Foundations* – provides the foundations of formal methods such as the Algebraic Decision Diagram construction, Boolean satisfiability (SAT) solving, and pseudo-Boolean optimization.

Chapter 4 – *State of the Art* – discusses the existing small delay fault test schemes, as well as monitoring approaches for aging degradation such as on-line built-in self-test techniques and prognostic monitor designs. State-of-the-art methods for on-chip monitor placement are reviewed.

Chapter 5 – *Workload Monitor for Stress Estimation* – introduces a workload monitor construction procedure [Baran13]. The workload monitor observes the most relevant subset of the circuit's primary and pseudo-primary inputs and assesses the experienced stress. The workload monitor can be used to pro-activate the timely adoption of aging countermeasures to prevent or reduce the performance degradation.

Chapter 6 – *Efficient Monitor Placement for Early Aging Prediction* – describes a delay monitor placement method by analyzing the topological circuit structure and sensitization of paths [Liu15]. The positions for delay monitors integration are meticulously selected in the combinational circuit nets, named observation points. With the novel

placement, the number of inserted monitors is significantly reduced, compared to the placement at the end of long paths.

Chapter 7 – *Aging Monitor Reuse for Small Delay Fault Testing* – addresses a monitor reuse approach on performing cost-effective small delay fault tests [Liu17]. By reusing the monitor integrated into the device under test for aging prediction, complex on-chip structures or expensive high-speed ATE for test response evaluation can be avoided. To prevent all possible false alerts of monitors at higher frequencies, we introduce a masking scheme of monitors and patterns. For a maximum fault coverage, the masking scheme maps the monitor and pattern selection to a pseudo-Boolean optimization problem.

Chapter 8 – *Conclusion* – recapitulates the contributions of this thesis and indicates future research directions that may benefit from this work.

CHAPTER 2

Fundamental Basics

This chapter briefly outlines the basics of fault modeling and an overview of aging mechanisms which induce delay deviations into the aged devices. The well-defined aging models in the literature are presented and later utilized to evaluate and validate our proposed monitoring (cf. Chapter 5) and placement approach (cf. Chapter 6). Additionally, this chapter also introduces the standard hardware test techniques targeting delay faults.

2.1 Fault Modeling

A *defect* is an unintended structure deviated from the specification of the manufactured design. Defects are often caused by distortion of silicon crystal structure and additional, missing or wrong material constructed on-chip, e.g. opens, shorts or bridges in interconnects or damaged oxides of transistors. They can be induced by impurities of the materials, the strike of dust particles, imperfect production environment or manually handling issues at any step during the manufacturing process [Segur04].

Fault modeling is a procedure to formulate the behavior of physical defects at the different abstraction levels of VLSI circuits to allow algorithmic test generation and fault simulation. *Fault models* abstract the timing or logical behavior from physical defects and are often described at the transaction, register-transfer, gate, and layout

level. The classic fault models such as stuck-at faults, bridges, stuck-open faults, delay faults, and transients are intensively studied by the academic [Wunde10] and generally applied in the industry. This work focuses on the delay faults at the gate level. Thus a short introduction of the relevant models is listed as follow.

2.1.1 Delay Fault

Delay Faults are applied to model the defective timing behavior of a single logic block (e.g. a transistor, an interconnect, a gate or a circuit path), causing a transition delay above its manufacturing specification. A delay fault models many types of physical defects in real silicon such as effects of process variations, power supply noise, crosstalk, resistive opens, resistive shorts etc. [Tehra11], which slow down the signal propagation and reduce the operation performance of the circuit instead of directly alter its logic functionality.

The transition delay, gate delay, and path delay fault model are the three most commonly used delay fault models and are discussed following [Wunde10].

Gate delay fault

The *gate delay fault* model is under the assumption that the increased transition delay arising from a defect in such gate is lumped at that single node and results in at least one path delay violating the functional clock [Store77]. The gate delay faults are observable only when the slack of a propagation path is smaller than the fault size. A slow-to-rise (delayed rising transition) and a slow-to-fall delay fault (delayed falling transition) are associated with each node in the circuit.

Transition delay fault

The *transition delay fault* model is a simplified version of the gate delay fault model assuming that a significant amount of delay, larger than any path slack through the fault site is added to a single node. Such transition delay faults will lead to set-up time violations at all reachable flip-flops and outputs at nominal operation frequency. It indicates that any propagation path with any slack size through the particular transition fault site can be utilized for fault detection [Waicu87].

Due to the large delay size, the test pattern generation approach targeting stuck-at faults can be modified and applied to transition faults with limited effort. Usually,

the transition fault model can cover more physical defects and keep a similar high testability compared to stuck-at faults [Krsti98]. A circuit with $|V|$ number of nets has $2 \times |V|$ number of transition faults in total (a slow-to-rise and slow-to-fall fault at each net).

Path delay fault

A *path delay fault* represents an increased propagation delay of a particular path in the combinational circuit. A *circuit path* refers to a combinational gate chain, starting from a primary or pseudo-primary input (i.e. the output of a clocked flip-flop) and ending at a primary or pseudo-primary output (i.e. the input of a flip-flop). Opposite to the local delay fault models such as gate delay fault or transition fault, a path delay fault is a global model, because it is associated with an entire circuit path. The delay increment of the faulty path can be induced by a single or multiple nodes (gates or interconnects) along the path, due to various defects or the parametric shifting at multiple positions on the path [Smith85].

However, the number of paths increases exponentially with the number of gates in a circuit, resulting in a significant total number of path delay faults. Enumerating all path delay faults in an industrial design during manufacturing tests becomes infeasible. Consequently, the path delay fault model is often applied to a small portion of selected critical paths for pattern generation or circuit timing estimation. In Section 6.4.1, long paths are chosen to analyze the accumulated delay along target paths after aging degradation, and path-delay ATPG patterns are used to evaluate the proposed *Observation Point Selection* approach.

2.1.2 Small Delay Fault

Small delay faults are a form of gate delay faults, which introduce a small amount of extra delay to a design node and were firstly alluded in [Park88]. Defects such as resistive opens, resistive bridges, gate-oxide defects or parametric deviations of transistors often manifest themselves as small delay faults (SDFs), introducing an additional small delay at cells or interconnects [Monta02, Tayad07]. SDFs can also arise from power supply noise, crosstalk or aging degradation, intensified by process variations [Natar98, Tehra11].

Hidden delay faults (HDFs) [Helle14] are a subset of SDFs for which the slack of the longest sensitizable path through the fault site is larger than the fault size, thus the fault effect cannot be detected by any at-speed or even timing-aware delay test.

Test methods targeting SDFs or HDFs are discussed in Section 4.1. It is also the focus of this work. A novel testing approach to detect small delay faults by reusing the in-situ aging monitors are presented in Chapter 7.

2.2 Aging Mechanisms

The relentless technology scaling comes along with a growing susceptibility to hardware aging, due to the wear-out mechanisms such as Bias Temperature Instability (BTI), Hot-Carrier Injection (HCI), or electromigration (EM), causes a parameter shift in transistors or interconnects over the lifetime. After introducing the concept of *delay fault* (cf. Section 2.1.1), this section presents how the performance degradation induced by aging, will grow into delay faults and may eventually bring about a life-threatening failure.

The electrical, thermal, mechanical, and chemical environment for device operation, as well as the system workload, affect the hardware wear-out speed and its life span (i.e. time to failure). The physical basis and the influencing factors of the aging mechanisms are intensely studied and analyzed to accurately estimate, predict and effectively eliminate the degradations and improve the reliability and system lifetime.

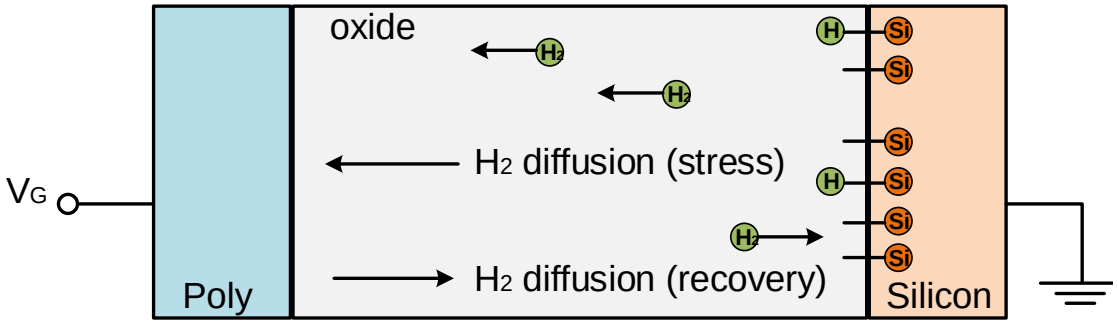
This section introduces the dominant aging mechanisms for transistors: Bias Temperature Instability (BTI), Hot Carrier Injection (HCI), Time Dependent Dielectric Breakdown (TDDB) and interconnects: Electromigration (EM). In each subsection, we introduce the physical mechanism of aging effects and then present the device-level parameter drift caused by transistors or interconnects wear-out. The time-to-failure model is expressed to show the aging impacts on the device reliability. Besides, we review the circuit-level models of the transition propagation delay through circuit gates for two dominated aging mechanisms: BTI and HCI.

2.2.1 Bias Temperature Instability (BTI)

Bias temperature instability is one of the major causes of wear-out degradation and affects the MOS-structure. BTI happens when the gate of a MOSFET is biased in inversion. When the gate is under the negative bias, the degradation effect is called

Negative Bias Temperature Instability (NBTI). Similarly, PBTI appears when the gate of a MOSFET is biased positively. Based on the biased voltage and types of transistors, four different permutations of BTI exist: NBTI/PMOS, PBTI/PMOS, NBTI/NMOS, PBTI/NMOS. PBTI/PMOS and NBTI/NMOS combinations are less prone to the BTI degradation. PBTI for n-MOSFETs was negligible for devices with the SiON gates and became severe with the High-K Metal Gate (HKMG) insulators. However, according to the latest reports from sub-22 nm Replacement Metal Gate (RMG), PBTI is no longer a significant reliability concern for FinFET technologies [Mahap16]. NBTI in PMOS transistors exhibits a dominant BTI effect for SiON devices and remains a serious issue for sub-22 nm FinFET technologies [Mahap16]. Therefore we choose NBTI in PMOS as the BTI form to detail in this subsection.

When an electric field stresses the gate oxide of a PMOS transistor, the holes in the inversion layer interact with the $Si-H$ bonds at the silicon and oxide interface, shown in Fig. 2.1. This interaction weakens the $Si-H$ bonds at elevated temperature and leads to separation of H atoms. The H_2 diffuses and the Si dangling bonds cause the traps (the number of traps is denoted as N_{IT}) to build-up at the $Si-SiO_2$ interface. The threshold voltage (V_{th}) increases when more traps form (i.e. the threshold voltage deviation is proportional to the number of traps at the interface: $\Delta V_{th} \propto N_{IT}$), leading to a reduction in drive current (I_{Dsat}) [Masse04]. This degradation process gets intensified with the scaling of oxide thickness t_{ox}.



▲ **Figure 2.1** — The physical mechanism of NBTI [Alam03]

However, when the negative bias of the PMOS transistor is removed, and a reverse bias is applied, the interface traps generated under stress are subsequently passivated and bring about a recovery from the V_{th} or I_{Dsat} degradation [Chen03].

The Reaction-Diffusion (R-D) model [Ogawa95] accurately represents the stress ($V_{gs} = -V_{dd}$) and recovery phase ($V_{gs} = 0$) of the NBTI effects. V_{gs} refers to the gate-source voltage and V_{dd} denotes the drain voltage. The increase of the threshold voltage at a particular moment $\Delta V_{th}(t)$ can be formulated as the iterations from a previous time period [Wang07a]:

$$\text{Stress:} \quad \Delta V_{th}(t) = (K_v(t - t_0)^{0.5} + \sqrt[2n]{\Delta V_{th}(t_0)})^{2n} \tag{2.1}$$

$$\text{Recovery:} \quad \Delta V_{th}(t) = \Delta V_{th}(t_1)(1 - \frac{2\xi_1 t_e + \sqrt{\xi_2 C(t - t_1)}}{(1+\delta)t_{ox} + \sqrt{Ct}})$$

where $\Delta V_{th}(t_0) = qN_{IT}(t_0)/C_{ox}$, q is the electronic charge; $N_{IT}(t)$ is the number of traps at the $Si - SiO_2$ interface at a specific time; C_{ox} is the interface capacitance. The times t_0 and t_1 refer to the moment at which the stress or recovery phase starts, respectively. For a H_2 diffusion-based model, n equals 0.167, and for a H-based model, n equals 0.25. K_v can be presented as:

$$K_v = (\frac{qt_{ox}}{\epsilon_{ox}})^3 K_1^2 C_{ox}(V_{gs} - V_{th})\sqrt{C}exp(\frac{2E_{ox}}{E_{o1}}) \tag{2.2}$$

C is dependent on the absolute temperature T and can be expressed as:

$$C = T_0^{-1} exp(-E_a/kT) \tag{2.3}$$

where k is the Boltzmann's constant, T_0 is 10^8 (s/nm^2) and E_a refers to the activation energy. ϵ_{ox} is the oxide permittivity. E_{ox} is the electric field over gate oxide and equals $(V_{gs} - V_{th})/t_{ox}$. K_1, E_{01} and δ are technology dependent constants. ξ_1 and ξ_2 are the back-diffusion constants. The effective oxide thickness t_e is equal to either t_{ox} or the diffusion distance in the oxide.

In reality, it is often infeasible to simulate the NBTI degradation cycle-by-cycle or with every stress and recovery phase. [Alam03] proves that the intrinsic symmetry of the stress and recovery phases weakens the frequency dependence of the dynamic NBTI effects. During a long term, the transistor threshold voltage drift is less affected by the alternation speed of the two NBTI phases, but more determined by the entire operation time of the device. A numerical solution is derived from the Reaction-Diffusion model

and provides a closed form for the upper bound of ΔV_{th} as an expression of the duty cycle α, clock period T_{clk} and the operation time period $(0, t]$ [Cao11]:

$$\Delta V_{th}(t) = \frac{K_v \sqrt{\alpha T_{clk}}}{1 - [\beta(t)]^{\frac{1}{2n}}} \tag{2.4}$$

where K_v has been expressed in Equ. 2.2 and n has the identical value as in Equ. 2.1. β is a function of time t presented as:

$$\beta(t) = 1 - \frac{2K_1 + \sqrt{K_2 C(1-\alpha) T_{clk}}}{(1+\delta)t_{ox} + \sqrt{Ct}} \tag{2.5}$$

where K_1 and K_2 are constants. C, t_{ox} and δ share the same value and definition as in Equ. 2.1.

The mean time to failure $MTTF$ in hours is often estimated by the following failure model [Elect17]:

$$MTTF = A \cdot exp(-\beta_0 E_{ox}) \cdot exp(\frac{E_a}{kT}) \tag{2.6}$$

where A is a constant, β_0 is the electric field intensity coefficient (in cm/MV) and the rest of the variables have the identical meaning and value as mentioned above.

The NBTI-affected threshold voltage increase causes a growth of the circuit gate delay and degrades the device performance. A NBTI gate delay model and its deriving process are shown below. The delay τ of a gate can be approximately represented as [Paul07]:

$$\tau = \frac{C_L V_{dd}}{I_d} = \frac{K}{(V_g - V_{th})^{\alpha}} \tag{2.7}$$

$$K = \frac{C_L V_{dd}}{\mu C_{ox} W_{eff}/L_{eff}} \tag{2.8}$$

where C_L is the load capacitance, V_{dd} is the supply voltage and V_g is the gate voltage of a transistor. α refers to the velocity saturation index and has value ranging from 1 to 2. μ denotes the mobility and C_{ox} is the oxide capacitance. L_{eff} and W_{eff} are the effective channel length and width of a transistor respectively. Differentiating Equ. 2.7 w.r.t. V_{th} we obtain,

$$\frac{\Delta\tau}{\tau} = \frac{\alpha \Delta V_{th}}{(V_g - V_{th})} \tag{2.9}$$

Then, we integrate Equ. 2.9 from both sides.

$$\int_{\tau_0}^{\tau_0+\Delta\tau} \frac{d\tau}{\tau} = \int_{V_{th0}}^{V_{th0}+\Delta V_{th0}} \frac{\alpha dV_{th}}{(V_g - V_{th0})} \tag{2.10}$$

After integration we derive,

$$ln(1+\frac{\Delta\tau}{\tau_0}) = -\alpha \cdot ln(1-\frac{\Delta V_{th}}{V_g - V_{th0}})$$

where V_{th0} is the threshold voltage at the time t_0 and τ_0 is the corresponding gate delay at such moment. We use Taylor series on both sides of Equ. 2.10 and neglect the higher order terms, then obtain:

$$\Delta\tau = \frac{\alpha\Delta V_{th}}{(V_g - V_{th0})} \cdot \tau_0 \tag{2.11}$$

There might be several PMOS transistors in one gate and the voltage degradation ΔV_{th} can be different between transistors. In such cases, usually, the largest ΔV_{th} is chosen to calculate worst case delay degradation of the gate. The transistor threshold voltage drift ΔV_{th} can be derived from Equ. 2.4.

From the extensive works of NBTI characterization and modelization, we can see that NBTI increases the threshold voltage V_{th} of p-MOSFETs when the transistors are under inversion and endure an elevated temperature. Consequently, the factors such as the number of inverted PMOS transistors and temperature can be used to estimate the NBTI stress and predict the aging effects before any degradation takes place (cf. Chapter 5). During a short observation time, the degraded V_{th} partially anneals if the stress (i.e. the bias gate voltage) is removed. In the long run, V_{th} shows a power law time dependence [Cao11]. At the circuit-level, the gate delay increases gradually by time due to NBTI, and its increment is proportional to the threshold voltage growth of the stressed transistors (cf. Equ 2.11). As a result, various monitors are developed. They periodically sens the timing deviation, represented as the path delay, operation frequency or signal slew-rate to predict aging failures (cf. Section 4.4).

2.2.2 Hot Carrier Injection (HCI)

In MOSFET transistors, especially in the region with a high electrical field near the drain, the carriers (electrons or holes) flow in and are accelerated by the field. Carriers obtained high kinetic energies become "hot" and may potentially pass the barrier between the substrate and the gate oxide film. Some of the injected hot carriers are trapped in gate oxide and produce a space charge. Those hot carriers which have not been trapped in the oxide generate the gate current flowing in the direction of the substrate [Segur04].

Due to various bias voltage conditions of transistors, the carriers acquire energy and become "hot" through four physical mechanisms, in Fig. 2.2:

Drain Avalanche Hot Carrier (DAHC) injection in Fig. 2.2 (a):

When a high voltage is applied to the drain of a transistor (i.e. $V_d > V_g$), channel carriers are accelerated due to the high electronic field near the drain. As a result, some carriers enter the drain depletion region and may collide with the Si atoms in the lattice producing electron-hole pairs. A portion of those electron-hole pairs may arise further impact ionization and lead to some injected carriers trapped in the gate oxide. Carriers that do not get trapped in the oxide generate the gate current and the remaining electron-hole pairs in the substrate form the substrate current (I_{sub}). When $V_d = 2 \cdot V_g$, the worst degradation appears and I_{sub} reaches its maximum value.

Channel Hot Electron (CHE) injection in Fig. 2.2 (b):

When both gate and drain voltage are high, carriers accelerated toward the drain crash on the gate oxide before reaching the drain due to the high vertical field generated by the gate voltage. The carriers are injected without any energy-losing collision. The worst degradation condition is $V_g = V_d$.

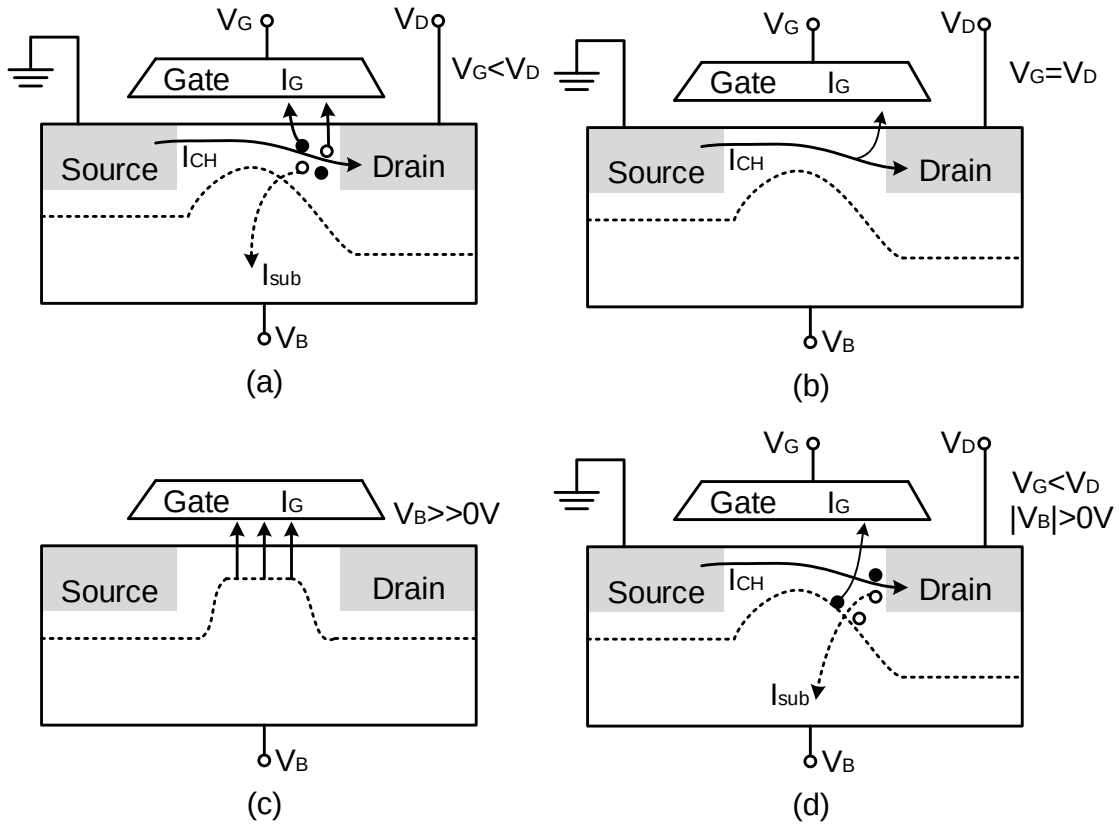
Substrate Hot Electron Injection in Fig. 2.2 (c):

When substrate bias is high (i.e. $|V_b| >> 0$), the carriers in the substrate are accelerated towards the channel, acquire kinetic energy due to the surface field and get trapped in the gate oxide.

Secondary Generated Hot Electron (SGHE) injection in Fig. 2.2 (d):

Under a similar condition as DAHC, secondary carriers (electrons or holes) generated by previous impact ionization, flow towards the transistor channel and get accelerated due to the substrate bias. As a result, the secondary carriers strike the gate oxide and get trapped there.

The trapped charges, produced by any of the HCI mechanisms mentioned above, degrade the characteristics of transistors, such as the threshold voltage V_{th} and transconductance g_m, and increase the gate-induced drain leakage. The V_{th} determines the drain current I_{Dsat} of the transistor, then affects the speed of charging and discharging the load capacitances in the circuit structure. As a result, HCI will influence the transition propagation delay in circuit paths.



▲ **Figure 2.2** — Mechanisms of hot carrier injection: (a) Drain Avalanche Hot Carrier (DAHC) injection; (b) Channel Hot Electron (CHE) injection; (c) Substrate Hot Electron (SHE) injection; (d) Secondary Generated Hot Electron (SGHE) injection [Elect17]

The reaction-diffusion model is also applicable to the HCI process, in which the recovery phase is negligible. The degradation of the transistor threshold voltage ΔV_{th} follows a power-law dependence on time and is formulated as [Wang07a]:

$$\Delta V_{th} = \frac{q}{C_{ox}} K_2 \sqrt{Q_i} exp(\frac{E_{ox}}{E_{02}}) exp(-\frac{\phi_{it}}{q_m}) t^{n'} \tag{2.12}$$

where , K_2, ϕ_{it}, E_m and E_{02} are technology dependent constants. q, C_{ox} and E_{ox} share the same definitions as in the NBTI mechanism (cf. Section 2.2.1). Q_i is the inversion charge. n' is the time exponential constant and equals 0.45 in HCI process .

The transistor lifetime degradation D (e.g. a 10% increase of V_{th}) due to hot-carrier injection can be predicted by a theoretical model in [Hu85] which is expressed as:

$$D = C' \frac{W}{I_d} (\frac{I_{sub}}{I_d})^{-\Phi_{it}/\Phi_i} \tag{2.13}$$

where C' is a constant, W is transistor width, I_d is drain current, I_{sub} is substrate current, Φ_{it} is the interface state activation energy ($\approx 3.7\,eV$), and Φ_i is the impact ionization activation energy ($\approx 1.2\,eV$). $\Phi_{it}/\Phi_i \approx 3$.

HCI causes an adverse impact on the signal propagation delay of gates through the drain current I_d and threshold voltage ΔV_{th} degradation of transistors. [Yonez98] introduces the first HCI delay model for gate-level timing simulation. The model is ratio based and consists of the following expressions Equ. 2.14 to Equ. 2.16:

$$\tau_{aged} = \alpha \tau_0 \tag{2.14}$$

The degraded delay τ_{aged} of a gate is obtained from its fresh delay τ_0 and the prechar-acterized aged-to-fresh delay ratio α.

$$\alpha = \sum_{i=1}^{n} \alpha_i - (n-1), \quad i = 1, 2, \ldots, n \tag{2.15}$$

$$\alpha_i = f(T_{slew}, C_L, N_{sw}) \tag{2.16}$$

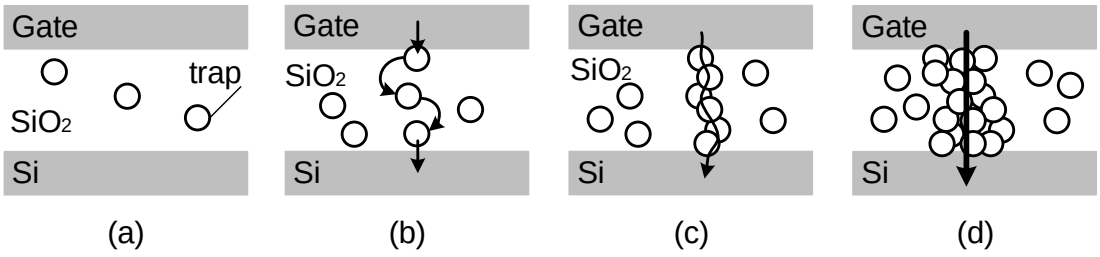
where n is the number of series-connected transistors. i is an index, ranging from 1 to n. α_i is the ratio of the i-th input pin of a gate without taking the influence of other input pins into account and can be represented as the function of the input slew rate T_{slew}, the load capacitance C_L and the number of *effective switchings* N_{sw} of the input pin. If an input toggle causes a transition at the gate output, we call this input toggle an *effective switching*. For each cell type in the library, the ratio α_i is characterized by performing transistor level simulations under certain operation and process conditions [Wu00].

Carriers are accelerated and become "hot" due to high operating voltage V_{DD}. When the energy is over a threshold, the hot carriers are injected into the gate oxide, resulting in the performance deterioration of transistors such as the V_{th} increase. HCI is an essential wear-out mechanism for NMOS transistors but is negligible for PMOS devices. Contradict to NBTI, the degradation of HCI is unreversible. Besides the high operating voltage and temperature, HCI depends on the switching activity of transistors instead of the static logic states of the circuit. The degradation of V_{th} due to HCI follows a power-law dependence on time (cf. Equ 2.12) and lead to a gradual delay increase in circuit-level. Thus, HCI-induced timing failures can be predicted similarly as for NBTI, by measuring the deviated circuit behavior periodically in-field by aging monitors (cf. Section 4.4).

2.2.3 Time Dependent Dielectric Breakdown (TDDB)

Time dependent dielectric breakdown is a wear-out mechanism in transistor oxide. Due to the electric field across the gate oxide, the high-energy charges (holes or electrons) are injected into the transistor oxide, generating randomly distributed traps in the bulk of the oxide. When enough traps are aligned close enough to form a conduction path between the gate and substrate, a thermally damaging leakage current passes through the oxide, and a catastrophic dielectric breakdown occurs. The oxide failure causes a short-circuit in the gate and makes the device unresponsive to its input stimulus. Based on different types of stress-induced leakage currents, the TDDB procedure can be separated into three levels [Segur04]:

- As shown in Fig. 2.3 (b), the neutral traps in the oxide form the defect paths from gate to the substrate. The electrons pass through the paths due to the electrical field and generate the stress-induced leakage current. This physical mechanism is called a trap-assisted tunneling.
- With a growing trap density over time, the oxide experiences a partial breakdown, also known as soft breakdown, shown in Fig. 2.3 (c). A soft breakdown will raise the gate current and noise permanently.
- When the oxide fails to isolate the gate and silicon, a hard breakdown occurs, shown in Fig. 2.3 (d). The gate current will encounter a continuous exponential increase.



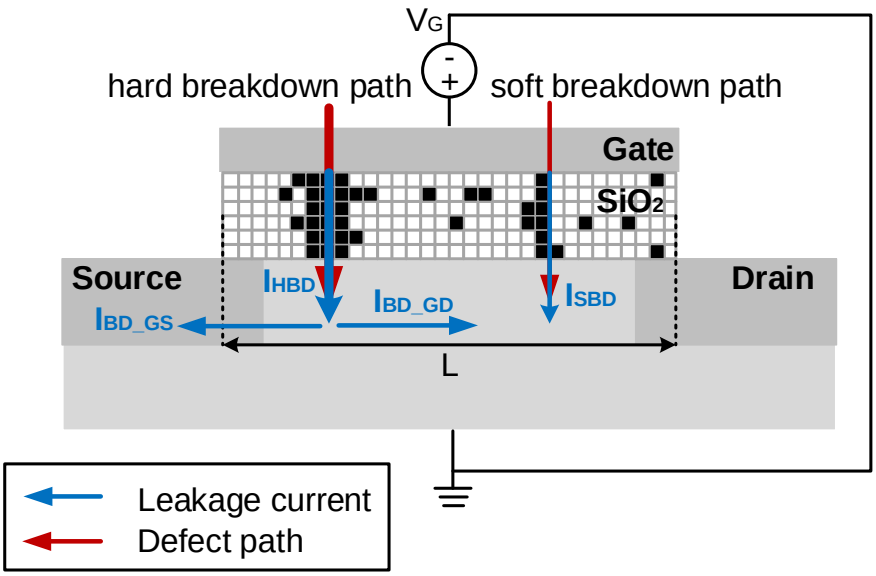
▲ **Figure 2.3** — The physical mechanism of the TDDB process [Jacks99]: (a) a fresh device; (b) a trap-assisted tunneling; (c) a soft breakdown; (d) a hard breakdown

The trap density in the oxide determining the defect path construction is a function of the gate voltage V_G and a power law of the stress time t in the anode hole injection model [Cha14]:

$$N_{trap}(t, V_G) = Aexp(BV_G)t^{\beta} \cdot t_{ox}WL \tag{2.17}$$

where A, B and β are fitting constants. t_{ox}, W and L are the gate oxide thickness, channel width and length respectively.

The Percolation Model (PM) represents the gate oxide under stress as a three-dimensional (3D) matrix. It applies calculations for placing neutral traps randomly within the 3D space model. Based on the placement analysis of neutral traps, the Percolation model can analyze whether any defect path is constructed and compute the number of conduction paths (Fig. 2.4). After the simulation of the percolation model, if any resistive defect

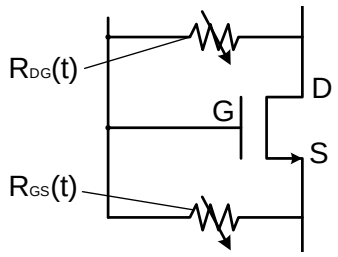


▲ **Figure 2.4** — The percolation model of the TDDB [Cha14]

path is formed, the transistor model can be modified by introducing two time-dependent resistances, shown in Fig. 2.5. If the defect path occurs between the drain and gate, R_{DG} is the breakdown resistance, calculated by Equ. 2.18 or Equ. 2.20 while the resistance between the gate and source R_{GS} is large enough and the resistive effect is negligible. Similarly, if the defect path happens between the gate and source, R_{GS} is the breakdown resistance and R_{DG} is large enough to be neglected. When Equ. 2.17 is applied to the percolation model analysis, the number of soft breakdown paths is a function of time. Hence, the resistance of the soft breakdown paths is modeled as [Cha14]:

$$R_{SBD} \cong V_G/[\frac{4e}{\hbar\alpha}N \cdot exp(-\alpha\Phi) \cdot sinh(\frac{\alpha e(V_G - V_0)}{2})] \tag{2.18}$$

where $\Phi = 3 \sim 4eV$, $V_0 = 0 \sim 0.5V$, $\alpha = 2 \sim 3eV^{-1}$, $\hbar$ is Planck's constant, e is the electron charge and N is number of soft breakdown conduction paths.



▲ **Figure 2.5** — A transistor model under the TDDB stress [Taizh15]

As shown in Equ. 2.17, the trap density increases with the stress time t, when the trap density exceeds a critical threshold (N_{DB}), the catastrophic hard breakdown occurs. The critical defect density N_{BD}, is obtained from experiments and the percolation model and expressed as:

$$N_{BD} = \frac{t_{ox}}{a_0^3} \cdot exp - \frac{a_0}{t_{ox}} \cdot ln(\frac{A_{ox}}{a_0^2}) \tag{2.19}$$

Where t_{ox} is the oxide thickness, A_{ox} is the oxide area and a_0 is a defect size for percolation model analysis.

When the hard breakdown happens, the resistance of the breakdown path is formulated as:

$$R_{HBD} \cong V_G^{1-p}/K \tag{2.20}$$

where K and p are the technology dependent coefficients. R_{HBD} has the value in the range of $K\Omega$.

A time-to-breakdown model (T_{BD}) for the ultrathin oxide is expressed a follow [Monsi01]:

$$T_{BD} = T_0 \cdot exp[\gamma \cdot (\alpha \cdot t_{ox} + \frac{E_a}{k \cdot T} - V_G)] \tag{2.21}$$

where γ is the acceleration factor, E_a is the activation energy, α is the oxide thickness acceleration factor, T_0 is a constant for a given technology, and T is the average junction temperature. The values of physical parameters in Equation 2.21 such as: $(\gamma \cdot \alpha) = 2.0\ 1/\text{Å}$, $\gamma = 12.5\ 1/V$ and $(\gamma \cdot E_a) = 575\ meV$ are extracted from experiments and published in [Monsi01]. As shown in the model, the time-to-breakdown (T_{bd}) is exponentially dependent on the temperature (T), gate voltage (V_G) and oxide thickness (t_{ox}).

A long-time application of transistor gate inversions causes tape injections in the oxide and may format a conducting path through the gate and substrate, resulting in the

TDDB. The dielectric of transistors experiences a gradual increase of the stress-induced leakage current at the beginning of the degradation. Then, a sudden death of the transistor function (i.e. a hard breakdown) occurs when the trap density exceeds a critical threshold. To prevent any unexpected hard breakdown, the TDDB degradation should be observed as early as possible, which motivates the work of the stress monitor construction in Chapter 5.

2.2.4 Electromigration (EM)

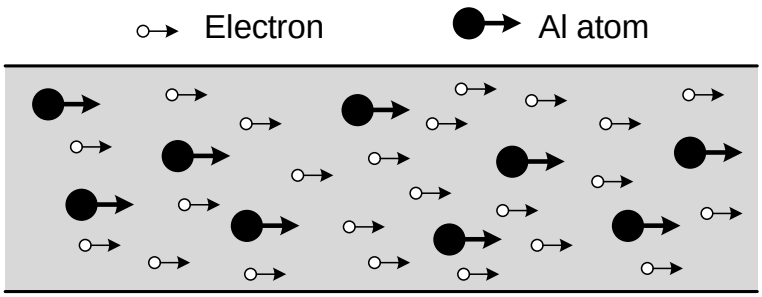
Electromigration is a wear-out mechanism of interconnects. Electromigration is a mass transport of metal atoms (e.g. Al, AlCu or Cu) caused by the passage of applied currents. The movement of metal atoms arises from high-momentum electrons, passing some of their momentum to the activated metal atoms [Stron09]. Therefore, Electromigration often happens under high current densities, and high temperatures. The atom flux moves along the direction of electron flow [Goel89].

Fig. 2.6 shows Al atoms moving with electrons if their thermal energy is at a certain active level and there are vacancies of atoms in the metal line to jump into. Flux J is defined as the number of particles (metal atoms) passing a unit area per unit time. The flux J_{em} of the metal atoms under electromigration is expressed as:

$$J_{em} = \frac{D}{kT}(Z \cdot q)\rho \cdot j_e \cdot N \quad (\frac{atoms}{cm^2 \cdot s}) \tag{2.22}$$

where Z is the electron-to-atom ratio, q is the electron charge, ρ refers to the metal resistivity, j_e is the electron current density, N denotes the concentration of moving metal atoms. D is the self-diffusion coefficient $D = D_0 exp(\frac{-E_a}{kT})$, k is the Boltzmann constant, T is the absolute temperature and E_a is the activation energy, which differ from each metal material. Copper has higher activation energy than aluminum, therefore is less prone to electromigration. However, copper can only mitigate the electromigration instead of fully solving the issue.

The electromigration can let voids (holes) or extrusion (hillocks) form at certain positions of the interconnects (e.g. at a via or a contact). The nucleated void grows in dimensions and reduces the cross-section area of the wires, consequently, elevates the resistance of the interconnects and increases the signal delay. Under further degradation, the void may eventually break the interconnect and lead to a full open defect. Contradictorily, the formation of extrusions may grow into a short defect. As a result,



▲ **Figure 2.6** — The moving metal atoms and electrons under electromigration [Segur04]

failures occur in-field after an unexpectedly short time of operation, although the circuits have passed all manufacturing tests.

Black's law [Black67] is empirically derived to calculate the MTTF for electromigration is represented as:

$$MTTF_{EM} = A \cdot (j_e^{-n}) \cdot exp(\frac{E_a}{k \cdot T}) \tag{2.23}$$

where E_a, j_e, T and k have the same meaning as in Equation 2.22. A is the original cross-section area, and n is a scaling factor, depending on the metal structure and in the range of $1 < n \leq 2$.

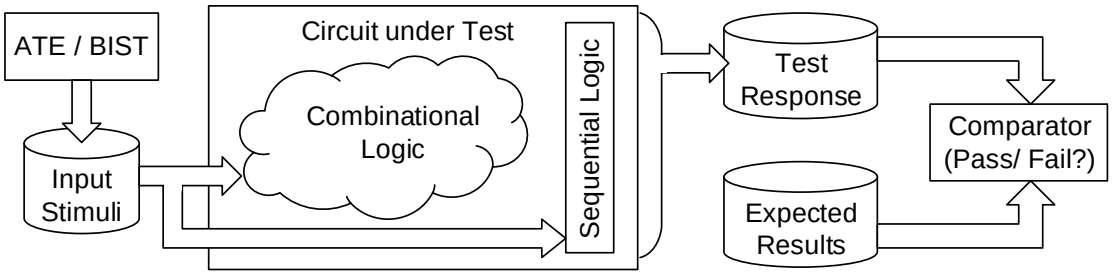
We can conclude from the equations (Equ. 2.22 to 2.23) that electromigration is inclined to affect the interconnects with a small cross-section area, an irregular granularity, locating at circuit hot-spots and suffering from a high current density. Since ideally in CMOS technology current occurs only during transistor switching, a high switching activity at the interconnect will enhance the stage of degradation. Additionally, With a constant supply voltage, the interconnects with a large fanout need longer time to charge or discharge the load capacitance, causing a longer-lasting current through and thus, intensifying its electromigration [Goel89, Ingel11]. Electromigration starts with varying the resistance of the interconnects. The voids and extrusions on wires may eventually deteriorate into open or short defects after degradation.

2.3 Hardware Test

The manufacturing test aims to detect any possible defects in the circuits, i.e. ensure a good product quality before shipment and improve the fabricating process for a better

yield. Thus, the fundamental principle of testing is to distinguish the defective circuits from the defect-free ones. *Delay test* is the manufacturing test targeting *delay faults*. It is an essential technique to prevent timing failures and detailed below (cf. Section 2.3.1).

During a test (shown in Fig. 2.7), the automatic test equipment (ATE) or other test hardware (e.g. built-in self-test (BIST)) provides the test vector (i.e. test patterns or input stimuli) at primary inputs or flip-flops (by the scan structure) and evaluates the test responses by comparing them to the expected results. The test quality depends on the thoroughness of the input patterns. A larger test pattern set may help to improve the test quality, however, will increase the test time, leading to a high test cost and postponing the time-to-market.



▲ **Figure 2.7** — The principle of a hardware test

The *parametric test*, *functional test*, and *structural test* are three major types of manufacturing tests [Tehra11]. The parametric test measures the electric properties (e.g. current or voltage) at certain positions of the fabricated chips. Functional testing verifies the correctness of functionality by activating each working feature with operation input stimuli. Structural testing focuses on the fault detection. By analyzing how a defect manifests the circuit behavior at the logic level, the faults are modeled, and test patterns are generated.

2.3.1 Delay Test

Aging mechanisms (cf. Section 2.2) degrade the timing behavior of microelectronic devices and makes them more prone to manufacturing defects [Tehra11]. Delay test is intended to target those delay-induced defects in the circuit and ensure all propagated data in the combinational network can reach the sequential elements (e.g. flip-flops) in time. Previously functional at-speed tests were applied for preventing the timing

failures, however, facing the issues of high test time and low defect coverage for industry designs.

The scan-based structural tests performing at-speed is a cost-effective alternative. Transition and path-delay fault models can be utilized for test pattern generation to achieve a relatively good coverage of delay-induced defects.

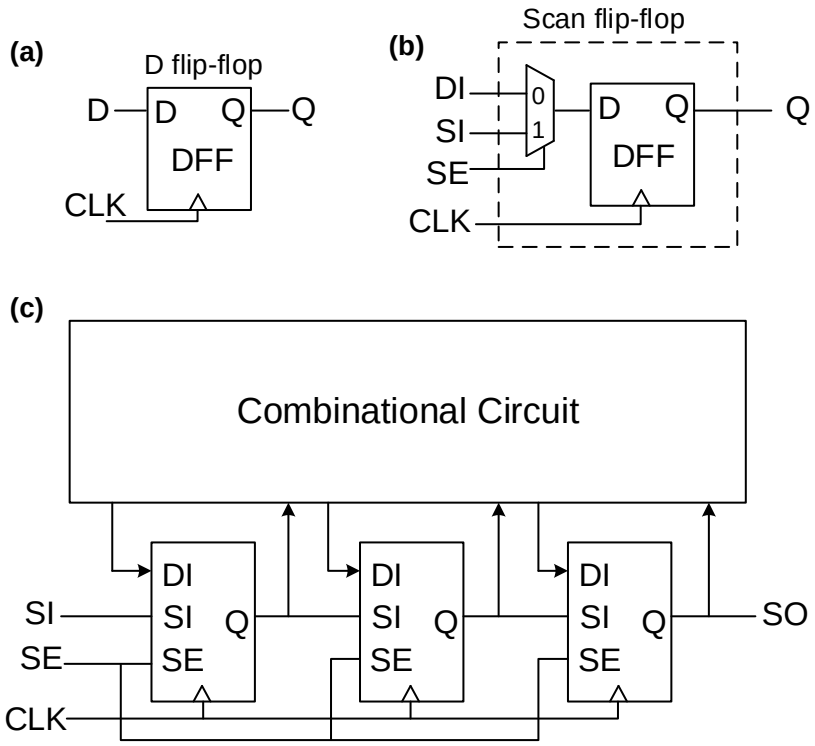
Scan-based test structure

To achieve a good controllability and observability of circuit logic states stored in flip-flops, scan flip-flops (Fig. 2.8 (b)) are developed and applied to replace the standard registers (Fig. 2.8 (a)) in the design for testing. As shown in Fig. 2.8 (b), two additional inputs, scan-in (*SI*) and scan enable (*SE*), are added to the scan flip-flop. The *SE* signal determines whether the scan flip-flop is in the operation mode or the test mode.

Fig. 2.8 (c) shows an example of scan-chains with three scan flip-flops. The *SI* pin of a scan flip-flop is connected to the output (*Q*) of the previous one. For the first element in the chain, the *SI* pin is connected to the primary input, and *Q* pin of the last scan register is linked to the primary output (*SO*). In the test mode, the scan path through these three registers are activated, and any desired logic states can be shifted into the flip-flops. After performing the test, similarly, the response of the circuit are shifted out from *SO* for evaluation. This active scan path makes the internal logic states of the circuit accessible by the external testers. Thus, it significantly reduces the complexity of test generation and improves the test coverage.

When delay tests are applied to scan-based designs, each pattern pair to generate transitions at circuit inputs consists of two test vectors: "initialization vector" V_1 and "launch vector" V_2. After the signal propagation through the fault sites, the test responses of the circuit under test (CUT) are captured at operation speed and later shifted out for evaluation.

To efficiently generate the launch transitions, and capture the test result at-speed with less control effort, different test methods are proposed. Launch-off-shift (LOS) (i.e. skewed-load [Savir92]), launch-off-capture (LOC) (i.e. broadside method [Savir94]), and Enhanced Scan [Dervi91] are the three most important approaches used in the industry.



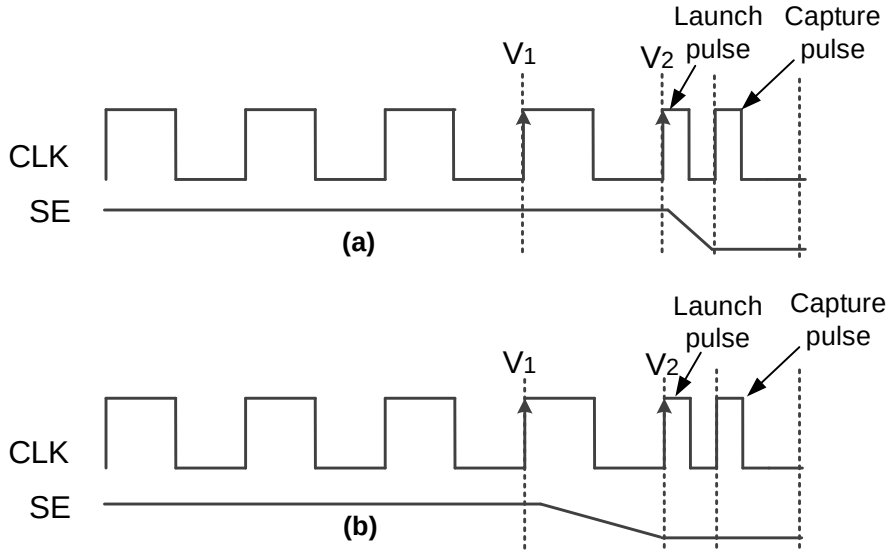
▲ **Figure 2.8** — (a) a normal D flip-flop, (b) a equivalent scan flip-flop, (c) a scan-chain with three scan flip-flops [Tehra11]

Launch-off-shift (LOS)

For the launch-off-shift method, the launch vector V_2 is applied by shifting one more bit through the scan-chain (Fig. 2.9 (a)). Therefore, the test vector V_2 is V_1 shifted by one bit (e.g. when V_1 is 001101, V_2 is 01101X. X refers to the last shift-in bit.) To change the scan flip-flops from the pattern shifting phase to circuit operation mode for testing, the scan enable signal (*SE*) is set to low after V_2 launches transition at circuit inputs. To obtain the test response at operation speed, the *SE* signal needs to be stabled as logic low before the capture clock edge. Consequently, LOS requires a timing critical *SE* signal, which can quickly switch to low at functional speed, leading to an expensive design for test (DFT) hardware.

Launch-off-capture

The launch-off-capture method does not require an at-speed SE signal. In the LOC method, the shifting phase is only used to provide V_1 initializing the logic state of CUT.



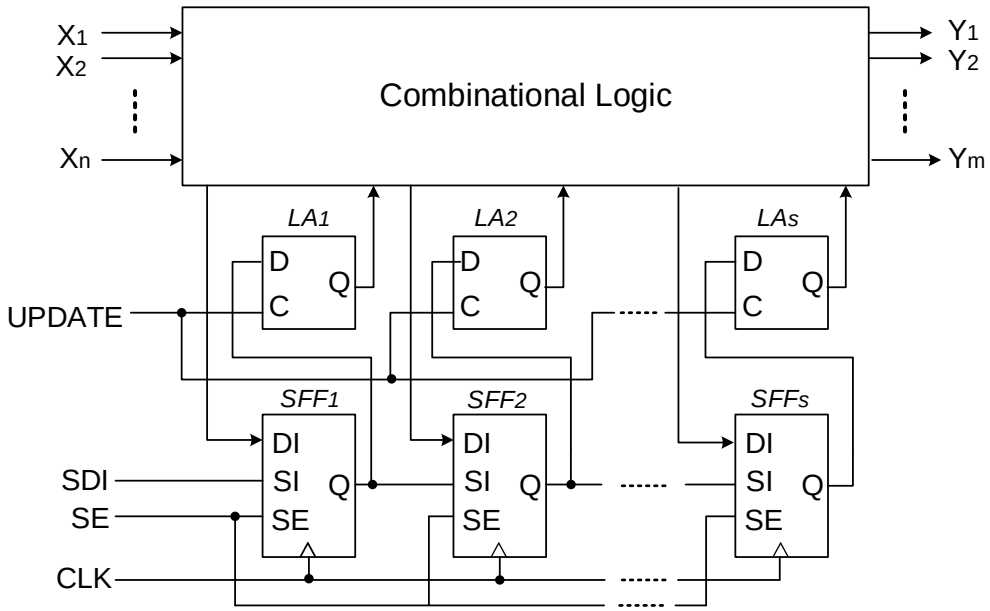
▲ **Figure 2.9** — Waveform of control signals: (a) launch of shift (LOS), (b) launch of capture [Tehra11]

V_2 is the circuit functional response of V_1. If we use the math term $outputs = f(inputs)$ to represent the functional dependency between input and output vectors, V_2 equals $f(V_1)$. Thus, shown in Fig. 2.9 (b), *SE* is set to low after the shifting phase (when V_1 is provided) and reaches its stable value before the launch cycle (when V_2 is applied). Usually, the shifting clock frequency is much lower than circuit function speed, resulting much weaker timing requirement of the *SE* signal.

Enhanced-scan

The enhanced scan can provide an arbitrary pair of vectors to the CUT by storing both test vectors (V_1 and V_2) on-chip. For this purpose, additional latches (LA_1 to LA_s) are integrated (Fig. 2.10). V_1 is first shifted into the scan cells (SFF_1 to SFF_s) and then put into latches when the *UPDATE* signal is high. Then, *UPDATE* is set to low (to maintain V_1 in latches) and V_2 is shifted into the scan cells. Once V_2 is ready in registers, *UPDATE* switches to high to change V_1 to V_2 and launches transitions at circuit inputs. At next clock edge, test responses are captured into the scan cells. To capture the test results at-speed, clock signal *CLK* has to cooperate perfectly with *UPDATE* to ensure that the next sampling edge comes exactly after one clock cycle.

Opposed to the functional (LOC) or the shifting (LOS) data dependency between the two



▲ **Figure 2.10** — Structure of enhanced-scan [Laung06]

vectors of each test pair, arbitrary test pairs allow the enhanced scan method to achieve the maximum coverage of the delay faults, however many false paths in operation may be sensitized during testing, causing the over-test issue. Each enhanced-scan cell requires an additional latch inducing a higher hardware cost. Additionally, an extra control signal *UPDATE* is added and it could be very difficult to maintain the timing relation between *UPDATE* and *CLK* at-speed.

CHAPTER 3

Formal Foundations

This Chapter establishes a brief introduction to *Boolean Algebra* as modeled in later chapters following the textbook [Brown90]. The formal notations and basic concepts of *Boolean Function* and its *Representations* form the basis of the Binary Decision Diagram (BDD) and Algebraic Decision Diagram (ADD), utilized for the workload monitor construction in Chapter 5. This chapter also presents the problems of Boolean satisfiability and pseudo-Boolean optimization, which compose the foundation of the algorithms applied in Chapter 7.

3.1 Representations of Boolean Functions

We assume $x_1, x_2, ..., x_n$ are n Boolean variables with values $\mathbb{B} = \{0, 1\}$, then a **Boolean Function** with n variables $f(x_1, ...x_n)$ is a mapping from the n-dimensional Boolean domain $\mathbb{B}^n = \{(x_1, x_2, ..., x_n) | x_i \in \mathbb{B})\}$ to the codomain $\mathbb{B}$ (denoted as: $f : \mathbb{B}^n \mapsto \mathbb{B}$), which can be represented by a *Boolean formula*. n is the *degree* or the *order* of the Boolean function f. For the preciseness, the n-variable Boolean functions are also defined by a conforming set of rules [Brown90]:

1. For any element $b \in \mathbb{B}$, the *constant function*, denoted by

$$f(x_1, x_2, x_3..., x_n) = b \qquad \forall (x_1, x_2, ..., x_n) \in \mathbb{B}^n,$$

is a n-variable Boolean function.

2. For any symbol $x_i \in \{x_1, x_2, ..., x_n\}$, the *projection-function*, defined, for any specified $i \in \{1, 2, ..., n\}$, by

$$f(x_1, x_2, x_3 ..., x_n) = x_i \qquad \forall (x_1, x_2, ..., x_n) \in \mathbb{B}^n,$$

is a n-variable Boolean function.

3. If g and h are n-variable Boolean functions, then so are the functions $g \vee h$, $g \wedge h$ and $\bar{g}$ for all $(x_1, x_2, ..., x_n) \in \mathbb{B}^n$, expressed as:

- $(g \vee h)(x_1, x_2, ..., x_n) = g(x_1, x_2, ..., x_n) \vee h(x_1, x_2, ..., x_n)$
- $(g \wedge h)(x_1, x_2, ..., x_n) = g(x_1, x_2, ..., x_n) \wedge h(x_1, x_2, ..., x_n)$
- $(\bar{g})(x_1, x_2, ..., x_n) = \overline{g(x_1, x_2, ..., x_n)}$

4. Any function derived from applying the above three rules for a finite number of times is also a Boolean function.

Two Boolean functions, f and g are *equivalent* if

$$f(x_1, x_2, x_3 ..., x_n) = g(x_1, x_2, x_3, ..., x_n)$$

for every ordered n-vector $(x_1, x_2, x_3, ..., x_n) \in \mathbb{B}^n$.

For a Boolean function f, the set of Boolean assignments in $\mathbb{B}^n$ mapping f to true is named the *on-set* (or *satisfying set*) of f, while the set of assignments mapping f to false is the *off-set* of f. If the on- and off-set of f partition the entire assignment space $\mathbb{B}^n$, then f is *completely specified*, otherwise it is called *incompletely* or *partially specified.*

when a variable x_i of function f is replaced by a constant $b \in \mathbb{B}$, the remaining function is called a *co-factor* (or a *restriction*) of f, denoted as: $f|_{x_i=b}$. For any variable $x_1, x_2, ..., x_n$, a co-factor of x_i is expressed as:

$$f|_{x_i=b}(x_1, x_2, x_3 ..., x_n) = f(x_1, ..., x_{i-1}, b, x_{i+1} ..., x_n)$$

Using this notation, the Shannon expansion [Shann38] of a function f around variable x_i is given by

$$f = (x_i \wedge f|_{x_i=1}) \vee (\bar{x}_i \wedge f|_{x_i=0}) \tag{3.1}$$

Some functions may not depend on all variables. The *dependency set* of a function f, denoted I_f , contains those variables on which the function depends, i.e. $I_f = \{i |\ f|_{x_i=0} \neq$

$f|_{x_i=1}\}$ The function which for all variable assignments yields 1 (0) is denoted **1** (**0**). Then, the dependency sets of Boolean functions **1** and **0** are the empty sets.

Boolean functions can also be represented by tables, maps, cubes or graphs. The properties and size of different representations will affect the effort and efficiency of the manipulation of Boolean functions. The representations used in the thesis is the focus of the reminder.

3.1.1 Representation as Formula

In a Boolean formula, a *literal* is a Boolean variable or its negation. A conjunction of literals is a product term, a disjunction of literals is a sum. If the conjunction (disjunction) contains all variables of the function, it is called *minterm* (*maxterm*). A Boolean formula h is in Disjunctive Normal Form (DNF), if it is a disjunction of conjunctions and individual literals. A DNF of h is expressed as:

$$h = \bigvee_{j=1}^{l} \left(\bigwedge_{i=1}^{k_j} x_i^j \right) \tag{3.2}$$

where x_i^j refers to literals and k_j is the number of literals in each conjunction. i and j are the indexes of literals and conjunctions respectively.

Similarly, a Conjunctive Normal Form (CNF) is a conjunction of disjunctions and literals and can be written as:

$$h = \bigwedge_{j=1}^{l} \left(\bigvee_{i=1}^{k_j} x_i^j \right) \tag{3.3}$$

where k_j is the number of literals in each disjunction. j is the index of disjunctions.

In general, there could be an infinite number of Boolean formulas representing one Boolean function. To identify whether two Boolean representations are isomorphic requires solutions to NP-Complete or coNP-Complete problems [Garey79]. Boolean formulas are restricted to be an unique representation of Boolean functions. Such a Boolean formula under constraints is in a *canonical form*, which enables the Boolean function comparison through the comparison of their representations.

3.1.2 Table-, Map- and Cube-Based Representation

An n-variable Boolean function can be explicitly represented by a truth table, in which the function values (either 0 or 1) are listed in 2^n rows according to variable assignments.

The truth-table is a canonical representation of the Boolean function. However, the size of the table grows exponentially w.r.t. the number of variables.

A Karnaugh map [Karna53] is a two-dimensional representation of a Boolean function.

In the cube-based representation, the on-set of an n-variable Boolean function f is represented as a collection of vertices on an n-dimensional hypercube with 2^n vertices and $n \cdot 2^{n-1}$ edges. Each single vertex in the n-dimensional space is a minterm or product with n literals of the function f. A *cube* is a conjunction of literals. A set of cubes that represents f is called a *cover* of f, i.e. a cover of f contains all vertices of the on-set but no vertices of its off-set. The cube-based representation is an efficient data structure for the manipulation and minimization of Boolean functions.

3.1.3 Graph-Based Representation

Binary decision diagrams form the most important graph-based representation of Boolean functions and have been first presented in [Lee59] for a switching circuit implementation. The concept has later been extended to multi-valued decision diagrams or algebraic decision diagrams (cf. Section 3.2) for partially specified functions or more sophisticated modeling applications.

Binary Decision Diagrams

For manipulating Boolean functions, the notations and concepts of Binary decision diagrams (BDDs) are introduced by Lee [Lee59] and further investigated by Akers [Akers78]. BDDs become an essential means for digital circuit design, synthesis and verification after Bryant puts further restrictions on the ordering of decision variables in the vertices. The Reduced Ordered Binary Decision Diagrams (ROBDDs) presented in [Bryan86] is a canonical form, therefore improves the efficiency of algorithm development and reduces the effort for representation manipulation.

In [Bryan86], Bryant defines a BDD as a rooted, directed and acyclic graph G with two types of vertices (V: the set of vertex): A non-terminal vertex or leaf $v \in V$ has a variable index $index(v) \in \{1, ..., n\}$, and two children (successors) $low(v)$, $high(v) \in V$. A terminal vertex $v \in V$ has a value $value(v) \in \{0, 1\}$.

Furthermore, for all non-terminal vertex v, if it has a non-terminal child, then the index of the child is larger than the index of the vertex, i.e. if $low(v)$ ($high(v)$) is non-terminal, then $index(v) < index(low(v))$ ($index(v) < index(high(v))$). This restriction forces

all variable indices on every path (from the BDD root to a terminal) to have a rising order, thus guarantees the acyclicity of the graph.

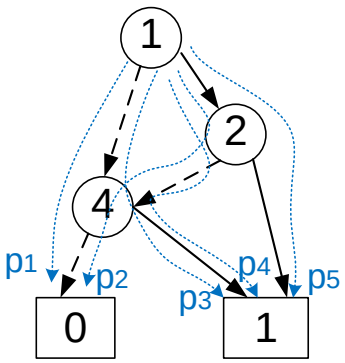
Bryant builds up the correspondence between a graph G with root vertex v and a Boolean function f_v recursively as follows [Bryan86]:

1. If v is a terminal vertex:
   - If $value(v) = 1$, then $f_v = \mathbf{1}$
   - If $value(v) = 0$, then $f_v = \mathbf{0}$
2. If v is a non-terminal vertex with $index(v) = i$, then f_v is the function

$$f_v(x_1, ..., x_n) = \bar{x}_i \wedge f_{low(v)}(x1, ..., x_n) \vee x_i \wedge f_{high(v)}(x_1, ..., x_n). \tag{3.4}$$

Thus, a non-terminal vertex v with $index(v) = i$ corresponds to Shannon expansion of f_v w.r.t. variable x_i (Equ. 3.1). The *high* and *low* children of v are the co-factors of f_v.

Fig. 3.1 provides an example of the BDD w.r.t the Boolean function $f_v = x_1 \wedge x_2 \vee x_4$. The dash line of a vertex v points to $low(v)$ and the solid line links $high(v)$. In graph G, every path can be described by a conjunction of literals. The value of a terminal at a path end is the function value with variable assignments defined by the path. Every vertex in the graph is contained in at least one path, i.e. no part of the graph is "unreachable." In Fig. 3.1, function values and the corresponding Boolean expression of the five paths are listed with the exemplar BDD (the $\wedge$ symbols are omitted from expressions).



expression	value
p1: $\overline{x_1}\overline{x_4}$	0
p2: $x_1\overline{x_2}\overline{x_4}$	0
p3: $\overline{x_1}x_4$	1
p4: $x_1\overline{x_2}x_4$	1
p5: x_1x_2	1

▲ **Figure 3.1** — A BDD constructed from a Boolean function

Function graphs G and G' are isomorphic if there exists a one-to-one function σ mapping from the vertices of G onto the vertices of G' such that for any vertex v if

$\sigma(v) = v'$, then either both v and v' are terminals with $value(v) = value(v')$, or both v and v' are non-terminal vertices with $index(v) = index(v')$, $\sigma(low(v)) = low(v')$, and $\sigma(high(v)) = high(v')$.

A BDD G is *reduced* if it contains neither a vertex v with identical children $low(v) = high(v)$, nor distinct vertices v and v' such that the sub-graphs rooted by v and v' are isomorphic. Such BDDs are called Reduced Ordered Binary Decision Diagrams (ROBDDs) and form a canonical representation of Boolean functions [Bryan86]. The equivalence checking by the comparison of two ROBDDs with identical variable order requires a linear time w.r.t. the graph size. The size of an ROBDD depends on the variable order. The search for an optimal variable order minimizing the graph size is an coNP-complete problem [Bryan86]. There exist functions whose ROBDDs grow exponentially in the size independent of the variable order e.g. the output function of an integer multiplier [Bryan86].

3.2 Algebraic Decision Diagram

An Algebraic Decision Diagram (ADD) is an extend Binary Decision Diagram (BDD) [Bryan86] which allows to have multiple terminal notes with a set of constant values from an arbitrary finite domain [Bahar93]. An ADD is a directed acyclic graph $G = (V \cup \Phi \cup T, E)$ constructed by a set of internal nodes V, a set of functions nodes Φ, a set of terminals T and a set of edges E. The ADD maps a set of Boolean functions F to a set of constants S, expressed as $F : \{0,1\}^n \mapsto S$. Each internal node $v \in V$ labeled as $l(v)$ has two successors. The two successors of node v are labeled as $low(v)$ and $high(v)$, respectively. The *zero-edge* (dash line) points to $low(v)$ and the *one-edge* (solid line) connects $high(v)$ from node v . The label identifies a variable on which the set of Boolean functions F depend. A function node $\phi \in \Phi$ has one successor and no predecessor. The function nodes are mapped one-to-one to the functions F. Each terminal node $t \in T$ is labeled with an element $s(t) \in S$. Terminal nodes have no successors. The function of a terminal node t is a constant function $s(t)$ and independent from all Boolean variables. Variables in the ADD are in a increasing order. If v_j is in a sub-graph under v_i ($v_i, v_j \in V$), then $l(v_i) < l(v_j)$. An ADD is a representation of a set of Boolean functions, defined as:

1. The constant function $s(t)$ (i.e. the function of a terminal node t) is interpreted as an element of a Boolean algebra with the size larger than or equal to $|S|$.

2. The function of a node $v \in V$ is the Shannon expansion (Equ. 3.1):

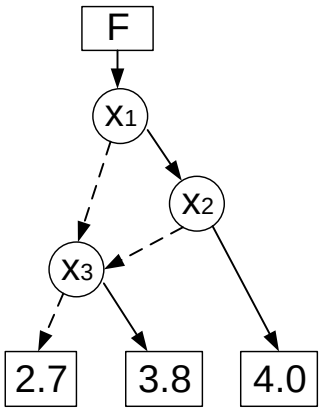
$$f_v = l(v) \wedge f_{high(v)} \vee \overline{l(v)} \wedge f_{low(v)}$$

where $f_{high(v)}$ and $f_{low(v)}$ are the functions of the two successor of node v.

3. The function of $\phi \in \Phi$ is the function of its only successor.

As shown in Fig. 3.2, a table and an ADD represent the mapping from a set of Boolean functions $F = \{\bar{x_1}\bar{x_3},\ \bar{x_1}x_3,\ x_1\bar{x_2}x_3,\ x_1x_2\}$ to a set of real values $S = \{2.7,\ 3.8,\ 4.0\}$.

x1	x2	x3	s(t)
0	-	0	2.7
0	-	1	3.8
1	0	1	3.8
1	1	-	4.0



▲ **Figure 3.2** — An ADD constructed from a mapping table

3.3 Boolean Satisfiability

Boolean SATisfiability (SAT) is a decision problem (i.e. requiring yes/ no answer) that checks whether an assignment of the variables exists satisfying a given Boolean formula. More precisely, a SAT problem can be presented as:
For a given Boolean formula representing a Boolean function $f(x_1, x_2, ..., x_n) : \mathbb{B}^n \mapsto \mathbb{B}$, we search for an assignment $(a_1, a_2, ..., a_n) \in \mathbb{B}^n$ which evaluates $f(a_1, a_2, ..., a_n)$ to *true*.
A Boolean formula conducting the satisfiability checking is called a *SAT instance*. The assignment of Boolean variables yielding the SAT instance to true is a *satisfying assignment*. If a satisfying assignment exists for a SAT instance, the instance is *satisfiable*. If no such satisfying assignment exists, the SAT instance is *unsatisfiable*. When a satisfiable instance always yields true regardless the assignment of variables, it is

called a *tautology*. Determining whether a Boolean formula is a tautology can be easily transformed into a SAT problem. When we negate the formula for tautology checking, the problem is converted to check the unsatisfiability of the negated formula. If the negated formula is satisfiable, the original formula is not a tautology.

The SAT problem is proven to be NP-complete [Cook71]. Thus, all known algorithms face an exponential complexity w.r.t. the number of variables in the SAT instance either in time or in the storage space. The majority of SAT solving algorithms process the instances in CNF (cf. Section 3.1.1). The disjunctions in the formula are named *clauses*.

Davis-Putnam (DP) algorithm [Davis60] is the first algorithm developed for SAT problem-solving. The DP algorithm iteratively selects variables to perform transformations, that cover the original SAT instance to an equivalent formula. The iteration continues until an empty clause is derived from the transformation or all variables are considered. When the algorithm terminates due to the occurrence of the empty clause, the original SAT instant is unsatisfiable. Otherwise, the represented function is satisfiable.

The Davis-Putnam-Logemann-Loveland (DLL or DPLL) algorithm [Davis62] is a search-based procedure and the basis of most modern SAT solvers. The algorithm iteratively sets the value to the variables until a satisfying assignment or conflict is found. When a conflicting assignment occurs, the algorithm performs the backtrack and examines a different assignment.

A systematic review of SAT solvers can be found in [Biere09].

3.4 Pseudo Boolean Optimization

3.4.1 Pseudo Boolean Function

Let n denote a positive integer. A *pseudo-Boolean function* is a mapping from a set of binary vectors $\mathbb{B}^n = \{0,1\}^n$ to a set of reals $\mathbb{R}$, i.e. a real value is associated to a finite number of 0-1 variables. A pseudo-Boolean function $f(x_1, x_2, ..., x_n) : \mathbb{B}^n \mapsto \mathbb{R}$ can be uniquely expressed as a multilinear polynomial:

$$f(x_1, x_2, ..., x_n) = c_0 + \sum_{k=1}^{m} c_k \prod_{i \in A_k} x_i \tag{3.5}$$

where $c_0, c_1, ..., c_m$ are the real coefficients, and $A_1, A_2, ..., A_m$ are nonempty subsets of $N = \{1, 2, ..., n\}$ [Boros02].

3.4.2 Pseudo-Boolean Satisfiability and Optimization

A *pseudo-Boolean constraint* (i.e. an extended clause) is a linear 0-1 inequality: $c_1 \cdot L_1 + c_2 \cdot L_2 + ... + c_n \cdot L_n \geq d$, where $c_i, d \in \mathbb{Z}$ are integer coefficients and L_i are Boolean literals. Index $i \in N = \{1, 2, ..., n\}$. The pseudo-Boolean constraint is satisfiable if $(\sum_{i \in N} |c_i|) \geq d$. If all integer coefficients (c_i and d) are equal to 1, the peseudo-Boolean constraint $(\sum_{i \in N} L_i) \geq 1$ (can also be represented as disjuctions of literals $\bigwedge_{i \in N} L_i$) becomes a classical clause.

Pseudo-Boolean Optimization (PBO) is a procedure searching the solution that satisfy all pseudo-Boolean constraints and optimize the *objective function.* An objective function is a pseudo-Boolean function, denoted as $g(x_1, x_2, ..., x_n) : \mathbb{B}^n \mapsto \mathbb{R}$ for maximization (or minimization). The Boolean function $f(x_1, x_2, ..., x_n) : \mathbb{B}^n \mapsto \mathbb{B}$ is formulated as a conjunction of pseudo-Boolean constraints, which require being satisfied during the optimization. A PBO problem is to explore an optimal Boolean vector $V = \{v_1, v_2, ..., v_n\} \in \mathbb{B}^n$, to obtain $f(V) = 1$ and $\forall_{Y \in \mathbb{B}^n}\{[f(Y) = 1] \Longrightarrow [g(Y) \leq g(V)]\}$ (in case of maximization) Y is an arbitrary Boolean vector. The pseudo-Boolean optimization solvers have been extensively published [Hamme86,Boros02] and become commonly supported by the commercial tools.

CHAPTER 4

State of the Art

Aggressive technology scaling increases the possibility of manufacturing defects, the susceptibility to aging and it aggravates process variations, thus threatens the system reliability and diminishes the device functioning lifetime. To overcome these challenges of the hardware reliability, the work at hand aims to elevate the detectability of fabricating imperfections and marginal devices and improve the system online observability for an effective stress reduction and accurate failure prediction.

This chapter first gives a brief overview of the state-of-the-art test schemes for small delay fault detection, followed by a short discussion of monitoring approaches for aging degradation. To avoid pessimistic static design margins, aging monitoring and adaptation approaches are often applied in fields. As a result, the On-line built-in self-test techniques, degradation measurement methods and monitor on-chip placement are the focus of our following review.

4.1 Small Delay Fault Detection

Low-power and near-threshold operation reduces the noise immunity and makes the circuit highly susceptive to small delay deviations. Although small delay faults cause only a limited variation of path delay and such *hidden delay faults* [Helle14] (cf. Section 2.1.2) even do not violate the nominal operation timing, the marginal

hardware may pose a reliability risk later due to aging and degradation in-field [Kim10], when for instance oxide defects grow or the delay of a resistive via magnifies due to electromigration. Thus, an early life failure (ELF) or a functional timing failure may occur after product shipment [Yan04].

Consequently, small delay faults require thorough test schemes to achieve a high test quality and efficiency (i.e. to obtain a high fault coverage with low test cost). Because of the small magnitude of deviated delay, SDFs are commonly screened via the sensitizable long paths through the fault site or detected by Burn-in or faster-than-at-speed tests.

4.1.1 Pattern Generation and Arrangement

Various test pattern generation, selection and grouping methods are published targeting SDF detection.

Under the guidance of the circuit timing information, timing-aware ATPG methods target SDFs through paths with minimal slack to gain a better observability of the fault effects. However, detecting SDF through long path often leads to a significant CPU runtime and pattern count. [Lin06] proposes a weighted random scheme for pattern generation, which activates various path branches for fault effect propagation to achieve an improved path coverage. To facilitate the test efficiency and trade-off the fault detectability against test pattern count, the authors also introduce a new fault-dropping criterion: Dropping based on Slack Margin (DSM). During simulation, faults are dropped from the target list when they are detected through the paths satisfying the slack requirement.

When the circuit timing information is unavailable, the probability that an SDF is detected through a long path can be increased with n-detect ATPG, which generate the test patterns attempting to screen the fault n times via different paths through the fault site. n-detect ATPG is an effective method for SDF detection and requires much less CPU runtime in comparison with timing-aware ATPG since the computational expensive analysis of path length can be avoided. However, the usage of such n-detect ATPG scheme is significantly limited by the high pattern count for a large n number. [Huang06] optimizes the traditional n-detect ATPG method by modeling the pattern generation process as a minimum covering problem. A minimum number of patterns is obtained by providing Integer linear programming (ILP). A heuristic method

with less computation complexity is also proposed as an alternative for sub-optimal solutions.

[Ahmed06] proposes a pattern selection scheme which picks the patterns sensitizing a higher percentage of long paths and masking all short paths from a n-detect ATPG pattern set, to enable the pattern generation with timing-unaware ATPG tools for high-quality SDF detection.

[Ahmed09] presents a pattern grouping technique cooperating with faster-than-at-speed tests (cf. Section 4.2). The technique sorts the patterns of a given a test set w.r.t its sensitized path length and groups them into several subsets with similar path delay. The path length in each pattern group as well as IR-drop induced delay determine the test frequency of the subset, which can achieve a good observability of SDFs and currently maintains a positive slack for fault-free paths to avoid X-masking (cf. Section 4.2).

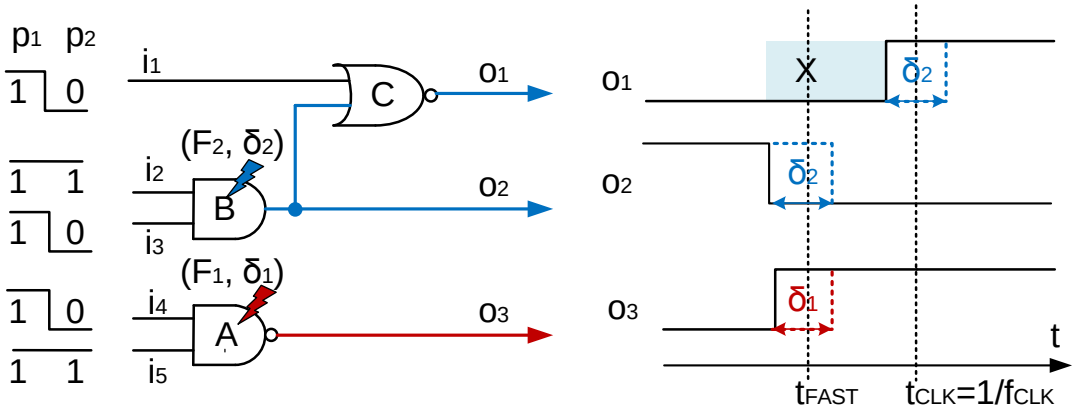
4.1.2 Burn-in Test

Burn-in stresses the hardware, e.g. by increased voltage or temperature, to accelerate defect degradation and to detect imperfections and ”infant mortality” (ELFs). The effectiveness of burn-in test with increased voltage has been reduced exponentially due to voltage scaling [Nigh00]. Burn-in may also reduce the reliability of the devices [Volle99]. Additionally, burn-in is very expensive due to the required equipment and high test time.

4.2 Faster-than-at-speed Test

Faster-than-at-Speed test (FAST) applies test patterns at frequencies above the nominal speed to reduce the positive slack of the path for fault effect propagation and screen small delay faults or hidden delay faults [Mao90, Yan03]. Fig. 4.1 shows the fundamental idea of FAST through a simple example.

We assume two small delay faults (F_1, δ_1) and (F_2, δ_2) are located in gate A and B respectively (shown in Fig. 4.1). δ_1 and δ_2 are the fault magnitude of F_1 and F_2, i.e. the deviated delay w.r.t. the circuit timing specification. By applying the test pattern pair (p_1, p_2), fault F_2 propagates to output o_1 and o_2, while fault F_1 propagates only to output o_3. If the delay test is performed at nominal operation frequency f_{CLK}, due to the propagation path slack and the fault magnitude, only the fault effect of F_2 can



▲ **Figure 4.1** — The faster-than-at-speed test for hidden delay faults

be observed at output o_1. F_1 located on the short path is hidden by the path slack at f_{CLK}. If the test patterns are provided at a higher frequency f_{FAST}, the faulty behavior of F_1 can be observed at output o_3 with the early sampling time $T_{FAST} = 1/f_{FAST}$. However, the signal propagating from a longer path to o_1 has not reached its stable state. Therefore the unstable signal value has to be presented as unknown (X-value).

As shown Fig. 4.1, FAST improves the detectability of small delay faults, i.e. previously undetectable faults become observable at a higher testing speed. The high test frequency of FAST induces a significant IR-drop into the circuit, which was targeted and solved by the IR-drop analysis and pattern reordering in [Ahmed09].

FAST can be applied externally with automatic test equipment (ATE). The ATEs supporting high-speed tests are often very costly since they require dedicated designs to overcome the electrical effects (e.g. tester skews, parasitic capacitances) affecting each measurement [Yan03]. An alternative solution is to support FAST with a Built-In-Self-Test (BIST) structure. Programmable on-chip clock generators [Pei10, Tayad08] are proposed for this purpose. The first FAST-BIST is introduced in [Helle14]. However, both of the FAST frameworks face the same difficulty: the unknown values appearing at long path ends (in Fig. 4.1).

In the remainder of the section, we discuss the state-of-the-art techniques overcoming the challenges of FAST.

4.2.1 On-chip Infrastructure for Test Evaluation

Both the ATE and BIST frameworks require test response compaction structures on-chip. However, X-values in test responses during FAST (in Fig. 4.1) corrupt the compaction results. The scheme in [Amode05] directly blocks all X-values at their appearing circuit outputs but induces a significant hardware and control overhead. More often the compaction hardware plays the role of handling X-values. Although many X-handling schemes [Mitra02, Narus03, Rajsk03, Sharm05, Tang04, Touba07, Wohl07] have been proposed for BIST at nominal speed, those standard solutions can not be directly implemented for FAST due to the following reasons. The distribution of X-values depends on the circuit topology i.e. the distribution of long paths. The simulation results published in [Kampm16] show that the number of X-values grows dramatically with the increase of test frequency. The varying X-rates in different test frequency challenges several compaction schemes [Garg08, Mitra04, Lien14] coping with only a constant X-rate. As a result, such compactor has to be designed for the worst-case X-values, leading to a high overhead and decreased fault coverage. The extremely high number of X-values due to aggressive over-clocking invalidates the X-tolerant compaction schemes [Rajsk03, Sharm05, Rajsk06, Wohl07] tolerating only certain moderate X-rates.

To cope with the potentially unbounded X-rates, [Singh10] has proposed a multiplexer-based outputs compression scheme to avoid the test response compaction. Since only a very small portion of response bits are observed for the fault detection, the method requires an extra amount of test patterns to achieve the target fault coverage. Additionally, the compression reduces the possibility of observing unmodeled faults.

[Helle14] introduces a programmable X-handling compaction scheme adapted to each test frequency. It combines an X-canceling MISR in [Touba07] with a small on-chip storage for intermediate signatures. The intermediate signatures are later processed off-chip to observe the targeted hidden delay faults.

[Czysz10] provides an X-masking approach based on a flexible scan-chain selection. The masks for scan-chain selections are generated at each cycle by exploiting the correlation between X-states.

[Kampm16] presents a test response compaction technique based on scan flip-flop clustering. The proposed method puts the scan flip-flops with similar X-behavior into the same scan chain by analyzing both the circuit topological and the simulation-based X-profiles. Clustering and blocking only a few scan-chains per test pattern can

dramatically reduce the X-values appearing in the compactor circuit, while a high detection efficiency of small delay faults can still be achieved.

4.2.2 Frequency Selection for FASTs

For a high fault coverage, a high maximum test frequency f_{max} and many different frequencies are often required. Typically f_{max} is set up to 3 times of the nominal clock f_{nom} [Amode05, Ahmed09, Lee08]. When selecting test frequencies, there are mainly two strategies: The first strategy simply selects the frequencies evenly distributed in the range $[f_{nom}, f_{max}]$ (e.g. four equal-distant frequencies $1.5 \cdot f_{nom}$, $2 \cdot f_{nom}$, $2.5 \cdot f_{nom}$, and $3 \cdot f_{nom} = f_{max}$ are chosen.). With the predefined fixed frequencies the first strategy tries to adjust the pattern set and test control scheme [Liu17] appropriately for a high detection efficiency. In contrary, the second strategy provides a given set of test patterns as the precondition and searches for a minimum set of frequencies to reach the maximum fault coverage.

[Helle14] uses a greedy heuristic algorithm to select a minimum number frequencies to cover all the target faults. With the selected test frequencies the fault set is partitioned into groups such that all the faults in one group can be tested by the same frequency.

The paper [Kampm15] proposes a new hybrid approach combining both strategies for FAST frequency selection. It firstly performs the tests with a basic transition fault pattern set at preselected frequencies to cover most of the target faults and identify the remaining hard-to-detect ones. Then extra timing-aware patterns are generated for the hard-to-detect faults and provide to circuits together with the basic test set. Finally, an optimal set of frequencies are selected based on the extended test set and guarantees a maximum fault coverage.

4.3 On-line Built-in Self-Test for Aging Monitoring

On-line built-in self-test techniques [Li08, Inoue08, Kocht10, Sato09, Helle14] can be applied for circuit aging prediction.

The Concurrent Autonomous chip self-test using Stored test Patterns (CASP) in [Li08] measures the delay of target paths on one or more cores in a multi-core system periodically by performing on-line testing with thoroughly generated pre-stored test patterns,

without interrupting the functional operation of the entire system. The Virtualization-Assisted concurrent autonomous Self-Test (VAST) in [Inoue08] provides an on-line scheduling approach considering core unavailability to minimize the functional interruption and keep the measurement frequency at a high level.

In [Kocht10], partially specified patterns are provided to construct the concurrent BIST circuitry. Due to the large portion of don't care bits in the pattern set targeting the most critical faults, the synthesized concurrent self-test hardware results in a much lower hardware overhead and test latency than previous techniques.

[Sato09] performs the on-line delay testing with the BIST architecture at the power-on/off time of a system for degradation measurement. The monitoring inaccuracy induced by temperature deviation or supply voltage noise is adjusted by utilizing the embedded ring oscillators.

4.4 Aging Monitors

If an individual hardware is developed and integrated on-chip for aging monitoring, we call this logic component an *aging monitor*. In this section, we review various designs of state-of-the-art aging monitors. We study the working principles of monitors and compare their advantages and disadvantages from many different aspects. By analyzing the observables of the monitors (i.e. the aging indicators), we notice up-to-date monitors measure only the degradation effects (such as deviated delay, working current). To predict reliability risks as early as possible and pro-activate the aging mitigation techniques, we introduce a stress screening monitor in Chapter 5. After examining the monitoring overhead (cf. Section 4.5), we provide a novel monitor placement approach to reduce the prediction latency as well as the monitoring cost (cf. Chapter 6). Furthermore, the delay increase can be induced not only by aging but also by small delay defects. We present a method in Chapter 7 to reuse state-of-the-art delay monitors for small delay fault test, avoiding the complex X-handling structures in test compactors or evaluators during faster-than-at-speed-test.

Monitors sensing the degradation effects in the circuit can be classified in many ways based on:

- whether the monitor is sensitive to aging degradation: self-stressed [Carls07, Kim08] or aging resilient [Vazqu10, Saliv15];

- what is the degradation indicator for measurement: working current [Hsieh07, Wang11], threshold voltage [Carls07], frequency [Kim08], signal slew rate [Ghosh09], delay or slack [Agarw08, Saliv15];
- whether the monitor is prognostic [Agarw08, Saliv15] or applying error detecting and correcting schemes [Das09, Sato07];
- whether the monitor takes the circuit workload into account for aging estimation: standalone [Tscha09, Wang12] (cf. Section 4.4.1) or in-situ monitors [Agarw08, Das09] (cf. Section 4.4.2);
- on which abstraction level the monitor is proposed: micro-architecture level [Blome07, Pei09] or device level [Carls07, Kim08, Saliv15].

The circuit workload plays an important role in aging degradation and will significantly influence the monitor placement and measurement accuracy. Therefore, we distinguish aging monitors by checking whether the operation workload or its effects are under observation. Then, we categorize them into standalone and in-situ monitors.

4.4.1 Standalone Monitors

We call the monitors that read in neither directly the workload nor any aging indicators depending on the workload from the monitored circuit *standalone monitors*. Instead, this type of monitors often includes a stressed cell which is overcritically designed or overstressed, to guarantee that the stressed cell degrades faster than the monitored circuit under actual operation [Carls07, Kim08].

The tunable replica circuit [Tscha09] is designed and calibrated to track local critical path delays of each pipeline stage of a processor. Representative Critical Reliability Paths (RCRPs) [Wang12] as a individual measurement unit estimates the aging degradation of the design in the field. This implementation improves the accuracy of aging estimation, but brings extra hardware cost and sampling control signals. Other effects, such as process variations, crosstalk and power supply noise, haven't been taken into account, which may also cause different delays in RCRPs and the functional circuit.

Standalone monitors are non-intrusive and with better hardware efficiency. However, their effectiveness bases on the assumption that the mission logic does not degrade faster than the stressed cell. Unfortunately this cannot be guaranteed since degradation strongly depends on transistor workload and working conditions (temperature or supply

voltage). Due to large workload differences between applications [Chand14, Baran15], the self-stressed standalone monitors have to be pessimistic and underestimate the real device lifetime.

4.4.2 In-situ Monitors

In-situ monitors measure a performance indicator directly in the functional part of the circuit. Razor flip-flops [Das09] are augmented with shadow latches synchronized by a delayed clock. By comparing the values in original registers and shadow latches, an aging induced timing failure can be detected. [Sato07] simplifies the Razor flip-flop design by eliminating the delayed clock and adds a delay buffer before the signal line of the shadow register. The signals from the functional circuit are *pre*sampled by the shadow flip-flops to generate alerts for the imminent failure.

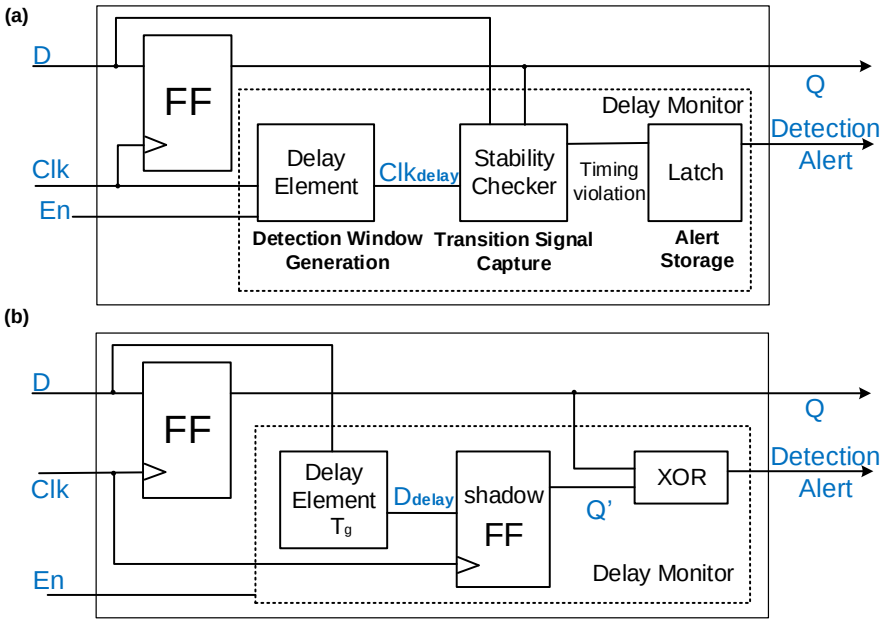
[Blome07] constructs an online path slack measurement unit named wear-out detection unit (WDU) by modeling the timing degradation of oxide breakdown at microarchitectural level. Cooperating with a proposed statistical analysis, the WDU is self-calibrating, generic to various micro-architectures, and capable of identification of significant signal latencies amount those abnormal samples due to the noise or temperature fluctuations. [Pei09] builds an on-chip path delay measurement (OCDM) component, which can transform the delay of the candidate paths into a sequence of digital values, stored in the flip-flop of the Vernier Delay Line (VDL) chain.

Delay detecting flip-flops [Agarw08, Dadgo10, Vazqu10, Saliv15] detect the degradation progress when a transition of the observed signal violates the predefined detection window (guard band) before the clock sampling edge. Scout flip-flop [Semia14] generates a tolerance window for late transitions after the clock. The late transitions trigger the scout flip-flop to predict as well as correct the failure. Low-cost transition detectors [Omana13] are based on transmission gates and allow to mask late transitions by shifting the clock.

A delay detecting flip-flop is a standard flip-flop extended with a delay monitor [Agarw07]. The monitors can detect a delay increase by sensing the transitions during a predefined detection window and are often placed at the end of critical paths (cf. Section 4.5.2) or selected intermediate positions of combinational circuits (cf. Chapter 6). Two delay monitor structures are proposed in [Agarw07].

A delay monitor (Fig. 4.2 (a)) consists of a delay element, stability checker and latch:

The delay element shifts the clock to generate a reference signal (CLK_{delay} in Fig. 4.3 (a)). The time period T_g between the rising edges of the clock (CLK) and CLK_{delay} is the guard band. The duration of guard band (T_g) equals the clock period minus the delay (Δ_a) from the delay element. The stability checker measures signal stability during the guard band i.e. the detection window. The design margin (Fig. 4.3 (a)) prevents a false activation due to process variation or early aging degradation. If in the nominal case the signal D at a path endpoint reaches its stable value before the guard band, the path is not critical for operation. On the contrary, if the signal is unstable during the guard band (D'), an alert is generated and stored in the latch.

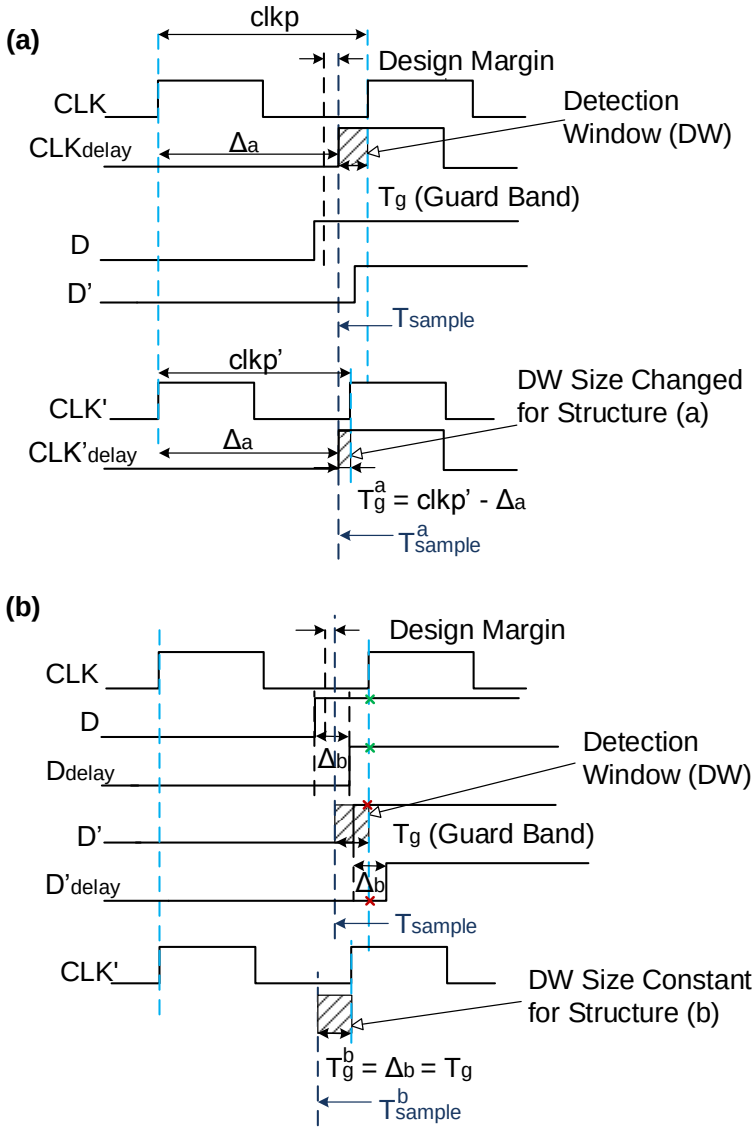


▲ **Figure 4.2** — Structure of a delay detecting flip-flop with the (a) stability checker [Agarw07, Vazqu10] or (b) comparator logic [Agarw07, Saliv15]

An alternative design (Fig. 4.2 (b)) contains a delay element (with delay $\Delta_b = T_g$), shadow register and an XOR gate:

Because of the delay element, the observed signal D is *pre*sampled by the shadow register. The XOR gate comparing the outputs of the original (Q) and shadow register (Q') constructs the detection window with T_g interval before clock rising edge. If an

observed signal D stables before the detection window, the same values are sampled from D and D_{delay} (the green crosses in Fig. 4.3 (b)) and written into both of the registers, indicating a well-functioning circuitry. If the a signal D' changes its value within the detection window, i.e. the transition of signal D'_{delay} (D' delayed by Δ_b) arrives after the clock rising edged, different values are captured in the original and shadow registers (red crosses in Fig. 4.3 (b)) and then compared by the XOR gate. In this case, a timing alert is issued.



▲ **Figure 4.3** — Signal waveform of a delay detecting flip-flop with the: (a) stability checker or (b) comparator logic

To prevent the self-degradation of monitors, the enable signal *En* of the both design structures in Fig. 4.2 is used for periodic monitor activation for aging measurement [Vazqu10]. Signal *En* can be used to disable those monitors with false alerts when applying monitors to the reuse approach during delay test (cf. Chapter 7).

A major point to identify whether a monitor structure can be utilized to our approaches (cf. Chapter 6 and 7) is how the clock period influences the detection window. In structure Fig. 4.2 (a), the delay (Δ_a) added to *CLK* is a constant, i.e. the detection window starts always at time T_{sample}. When the clock period reduces to $clkp'$, the size of detection window (T_a) decreases accordingly. Contrarily, in the structure of Fig. 4.2 (b), a delay Δ_b is added to the data signal of the FF. No matter how the clock frequency changes, the interval between the sampling time of the original and shadow register remains as a constant. For this reason, when integrating the monitors of Structure (a) to the observation points (cf. Chapter 6), the delay (Δ_a) needs to be adjusted accordingly. Monitors of Structure (b) can be directly applied to both monitor replacement and reuse schemes (cf. Chapter 6 and 7).

4.5 Monitor Placement for Reduction of Hardware Overhead

4.5.1 Standalone Monitor Placement

For an accurate aging monitoring, *standalone monitors* (defined in Section 4.4.1) are often placed adjacent to the subject under measurement to share an identical environment which affects aging degradation, such as temperature or supply voltage. Tunable replica circuit [Tscha09] are inserted next to each pipeline stage of a processor. Representative Critical Reliability Paths (RCRPs) [Wang12] are located at the position where the critical reliability paths (i.e. the monitoring target) are under the similar voltage and temperature. Prognostic cells in [Carls07] are placed close to the host circuit to obtain similar environmental conditions and same process parameters.

The monitors can also be applied to analyze the distribution or degradation speed of aging. With such intention, monitors can be evenly distributed on-chip or placed at the "hot-spots" due to spatially nonuniformed power dissipation.

4.5.2 In-situ Monitor Placement

Due to the hardware overhead, it is infeasible to integrate the aging monitors into all flip-flops, thus monitors are often placed at the end of the critical or long paths [Agarw08, Wang07b].

In low power designs, the path lengths are equalized and many paths have lengths close to the critical path. Delay uncertainties caused by process variations, transistor workload or working conditions are hard to predict and compensate in the design phase. The critical path of the design may vary from chip to chip and over time. Thus, not only the nominal critical path but also the near critical ones have to be monitored. A non-enumerative technique [Baba09] and a representative critical-path selection scheme [Firou13] are proposed to reduce the number of paths necessary for observation.

[Gomez15] selects the critical paths for end-path monitor insertion by taking the effects of process variation and aging degradation due to NBTI into account. The critical paths can be identified in an early design phase without layout information for computation of spatial correlation between gates. The size of selected path set can be traded off against the required circuit reliability.

To reduce the hardware overhead [Ingel11] selects a small number of measurement points at circuit primary outputs for wear-out monitor insertion to sens the in-field degradation due to electromigration. Wear-out sensitive interconnects are identified by analyzing the nets switching activity and the fan-outs, and are targeted for degradation monitoring. For a minimum number of monitors observing all wear-out sensitive interconnects the authors formulate the selection method as an Integer Linear Programming problem. The SlackProbe in [Lai13] does not limit monitor placement to the end of paths, but also allows intermediate placement to save hardware cost. However, it does not take path sensitization into account when placing the monitors. Additionally, the control signals of monitors may require global wiring or clock balancing schemes, increasing design complexity and possibly requiring to re-route the target design.

CHAPTER 5

Workload Monitor for Stress Estimation

The growing demand for reliability in safety-critical systems does not only place stringent quality requirements on manufacturing test but also calls for in-field monitoring schemes throughout the whole system lifetime. These monitoring techniques have to detect any form of hardware degradation and prognosticate the imminent failures induced by aging or harsh environmental effects, like elevated temperature and electromagnetic interference.

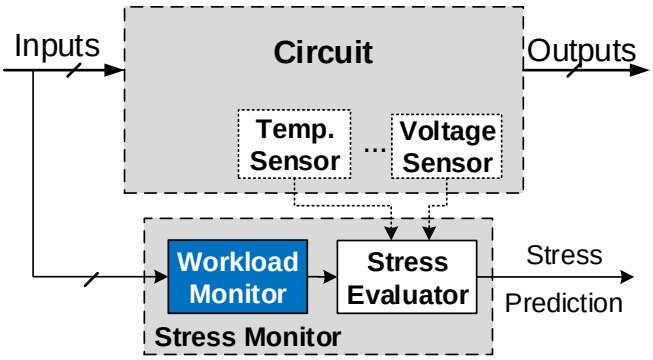
Traditional approaches for reliability monitoring (cf. Section 4.4) measure *degradation effects* such as: the threshold voltage of transistors [Carls07], signal slew rate [Ghosh09], path delay [Agarw08,Dadgo10,Vazqu10], frequency [Kim08], working current [Hsieh07, Wang11], etc. Therefore, they are useful only after the chip starts to deviate from its expected behavior.

To enable prediction of reliability issues at a very early stage and proactive the adoption of appropriate countermeasures preventively, we aim to quantify reliability risk factors *before* any measurable degradation effect takes place.

Various stress factors including the chip temperature, supply voltage, application workload or logic states of the device are the causes of circuit degradation. As an extension

of [Baran13], this chapter provides a stress monitoring scheme, estimating the *stress* experienced by a circuit during its operation. While the physical stress (e.g. temperature or voltage) can be measured directly with proper sensors, workload monitoring remains an open challenge. To address this problem, we present hardware structures which effectively monitor the workload and estimate the stress.

Fig. 5.1 shows the *stress monitor* composed of two main units: a workload monitor and stress evaluator. The *workload monitor* observes the logic state of a circuit and provides an instantaneous stress estimation. The *stress evaluator* aggregates the output of the workload monitor, together with that of any available on-chip sensors. The evaluator provides the cumulative stress suffered by the circuit over a recent period, e.g. by integration or calculation of a moving average. This monitoring approach enables the timely application of any available preventive technique, like the workload balancing or power gating, to make the system more resilient to stress and less prone to degradation.



▲ **Figure 5.1** — The structure of a stress monitor

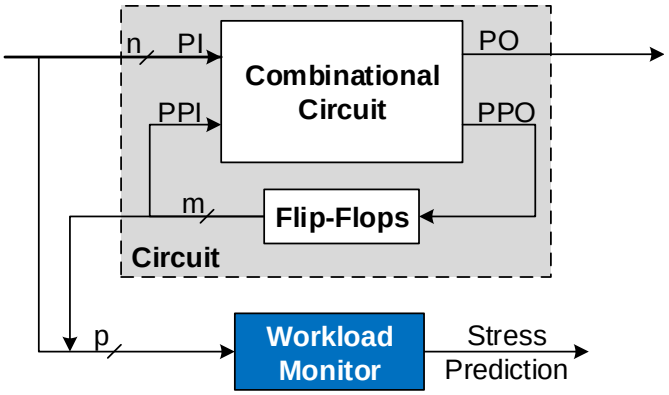
The contributions of the chapter include:

- a fine-grained stress monitoring approach measuring the cause of degradation before real parameter deviation of the circuit;
- a stress approximation method by construction of the algebraic decision diagram;
- a detailed synthesis flow of general-purpose workload monitors which can be employed for various stress mechanisms depending on the states of the circuit;
- a case study for NBTI showing that an accurate estimation of stress can be achieved with a limited hardware cost.

Section 5.1 provides an overview of the workload monitor. In Section 5.2, we detail the construction and synthesis flow for workload monitors. An application of this method to monitoring of NBTI-induced stress in Section 5.3 is followed by the experimental results (Section 5.4).

5.1 Workload Monitor

As shown in Fig. 5.2, there are n primary inputs (PI) and m pseudo-primary inputs (PPI) in the circuit under monitoring. The complete set of PI and PPI is collectively referred to as *observables*. A workload monitor is a combinational circuit which reads in observables and provides an instantaneous approximation of a stress metric. As on-line monitoring of all PIs and PPIs is infeasible, we search for a small subset of observables ($p \leq n + m$) which is highly correlated with the target stress mechanism. The stress approximated from a metric (cf. Section 5.2.1) is constructed in the workload monitor and the state of the observables are used as the index key for searching the corresponding stress approximation (cf. Section 5.2.3).



▲ **Figure 5.2** — Monitoring of workload-induced stress

Since only PI and PPI nodes are observed and buffered, the presented monitoring technique has only minimal impact on the mission logic. As the original design is not modified, the method can be applied for IP cores and fixed macros. Note that we do not consider our monitors as a replacement for on-chip sensors; they complement the capabilities of available sensors to enable timely prediction and avoidance of reliability problems.

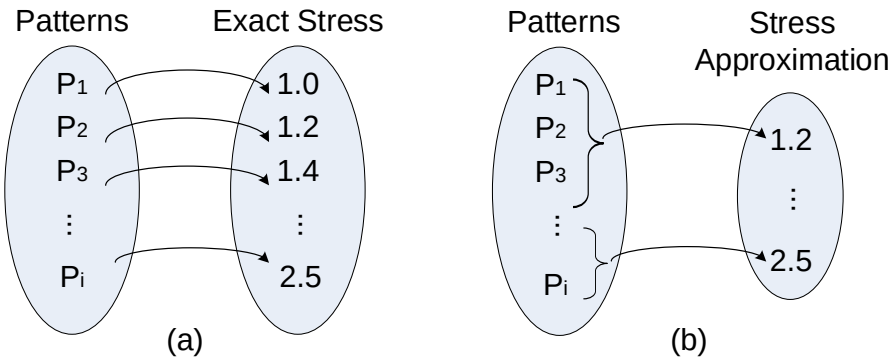
5.2 Construction of Workload Monitors

Our goal is to construct a workload monitor that approximates the stress experienced by a digital hardware component during regular system operation. The functional behavior of a workload monitor is described by an algebraic decision diagram (ADD) [Bahar93] (cf. Section 3.2) called as *approximation diagram*. The approximation diagram generates the stress approximation for each assignment of the selected observables (i.e. monitor inputs) and is constructed in an iterative procedure. In each iteration, the approximation diagram is extended with more observables for a higher approximation accuracy of the stress. The network structure of workload monitors is synthesized from the constructed diagram (i.e. functional behavior) when the accuracy of stress approximation meets the predefined requirements.

5.2.1 Stress Metric and Stress Approximation

The logic state of a digital circuit is defined by the assignment to its PIs and PPIs. The state space is defined as $\{0,1\}^{n+m}$, where n and m are the numbers of circuit's PIs and PPIs, respectively.

We define a stress metric S for a digital circuit as a mapping of its logic state to a real-valued stress measure: $S : \{0,1\}^{n+m} \rightarrow \mathbb{R}$. As shown in Fig. 5.3 (a), the binary states of the circuit are specified by the input patterns and are mapped to the real-valued stress number.



▲ **Figure 5.3** — (a) exact stress estimation; (b) stress approximation

The definition of the stress metric is very general and also applicable to various instantaneous stress mechanisms depended on logic states of the circuit, such as logic

vulnerability to soft errors [Mukhe09] or wear-out due to electromigration [Ingel11] and NBTI. For instance, an NBTI stress metric can be defined as the number of PMOS transistors suffering from NBTI-induced stress on the critical path (cf. Section 5.3).

To estimate the stress with limited cost, the workload monitor implements a stress approximation function $\hat{S} : \{0,1\}^p \to \mathbb{R}$ which maps each assignment of p observables to a real-valued stress approximation, where $p \leq n+m$. Since the observables are a small subset of circuit inputs (PIs and PPIs), some patterns which have identical observable assignments are mapped to the same stress approximation (Fig. 5.3 (b)). Each approximation is an average stress metric over 2^{n+m-p} logic states.

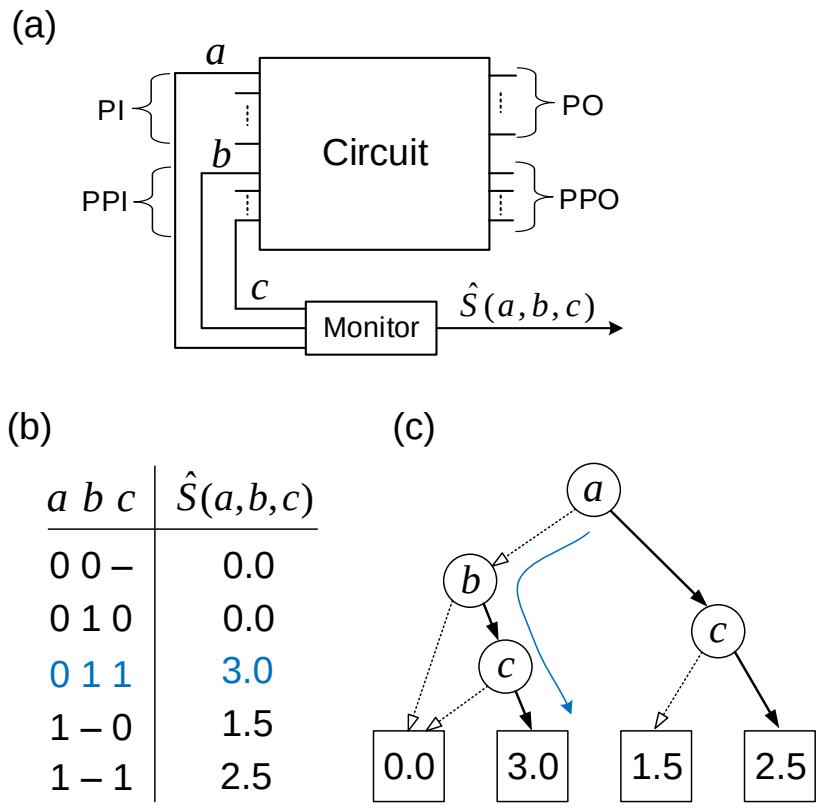
5.2.2 Approximation Diagram

The approximation function $\hat{S} : \{0,1\}^p \to \mathbb{R}$ describes the behavior of a workload monitor and is represented by an approximation diagram. The approximation diagram is an algebraic decision diagram (cf. Section 3.2), denoted as $G(V,E)$, where V is the set of vertices. Let $T \subseteq V$ be the set of terminal nodes which represent the stress approximations. The rest nodes (i.e. internal vertices) refer to input observables. $E \subset V \times V$ is the set of edges connecting vertices and provides the stress approximation for any assignment of monitor inputs. The vertex labeling $l : V \to \texttt{PI} \cup \texttt{PPI} \cup \mathbb{R}$ is defined as a function that maps each vertex $v \in V$ to either an observable or a real-valued stress approximation:

$$l(v) := \begin{cases} \texttt{observable} \in \texttt{PI} \cup \texttt{PPI} & \text{for } v \in V \setminus T, \\ \texttt{stress} \in \mathbb{R} & \text{for } v \in T. \end{cases}$$

Fig. 5.4 (c) gives an example of such approximation diagram, representing the circuit observables (i.e. monitor inputs *a, b, c* in Fig. 5.4 (a)) and the stress approximations (i.e. the monitor output $\hat{S}(a,b,c)$) as the labels of internal vertices and terminals respectively. The approximation function (i.e. the mapping between Boolean assignments of observables and the stress approximation) is shown by the table (Fig. 5.4 (b)).

Each internal vertex $v \in V \setminus T$ has two outgoing edges in E, a 0-edge and a 1-edge, which specify the successor of v according to the assignment of observable $l(v)$: The 0-edge corresponds to the case when the assignment of observable $l(v)$ is 0 (depicted as dash lines in Fig. 5.4 (c)). Similarly, the 1-edge corresponds to the case when $l(v)$ is 1 (solid lines). The successor of a vertex connected with a 0-edge (1-edge) is referred to as 0-successor (1-successor), e.g. the node labeled b is the 0-successor of a.



▲ **Figure 5.4** — (a) an exemplar circuit and its workload monitor; (b) the stress approximation function; (c) the corresponding approximation diagram

The path π from the root vertex v_0 to a terminal node $v_n \in T$ is the sequence of vertices $v_i \in V$ such that $(v_i, v_{i+1}) \in E$ for $i \in \{0, 1, \ldots, n-1\}$:

$$\pi(v_0, v_n) := (v_0, v_1, \ldots, v_n).$$

Multiple vertices in V may share the same label, but the vertices in every path $\pi(v_0, v_n)$ have unique labels. Each path $\pi(v_0, v_n)$ corresponds to a single assignment of observables determined by the edges connecting consecutive vertices, while $l(v_n)$ provides the real-valued stress approximation for this assignment. For instance, in the diagram Fig. 5.4 (c) two vertices are labeled as c, but locate in different paths. For a given assignment to the observables, the value of function $\hat{S}$ is determined by tracing a path from the root vertex to a terminal node, following the edges that correspond to the given assignment of observables. Each path in the diagram represents a row of the table. By tracing the highlighted path, we can see the stress approximation for $abc = 011$ is

3.0. The *depth* of the approximation diagram is defined as the maximum number of non-terminal vertices on any path. Thus, the depth of the exemplar diagram is 3.

5.2.3 Diagram Construction Procedure

The problem of diagram construction is formulated as follows: Given a digital circuit, the specification of a stress metric, and the accuracy requirements, construct an approximation diagram that estimates the stress metric for an arbitrary workload (application) with sufficient accuracy.

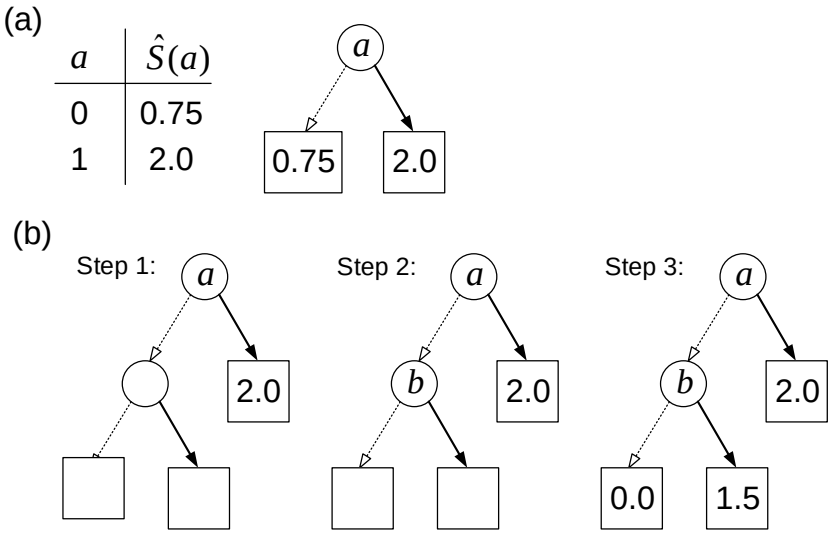
To synthesize the monitor with a better hardware efficiency, we search for observables that are highly correlated with the stress metric and hence are good candidates for online observation. The correlation analysis can be performed by Monte Carlo simulation or an analytic method estimating the signal probabilities [Parke78, Wunde85]. To allow for arbitrary workload, we do not make any assumptions about the application and analyze the Pearson correlation coefficient (cf. Section 5.2.4) in Monte Carlo experiments with random input patterns.

The approximation diagram is constructed with an iterative procedure. In each iteration, the diagram is extended with an additional level of vertices (i.e. additional observables are considered) to improve the approximation accuracy.

In the first iteration, we create an approximation diagram with a single (root) vertex connected to two terminal nodes (Fig. 5.5 (a)). The root vertex is labeled with an observable that exhibits the *highest correlation* to the stress metric. The terminal nodes connected by the 0- and 1-edge are labeled with the *average* stress metric for the case when the observable is 0 and 1, respectively. Both the correlation analysis and the calculation of the average stress metric are performed in Monte Carlo simulation experiments (cf. Section 5.4).

In each following iteration, the depth of the approximation diagram is increased by one level. The diagram is extended as follow (Fig. 5.5 (b)):

- Step 1: Replace each terminal node $v_t \in T$ with a new vertex v'_t connecting to two new terminals (0-successors v'_{t0} and 1-successors v'_{t1}).
- Step 2: Find the conditionally most correlated observable and assign it to the new vertex v'_t: For each path from the root vertex v_0 to previous terminal v_t , we find an observable that exhibits the highest correlation to the stress metric when the



▲ **Figure 5.5** — The construction procedure of an approximation diagram

other observables are fixed to the assignment of $\pi(v_0, v_t)$. Then the new vertex v'_t is labeled with the observable with highest correlation.

- Step 3: Calculate the average stress metrics and assign them to the new terminals: For each path from the root vertex v_0 to a newly created terminal v'_{t0} (or v'_{t1}), the terminal node v'_{t0} (or v'_{t1}) is labeled with the average stress metric for the case when the observables are fixed to the assignment of $\pi(v_0, v'_{t0})$ (or $\pi(v_0, v'_{t1})$).

The procedure is repeated from step 1 until a user-specified boundary of the diagram depth is reached, or until the accuracy of the stress approximation meets requirements (cf. Section 5.2.5).

For instance, in order to find the 0-successor of the root node in Fig. 5.5 (b), we perform the correlation analysis with observable a fixed to 0. As observable b exhibits the highest correlation to the stress metric when $a = 0$, the 0-successor of a is labeled with b. The terminal node connected to vertex b with the 0-edge is labeled with the average stress metric (value 0.0), which is calculated for the case when ab are constantly assigned 00.

5.2.4 Correlation Analysis

Given a candidate observable a and a stress metric S, the correlation coefficient between the observable and the stress metric, denoted by $C(a, S)$, is calculated as a Pearson

product-moment correlation coefficient:

$$C(a,S) := \frac{1}{N-1}\sum_{i=1}^{N}\left(\frac{a_i-\overline{a}}{s_a}\right)\left(\frac{S_i-\overline{S}}{s_S}\right)$$

where:

- N is the number of simulated random patterns,
- a_i and S_i are the state of the observable (0 or 1) and the value of the stress metric for the i-th simulated pattern,
- $\overline{a}$ and $\overline{S}$ are the average values of the observable (i.e. signal probability) and the stress metric,
- s_a and s_S are the sample standard deviations of the observable and the stress metric, respectively.

If the coefficient value $C(a,S)$ approaches to 1, the observable and stress metric have a high correlation. The observable that exhibits the highest correlation to the stress metric is considered for on-line observation, i.e. assigned to a vertex in the approximation diagram.

5.2.5 Evaluation of Accuracy

The approximation accuracy is evaluated for the termination criterion during the construction of the approximation diagram, and to analyze the final quality of the predictor.

The approximation error is defined as the difference between the exact stress metric S and the approximation $\hat{S}$. As the evaluation of the error for the entire state space of a circuit is usually unfeasible, Monte Carlo simulation experiments are performed. In each experiment, we simulate a single random pattern, evaluate the exact stress metric S and derive its approximation $\hat{S}$ from the approximation diagram.

We consider three types of approximation errors: maximal error $E_{\texttt{MAX}}$, mean error $E_{\texttt{MEAN}}$, and root mean squared error $E_{\texttt{RMS}}$:

$$E_{\texttt{MAX}} = \texttt{MAX}_{i=1}^{N}|\hat{S}_i - S_i| \tag{5.1}$$

$$E_{\texttt{MEAN}} = \frac{1}{N}\sum_{i=1}^{N}|\hat{S}_i - S_i| \tag{5.2}$$

$$E_{\mathrm{RMS}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{S}_i - S_i\right)^2} \tag{5.3}$$

where S_i and $\hat{S}_i$ are the exact and approximated stress metrics for the i-th simulated pattern, and N is the number of Monte Carlo experiments.

5.2.6 Monitor Synthesis

The final approximation diagram constitutes the behavioral model of the workload monitor. Each path from the root vertex to one of the terminal nodes corresponds to one assignment of observables, while the label of the terminal node provides the stress approximation. The diagram is representable as a look-up table (Fig. 5.5) which can be transformed into a hardware description language and synthesized with a tool for combinational logic synthesis.

The hardware overhead of the monitor depends on the depth of the approximation diagram, and the precision of stress approximations. To trade off monitor area for accuracy, real-valued stress approximations are quantized. The smallest change in the stress metric measurable by the workload monitor is referred to as *quantization step size*. As shown in the experimental results (Section 5.4), the quantization step size has a significant impact on both the accuracy of a monitor and its area overhead.

5.3 Application to NBTI Monitoring

In this section, we exemplarily apply the presented method to NBTI aging monitoring. We construct a monitor that approximates the number of PMOS transistors which suffer from NBTI stress on the critical and near-critical paths. The monitor can be combined with temperature sensors and used to guide NBTI-aware adaptation, e.g. to prevent that an application causing severe degradation is executed at high temperature.

5.3.1 NBTI Stress Modeling for the Critical Path

The NBTI effect in PMOS transistors consists in oxide degradation caused by formation of traps. The degradation results in a gradual shift in the threshold voltage, which in turn causes an increased propagation delay. Eventually, the NBTI stress may significantly increase the critical path delay and lead to timing violations [Wang07a].

The NBTI-induced degradation occurs when a negative voltage is applied between the gate and source of a PMOS device ($\mathtt{U_{GS}} < 0$). For instance, in a CMOS inverter the PMOS transistor suffers from NBTI stress when the input to the inverter is logic 0. In CMOS gates with stacked PMOS transistors, the stress conditions depend on the state of multiple gate inputs [Saluj08].

We define the NBTI stress metric as the number of PMOS transistors that suffer from NBTI degradation on the critical path. More formally, we define a function $S_{\mathtt{NBTI}} : \{0,1\}^{n+m} \to \mathbb{N}$ that maps the state of device's PIs and PPIs to a natural number that reflects the number of PMOS transistors on a critical path suffering from NBTI stress. The value of this function for a certain input pattern is easily found in simulation by counting the number of PMOS transistors on the critical path with $\mathtt{U_{GS}} < 0$. Note that our NBTI stress metric does not depend on any NBTI model and is technology independent.

Using the approach described in Section 5.2, we approximate the NBTI stress metric $S_{\mathtt{NBTI}}$ with a function $\hat{S}_{\mathtt{NBTI}} : \{0,1\}^{p} \to \mathbb{R}$ that maps the assignment of $p \leq n+m$ observables to an average number of PMOS transistors that suffer from NBTI stress under this assignment.

As the NBTI stress metric is a natural number, a quantization step size of 1 is used for approximations, i.e. the real-valued stress in the approximation diagram is rounded to the nearest integer. The resulting monitor gives an integer approximation of the number of PMOS transistors on the critical path suffering from NBTI degradation.

5.3.2 NBTI Stress for k-longest Paths

As the technology scales, digital circuits become more and more balanced, with many paths that may potentially become critical. To deal with a large number of near-critical paths, we construct a monitor for a cumulative NBTI stress metric for k-longest paths.

Our goal is to monitor the *maximum* number of PMOS transistors subject to NBTI stress on the longest paths. Let $S^{i}_{\mathtt{NBTI}}$ be the NBTI stress metric of an i-th longest path. The NBTI stress metric for k-longest paths is defined as:

$$S^{\mathtt{MAX}}_{\mathtt{NBTI}} = \mathtt{MAX}^{k}_{i=1}\left(S^{i}_{\mathtt{NBTI}}\right) \tag{5.4}$$

The NBTI stress metric for k-longest paths does not identify which path has the maximum number of PMOS transistors under stress, but provides the number of stressed

transistors. For instance if we have two near-critical paths with 7 and 10 transistors that are currently aging, the metric value is 10.

5.4 Experimental Evaluation

5.4.1 Experimental Setup

The method is evaluated on ITC99 [itc] and NXP [nxp] benchmarks. We choose 3 exemplar circuits (p77k, p100k and p330k) to show the tendency and impacts of different experiment settings. The results of all benchmarks are listed in tables of Appendix B. The Nangate 45nm open cell library [nan] is used to synthesize the circuits and monitors. The critical and near-critical paths of the circuits are extracted with a commercial Static Timing Analysis (STA) tool [pri].

For each circuit, we build NBTI monitors for the critical path that limits the performance of the circuit, as well as for 10 and 100 paths that may become critical due to aging. During the construction of approximation diagrams, we simulate 20 000 random patterns to find the successors of each vertex (cf. Section 5.2.3).

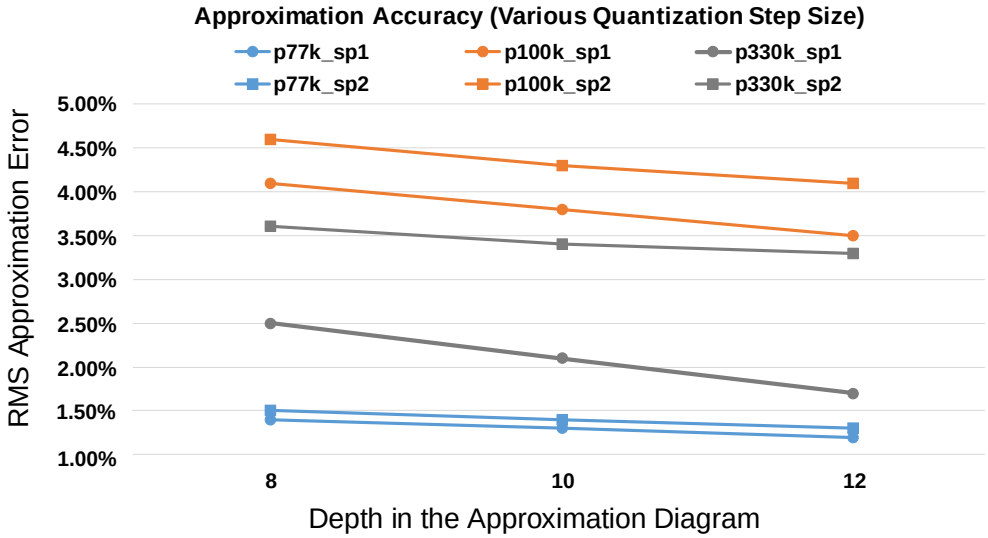
The monitoring accuracy and its area overhead is analyzed for various depths of the approximation diagram (cf. Section 5.2.2) and different quantization step sizes (Section 5.2.6). The accuracy is evaluated in 20 000 Monte Carlo experiments using the metrics defined in Section 5.2.5.

5.4.2 Single path monitoring

Fig. 5.6 presents the accuracy of NBTI monitors for a single critical path with two different quantization step sizes. The monitors approximate the number of PMOS transistors under NBTI stress on the critical path. The approximations are rounded to the nearest integer (quantization step size of 1 or 2). For each benchmark, we evaluate three monitors with the depth of 8, 10, and 12 in the approximation diagram. The *root mean squared (RMS)* approximation errors (cf. Section 5.2.5) are calculated for each monitor with various depth in the approximation diagram. Points in the figure represent the values of the RMS errors and are connected with a line when an identical quantization step size is applied for one benchmark.

The lines in Fig. 5.6 show that the approximation accuracy improves when a workload monitor (NBTI stress monitor) is constructed with a larger depth of the approximation

diagram i.e. when more inputs are under observation, as a result, the monitor area overhead is higher (cf. Section 5.4.4). For the largest circuit (p330k) with quantization step size of 1, the RMS error is 2.5% with 8-, 2.1% with 10-, and 1.7% with 12-depth. A similar improvement is observed for other large benchmarks.



▲ **Figure 5.6** — Accuracy of NBTI monitors for a single critical path with quantization step size of 1 and 2

The lines with quantization step size of 1 (benchmark_sp1) and 2 (benchmark_sp2) are marked with circles and squares respectively (Fig. 5.6). Compared to the results with a quantization step size of 1, the approximation error with step size of 2 increases, but it is still within ±10% for 95% of random patterns since the RMS error is below 5% for all benchmarks.

Table B.1 in Appendix B estimates the average NBTI stress S_{avg} and evaluates the *maximum*, *mean* and *root mean squared (RMS)* approximation errors for monitors with various depth (e.g. 8-, 10-, 12-depth) in the approximation diagram. When a single critical path is under NBTI stress observation, we define the average NBTI stress S_{avg} as the *average* number of PMOS transistors under the negative voltage bias on the critical path over all applied test patterns. The presented relative errors are calculated as the absolute errors defined in equations Equ. 5.1 to Equ. 5.3 (cf. Section 5.2.5) divided by the average NBTI stress S_{avg}.

The maximum error for 8-depth monitors and the step size of 1 ranges between 5.5% and 21.6% for evaluated benchmarks, i.e. in the worst case (p295k), there exists an

input pattern, for which the approximation of the NBTI monitor differs from the exact stress by 21.6%. The mean error is significantly lower: In the worst case (p269k), the approximation differs from the exact value by 3.3% on average. The RMS error is bound to a maximum of 4.4% (p269k): Assuming that the approximation error is normally distributed, the error is within ±8.8% for 95% of patterns.

The results prove that an accurate stress estimation is achieved for most input patterns in all experimented designs. A higher depth of the approximation diagram and a smaller quantization step size can improve the estimation accuracy.

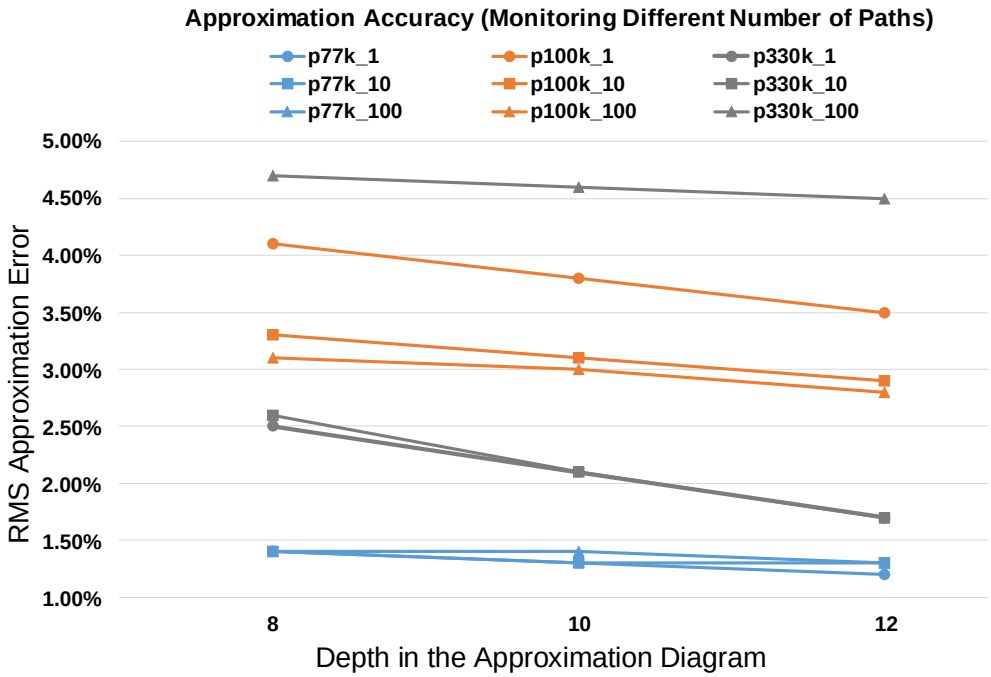
5.4.3 K-longest paths monitoring

Now we discuss the accuracy for 10- and 100-longest paths monitoring. When multiple paths are under observation, we define the average NBTI stress S_{avg} as the average value of S_{NBTI}^{MAX} in Equ. 5.4 over all given test patterns (cf. Section 5.3.2). The absolute RMS errors are calculated with Equ. 5.3. The relative errors are the absolute error as the percentage of the average stress S_{avg} and depicted in Fig. 5.7. The quantization step size of 1 is set for stress approximation. Results for each benchmark are presented as lines in the same color. Homochromy lines (benchmark_#path) with different observing path group are marked with circles, squares, and triangles respectively. The lines for a single target path (benchmark_1) is also shown in the figure as a reference for comparison.

For some benchmarks, such as p330k, the more paths are monitored, the larger depth in the approximation diagram are required to preserve the accuracy. However, the monitors for other benchmarks (e.g. p100k) become more accurate when more paths are under monitoring. For the latter circuits, the maximum stress metric for multiple paths encounters less variation than the stress of a single path (i.e. the value of the stress depends less on the input patterns). Thus, the maximum stress metric is sometimes easier to approximate than the stress of a single critical path.

5.4.4 Hardware Overhead

Fig. 5.8 present the monitoring overhead for various quantization step size. Fig. 5.9 illustrates the cases for observing various number of paths. In both figures, the points represent the relative overhead w.r.t. the benchmark area (i.e. the absolute hardware area induced by the stress monitor as the percentage of the corresponding benchmark area).

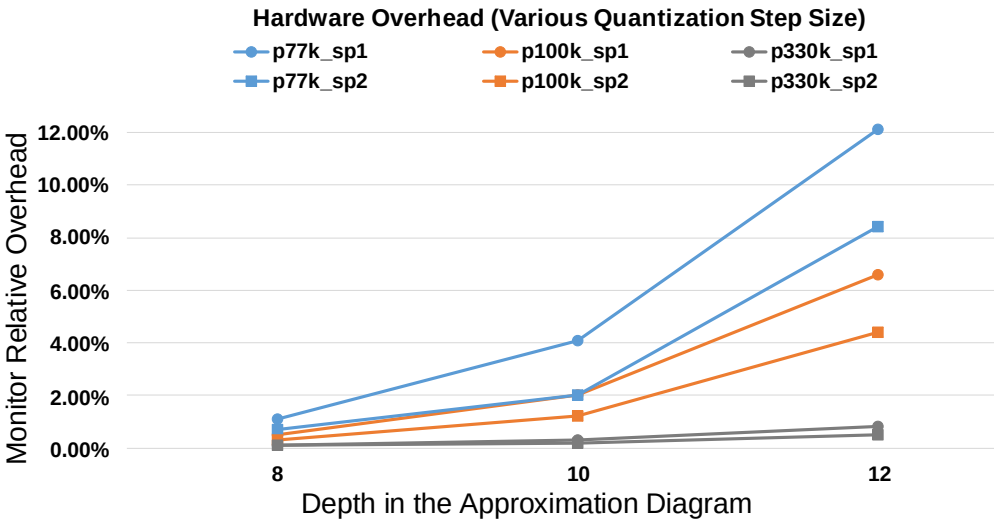


▲ **Figure 5.7** — Accuracy of NBTI monitors for different number of paths with quantization step size of 1

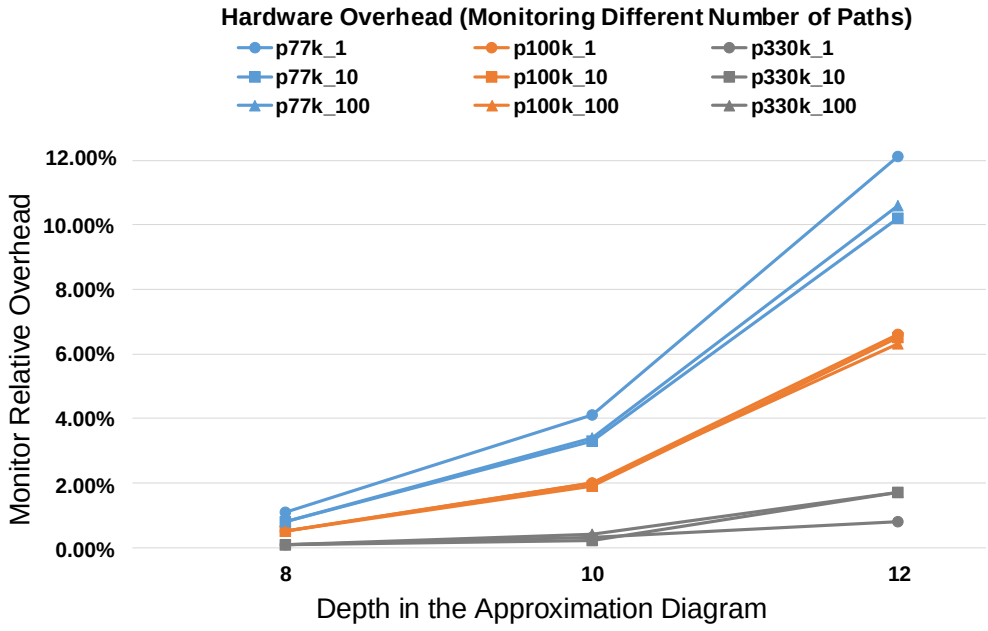
The points with the identical quantization step size (1 or 2) but different approximation diagram depths (8, 10 and 12) are connected in one line.

As shown in Fig. 5.8, the area overhead of a workload monitor increases with the number of diagram depth and also depends on the quantization step size. For monitoring a single critical path with the quantization step size of 1, the maximum overhead is 328, 1220 and 3560 μm^2 and the relative overhead is around 1%, 4% and 12% respectively for 8-, 10-, and 12-depth monitors of p77k. For the quantization step size of 2, the overhead is reduced by up to 66%. As the monitor area depends little on the size of the monitored circuit, the relative cost of monitoring decreases with an increased circuit size, e.g. the 12-depth monitor induces more than 12% overhead for p77k. However, the relative cost reduces to less than 1% for p330k.

Monitoring multiple paths induces no significant overhead increase of workload monitors. For large-sized circuits (p100k, p330k), the lines for 10- and 100-longest paths monitoring are almost overlapped the one for a single critical path observation (Fig. 5.9).



▲ **Figure 5.8** — Hardware overhead of NBTI monitors for a single critical path with various quantization step size



▲ **Figure 5.9** — Hardware overhead of NBTI monitors for different number of paths with quantization step size of 1

5.5 Summary

Online stress estimation has become mandatory for applications with reliability requirements. A novel method is introduced to monitor workload-induced stress to predict the degradation *before* any measurable parameter deviation and *proactively* perform the countermeasures for wear-out elevation. The degradation rate, i.e. stress, of a circuit depends on the workload of the currently operating application, which is estimated by observing its logic state. The method is suitable for the monitoring of various degradation mechanisms depending on the logic states of the circuit. As a case study, the technique is applied to measure the number of transistors suffering from NBTI degradation on the critical or near critical paths. Typically, the developed monitors require an area overhead of under 1% and offer an average estimation error below 3.3%. The number of biased PMOS transistors represents the stress deviation between different applications. This quantified stress variation helps to enable NBTI-aware scheduling, load balancing, or guide the insertion of healing patterns or instructions.

To estimate the aging-induced degradation within a configurable interval, e.g. a fraction of a second, we can use the workload monitor to assess the stress of each representative critical gate and evaluate the observed stress with a software [Baran15]. The representative critical gates are a small set of selected gates in the monitored circuit, the workload of which is highly correlated with the degradation of the entire circuit. The outputs of the workload monitor are summed up over a short period. Together with the observed temperature profile, the aggregated stress is periodically fed to an aging prediction model in software, ideally running during the idle time of any available processing unit.

CHAPTER 6

Efficient Monitor Placement for Early Aging Prediction

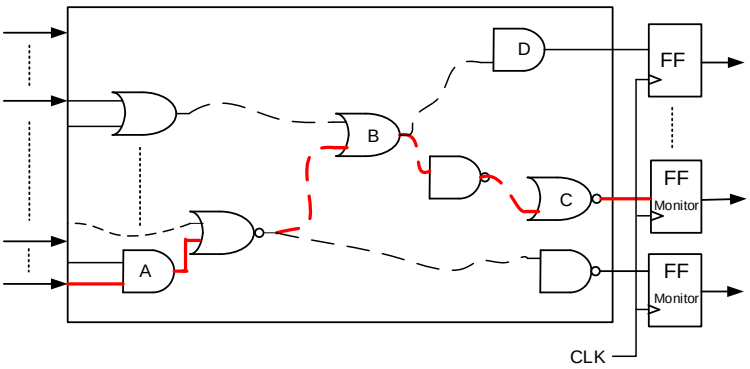
Circuit aging causes a performance degradation and eventually a functional failure. It depends on the workload and the environmental condition of the system, which are hard to predict in early design phases resulting in pessimistic design (cf. Section 2.2). Existing in-situ delay monitoring schemes (e.g. the delay detecting flip-flop cf. Section 4.4.2) measure the remaining slack of paths in the circuit, but have a high hardware penalty. A conventional solution is to place the monitors at the terminals of long or critical paths for reduction of hardware overhead. However, in recent low-power designs, the path lengths are equalized, and many paths have lengths close to the critical path. Additionally, delay uncertainties caused by process variations, transistor workload or working conditions are hard to predict and compensate in the design phase. The critical path of the design may vary from chip to chip and over time. Thus, not only the nominal critical path but also the near critical ones have to be sensed which leads to a significant monitoring penalty. More importantly, locating the monitors only at long path-ends may cause unpredicted timing failures for two reasons:

1. The increased delay of short paths can never be measured due to their large slacks, i.e. the aging degradation in a significant fraction of circuit paths are

unobservable. After a period of gradual deterioration, a sudden failure may occur because of a TDDB on unmonitored short paths.

2. The endpoint placement of monitors implies that the path delay is only measured when the entire path is sensitized, which highly depends on the circuit structure, path length and input stimuli. Some applications hardly sensitize the whole circuit network [Baran15], and it is possible that individual paths are only sensitized rarely during operation, causing great testing latencies and in the worst case unmonitored timing violations.

Fig. 6.1 illustrates one of such cases. A path through gates A, B and C, ABC for short, is a monitored long path. If ABC is seldom sensitized, the aged circuit may work fine without any alert or timing violation until a sudden breakdown happens on path ABD. In another case without TDDBs, after a long time, the path delay of ABC deteriorates and exceeds the timing constraints of the circuit. However, this increased delay is unobservable because ABC is not sensitized. When ABC is once again activated, a transition along the degraded ABC violates not only the predefined timing margin but the clock as well. A timing violation occurs without indication.



▲ **Figure 6.1** — Limitation of the conventional monitor placement

To avoid such unmonitored timing violations and measure aging degradation at a higher frequency, we propose a placement method for in-situ monitors. It analyzes the topological circuit structure, the path segment slack, and sensitization probability. Monitors are then inserted at meticulously selected positions in the circuit, named observation points (OPs), and measure the delay of path prefix-segments more frequently. Only very few monitors are required to achieve a high coverage of target paths.

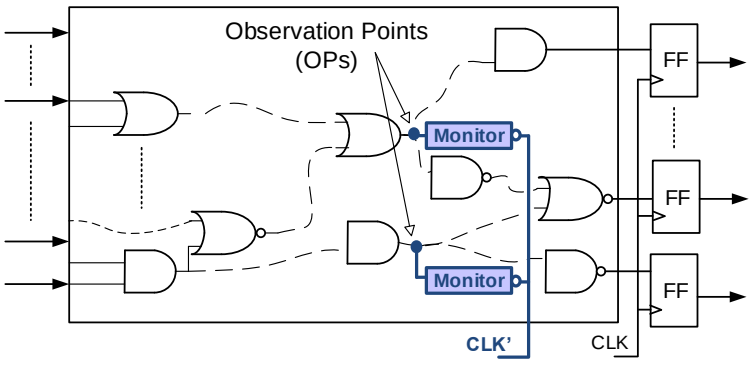
The contributions of this work consist of:

- an efficient monitor placement approach employing in-situ aging monitors for more frequent measurement of prefix-segments of long paths;
- an OP selection algorithm to identify the positions for sensor insertion in the intermediate circuit nets;
- the effectiveness validation flow of this approach to show that a monitor always activates earlier than any imminent timing failure.

The remainder of the chapter is an extension of [Liu15] and organized as follows: Section 6.1 gives an overview of the monitor placement scheme. The OP placement method is presented in Section 6.2. The validation setup and experimental results are discussed in Sections 6.3 and 6.4.

6.1 Delay Monitor Placement Approach

The goal of our placement approach is to achieve an early prediction of aging degradation by more frequent path delay measurement and concurrently reduces the hardware cost for aging monitoring. The proposed method selects observation points at intermediate nodes in the combinational circuit (e.g. blue points for monitor insertion in Fig. 6.2).



▲ **Figure 6.2** — Delay detecting monitor relocation

Path segments are impacted by aging. Thus the length of path segments can be screened as a degradation indicator. Shorter paths or prefix-segments of paths are sensitized more often during operation, resulting in a higher measurement frequency of the path delay. Since path segments are typically shared by multiple paths, the selection of a

subset of segments for monitoring allows assessing partial degradation of all or at least a high fraction of longer paths.

Positions closer to inputs are reached along shorter prefix-segments of paths, enlarging the sensitization probability and delay measurement frequency. However, the prefix-segments should also include a proper number of gates to represent the degradation pattern of the entire circuit. By monitoring the prefixes with lengths close to half the clock period, both requirements: the frequent measurement and the representative timing behavior can be satisfied and balanced.

Then we need to justify the detection window of monitors to sense the transitions propagating along such prefix-segments of paths. According to the working principle (cf. Section 4.4.2), monitors generate the detection window before the rising edge of their reference clock. Therefore, the circuit clock delayed by a half clock period can be applied as the monitor reference clock. If the clock signal has a duty cycle of 50% (i.e. the clock falling edge comes always a half period after the rising edge, as shown in Fig. 6.3 (a)), then we can use the inverted clock as a reference clock for monitors to generate the guard band at OPs. Utilizing the inverted clock as a common reference can avoid multiple additional clocks and simplify the clock generation.

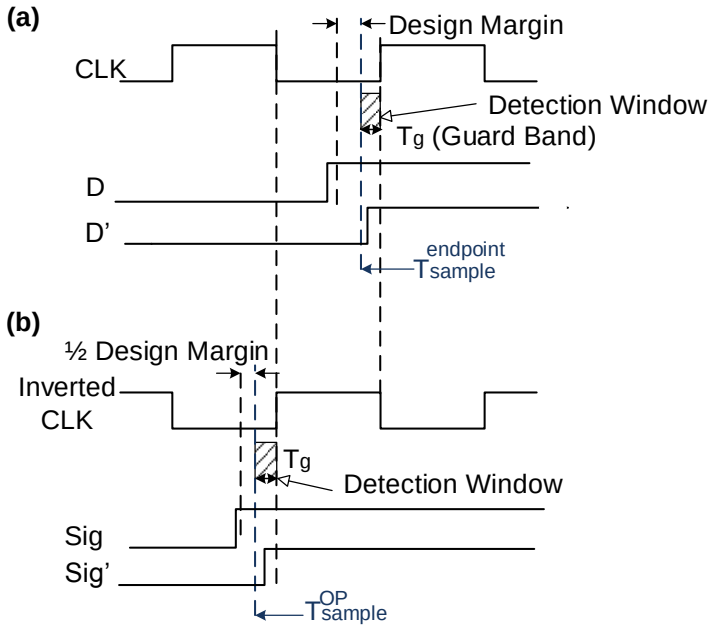
Both monitor structures (cf. Section 4.4.2) can be applied and embedded in OPs. If a transition of a delayed signal (*D'* or *Sig'*) occurs during the detection window, an aging alert is generated. At OPs, the start of the monitor detection window moves to half the clock period minus the guard band T_g (i.e. T^{OP}_{sample} in Fig. 6.3 (b)). The monitored path length is halved. If only the systematic variation is under consideration, the design margin can shrink proportionally to half of its original value.

To avoid complicated global wiring and clock balancing issues in [Lai13], the guard band and T^{OP}_{sample} are unified for all selected observation points.

6.2 Selection of Observation Points

6.2.1 Terminology

Let the *topological depth of a gate g* be the length of the longest path segment from an input to g. The set of *target paths* comprises the critical and near-critical long paths in the circuit, obtained e.g. by static timing analysis (STA). The *target outputs* are the primary and pseudo-primary outputs at the end of target paths. An *OP covers*



▲ **Figure 6.3** — Signal waveform for a delay detecting flip-flop: (a) at path endpoint (b) at observation point (OP)

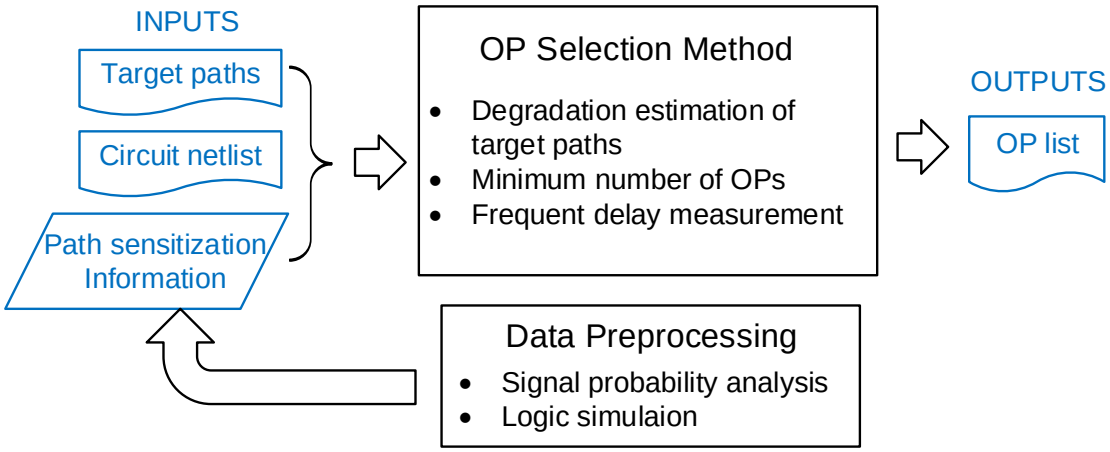
a path if the delay of the path initial segment, which can be directly measured by a monitor inserted at OP, exceeds a predefined threshold (Section 6.2.3). The *target path coverage* is the percentage of target paths covered by OPs. The *OP candidates* are possible monitor insertion locations, limited by timing constrains to reduce the search space of OP selection (Section 6.2.4). The *OP slack* or *OP candidate slack* is the time difference between the start of the monitor detection window (T^{OP}_{sample}) and the topological depth of the OP or OP candidate. In the nominal case, the OP slack is always larger than or equal to half the design margin (Fig. 6.6 (b)). The *OP upper bound* (OP_{UB}) bounds the topological depth of the OP candidates. As mentioned in Section 6.1, half the clock period is chosen as sampling reference and every transition occuring after the sampling time (T^{OP}_{sample}) triggers a detecting alert. To prevent monitor false activation, OP_{UB} is set as half the clock period minus guard band and half the design margin. A *path prefix* is the prefix of a path starting from its primary or pseudo-primary input to the gate with maximum topological depth less than OP_{UB}.

6.2.2 Problem Statement

Observation point selection aims to use the observability provided by a limited number of OPs to estimate the degradation of target long paths in the circuits frequently. This general goal can be translated into three fundamental requirements of the selection method:

1. The delay of a path prefix (from a circuit primary or pseudo-primary input to an OP) should represent the aging process in its corresponding target path.
2. The number of OPs for degradation measurement should be minimized for low hardware cost.
3. The selected OPs should have the high sensitization probability during operation to ensure a frequent measurement of covered target paths.

As shown in Fig. 6.4, we provide the target paths, circuit netlist, and the path sensitization information as the selection inputs. The target paths can be collected by performing the timing analysis of the circuit. The path sensitization information is computed by a data preprocessing procedure based on the signal probability analysis of a particular application or the logic simulation w.r.t. an input pattern set. Then with the given inputs, the selection method generates an OP list.



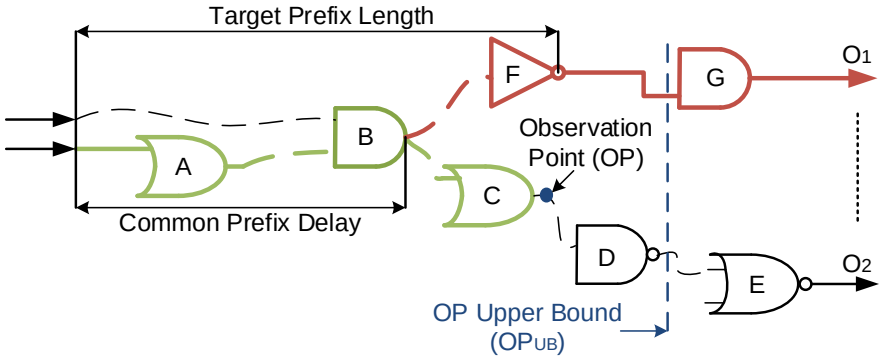
▲ **Figure 6.4** — The inputs, outputs and requirements of the OP selection

During the selection, initially, we need to identify the monitor insertion positions (i.e. OP candidates) satisfying the first requirement. Compared to all intermediate nets in the

combinational circuit, the OP candidates identification significantly narrows down the search space, thus reduces the computational effort for OP selection (cf. Section 6.2.4). Since minimizing the number of OPs (i.e. the second requirement) is not the only goal in this optimization, we also need to take the sensitization possibility of the observing path prefixes (i.e. the third condition) into account. Therefore, the OP selection task can not be directly modeled as a set-covering problem. We introduce a heuristic OP selection algorithm (cf. Section 6.2.5) to find a small set of OPs among the candidates.

6.2.3 Target Path Coverage of Observation Points

The principle of our monitoring approach is to assess the degradation by measuring the delay of target path prefixes. Many target paths have branches close to OP_{UB}. The branched prefixes, e.g. segments ABF and ABC in Fig. 6.5, share most of the path elements. The large common path prefix AB implies a high correlation between the lengths of the branches ABF and ABC, allowing to estimate the length of all branches by observing one of them. If the length of the common prefix (as percentage of target prefix length) exceeds a given threshold (e.g. $length_{\mathrm{AB}}/length_{\mathrm{ABF}} > 70\%$), we say the target path ABG is partially covered by the OP (blue point at gate C), although the corresponding output O_1 is unreachable from the OP at C.



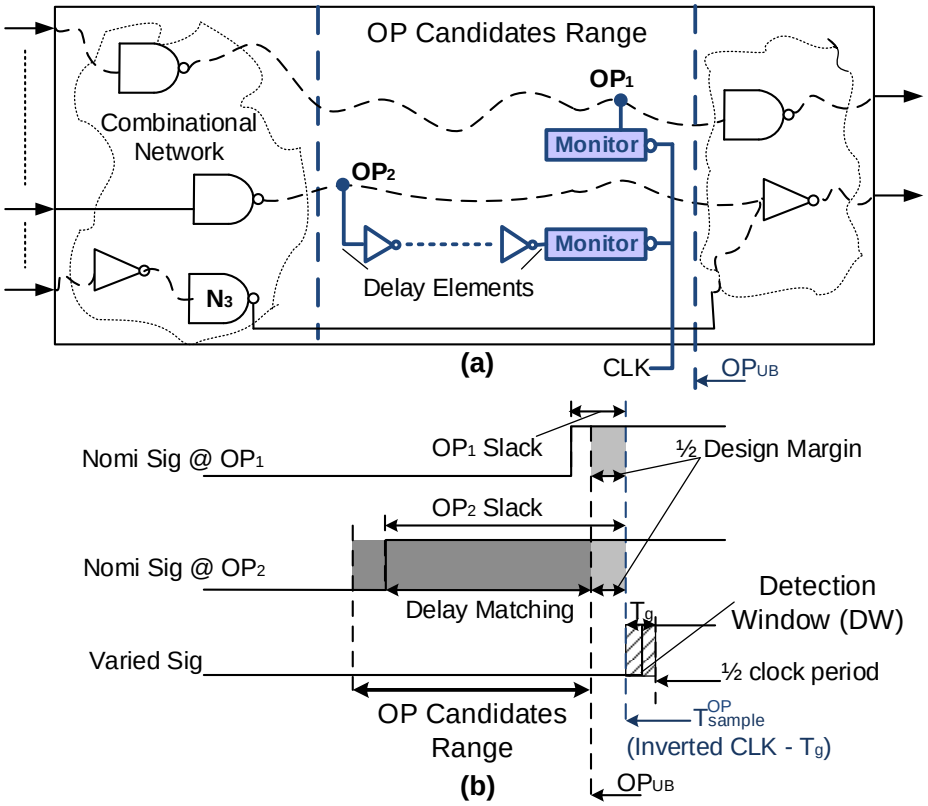
▲ **Figure 6.5** — OP covering a target path prefix

6.2.4 Observation Point Candidate Range

In principle, all gates with a depth less than OP_{UB} are potential OP candidates. To reduce the search space for OP selection, a candidate range is introduced that limits

the considered candidates. It is computed by balancing the monitoring quality and overhead. The quality depends on the target path coverage and the OP candidate slacks.

If a candidate's depth is close to OP_{UB} (e.g. OP_1 in Fig. 6.6 (a)), the slack of the candidate will be close to the design margin (light gray area shown in Fig. 6.6 (b)). During operation, OP_1 slack gradually reduces due to degradation and may eventually cause a timing alert by the monitor. However, it may be that the candidates close to OP_{UB} cannot cover all target paths. For high target path coverage, the gates with larger slacks are also considered as candidates (e.g. OP_2). To maintain a similar margin for degradation, the time difference between the OP slack and half the design margin, named delay matching region, is compensated by inserting delay elements (e.g. inverters).



▲ **Figure 6.6** — OP candidate range identification: (a) topological position in the circuit netlist; (b) signal waveform

The required number of inverters can be calculated based on the OP slack in either nominal or degraded case. Since the degradation of prefixes and paths depends on the workload and operation condition, the prefix segments can degrade at varying speed

with the corresponding paths. Usually the general application scenarios and average operation condition are already known at design time. They can be utilized for aging analysis to adjust the OP delay matching and further improve the monitoring accuracy. The dark gray area in Fig. 6.6 (b) illustrates the topological range of OP candidates ($TopoDelay_{\text{cand}}$):

$$OP_{\text{UB}} - MD_{\text{max}} < TopoDelay_{\text{cand}} < OP_{\text{UB}} \tag{6.1}$$

MD_{max} is the maximal matching delay of the inverter chain. The OP candidate range should satisfy the following criteria:

- The OP candidates should provide a high target path coverage to avoid insertion of extra delay monitors at the end of uncovered target paths.
- The hardware overhead caused by delay matching should be minimal.
- The path prefixes covered by OP candidates should contain enough degradation information for aging monitoring, i.e. the prefix length should be long enough to ensure high timing correlation to the entire target path.

Due to the third criterion, a full target path coverage cannot always be achieved by simply increasing the matching delay (length of the inverter chain). For instance, as shown in Fig. 6.6 (a), for the path at the bottom of the figure, N_3 is the gate closest to OP_{UB}. However, if the path prefix length (propagation delay from input to N_3) is not large enough, the delay increase from the inverter chain may dominate the overall degradation of the path prefix, which is not representing the aging in target paths or the entire circuit. In this case, to keep the monitoring quality, sensing the entire path length would be a better solution than measuring only the prefix segment. As shown later in the experiment results, such uncovered paths are very rare.

6.2.5 OP Selection Algorithm

The OP selection problem can be formulated as follow: for a computed set of OP candidates $C := \{c_1, c_2, c_3, \ldots, c_M\}$ (Section 6.2.4), identify a minimal subset $V \subseteq C$ (i.e. OPs) such that the delay degradation of the target paths $P := \{p_1, p_2, p_3, \ldots, p_N\}$ can be observed by monitors placed at V with a high measurement frequency. The measurement frequency is quantified by the OP sensitization ratio, generated by logic simulation of functional or random patterns. The sensitization ratio $sr(c_i)$ at particular

OP candidate c_i is calculated as the number of cycles in which the signal at c_i toggles divided by the total number of simulation cycles. For each candidate, a merit factor μ is computed as the product of the number of covered paths and the sensitization ratio. The heuristic algorithm selects OPs from candidates in order of decreasing merit μ. To reduce the coverage overlap, the paths covered by fewer candidates are considered with higher priority.

Before OP selection, the netlist and set of target paths P are used to identify the OP candidates C. The target path coverage is calculated w.r.t. the entire candidate group. As discussed in Section 6.2.4, uncovered paths ($U \subseteq P$) are removed from the target group ($P := P \setminus U$), and delay monitors will be integrated into flip-flops at the end of uncovered paths.

Then, OP selection iterates as follows:

- Step 1: Analyze the path and OP candidate relation
  - For every target path p_i, the OP candidates covering the path are identified, denoted as $SC_i \subseteq C$. Let $\mid SC_i \mid$ be the size of SC_i.
  - For every candidate c_i, the set of covered paths is defined as $SP_i \subseteq P$. $\mid SP_i \mid$ is the size of SP_i.
- Step 2: To reduce the path coverage overlap and obtain an optimal OP set, the hard-to-cover paths (i.e. paths covered by a minimal number of OP candidates) are considered before the rests in OP selection: select p_s, when $\mid SC_s \mid = \mathtt{MIN}_{i=1}^{N} \{\mid SC_i \mid\}$. SC_s is the candidate set covering path p_s. This set is irreplaceable by other candidates for a high target path coverage.
- Step 3: Select the candidate c_s in SC_s with maximal merit factor $\mu_s = \mathtt{MAX}_{i=1}^{|SC_s|} \{\mid SP_i \mid \cdot sr(c_i)\}$ as OP. This OP c_s covers not only the hard to cover path p_s, but has also a large potential to cover other target paths with high measurement frequency.
- Step 4: Remove the selected OP and covered target paths: $C := C \setminus c_s$, $P := P \setminus SP_s$. If $P \neq \emptyset$, repeat from step 1.

6.3 Observation Point Effectiveness Validation

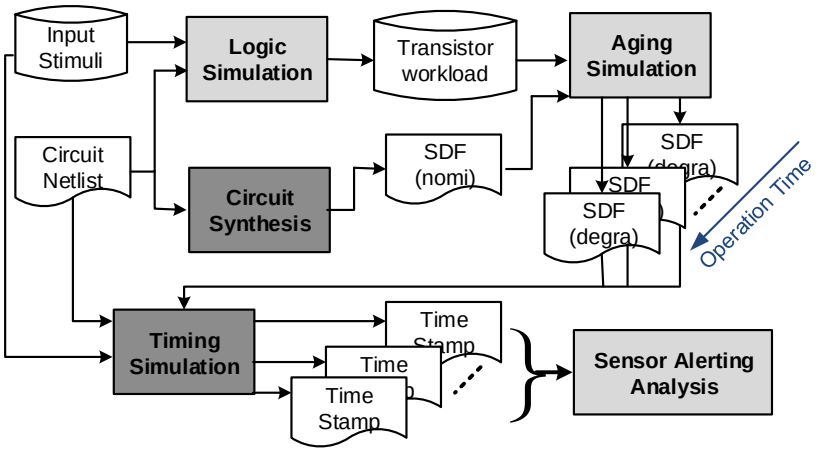
The effectiveness of the selected OPs are evaluated from two aspects:

- (1) Failure predictability: a timing alert is generated at OPs before any actual timing failure.
- (2) Prediction validity: every monitor alert refers to an imminent failure, i.e. no false positive alerts are generated.

In the validation (Fig. 6.7), the transistor stress is estimated by logic simulation of functional or random input stimuli. The nominal standard delay file (SDF) and the transistor stress are input to an aging model to create the SDF for degraded gates. Using the degraded delay information, timing simulation is done for an input pattern set. The time of the last transition at OPs and target outputs are recorded for each cycle and stored in the *Time Stamp file*. In sensor altering analysis, the time stamp values at OPs and target outputs are compared with the corresponding signal sampling time. If a transition at an OP arrives after T^{OP}_{sample} (half clock period minus guard band T_g), an aging alert is issued. A transition at a target output exceeding the clock period indicates a timing failure.

Different degraded timing profiles are generated per circuit for different system operation times. For each degraded SDF, timing simulation is repeated and the transitions at OPs and outputs are analyzed. The times of the first alert activation and the first timing failure at outputs are recorded. If the first alert occurs earlier than any failure, the property of failure predictability holds.

To show the prediction validity, the remaining timing margin of the nominal critical path is analyzed when the first aging alert occurs at an OP.



▲ **Figure 6.7** — The work-flow of OP effectiveness validation

6.4 Results of Experiments

We evaluate the approach on ISCAS'89 [Brgle89] and NXP [nxp] benchmarks. The OP selection algorithm and the light gray blocks of the validation scenario (Fig. 6.7) are implemented in our in-house EDA environment. The Static Timing Analysis (STA) [pri], timing simulation [mod] and circuit netlist synthesis [des] (dark gray blocks in the figure) are realized by the commercial tools. The Nangate 45 nm open cell library [nan] is used for synthesis and timing analysis. The design margin is set to 10% of the critical path length (cpl), i.e. the clock period is $clkp = 1.10 \cdot cpl$.

6.4.1 Observation Point Selection Results

The results of the OP selection for all benchmarks are listed in Appendix C Table C.1. The target path group are selected by STA as the 100-longest paths per output with a length from 70% of the critical path delay to a specific upper bound. The upper bound is defined as the length of the longest sensitizable path through such output. The number of target paths ranges from several hundreds up to more than thirteen thousands (p78k) and depends on the topological structure of circuits. The (primary and pseudo-primary) target outputs are locations at target path ends for the conventional monitor placement.

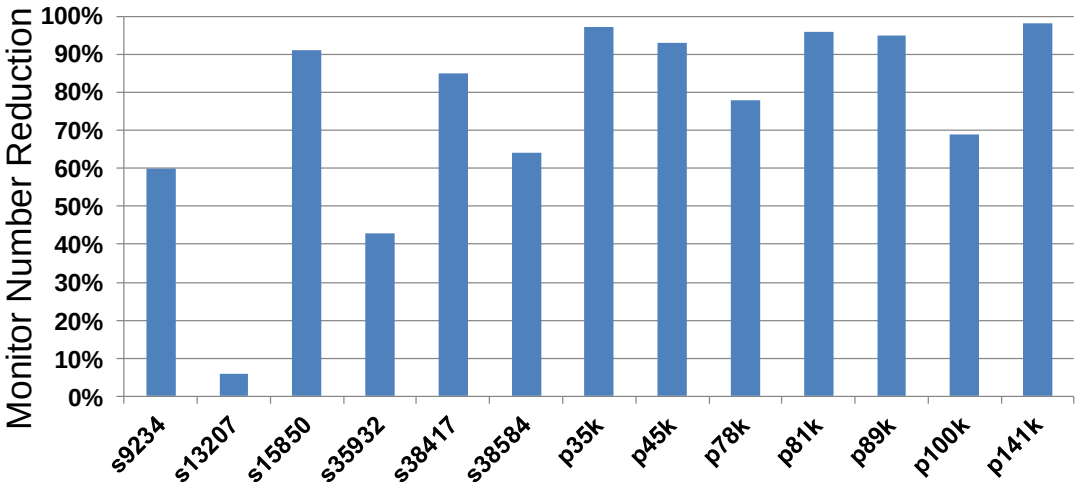
To identify the OP selection candidates, the maximal matching delay $MD_{\max}$ in Equ. 6.1 is set to the delay of six inverters. The prefix covering threshold (Section 6.2.3) is set to 70%, i.e. for every target path if more than 70% of its prefix delay can be directly measured by an OP candidate, we say the candidate covers the corresponding target path. Candidates covering no target paths are removed from the group. p78k has 2227 candidates due to its large number of target paths. Except for this outlier, the remaining benchmarks have below 500 candidates. We check the target path coverage of the candidates before the OP selection. A high target path coverage is achieved by the candidate range defined in Equ. 6.1 and this coverage is maintained during OP selection. Additional monitors need to be inserted at the ends of uncovered target paths. We call these monitors *endpoint insertions*. Thus, the number of monitors required by our new placement method is the sum of the number of OPs and endpoint insertions.

Compared to conventional endpoint placement, our method reduces the number of monitors significantly (Fig. 6.8). We define the *monitor number reduction* as number of monitors saved by our method divided by the number of monitors required

conventionally, expressed as:

$$reduction := \frac{\#moni_{path-end} - \#moni_{new}}{\#moni_{path-end}} = \frac{\#tar\_outputs - \#OPs - \#endpoints}{\#tar\_outputs}$$

where $\#tar\_outputs$ is the number of target outputs, $\#OPs$ refers to the number of observation points and $\#endpoints$ is the number of endpoint insertions due to uncovered target paths (in Table C.1).



▲ **Figure 6.8** — The monitor number reduction of our new method compared to conventional endpoint placement

The *monitor number reduction* ratio (Fig. 6.8) depends on the circuit structure, i.e. the long path distribution, therefore varies between different designs. The reduction of inserted monitors ranges from 6% (s13207) up to 98% (p141k). For larger circuits, the reduction improves. On average, the reduction is 58% for the ISCAS'89 circuits and 89% for the NXP circuits.

To avoid the self-degradation, aging monitors are activated periodically. We assume the power consumption of monitors has little impact to the entire sensing design and consequently, reduces proportionally to the reduction of monitors.

The experiment is processed on a Xeon server with 2.67GHz CPUs and 80 GB memory. The run-time of the OP selection algorithm is ca. 6 minutes for p141k.

6.4.2 Results of OP Effectiveness Validation

The goal of the validation experiment of Section 6.3 is to investigate whether an accurate failure prediction can be achieved by more frequent path prefix measurements in the proposed method. Random inputs (or functional patterns) are used to compute the sensitization ratio in the OP selection (cf. Section 6.2.5).

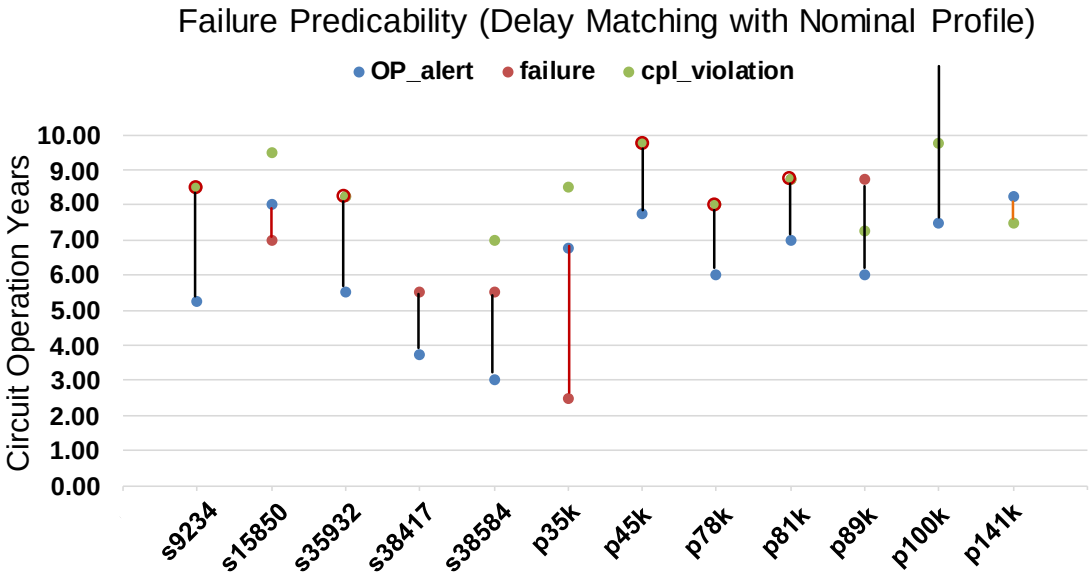
Since the number of random patterns needed to sensitize a path grows exponentially with the path length, for deep circuits long target paths will be sensitized only with a very low probability. In the validation experiments, the limited number of random patterns is insufficient to sensitize a significant fraction of paths through the OPs and target paths. For that reason, path delay fault ATPG patterns for the target paths are generated and applied in the OP validation flow. Due to the ATPG abort limit, some sensitizable target paths may not be activated during validation.

Additionally, 10240 random patterns are simulated to calculate the workload-induced stress of each transistor. The workload is assumed to be constant for the circuit lifetime. NBTI is considered as the dominant mechanism of aging degradation. Similar to [Baba09], the NBTI model published in [Bhard06] is used here for aging simulation. According to the aging model, the gate delay increase (Δt) is computed as: $\Delta t = A \cdot (\alpha \cdot t)^n \cdot t_0$. The stress probability α is the probability that a PMOS transistor is under stress in one cycle. t refers to the circuit total operation time. $n = 1/6$ is a characteristic constant of the NBTI effect. t_0 denotes the nominal pin to pin delay of the gate. A is a constant parameter and is adjusted so that the delay degradation is 10% over 5 years under 50% stress probability. Different operation times from 0 to 10 years with the resolution of a quarter year (0.25, 0.5 ... 9.75, 10 years) are applied to the aging model to generate the degraded gate delay over the operation time. Later, the above mentioned path delay fault patterns are used in the timing simulation. The timing simulation is repeated with the degraded delays of the different operation times until 10 years.

Since the inverter chain for OP delay matching can be implemented based on nominal or degraded timing (Section 6.2.4), the two cases are investigated separately. The results of the two delay matching cases are listed in Appendix C Tables C.2 and C.3 respectively. For circuit s13207, no aging alert or timing failure was observered at all. The aging degradation of the entire circuit s13207 can still be compensated by the design margin.

Delay Matching based on Nominal Timing Profile

Points in Fig. 6.9 provide the operation time (in years) until the occurrence of an event. The event can refer to the first observed aging alert activation (OP_alert: blue dots), the first observed timing failure ($failure$: red dots). We collect the occurrence time of both events: OP_alert and $failure$ by timing simulations. Then by performing static timing analysis, we record the first time when the critical path violates the operation clock ($cpl\_violation$: green points). When the first timing failure is caused by the critical path (i.e. red and green dots overlap), we depict such situation with a red circled green dot in the figure. If any of the events (the alert, failure or violation) is not observed by the applied input stimuli during 10 years of operation, the corresponding point is missing in the figure (e.g. s38417, p100k and p141k).

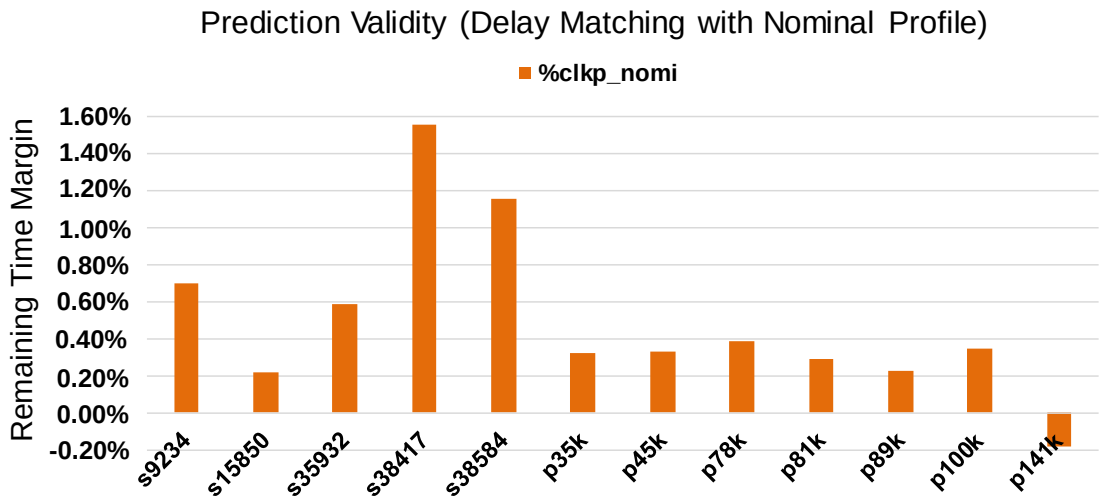


▲ **Figure 6.9** — Failure predictability validation of observation points (with nominal timing profile)

The blue and red points in Fig. 6.9 present the predictability of OPs (cf. 6.3). When the blue point of a benchmark is below its red one, we link the two points with a black line, which indicates that the first aging alert occurs earlier than any timing failures. The predictability of OPs is proven for most of the circuits. However, for circuits s15850 and p35k, the monitored prefixes degrade slower than the entire paths (marked as red lines). This can be avoided when the workload and operating condition of the circuit are applied for delay matching (cf. Section 6.4.2). For some circuits (e.g. s38417, s38584),

a timing failure occurs before the degraded critical path violates the clock period. This is because the critical path changes during degradation.

Fig. 6.10 displays the remaining time margin of the nominal critical path when the first monitor alert activates. The relative value is the absolute time margin as the percentage of the nominal critical path length. Compared to the 10% original slack of the critical path, the remaining margin is less than 1% in average, illustrating the degradation process in the circuits. Therefore, the the prediction validity is proven.



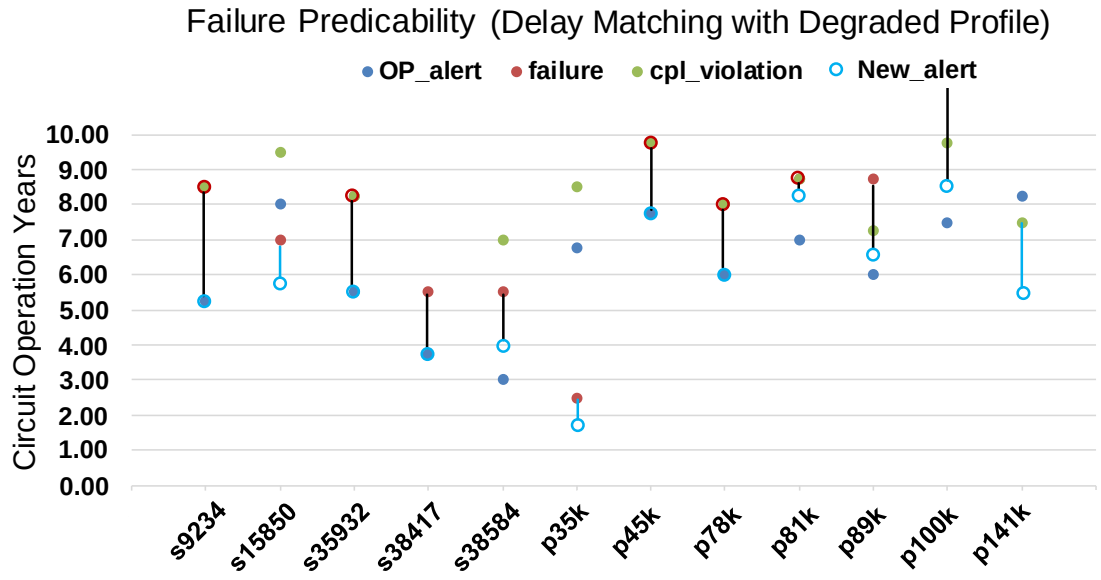
▲ **Figure 6.10** — Prediction validity of observation points (with nominal timing profile)

A timing violation of the nominal critical path indicates a potential aging failure, but it may not be activated by the input stimuli. For p141k, for instance, the critical path violates the timing after 7.5 years. This degradation could be undetectable if the monitors are placed at path endpoints and the entire path is not sensitized by the input stimuli. As shown, no timing failure has been observed during 10 years of operation (no red dot in Fig. 6.9 for p141k). By measuring the path prefix, this potential failure is indicated in the 8.25th year. Still, the OP activation occurs later than the potential failure (the blue point is connected with the green dot with an orange line). As the result, a negative margin is shown in Fig. 6.10. This situation can be avoid when the degraded timing profile is applied for delay matching.

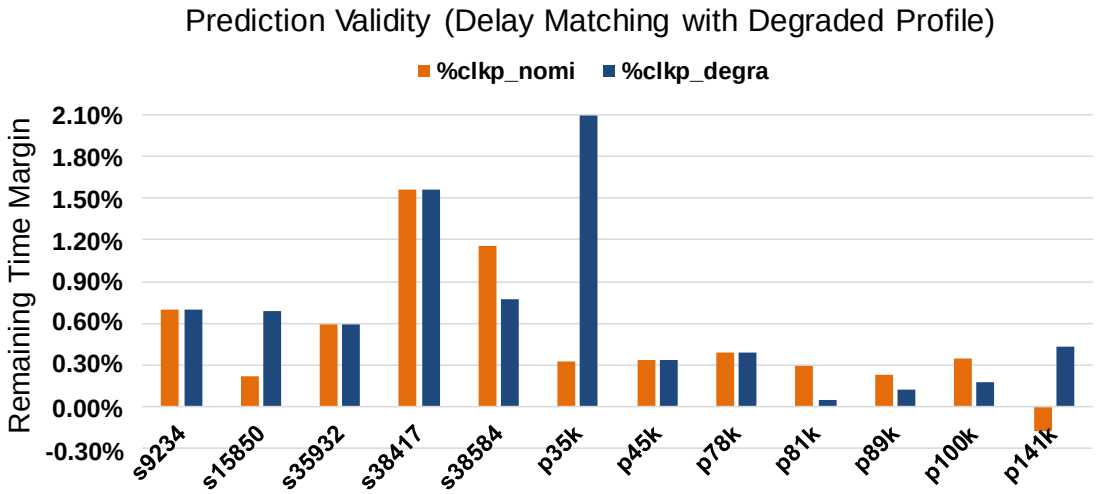
Delay Matching based on Degraded Timing Profile

By using the degraded timing profile for delay matching, the unpredictable or potential failures (red or orange lines in Fig. 6.9) can be warned in time. The blue, green and red points in Fig. 6.11 are identical as in Fig. 6.9. We denote the time of the first alert obtained under the degraded timing profile as New_alert (blue circles in Fig. 6.11). Compared to the nominal profile case, monitors in s15850, p35k and p141k trigger an alert earlier, i.e. blue circles of those circuits are under the blue dots. With the degraded timing profile, aging alerts (blue circles) are always generated before any actual or potential failure (red and green dots). Therefore the red and orange lines in Fig. 6.9 are replaced with blue lines. The predictability holds for all benchmarks. The remaining time margins of such circuits become larger (Fig. 6.12).

Furthermore, for rest of the circuits (e.g. s38584, p81k, p89k and p100k) the time of the first alert (blue circle) is less pessimistic, i.e. the time of alerts get closer to the occurrence time of imminent failures. Therefore, we observe a reduction of remaining timing margin for those circuits in Fig. 6.12.



▲ **Figure 6.11** — Failure predictability validation of observation points (with degraded timing profile)



▲ **Figure 6.12** — Prediction validity of observation points (with degraded timing profile)

6.5 Summary

This chapter presents a novel delay monitor placement method, reducing both the measurement latency and monitor overhead. The method utilizes the inverted clock as the unified reference timing for detection window generation of all monitors, therefore avoiding global wiring and clock balancing problems induced by monitor insertion at arbitrary nets. The proposed algorithm takes target path branches in the circuit structure and gate sensitization probability during operation into account. The results show that our approach reduces the number of required monitors by up to 98%. Due to the high measurement frequency at OPs, unmonitored timing violations can be avoided. In the experimental validation, the comparison of alerts and failure occurrences shows that this cost-efficient placement effectively indicates imminent timing failures.

CHAPTER 7

Aging Monitor Reuse for Small Delay Fault Testing

Small delay faults (cf. Section 2.1.2) receive more and more attention, since they indicate the marginality of the circuits which may experience the timing failures at operational speed or face reliability issues after the shipment. If the magnitude of a fault is so small that the fault effect can be masked by all slacks of paths through the fault site, we call it a *hidden delay fault* (cf. Section 2.1.2). Although such hidden delay faults do not violate operation timing at the moment of production, the faults might magnify during subsequent aging in the field after a short operation time and lead to an early life failure.

Industry usually use the standard scan-based structures (cf. Section 2.3.1) and perform at-speed and faster-than-at-speed tests (FAST) to target small delay faults. However, when we run tests at faster-than-at-speed frequencies, signals propagating along the long paths may not reach their stable states at the sampling time and X-values occur in the test responses (cf. Section 4.2). To evaluate the test responses with the ATE or FAST-BIST framework, a complex X-handling structure for test results compaction or evaluation (e.g. an X-tolerant MISR [Singh10, Helle14]) and a on-chip storage of intermediate signatures for offline analysis are required. The challenges in test evaluation make the FAST a rather expansive task.

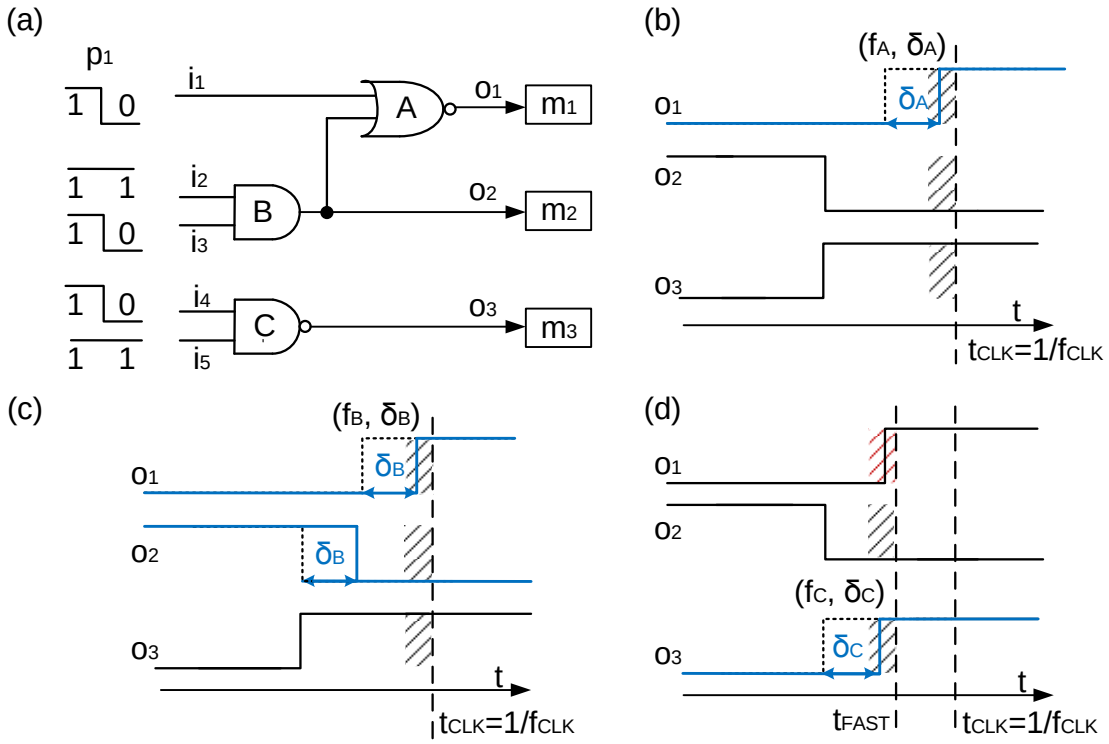
To ensure a high reliability of the microelectronic hardware, aging monitors are intensively studied (cf. Chapter 4 and 5) and widely integrated on-chip. Since both aging and delay faults change the circuit path delay, as an extension of [Liu17] this work screens both issues by detecting the deviated delay with a uniform structure. Without losing the generality, delay detecting flip-flops (cf. 4.4.2) are taken as an example. Unlike the traditional delay tests, instead of sampling and evaluating the logic value of outputs at a particular time, the signal stability of such outputs are measured during the predefined detection period by aging monitors. If a faulty transition propagated from a sensitized path occurs during the detection window of a monitor, a delay fault is detected.

In this case, monitors can directly evaluate the test results without any X-tolerant structure. The monitor alerts indicate whether the circuits pass or fail each test case, therefore, the test results require no response compaction and the data transformation (i.e. sending the test results to the FAST-BIST, ATE or other test controllers) is significantly simplified. Additionally, since the detection window is designed as a period before the clock sampling time, with monitor reuse some hidden delay faults violating the detection window are detectable at nominal speed, while they were before only detected by FAST. However, many hidden faults still remain unobservable.

For a high coverage of hidden delay faults, FAST is performed with the monitor reuse approach at a number of predefined frequencies. The main challenge of the aging monitor reuse for FAST is the possible occurrences of false alerts at higher frequencies. While a particular test vector pair makes a delay fault observable at one monitor, it may also trigger transitions violating the detection window of another monitor in the fault-free case. Thus, the transitions which activate the false positive alerts of monitors require masking.

Fig. 7.1 shows how delay monitors detect hidden delay faults and how monitor false alerts occur at faster-than-at-speed frequencies through an example. (f_A, δ_A), (f_B, δ_B) and (f_C, δ_C) are our target small delay faults, located in gate A, B and C respectively. δ_A, δ_B and δ_C are the fault magnitude of f_A, f_B and f_C. We use the pattern pair p_1 for testing and assume a single fault may exist in the circuit of Fig. 7.1 (a). Fig. 7.1 (b)-(d) present the corresponding waveforms at all outputs. The shadow regions on waveforms depict the detection windows of monitors and the dash lines refer to the signal waveforms in the fault free case. If (f_A, δ_A) or (f_B, δ_B) exists in the circuit (Fig. 7.1 (b) and (c)), the faulty transition at output o_1 can be observed by the detection window of monitor m_1. Such hidden delay faults are detected at the nominal frequency, while conventionally

their detection requires FAST. As shown in Fig. 7.1 (d), due to the small fault size and short length of the propagation path the fault effect of f_C is invisible at nominal speed. To detect f_C, we reuse monitors at a higher test frequency. Since the size of the detection window is designed to be a constant (cf. Section 4.4.2) and the clock has a lower period t_{FAST}, the detection window moves to the left of the time axis. In this case, the faulty transition of f_C is detected by m_3. However, at the same time, the fault-free transition (in the following called *good transition*) at o_1 violates the detection window of m_1 (highlighted in red) and triggers a false alert.



▲ **Figure 7.1** — The transition at t_1 in the fault-free circuit causes a false alert

To overcome the false alerts of monitors, the proposed approach either removes the pattern pair that launches the culprit transition from the original test set or disables the corresponding monitor as explained in Section 7.1.1. The task is modeled as a multidimensional optimizing problem: minimizing the masking overhead and the number of test vectors while maximizing delay fault coverage w.r.t. a group of predefined FAST frequencies.

The contributions of this work are:

- a novel approach is proposed for hidden delay faults detection by *reuse of aging monitors* for at-speed and faster than at-speed delay test;
- to prevent false alerts of monitors during FAST, a monitor and test pattern selection scheme is introduced which tailors the test pattern set and selectively enables the monitors for each test frequency;
- to maximize the fault coverage, the monitor and pattern selection is modeled as a Pseudo-Boolean optimization problem and solved by a SAT-solver.

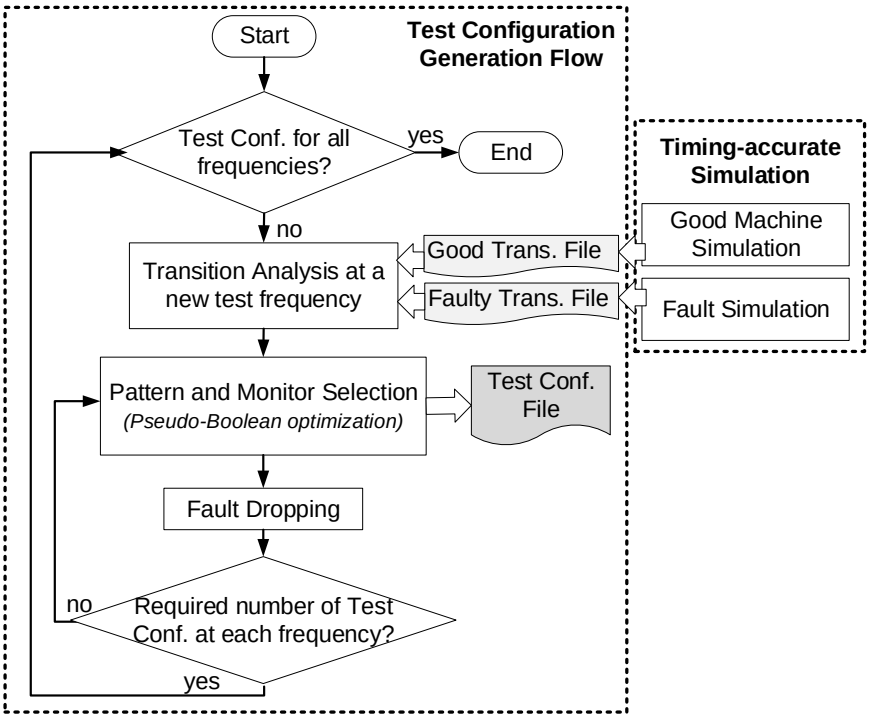
In Section 7.1, the monitor and pattern selection is formulated as a Pseudo-Boolean optimization problem for fault coverage maximization. Then in Section 7.2, the experimental setup and results are discussed.

7.1 Monitor and Pattern Selection for FAST

We aim to detect as many hidden delay faults as possible by reuse of aging monitors at the nominal and predefined FAST frequencies. The test procedure starts from the nominal speed and the frequency increases step by step. Usually, the detection window size of monitors is smaller than the slack of paths. Thus good transitions trigger no false alert at nominal speed. To prevent false positive detections at monitors during FAST, the related pattern pairs are removed from the test set or the corresponding monitors are deactivated. We call the selected patterns and the assignments of monitor enable signals a *test configuration*.

All test configurations for different test frequencies are generated by the workflow shown in Fig. 7.2 and stored in the *Test Conf. File*. For each test frequency, the timing information of good and faulty transitions are read in and analyzed according to the detection window of each monitor at such frequency. The timing information in the *Good and Faulty Trans. Files* is collected by timing-accurate good machine and fault simulation. Based on the transition analysis, Pseudo-Boolean optimizations are performed for pattern and monitor selection. Each Pseudo-Boolean optimization generates one test configuration with maximal coverage of hidden delay faults at a particular test frequency. The *Test Conf. File* is extended and the fault dropping is performed after each optimization step. The subsequent test configuration is generated targeting the remaining undetected faults. A certain number of test configurations

are generated at each frequency. The flow iterates until test configurations have been generated for all predefined test frequencies.



▲ **Figure 7.2** — Test configuration generation flow

7.1.1 Modeling for Pseudo-Boolean Optimization

To generate the test configurations, we model the pattern and monitor selection into a Pseudo-Boolean optimization problem. We define the objective function for maximization as the number of detected target faults and translate the conditions preventing monitor false positive detections to Boolean constraints. A monitor detects a fault or generates a false alert when a transition lies in its detection window. Therefore, the focal points of this section are modeling transitions and building up the relation between the transitions and the corresponding patterns and monitors.

We assume a set of delay monitors $M = \{m_1, m_2, \ldots, m_S\}$ has been implemented at the end of critical or long paths in the circuit under test. We target the hidden delay faults in set $F = \{f_1, f_2, \ldots, f_K\}$ for testing and keep the single fault assumption. We use a test set $P = \{p_1, p_2, \ldots, p_Q\}$ with Q pattern pairs to detect target faults at preselected FAST frequencies.

We model a pattern pair p_i in the test set P with a Boolean variable pb_i. i is the index of pattern pairs, $i \in \{1, 2, \ldots, Q\}$. If $pb_i = 1$, the pattern pair p_i is applied at the currently considered FAST frequency. If $pb_i = 0$, the pair p_i is removed from the test set for that FAST frequency. Similarly, we represent a monitor m_j in M with a Boolean variable mb_j. j is the index of monitors, $j \in \{1, 2, \ldots, S\}$. Variable $mb_j = 1$ if monitor m_j is enabled at the currently considered frequency. In timing simulation, if pattern pair p_i causes the input of monitor m_j to toggle during the detection window, we model this transition with a Boolean variable tr_l and the equivalence: $tr_l \Leftrightarrow pb_i \wedge mb_j$. l encodes the indices (i, j). If $tr_l = 1$, a monitor alert is generated at m_j when p_i is applied.

- If the transition is from the good machine simulation, we denote it as tr_l^g. $tr_l^g = 1$ indicates that a good transition triggers a false positive detection at the monitor. To prevent this, we provide the Boolean constraint $\bigvee_{l=1}^{L} tr_l^g = 0$, $l \in \{1, \ldots, N_g\}$, N_g is the number of transitions that trigger monitor false alerts, obtained from good machine simulation. Because such good transition can be represented by the monitor and pattern combination above, we replace tr_l^g with $pb_i \wedge mb_j$ in the Boolean constraint and get $\bigvee_{l=1}^{N_g} (pb_i \wedge mb_j) = 0$.
  We negate the equation at both sides, and finally get the term that needs to evaluate to true:
  $$\varphi_a : \bigwedge_{l=1}^{N_g} (\overline{pb_i} \vee \overline{mb_j}) \;\; (i, j) \mapsto l \tag{7.1}$$
- If the transition is the effect of a fault $f_k \in F$, we denote it as tr_l^k. k is the index of faults, $k \in \{1, \ldots, K\}$. We assume a particular fault magnitude for optimization and perform K fault simulations to collect the faulty transitions at all monitors. When $tr_l^k = 1$, a faulty transition of f_k is detected by a monitor. We model the detection of a fault f_k with a Boolean variable d_k. $d_k = 1$, when the fault f_k is detected. A fault f_k is detected by observing at least one of its faulty transitions, therefore $d_k \Rightarrow \bigvee_{l=1}^{N_k} tr_l^k$, $k \in \{1, \ldots, N_k\}$. N_k is the number of faulty transitions resulting from the fault f_k. After transformation we obtain:
  $$\varphi_b(f_k) : \overline{d_k} \vee (\bigvee_{l=1}^{N_k} tr_l^k) \tag{7.2}$$
  Additionally, the Boolean equivalence $tr_l^k \Leftrightarrow pb_i \wedge mb_j$ can be transformed to $(\overline{tr_l^k} \vee (pb_i \wedge mb_j)) \wedge (tr_l^k \vee \overline{pb_i \wedge mb_j})$, and into conjunctive normal form (CNF) as:
  $$\varphi_c(f_k, l) : (\overline{tr_l^k} \vee pb_i) \wedge (\overline{tr_l^k} \vee mb_j) \wedge (tr_l^k \vee \overline{pb_i} \vee \overline{mb_j}) \tag{7.3}$$

To optimize the fault coverage at each test frequency, we maximize the pseudo-Boolean function $\sum_{f_k \in F} d_k$. The Boolean constraints of Eq. 7.1 to Eq. 7.3 need to be satisfied for all faults:

$$\varphi_a \wedge (\bigwedge_{f_k \in F} (\varphi_b(f_k) \wedge \bigwedge_{l=1}^{N_k} \varphi_c(f_k, l))).$$

To generate the test configuration, i.e. the selected patterns and monitors at a certain test frequency, the objective function and Boolean constraints are analyzed by a SAT solver. The Boolean assignments of pb_i and mb_j determine which pattern pairs are applied and which monitors are active.

We use the circuit in Fig. 7.3 (a) as an example to explain the modeling process above. Monitors are placed at outputs. The hidden delay faults (f_A, δ), (f_B, δ) and (f_C, δ) locate in gate A, B and C respectively and have an identical fault magnitude δ. Pattern pairs p_1 and p_2 are applied to detect the target faults at a preselected FAST frequency $1/t_{FAST}$. Fig. 7.3 (b) and (c) present the fault-free and faulty waveforms at all outputs for each pattern pair under the single fault assumption. The detection windows of monitors have an identical size and are depicted as shadow regions. The fault-free transitions from good machine simulations are denoted as Tr^g and presented as black lines. The faulty transitions from fault simulations are marked as Tr^f and in blue. The index of a transition encodes the indices of the pattern pair and monitor, e.g. the transition generated by p_1 and propagated to m_2 is denoted as Tr_{12}.

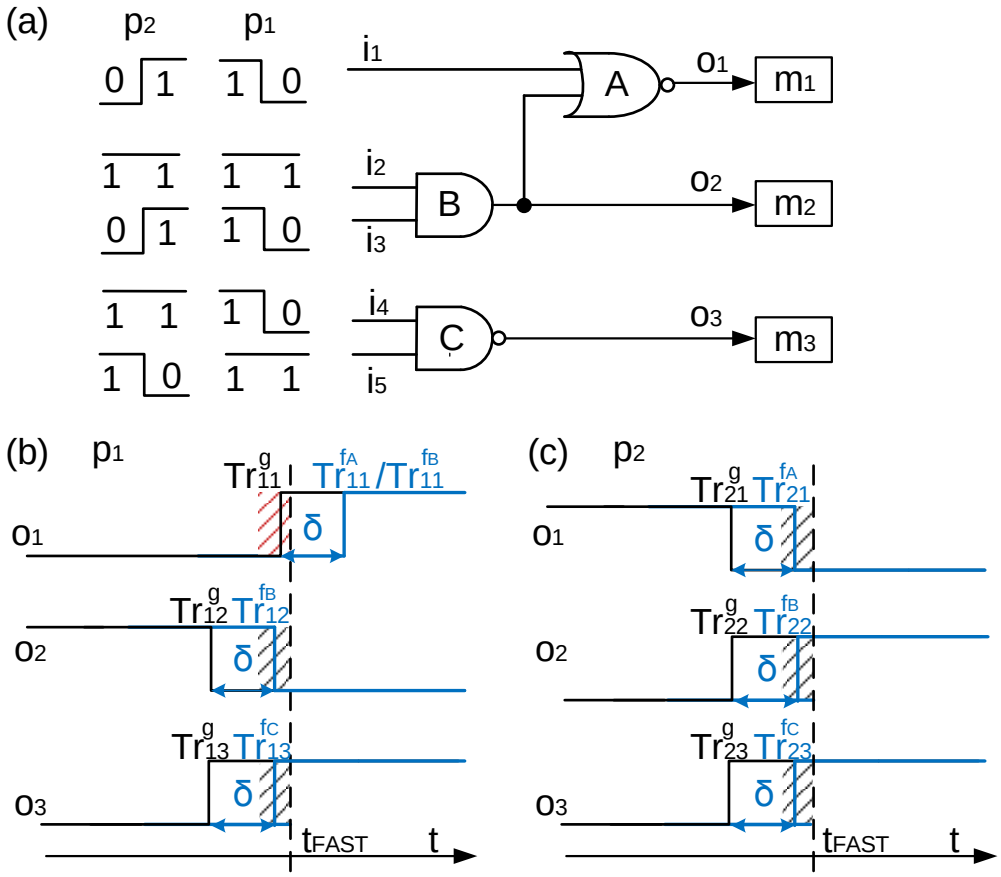
As shown in Fig. 7.3 (b), the good transition Tr^g_{11} violates the detection window of m_1 (highlighted in red) and will cause a false alert. To prevent this false positive detection, we need to either disable m_1 or remove p_1 from the test set. We apply the modeling procedure mentioned above for a better option detecting more faults. We use Boolean variables pb_1 and pb_2 to model pattern pairs. $pb_1 = 1$ when pattern pair p_1 is applied. Boolean variables mb_1, mb_2 and mb_3 model the monitors with the identical index. When $mb_1 = 1$, monitor m_1 is enabled. Boolean variables tr^g_l represent the good transitions Tr^g_l and tr^f_l refers to the faulty ones. When $tr_l = 1$, a transition violates the detection window and triggers a monitor alert.

The transitions outside the detection window neither trigger any false alters nor detect any faults. Their representing Boolean variables always have the Boolean value "false" and do not influence the optimization. Therefore, we discuss only the transitions observable to monitors. Tr^g_{11} is the only good transition need to be considered. $tr^g_{11} \Leftrightarrow$

$pb_1 \wedge mb_1$. To avoid the monitor false positive detection a Boolean constraint

$$\varphi_a : \overline{pb_1} \vee \overline{mb_1}$$

is generated according to Equ. 7.1.



▲ **Figure 7.3** — An example of the monitor pattern selection method

Then, we analyze the faulty transitions in the detection window. We use the Boolean variable d_f to denote a detected target fault f. When the fault f is detected, $d_f = 1$. Since fault f_B can be detected by observing either of its two faulty transitions $Tr_{12}^{f_B}$ and $Tr_{22}^{f_B}$, We obtain

$$\varphi_b^{f_B} : \overline{d_{f_B}} \vee tr_{12}^{f_B} \vee tr_{22}^{f_B}$$

according to Equ. 7.2. The two faulty transitions of f_B can be represented by the pattern and monitor combinations. From Equ. 7.3, we acquire

$$\varphi_{c1}^{f_B} : (\overline{tr_{12}^{f_B}} \vee pb_1) \wedge (\overline{tr_{12}^{f_B}} \vee mb_2) \wedge (tr_{12}^{f_B} \vee \overline{pb_1} \vee \overline{mb_2})$$

$$\varphi_{c2}^{f_B} : (\overline{tr_{22}^{f_B}} \vee pb_2) \wedge (\overline{tr_{22}^{f_B}} \vee mb_2) \wedge (tr_{22}^{f_B} \vee \overline{pb_2} \vee \overline{mb_2})$$

The conjunction of all constraints of f_B is denoted as $\varphi_{f_B} : \varphi_b^{f_B} \wedge \varphi_{c1}^{f_B} \wedge \varphi_{c2}^{f_B}$.

Similarly for other faults, fault f_A can only be detected by observing $Tr_{21}^{f_A}$ when p_2 is applied, thus we get:

$$\varphi_b^{f_A} : \overline{d_{f_A}} \vee tr_{21}^{f_A}$$

$$\varphi_{c1}^{f_A} : (\overline{tr_{21}^{f_A}} \vee pb_2) \wedge (\overline{tr_{21}^{f_A}} \vee mb_1) \wedge (tr_{21}^{f_A} \vee \overline{pb_2} \vee \overline{mb_1})$$

The conjunction of all constraints of f_A is denoted as $\varphi_{f_A} : \varphi_b^{f_A} \wedge \varphi_{c1}^{f_A}$.

For fault f_C, we obtain the constraints:

$$\varphi_b^{f_C} : \overline{d_{f_C}} \vee tr_{13}^{f_C} \vee tr_{23}^{f_C}$$

$$\varphi_{c1}^{f_C} : (\overline{tr_{13}^{f_C}} \vee pb_1) \wedge (\overline{tr_{13}^{f_C}} \vee mb_3) \wedge (tr_{13}^{f_C} \vee \overline{pb_1} \vee \overline{mb_3})$$

$$\varphi_{c2}^{f_C} : (\overline{tr_{23}^{f_C}} \vee pb_2) \wedge (\overline{tr_{23}^{f_C}} \vee mb_3) \wedge (tr_{23}^{f_C} \vee \overline{pb_2} \vee \overline{mb_3})$$

The conjunction of all constraints of f_C is $\varphi_{f_C} : \varphi_b^{f_C} \wedge \varphi_{c1}^{f_C} \wedge \varphi_{c2}^{f_C}$.

Finally, we need to maximize the pseudo-Boolean function

$$d_{fA} + d_{fB} + d_{fC}$$

and ensure

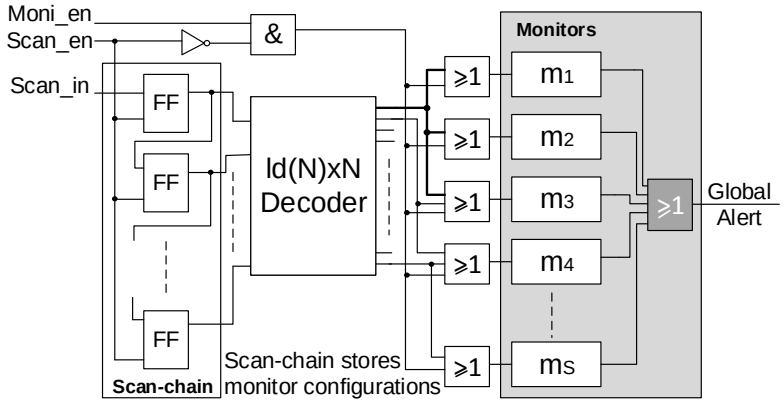
$$\varphi_a \wedge \varphi_{f_A} \wedge \varphi_{f_B} \wedge \varphi_{f_C}$$

is satisfied. To satisfy φ_a, either p_1 needs to be removed or m_1 is deactivated. When we disable m_1, only f_B and f_C remain detectable. However, by removing p_1, we can still detect all the faults. In this case removing the patter pair is a better option. By solving the pseudo-Boolean optimization problem, i.e. by maximizing the objective function and satisfying SAT instances mentioned above, we receive the same result.

7.1.2 Test Control Hardware

If the proposed monitor reuse approach is employed in a self-test or embedded deterministic test scenario, the patterns and test configuration information can be stored in the system in compressed form. The patterns are applied at the corresponding FAST frequencies according to the test configurations.

To setup the monitors in one test configuration, the on-chip control structure shown in Fig. 7.4 enables the selected monitors based on the test configuration index. This configuration index is shifted in from the *Scan_in* signal. The decoding logic translates the index number into control assignments of monitors. The monitor on/off state remains unchanged for all the pattern pairs in the selected test configuration during testing.



▲ **Figure 7.4** — Test control hardware for monitor selection

To store N test configurations, $\lceil ld(N) \rceil$ scan flip-flops are required. The output of the OR gates is connected to the enable signal of the monitor m_i. During aging prediction, *Scan_en* is set to low and *Moni_en* is used to periodically start the degradation measurement with all monitors. The OR gate (grey) collects *Detection Alert* (Fig. 4.2) of all monitors and generates a *Global Alert* if any monitor alert is issued.

When *Scan_en* is high, the circuit is switched into test mode. The values of the flip-flops control which test configuration is applied. In every test, one of the outputs of the decoder is high and at the same time all monitors connected to this end are enabled. For example (Fig. 7.4), when the first output (count from above) of the decoder is high, monitors m_1, m_2, m_3 are activated.

The monitors and grey OR gate are integrated on-chip and reused. *Global Alert* indicates the test results. If required for diagnosis, the shadow registers (Fig. 4.2 (b)) can be included in a standard scan-chain. The delayed outputs (Q') of shadow registers are read out only when the test case fails.

Thus, the hardware overhead is only induced by monitor activation control logic, which consists of $\lceil ld(N) \rceil$ scan flip-flops, S (S: number of monitors) OR gates, one inverter, one AND gate and a $\lceil ld(N) \rceil \times N$ decoder.

7.2 Experimental Evaluation

In the experiments, we generate a compacted test pattern set targeting transition delay faults using a commercial tool [tet]. We assume monitors are placed at 25% of the pseudo outputs at long path ends [Agarw08]. When generating the test configurations for the monitor-based delay test, we target slow-to-rise and slow-to-fall small delay faults at inputs or outputs of all gates in the input cone of monitors. The delay fault size is set to 6σ, where σ is the standard deviation of the normal distribution of the process variation and set to 0.2 of the nominal gate delay. The nominal clock period ($clkp$) is set to the critical path length plus a small time margin of 5%. Eight preselected FAST frequencies are evenly distributed from nominal (f) to three times of nominal frequency ($3f$). The detection window of the monitors is set to 50 ps.

The experiment is performed for ISCAS89 [Brgle89], ITC99 [itc] and NXP [nxp] benchmarks. The basic information of the circuits such as: the critical path length, number of patterns in the original test set, the number of monitors on-chip, the number of gates in the combinational logic and the number of flip-flops can be found in Appendix D Table D.1.

7.2.1 Generation of Test Configurations

Monitor and pattern selection allow to mask false alerts of monitors, but at same time reduce the observability of some detection effects. Some faults may only be activated by specific patterns and detectable at certain monitors. For a high coverage of target faults, two test configurations are generated for each of the eight FAST frequency to alleviate this effect. In this case, $N = 16$ test configurations are controlled by a 4 bit scan-chain and 4x16 decoder (Fig. 7.4). Every pseudo-Boolean optimization generates one test configuration.

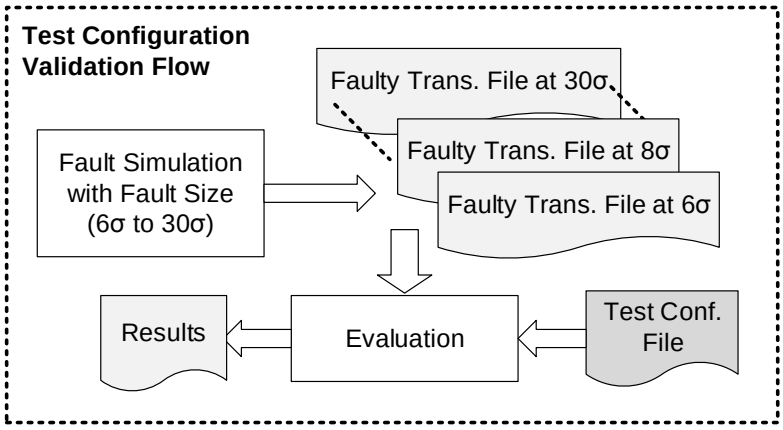
The method is implemented in Java and uses the Sat4j library. It is executed on an intel Xeon core with 3.33 GHz.

The computational effort of the Pseudo-Boolean optimization depends on the number of monitors, patterns, false positive activations of monitors (i.e. good transitions occur during the detection window) and the number of detection effects (i.e. faulty transitions happen in the detection window) at a particular FAST frequency. For the feasibility of our experiments the optimization procedure is aborted when a timeout of 1 hour is reached. Fault dropping is done after each optimization, i.e. test configurations are

generated for the remaining uncovered faults from the previous configurations (Fig. 7.2). The runtime of the entire procedure is dominated by the exhaustive timing accurate fault simulation [Schne16] shown in Fig. 7.2. The runtime of the Pseudo-Boolean optimization is constrained by the timeout.

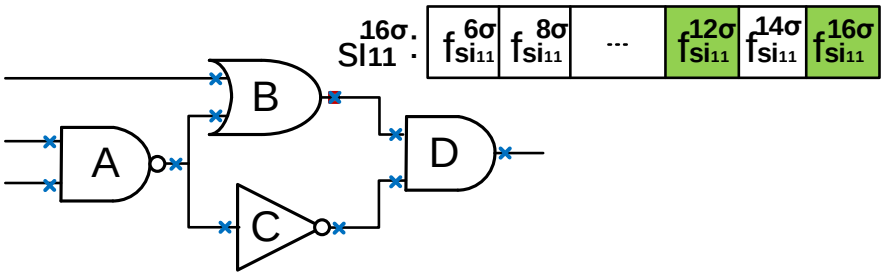
7.2.2 Covered and Uncovered Fault Sites w.r.t. Various Fault Magnitudes

As shown in Fig. 7.5, to validate the effectiveness, fault simulations are performed for different fault sizes from 6σ to 30σ, with increments of 2σ. The *Test Conf. File* generated for hidden delay faults with fault size 6σ (cf. Sec. 7.2.1) is evaluated by the coverage of target fault sites with variant fault sizes. The target fault sites include all delay faults at inputs and outputs of gates in the input cone of monitors.



▲ **Figure 7.5** — The validation flow for test configurations

A fault at fault site si_x with size δ is denoted as $f^{\delta}_{si_x}$, where x is the fault site index. We say a fault site si^{δ}_x is *covered*, if at least one of the faults located at the fault site si_x with fault size smaller than or equal to δ is detected. For instance in Fig. 7.6, all possible positions for delay fault insertion are marked by the blue crosses. Two fault sites (slow-to-rise and slow-to-fall) exist in each position. Thus, there are 22 target fault sites in the example. $si^{16\sigma}_{11}$ is the slow-to-rise fault site at output of gate *B*. $si^{16\sigma}_{11}$ is covered if any of the faults from $f^{6\sigma}_{si_{11}}$ to $f^{16\sigma}_{si_{11}}$ is detected. In this case (Fig. 7.6), due to the detected faults $f^{12\sigma}_{si_{11}}$ and $f^{16\sigma}_{si_{11}}$, $si^{16\sigma}_{11}$ is covered.



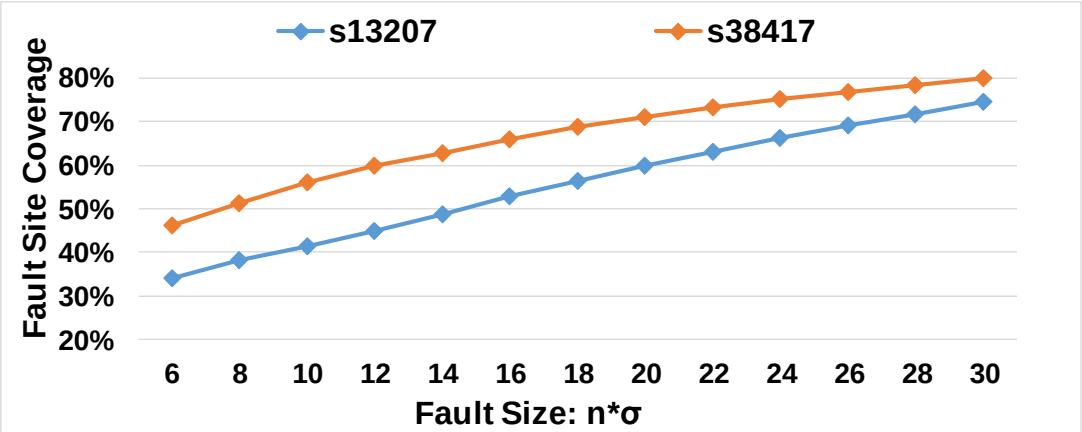
▲ **Figure 7.6** — An example of the fault site coverage

The 16 test configurations in the *Test Conf. File* (Fig. 7.5) are applied for all fault sizes (6σ to 30σ). The fault sites covered by the eight fixed frequencies w.r.t. a certain fault size δ are collected.

For two exemplary circuits (s13207 and s38417), Fig. 7.7 depicts the *fault site coverage* cov^{δ} which is the number of covered fault sites as the percentage of the number of total target fault sites.

$$cov^{\delta} = \frac{\#CoveredFaultSites}{\#TargetFaultSites} \cdot 100\%$$

where $\#TargetFaultSites$ is a constant independent of the fault magnitude, listed in Col. $\#fault\_site$ Table D.4. The covered fault sites have at least one fault with fault size 6σ to 30σ detected by at-speed delay test or the monitor reuse method. cov^{δ} increases when the fault size δ grows. Even if a hidden delay fault with small size is not detected initially, the monitors detect it if it grows in magnitude. The fault site coverage of all benchmarks can be found in Table D.2 in Appendix D.

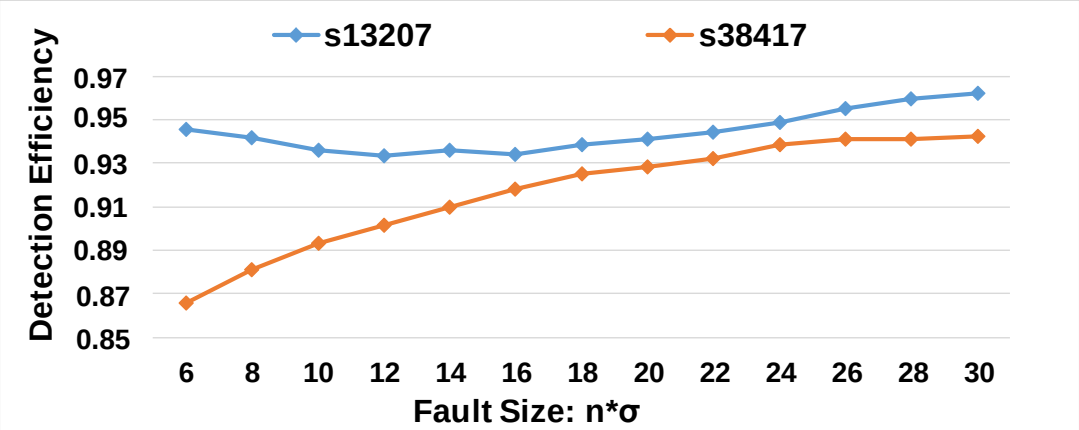


▲ **Figure 7.7** — The fault site coverage w.r.t. different fault size

Multiple reasons constrain the coverage of fault sites. For instance, if the test pattern set does not sensitize a path through the fault site to a monitor with sufficiently small slack, the fault site cannot be covered. To evaluate the monitor reuse approach with less impact of test patterns, we calculate the *detection efficiency* of the method. Fig. 7.8 shows the detection efficiency of two circuits with the different fault sizes. the detection efficiency is defined as the ratio of the number of detected fault sites to the number of detectable ones:

$$\text{efficiency} = \frac{\#DetectedFaultSites}{\#DetectableFaultSites}$$

where *#DetectableFaultSites* shows the upper limit of the number of fault sites that can possibly be detected by the monitor reuse approach. For this upper limit, we assume that each monitor can be selected individually in each cycle to reach the best observability of faults and at the same time prevent all false alerts of monitors. However, in practice, configuring monitors cycle per cycle is very costly. Thus, in our approach, monitors are controlled by test configurations that are applied per test frequency. *#DetectedFaultSites* provides the number of covered fault sites with the 16 computed test configurations.

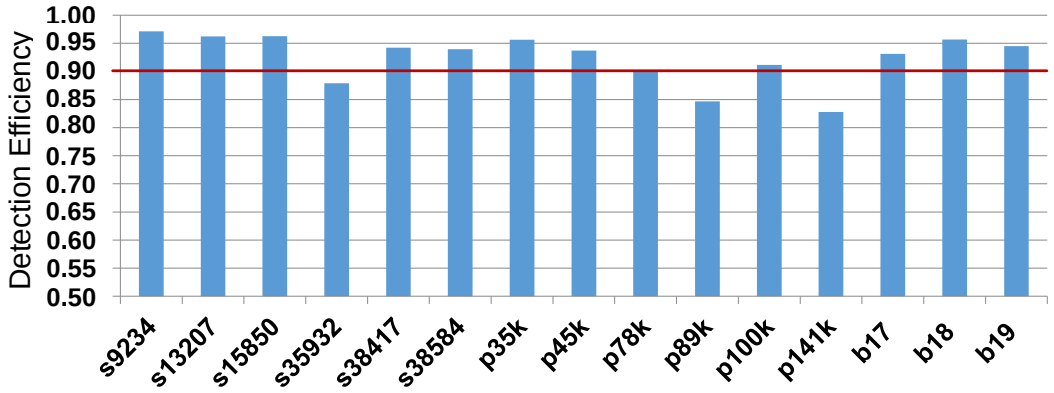


▲ **Figure 7.8** — The detection efficiency of monitors w.r.t. different fault size

For both circuits and all fault sizes, the efficiency is high and ranges from 0.865 to 0.962. The detection efficiency details of all benchmark circuits are attached in Appendix D Table D.3.

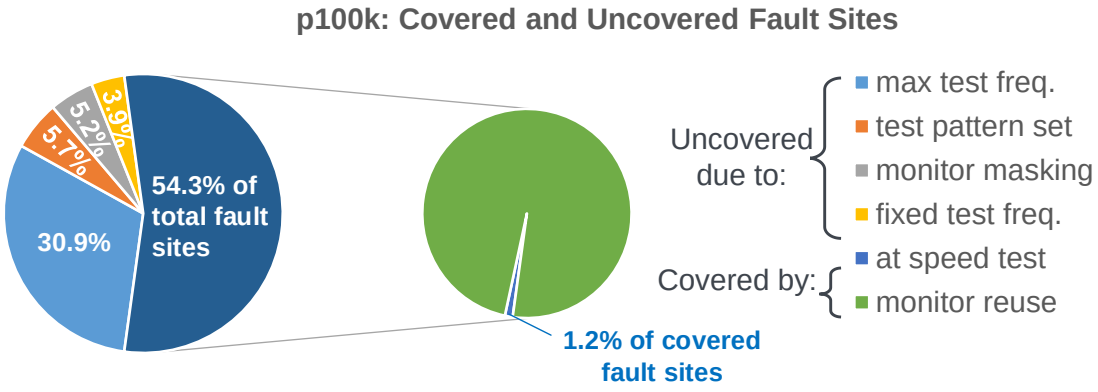
Fig. 7.9 represents the fault detection efficiency with the magnitude of 30σ. The efficiency ratio ranges from 0.828 to 0.971 and is above 0.9 for most of the circuits.

The number of target fault sites as well as the absolute number of covered, detectable and detected fault sites with the magnitude of 30σ for all circuits are listed in Table D.4.



▲ **Figure 7.9** — The detection efficiency of monitors with fault size 30σ for different benchmarks

Fig. 7.10 shows two pie charts of the circuit p100k. The left chart represents the target fault sites. About 54.3% of them are covered (dark blue region) by either at-speed delay test or the monitor reuse. How each test method contributes the fault detection is illustrated by the right chart. As shown, most fault sites remain hidden w.r.t. the given fault size 30σ (i.e. the fault effect can be masked by the smallest slack of the activated paths through the fault site at the operation frequency). Furthermore, 98.8% of faults are detected by the monitor reuse method.



▲ **Figure 7.10** — The covered and uncovered fault sites of the circuit p100k

For those amount of uncovered fault sites (*#uncovered* in Table D.5), we analyzed in more detail the reasons that limits the detection of target fault sites and categorized the fault sites accordingly. We say a fault site is outside range (*#outside_range*) if the maximum fault size 30σ plus the length of the longest topological path through the fault

site is smaller than the minimum test period $1/3clkp$, i.e. the faults at *outside range* sites can never be observed by a test frequency lower than or equal to $3f$. $\%outside\_range$ is $\#outside\_range$ as the percentage of the number of target fault sites. For p100k, ca. 30.9% of target faults are undetectable due to the maximum test frequency (light blue region in Fig. 7.10).

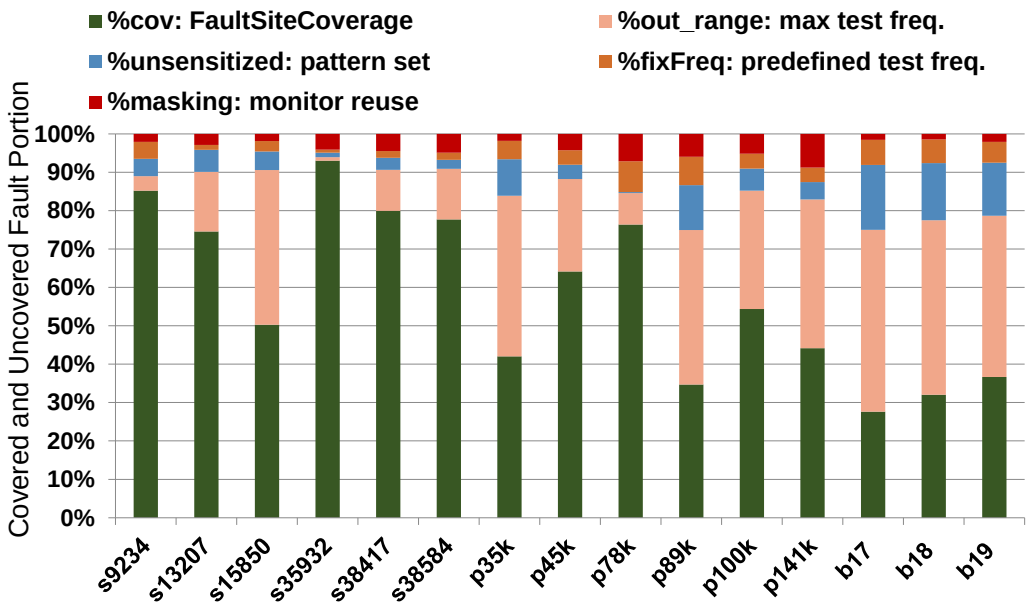
A fault site is unsensitized if no pattern in the test pattern set activates a path from the fault site to a pseudo output with monitor. Similarly, $\#unsensitized$ is the number of unsensitzed fault sites and $\%unsensitized$ is percentage w.r.t. the target set. Some fault sites cannot be covered due to the preselected test frequencies. If no faulty transition through a certain fault site can trigger any monitor alert with all possible fault sizes at all frequencies, the fault site is sorted into frequency constraint. $\#freq\_constraint$ and $\%freq\_constraint$ denote the number and the percentage of them. In Fig. 7.10, we can see ca. 5.7% faults (orange region) are undetected due to the test pattern set in the experiment and about 3.9% fault sites (yellow part) are invisible to those eight preselected test frequencies. The remaining fault sites ($\#masking$) are uncovered because of the pattern and monitor selection for false alert masking. $\%masking$ is about 5.2% for p100k (gray part in Fig. 7.10).

Fig. 7.11 depicts the percentage of covered and uncovered fault sites for each benchmark. As shown in the results, the portion of fault sites uncovered due to pattern and monitor selection (red part) is quite low, about 1.4% - 8.8% of the target fault sites. The *outside range* cause is the major reason for uncovered fault sites. For example for circuit p35k, 41.9% of the fault sites cannot be covered when the maximum FAST frequency is set to three times the nominal frequency. The principal limitations (i.e. the maximum frequency, patterns, predefined test frequencies and false alert masking) bound the coverage ratio to 42.0% for circuit p35k.

For more detail results, the absolute number of uncovered fault sites and the portions categorized by different limitations for all benchmarks are shown in Table D.5 Appendix D.

7.2.3 Hardware Overhead

The area of the circuit under test (CUT) A_{CUT} is calculated as the sum of the combinational, sequential and monitor area. The relative overhead $R_{overhead}$ is the hardware area induced by the monitor control logic A_{ctrl} as the percentage of the area of the

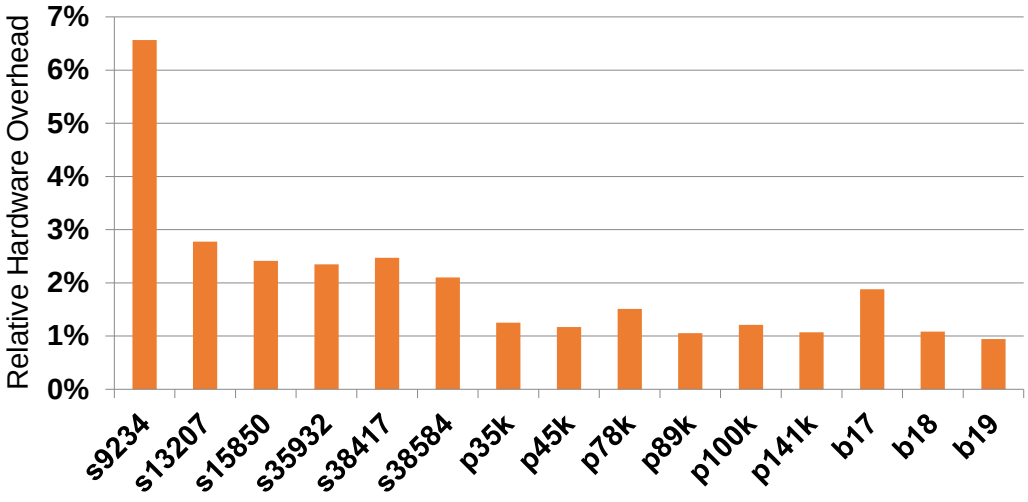


▲ **Figure 7.11** — The covered and uncovered fault sites of benchmarks

original design A_{CUT}.

$$R_{overhead} = \frac{A_{ctrl}}{A_{CUT}} \cdot 100\%$$

The overhead ranges from 0.9% to 6.6% of the benchmarks and decreases for larger circuits.



▲ **Figure 7.12** — The relative hardware overhead of benchmarks

7.3 Summary

In this chapter, we reuse in-situ aging monitors for efficient small delay fault tests. To mask the false alert of monitors at FAST, the relevant patterns of a test pattern set are selected, and the monitors are enabled or disabled according to the given test frequencies. Pattern and monitor selection is modeled as a Pseudo-Boolean optimization problem to maximize the fault coverage. Experimental results show that a high detection efficiency of small delay faults is achieved, ranging from 82.8% up to 97.1% with very limited hardware overhead.

CHAPTER 8

Conclusion and Future Research Directions

Technology scaling comes along with more device susceptibility to manufacturing imperfections, process variations, operational environments, intermittent faults and aging degradation. Consequently, it threatens the reliability of hardware components and systems. On the other hand, consumer requirements of the system reliability keep increasing due to the extremely high cost of the unplanned system outage, and especially in safety-critical applications.

Diverse techniques have been published and applied to improve the system reliability, especially during the two most vulnerable periods over the circuit lifetime: the early life and wear-out failure periods. However, certain conventional methods for the reliability improvement are rather expensive regarding the additional area and power overhead, the increased structural complexity, the extra test data volume and extended test time. Some others have limitations in the accuracy and timeliness of aging predictions, and accordingly, are less capable of supporting the countermeasure schemes to mitigate the degradation impact and enlarge the system lifetime.

The work at hand addresses the system reliability improvement during both the early-life time and wear-out period of the hardware. For the aging monitoring, the thesis contributes cost-effective solutions in monitor designs as well as their on-chip placement.

Then the entire aging prediction framework is re-utilized for testing small delay faults, which may indicate early life failures.

To predict the aging degradation before any measurable parameter deviation and to take proactive countermeasures for wear-out mitigation, this thesis introduces an innovative workload monitor to estimate the relevant workload-induced stress, i.e. the cause of the aging process, by observing the logic states of the circuit under test. Workload monitors can be constructed for various degradation mechanisms, and their synthesis flow is fully automated. Since only a subset of primary and pseudo-primary input nodes are buffered for circuit states observation, the workload monitor has a minimal impact on the mission logic. It is non-intrusive and applicable for IP cores and fixed macros. A case study of NBTI monitoring shows that an effective on-line stress estimation with an average error below 3.3% can be achieved with a low area cost of less than 1%.

To reduce the measurement latency, i.e. improve the prediction timeliness, and monitoring overhead, this thesis presents a novel placement method for commonly used delay monitors. Observation Points (OPs) are meticulously selected at intermediate circuit nets for delay monitor insertion, rather than integrating monitors only at the end of long paths. Since the selection algorithm takes the path prefix length, branching structures, and gate sensitization probability into account, a small number of monitors is sufficient to obtain a frequent and accurate measurement of the aging degradation in a high percentage of target paths. Choosing OPs from circuit nodes with topological depth closed to half clock period not only satisfies the requirements of measurement frequency and accuracy, but can also utilize the inverted clock to generate the detection window for all monitors. This avoids global wiring and clock balancing problems induced by monitor insertion at arbitrary nets. The experimental results show that the reduction of required monitors can reach up to 98%. The resulting, more frequent delay measurements at OPs prevent the unmonitored timing violations, and each monitor alert indicates an imminent timing failure.

In the last part of the thesis, in-situ aging monitors are reused for efficient small delay fault tests. Monitors can be used to evaluate the circuit responses to tests and mark failing cases with alerts. This allows to avoid the otherwise required complex X-tolerant on-chip structures for test response compaction, the high speed ATE or the BIST components for test evaluation. To mask possible false alerts of monitors, the thesis develops a pattern and monitor selection algorithm that tailors the original pattern set and controls the monitors according to the FAST frequencies. The selection

process is modeled in each given frequency as a Pseudo-Boolean optimization problem to maximize the fault coverage and guarantee no false-positive monitor activation. Experimental results show a high detection efficiency of small delay faults of up to 97.1% and a very limited hardware cost of less than 1% for large designs.

The results and methods presented in this work can be extended to consider additional constraints such as variations. Besides the manufacturing defects and aging processes, process variations and operation conditions also impact the circuit delay with an increasing importance for newer technology nodes. When process variation and operation conditions have to be taken into account, the tasks to efficiently detect small delay faults and predict imminent aging-induced failures become more challenging. A Monte-Carlo-based analysis or methods based on signal probability can be explored to obtain the target accuracy of degradation assessments or coverage of small delay faults with a user-defined confidence level. Additionally, more characteristics collected by various on-chip sensors, e.g. current, temperature or workload monitors, that support the proposed methods, may provide new possibilities to estimate the influence of variations.

The problem of masking the monitor false alerts for small delay fault detection is modeled as a Pseudo-Boolean Optimization with the constraint of predefined test frequencies. If selecting appropriate test frequencies is given as a new dimension of freedom for optimization, more efficient test procedures, e.g. fewer test frequencies, a lower maximum test frequency or fewer patterns, may be achieved with the assistance of novel BIST or on-chip clock generation structures. Furthermore, to avoid the yield loss due to the overtesting, delay deviations caused by IR-Drop or Current-Droop should be considered at high test speeds.

The observation points introduced for aging monitoring improve the observability and controllability of the mission logic and can be possibly reused for other purposes as well such as the delay tests, the small delay diagnosis and process characterization.

This work focuses on the reliability of the mission logic. However, more and more nonfunctional instruments are integrated on-chip ranging from design for test, for diagnosis, debug, reliability infrastructures to security controllers and play an essential role in the circuit operation and maintain. Consequently, the test, diagnosis, and reliability of the nonfunctional infrastructures will attract more attentions and research interests in the future.

Bibliography

[Agarw07] M. Agarwal, B. C. Paul, M. Zhang, and S. Mitra. Circuit Failure Prediction and Its Application to Transistor Aging. In *Proc. 25th IEEE VLSI Test Symposium (VTS)*, pages 277–286. May 2007. [pages x, x, 51, and 52]

[Agarw08] M. Agarwal, V. Balakrishnan, A. Bhuyan, K. Kim, B. Paul, W. Wang, B. Yang, Y. Cao, and S. Mitra. Optimized Circuit Failure Prediction for Aging: Practicality and Promise. In *Proc. IEEE International Test Conference (ITC)*, pages 1–10. Oct. 2008. [pages 50, 51, 55, 57, and 103]

[Ahmed06] N. Ahmed, M. Tehranipoor, and V. Jayaram. Timing-Based Delay Test for Screening Small Delay Defects. In *43rd ACM/IEEE Design Automation Conference (DAC)*, pages 320–325. 2006. [page 45]

[Ahmed09] N. Ahmed and M. Tehranipoor. A Novel Faster-Than-at-Speed Transition-Delay Test Method Considering IR-Drop Effects. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 28(10):1573–1582, Oct 2009. [pages 45, 46, and 48]

[Akers78] S. B. Akers. Binary decision diagrams. *IEEE Transactions on Computers*, 27(6):509–516, June 1978. URL https://doi.org/10.1109/TC.1978.1675141. [page 36]

[Alam03] M. A. Alam. A Critical Examination of the Mechanics of Dynamic NBTI for PMOSFETs. In *IEEE International Electron Devices Meeting*, pages 14.4.1–14.4.4. Dec 2003. [pages ix, 15, and 16]

[Amode05] M. Amodeo and B. Cory. Defining faster-than-at-speed delay tests. In *Cadence Nanometer Test Quarterly eNewsletter*. May 2005. [pages 47 and 48]

[Ander71] D. A. Anderson. *Design of Self-checking Digital Networks Using Coding Techniques*. Ph.D. thesis, Champaign, IL, USA, 1971. AAI7212065. [page 5]

[Baba09] A. Baba and S. Mitra. Testing for transistor aging. In *27th IEEE VLSI Test Symposium (VTS)*, pages 215–220. May 2009. [pages 55 and 88]

[Bahar93] R. I. Bahar, E. A. Frohm, C. M. Gaona, G. D. Hachtel, E. Macii, A. Pardo, and F. Somenzi. Algebraic Decision Diagrams and Their Applications. In *Proc. of IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 188–191. Nov 1993. [pages 38 and 60]

[Baran13] R. Baranowski, A. Cook, M. E. Imhof, C. Liu, and H. J. Wunderlich. Synthesis of workload monitors for on-line stress prediction. In *IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS)*, pages 137–142. Oct 2013. [pages 7, 9, and 58]

[Baran15] R. Baranowski, F. Firouzi, S. Kiamehr, C. Liu, M. Tahoori, and H.-J. Wunderlich. On-Line Prediction of NBTI-induced Aging Rates. In *Conference on Design, Automation and Test in Europe (DATE)*, pages 589–592. EDA Consortium, 2015. [pages 51, 73, and 76]

[Bhard06] S. Bhardwaj, W. Wang, R. Vattikonda, Y. Cao, and S. Vrudhula. Predictive Modeling of the NBTI Effect for Reliable Design. In *Proc. IEEE Custom Integrated Circuits Conference(CICC)*, pages 189–192. 2006. [page 88]

[Biere09] A. Biere, H. van Maaren, and T. Walsh. *Handbook of Satisfiability: Volume 185 Frontiers in Artificial Intelligence and Applications*. IOS Press, Amsterdam, The Netherlands, 2009. ISBN 1586039296, 9781586039295. [page 40]

[Birol04] A. Birolini. *Reliability Engineering - Theory and Practice*. Springer, 2004. ISBN 978-3-662-54208-8. [page 2]

[Black67] J. R. Black. Mass transport of aluminum by momentum exchange with conducting electrons. In *6th IEEE Annual Reliability Physics Symposium*, pages 148–159. Nov 1967. [page 26]

[Blome07] J. Blome, S. Feng, S. Gupta, and S. Mahlke. Self-calibrating online wearout detection. In *40th IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 109–122. Dec 2007. [pages 4, 50, and 51]

[Borka05] S. Borkar. Designing reliable systems from unreliable components: The challenges of transistor variability and degradation. *IEEE Computer Society*, 25(6):10–16, Nov 2005. [page 1]

[Borka08] S. Borkar. Probabilistic statistical design – the wave of the future. pages 69–79, 2008. URL http://dx.doi.org/10.1007/978-0-387-74909-9_5. [page 7]

[Boros02] E. Boros and P. L. Hammer. Pseudo-boolean optimization. *Discrete Applied Mathematics*, 123(1–3):155 – 225, 2002. URL http://www.sciencedirect.com/science/article/pii/S0166218X01003419. [pages 40 and 41]

[Brgle89] F. Brglez, D. Bryan, and K. Kozminski. Combinational profiles of sequential benchmark circuits. In *IEEE International Symposium on Circuits and Systems*, pages 1929–1934 vol.3. May 1989. [pages 86 and 103]

[Brown90] F. M. Brown. *Boolean Reasoning - The Logic of Boolean Equations*. Springer Science + Business Media, LLC, 1990. ISBN 978-1-4757-2080-8 (Print) 978-1-4757-2078-5 (eBook). [page 33]

[Bryan86] R. E. Bryant. Graph-Based Algorithms for Boolean Function Manipulation. *IEEE Transactions on Computers*, 35(8):677–691, August 1986. URL http://dx.doi.org/10.1109/TC.1986.1676819. [pages 36, 37, and 38]

[Cao11] Y. Cao. *Predictive Technology Model for Robust Nanoelectronic Design*. Springer, 2011. ISBN 978-1-4614-0444-6 (Print) 978-1-4614-0445-3 (Online). [pages 17 and 18]

[Carha53] R. R. Carhart. *A Survey of the Current Status of the Electronic Reliability Problem*. RAND Corporation, 1953. URL https://www.rand.org/pubs/research_memoranda/RM1131.html. [page 3]

[Carls07] R. Carlsten, J. Ralston-Good, and D. Goodman. An Approach to Detect Negative Bias Temperature Instability (NBTI) in Ultra-Deep Submicron Technologies. In *IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1257 –1260. 2007. [pages 49, 50, 54, and 57]

[Cha14] S. Cha, W. Kim, and L. Milor. Gate oxide breakdown parameter extraction with ground and power supply signature measurements. In *Design of Circuits and Integrated Systems*, pages 1–6. Nov 2014. [pages ix, 22, and 23]

[Chand14] V. Chandra. Monitoring Reliability in Embedded Processors - A Multi-layer View. In *Proc. ACM/IEEE Design Automation Conference (DAC)*, pages 46:1–46:6. 2014. ISBN 978-1-4503-2730-5. [page 51]

[Chen03] G. Chen, K. Y. Chuah, M. F. Li, D. S. H. Chan, C. H. Ang, J. Z. Zheng, Y. Jin, and D. L. Kwong. Dynamic NBTI of PMOS transistors and its impact on device lifetime. In *IEEE 41st International Reliability Physics Symposium*, pages 196–202. March 2003. [page 15]

[Cook71] S. A. Cook. The complexity of theorem-proving procedures. In *Proc. of ACM Symposium on Theory of Computing (STOC)*, pages 151–158. ACM, New York, NY, USA, 1971. URL http://doi.acm.org/10.1145/800157.805047. [page 40]

[Czysz10] D. Czysz, G. Mrugalski, N. Mukherjee, J. Rajski, and J. Tyszer. On compaction utilizing inter and intra-correlation of unknown states. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 29(1):117–126, Jan 2010. [page 47]

[Dadgo10] H. Dadgour and K. Banerjee. Aging-resilient design of pipelined architectures using novel detection and correction circuits. In *Conference on Design, Automation and Test in Europe (DATE)*, pages 244–249. 2010. [pages 51 and 57]

[Das09] S. Das, C. Tokunaga, S. Pant, W.-H. Ma, S. Kalaiselvan, K. Lai, D. Bull, and D. Blaauw. RazorII: In Situ Error Detection and Correction for PVT and SER Tolerance. *IEEE Journal of Solid-State Circuits*, 44(1):32–48, Jan 2009. [pages 50 and 51]

[Davis60] M. Davis and H. Putnam. A computing procedure for quantification theory. *J. ACM*, 7(3):201–215, July 1960. URL http://doi.acm.org/10.1145/321033.321034. [page 40]

[Davis62] M. Davis, G. Logemann, and D. Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5(7):394–397, July 1962. URL http://doi.acm.org/10.1145/368273.368557. [page 40]

[Dervi91] B. I. Dervisoglu and G. E. Stong. Design for testability using scanpath techniques for path-delay test and measurement. In *Proc. International Test Conference (ITC)*, pages 365–. Oct 1991. [page 28]

[des] URL https://www.synopsys.com/implementation-and-signoff/rtl-synthesis-test/design-compiler-graphical.html, Synopsys Design Compiler Version D-2010.03. [page 86]

[Elect17] R. Electronics. *Semiconductor Reliability Handbook*. 2017. URL www.renesas.com. [pages ix, 17, and 20]

[Firou13] F. Firouzi, F. Ye, K. Chakrabarty, and M. Tahoori. Representative Critical-Path Selection for Aging-Induced Delay Monitoring. In *Proc. IEEE International Test Conference (ITC)*, pages 1–10. Sept 2013. [page 55]

[Fujiw06] E. Fujiwara, editor. John Wiley Sons, Inc., 2006. [page 5]

[Garey79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979. ISBN 0716710447. [page 35]

[Garg08] R. Garg, R. Putman, and N. A. Touba. Increasing output compaction in presence of unknowns using an x-canceling misr with deterministic observation. In *IEEE VLSI Test Symposium (VTS)*, pages 35–42. April 2008. [page 47]

[Ghosh09] A. Ghosh, R. Brown, R. Rao, and C.-T. Chuang. A precise negative bias temperature instability sensor using slew-rate monitor circuitry. In *Proc. IEEE Intl. Symposium on Circuits and Systems (ISCAS)*, pages 381–384. 2009. [pages 50 and 57]

[Goel89] A. K. Goel and Y. T. Au-Yeung. Electromigration in the VLSI interconnect metallizations. In *32nd Midwest Symposium on Circuits and Systems*, pages 821–824 vol.2. Aug 1989. [pages 25 and 26]

[Gomez15] A. Gomez, L. Poehls, F. Vargas, and V. Champac. An Early Prediction Methodology for Aging Sensor Insertion to Assure Safe Circuit Operation due to NBTI Aging. In *IEEE 33rd VLSI Test Symposium (VTS)*, pages 1–6. April 2015. [page 55]

[Hamme86] P. Hammer and B. Simeone. *Combinatorial Optimization*, volume 1403 of Lecture Notes in Mathematics. Springer-Verlag, 1986. ISBN 9783540517979. [page 41]

[Helle14] S. Hellebrand, T. Indlekofer, M. Kampmann, M. A. Kochte, C. Liu, and H.-J. Wunderlich. FAST-BIST: Faster-than-At-Speed BIST Targeting Hidden Delay Defects. In *Proc. IEEE International Test Conference (ITC)*, pages 1–8. 2014. [pages 14, 43, 46, 47, 48, and 93]

[Hsieh07] W. C. Hsieh and W. Hwang. Low power on-chip current monitoring medium-grained adaptive voltage control. In *IEEE International Symposium on Circuits and Systems*, pages 1637–1640. May 2007. [pages 50 and 57]

[Hu85] C. Hu, S. C. Tam, F.-C. Hsu, P.-K. Ko, T.-Y. Chan, and K. W. Terrill. Hot-Electron-Induced MOSFET Degradation Model, Monitor, and Improvement. *IEEE Transactions on Electron Devices*, 32(2):375–385, Feb 1985. [page 20]

[Huang06] Y. Huang. On n-detect pattern set optimization. In *7th International Symposium on Quality Electronic Design (ISQED'06)*, pages 6 pp.–450. March 2006. [page 44]

[Ingel11] U. Ingelsson, S. Y. Chang, and E. Larsson. Measurement Point Selection for In-Operation Wear-Out Monitoring. In *14th IEEE International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS)*, pages 381–386. April 2011. [pages 26, 55, and 61]

[Inoue08] H. Inoue, Y. Li, and S. Mitra. VAST: Virtualization-Assisted Concurrent Autonomous Self-Test. In *IEEE International Test Conference (ITC)*, pages 1–10. Oct 2008. [pages 48 and 49]

[itc] URL `http://www.cerc.utexas.edu/itc99-benchmarks/bench.html`. [pages 68 and 103]

[ITR12] International Technology Roadmap for Semiconductors (ITRS), 09 2012. URL http://www.itrs.net/. [page 1]

[Jacks99] J. Jackson, Ö. Oralkan, D. Dumin, and G. Brown. Electric breakdowns and breakdown mechanisms in ultra-thin silicon oxides. In *Microelectronics Reliability*, volume 39, pages 171–179. March 1999. [pages ix and 22]

[Jha93] N. K. Jha and S. J. Wang. Design and synthesis of self-checking vlsi circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 12(6):878–887, Jun 1993. [page 5]

[Kaise15] M. Kaiser. *Electronic control unit (ECU)*, pages 254–259. Springer Fachmedien Wiesbaden, Wiesbaden, 2015. ISBN 978-3-658-03964-6. URL http://dx.doi.org/10.1007/978-3-658-03964-6_16. [page 4]

[Kampm15] M. Kampmann, M. A. Kochte, E. Schneider, T. Indlekofer, S. Hellebrand, and H. J. Wunderlich. Optimized selection of frequencies for faster-than-at-speed test. In *IEEE 24th Asian Test Symposium (ATS)*, pages 109–114. Nov 2015. [page 48]

[Kampm16] M. Kampmann and S. Hellebrand. X marks the spot: Scan-flip-flop clustering for faster-than-at-speed test. In *IEEE 25th Asian Test Symposium (ATS)*, pages 1–6. Nov 2016. [page 47]

[Karna53] M. Karnaugh. The map method for synthesis of combinational logic circuits. *Transactions of the American Institute of Electrical Engineers, Part I: Communication and Electronics*, 72(5):593–599, Nov 1953. [page 36]

[Kim08] T.-H. Kim, R. Persaud, and C. Kim. Silicon Odometer: An On-Chip Reliability Monitor for Measuring Frequency Degradation of Digital Circuits. *IEEE Journal of Solid-State Circuits*, 43(4):874–880, April 2008. [pages 49, 50, and 57]

[Kim10] Y. M. Kim, Y. Kameda, H. Kim, M. Mizuno, and S. Mitra. Low-cost gate-oxide early-life failure detection in robust systems. In *Symposium on VLSI Circuits*, pages 125–126. June 2010. [page 44]

[Kocht10] M. A. Kochte, C. G. Zoellin, and H.-J. Wunderlich. Efficient concurrent self-test with partially specified patterns. *Journal of Electronic Testing: Theory and Applications (JETTA)*, 26(5):581–594, 2010. [pages 48 and 49]

[Krsti98] A. Krstic and K.-T. Cheng. *Delay Fault Test for VLSI Circuits*. Springer, 1998. ISBN -10: 0792382951 -13: 978-0792382959. [page 13]

[Lai13] L. Lai, V. Chandra, R. Aitken, and P. Gupta. SlackProbe: A low overhead in situ on-line timing slack monitoring methodology. In *Conference on Design, Automation and Test in Europe (DATE)*, pages 282–287. March 2013. [pages 55 and 78]

[Laung06] C.-W. W. Laung-Terng Wang and X. Wen. *VLSI Test Principles and Architectures Design for Testability*. ELSEVIER, 2006. ISBN 978-0-12-370597-6. [pages ix and 31]

[Lee59] C. Y. Lee. Representation of switching circuits by binary-decision programs. *The Bell System Technical Journal*, 38(4):985–999, July 1959. [page 36]

[Lee08] J. Lee and E. J. McCluskey. Failing frequency signature analysis. In *IEEE International Test Conference (ITC)*, pages 1–8. Oct 2008. [page 48]

[Li08] Y. Li, S. Makar, and S. Mitra. CASP: Concurrent Autonomous Chip Self-Test Using Stored Test Patterns. In *Conference on Design, Automation and Test in Europe (DATE)*, pages 885–890. March 2008. [page 48]

[Lien14] W. C. Lien, K. J. Lee, K. Chakrabarty, and T. Y. Hsieh. Output-bit selection with x-avoidance using multiple counters for test-response compaction. In *IEEE European Test Symposium (ETS)*, pages 1–6. May 2014. [page 47]

[Lin06] X. Lin, K. h. Tsai, C. Wang, M. Kassab, J. Rajski, T. Kobayashi, R. Klingenberg, Y. Sato, S. Hamada, and T. Aikyo. Timing-Aware ATPG for High Quality At-speed Testing of Small Delay Defects. In *15th Asian Test Symposium (ATS)*, pages 139–146. Nov 2006. [page 44]

[Liu15] C. Liu, M. A. Kochte, and H. J. Wunderlich. Efficient observation point selection for aging monitoring. In *Proc. IEEE 21st International On-Line Testing Symposium (IOLTS)*, pages 176–181. July 2015. [pages 7, 9, and 77]

[Liu17] C. Liu, M. A. Kochte, and H. J. Wunderlich. Aging monitor reuse for small delay fault testing. In *IEEE 35th VLSI Test Symposium (VTS)*, pages 1–6. April 2017. [pages 7, 10, 48, and 94]

[Mahap16] S. Mahapatra, editor. *Characterization Methods, Process and Materials Impact, DC and AC Modeling*. Springer, 2016. [page 15]

[Mao90] W. W. Mao and M. D. Ciletti. A Variable Observation Time Method for Testing Delay Faults. In *Proc. of 27th ACM/IEEE Design Automation Conference(DAC)*, pages 728–731. Jun 1990. [page 45]

[Masse04] J. G. Massey. NBTI: what we know and what we need to know - a tutorial addressing the current understanding and challenges for the future. In *IEEE International Integrated Reliability Workshop Final Report*, pages 199–211. Oct 2004. [page 15]

[Mitra02] S. Mitra and K. S. Kim. X-compact: an efficient response compaction technique for test cost reduction. In *Proc. International Test Conference (ITC)*, pages 311–320. 2002. [page 47]

[Mitra04] S. Mitra, S. S. Lumetta, and M. Mitzenmacher. X-tolerant signature analysis. In *2004 International Conference on Test*, pages 432–441. Oct 2004. [page 47]

[mod] URL `https://www.mentor.com/products/fv/modelsim/`, ModelSim SE-64 6.6d. [page 86]

[Monsi01] F. Monsieur, E. Vincent, D. Roy, S. Bruyre, G. Pananakakis, and G. Ghibaudo. Time to breakdown and voltage to breakdown modeling for ultra-thin oxides (tox < 32å). In *IEEE International Integrated Reliability Workshop. Final Report (Cat. No.01TH8580)*, pages 20–25. 2001. [page 24]

[Monta02] R. R. Montanes, J. P. de Gyvez, and P. Volf. Resistance Characterization for Weak Open Defects. *IEEE Design Test of Computers*, 19(5):18–26, Sep 2002. [page 13]

[Moore68] G. E. Moore. Trends in silicon device technology. In *International Electron Devices Meeting*, volume 14, pages 12–12. 1968. [page 1]

[Mukhe09] S. Mukherjee, C. Weaver, J. Emer, S. Reinhardt, and T. Austin. A Systematic Methodology to Compute the Architectural Vulnerability Factors for a High-Performance Microprocessor. In *Proc. IEEE/ACM Intl. Symposium on Microarchitecture (MICRO-36)*, pages 29–40. 2009. [page 61]

[nan] URL http://www.nangate.com, Nangate 45nm Open Cell Library v1.3. [pages 68 and 86]

[Narus03] M. Naruse, I. Porneranz, S. M. Reddy, and S. Kundu. On-chip compression of output responses with unknown values using lfsr reseeding. In *Proc. International Test Conference (ITC)*, volume 1, pages 1060–1068. Sept 2003. [page 47]

[Natar98] S. Natarajan, M. A. Breuer, and S. K. Gupta. Process Variations and their Impact on Circuit Operation. In *Proc. IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFTS)*, pages 73–81. Nov 1998. [page 13]

[Nigh00] P. Nigh and A. Gattiker. Test Method Evaluation Experiments and Data. In *Proc. IEEE International Test Conference (ITC)*, pages 454–463. 2000. [page 45]

[nxp] URL http://www.nxp.com/, NXP benchmarks. [pages 68, 86, and 103]

[Ogawa95] S. Ogawa and N. Shiono. Generalized diffusion-reaction model for the low-field charge-buildup instability at the Si-SiO_2 interface. *Physical Review B Condens Matter*, 51:4218–4230, Feb 1995. URL https://link.aps.org/doi/10.1103/PhysRevB.51.4218. [page 16]

[Omana13] M. Omaña, D. Rossi, N. Bosio, and C. Metra. Low Cost NBTI Degradation Detection and Masking Approaches. *IEEE Transactions on Computers*, 62(3):496–509, March 2013. [page 51]

[Park88] E. S. Park, M. R. Mercer, and T. W. Williams. Statistical delay fault coverage and defect level for delay faults. In *Proc. International Test Conference (ITC)*, pages 492–499. Sep 1988. [page 13]

[Parke78] K. Parker and E. McCluskey. Sequential circuit output probabilities from regular expressions. *IEEE Transactions on Computers*, 27(3):222–231, 1978. [page 63]

[Paul07] B. C. Paul, K. Kang, H. Kufluoglu, M. A. Alam, and K. Roy. Negative Bias Temperature Instability: Estimation and Design for Improved Reliability of Nanoscale Circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(4):743–751, April 2007. [page 17]

[Pei09] S. Pei, H. Li, and X. Li. A low overhead on-chip path delay measurement circuit. In *IEEE Asian Test Symposium (ATS)*, pages 145–150. Nov 2009. [pages 50 and 51]

[Pei10] S. Pei, H. Li, and X. Li. An On-Chip Clock Generation Scheme for Faster-than-at-Speed Delay Testing. In *Conference on Design, Automation and Test in Europe (DATE)*, pages 1353–1356. March 2010. [page 46]

[pri] URL `https://www.synopsys.com/implementation-and-signoff/signoff/primetime.html`, Synopsys' PrimeTime static timing analysis tool. [pages 68 and 86]

[Rajsk03] J. Rajski, J. Tyszer, C. Wang, and S. M. Reddy. Convolutional compaction of test responses. In *Proc. International Test Conference (ITC)*, volume 1, pages 745–754. Sept 2003. [page 47]

[Rajsk06] W. Rajski and J. Rajski. Modular compactor of test responses. In *24th IEEE VLSI Test Symposium (VTS)*, pages 10 pp.–. April 2006. [page 47]

[Saliv15] M. Saliva, F. Cacho, V. Huard, X. Federspiel, D. Angot, A. Benhassain, A. Bravaix, and L. Anghel. Digital circuits reliability with in-situ monitors in 28nm fully depleted SOI. In *Conference on Design, Automation and Test in Europe (DATE)*, pages 441–446. March 2015. [pages x, 49, 50, 51, and 52]

[Saluj08] K. K. Saluja, S. Vijayakumar, W. Sootkaneung, and X. Yang. NBTI Degradation: A Problem or a Scare? In *Proc. IEEE Intl. Conference on VLSI Design (VLSID)*, pages 137–142. 2008. [page 67]

[Sato07] T. Sato and Y. Kunitake. A Simple Flip-Flop Circuit for Typical-Case Designs for DFM. In *Proc. International Symposium on Quality Electronic Design (ISQED)*, pages 539–544. March 2007. [pages 50 and 51]

[Sato09] Y. Sato, S. Kajihara, Y. Miura, T. Yoneda, S. Ohtake, I. Inoue, and H. Fujiwara. A circuit failure prediction mechanism (DART) for high field reliability. In *Proc. IEEE International Conference on ASIC (ASICON)*, pages 581–584. Oct 2009. [pages 48 and 49]

[Savir92] J. Savir. Skewed-load transition test: Part i, calculus. In *Proc. International Test Conference (ITC)*, pages 705–. Sep 1992. [page 28]

[Savir94] J. Savir and S. Patil. On broad-side delay test. In *Proc. of IEEE VLSI Test Symposium (VTS)*, pages 284–290. Apr 1994. [page 28]

[Schne16] E. Schneider, M. A. Kochte, S. Holst, X. Wen, and H.-J. Wunderlich. GPU-Accelerated Simulation of Small Delay Faults. *to appear in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2016. [page 104]

[Segur04] J. Segura and F. C. Hawkins. *CMOS Electronics: How It Works, How It Fails*. Wiley-IEEE Press, 2004. ISBN 0-471-47669-2. [pages ix, 11, 18, 22, and 26]

[Semia14] J. Semiao, D. Saraiva, C. Leong, A. Romao, M. Santos, I. Teixeira, and J. Teixeira. Performance Sensor for Tolerance and Predictive Detection of Delay-Faults. In *IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, pages 110–115. Oct 2014. [page 51]

[Shann38] C. E. Shannon. A symbolic analysis of relay and switching circuits. *Electrical Engineering*, 57(12):713–723, Dec 1938. [page 34]

[Sharm05] M. Sharma and W.-T. Cheng. X-filter: filtering unknowns from compacted test responses. In *IEEE International Test Conference (ITC)*, pages 9 pp.–1098. Nov 2005. [page 47]

[Shoom02] M. L. Shooman. *Reliability of Computer Systems and Networks: Fault Tolerance, Analysis, and Design*. John Wiley Sons, Inc., 2002. ISBN 0-471-29342-3 (Hardback); 0-471-22460-X (Electronic). [pages ix, 2, and 3]

[Singh10] A. Singh, C. Han, and X. Qian. An output compression scheme for handling x-states from over-clocked delay tests. In *VLSI Test Symposium (VTS)*, pages 57–62. April 2010. [pages 47 and 93]

[Smith85] G. L. Smith. Model for delay faults based upon paths. *Proc. International Test Conference (ITC)*, Nov 1985. [page 13]

[Store77] T. M. Storey and J. W. Barry. Delay test simulation. In *Proc. of the 14th Design Automation Conference (DAC)*, DAC '77, pages 492–494. IEEE Press, Piscataway, NJ, USA, 1977. URL http://dl.acm.org/citation.cfm?id=800262.809174. [page 12]

[Stron09] A. W. Strong, E. Y. Wu, R.-P. Vollertsen, J. Sune, G. L. Rosa, T. D. Sullivan, and S. E. Rauch. *Reliability Wearout Mechanisms in Advanced CMOS Technologies*. Wiley-IEEE Press, 2009. ISBN 978-0-471-73172-6. [page 25]

[Sverd13] Y. Sverdlik. One minute of data center downtime costs US $7,900 on average. 2013. URL `http://www.datacenterdynamics.com`. [page 4]

[Taizh15] S. C. Taizhi Liu, Chang-Chih Chen and L. Milor. System-level variation-aware aging simulator using a unified novel gate-delay model for bias temperature instability, hot carrier injection, and gate oxide breakdown. In *Microelectronics Reliability*, pages 1334–1340. Elsevier, June 2015. [pages ix and 24]

[Tang04] Y. Tang, H. J. Wunderlich, H. Vranken, F. Hapke, M. Wittke, P. Engelke, I. Polian, and B. Becker. X-masking during logic bist and its impact on defect coverage. In *International Test Conference (ITC)*, pages 442–451. Oct 2004. [page 47]

[Tayad07] R. Tayade, S. Sundereswaran, and J. Abraham. Small-Delay Defect Detection in the Presence of Process Variations. In *Proc. International Symposium on Quality Electronic Design (ISQED)*, pages 711–716. March 2007. [page 13]

[Tayad08] R. Tayade and J. A. Abraham. On-chip Programmable Capture for Accurate Path Delay Test and Characterization. In *Proc. IEEE International Test Conference (ITC)*, pages 1–10. Oct 2008. [page 46]

[Tehra11] M. Tehranipoor, K. Peng, and K. Chakrabarty. *Test and Diagnosis for Small-Delay Defects*. Springer, 2011. ISBN 9781441982964. [pages ix, ix, 12, 13, 27, 29, and 30]

[tet] URL `https://www.synopsys.com/implementation-and-signoff/rtl-synthesis-test/test-automation/tetramax-atpg.html`, TetraMax ATPG Version D-2010.03. [page 103]

[Touba07] N. A. Touba. X-canceling misr x2014; an x-tolerant methodology for compacting output responses with unknowns using a misr. In *IEEE International Test Conference (ITC)*, pages 1–10. Oct 2007. [page 47]

[Tscha09] J. Tschanz, K. Bowman, S. Walstra, M. Agostinelli, T. Karnik, and V. De. Tunable Replica Circuits and Adaptive Voltage-Frequency Techniques for Dynamic Voltage, Temperature, and Aging Variation Tolerance. In *IEEE Symposium on VLSI Circuits*, pages 112–113. June 2009. [pages 50 and 54]

[Vazqu10] J. Vazquez, V. Champac, I. Teixeira, M. Santos, and J. Teixeira. Programmable aging sensor for automotive safety-critical applications. In *Conference on Design, Automation and Test in Europe (DATE)*, pages 618–621. march 2010. [pages x, 49, 51, 52, 54, and 57]

[Volle99] R. Vollertsen and R. Hijab. Burn-in. In *IEEE International Integrated Reliability Workshop Final Report*, pages 132–133. 1999. [page 45]

[vonNe56] J. von Neumann. Probabilistic logics and synthesis of reliable organisms from unreliable components. In C. Shannon and J. McCarthy, editors, *Automata Studies*, pages 43–98. Princeton University Press, 1956. [page 5]

[Waicu87] J. A. Waicukauski, E. Lindbloom, B. K. Rosen, and V. S. Iyengar. Transition fault simulation. *IEEE Design Test of Computers*, 4(2):32–38, April 1987. [page 12]

[Wang07a] W. Wang, V. Reddy, A. T. Krishnan, R. Vattikonda, S. Krishnan, and Y. Cao. Compact modeling and simulation of circuit reliability for 65-nm cmos technology. *IEEE Transactions on Device and Materials Reliability*, 7(4):509–517, Dec 2007. [pages 16, 20, and 66]

[Wang07b] W. Wang, Z. Wei, S. Yang, and Y. Cao. An Efficient Method to Identify Critical Gates under Circuit Aging. In *Proc. IEEE/ACM International Conferenceon Computer-Aided Design (ICCAD)*, pages 735–740. Nov. 2007. [page 55]

[Wang11] T. Wang, D. Chen, and R. Geiger. Multi-site on-chip current sensor for electromigration monitoring. In *IEEE 54th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 1–4. Aug 2011. [pages 50 and 57]

[Wang12] S. Wang, J. Chen, and M. Tehranipoor. Representative Critical Reliability Paths for Low-Cost and Accurate On-Chip Aging Evaluation. In *IEEE/ACM*

International Conference on Computer-Aided Design (ICCAD), pages 736–741. Nov 2012. [pages 50 and 54]

[Wohl07] P. Wohl, J. A. Waicukauski, and S. Ramnath. Fully x-tolerant combinational scan compression. In *IEEE International Test Conference (ITC)*, pages 1–10. Oct 2007. [page 47]

[Wu00] L. Wu, J. Fang, H. Yonezawa, Y. Kawakami, N. Iwanishi, H. Yan, P. Chen, A. I.-H. Chen, N. Koike, Y. Okamoto, C.-S. Yeh, and Z. Liu. Glacier: a hot carrier gate level circuit characterization and simulation system for vlsi design. In *International Symposium on Quality Electronic Design*, pages 73–79. 2000. [page 21]

[Wunde85] H.-J. Wunderlich. Protest: A tool for probabilistic testability analysis. In *Proc. ACM/IEEE Design Automation Conference(DAC)*, pages 204–211. 1985. [page 63]

[Wunde10] H.-J. Wunderlich. *Models in Hardware Testing*. Springer, 2010. ISBN 978-90-481-3281-2 (Print) 978-90-481-3282-9 (Online). [page 12]

[Yan03] H. Yan and A. D. Singh. Experiments in Detecting Delay Faults using Multiple Higher Frequency Clocks and Results from Neighboring Die. In *Proc. IEEE International Test Conference (ITC)*, pages 105–111. Sept 2003. [pages 45 and 46]

[Yan04] H. Yan and A. D. Singh. On the Effectiveness of Detecting Small Delay Defects in the Slack Interval. In *Proc. of IEEE International Workshop on Current and Defect Based Testing (DBT)*, pages 49–53. April 2004. [page 44]

[Yonez98] H. Yonezawa, J. Fang, Y. Kawakami, N. Iwanishi, L. Wu, A. I. H. Chen, N. Koike, P. Chen, C.-S. Yeh, and Z. Liu. Ratio based hot-carrier degradation modeling for aged timing simulation of millions of transistors digital circuits. In *International Electron Devices Meeting (IEDM)*, pages 93–96. Dec 1998. [page 21]

Basic Information of Benchmark Circuits

▲ **Table A.1** — Basic Information of Benchmark Circuits

	Benchmark *(1)*	#Inputs *(2)*	#Outputs *(3)*	#gates *(4)*	#FF *(5)*	Critical Path Length *(6)*
ISCAS89	s9234	19	20	1766	228	0.872
	s13207	30	119	2867	669	1.194
	s15850	14	83	3324	597	1.983
	s35932	35	320	11168	1728	0.423
	s38417	28	106	9796	1636	1.186
	s38584	4	249	12213	1451	1.158
ITC99	b14	32	54	5082	245	4.260
	b15	36	70	8164	449	2.140
	b17	37	97	26292	1415	4.730
	b18	37	44	72359	3320	6.070
	b19	24	71	143296	6642	5.440
	b20	32	22	10629	490	4.260
	b21	32	22	10706	490	4.300
NXP	p35k	740	56	23294	2173	3.192
	p45k	1409	219	25406	2331	2.306
	p77k	102	14	36428	3386	11.47
	p78k	172	507	70495	2977	1.701
	p81k	153	75	81540	3877	3.800
	p89k	332	256	58726	4301	2.886
	p100k	168	94	60767	5735	2.759
	p141k	790	1	107655	10501	2.828
	p239k	311	113	156514	18382	5.090
	p259k	316	97	198719	18398	5.230
	p267k	805	93	125040	16528	2.400
	p269k	806	93	125300	16528	2.400
	p279k	551	303	160688	17524	4.410
	p286k	639	122	210599	17713	4.800
	p295k	44	56	159648	18465	3.770
	p330k	1236	693	158697	16775	2.910

APPENDIX B

Results: Workload Monitor

Table B.1 and Table B.2 evaluate the monitoring accuracy of stress approximation. Table B.1 estimates the NBTI stress and calculates the approximation error for a single critical path. Table B.2 shows the results when a NBTI monitor is built for observing the maximum stress of multiple paths. Table B.4 and Table B.3 present the hardware overhead of the NBTI monitor for a single critical path and k-longest paths respectively.

B.1 Single Path Monitoring

In Table B.1, the column S_{avg} describes the *average* number of PMOS transistors under NBTI stress on the critical path. The following three columns give the *maximum*, *mean* and *root mean squared (RMS)* approximation error (cf. Section 5.2.5) for the 8-depth monitor. The presented error metrics are relative to the average stress metric S_{avg}. Then, these three approximation error columns are repeated for the remaining two monitors, with 10- and 12-depth in the approximation diagram. The upper half of Table B.1 shows the results with a quantization step size of 1; accordingly, the lower half is for step size of 2.

B.2 K-longest Paths Monitoring

Table B.2 lists the approximation stress and errors for the 10- and 100-longest paths monitoring with quantization step size of 1.

B.3 Hardware Overhead

Table B.4 and Table B.3 present the monitoring overhead for various quantization step size and different number of observed paths. In both tables, the second column shows the hardware area of the benchmark circuit and the next two columns state the absolute area of the monitor as well as the area overhead (+%) w.r.t. the benchmark area from Column 2. The absolute and relative area are repeatedly listed for monitors with the approximation diagram depth of 8, 10 and 12.

▲ **Table B.1** — Accuracy of NBTI monitors for a single critical path with various quantization step size

	Benchmark		Diagram Depth: 8			Diagram Depth: 10			Diagram Depth: 12		
		S_{avg}	Error [%]			Error [%]			Error [%]		
	Name (1)	[#] (2)	*max* (3)	*mean* (4)	*rms* (5)	*max* (6)	*mean* (7)	*rms* (8)	*max* (9)	*mean* (10)	*rms* (11)
quantization step size of 1	b14	48.9	10.2	1.7	2.4	10.2	1.6	2.3	10.2	1.5	2.2
	b15	30.9	16.2	1.7	2.5	16.2	1.5	2.3	16.2	1.2	2.1
	b17	27.6	18.1	2.6	3.6	14.5	2.1	3.1	14.5	1.7	2.7
	b18	62.9	9.5	1.1	1.9	9.5	1.0	1.9	9.5	1.0	1.9
	b19	51.2	11.7	1.2	2.2	11.7	1.2	2.2	11.7	1.2	2.1
	b20	52.3	13.4	2.2	2.9	13.4	2.2	2.8	11.5	2.2	2.8
	b21	47.5	16.8	2.6	3.4	14.7	2.2	3.0	12.6	2.0	2.8
	p35k	37.2	13.4	2.7	3.6	13.4	2.6	3.5	13.4	2.5	3.3
	p45k	25.3	15.8	2.1	3.1	11.9	1.8	2.8	11.9	1.4	2.5
	p77k	127.1	5.5	1.1	1.4	5.5	1.0	1.3	5.5	0.9	1.2
	p81k	59.0	11.9	1.9	2.6	10.2	1.8	2.4	10.2	1.7	2.3
	p89k	38.4	18.2	3.0	3.9	18.2	2.8	3.7	18.2	2.7	3.5
	p100k	25.0	16.0	3.0	4.1	16.0	2.7	3.8	16.0	2.5	3.5
	p141k	27.6	10.9	1.8	2.7	10.9	1.6	2.5	10.9	1.3	2.3
	p239k	63.4	17.4	2.2	3.4	15.8	2.1	3.3	15.8	2.1	3.2
	p259k	65.3	15.3	2.1	3.2	15.3	2.0	3.2	16.8	2.0	3.1
	p267k	27.3	18.3	3.1	4.2	18.3	3.0	4.1	14.6	2.9	4.0
	p269k	26.5	18.9	3.3	4.4	15.1	3.2	4.3	15.1	3.1	4.2
	p279k	31.5	12.7	1.3	2.2	12.7	1.1	2.1	12.7	1.0	1.9
	p286k	35.0	14.3	1.9	2.7	14.3	1.6	2.4	11.4	1.3	2.1
	p295k	46.4	21.6	3.1	3.9	19.4	2.9	3.8	19.4	2.8	3.6
	p330k	22.2	9.0	1.3	2.5	9.0	0.9	2.1	13.5	0.6	1.7
quantization step size of 2	b14	48.9	10.2	2.0	2.7	12.3	1.9	2.6	10.2	1.8	2.5
	b15	30.9	19.4	2.1	3.0	19.4	2.0	2.8	19.4	1.9	2.6
	b17	27.6	14.5	3.0	4.0	14.5	2.6	3.6	14.5	2.3	3.2
	b18	62.9	9.5	1.1	2.0	9.5	1.1	2.0	9.5	1.1	1.9
	b19	51.2	11.7	1.3	2.4	11.7	1.3	2.3	11.7	1.3	2.3
	b20	52.3	13.4	2.3	3.0	13.4	2.3	3.0	13.4	2.3	2.9
	b21	47.5	16.8	2.7	3.5	14.7	2.4	3.2	12.6	2.2	3.0
	p35k	37.2	16.1	2.9	3.8	16.1	2.8	3.7	13.4	2.7	3.6
	p45k	25.3	15.8	2.7	3.7	11.9	2.4	3.4	11.9	2.3	3.2
	p77k	127.1	6.3	1.1	1.5	6.3	1.1	1.4	5.5	1.0	1.3
	p81k	59.0	11.9	2.1	2.7	11.9	2.0	2.6	10.2	1.9	2.5
	p89k	38.4	20.8	3.1	4.1	20.8	3.0	3.9	18.2	2.9	3.8
	p100k	25.0	16.0	3.4	4.6	16.0	3.2	4.3	16.0	3.0	4.1
	p141k	27.6	10.9	2.3	3.2	10.9	2.2	3.1	10.9	2.1	2.9
	p239k	63.4	17.4	2.7	3.5	17.4	2.6	3.5	17.4	2.6	3.4
	p259k	65.3	16.8	2.4	3.4	15.3	2.3	3.3	16.8	2.3	3.2
	p267k	27.3	21.9	3.4	4.5	21.9	3.3	4.4	18.3	3.2	4.3
	p269k	26.5	18.9	3.6	4.8	18.9	3.5	4.6	18.9	3.4	4.5
	p279k	31.5	12.7	2.1	2.9	12.7	1.9	2.7	12.7	1.9	2.6
	p286k	35.0	14.3	2.2	3.0	14.3	2.0	2.8	11.4	1.8	2.6
	p295k	46.4	21.6	3.2	4.1	19.4	3.1	3.9	19.4	2.9	3.8
	p330k	22.2	13.5	2.5	3.6	13.5	2.4	3.4	13.5	2.4	3.3

▲ **Table B.2** — Accuracy of NBTI monitors for 10 and 100 longest paths with quantization step size of 1

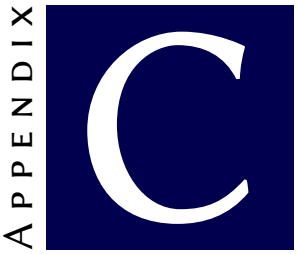
	Benchmark		Diagram Depth: 8			Diagram Depth: 10			Diagram Depth: 12		
		S_{avg}	Error [%]			Error [%]			Error [%]		
	Name	[#]	*max*	*mean*	*rms*	*max*	*mean*	*rms*	*max*	*mean*	*rms*
	(1)	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(7)*	*(8)*	*(9)*	*(10)*	*(11)*
10 Paths	b14	54.7	11.0	1.8	2.5	9.1	1.6	2.4	9.1	1.6	2.3
	b15	35.9	13.9	1.2	2.0	13.9	1.1	1.9	16.7	0.9	1.7
	b17	62.3	17.7	3.2	4.2	17.7	3.0	3.9	17.7	2.7	3.7
	b18	71.8	7.0	0.4	1.0	7.0	0.4	0.9	7.0	0.4	0.9
	b19	62.6	8.0	0.5	1.2	8.0	0.5	1.1	8.0	0.5	1.1
	b20	57.1	8.7	1.9	2.5	8.7	1.9	2.5	8.7	1.8	2.4
	b21	51.4	9.7	1.9	2.4	9.7	1.8	2.4	9.7	1.8	2.4
	p35k	40.3	12.4	2.4	3.1	12.4	2.3	3.0	9.9	2.2	3.0
	p45k	25.9	11.6	2.0	3.0	11.6	1.7	2.7	11.6	1.3	2.3
	p77k	131.5	6.8	1.0	1.4	6.8	1.0	1.3	6.8	1.0	1.3
	p81k	69.2	10.1	1.7	2.2	10.1	1.6	2.1	10.1	1.5	2.0
	p89k	39.8	10.0	2.0	2.8	10.0	2.0	2.7	10.0	1.9	2.6
	p100k	32.9	15.2	2.5	3.3	12.2	2.3	3.1	12.2	2.1	2.9
	p141k	29.5	10.2	1.3	2.3	10.2	1.1	2.1	10.2	1.0	1.9
	p239k	66.4	12.1	1.8	2.7	12.1	1.7	2.7	12.1	1.7	2.6
	p259k	68.6	10.2	1.4	2.1	10.2	1.3	2.1	10.2	1.3	2.0
	p267k	29.1	17.1	2.9	3.9	17.1	2.8	3.8	17.1	2.7	3.7
	p269k	29.0	17.3	3.2	4.3	17.3	3.1	4.2	17.3	3.0	4.1
	p279k	55.1	9.1	1.7	2.3	9.1	1.6	2.2	9.1	1.5	2.1
	p286k	35.2	11.4	1.7	2.5	11.4	1.5	2.3	11.4	1.3	2.1
	p295k	47.5	23.2	2.9	3.8	21.1	2.8	3.6	23.2	2.6	3.5
	p330k	22.7	13.2	1.4	2.6	13.2	1.0	2.1	8.8	0.7	1.7
100 Paths	b14	55.5	10.8	1.6	2.3	9.0	1.5	2.2	9.0	1.4	2.1
	b15	37.9	15.8	1.1	1.8	15.8	1.0	1.7	15.8	0.8	1.6
	b17	71.3	18.2	2.7	3.4	16.8	2.5	3.2	16.8	2.3	3.0
	b18	74.3	5.4	0.0	0.3	5.4	0.0	0.3	5.4	0.0	0.3
	b19	65.5	6.1	0.1	0.5	6.1	0.1	0.5	6.1	0.1	0.5
	b20	60.0	8.3	1.6	2.1	10.0	1.6	2.1	10.0	1.6	2.1
	b21	56.6	10.6	1.7	2.3	8.8	1.6	2.2	8.8	1.5	2.2
	p35k	41.4	12.1	2.2	2.9	12.1	2.1	2.8	12.1	2.1	2.7
	p45k	28.0	10.7	1.9	2.8	14.3	1.7	2.6	14.3	1.4	2.4
	p77k	135.9	6.6	1.1	1.4	5.9	1.0	1.4	5.9	1.0	1.3
	p81k	69.8	10.0	1.7	2.2	10.0	1.6	2.1	10.0	1.5	2.0
	p89k	40.8	9.8	1.8	2.5	9.8	1.8	2.4	9.8	1.7	2.4
	p100k	36.9	13.6	2.3	3.1	13.6	2.2	3.0	13.6	2.0	2.8
	p141k	34.0	14.7	2.5	3.4	14.7	2.4	3.3	14.7	2.3	3.1
	p239k	68.5	11.7	1.7	2.6	13.1	1.6	2.5	10.2	1.6	2.5
	p259k	69.0	10.1	1.3	2.0	10.1	1.3	2.0	10.1	1.3	2.0
	p267k	30.7	13.0	2.6	3.5	13.0	2.6	3.5	13.0	2.5	3.4
	p269k	30.4	13.1	2.6	3.5	13.1	2.6	3.5	13.1	2.5	3.4
	p279k	57.3	10.5	1.7	2.2	10.5	1.6	2.1	10.5	1.5	2.0
	p286k	36.1	8.3	1.5	2.3	8.3	1.4	2.1	8.3	1.3	2.0
	p295k	48.2	22.8	3.0	3.9	22.8	2.9	3.8	20.8	2.8	3.7
	p330k	35.0	20.0	3.6	4.7	20.0	3.5	4.6	17.2	3.5	4.5

▲ **Table B.3** — Area overhead of NBTI monitors for a single critical path with various quantization step size

	Benchmark		Diagram Depth: 8		Diagram Depth: 10		Diagram Depth: 12	
		Area	Area		Area		Area	
	Name (1)	$[\mu m^2]$ (2)	$[\mu m^2]$ (3)	[+%] (4)	$[\mu m^2]$ (5)	[+%] (6)	$[\mu m^2]$ (7)	[+%] (8)
quantization step size of 1	b14	4140	194	+4.7%	643	+15.5%	2129	+51.4%
	b15	6493	92	+1.4%	487	+7.5%	2047	+31.5%
	b17	21058	293	+1.4%	1020	+4.8%	3043	+14.5%
	b18	58830	67	+0.1%	295	+0.5%	1234	+2.1%
	b19	117150	94	+0.1%	348	+0.3%	1335	+1.1%
	b20	8645	138	+1.6%	484	+5.6%	1748	+20.2%
	b21	8762	294	+3.4%	1008	+11.5%	2578	+29.4%
	p35k	17966	206	+1.1%	748	+4.2%	2837	+15.8%
	p45k	18798	217	+1.2%	733	+3.9%	2610	+13.9%
	p77k	29521	328	+1.1%	1220	+4.1%	3560	+12.1%
	p81k	64231	238	+0.4%	851	+1.3%	2741	+4.3%
	p89k	44523	244	+0.5%	925	+2.1%	2725	+6.1%
	p100k	45056	225	+0.5%	921	+2.0%	2976	+6.6%
	p141k	81148	174	+0.2%	640	+0.8%	2014	+2.5%
	p239k	126082	203	+0.2%	669	+0.5%	2346	+1.9%
	p259k	162242	163	+0.1%	581	+0.4%	1838	+1.1%
	p267k	101822	152	+0.1%	448	+0.4%	1314	+1.3%
	p269k	102001	153	+0.2%	559	+0.5%	1789	+1.8%
	p279k	127297	105	+0.1%	394	+0.3%	1281	+1.0%
	p286k	169555	183	+0.1%	710	+0.4%	2482	+1.5%
	p295k	127161	201	+0.2%	727	+0.6%	2410	+1.9%
	p330k	133719	105	+0.1%	352	+0.3%	1074	+0.8%
quantization step size of 2	b14	4140	132	+3.2%	413	+10.0%	1466	+35.4%
	b15	6493	42	+0.7%	271	+4.2%	1113	+17.1%
	b17	21058	146	+0.7%	614	+2.9%	1973	+9.4%
	b18	58830	23	+0.0%	143	+0.2%	778	+1.3%
	b19	117150	47	+0.0%	180	+0.2%	692	+0.6%
	b20	8645	65	+0.7%	219	+2.5%	992	+11.5%
	b21	8762	200	+2.3%	609	+6.9%	1712	+19.5%
	p35k	17966	118	+0.7%	393	+2.2%	1694	+9.4%
	p45k	18798	115	+0.6%	462	+2.5%	1594	+8.5%
	p77k	29521	202	+0.7%	593	+2.0%	2470	+8.4%
	p81k	64231	145	+0.2%	422	+0.7%	1524	+2.4%
	p89k	44523	127	+0.3%	450	+1.0%	1547	+3.5%
	p100k	45056	132	+0.3%	551	+1.2%	1978	+4.4%
	p141k	81148	107	+0.1%	413	+0.5%	1240	+1.5%
	p239k	126082	123	+0.1%	410	+0.3%	1227	+1.0%
	p259k	162242	117	+0.1%	280	+0.2%	1156	+0.7%
	p267k	101822	75	+0.1%	269	+0.3%	853	+0.8%
	p269k	102001	77	+0.1%	306	+0.3%	1001	+1.0%
	p279k	127297	92	+0.1%	386	+0.3%	1320	+1.0%
	p286k	169555	74	+0.0%	392	+0.2%	1124	+0.7%
	p295k	127161	125	+0.1%	357	+0.3%	1841	+1.4%
	p330k	133719	126	+0.1%	316	+0.2%	625	+0.5%

▲ **Table B.4** — Area overhead of NBTI monitors for 10 and 100 longest paths with quantization step size of 1

	Benchmark		Diagram Depth: 8		Diagram Depth: 10		Diagram Depth: 12	
		Area	Area		Area		Area	
	Name (1)	[μm^2] (2)	[μm^2] (3)	[+%] (4)	[μm^2] (5)	[+%] (6)	[μm^2] (7)	[+%] (8)
10 Paths	b14	4140	202	+4.9%	741	+17.9%	2284	+55.2%
	b15	6493	117	+1.8%	398	+6.1%	1586	+24.4%
	b17	21058	278	+1.3%	1018	+4.8%	3987	+18.9%
	b18	58830	64	+0.1%	211	+0.4%	717	+1.2%
	b19	117150	64	+0.1%	163	+0.1%	624	+0.5%
	b20	8645	123	+1.4%	381	+4.4%	1348	+15.6%
	b21	8762	128	+1.5%	523	+6.0%	1752	+20.0%
	p35k	17966	164	+0.9%	632	+3.5%	2292	+12.8%
	p45k	18798	231	+1.2%	821	+4.4%	2464	+13.1%
	p77k	29521	251	+0.8%	989	+3.3%	3021	+10.2%
	p81k	64231	260	+0.4%	823	+1.3%	2817	+4.4%
	p89k	44523	129	+0.3%	600	+1.3%	2144	+4.8%
	p100k	45056	244	+0.5%	852	+1.9%	2936	+6.5%
	p141k	81148	113	+0.1%	353	+0.4%	1252	+1.5%
	p239k	126082	188	+0.1%	627	+0.5%	2032	+1.6%
	p259k	162242	126	+0.1%	405	+0.2%	1654	+1.0%
	p267k	101822	106	+0.1%	498	+0.5%	1830	+1.8%
	p269k	102001	147	+0.1%	530	+0.5%	2257	+2.2%
	p279k	127297	240	+0.2%	742	+0.6%	2767	+2.2%
	p286k	169555	166	+0.1%	650	+0.4%	2396	+1.4%
	p295k	127161	252	+0.2%	880	+0.7%	2917	+2.3%
	p330k	133719	117	+0.1%	308	+0.2%	950	+0.7%
100 Paths	b14	4140	204	+4.9%	720	+17.4%	2107	+50.9%
	b15	6493	156	+2.4%	545	+8.4%	2243	+34.5%
	b17	21058	266	+1.3%	1192	+5.7%	4224	+20.1%
	b18	58830	7	+0.0%	21	+0.0%	77	+0.1%
	b19	117150	16	+0.0%	63	+0.1%	209	+0.2%
	b20	8645	107	+1.2%	391	+4.5%	1386	+16.0%
	b21	8762	223	+2.5%	720	+8.2%	2191	+25.0%
	p35k	17966	171	+0.9%	590	+3.3%	2297	+12.8%
	p45k	18798	180	+1.0%	635	+3.4%	2309	+12.3%
	p77k	29521	249	+0.8%	997	+3.4%	3123	+10.6%
	p81k	64231	254	+0.4%	992	+1.5%	2923	+4.6%
	p89k	44523	94	+0.2%	390	+0.9%	1611	+3.6%
	p100k	45056	246	+0.5%	912	+2.0%	2850	+6.3%
	p141k	81148	158	+0.2%	525	+0.6%	1958	+2.4%
	p239k	126082	192	+0.2%	688	+0.5%	2080	+1.6%
	p259k	162242	116	+0.1%	486	+0.3%	1791	+1.1%
	p267k	101822	111	+0.1%	528	+0.5%	1898	+1.9%
	p269k	102001	133	+0.1%	501	+0.5%	1713	+1.7%
	p279k	127297	237	+0.2%	825	+0.6%	2966	+2.3%
	p286k	169555	140	+0.1%	520	+0.3%	1849	+1.1%
	p295k	127161	233	+0.2%	903	+0.7%	2920	+2.3%
	p330k	133719	180	+0.1%	581	+0.4%	2286	+1.7%



Results: Monitor Placement

The results of OP selection for all benchmarks are listed in Table C.1.

C.1 Observation Point Selection Results

The number of gates in the target design is shown in Column 2. The number of target paths is presented in Column 3. The number of OP candidates after eliminating the ones with no target path coverage is presented in Column 4. The target path coverage of the OP candidates is shown in Column 5. The number of (primary and pseudo-primary) target outputs is in Column 6. The number of OPs and the number of sensors inserted at path ends are listed in columns 7 and 8 respectively. The monitor number reduction

$$reduction := \frac{\#tar\_outputs - \#OPs - \#endpoints}{\#tar\_outputs}$$

is provided in the last column of the table.

C.2 Results of OP Effectiveness Validation

Since the inverter chain for OP delay matching can be implemented based on nominal or degraded timing (Section 6.2.4), the results of the two cases are listed in Tables C.2

▲ **Table C.1** — OP selection results

Benchmark *(1)*	#gates *(2)*	#paths *(3)*	#candidates *(4)*	coverage *(5)*	#tar_outputs *(6)*	#OPs *(7)*	#endpoints *(8)*	reduction *(9)*
s9234	1764	2284	33	0.99	55	13	9	0.60
s13207	2865	346	49	0.94	18	16	1	0.06
s15850	3320	4937	49	1.00	78	7	0	0.91
s35932	11168	6833	472	1.00	320	184	0	0.43
s38417	9796	12402	104	0.99	197	28	2	0.85
s38584	12183	622	21	0.95	47	11	6	0.64
p35k	23294	6207	2	1.00	71	2	0	0.97
p45k	25406	3727	37	1.00	54	4	0	0.93
p78k	70495	132545	2227	1.00	1417	306	0	0.78
p81k	82265	6268	112	1.00	69	3	0	0.96
p89k	58726	34184	62	1.00	352	16	0	0.95
p100k	60767	11151	271	1.00	116	36	0	0.69
p141k	107655	32485	186	1.00	333	8	0	0.98

and C.3 respectively. Dashes in the table indicate that aging alerts were not issued or timing failures not caused by the applied input stimuli during 10 years of operation.

C.2.1 Delay Matching based on Nominal Timing Profile

In Table C.2, Column 2 to 4 provide the operation time (in years) until the occurrence of an event. The event can refer to the first observed aging alert activation (*OP_activation*), the first observed timing failure (*failure*, both based on timing simulation), or the first clock violation by the critical path (*cpl_violation*, computed by STA). The results in the second and third column present the predictability of OPs (cf. Section 6.3). The last two columns display the remaining time margin of the nominal critical path when the first monitor alert activates. The absolute time margins in femtoseconds and the relative values as the percentage of the clock period are listed.

C.2.2 Delay Matching based on Degraded Timing Profile

The year of first timing failure occurrence and the clock violation time of the nominal critical path (values in Column 3 and 4) are identical in both tables. The results in the last two columns only differ from the previous table if the time of the first timing alert changes.

▲ **Table C.2** — OP validation (delay matching based on nominal profile)

Benchmark *(1)*	OP_activation *(2)*	failure *(3)*	cpl_violiation *(4)*	re_margin (fs) *(5)*	%clkp *(6)*
s9234	5.25	8.50	8.50	6681	0.70%
s13207	–	–	–	–	–
s15850	8.00	7.00	9.50	4613	0.22%
s35932	5.50	8.25	8.25	2631	0.59%
s38417	3.75	5.50	–	19469	1.56%
s38584	3.00	5.50	7.00	14007	1.16%
p35k	6.75	2.50	8.50	11085	0.32%
p45k	7.75	9.75	9.75	8417	0.33%
p78k	6.00	8.00	8.00	7174	0.39%
p81k	7.00	8.75	8.75	17646	0.29%
p89k	6.00	8.75	7.25	7430	0.23%
p100k	7.50	–	9.75	10648	0.35%
p141k	8.25	–	7.50	-5292	-0.18%

▲ **Table C.3** — OP validation (delay matching based on degraded profile)

Benchmark *(1)*	OP_activation *(2)*	failure *(3)*	cpl_violiation *(4)*	re_margin (fs) *(5)*	%clkp *(6)*
s9234	5.25	8.50	8.50	6681	0.70%
s13207	–	–	–	–	–
s15850	**5.75**	7.00	9.50	14600	0.69%
s35932	5.50	8.25	8.25	2631	0.59%
s38417	3.75	5.50	–	19469	1.56%
s38584	4.00	5.50	7.00	9302	0.77%
p35k	**1.75**	2.50	8.50	72437	2.09%
p45k	7.75	9.75	9.75	8417	0.33%
p78k	6.00	8.00	8.00	7174	0.39%
p81k	8.25	8.75	8.75	2757	0.05%
p89k	6.50	8.75	7.25	3662	0.12%
p100k	8.50	–	9.75	5102	0.17%
p141k	**5.50**	–	7.50	**12932**	**0.43%**

Results: Small Delay Fault Testing

D.1 Basic Information of the Benchmarks

The basic information of the circuits is listed in the first seven columns of Table D.1. The critical path length (*cpl*), number of patterns (*#pat*) in the original test set and the number of monitors (*#moni*) integrated in the circuit are listed in Column 2 to 4 respectively. *#gates* is the number of gates in the combinational part of the circuits and *#flip-flops* is the number of flip-flops. The last three columns relate to the hardware overhead: Column 7 provides the area of Circuit Under Test (CUT) A_{CUT} in μm^2; The absolute overhead of the monitor control logic A_{ctrl} and the relative overhead (ratio of the overhead to the original design A_{ctrl}/A_{CUT}) are listed in the last two columns respectively (cf. Section 7.2.3).

D.2 Covered and Uncovered Fault Sites w.r.t. Various Fault Magnitudes

In this section, we tabulate the fault site coverage and detection efficiency of the monitor reuse method. Afterward, for those uncovered target fault sites, we analyze the reasons and categorize the results (cf. Section 7.2.2).

▲ **Table D.1** — Basic information of circuits

Benchmark *(1)*	cpl [ns] *(2)*	#pat *(3)*	#moni *(4)*	#gates *(5)*	#flip-flops *(6)*	A_{CUT} [μm^2] *(7)*	A_{ctrl} [μm^2] *(8)*	A_{ctrl}/A_{CUT} *(9)*
s9234	0.872	304	63	1766	228	4270	280	0.066
s13207	1.194	376	198	2867	669	10815	300	0.028
s15850	1.983	261	171	3324	597	10252	247	0.024
s35932	0.423	75	513	11168	1728	30781	722	0.023
s38417	1.186	251	436	9796	1636	28249	698	0.025
s38584	1.158	313	426	12213	1450	28250	593	0.021
b14	4.260	2446	75	5082	245	7319	271	0.037
b15	2.140	3093	130	8164	449	12228	468	0.038
b17	4.730	3855	379	26292	1415	38767	728	0.018
b18	6.070	5197	842	72359	3320	99818	1079	0.011
b19	5.440	7657	1679	143296	6642	199085	1877	0.009
b20	4.260	2391	129	10629	490	14751	376	0.026
b21	4.300	2331	129	10706	490	14868	350	0.024
p35k	3.192	2102	558	23294	2173	45456	569	0.013
p45k	2.306	5349	638	25406	2331	49567	580	0.012
p78k	1.701	136	872	70495	2977	94384	1424	0.015
p89k	2.886	1920	1140	58726	4301	99825	1051	0.011
p100k	2.759	5178	1458	60767	5735	119001	1441	0.012
p141k	2.828	1621	2626	107655	10501	213753	2290	0.011

D.2.1 Fault Site Coverage and Detection Efficiency

This subsection first shows the results of fault site coverage (Table D.2) and detection efficiency (Table D.3) for each circuit w.r.t. all considered fault magnitudes. Then it gives an overview of coverage details (e.g. the number of covered fault sites, the number of detected or detectable fault sites, etc.) for all benchmarks w.r.t. the fault magnitude of 30σ.

Table D.4 tabulates the results of fault site coverage and detection efficiency for all circuits. Column 2 shows the number of target fault sites in each circuit. For a fault size of 30σ, the number of covered fault sites ($\#cov^{30\sigma}$) is listed in Column 3. 4 and 5 present the number of detectable and detected fault sites respectively. *#detectable* shows the upper limit of the number of fault sites that can possibly be detected by the monitor reuse approach, when cycle-accurate monitor configurations are provided. The column *#detected* gives the number of fault sites covered by the 16 computed test configurations. As stated in Section 7.2.2, the efficiency of monitor reuse is the ratio *efficiency := #detected / #detectable* and listed in Column 6.

D.2.2 Discussions of the Uncovered Fault Sites

For those amount of uncovered fault sites (*#uncovered* in Table D.5), we analyzed in more detail the reasons that limits the fault detection. The numbers of uncovered fault sites in each category are shown in Column 3 to 6 of Table D.5.

▲ **Table D.2** — Target fault site coverage w.r.t. various fault magnitudes

coverage$^{\delta}$ \ ISCAS89	s9234	s13207	s15850	s35932	s38417	s38584
$cov^{6\sigma}$	0.540	0.339	0.256	0.446	0.462	0.384
$cov^{8\sigma}$	0.592	0.380	0.288	0.538	0.511	0.445
$cov^{10\sigma}$	0.636	0.414	0.315	0.603	0.559	0.492
$cov^{12\sigma}$	0.669	0.448	0.340	0.660	0.598	0.531
$cov^{14\sigma}$	0.700	0.488	0.365	0.698	0.629	0.563
$cov^{16\sigma}$	0.725	0.528	0.386	0.740	0.659	0.601
$cov^{18\sigma}$	0.748	0.563	0.407	0.772	0.687	0.630
$cov^{20\sigma}$	0.773	0.598	0.422	0.817	0.710	0.658
$cov^{22\sigma}$	0.792	0.629	0.437	0.842	0.733	0.684
$cov^{24\sigma}$	0.810	0.663	0.453	0.872	0.752	0.707
$cov^{26\sigma}$	0.826	0.692	0.470	0.893	0.769	0.733
$cov^{28\sigma}$	0.841	0.717	0.486	0.904	0.785	0.753
$cov^{30\sigma}$	0.852	0.746	0.503	0.930	0.799	0.777

coverage$^{\delta}$ \ ITC	b14	b15	b17	b18	b19	b20	b21
$cov^{6\sigma}$	0.210	0.395	0.179	0.198	0.230	0.256	0.255
$cov^{8\sigma}$	0.253	0.421	0.193	0.218	0.252	0.299	0.296
$cov^{10\sigma}$	0.284	0.442	0.204	0.234	0.269	0.335	0.332
$cov^{12\sigma}$	0.313	0.459	0.213	0.247	0.284	0.368	0.363
$cov^{14\sigma}$	0.338	0.475	0.221	0.258	0.296	0.399	0.390
$cov^{16\sigma}$	0.359	0.492	0.228	0.268	0.307	0.426	0.413
$cov^{18\sigma}$	0.379	0.509	0.235	0.277	0.317	0.451	0.435
$cov^{20\sigma}$	0.398	0.525	0.241	0.286	0.326	0.471	0.456
$cov^{22\sigma}$	0.417	0.540	0.248	0.293	0.335	0.491	0.474
$cov^{24\sigma}$	0.435	0.555	0.254	0.300	0.343	0.510	0.492
$cov^{26\sigma}$	0.451	0.569	0.261	0.307	0.352	0.528	0.507
$cov^{28\sigma}$	0.466	0.580	0.269	0.314	0.359	0.543	0.523
$cov^{30\sigma}$	0.483	0.590	0.276	0.320	0.367	0.556	0.538

coverage$^{\delta}$ \ NXP	p35k	p45k	p78k	p89k	p100k	p141k
$cov^{6\sigma}$	0.178	0.375	0.426	0.157	0.322	0.237
$cov^{8\sigma}$	0.201	0.404	0.489	0.178	0.344	0.267
$cov^{10\sigma}$	0.222	0.429	0.535	0.195	0.363	0.292
$cov^{12\sigma}$	0.242	0.455	0.574	0.212	0.382	0.312
$cov^{14\sigma}$	0.260	0.477	0.607	0.227	0.401	0.330
$cov^{16\sigma}$	0.277	0.498	0.636	0.243	0.401	0.347
$cov^{18\sigma}$	0.295	0.523	0.662	0.257	0.436	0.363
$cov^{20\sigma}$	0.312	0.545	0.684	0.272	0.454	0.378
$cov^{22\sigma}$	0.331	0.567	0.704	0.286	0.472	0.391
$cov^{24\sigma}$	0.352	0.586	0.722	0.301	0.490	0.404
$cov^{26\sigma}$	0.373	0.605	0.737	0.316	0.508	0.417
$cov^{28\sigma}$	0.397	0.623	0.750	0.331	0.526	0.429
$cov^{30\sigma}$	0.420	0.642	0.764	0.347	0.544	0.442

▲ **Table D.3** — Target fault detection efficiency w.r.t. various fault magnitudes

efficiency$^{\delta}$ \ ISCAS89	s9234	s13207	s15850	s35932	s38417	s38584
$eff^{6\sigma}$	0.947	0.946	0.952	0.599	0.865	0.827
$eff^{8\sigma}$	0.945	0.942	0.955	0.667	0.881	0.858
$eff^{10\sigma}$	0.945	0.936	0.963	0.711	0.893	0.875
$eff^{12\sigma}$	0.949	0.934	0.966	0.742	0.902	0.888
$eff^{14\sigma}$	0.951	0.936	0.967	0.768	0.910	0.899
$eff^{16\sigma}$	0.953	0.934	0.967	0.799	0.918	0.908
$eff^{18\sigma}$	0.952	0.939	0.967	0.821	0.925	0.906
$eff^{20\sigma}$	0.957	0.941	0.964	0.840	0.928	0.908
$eff^{22\sigma}$	0.960	0.944	0.964	0.853	0.932	0.910
$eff^{24\sigma}$	0.962	0.949	0.966	0.864	0.939	0.920
$eff^{26\sigma}$	0.965	0.955	0.965	0.868	0.941	0.933
$eff^{28\sigma}$	0.969	0.960	0.965	0.873	0.941	0.939
$eff^{30\sigma}$	0.971	0.962	0.963	0.879	0.942	0.940

efficiency$^{\delta}$ \ ITC	b14	b15	b17	b18	b19	b20	b21
$eff^{6\sigma}$	0.964	0.910	0.917	0.981	0.933	0.973	0.977
$eff^{8\sigma}$	0.958	0.914	0.919	0.978	0.936	0.962	0.968
$eff^{10\sigma}$	0.957	0.917	0.920	0.974	0.937	0.956	0.961
$eff^{12\sigma}$	0.958	0.919	0.920	0.972	0.938	0.950	0.955
$eff^{14\sigma}$	0.957	0.918	0.921	0.970	0.938	0.947	0.952
$eff^{16\sigma}$	0.958	0.921	0.920	0.967	0.938	0.947	0.950
$eff^{18\sigma}$	0.960	0.923	0.922	0.963	0.938	0.948	0.948
$eff^{20\sigma}$	0.961	0.923	0.924	0.962	0.938	0.949	0.949
$eff^{22\sigma}$	0.960	0.923	0.926	0.960	0.939	0.949	0.952
$eff^{24\sigma}$	0.961	0.923	0.927	0.959	0.940	0.951	0.951
$eff^{26\sigma}$	0.962	0.925	0.928	0.959	0.942	0.953	0.952
$eff^{28\sigma}$	0.965	0.926	0.930	0.958	0.943	0.955	0.955
$eff^{30\sigma}$	0.967	0.927	0.931	0.957	0.945	0.957	0.958

efficiency$^{\delta}$ \ NXP	p35k	p45k	p78k	p89k	p100k	p141k
$eff^{6\sigma}$	0.946	0.903	0.678	0.842	0.848	0.651
$eff^{8\sigma}$	0.945	0.909	0.731	0.843	0.862	0.688
$eff^{10\sigma}$	0.947	0.911	0.767	0.841	0.868	0.717
$eff^{12\sigma}$	0.948	0.917	0.794	0.841	0.871	0.737
$eff^{14\sigma}$	0.949	0.914	0.816	0.840	0.874	0.754
$eff^{16\sigma}$	0.948	0.917	0.834	0.841	0.874	0.767
$eff^{18\sigma}$	0.951	0.923	0.849	0.842	0.884	0.779
$eff^{20\sigma}$	0.952	0.926	0.862	0.843	0.889	0.790
$eff^{22\sigma}$	0.952	0.928	0.872	0.844	0.894	0.799
$eff^{24\sigma}$	0.954	0.930	0.881	0.845	0.900	0.807
$eff^{26\sigma}$	0.955	0.933	0.889	0.846	0.903	0.814
$eff^{28\sigma}$	0.956	0.935	0.897	0.846	0.907	0.821
$eff^{30\sigma}$	0.956	0.937	0.902	0.847	0.911	0.828

▲ **Table D.4** — The target fault site coverage and detection efficiency w.r.t. the fault magnitude of 30σ

Benchmark *(1)*	#fault_site *(2)*	$\#cov^{30\sigma}$ *(3)*	#detectable *(4)*	#detected *(5)*	efficiency *(6)*
s9234	7178	6118	5869	5700	0.971
s13207	9348	6974	7118	6848	0.962
s15850	10936	5499	5482	5278	0.963
s35932	43390	40374	34918	30684	0.879
s38417	36600	29260	28997	27323	0.942
s38584	29630	23035	24229	22766	0.940
b14	19963	9636	9451	9137	0.967
b15	30656	18094	17870	16569	0.927
b17	82528	22812	21571	20090	0.931
b18	231041	74007	76145	72850	0.957
b19	441423	161900	164390	155329	0.945
b20	35457	19726	17457	16701	0.957
b21	36762	19781	18337	17565	0.958
p35k	102131	42926	42435	40580	0.956
p45k	85123	54623	57416	53804	0.937
p78k	327768	250458	256361	231262	0.902
p89k	195872	68006	75874	64234	0.847
p100k	224464	122006	132245	120534	0.911
p141k	335571	148287	172655	142904	0.828

▲ **Table D.5** — Uncovered fault sites categorization w.r.t. the fault magnitude of 30σ

Benchmark *(1)*	#uncovered *(2)*	#outside_range *(3)*	#unsensitized *(4)*	#freq_constraint *(5)*	#masking *(6)*
s9234	1060	272	323	316	149
s13207	2374	1448	537	119	270
s15850	5437	4411	524	301	201
s35932	3016	384	506	353	1773
s38417	7340	3922	1148	647	1623
s38584	6595	3896	695	559	1445
b14	10327	2691	4318	3007	311
b15	12562	5623	4008	1668	1263
b17	59716	39101	13934	5424	1257
b18	157034	105022	34513	14208	3291
b19	279523	185314	61183	23996	9030
b20	15731	6132	4425	4439	735
b21	16981	6470	4968	4790	753
p35k	59205	42754	9740	4873	1838
p45k	30500	20481	3172	3240	3607
p78k	77310	26742	855	26343	23370
p89k	127866	78867	22880	14499	11620
p100k	102458	69324	12841	8677	11616
p141k	187284	130000	15240	12570	29474

Publications of the Author

- **Aging Monitor Reuse for Small Delay Fault Testing**, Liu, C., Kochte, M.A. and Wunderlich, H.-J. Proceedings of the 35th VLSI Test Symposium (VTS'17) Caesars Palace, Las Vegas, Nevada, USA, 9-12 April 2017, pp. 1-6

- **Efficient Observation Point Selection for Aging Monitoring**, Liu, C., Kochte, M.A. and Wunderlich, H.-J. Proceedings of the 21st IEEE International On-Line Testing Symposium (IOLTS'15) Elia, Halkidiki, Greece, 6-8 July 2015, pp. 176-181

- **On-Line Prediction of NBTI-induced Aging Rates**, Baranowski, R., Firouzi, F., Kiamehr, S., Liu, C., Tahoori, M. and Wunderlich, H.-J. Proceedings of the ACM/IEEE Conference on Design, Automation Test in Europe (DATE'15) Grenoble, France, 9-13 March 2015, pp. 589-592

- **FAST-BIST: Faster-than-At-Speed BIST Targeting Hidden Delay Defects**, Hellebrand, S., Indlekofer, T., Kampmann, M., Kochte, M.A., Liu, C. and Wunderlich, H.-J. Proceedings of the IEEE International Test Conference (ITC'14) Seattle, Washington, USA, 20-23 October 2014, pp. 1-8

- **Synthesis of Workload Monitors for On-line Stress Prediction**, Baranowski, R., Cook, A., Imhof, M.E., Liu, C. and Wunderlich, H.-J. Proceedings of the 16th IEEE Symp. Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT'13) New York City, New York, USA, 2-4 October 2013, pp. 137-142

Declaration

All the work contained within this thesis, except where otherwise acknowledged, was solely the effort of the author. At no stage was any collaboration entered into with any other party.

Chang Liu