# Efficient In-Situ Delay Monitoring for Chip Health Tracking

Hadi Ahmadi Balef

Committee:

| | |
|---|---|
| prof.dr. K.G.W. Goossens | Eindhoven University of Technology, $1^e$ *promotor* |
| prof.dr. J. Pineda de Gyvez | Eindhoven University of Technology, $2^e$ *promotor* |
| prof.dr.ir. P.G.M. Baltus | Eindhoven University of Technology, *chairman* |
| dr. S.D. Cotofana | Technische Universiteit Delft |
| prof.dr. M. Baradaran Tahoori | Karlsruhe Institute of Technology |
| prof. dr. H. Corporaal | Eindhoven University of Technology |



This thesis is typeset in LaTeX, was written in *Sublime Text* and built with *SCons*.

Printed by ProefschriftMaken – The Netherlands.

A catalogue record is available from the Eindhoven University of Technology Library.
ISBN: 978-90-386-4930-6

# Efficient In-Situ Delay Monitoring for Chip Health Tracking

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Technische Universiteit Eindhoven, op gezag van de
rector magnificus prof.dr.ir. F.P.T. Baaijens, voor een
commissie aangewezen door het College voor
Promoties in het openbaar te verdedigen
op vrijdag 13 december 2019 om 11:00 uur

door

**Hadi Ahmadi Balef**

geboren te Ray, Iran

Dit proefschrift is goedgekeurd door de promotoren en de samenstelling van de promotiecommissie is als volgt:

| | |
|---|---|
| voorzitter: | prof.dr.ir. P.G.M. Baltus |
| $1^e$ promotor: | prof.dr. K.G.W. Goossens |
| $2^e$ promotor: | prof.dr. J. Pineda de Gyvez |
| leden: | prof.dr. M. Baradaran Tahoori (Karlsruhe Institute of Technology) |
| | dr. S. D. Cotofana (Technische Universiteit Delft) |
| | prof.dr. H. Corporaal |

*Het onderzoek dat in dit proefschrift wordt beschreven is uitgevoerd in overeenstemming met de TU/e Gedragscode Wetenschapsbeoefening.*

Dedicated to my lovely wife, Hamideh

*He who does not thank the people is not thankful to Allah.*

— Prophet Muhammad

## ACKNOWLEDGMENTS

"Glory be to Him Who has subjected this to us otherwise, we could never have accomplished this."

This thesis is the result of a four-year development process during which many people helped me.

First of all, I thank my first promoter, Kees Goossens, for giving me the opportunity to work on my PhD project, and trusting me. His supportive approach has been a blessing for me and his meticulous way of working not only helped me to improve the quality of this work but also helped me to develop some habits that will improve the quality of my future works. Furthermore, I would like to thank my second promoter, José Pineda de Gyvez, for sharing his priceless wisdom and experience with me generously. José has always been available for me when I wanted to discuss new ideas with him. He also gave me the chance to work at NXP Semiconductors. Many thanks to Hamed Fatemi for hosting me at NXP Semiconductors for more than two years, allowing me to have a real sense of the semiconductor industry. In addition, I would like to extend my gratitude to the members of my defense committee, Sorin Cotofana from the Delft University of Technology, Mehdi Tahoori from the Karlsruhe Institute of Technology, and Henk Corporaal from the Eindhoven University of Technology,

My PhD research is conducted at Electronic Systems group, working with many good colleagues. I had the opportunity to do two tapeouts in my project and doing these tapeouts and performing the experiments would not be achievable without helps from Andrew Nelson (RTL design), Mark Wijtvliet (PCB design), and Martijn Koedam (laboratory instruments). Having Kamlesh Singh and Paul Detterer was very valuable for me as we could learn a lot of things together. I also thank Hailong Jiao for helping me a lot in the beginning of my PhD project as my daily supervisor. I am grateful to the secretaries of the group, Marja and Feyza, for their kindness and cares. I have also enjoyed being together with other people in the group. Thank you Alessandro, Ali, Amr, Barry, Cumhur, Emad, Gabriela, Hamideh, Juan, Luc, Mahdi, Mahsa, Majid, Mojtaba, Rasool, Reinier, Sahar, Sajid, Shayan, Shima, Shubhendu, Sven, Victor, Yonghui, and Zhan.

I would like to also thank people from outside my professional environment for supporting me during these years of being far from home. Thank you Hamid Pourshaghaghi, Mohammad Tahghighi, Esmaeil Najafi, and all other people in our Iranian community. Special thank goes to my dear friend, Sobhan Niknam, who we could have very joyful times together.

A very special thank is in place for my parents, Hooshang and Shirin, who helped me to develop myself into who I am now. I would like to also mention my brother, thank you Dash Hooman! I also thank my wife's family for being strong support for us. I am very thankful to have my cute and sweet daughter, Helena, as she could make cheerful moments in my life.

And last, but foremost, my lovely wife, Hamideh. You gave me the energy and motivation and without your love and support, this PhD would not have been the same. You shared with me the difficult and the good times. Thank you very much for your patience and endurance.

Eindhoven, November 2019
Hadi Ahmadi Balef
هادی احمدی بالف

SUMMARY

## Efficient In-Situ Delay Monitoring for Chip Health Tracking

The growing use of digital integrated circuits in safety-critical applications together with variability issues of the technology scaling trends necessitate reliability awareness in digital circuits. Timing is one of the main concerns for the reliability of digital circuits as correct timing is important for the correct operation of the circuits. This thesis investigates techniques to realize an efficient and reliable chip health tracking technique based on monitoring circuit timing.

Continuous tracking of chip health status is essential to minimize the risk of failure. Therefore, the demanding chip-health tracking technique must be real-time, i.e., it should run in parallel to the main workload and it must predict failure. Besides, to make sure that the chip-health tracking system is reliable itself, it must provide enough coverage of critical circuit components. Otherwise, it fails to report a problem in the system. The technique must be also cost-efficient to make sure that the benefits of technology scaling are preserved. Moreover, the proposed solution must be able to be integrated into the standard IC design flow.

Based on these requirements, the main goal of this thesis is to develop an efficient technique for timing reliability of digital circuits. Following this goal, some contributions are made in this thesis.

**More Accurate Timing Model**: Accurate modeling of circuit timing and its unreliability effects are essential for developing a well-founded chip health tracking technique. A novel timing model is proposed for sequential components of the digital circuit (i.e., flip-flops). The proposed model takes the inter-dependency of timing constraints into account with more accuracy and less characterization effort and it is integrated into the timing analysis flow. Moreover, a new procedure is proposed for modeling the timing unreliability in standard IC design and timing analysis tools.

**Effective In-Situ Delay Monitoring**: In-situ delay monitoring is an advanced technique to sense timing slacks. A new in-situ delay monitoring technique is proposed in this thesis that inserts the monitors at particular circuit nodes to achieve cost reduction and increase coverage. With the new technique, the cost per monitor and the number of monitors are reduced, and the coverage and protectiveness of slack monitoring are improved.

**Novel Chip-Health Tracking System**: In sharp contrast to the conventional deterministic approach that does not take advantage of the underlying hardware in the most efficient way, a new chip health tracking is proposed which is based on a probabilistic approach. The probability of monitor excitation is captured as an indication of slack

values and delay degradation. Hence, by analyzing the monitor excitation rate, more information about delay degradation is extracted from each in-situ delay monitor.

**Within-Cycle Error Avoidance Technique**: A new timing speculation system is proposed in this thesis that masks the detected faults and avoids timing errors within one clock cycle. This error avoidance is necessary for increasing the reliability of the proposed chip health tracking technique for safety-critical applications. In this way, the failure risk is minimized by avoiding the occurrence of timing errors.

CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

*"Shallow men believe in luck or in circumstance.*
*Strong men believe in cause and effect."*

— Ralph Waldo Emerson, *The Conduct of Life*, 1980

# 1

# INTRODUCTION

Electronic systems are so widespread that it is impossible to find a piece of equipment that is not produced with them or does not contain them. Most of the electronic systems nowadays use semiconductor components. The advances in semiconductor technology have certainly been one of the key elements in making Apple, Alphabet (the parent company of Google), and Microsoft the top three largest companies of America as measured by market capitalization today [3]. Fig. 1.1 shows the largest firms of America over the past century. In 1917, more than 40% of America's largest companies (ranked by assets), were oil, steel or mining-related companies [1]. The invention of the transistor, as an alternative for the vacuum tubes, was announced on July 1, 1948, on New York Times, reported as "a device called a transistor, which has several applications in radio where a vacuum tube ordinarily is employed" [4]. In 1964, IBM introduced the industry's first high-volume, automatic, micro-miniature production of semiconductor circuits (Solid Logic Technology) [5]. Three years later, the most valuable company in America was IBM. Although IBM was the only technology company in that time, the technology firm was the second largest firm in America in 1967, as shown in Fig. 1.1. On July 18, 1968, Robert Noyce and Gordon Moore founded Intel, and three years later, Intel created the world's first commercially available microprocessor chip (Intel 4004) [6]. This microprocessor was built in $10\mu m$ semiconductor process technology and included 2300 transistors [7]. The semiconductor industry continued to advance with exponential growth in the number of transistors that are integrable into a circuit. This exponential growth has been asserted by the well-known Moore's law which states that in a dense *Integrated Circuit (IC)*, the number of transistors doubles about every 18 months at the same cost. In practice, it meant that the number of calculations per second per $1 raised from 728.6 in 1973 to 133300 in 1998 [8]. In 2017, Qualcomm introduced the world's first 10nm server processor (Centriq 2400) which has ~7.8 million times more transistors inside compared to Intel 4004 [9]. As shown in Fig. 1.1, in 2017 America's largest firm by sector is technology. Today, with adjusting for inflation, Apple is worth more than twice what IBM was worth fifty years ago [1].

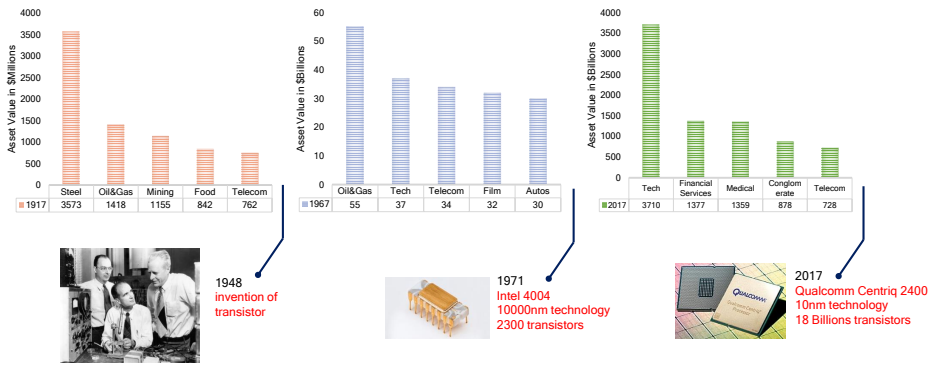



Figure 1.1: America's largest firms by sector in 1917, 1967, and 2017 [1].

The human's life has mainly been affected by industrialization, and the development of automation is inherent to industrialization because industrialization is defined to be the replacement of human or animal power by mechanical power [10]. A control system, as a necessary element of an automated system, is nowadays based on digital computers to perform the control calculations and transmit the proper commands [11]. Semiconductor technology scaling has been affecting human's life by making the computations faster, more power efficient, and cheaper [12]. However, technology scaling faces many challenges, including reliability [13].

The reliability of a system is connected with its failure rate [14]. The bathtub curves that are shown in Fig. 1.2 illustrate the effect of technology scaling on the failure rate of ICs. The bathtub curve is, in fact, the composition of three curves: early failures, constant failures, and wear-out failures. Depending on the dominance of each failure rate curve, the lifetime of the circuit is divided into three phases: Burn-in phase, normal life, and wear-out phase. The failure rate is high in the burn-in phase, mainly because of manufacturing defects, process variations, and design mistakes [14] [15]. The increased integration of transistors with technology scaling raises the rate of manufacturing defects, increases the spread in parameters of transistors and interconnects, and makes design more complicated [12]. During the normal life phase, the failure rate remains constant. In this phase, failures typically show themselves as temporary malfunctioning that happens randomly [14]. With technology scaling, the rate of these failures during the normal life period increases [12]. In the final phase of the lifetime, the failure rate increases due to silicon aging effects which cause permanent malfunctioning [14]. Silicon aging effects have become more pronounced in the scaled technology nodes [14]. These reliability issues have become critical considering the use of electronic systems in safety-critical applications.

Safety-critical systems are the systems which their failure causes injury or death, e.g., medical, automotive, or aviation systems. The use of electronic systems in safety-

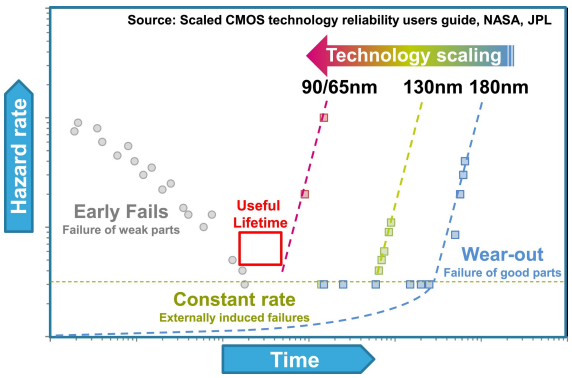



Figure 1.2: The effect of technology scaling on the failure rate bathtub curve [2].

critical applications has become more popular over the years as shown by the following examples. A variety of parameters are regulated and monitored by electronic systems in a heart-lung machine to perform the oxygenation and pumping functions properly [16]. The modern radiation therapy machines rely on electronic systems to design patient-specific parameters to collimate and direct radiation beams for treating localized cancer, considering that excessive radiation is life-threatening [17]. In a modern passenger car, *Advanced Driver Assistance Systems (ADAS)* take care of some safety-related features such as adaptive cruise control, lane change assistance, and collision avoidance [18]. Without the *Fly-By-Wire (FBW)* system, many modern airplanes would not be able to fly. FBW replaces the traditional flight control interface, which is mechanical, with the electronic interface to reduce mechanical complexity and saving weight [19]. These examples not only show the importance of reliability in electronic systems but also they imply the importance of having a mechanism that can report the reliability status of the system while it is working.

Considering the importance of reliability in electronic systems (especially for safety-critical applications) this thesis is about finding an efficient way to make systems aware of their reliability. In what follows, the reliability issues of digital ICs are briefly reviewed, and timing unreliability is explained. Then, we state the problem that this thesis addresses. The requirements of the solution to this problem are explained next. The state of the art solutions are then surveyed. Then, the contributions of this thesis are clearly explained. And finally, this chapter is concluded by outlining the rest of the chapters of this thesis.

## 1.1 RELIABILITY OF DIGITAL ICS

Digital circuits are essential components of electronic systems. They allow us to implement algorithmic solutions, such as signal processing, communication, and control

algorithms. Digital circuits take more advantage of technology scaling compared to their analog and RF counterparts [20], and reliability is the main issue that they face in the advanced process nodes. In this section, the critical reliability concepts that are used in this thesis are defined first. Then the sources of reliability issues for digital circuits are investigated. Finally, timing reliability as one of the primary requirement in digital circuits is explained.

### 1.1.1 *Definitions*

To define the terms more clearly, an example which implements the following function is considered

$$Y = a \times X + b, \tag{1.1}$$

where X and Y are input and output, respectively, while $a$ and $b$ are constants. The system is therefore made up of multiplication by constant $a$ and addition to constant $b$. A **physical source** is the natural source of a reliability problem, i.e., it is the root cause of the problem. Due to this effect, a **fault** may appear in the system as a "weakness, blemish or shortcoming of a particular hardware component or unit" [21]. In the example system, a fault means malfunctioning of the adder or multiplier or changing the constants $a$ and $b$. A fault can manifest itself in the system as an **error**, which is a deviation from the correctness of information inside the system [21]. In the example system, if a fault changes $a$, and the rest of the system is not faulty, the error does not happen as long as $X = 0$, while for other inputs, the fault manifests in the system as an error. Finally, a system **failure** occurs if it does not implement the intended function as a result of an error. In the example system, any deviation from the intended output according to (1.1) is considered as a failure for the system. Consequently, there is a cause-and-effect relationship between the physical source, fault, error, and failure.

**Reliability** is one of the most straightforward terms in the field with a quantitative definition. The reliability $R(t)$ is defined for a system as the probability of its correct operation during the time interval [0, t] assuming that the system is operating correctly at time 0 [14]. In electronic systems, reliability is often modelled as a function of failure rate $\lambda$ (shown in Fig.1.2) [14]

$$R(t) = e^{-\lambda t}. \tag{1.2}$$

According to (1.2), an electronic system is more reliable if the failure rate in that system is lower. The reliability status of a system can be tracked by sensing the physical sources of the problem, or by detecting faults or errors, or by observing system failure.

**Resilience** is defined as the "ability to deliver correct service adapting to disturbance, disruption, and change within specified time constraints" [21]. Therefore, a resilient system is inherently elastic and can adapt itself such that failure is avoided.

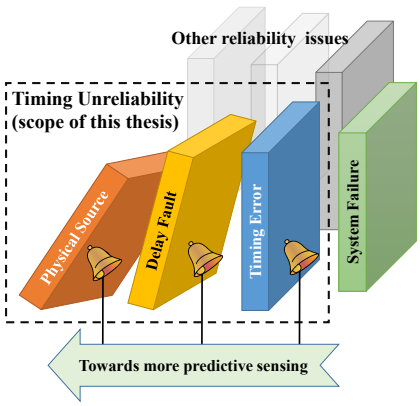



Figure 1.3: Cause-and-effect chain of timing unreliability.

### 1.1.2 *Delay-Fault and Timing Unreliability*

Digital circuits work based on logical operations which must be performed with appropriate timings. The logical operations are performed through logical gates that are based on compositions of transistors. The delays of logical gates and the wires which connect them are analyzed with design tools, and the selection of circuit structure and gates is made based on the specified design constraints. The design style of digital circuits is predominantly synchronous. A synchronous design is based on logical gates, memory elements, and a clock signal that regulates the timing in the circuit. The data which propagates inside a digital circuit can become erroneous if the circuit delays change such that the timing regulation which is imposed by the clock signal is violated, i.e., a timing error happens. If the timing error changes the functionality of the circuit, the circuit fails to perform its intended function. Therefore, it is essential to make sure that the circuit timing is reliable. Traditionally, the circuits are over-designed to make sure that their timings are reliable. This over-design approach is based on pessimistic timing characteristics at the design time.

The cause-and-effect chain of timing unreliability is shown in Fig. 1.3. Process variations, voltage drop, and temperature variations can be the physical sources of delay degradation in circuit components. Furthermore, silicon aging effects can increase the delays of the circuit components over time. Delay degradation is the *fault* which causes timing *error* in digital circuits. If timing errors are not prevented, they may result in system *failure*. The cause-and-effect chain of timing unreliability guides us towards techniques to increase the reliability of digital circuits and monitor their reliability (status) effectively.

### 1.1.3 *Physical Sources of Timing Unreliability*

The main physical sources that cause timing unreliability problems for the timing of digital circuits are variability and silicon aging effects. In this section, these effects are briefly reviewed.

The sources of variability are process and environmental variations [12, 22]. **Process variations** are due to the manufacturing process, while **environmental variations** affect the circuit when it is operating [22]. Variability causes deviations in the characteristics of circuit components. Extreme conditions are assumed at the design time to have a circuit which is reliable even in the presence of these effects. Alternatively, variability effects are modeled more precisely with a statistical approach to reduce the inherent pessimism of extreme condition assumption [22]. The environmental variations include temperature and supply voltage. The design should ideally work at the designed voltage, but due to effects like tolerance of voltage regulators, IR drops, or noise, the operating voltage can change. Similarly, the operating temperature of the circuit can change due to its environmental temperature or its own activity. The variations of temperature and supply voltage affect the characteristics of the circuit components. Technology scaling magnifies the sensitivity of circuit components to the environmental variations [23].

The effects of process variations are divided into global and local factors. Global process variations affect the parameters of all components of a die in the same way, while the effects of local process variations vary across the components of the same die. In [15], the global and local variability effects are characterized by inverter-based *Ring Oscillators (RO)* in different technology nodes. The results of their experiments are plotted in Fig. 1.4-(a) for 7-stage inverter RO and 29-stage inverter RO. As can be observed, global variations had a decreasing trend until 65nm while, and it increased again in the 40nm process node. On the other hand, the effect of local variations has been monotonically increasing with technology scaling trend. Traditional sources of process variations, such as mask offsets, mainly cause the global variation effects [24] while entering into atomic sizes with technology scaling magnifies the local variations. In Fig. 1.4-(b), the primary sources of local process variations in the smaller technology nodes are shown. The first effect is called *Line Edge Roughness (LER)*, and it explains the fluctuations in lines due to the lithography limitations in the fabrication process. The other effect is called *Random Dopant Fluctuations (RDF)* which is about the fluctuations in the number of dopant atoms (which is a discrete number) inside the channel of transistors. Shrinking the sizes with technology scaling increases the relative effect of these fluctuations on the parameters of circuit components [24] [15].

Silicon aging effects are temporal reliability issues, i.e., they cause change in the characteristics of circuit components over its lifetime. The extent of these effects depends on the stress factors (i.e., working conditions of the circuits) and time. Some silicon aging effects manifest themselves by changing the performance parameters of the circuit (e.g., power or delays) gradually, while others lead to abrupt malfunctioning. Technology scal-

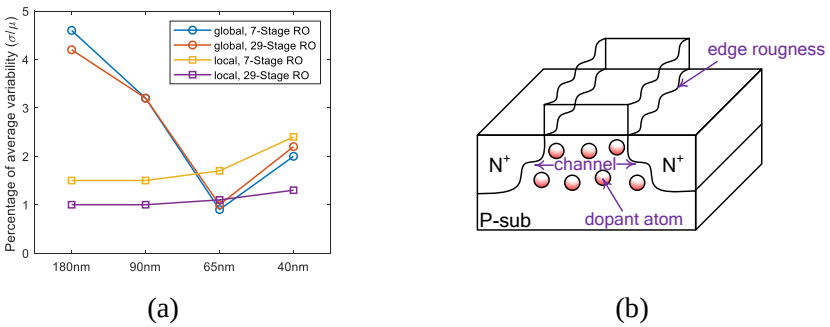



Figure 1.4: Variation effects. (a) The effect of process variations on inverter-based ROs with 7 and 29 stages in different technology nodes. (b) Cross-section of a transistor illustrating the roughness in the channel edges and the randomly placed dopant atoms inside channel.

ing has increased the effects of silicon aging [15, 24, 25]. The most severe silicon aging effects in sub-90nm technologies are *Time-Dependent Dielectric Breakdown (TDDB)*, *Hot Carrier Injection (HCI)*, *Bias Temperature Instability (BTI)*, and *Electromigration (EM)* [25]. TDDB, HCI, and BTI effects are sources of unreliability for transistors, while EM affects circuit wires. TDDB and EM cause circuit breakdown abruptly, while BTI and HCI affect circuit timing gradually [15].

## 1.2 PROBLEM STATEMENT

Reliability of electronic systems is getting harder due to the technology scaling trends and challenges. Furthermore, the growing use of electronic systems in safety-critical applications intensifies the necessity of reliability awareness in electronic systems. Digital ICs are the typical component of almost all electronic systems, and they take the most advantage of technology scaling. However, the physical sources of unreliability, such as silicon aging and variability, challenge their reliability. Timing is one of the main concerns for the reliability of digital circuits as correct timing is important for correct operation of the circuits. Monitoring the timing reliability of digital circuits is therefore essential. This thesis investigates techniques to realize an efficient and reliable chip health tracking technique. The problem statement for this work is, therefore: *How to monitor the timing reliability of a digital circuit provided that the solution must 1) be reliable considering variability effects, 2) be cost-efficient to preserve the benefits of technology scaling, and 3) predict failures to avoid them, required in safety-critical applications.*

## 1.3 REQUIREMENTS

Based on the problem statement, the following requirements are set for the demanding chip health tracking solution.

### 1.3.1 *Real-Time*

Continuous tracking of chip health status is essential to minimize the risk of failure. In safety-critical applications, the system must be checked during its operation because any risk of failure must be minimized in these applications. Therefore being able to perform continuous chip health tracking during operation (i.e., being real-time) is another requirement for the demanding chip health tracking technique.

For real-time chip health tracking, the tracking mechanism must run in parallel to the main workload of the circuit, i.e., the technique is not allowed to stop the system. Furthermore, the technique must be predictive of system failure, rather than being a detective technique. Being predictive allows minimizing the probability of failure by having some time to avoid failure. On the other hand, the detection-based approach tries to compensate for the failure effects, which is expensive and unacceptable for safety-critical systems. Since circuits are typically over-designed to be reliable, if the technique can report the available margin, it can predict the future failures, i.e., if the margin is low (high), the chance of failure is high (low).

### 1.3.2 *High Coverage*

The demanding chip health tracking technique can be seen as a system itself. The expected functionality of the system is to report all faults in the monitored system. The health tracking system fails when it does not report a fault in the monitored system, while the fault exists in reality. Hence, a reliable chip health tracking system must have high enough coverage. The definition of coverage here is similar to the definition of test coverage for *Design for Test (DfT)* techniques. In DfT techniques, the probability of a chip with fault not passing the tests is (1-T) where T is the test coverage. Therefore, it is necessary for a demanding chip health tracking technique to observe faults in all critical components of the circuit without missing any of them.

### 1.3.3 *Cost-Efficient*

Technology scaling has enabled digital circuits to be faster, more power efficient, and smaller. Any solution for the reliability problem which is faced by technology scaling must be cost-efficient in terms of speed, power, and area overheads. Otherwise, the benefits of technology scaling can diminish substantially. Therefore, one of the requirements

for the demanding chip health tracking is being cost-efficient in terms of speed, power, and area overheads.

### 1.3.4 *Integrable in Standard IC Design Flow*

The proposed solution for chip health tracking must be able to be integrated into the standard IC design flow. For this purpose, the technique must be based on a solid understanding of circuit timing and reliability models. Furthermore, the technique must be developed based on the standard IC design flow and the capabilities of industrial IC design tools. Otherwise, the reliability and practical feasibility of the technique are not appealing.

## 1.4 STATE OF THE ART

In this section, the state of the art in monitoring the timing reliability of digital circuits is reviewed. Based on what which is sensed in the cause-and-effect chain of timing unreliability (see Fig. 1.3), the state of the art is classified as follows:

- Physical source sensing techniques: The techniques in this category sense the physical source of timing unreliability, i.e., process, voltage, and temperature variations, and aging effects.
- Delay-fault sensing techniques: The techniques in this category aim to sense deviations of delay characteristics in digital circuits.
- Timing error sensing techniques: The techniques in this category check if the data has become invalid due to timing error.

In Fig. 1.5, the categories of related works are illustrated. In what follows, the related works in each of these categories are reviewed and assessed based on the set of requirements that were discussed before.

### 1.4.1 *Physical Source Sensing Techniques*

The effect of process variations on circuit performance is traditionally captured with worst-case assumptions for delays, i.e., adding timing margin. However, considering the worst-case delays imposes design costs in terms of power and area to meet a target speed for the circuit. The effects of process variations on the circuit performance are captured by employing either RO circuits [26, 27] or sensing the parameters of transistors such as threshold voltage [28–31]. Adding timing margins has also been the traditional way to address supply voltage variations. Supply voltage sensors are employed to reduce the cost of traditional over-design approach. In [32], the supply voltage sensors are categorized into three groups: Based on comparison to an external reference [33], over/under-shoot-detection-based [34], and Delay-based [35]. Furthermore, novel techniques like [36] predict dynamic delay variation due to the voltage drop caused by work-

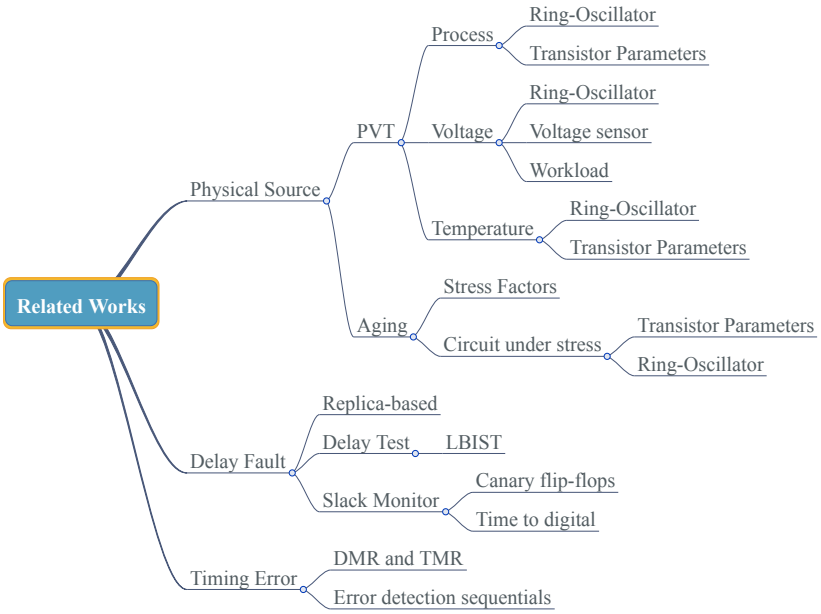



Figure 1.5: The categorization of state of the art works in timing unreliability monitoring.

load variations with machine learning models. Due to the considerable effect of temperature variation on the circuit parameters, and the importance of temperature for circuit reliability, design of temperature sensors with high sensitivity has been extensively researched. Similar to process variation sensors, the existing temperature sensors are RO based (see, e.g. [37–39], or based on sensing the transistor parameters such as drain current (see, e.g. [40–42]. The cause of silicon aging effects is some specific physical stress on the transistors, e.g., temperature and voltage. Techniques that sense the silicon aging effects on circuit timing are either based on sensing the stress factors [43–46] or the extent of which the parameters of a transistor [47] (or the frequency of a RO [48]) under stress is degraded.

The techniques which are based on sensing the physical source of timing unreliability use sensors which are smaller than the main circuit. Therefore, the area, speed and power cost of these techniques are modest. Furthermore, since these techniques do not intrude the main design, their functionality does not require stopping the main circuit, and they can perform real-time chip health tracking in parallel to the main workload. Since these techniques are based on sensing the cause of timing unreliability, they are inherently predictive. Furthermore, the sensors can be designed to be more pessimistic and report the problem before the main circuit fails. Integration of the sensors in the standard IC design flow is not challenging as they can be considered like any other

standard macro which is instantiated and physically placed and routed inside the design. However, the coverage of physical source sensors is minimal because they are not truly part of the main circuit and have limited accuracy in sensing the actual timing degradation of circuit components.

### 1.4.2 *Delay-Fault Sensing Techniques*

Any deviations from operational delay values of circuit components must be sensed to detect delay-faults in a circuit. There are three approaches to sense delay-faults: replica-based, delay testing, and slack monitoring.

In the replica-based techniques (see, e.g. [49–51], the critical timing components of the circuit are replicated, and their delays are measured. The main challenge in these works is the proper selection of critical components at the design time [52]. That is because replicating too many components of the circuit can annul the main advantage of this approach which is being low-cost in terms of power, area, and speed. Similar to physical source sensing techniques, the replica-based techniques can run in parallel to the main design, but they are less predictive than those techniques. Integrating the replica-based techniques is challenging because the timing-critical components of the circuit are identified at the very late design stages which means that the sensing circuitry should be added to the design when the design is almost finalized. This approach can add too much engineering effort and limit the accuracy of the technique. Finally, although the correlation between the delay defect in the main circuit and the sensed delay defect with the replicated circuit increases with this approach, still the number of replicated components are limited and the main components are not directly monitored with this approach. Therefore, the coverage of these techniques is limited.

DfT techniques add hardware to the main design and allow testing the IC for manufacturing faults. The use of DfT hardware to detect delay-faults is a standard technique [53]. Testing is performed by applying some test data to specific internal nodes of digital circuits, running the circuit to change the internal node data, and collecting the new internal data (test results) to compare them with a golden reference. *Built-In-Self-Test (BIST)* hardware is added to the main circuit to apply the test data and compare the test result in the field. Use of BIST for delay testing has been considered in the literature [54–62]. However, the hardware overhead of this approach is considerable. Furthermore, testing the circuit for delay-faults requires stopping the main functionality and applying test vectors and collecting the test results afterward. Since with this approach delay-faults are sensed, if some pessimism is added to the test [61] the test result can be predictive of the problem. Delay testing uses the mature DfT insertion flow which is integrated into standard IC design flow. The additional hardware which is specific for delay testing (mainly to manipulate the clock signal) is also nowadays added to the design in a standard approach using commercial IC design tools. However, the main challenge for on-line delay testing is the coverage. Since delay testing relies on activating the delay-

faults by applying the test vectors, the coverage of the approach depends on the testing data. Most of the research efforts in this field have been dedicated to solve this issue.

Slack monitoring is another approach for delay-fault sensing. In the digital circuits, the clock signal determines the times at which the data is captured. If the delays increase in the system, the proper data is not captured, and a timing error happens. Slack is defined as the difference between the latest time that the data that should be captured arrives and the capturing moment which is determined by the clock signal. In practice, due to the worst-case assumption during design time, the slack value is non-zero (which indicated as the design margin). The slack monitoring techniques measure the slack value and report it as an indication of circuit delays. If the slack is smaller than a specific value, a delay-fault has occurred. The slack monitoring techniques are either based on canary flip-flops [63] or time-to-digital converters [62]. Both approaches add additional hardware components that capture the data earlier than the clock capturing moment. If the earlier captured data is different from the main captured data, it is considered as a warning signal that warns tight slack in the design (i.e., a delay-fault). Although slack monitoring techniques can directly sense the delay-faults of a critical component in the circuit, since during optimizations with circuit design tools a large number of circuit components become timing critical [52], the cost of slack monitoring techniques is too high. Slack monitoring techniques can run in parallel to the main workload of the design without requiring to stop the main functionality. Furthermore, these techniques detect delay-fault which makes them more predictive compared to timing error detection techniques. The challenge for integration of these techniques in the standard IC design flow is identifying the critical components which should be monitored. Finally, the coverage of these techniques relies on activation of delay-faults during normal operation of the circuit. Since not all delay-faults are activated during the normal operation of the circuit, the coverage of these techniques is limited.

### 1.4.3 *Timing Error Sensing Techniques*

A timing error occurs when the data inside the circuit becomes invalid due to delay-faults. Generally, error detection techniques are based on some redundancy, i.e., hardware, information, ..., is replicated and the corresponding data from the primary and redundant counterpart(s) are compared to detect error occurrence [64]. The standard techniques in this category are *Double Modular Redundancy (DMR)* and *Triple Modular Redundancy (TMR)* techniques. These techniques can be employed with different design granularities (e.g., circuit components or systems-level). Using redundancy at the system-level suffers from common cause failure problems for aging [65]. In [66] redundancy was introduced to the sequential components of digital circuits (i.e., flip-flops) for the first time to achieve soft error resilience. The idea was later extended in [67], known as Razor flip-flops. Razor flip-flops detect timing error by sampling the data multiple times and comparing the results to detect a timing error. Later, the idea of Razor

Table 1.1: Assessment of the state-of-the-art works in timing unreliability monitoring based on the set of requirements for the demanding chip health tracking system.

| Technique | cost | real-time | | Integrability | coverage | |
|---|---|---|---|---|---|---|
| | | parallel | predictive | | in-situ | critical elements |
| physical source | low | yes | yes | good | no | none |
| replica-based | low | yes | yes | moderate | no | low |
| delay testing | high | no | yes | good | yes | low |
| slack monitoring | high | yes | yes | poor | yes | low |
| error detection | high | yes | no | moderate | yes | high |

flip-flops was investigated extensively, and different flavors of the idea were introduced in the literature [68–73]. These techniques are generally known as *Error Detection Sequential (EDS)*. The main advantage of EDS techniques is that they sit at every flip-flop and in this way they sense if any data has become invalid due to timing degradation. Note that error detection with this approach still relies on the error activation by the workload of the design. Furthermore, they can run in parallel to the main workload without requiring to stop the main functionality. However, the cost of this approach is too high due to the significant design overhead, and they perform detection instead of a prediction of timing error. Integration of these techniques into the standard IC design flow is done by replacing the regular flip-flops with the EDS counterparts and setting appropriate timing constraints. Therefore, these techniques can be integrated into the standard IC design flow.

### 1.4.4 *Summary of Related Work*

In Table 1.1, assessment of the state of the art is summarized. Increasing the coverage of physical source based and replica-based techniques requires adding too many sensing which annuls the low-cost benefits of those techniques. Delay testing techniques inherently cannot be real-time because applying the test data and collecting the test results requires stopping the circuit and its main functionality. Similarly, error detection techniques cannot be real-time techniques because they are naturally detect timing errors instead of predicting them. On the other hand, slack monitoring techniques can be improved and could then fulfill the requirements. The focus of this thesis is investigating the slack monitoring based techniques and improving their cost-efficiency, integrate them into the standard IC design flow, and improve their coverage.

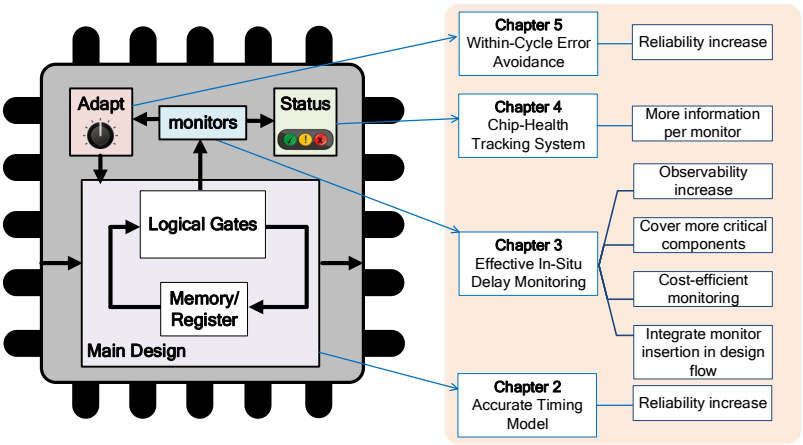



Figure 1.6: Thesis contributions.

## 1.5 APPROACH AND CONTRIBUTIONS

Based on the set of requirements that were discussed in Section 1.3, and the state of the art works, the main goal of this thesis is to develop a reliable, low-cost, and predictive technique for monitoring timing reliability of digital circuits based on in-situ delay monitoring. Following this goal, some contributions are made in this thesis. These contributions are shown in Fig. 1.6 and briefly explained in what follows.

### 1.5.1 *More Accurate Timing Model*

Accurate modeling of circuit timing and its unreliability effects are essential to develop a well-founded chip health tracking technique. In Chapter 2 of this thesis, the concepts related to the timing of the digital circuit is reviewed, and timing unreliability is modeled. Besides, a novel timing model is proposed for sequential components of the digital circuit (i.e., flip-flops). The proposed model takes the inter-dependency of timing constraints into account with higher accuracy and less characterization effort. The proposed model is integrated into the timing analysis flow. Consequently, the design efficiency is improved in terms of power, area, and speed, and timing sign-off is moderated by reducing pessimism in timing analysis. Moreover, a new procedure is proposed in Chapter 2 to model the timing unreliability in standard IC design and timing analysis tools.

### 1.5.2 *Effective In-Situ Delay Monitoring*

As was discussed before, slack monitoring techniques have an excellent potential to evolve towards the required chip health tracking system. In-situ delay monitoring is an advanced technique to sense timing slacks. A new in-situ delay monitoring technique is proposed in Chapter 3 of this thesis that inserts the monitors at particular circuit nodes to achieve cost reduction and increase coverage. With the new technique, the cost per monitor and the number of monitors are reduced, and the coverage and protectiveness of slack monitoring are improved.

### 1.5.3 *Novel Chip-Health Tracking System*

Based on the in-situ monitoring technique of Chapter 3, a new chip health tracking system is proposed in Chapter 4 of this thesis. In sharp contrast to the conventional deterministic approach that does not take advantage of the underlying hardware in the most efficient way, the proposed technique is based on a probabilistic approach. In practice, multiple slack values are sensed at each in-situ delay monitor. Each monitor is excited depending on the activation of the sensed slacks. The probability of monitor excitation increases when more slack values become critical due to delay degradation. Hence, by analyzing the monitor excitation rate, more information about delay degradation can be extracted from each in-situ delay monitor. Based on this fact, a new chip health tracking is proposed which is more efficient and precise compared to the conventional methods.

### 1.5.4 *Within-Cycle Error Avoidance Technique*

Timing speculation systems are generally used for better-than-worst-case design. Instead of putting large slack margins by designing at the worst-case corner, timing speculation increases the slacks dynamically if necessary. A new timing speculation system is proposed in Chapter 5 of this thesis to guarantee the coverage of the proposed chip health tracking approach. That is because the health tracking approach is based on in-situ monitoring and thereby its coverage relies on the activation of timing faults. The proposed timing speculation masks the detected faults and avoids timing errors in this way. This error avoidance is necessary to increase the reliability of the proposed chip health tracking technique for safety-critical applications. In this way, the failure risk is minimized by avoiding the occurrence of timing errors. In Chapter 5 of this thesis, a low-overhead timing speculation system is proposed that prevents timing errors by adjusting the clock period within one clock cycle.

## 1.6 THESIS OUTLINE

The remainder of this thesis is organized as follows. In Chapter 2, circuit timing concepts and threats are defined and modeled. Besides, a new timing model is proposed for flip-flops, and a new procedure is proposed to model timing unreliability in standard IC design and timing analysis tools. The proposed in-situ delay monitoring is introduced in Chapter 3, and the design of the monitor and their insertion technique are explained. Based on this idea, a novel chip health tracking is proposed in Chapter 4. In Chapter 5, a new timing speculation system is proposed to predict and avoid timing error within one clock cycle. Furthermore, silicon results of implementing the initial version of the ideas are provided in Chapter 6. Finally, Chapter 7 concludes this thesis and some ideas are proposed for future works.

*"The assumption of an absolute determinism*
*is the essential foundation of every scientific enquiry."*

— J.L. Heilbron, *Max Planck, The Dilemmas of an Upright Man*, 1986

# 2

# MODELING CIRCUIT TIMING AND ITS UNRELIABILITY

## 2.1 INTRODUCTION

Complex digital ICs include billions of transistors. Design abstraction is a must to manage such huge designs. Therefore, design tools use models to be able to manage circuit components properly. However, the limited accuracy of these models is a significant source of uncertainty in the design flow, which results in an optimistic or pessimistic design. A design which is based on larger (smaller) than actual delays is pessimistic (optimistic). Since being functional has the priority, the pessimistic design is generally preferred over the optimistic design. Hence, the traditional static over-design approach is based on the worst-case corners to make sure that the design is functional in the presence of uncertainty effects. However, for most of the manufactured IC samples and in most of the operating conditions, there is a large timing slack in the design. Employing more accurate models in IC design tools is thereby essential to reduce the unnecessary design margins and benefit more from technology scaling. Techniques like dynamic voltage/frequency scaling reuse the timing slack at run-time to provide a better power-performance trade-off. Nevertheless, it is vital to guarantee that the expected functionality of the circuit is still preserved. However, tightening slack margins mandates more accurate timing analysis to make sure that a timing violation does not happen due to missing some picoseconds of slack margin.

In this chapter, the concepts and definitions that are relevant in the context of timing in digital circuits are reviewed to serve as the basis of the rest of this thesis. Furthermore, the silicon aging effects, as the effects that can endanger circuit timing are briefly reviewed. A new framework is proposed to model the effect of aging on circuit timing. A new analytical model is proposed for more accurate calculation of timing slacks. The model is more accurate by considering the inter-dependency between timing constraints. With the proposed model, the design is made less pessimistic or optimistic and more efficient and reliable.

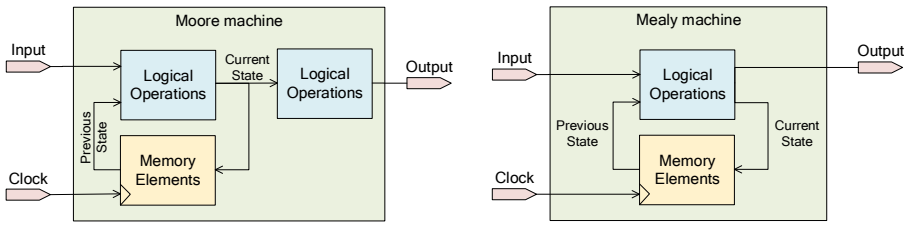



Figure 2.1: Mealy and Moore Finite State Machines.

## 2.2 TIMING OF DIGITAL CIRCUITS

The input-output relation that a digital circuit implements is most widely described with a *Finite State Machine (FSM)*. In Fig. 2.1 the block diagram of two types of FSM are illustrated. At each point of time, the machine is in a specific state, and the output of the machine is determined based. In the Moore machine the output is determined solely based on the current state, while in the Mealy machine, the output is determined based on the input and the state. The state of the machine is determined based on the inputs and the previous state. To have access to the previous state, FSM requires memory elements. Logical operations are performed to obtain the state from the previous state and the inputs. Also, the outputs are obtained from performing logical operations on the current state (and the inputs). This model of computation is typically implemented using digital circuits by using combinational gates (e.g., AND, OR, NOT) for logical operations and sequential gates (e.g., flip-flop) for memory elements. Nearly all digital circuits are nowadays synchronous, i.e., a clock signal, which is a periodic pulse, rules the timing of the circuit by indicating the moment that the memory element must capture the calculated state.

### 2.2.1 *Definitions*

In practice, a digital circuit typically consists of multiple stages of combinational logic gates ending to flip-flops, as shown in Fig. 2.2. Each combinational logic stage is a network of logical gates. Each flip-flop is a link between the output of a stage and the input of another or the same stage. The design speed is determined based on the delays of combinational gates, flip-flops, and the clock signal path. The capturing moment of a flip-flop is typically the rising (a.k.a positive) edge of the clock. If the delays are such that the appropriate value is not present (and stable) at the flip-flop at the capturing moment, an erroneous value is captured, i.e., a timing error occurs. To formulate this criteria the relevant terms are defined in what follows.

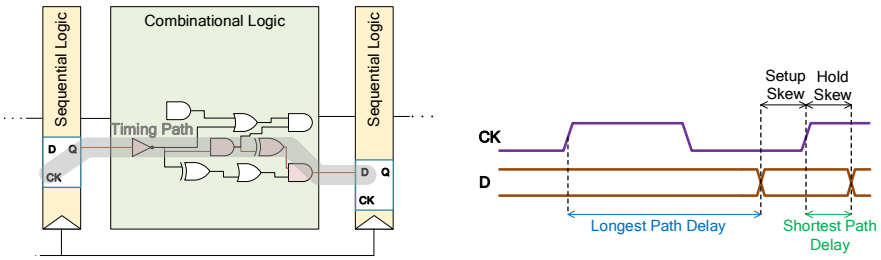



Figure 2.2: The circuit timing model. (a) timing path, and (b) waveforms at the destination flip-flop.

For each input-to-output relation (a.k.a. **timing arc**) of a logic gate or wire, a specific **delay** is considered which depends on the driving strength of the gate and the capacitance loading of the output. A **timing path** is a sequence of connected timing arcs starting from a flip-flop and ending at another or the same flip-flop (see Fig. 2.2). In a typical digital circuit, there are several timing paths. The delays of the shortest and the longest paths must be analyzed and optimized such that the circuit timing constraints are met.

In Fig. 2.2, the waveforms at each flip-flop are shown. Since the clock signal also passes through some gates (i.e., clock buffers), the longest path and the shortest path include clock buffers, flip-flops (the source flip-flops and the destination flip-flop), and combinational logic. The delay of the longest path $d_{lp}$ (shortest path $d_{sp}$) determines how long before (after) the clock capturing edge the data becomes stable. According to the definitions, setup (hold) skew is the time difference between the change of the data before (after) the clock capturing edge. Therefore, the **setup skew** is

$$d_{ss} = t_{clock} - d_{lp}, \tag{2.1}$$

where $t_{clock}$ is the clock period, and **hold skew** is

$$d_{hs} = d_{sp}. \tag{2.2}$$

Hence, the flip-flop input data is a pulse that is stable for $d_{ss}$ before the clock capturing edge and remains stable for $d_{hs}$ after the clock capturing edge, with a width

$$W_D = d_{ss} + d_{hs}. \tag{2.3}$$

If $d_{ss}$ and/or $d_{hs}$ are too small, input data is not captured by the flip-flop. To guarantee the capturing of input data, the minimum allowed $d_{ss}$ and $d_{hs}$ are specified as the timing constraints of flip-flops. The former is the **setup time** $d_{st}$, while the latter is the **hold**

**time** $d_{ht}$ of the flip-flop. Traditionally, $d_{st}$ and $d_{ht}$ are found by sweeping the corresponding skew while the counterpart skew is fixed. $d_{st}$ and $d_{ht}$ are stored in timing libraries to be used for timing analysis of the circuit. The timing analysis tool reports **setup slack** $Slk_s$ and **hold slack** $Slk_h$ for all flip-flops as

$$Slk_s = d_{ss} - d_{st}, \tag{2.4}$$

and

$$Slk_h = d_{hs} - d_{ht}, \tag{2.5}$$

at each flip-flop.

The definitions of $d_{st}$ and $d_{ht}$ are based on specific criteria on the delay of flip-flop from its clock input to its data output (i.e., clock-to-Q delay) increase. In Fig. 2.3, the definition of $d_{st}$, $d_{ht}$, and clock-to-Q delay $d_{cq}$ are illustrated. When $d_{ss}$ and $d_{hs}$ are relatively large, $d_{cq}$ is at its lowest value. This low value of $d_{cq}$ is called clock-to-Q contamination delay $d_{ccq}$). If the input data of a flip-flop changes close to the clock capturing edge (i.e. small $d_{ss}$ and/or $d_{hs}$), the flip-flop delay from its clock input to its data output $d_{cq}$ increases, i.e. the flip-flop enters into metastability (see the plots illustrated in Fig. 2.3). To define $d_{st}$ and $d_{ht}$, a specific criterion is selected for the upper bound of $d_{cq}$. The upper bound for $d_{cq}$, which is called the clock-to-Q propagation delay $d_{pcq}$, is normally chosen to be 10% longer than $d_{ccq}$.

### 2.2.2 *Setup/Hold-Time inter-dependency*

Traditionally, setup time and hold time are characterized independently. The independent characterization methods result in constant setup (hold) time that is independent of the counterpart skew. As illustrated in Fig. 2.4, a surface is obtained for $d_{cq}$ by sweeping $d_{ss}$ and $d_{hs}$ together. The intersection of this surface with the constant $d_{cq} = d_{pcq}$ surface defines the valid points of $d_{st}$ and $d_{ht}$. The setup/hold-skew space is therefore divided into the capturing and violation regions. All the points that are on the boundary between the capturing and the violation regions are valid setup time and hold time points. As illustrated with the blue curve in Fig. 2.4, the actual value of hold time (setup time) increases by decreasing the setup skew (hold skew). This is called the inter-dependency between the setup time and hold time of flip-flops [74–76]. Note that the independently characterized setup/hold-time can be either larger (the green curve) or smaller (the red curve) than the actual values.

## 2.3 GRADUAL TIMING DEGRADATION EFFECTS

As discussed in Chapter 1, the significant sources of timing unreliability are variability and silicon aging effects. The design and analysis tools take the variability effects into

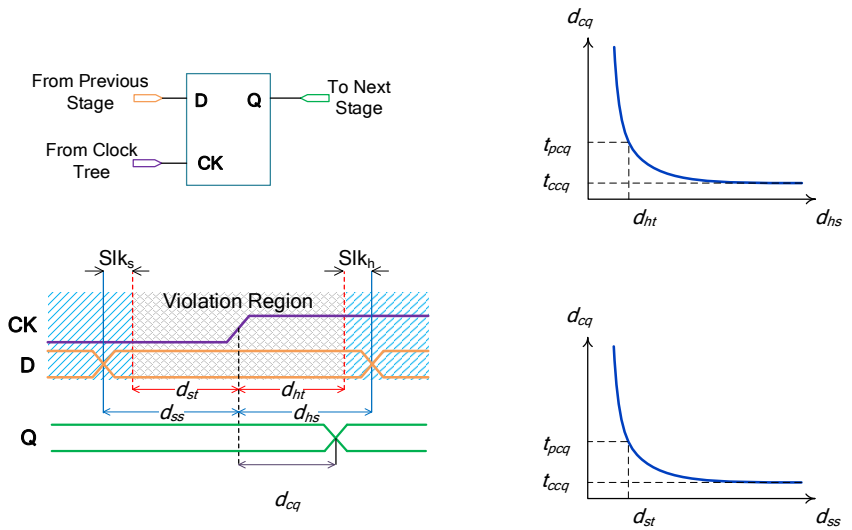



Figure 2.3: Timing definitions at a flip-flop.

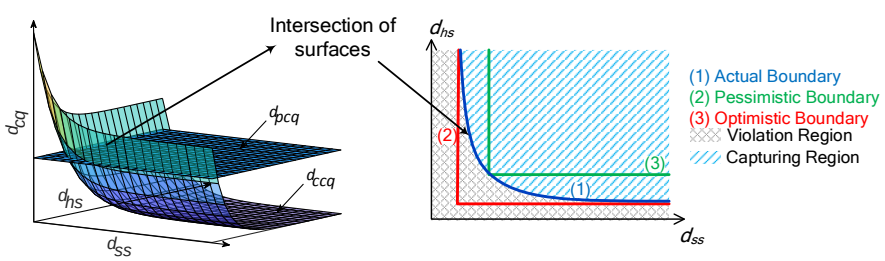



Figure 2.4: The definition of setup/hold-time interdependency curve.

account by characterizing the delays at the different process, voltage, and temperature corners. However, the effects of aging depend on the workload of the system. Hence, the static corner-based approach does not provide enough accuracy for timing analysis; i.e., it makes the analysis either too pessimistic or optimistic. Therefore, a new methodology is needed to consider the dynamic behavior of the circuit workload and calculate the degraded delay values. BTI and HCI are prominent transistor degradation phenomena that affect the circuit performance gradually. The parameters of transistor such as its threshold voltage ($V_{th}$) degrade with aging, resulting in overall performance loss. In this section, the physics of gradual degradation effects are briefly reviewed.

### 2.3.1 *Bias Temperature Instability*

The overall effect of BTI is a composition of two phases: stress and recovery. In the stress phase, the parameters of the transistor degrade, e.g., the magnitude of threshold voltage value increases. In the recovery phase, a fraction of the degradation can be recovered. There are two explanations for the BTI phenomena: *Reaction-Diffusion (RD)*, *Trapping-Detrapping (TD)*. In the classical RD explanation, interface traps (which trap carriers) cause the transistor to have lower mobility. Thus, in the manufacturing process, the Silicon/Oxide interface is annealed with hydrogen to reduce the dangling bonds of silicon atoms. However, these Silicon-Hydrogen bonds break under voltage stress and generate the traps. The Silicon-Hydrogen bonds break because the inversion holes in the PMOS channel interact with the Silicon-Hydrogen bond at the Silicon/Oxide interface. This effect is stronger at higher temperatures [77]. As mentioned before, BTI is modeled as a shift in transistor parameters (e.g., threshold voltage, trans-conductance, saturation current). This shift is due to the generation of interface traps ($N_{IT}$) when the device is under stress. According to [78] the RD model has several limitations like not being able to explain the experimental results indicating the sensitivity of degradation to the applied gate bias. Also, the recovery starts sooner and lasts more than predicted by RD-based models. Some authors [79] tried to solve these problems by using TD (a.k.a. charge trapping) based explanation, which was the first explanation of the BTI effect (before 1977). In the TD-based explanation, the model of a similar case of *random telegraph noise (RTN)* is used, which means a statistical component is considered for BTI. In the TD-based model for BTI, each transistor is characterized by the number of defects, the capture-emission time for each defect, and the impact of charging of each defect on the threshold voltage. The time constants of capture and emission depend on the voltage and temperature [78].

### 2.3.2 *Hot Carrier Injection*

The HCI effect is an aging mechanism which is mainly observed in NMOS devices. HCI can also increase the NBTI effect, and in this regard, it also affects the PMOS devices.

Considering the slower scaling rate of supply voltage compared to the scaling rate of transistor dimensions, the HCI effect is becoming more significant with technology scaling. Most of the models of HCI in the literature are based on the lucky electron explanation of the effect [80]. Based on this explanation, the carriers can obtain high energy, cause the generation of interface states, enter to the gate oxide, and cause damages there. This mechanism with the described behavior happens when the carrier is accelerated in the channel and deviates from its trajectory due to the high gate-drain electric. The mentioned mechanism is called *Channel Hot Carrier (CHC)*. The other models in the literature consider other mechanisms of heating the carriers, such as *Substrate Hot Carrier (SHC)*, *Drain Avalanche Hot Carrier Generation (DAHC)*, and *Secondary Generated Hot Carrier Injection (SGHC)* [25]. However, the CHC mechanism is much worse than the other mechanisms in current and future technologies. While BTI damages are distributed in the channel, the HCI damages are concentrated at the drain end of the channel. In contrast to the BTI effect, HCI effect cannot be recovered, i.e., there is no recovery phase for HCI. It should be noted that the stress time for HCI effect is the time that the transistor is conducting (i.e. transient time between logic one and zero). For the NBTI (PBTI) the stress is during the time which the transistor is negatively (positively) biased. While in the short term the HCI effect is dominated by the BTI effect, in the long term the HCI effect becomes more significant compared to the BTI effect.

### 2.3.3 *Aging Effect on Circuit Timing*

As discussed before, the delay of a flip-flop from its clock input to its data output ($d_{cq}$) is affected by the data skews (i.e., $d_{ss}$ and $d_{hs}$). Low data skews result in flip-flop meta-stability which increases its $d_{cq}$. Due to the typical design margins, for a fresh (i.e., undamaged) circuit, there is positive slack, i.e., data arrives well before the clock capturing edge. Therefore, for a fresh circuit $d_{cq} = d_{ccq}$, as illustrated by the green curves in Fig. 2.5. For an aged flip-flop, besides the degraded flip-flop characteristics, the slack also becomes smaller due to the aging of combinational logic gates, i.e., the arrival edge of data moves towards the clock capturing edge because of the degraded gate delays increase the critical path delay. Consequently, the $d_{cq}$ of the aged flip-flop increases. If the circuit ages severely, $d_{cq}$ increases significantly (see the purple curve in Fig. 2.5). In the extreme case, the flip-flop cannot capture the correct data (see the red curve in Fig. 2.5).

The lifetime of a digital system is extended by adding slack margins to compensate for the gradual delay degradation. The costs of the additional slack margins are lower speed (due to a longer clock period), increased power consumption (higher supply voltage value or using faster gates), and increased area (using faster cells which are typically larger). The effect of aging on $d_{cq}$ of a flip-flop is more significant than the aging of combinational logic paths because it degrades due to lower $d_{ss}$ and the degradation of $d_{ccq}$. The degradation of $d_{ccq}$ is naturally similar to the one for the combinational logic gates, while the lower $d_{ss}$ increases $d_{cq}$ due meta-stability. Hence, the delays of

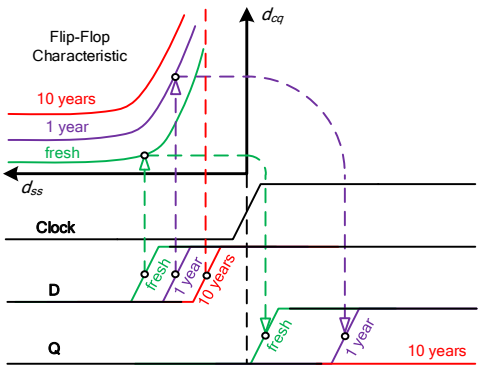



Figure 2.5: Effect of aging on the delay of a flip-flop.

the combinational logic paths as well as the clock distribution networks affect the $d_{cq}$ delays of flip-flops. The delay degradation of digital circuits due to silicon aging effects are thereby manifested in $d_{cq}$ of its flip-flops.

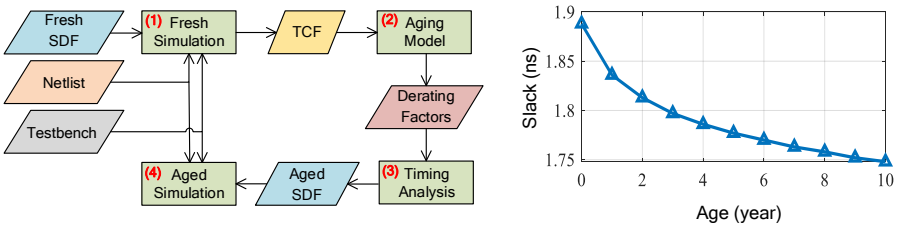
## 2.4 NOVEL MODELING OF CIRCUIT TIMING AND ITS UNRELIABILITY

In this section, two contributions of this thesis are explained. The first contribution is modeling the gradual silicon aging effects in the standard timing analysis and design tools. The second contribution is modeling the inter-dependency of setup/hold-time of flip-flops in a more efficient way.

### 2.4.1 *Novel Aging Simulation Framework*

A practical flow is proposed in this subsection to calculate the circuit timings in the standard timing analysis tools considering the aging effects. The proposed flow relates the aging stress factors to the slacks reported by the tools. It is also possible to run a functional simulation of the aged circuit with the corresponding timings, which are calculated with the proposed flow. The proposed flow is illustrated in Fig. 2.6 (a).

As mentioned before, the extent of gradual silicon aging effects (i.e., HCI and BTI) depend on the stress factors such as voltage, temperature, stress/recovery time. The stress/recovery time depends on the circuit workload because the data-induced toggling of the internal circuit nodes determine the time which each transistor is under stress. This information can be obtained by using the functional simulation-based activity of the circuit nodes. Conventionally, this activity information is used for power




(a) (b)

Figure 2.6: The aging simulation framework: (a) The flow of performing netlist simulation for the aged circuit. (b) The critical path slack in the typical corner considering aging for ARM Cortex M0 with 200MHz speed constraint.

calculations. To relate the node activity to the gate delay degradation, an aging model is required. We use the NBTI model from [24]. To calibrate the model, we assume that the threshold voltage of a transistor degrades by 10% after ten years of constant stress. With this model, the activity of circuit nodes is related to the threshold voltage shift of logic gates. The activities are obtained from simulating the circuit with fresh timings and they are stored *Toggle Count Format (TCF)* file, which is commonly used for power calculations. The ON current of transistor, $I_{on}$, can be modelled as a function of threshold voltage , $V_{th}$ [81]

$$I_{on} = \frac{W}{L_{eff}} P_C (V_{DD} - V_{th})^{\alpha}, \tag{2.6}$$

where $W$ and $L_{eff}$ are the width and effective channel length of transistor, $P_C$ and $\alpha$ are technology dependent constants, and $V_{DD}$ is the supply voltage value. Furthermore, the gate delay $d$ can be written as a function of the ON current of transistor [82]

$$d = k_0 \frac{C_{load} V_{DD}}{I_{on}}, \tag{2.7}$$

where $d$, $C_{load}$, and $k_0$ are the delay, load capacitance, and a constant for considering the non-modeled parameters, respectively. In this way, the aging-induced degradation of the individual gate delays can be calculated based on the activity of the circuit nodes. To apply this degradation to the timing analysis, delay derating factors are specified for each of the gate delays. Applying the derating factors scales the delays of the gates in the timing analysis (more information about derating factors can be found in [83]). The derating factors are set based on the obtained delay degradation factor for each gate from the aging model, e.g., if 5% delay degradation is calculated for a cell, the derating factor of 1.05X is applied for that cell. The timing slacks of the aged circuit are then

obtained with the timing analysis after applying the derating factors. In Fig. 2.6 (b) the minimum slack of an ARM Cortex M0 processor design is shown over ten years based on the proposed aging simulation framework. The processor is implemented using standard cell library in 40nm technology with 200MHz clock frequency, and includes ~8.5K gates. The results show that the longest critical path delay is degraded by ~4.5% after ten years. The *Standard Delay Format (SDF)* file can be dumped from the timing analysis tool after applying the aging-induced derated delays to performed netlist simulation of the aged circuit.

### 2.4.2 *Analytical Model for Inter-dependent Setup/Hold-Time of Flip-flops*

As discussed before, due to the joint effects of $d_{hs}$ and $d_{ss}$ on $d_{cq}$, the resulting $d_{st}$ and $d_{ht}$ are inter-dependent. This inter-dependency has been used to improve the accuracy of timing analysis. The initial work on this subject was done in [84] by Salman et al. In [75], the method of [84] is used to reduce the uncertainty in the critical path (that determines the setup time requirement) and the shortest path (that determines the hold time requirement) of a 90nm design by 100% and 50%, respectively. The authors in [85] propose an iterative timing analysis framework that considers the inter-dependency between the setup/hold-time and could reduce the clock period by up to 4.9% for a 90nm circuit. The work of [85] is improved in [76] by performing a path-based timing analysis and linear programming based optimization to find the best combination of flip-flop timing specifications in the circuit. Consequently, the critical path slack is increased by up to 130ps for a variety of 65nm test circuits. The authors in [86] propose a method to optimize clock skew using setup/hold-time inter-dependency, shortening the clock periods of various circuits by 4.2% on average. For all of these methods, a relationship between the setup time and hold time of each flip-flop is required. Obtaining this relationship is computationally an expensive task. There are efforts to characterize the inter-dependency between $d_{st}$ and $d_{ht}$ efficiently. The characterization methods in [75] and [85] are based on generating the $d_{cq}$ surface as a function of $d_{ss}$ and $d_{hs}$. The setup/hold-time inter-dependency curve is then approximated considering the intersection of $d_{cq}$ surface and the constant $d_{pcq}$ constraint. In [85] the relationship between setup/hold-time is obtained from an empirical model for $d_{cq}$ as a function of $d_{ss}$ and $d_{hs}$. However, it is observed in [85] that obtaining the parameters of the model is computationally expensive. Instead of generating the setup/hold-time inter-dependency curve from the $d_{cq}$ surface, which is a time-consuming brute-force method, the curve is characterized directly in [74] and [87]. The main focus of those works is on accelerating the characterization of the inter-dependency curve. In [74], the characterization is accelerated by increasing the convergence rate in a sufficiently narrow interval. In [87], the effort is limiting the search range based on a heuristic slope estimation for the seeking curve. Then, with having some points on the setup/hold-time inter-dependency curve, the piece-wise linear approximation is used to compose the curve. The accu-

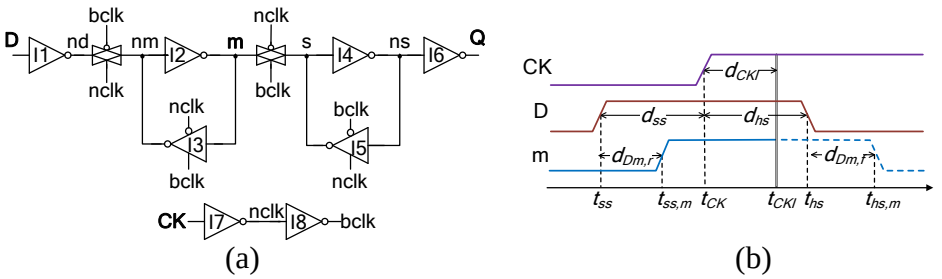



Figure 2.7: A master-slave flip-flop. (a) The circuit schematic. (b) The waveforms at the input D and the middle point m, obtained based on the shortest and longest paths to capture a rising input.

racy of these characterization methods increases by identifying more setup/hold-time points which increases the characterization time. Despite all these efforts, an analytical model is required to increase the accuracy and computation efficiency of setup/hold-time inter-dependency based timing analysis frameworks [76]. Therefore, we proposed an analytical model with higher accuracy and shorter characterization time compared to the existing piece-wise linear approximations.

#### 2.4.2.1 *The Proposed Model*

In this subsection, the proposed model for setup/hold-time inter-dependency is introduced. The model equations are derived based on circuit level parameters, specifically the voltage of an internal node in the flip-flop.

The most commonly used master-slave flip-flop is shown in Fig. 2.7 (a). Consider the clock-to-Q path of the flip-flop,

$$d_{cq} = d_{CKI} + d_{mQ}, \tag{2.8}$$

where $d_{CKI}$ is the delay from the clock capturing edge until when the slave latch becomes transparent. $d_{mQ}$ is the delay of the slave latch from the start of transparency to the switching of the flip-flop output. In (2.8), $d_{CKI}$ does not depend on the arrival time of the input data. On the other hand, a small setup skew $d_{ss}$ and/or hold skew $d_{hs}$ lowers down the effective voltage driving the slave latch at the onset of transparency ($V_{m,eff}$), thereby increasing $d_{mQ}$. By using the $I_{on}$ in (2.6) and the delay model in (2.7),

$$d_{mQ} \approx kV_{DD}/(V_{m,eff} - V_{th,eff})^{\alpha}, \tag{2.9}$$

where $k$ is a technology and sizing dependent parameter, and $V_{th,eff}$ is the effective threshold voltage of the corresponding transistors of the gates. If $d_{hs}$ and $d_{ss}$ are sufficiently large, $V_m$ has settled to 0 or $V_{DD}$ at the onset of the slave latch transparency. In this case, $d_{cq}$ is minimized by being driven with the maximum effort, i.e., $V_{m,eff}$ is maximized. As

mentioned before, $d_{cq}$ increases when the flip-flop enters into meta-stability due to a small $d_{ss}$ and/or $d_{hs}$. According to (2.8) and (2.9) the lower bound for $V_{m,eff}$ is therefore

$$V_{m,eff,L} = V_{th,eff} + (kV_{DD}/(d_{pcq} - d_{CKI}))^{1/\alpha}. \tag{2.10}$$

The inter-dependent setup/hold-time values are $d_{ss}$ and $d_{hs}$ points that make $d_{cq} = d_{pcq}$, and thereby $V_{m,eff} = V_{m,eff,L}$. Therefore, the setup/hold-time inter-dependency curve can be modeled by having $V_{m,eff}$ as a function of $d_{ss}$ and $d_{hs}$. Without loss of generality, consider the rise capturing waveforms shown in Fig. 2.7 (b). As discussed before, due to the delay requirements for the shortest and the longest paths in the combinational logic gates (see Fig. 2.7 (a)), the capturing data is a pulse at the input of the flip-flop (see Fig. 2.7 (b)). An approximation for $V_{m,eff}$ is $V_m$ at time $t_{CKI} = t_{CK} + d_{CKI}$ where $t_{CK}$ is the capturing edge of the clock signal. $V_m$ at $t = t_{CKI}$ determines the effective voltage that drives the slave latch, i.e. $V_{m,eff}$. A reshaped version of the input data pulse (i.e., with changed pulse width and slopes) appears at m if the master latch is transparent to the input. Although $V_m$ settles to $V_{DD}$ or 0 after $t = t_{CKI}$, since $t = t_{CKI}$ is the end of transparency of the master latch, it is assumed that the reshaped version of the input pulse is observable at node m, while this is not the case for $t > t_{CKI}$ as shown in Fig. 2.7 (b) with dashed lines. A pulse can be modeled as the sum of two exponential terms where its rise and fall times depend on the corresponding time constants. Without loss of generality, in a rise capturing scenario, $V_m$ is modeled for $V_m \geq V_{DD}/2$ as

$$V_m = V_{DD} - (\frac{V_{DD}}{2})[\exp\frac{t_{ss,m} - t}{\tau_r} + \exp\frac{t_{hs,m} - t}{\tau_f}] \quad \text{for} \quad t \leq t_{CKI}, \tag{2.11}$$

where

$$\begin{aligned} t_{CKI} &= t_{CK} + d_{CKI} \\ t_{ss,m} &= t_{ss} + d_{Dm,r} \\ t_{hs,m} &= t_{hs} + d_{Dm,f} \end{aligned} \tag{2.12}$$

In the above equations, $t_{ss} = t_{CK} - d_{ss}$ and $t_{hs} = t_{CK} + d_{hs}$ as shown in Fig 2.7 (b). Furthermore, $d_{Dm,r}$ and $d_{Dm,f}$ are the rising and falling delays from D to m, respectively. Moreover, $\tau_r$ and $\tau_f$ are the rising and falling time constants, respectively. According to (2.11) and (2.12),

$$\begin{aligned} &V_{m,eff} = V_m(t = t_{CKI}) = \\ &V_{DD} - (\frac{V_{DD}}{2})[\exp\frac{d_{ss0} - d_{ss}}{\tau_r} + \exp\frac{d_{hs0} - d_{hs}}{\tau_f}] \\ &\quad \text{for} \quad d_{ss0} < d_{ss} \quad \text{and} \quad d_{hs0} < d_{hs}. \end{aligned} \tag{2.13}$$

According to (2.12),

$$\begin{aligned} d_{ss0} &= t_{ss,m} - t_{CKI} + d_{ss} = d_{Dm,r} - d_{CKI}, \\ d_{hs0} &= t_{CKI} - t_{hs,m} + d_{hs} = d_{CKI} - d_{Dm,f}. \end{aligned} \tag{2.14}$$

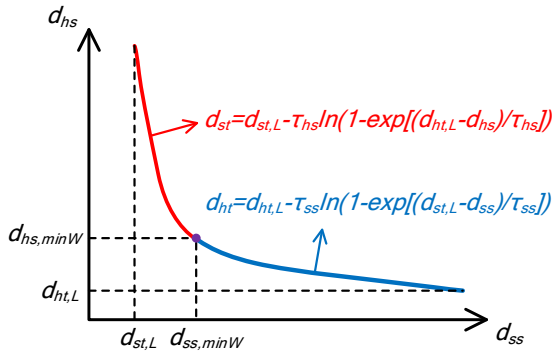



Figure 2.8: Illustration of the proposed setup/hold-time interdependency model and its parameters.

Note that $d_{ss0}$ ($d_{hs0}$) is the lower bound of $d_{st}$ ($d_{ht}$) for $V_{m,eff} \geq V_{DD}/2$, assuming $d_{hs}$ ($d_{ss}$) is sufficiently large. The model in (2.13) expresses $V_{m,eff}$ as a function of the input skews $d_{ss}$ and $d_{hs}$. Given the lower bound of $V_{m,eff}$ in (2.10), the setup/hold-time inter-dependency is obtained from (2.13) as

$$\begin{aligned} d_{ht} &= d_{ht,L} - \tau_{hs} \ln(1 - \exp \frac{d_{st,L} - d_{ss}}{\tau_{ss}}) \quad \text{for} \quad d_{ss} > d_{st,minW}, \\ d_{st} &= d_{st,L} - \tau_{ss} \ln(1 - \exp \frac{d_{ht,L} - d_{hs}}{\tau_{hs}}) \quad \text{for} \quad d_{hs} > d_{ht,minW}, \end{aligned} \tag{2.15}$$

where $\tau_{ss}$ and $\tau_{hs}$ are the setup and hold time constants, respectively. The parameters $d_{st,L}$ and $d_{ht,L}$ in (2.13) are the independently characterized $d_{st}$ and $d_{ht}$ when $d_{hs}$ and $d_{ss}$ are sufficiently large, respectively. The model equations are illustrated in Fig. 2.8. $d_{st,minW}$ and $d_{ht,minW}$ are the corresponding input skews for the minimum input capturing pulse width based on (2.3) and (2.15)

$$\begin{aligned} \frac{\partial W_D}{d_{ss}} = 0 \Rightarrow d_{st,minW} = d_{st,L} + \tau_{ss} \ln(1 + \frac{\tau_{hs}}{\tau_{ss}}), \\ \frac{\partial W_D}{d_{hs}} = 0 \Rightarrow d_{ht,minW} = d_{ht,L} + \tau_{hs} \ln(1 + \frac{\tau_{ss}}{\tau_{hs}}). \end{aligned} \tag{2.16}$$

Note that due to the form of the equations, the lower bound of the skews in the model is chosen to be the corresponding input skews for the minimum capturing input pulse width [84].

#### 2.4.2.2 *Model Characterization Flow*

In this section, a flow is introduced to find the parameters of the model in (2.15)

- $d_{st,L}$: independently characterized $d_{st}$ at large $d_{hs}$,

- $d_{ht,L}$: independently characterized $d_{ht}$ at large $d_{ss}$,
- $\tau_{ss}$: setup time constant, and
- $\tau_{hs}$: hold time constant.

The proposed flow is shown in Algorithm 1. Furthermore, the main steps of the flow are illustrated in Fig. 2.9.

**Algorithm 1** The flow of extracting the model parameters.

**procedure** MODEL_PARAMETERS_EXTRACTION
  $d_{st,L} \leftarrow$ characterize $d_{st}$ at large $d_{hs}$
  $d_{ht,L} \leftarrow$ characterize $d_{ht}$ at large $d_{ss}$
  $\tau_{ss,L}$, $\tau_{hs,L} \leftarrow$ result of initial simulation
  $D_{st}[1] = d_{st,L} + \tau_{ss,L} \ln(1 + \frac{\tau_{hs,L}}{\tau_{ss,L}})$
  $D_{ht}[1] \leftarrow$ characterize $d_{ht}$ at $d_{ss} = D_{st}[1]$
  $D_{ht}[2] = d_{ht,L} + \tau_{hs,L} \ln(1 + \frac{\tau_{ss,L}}{\tau_{hs,L}})$
  $D_{st}[2] \leftarrow$ characterize $d_{st}$ at $d_{hs} = D_{ht}[2]$
  $D_{st}[3] = d_{st,L} + 2 \times \tau_{ss,L} \ln(1 + \frac{\tau_{hs,L}}{\tau_{ss,L}})$
  $D_{ht}[3] \leftarrow$ characterize $d_{ht}$ at $d_{ss} = D_{st}[3]$
  $D_{ht}[4] = d_{ht,L} + 2 \times \tau_{hs,L} \ln(1 + \frac{\tau_{ss,L}}{\tau_{hs,L}})$
  $D_{st}[4] \leftarrow$ characterize $d_{st}$ at $d_{hs} = D_{ht}[4]$
  $DS_{st} =$ sort $D_{st}$ from low to high
  $DS_{ht} =$ sort $D_{ht}$ from low to high
  $R = (DS_{ht}[1] - DS_{ht}[4])/(DS_{st}[1] - DS_{st}[4])$
  $\tau_{ss} = \tau_{ss,L}$
  $e_{tau} = -1$
  **while** $e_{tau} \neq 0$ **do**
    $\tau_{ss,prev} = \tau_{ss}$
    $\tau_{hs} = R \times \tau_{ss}$
    $C = 1 + \exp((DS_{ht}[1] - DS_{ht}[4])/\tau_{hs})$
    $\tau_{ss} = \frac{DS_{st}[1] - DS_{st}[2]}{\ln(C - \exp((DS_{ht}[1] - DS_{ht}[3])/\tau_{hs}))}$
    $\tau_{hs} = R \times \tau_{ss}$
    $C = 1 + \exp((DS_{ht}[1] - DS_{ht}[4])/\tau_{hs})$
    $\tau_{ss} = \frac{DS_{st}[1] - DS_{st}[3]}{\ln(C - \exp((DS_{ht}[1] - DS_{ht}[2])/\tau_{hs}))}$
    $e_{tau} = \tau_{ss} - \tau_{ss,prev}$
  **end while**
**return** $d_{st,L}$, $d_{ht,L}$, $\tau_{ss}$, $\tau_{hs}$
**end procedure**

The parameters $d_{st,L}$ and $d_{ht,L}$ are characterized by using an independent characterization method. For example, $d_{st,L}$ is characterized using binary search to find $d_{ss}$ that makes $d_{cq} = d_{pcq}$, assuming large $d_{hs}$ (e.g., half of the clock period). Two other model

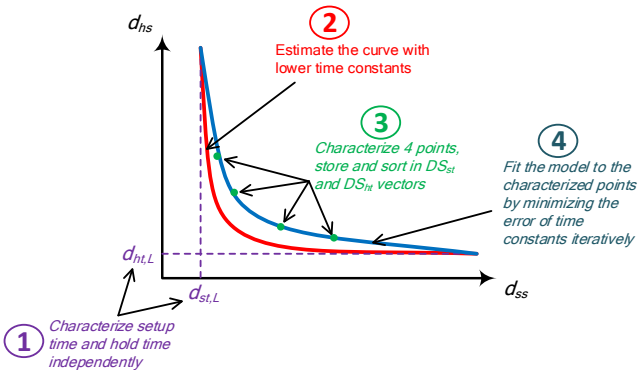



Figure 2.9: Illustration of steps to find the parameters of the proposed model for setup/hold-time interdependency.

parameters to be characterized are the time constants $\tau_{ss}$ and $\tau_{hs}$. The lower boundaries of the time constants $\tau_{ss,L}$ and $\tau_{hs,L}$ are found by performing an initial transistor level simulation. In the initial simulation, a sufficiently wide pulse (e.g., a third of the clock period) is applied to the input of the flip-flop during the low phase of the clock signal. The slopes of the transferred pulse to node m are then calculated to find the initial values for the time constants $\tau_{ss,L} = \frac{tr_{ss}}{\ln 5}$ and $\tau_{hs,L} = \frac{tr_{hs}}{\ln 5}$, where $tr_{ss}$ and $tr_{hs}$ are the corresponding transition time from 50% to 90% of the voltage change at node m. Since the effect of switch current is not considered in this simulation, the calculated time constant is lower than the actual one in $V_{m,eff}$. Therefore, the initial values obtained for the time constants are lower than the actual ones. Employing the lower than actual time constants in (2.13) results in a curve that is steeper than the actual curve (see Fig. 2.9). To find more accurate estimations for $\tau_{ss}$ and $\tau_{hs}$, the inverse of (2.13) are used for the points near the $d_{ss}$ and $d_{hs}$ point corresponding to the minimum capturing input pulse width.

Based on the estimation of the curve with lower time constants, the following equations are used to find the skews for characterizing 4 setup/hold-time points on the actual curve

$$d_{ss,i} = d_{st,L} + i \times \tau_{ss,L} \ln(1 + \frac{\tau_{hs,L}}{\tau_{ss,L}}), \tag{2.17}$$

$$d_{hs,i} = d_{ht,L} + i \times \tau_{hs,L} \ln(1 + \frac{\tau_{ss,L}}{\tau_{hs,L}}), \tag{2.18}$$

where $i$ is selected to be 1 or 2. The skews obtained at $i = 1$ and $i = 2$ in (2.18) and the corresponding setup/hold-time are used as four points which are close to the the minimum capturing input pulse width $d_{ss}$ and $d_{hs}$ point. The obtained points are illustrated in Fig. 2.9. The extracted points are then sorted in $DS_{ht}$ and $DS_{st}$ to find the bounding

points, i.e., the points indexed with 1 and 4. The remaining points (i.e., indexed with 2 and 3) are the middle points. Note that since the setup/hold-time inter-dependency is a monotonically decreasing function, the $i^{th}$ element in $DS_{st}$ is the pair of the element number $(4-i)$ in $DS_{ht}$ on the curve. According to (2.13), the inter-dependency curve that crosses the boundary points is

$$\begin{aligned} d_{st} =& DS_{st}[1] - \tau_{ss} \ln(C - \exp \frac{DS_{ht}[1] - d_{hs}}{\tau_{hs}}), \\ C =& 1 + \exp((DS_{ht}[1] - DS_{ht}[4])/\tau_{hs}), \\ \tau_{hs} =& \tau_{ss}(DS_{ht}[1] - DS_{ht}[4])/(DS_{st}[1] - DS_{st}[4]). \end{aligned} \tag{2.19}$$

Finally, the middle points are employed to find $\tau_{hs}$ and $\tau_{ss}$ iteratively, as shown by the while loop in Algorithm 1.

#### 2.4.2.3 *Using the Proposed Model in The Standard Timing Analysis Flow*

Transistor-level simulations (a.k.a., SPICE level simulations) are required to characterize the setup/hold-time of flip-flops. The industrial characterization tools automate and optimize the required simulations for these characterizations. While the proposed model characterization flow relies on transistor level simulations, it is demanding to use the existing standard characterization tools to characterize the model parameters. If the parameters of the model are obtained with the standard library characterization tools, the usage of model can be integrated into the standard timing sign-off flow. The standard characterization tools can characterize setup (hold) time at a specific hold (setup) skew. Therefore, we can obtain some points on the inter-dependency curve using this feature of the tools. By default, the characterization tools give $d_{ht,L}$ and $d_{st,L}$ in (2.15) (i.e., assuming infinite counterpart skew). To get the other two parameters of the model in (2.15) (i.e. $\tau_{ss}$ and $\tau_{hs}$), the following values are also employed

$$\begin{aligned} &d_{stH} = d_{st} \text{ when } d_{hs} = d_{ht,L}, \\ &d_{htH} = d_{ht} \text{ when } d_{ss} = d_{st,L}. \end{aligned} \tag{2.20}$$

The obtained points are therefore $(d_{st,L}, d_{htH})$ and $(d_{stH}, d_{ht,L})$. These points on the inter-dependency curve are located where the exponential term in (2.15) becomes negligible. Since $exp(-n) < 0.01$ when $n > 5$, we assume the following values for the time constants

$$\begin{aligned} \tau_{ss} &= (d_{stH} - d_{st,L})/5, \\ \tau_{hs} &= (d_{htH} - d_{ht,L})/5 \end{aligned} \tag{2.21}$$

Therefore, all of the four parameters of the model in (2.15), i.e., $d_{st,L}$, $d_{ht,L}$, $\tau_{ss}$, and $\tau_{hs}$, can be obtained by using the standard characterization tools and (2.21). In Fig. 2.10, the approximated setup/hold-inter-dependency curve is shown for a flip-flop of an industrial standard cell library in 40nm technology.

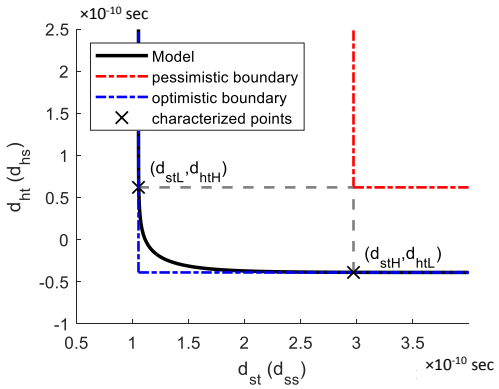



Figure 2.10: The setup/hold-time inter-dependency curve obtained with the model and a standard characterization tool for a flip-flop from an industrial 40nm library.

The output of the characterization tool is a file with *Liberty (LIB)* format which contains the delays (also power and area) of the gates. The LIB file contains the setup/hold-times of the flip-flops and the rest of the gate delays. We defined the optimistic (pessimistic) library as the library which includes $d_{ht} = d_{ht,L}$ ($d_{ht} = d_{htH}$) and $d_{st} = d_{st,L}$ ($d_{st} = d_{stH}$) values. In Fig. 2.11 the standard flow and the proposed flow for more accurate timing analysis are shown. In the standard flow, the analysis is done with the optimistic LIB file. The reported slacks with the standard flow are therefore optimistic. To use the inter-dependency curves for more accurate timing analysis, the proposed flow that is shown in Fig. 2.11 is followed. In the proposed flow, besides the optimistic library, the pessimistic library is also employed to have both pessimistic and optimistic slacks and skews. Then (2.15) and (2.21) are employed to find the actual slack at all flip-flops.

To use the proposed model for more accurate timing analysis, first we should find out if there are some flip-flops in the design which both of their setup and hold slacks are low, and mark them as the optimistic flip-flops. In fact, due to having concurrent low setup and hold slacks, taking the setup/hold-time inter-dependency into account is necessary for those flip-flops. That is because the actual slacks can be lower than the reported ones due to the low counterpart skews. For instance, the actual setup slack at a flip-flop can be lower than the reported one because the hold slack is also low at the same flip-flop and this makes the actual setup time higher than the one which is used for reporting the setup slack. Therefore, the setup/hold-times of the optimistic flip-flops must be updated to take the effect of counterpart low skew into account. To identify the flip-flops with concurrent setup/hold-skews, the pessimistic LIB file is also used to perform timing analysis. Assuming the slacks of all flip-flops are positive with using the optimistic LIB file (as the starting point of the analysis), if both hold and setup

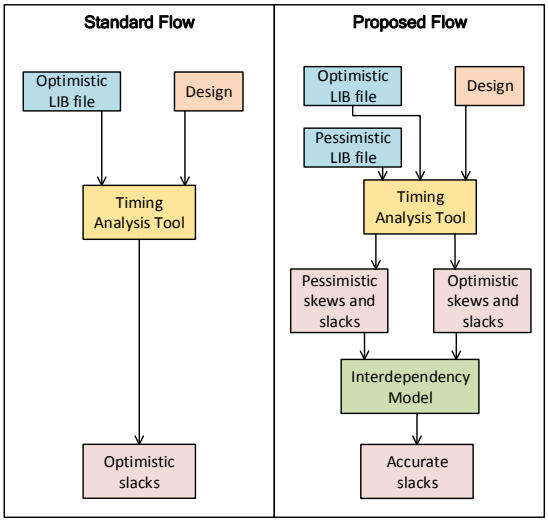



Figure 2.11: The standard timing analysis flow and the proposed timing analysis flow which also considers the interdependency of setup/hold-time.

slacks are negative with using the pessimistic LIB file, taking the inter-dependency curve into account is necessary for that flip-flop. Once those flip-flops and, the corresponding setup/hold-times, and setup/hold slacks, with both pessimistic and optimistic LIB files, are obtained, the model equations in (2.15) together with (2.21) are employed to calculate the actual slacks.

## 2.5 EXPERIMENTAL RESULTS FOR THE PROPOSED IDEAS

In this section, the proposed ideas in this chapter for modeling circuit timing and its unreliability are evaluated.

### 2.5.1 *Model Evaluation*

In this section, the accuracy and computation efficiency of the proposed model for setup/hold-inter-dependency are compared to the state of the art. The results in this section are obtained by performing SPICE simulations using Cadence Spectre with standard threshold voltage transistors of an industrial 40nm CMOS technology. In the conventional standard cell libraries, the setup/hold-time of flip-flops are characterized for different combinations of input slopes, i.e., the slopes of the clock input (CK) and the

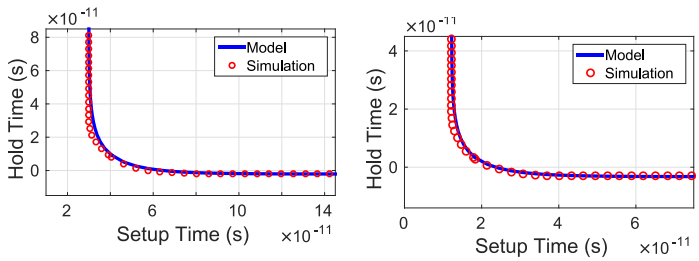



Figure 2.12: The setup/hold-time inter-dependency curve for data and clock input slopes of 50ps at (a) slow corner (Process:SS, $V_{DD} = 0.99V$, T=125°$C$) and (b) fast corner (Process:FF, $V_{DD} = 1.21V$, T=0°$C$).

Table 2.1: The maximum error of model with regard to the actual curve characterized with extensive SPICE simulations at different corners relative to input slope for input slopes ranging from 10ps to 100ps.

| Process | $V_{DD}$ | T | Setup Time (%) | Hold Time (%) |
|---|---|---|---|---|
| FF | 1.21 $V$ | $-40°C$ | 6.68 | 3.68 |
| FF | 1.21 $V$ | $0°C$ | 6.44 | 3.45 |
| FF | 1.21 $V$ | $125°C$ | 5.12 | 2.45 |
| SS | 0.99 $V$ | $-40°C$ | 7.18 | 6.06 |
| SS | 0.99 $V$ | $0°C$ | 7.48 | 5.88 |
| SS | 0.99 $V$ | $125°C$ | 7.44 | 5.28 |

data input (D). The setup/hold-time inter-dependency curves for the slow and fast corners according to timing library and with the average input slope (50ps) according to the timing report of an industrial design are illustrated in Fig. 2.12.

To assess the accuracy of the proposed model, the relative value of the maximum error (i.e., pessimism/optimism) with the proposed model at different corners of the timing library for different input slope combinations in the range of 10$ps$ to 100$ps$ are shown in Table 2.1 for hold time and setup time. As shown in Table 2.1, the maximum pessimism/optimism of the model with regard to SPICE simulation is less than 10% of input slope in the evaluated range of the slopes.

The model parameters are fitted by following the flow that is provided in Algorithm 1. The main computational complexity of the model is the four independent characterizations at specific setup/hold-skews. To compare the effectiveness of the proposed model with other methods, the number of required setup/hold-time points and the absolute pessimism/optimism is compared to the state of the art [74, 75, 87]. The accuracy of

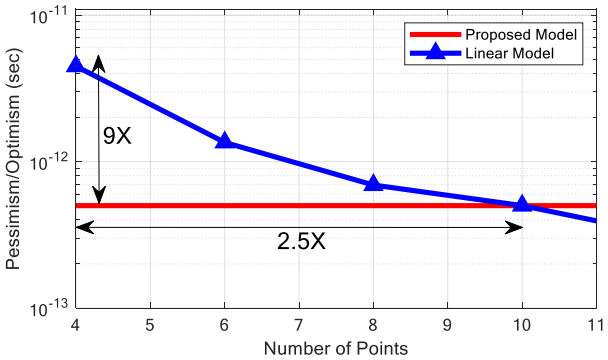



Figure 2.13: The pessimism/optimism of the model compared to the linear approximation. With four characterization points, the proposed method has the same pessimism/optimism that the linear approximation has with 10 characterization points (2.5X more computation efficiency).

the piece-wise linear approximation used in these methods increases when the number of points on the actual curve increases. The pessimism/optimism of the proposed model compared to the piece-wise linear approximation is illustrated in Fig. 2.13 for input slopes equal to 50*ps* (i.e., in the middle of the slope range). As discussed before, in the model characterization flow, four points of setup/hold-time on the inter-dependency curve are identified in addition to the traditional independent setup time and hold time. Other methods also require to have the traditional setup time and hold time to find other points in the setup/hold-time inter-dependency curve. Therefore, the additional setup/hold-time points are considered as the computation overhead to characterize the setup/hold-time inter-dependency curve. According to the results illustrated in Fig. 2.13, to achieve the same accuracy as the proposed model, the piece-wise linear approximation requires ten inter-dependent setup/hold-time points. Therefore, the proposed model is 2.5X more efficient than the piece-wise linear approximation based methods. Furthermore, with the same number of setup-hold points (i.e., four points), the proposed model is 10X more accurate compared to the piece-wise linear approximation based methods.

### 2.5.2 *Framework Evaluation*

Taking the setup/hold-time inter-dependency into account is only necessary if there are some flip-flops in the design for which the setup and hold slack are low concurrently. This case can happen for complex designs with multiple clock and power domains. For instance, in such a complex design, there might be flip-flops at the boundary of the domains with a critical short path from one domain and a critical long path from another

domain. On the other hand, in small designs with balanced pipeline stages, each flip-flop is typically at the endpoint of either a critical long path or a critical short path. Therefore, for the flip-flops of such a design, the setup and hold skew are typically not low concurrently.

The flow has been evaluated using a single-core ARM Cortex M0 based microprocessor platform. The platform contains components such as the processor, timers, AXI bus, and boot-loader. The size of the circuit is ~22K cells and there are ~3.3K flip-flops in the design. This platform has only one power domain and has a global clock. Nevertheless, there are still some critical flip-flops for which considering the inter-dependency of setup/hold-time is necessary. After going through synthesis and back-end flow with 3.5ns clock period as the speed constraint for the design, the optimistic LIB file and the pessimistic LIB file are used with the proposed flow (shown in Fig. 2.11) to identify flip-flops which setup/hold-time inter-dependency must be applied. As mentioned before, the setup/hold-time of flip-flops are characterized independently in the conventional IC design flow. Therefore, the optimistic LIB file is used in the standard timing analysis, thereby the reported slacks can be larger than the actual values. By applying the proposed, nine critical flip-flops were identified, and out of them, one flip-flop is marked to be failing according to the proposed model. In Fig. 2.14, the results of this analysis are shown. Instead of applying the proposed flow, one can also use the pessimistic libraries to make sure that the slack margins are safe. However, by using the pessimistic LIB files, the design costs increase. Based on our experiments, changing the libraries to the pessimistic ones increase the power consumption by 1.26%. Therefore, the proposed flow can reduce the cost of timing reliability by making the timing analysis more accurate.

## 2.6 SUMMARY

In this chapter, the concepts related to the timing of digital circuits are reviewed. New models are developed for circuit timing and its unreliability problems. A new flow is proposed to capture the gradual timing degradation effects in the standard timing analysis tools. To increase the accuracy of timing analysis, an analytical model for setup/hold-time inter-dependency is also proposed. The proposed model is based on circuit level parameters for conventional master-slave flip-flops. The accuracy of the model is ~10X higher than the existing piece-wise linear approximation based methods. The proposed model characterizes setup/hold-time inter-dependency with ~2.5X higher computation efficiency compared to the existing piece-wise linear approximation based methods. Furthermore, a flow is proposed to employ the model in the standard timing analysis tools.

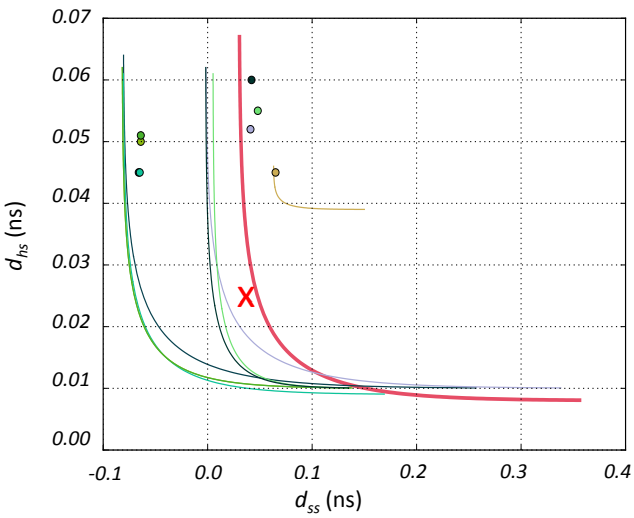



Figure 2.14: The setup/hold-time inter-dependency curve and the setup/hold-skew point for nine critical flip-flops (distinguished by color) for which the inter-dependency should be taken into account. One flip-flop (shown by the cross and red color) is marked to be failing according to the analysis.

*"Study the past if you would define the future"*

— Confucius, 551–479 BC

# 3

# IN-SITU DELAY MONITORING ALONG TIMING PATHS

## 3.1 INTRODUCTION

In-situ delay monitoring is an advanced technique to monitor the timing reliability of digital circuits. This technique is truly in-system, it can run in parallel to the primary workload, and it can predict timing errors. However, the main challenges for this technique are the prohibitive design overheads, poor integrability in the design flow, and low coverage of timing critical components. Conventionally, in-situ delay monitors are inserted at the endpoints of timing paths. In sharp contrast to this approach, a low-overhead technique is proposed in this chapter where the insertion points are selected along the timing paths. With the proposed approach, the cost per monitor and the number of required monitors are reduced while the coverage of timing critical components increases. Furthermore, a design integration flow is introduced to add monitors to the design effectively.

In this chapter, first, the state of the art of in-situ delay monitoring are reviewed in Section 3.2. Then, the design of the proposed in-situ monitor and its insertion flow are explained in Section 3.3. In Section 3.4, the trade-off for selecting the monitor insertion points are discussed. The experimental results of employing the technique are provided in Section 3.5. Finally, Section 3.6 summarizes this chapter.

## 3.2 PRELIMINARIES AND RELATED WORKS

In synchronous digital circuits, data propagates from primary inputs or sequential elements, through combinational logic, to another or the same sequential gate or the primary outputs of the circuit. Considering the data which propagates within the circuit (i.e., between flip-flops), the timing path starts from the clock pin of the launching flip-flop, continues through the timing arcs of some logic gates, and ends into the data pin of a capturing flip-flop. The path delay is the sum of the propagation delay of each timing arc in the path. Within one clock period, the maximum data arrival time to a circuit node is the maximum delay from the clock pin of a launching flip-flop to the node.

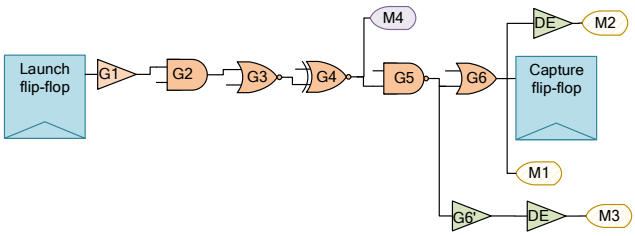



Figure 3.1: The insertion points of monitors along the timing paths of digital circuits for different in-situ delay monitoring techniques.

The maximum delay of all the timing paths that end at a capture flip-flop is the longest path. The maximum data arrival time to the data input pin of a flip-flop is equal to the delay of longest path that ends at the flip-flop. Furthermore, the maximum delay of each timing path should not be more than one clock period, $t_{clock}$. Otherwise, erroneous data may be captured, i.e., a timing error may happen. Timing analysis is performed to find the propagation delay of the paths and to make sure that a timing violation does not happen. The delay of all paths could be calculated in a path-based timing analysis. Alternatively, in block-based timing analysis, the maximum data arrival time is calculated from summation and maximum operations to find the maximum delay of all paths ending in each flip-flop. The block-based timing analysis is a practical method because its run-time is much less than the path-based approach [88].

In Fig. 3.1, an example of a timing path, and the possible insertion points of the in-situ monitors are illustrated. The timing path starts from the launching flip-flop, goes through the combinational logic gates G1 to G6, and ends into the capturing flip-flop. Conventionally, in-situ delay monitors are inserted at the end-point of the timing paths [66, 69] (see M1 in Fig. 3.1). Hence, a timing error is detected by sensing if the data arrival is too late, i.e., a setup time violation occurs at the capture flip-flop. To avoid flip-flop meta-stability and to avoid the complexity and overhead of error correction [89], the monitors detect timing degradation before the actual timing error in the main flip-flops. We call this guard banding between the monitors and the main flip-flops. The guard banding is performed by adding a delay margin between the insertion point and the input of the monitor (see DE before M2 in Fig. 3.1). In fact, DE makes the slack of M2 be less than the slack of the capturing flip-flop, which means that there is a guard band between the monitor triggering and the timing failure in the main design. Note that inserting the monitors at every flip-flop adds a considerable design cost. Observe as well that the detection of timing degradation depends on the excitation of the monitored paths, i.e., the paths which end at the monitors. The monitored paths are not always excited since path excitation is data dependent. Therefore, in-situ monitoring may have limited observability to variability effects. In [72], the monitors are inserted at interme-

diate points to reduce the number of monitors. The unmonitored parts of the paths are accounted for by adding a pessimistic delay margin between the insertion point and the input of the monitor (see G6' in Fig. 3.1). Note that the guard band is still applied by adding another delay margin before the monitor (see DE before M3 in Fig. 3.1). This approach adds to the cost of each inserted monitor. To reduce the design overhead per monitor, the monitor is therefore preferably inserted almost at the end of the paths to reduce the additional gates per monitor. An inspection window is also required (e.g., by sampling the data multiple times) in such methods to define the time at which any transition should be flagged as timing error. However, this also adds to the design cost of each monitor. Besides, inserting the monitors almost at the end of the path reduces the opportunities to prevent the error within one cycle, and adds to the number of required monitors. However, instead of additional gates for a proper guard banding, the insertion point of the monitor can be selected such that the slack of the monitor is made less than the slack of the main design without additional gates (see M4 in Fig. 3.1).

In [90] and [91], the temporal middle point of the critical path is monitored to check if a delay increase to the insertion point causes a transition after half of the clock period. Observe that in the presence of variability effects, the critical path ranking may change. Therefore, the candidate critical timing paths are those paths whose delay is longer than a specific value. Note that the insertion points must be identified based on the timing report of all paths. Path-based timing analysis is required to obtain a list with all critical paths. However, the path-based timing analysis is not feasible for medium to large size circuits due to the complexity of the analysis [92] making the methods of [90] and [91] impractical. Furthermore, in [90] and [91] no delay margin is added for guard banding because it is assumed that the delay of monitored and unmonitored parts of the paths degrade in a similar fashion.

In this chapter, a new in-situ chip-health monitoring technique is proposed with a reduced number of monitors compared to endpoint monitoring and improved observability to delay degradation through selective insertion at intermediate points along timing paths. Implicit guard banding is added by inserting the monitors at particular points of the critical paths and reduce design cost per monitor by removing additional delay margins (see M4 in Fig. 3.1). Furthermore, a novel technique is proposed, which is based on graph-based static timing analysis to make sure that all critical paths are monitored while the drawbacks of path-based static timing analysis are avoided. In order to reduce design intrusion, a new implementation flow is proposed that does not disturb the timing and layout of the main design during monitor insertion.

## 3.3 IN-SITU DELAY MONITORING WITHIN TIMING PATHS

In this section, the design of the in-situ monitor and our method to identify the set of insertion points are described. Furthermore, a new design flow is explained in which the proposed technique is integrated into it.

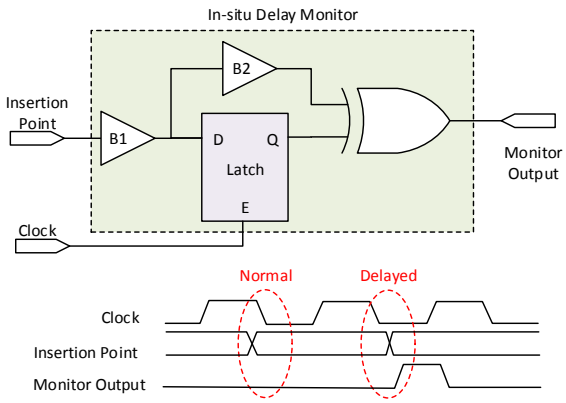



Figure 3.2: The design of in-situ monitor.

### 3.3.1 *The Design In-Situ Monitor*

The monitor detects delay degradation by checking if any late data transition occurs during the second half of the clock period. Therefore, the insertion points should be selected such that the maximum data arrival to the point is less than half of the clock period, in the absence of variation effects. Since the monitors are intended to detect the delay degradation, the insertion points should be selected such that in the presence of variation effects, the maximum data arrival to the point is more than half of the clock period. This late data arrival causes a transition in the second half of the clock period which results in monitor excitation.

The design of the in-situ delay monitor is shown in Fig. 3.2. The monitor consists of one latch which is transparent when the clock input is zero, one XOR gate, and two buffers (B1 and B2 in Fig. 3.2). The latch captures the data value of the insertion point at the negative clock edge. The XOR gate then flags any transition which occurs during the second half of the clock cycle. Delayed arrival of data to the insertion point implies delay degradation, and the monitor detects it. Buffer B1 minimizes the loading effect of the monitor on the insertion point. Buffer B2 is selected such that the inputs of the XOR gate arrive simultaneously to avoid glitches (and reduce power consumption). Note that B1 and B2 are small cells while the delay elements that are used for other techniques as shown in Fig. 3.1 are large and expensive in terms of power.

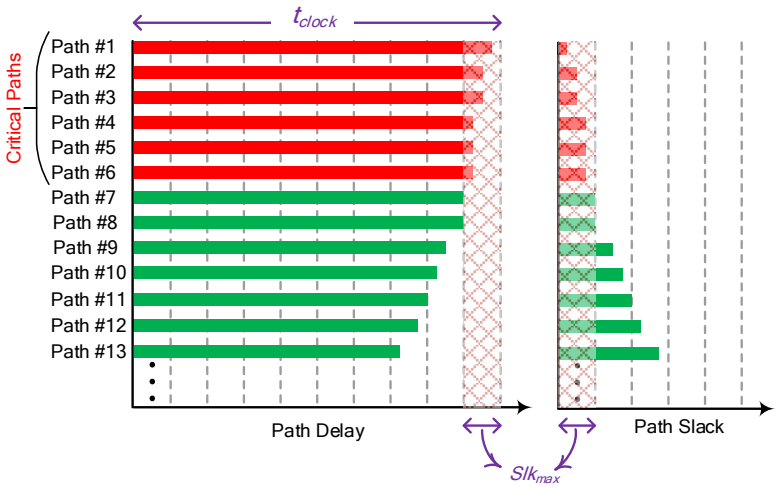



Figure 3.3: The definition of critical paths based on $Slk_{max}$.

### 3.3.2 *Identifying the Insertion Points*

The critical timing paths are defined as those paths whose delay is longer than a specific value. Therefore, we define the maximum monitored slack, $Slk_{max}$, as the constraint for selecting the critical timing paths to be monitored, i.e., the critical paths are all the paths whose delay is more than $(t_{clock} - Slk_{max})$. This definition is illustrated in Fig. 3.3.

The timing graph of a digital circuit is a weighted directed graph G. The set of vertices in G representing the ports and instance pins is $V(G)$, and the set of directed edges in G representing the timing arcs is $E(G)$. The timing arc from vertex $v_i \in V(G)$ to vertex $v_j \in V(G)$ is represented by $e_{ij} \in E(G)$. Let the vector $D \in R^{n\times 1}$ contain the maximum data arrival time to a vertex $v_i \in V(G)$. The maximum data arrival time to vertex $v_i \in V(G)$ is represented as $d_i \in D$. Industrial timing analysis tools can report the most critical path that includes any specific nodes, based on block-based timing analysis [93]. We employ this report to find the maximum data arrival time to each circuit node, i.e., to find $D$. Furthermore, based on this report, we can find the minimum slack of the paths that cross each specific circuit node. Therefore, block-based timing analysis also returns the vector $SLK \in R^{n\times 1}$, where $SLK$ contains the minimum slack of the paths that include each vertex $v_i \in V(G)$.

A cut $C = (S_1, S_2)$ is created based on D, where

$$S_1 = \{v_i \in V(G) | d_i < t_{mon}\}, \tag{3.1}$$

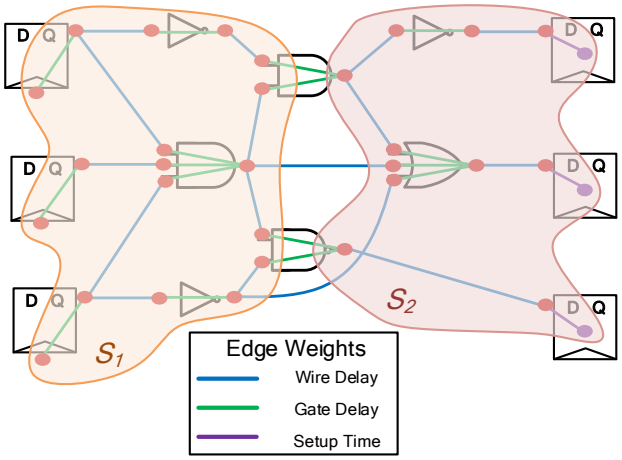



Figure 3.4: An example circuit graph showing the cut $C = (S_1, S_2)$.

and

$$S_2 = \{v_i \in V(G) | d_i \geq t_{mon}\}, \tag{3.2}$$

where $t_{mon}$ is the maximum delay from the launch flip-flop to the monitor insertion point. Note that during the clock tree synthesis, the design tool attempts to make the delay from the clock ports to the clock pin of all flip-flops the same. Naturally, in a more accurate analysis, clock skew should also be included. In the example of Fig. 3.4, a timing graph, and the cut are illustrated. The set of boundary vertices, $S_{bnd} \subset S_1$, is defined as

$$S_{bnd} = \{v_i \in V(G) | e_{ij} \in E(G), v_i \in S_1, v_j \in S_2\}, \tag{3.3}$$

where $S_1$ and $S_2$ are obtained from (3.1) and (3.2), respectively. The vertices in $S_{bnd}$ are candidates for insertion of the monitors. Screening is performed on $S_{bnd}$ to find the vertices which are in the critical paths and to insert the monitors at those vertices. The set of insertion points is therefore

$$S_{inp} = \{v_i \in V(G) | v_i \in S_{bnd} \wedge Slk_i < Slk_{max}\}, \tag{3.4}$$

where $S_{bnd}$ is obtained from (3.3), and $Slk_i \in SLK$. Once the set of insertion points is identified from (3.4), the monitors are inserted at the corresponding gate pins of all vertices in $S_{inp}$, according to the flow which is explained in the next subsection.

Alternatively, the built-in features of an industrial timing analysis tool can be employed to find the insertion points more effectively [93]. With the timing analysis tool, the most critical path that does not go through a set of particular circuit nodes can be reported. We use this feature of the tool to maximize $Slk_{max}$. Algorithm 2 is the proposed

algorithm to find the set of insertion points. The inputs of the algorithm are $n_{mon}$ which is the number of monitors, and $t_{mon}$ which is the maximum arrival time to the insertion point. The algorithm returns the set of insertion points $S_{inp}$ and $Slk_{max}$. $S_{inp}$ is initially an empty set, and the insertion points are added to this set in a for loop that iterates $n_{mon}$ times. In each iteration, the most critical timing path that does not go through any point in $S_{inp}$ is reported. Then, $Slk_{max}$ is set to the slack of the reported path, and the first point with an arrival time of more than $t_{mon}$ is added to $S_{inp}$. After $n_{mon}$ iterations, the final $S_{inp}$ and $Slk_{max}$ are obtained.

**Algorithm 2** The flow of find the insertion points with timing analysis tools.

```
1:  procedure FIND_INSERTION_POINTS
2:      S_inp ← ∅
3:      for i = 1 to n_mon do
4:          path ← report the most critical timing path which does not go through
5:                 any point in S_inp
6:          Slk_max ← slack of path
7:          inp ← find the first point in path with arrival time of more than t_inp
8:          S_inp ← S_inp ∪ {inp}
9:      end for
10: return S_inp, Slk_max
11: end procedure
```

### 3.3.3 *Integration in the Design Flow*

The monitors are conventionally added to the design after the place and route step (see e.g., [94] and [72]). However, such an implementation flow perturbs the layout and the timing of the main design [72]. To minimize the perturbation, we propose the implementation flow illustrated in Fig. 3.5. In the proposed flow, the monitors are added at the synthesis stage with dangling insertion point inputs of the monitors. Yet, the clock input of the monitors is connected such that their loading effect is taken into account during the design steps (e.g., clock tree synthesis). The implementation with unconnected monitors then goes through synthesis and back-end design steps. The insertion points of the monitors are identified based on the timing report when timing closure is achieved after the place and route step. The monitors are then truly inserted into the design by connecting them to the identified insertion points (i.e., $S_{inp}$). Note that with this approach, there is no loading on the non-critical paths and that the critical paths which are used to determine $S_{inp}$ are still the actual critical paths of the circuit after this step. The insertion is done by generating a TCL script which contains *Engineering Change Order (ECO)* commands. Alternatively, the TCL script provided in Code A.3 in

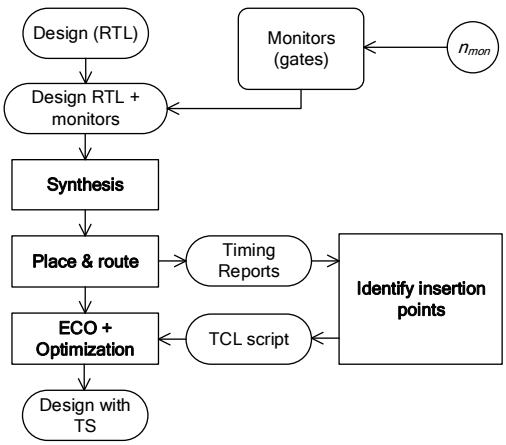



Figure 3.5: The proposed implementation flow.

Appendix A can be used to identify the insertion points with the built-in features of the timing analysis tool.

After connecting the monitor insertion points, optimization is performed to compensate for the loading effects of the monitors on the primary circuit. The maximum delay to the monitors is set to $t_{mon}$, to preserve guard-banding and to avoid over-optimization of the circuit during the final optimization. Hence, in the proposed implementation flow, the number of monitors $n_{mon}$ is decided at the front end. Since the monitored paths are the critical paths identified by $Slk_{max}$, with a given $n_{mon}$, we obtain a path coverage in terms of $Slk_{max}$. Intuitively, by increasing $n_{mon}$, a higher $Slk_{max}$ is possible since more paths are monitored. Alternatively, to monitor all of the critical paths according to a specific $Slk_{max}$, a prior analysis of the design is required to identify $n_{mon}$ according to $Slk_{max}$. The analysis is, in fact, going through the flow and identifying the set of insertion points in the design, without monitors and targeting $Slk_{max}$. $n_{mon}$ is then identified based on the number of identified insertion points. After this analysis, $n_{mon}$ is determined, and the monitors are added according to the flow that is shown in Fig. 3.5.

In this work, the conventional worst-case design approach for setup (i.e., slow corner) is followed, and the monitors are added based on timing reports in the slow corner. Inserting them at the points with the maximum arrival time of more than half of a clock cycle results in negative slack for the monitors, while the main design is still signed off with zero/positive slack. Note that in the typical corner all setup slacks increase, that the monitors' slacks also increase, but that the slacks of the monitors are still less than the slack of the main design to maintain the guard banding. Furthermore, since the ranking of the most critical paths can change with corner change, we select more

than one path to be monitored by defining $Slk_{max}$ as the criteria for path selection for monitoring. Therefore, one could insert the monitors in other timing corners provided that the monitored paths are appropriately selected (i.e., with proper selection of $Slk_{max}$) and the slack of the monitors are made less than the slack of the main design (i.e., with proper selection of insertion points).

## 3.4 ANALYSIS OF INSERTION POINT SELECTION

In this section, the selection of $t_{mon}$, as the central design knob in the proposed in-situ monitoring technique, is discussed. First, we discuss the effect of $t_{mon}$ on guard banding against variations.

### 3.4.1 *Guard banding*

Notice that when the monitor is inserted at a temporal intermediate point of the timing path, it does not capture the effect of timing spread on the remaining logic gates which are located after the insertion point. To investigate this effect, we performed 1000 Monte Carlo simulations on the SPICE netlist of the critical path of an industrial design in 40nm technology. The Monte Carlo simulation is performed to capture the effect of technology dependent local and global process variations on the gate delays of the path. In Fig. 3.6 (a), the scatter plot of delay samples is shown for a full path delay $t_{full}$, versus the delay of the path up to the insertion point of the monitor $t_{mon}$. In this figure, $t_{full}$ and $t_{mon}$ are normalized to clock period $t_{clock}$ and $0.5 \times t_{clock}$, respectively. The dashed lines in the figure divide the space into four regions according to the timing checks performed on $t_{mon}$ and $t_{full}$. These checks are at half of the clock cycle for $t_{mon}$, and at the end of the clock cycle for $t_{full}$. Each sample then occurs in one of these regions

- **True-Negative region**: $t_{full} < t_{clock}$ and $t_{mon} < 0.5 \times t_{clock}$. Therefore, the monitor truly detects no timing violation in the design.
- **True-Positive region**: $t_{full} > t_{clock}$ and $t_{mon} > 0.5 \times t_{clock}$. Therefore, the monitor truly detects timing violation in the design.
- **False-Negative region**: $t_{full} > t_{clock}$ and $t_{mon} < 0.5 \times t_{clock}$. Therefore, the monitor falsely detects no timing violation in the design.
- **False-Positive region**: $t_{full} < t_{clock}$ and $t_{mon} > 0.5 \times t_{clock}$. Therefore, the monitor falsely detects timing violation in the design.

True-Negative and True-Positive detection are desired, while False-Negative and False-Positive detections are not. False-Negative detection must be avoided because in this case, the gate delays after the insertion points may increase, but the monitors would falsely show that the system is reliable. On the other hand, False-Positive detection can be acceptable for chip-health monitoring since it reflects delay degradation of some gates while the full path still has enough margin. In fact, during regular circuit operation, only True-Negative detections are acceptable, and as the circuit degrades, it is pre-

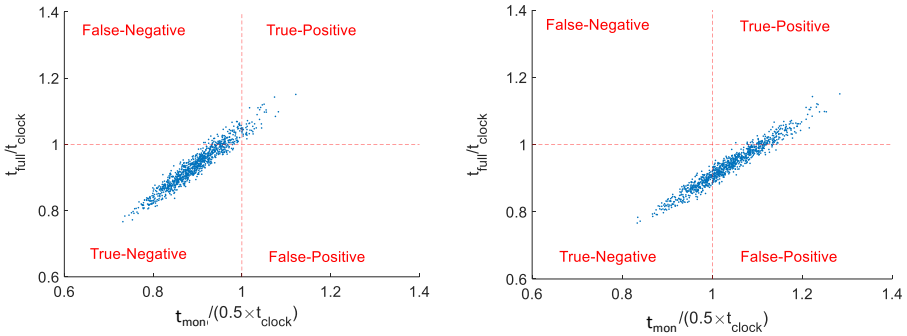



Figure 3.6: Scatter plot obtained from Monte Carlo simulations of a critical path delay of an industrial design showing $t_{full}$ versus $t_{mon}$ for the cases where (a) $t_{mon} = 0.5 \times t_{clock}$, and (b) $t_{mon} = 0.55 \times t_{clock}$.

ferred to have False-Positive detections before True-Positive ones, to have a guard band between the detection of delay degradation with the monitors and the actual timing violation in the design. Note that this guard banding is implemented without additional design cost by selecting the insertion points properly.

By inserting the monitor at a point closer to the endpoint, the delay to the insertion point is higher, i.e., there are more samples with $t_{mon} > 0.5 \times t_{clock}$. Therefore, the points are shifted to the right in the scatter plot, and thereby, more samples fall in the False-Positive region. The scatter plots shown in Fig.3.6 (a) and (b) are for the cases in which the insertion point is selected such that the average value of $t_{mon}$ is about $0.5 \times t_{clock}$ and $0.55 \times t_{clock}$, respectively. By inserting the monitors at the points which are after the temporal middle point of the clock period (i.e., $t_{mon} > 0.5 \times t_{clock}$), False-Negative cases are avoided at the cost of more False-Positive ones. In both plots, the clock period is selected as $t_{clock} = \mu_{full} + \sigma_{full}$, where $\mu_{full}$ and $\sigma_{full}$ are the average and the standard deviation of full path delay samples, respectively.

### 3.4.2 *Recall and Precision Analysis*

From a chip-health perspective, and for a more thorough investigation of the effect of $t_{mon}$ on the relevance of the monitors' outputs, we employ the *precision* and *recall* metrics from binary classification, defined as [95]

$$precision = \frac{N_{TP}}{N_{TP} + N_{FP}}, \tag{3.5}$$

and

$$recall = \frac{N_{TP}}{N_{TP} + N_{FN}}, \tag{3.6}$$

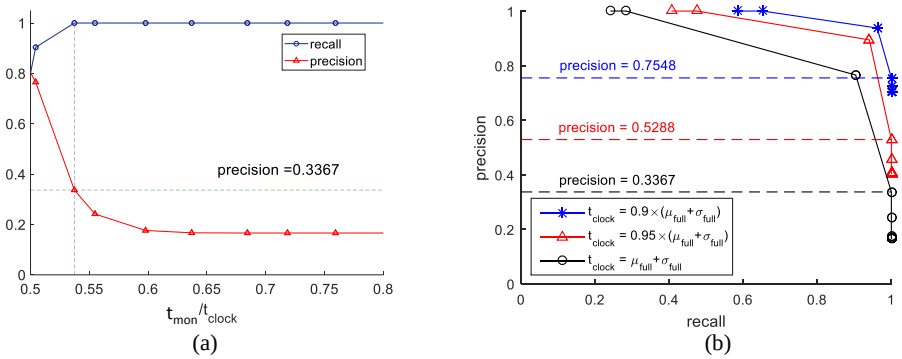



Figure 3.7: The effect of $t_{mon}$ on the *precision* and *recall* metrics (a) *precision* and *recall* versus $t_{mon}$ normalized to $t_{clock} = \mu_{full} + \sigma_{full}$ The maximum *precision* subject to no False-Negative prediction (*recall*=1) is 0.3367 (b) *precision* versus *recall* considering different $t_{clock}$ In a better-than-worst-case design (smaller $t_{clock}$ ) the *precision* of the monitor outputs increases.

where $N_{TP}$, $N_{FP}$, and $N_{FN}$ are the number of samples which are in the True-Positive, False-Positive, and False-Negative regions, respectively. In Fig. 3.7 (a), *recall* and *precision* are plotted versus $t_{mon}$ considering $t_{clock} = \mu_{full} + \sigma_{full}$. To avoid False-Negative samples, based on (3.6) $t_{mon}$ is selected such that $recall = 1$, i.e., $t_{mon} \approx 0.54 \times t_{clock}$. The obtained *precision* at this insertion point is 0.3367. There is a trade-off between *recall* and *precision* as $t_{mon}$ is varied. This trade-off is illustrated in Fig. 3.7 (b). The choice of $t_{clock}$ directly affects the available slack margin in the design. If $t_{clock}$ is longer (shorter), the slack margin is more (less) and the probability of timing errors is less (more). In Fig. 3.7 (b), *precision* versus *recall* is plotted for three choices of the $t_{clock}$: the worst-case value, $t_{clock} = \mu_{full} + \sigma_{full}$, and two better than worst-case values, being 5% and 10% shorter than the worst-case $t_{clock}$. The maximum *precision* considering *recall*=1 is also shown in the figure. As shown, a better-than-worst-case $t_{clock}$ results in a higher *precision* for the monitors. That is because with a shorter $t_{clock}$, more positive cases (i.e., timing violations) happen, and therefore this increases $N_{TP}$ (i.e., higher *precision*). On another hand, a longer $t_{clock}$ indicates that even though the monitor flags the delay increase, there is enough slack margin to avoid the timing violation at the expense of higher $N_{FP}$ (i.e., lower *precision*). Consequently, there is a trad-off between the choices of $t_{clock}$ and $t_{tmon}$ to maximize *precision* (with $recall = 1$).

Table 3.1: The runtime of path-based and the proposed method to identify monitor insertion points, considering $Slk_{max} = 0.1 \times t_{clock}$, for different ICT benchmark circuits synthesized for $1GHz$ clock frequency.

| Benchmark | Number of Gates | Path-based runtime (s) | Proposed Method runtime (s) | Speed up |
|---|---|---|---|---|
| **B03** | 76 | 14.918 | 2.496 | 5.977 |
| **B04** | 266 | 19.605 | 5.665 | 3.461 |
| **B05** | 438 | 2026.936 | 13.300 | 152.401 |
| **B07** | 228 | 14.386 | 5.166 | 2.785 |
| **B09** | 104 | 21.071 | 2.930 | 7.191 |

## 3.5 EXPERIMENTAL RESULTS

The proposed technique is employed for in-situ monitoring in an ARM Cortex M0 processor, implemented in an industrial 40nm technology. Cadence Genus and Cadence Innovus are used for synthesis and back-end design, respectively.

### 3.5.1 *Computation Efficiency of Monitor Insertion Method*

To show the computational efficiency for identifying the insertion points, we benchmarked it against the path-based method using six synthesized ITC benchmark circuits. The path based method requires the list of all paths whose slack is less than $Slk_{max}$. Path-based timing analysis is performed to get the list of the paths. The insertion points are then identified based on the arrival time to the intermediate points of all reported timing paths. On the other hand, with the proposed method, the set of insertion points are obtained, given the maximum arrival time to all circuit points through block-based timing analysis. We employed both methods to find the insertion points of the monitors to cover the same set of critical timing paths for $Slk_{max} = 0.1 \times t_{clock}$. All the circuits were synthesized with a 1GHz clock frequency. Depending on the circuit structure and size, the list of paths which should be monitored can be long or short. If the list of paths is long, the path-based method requires more time and effort to report it. In Table 3.1, the circuit size in terms of the number of gates, and the run-time of the path-based method as well as the run-time of our method are shown. Based on the results, the run-time of the path-based method is ~2.8X to ~152X more than our method, depending on the circuit structure and size.

Table 3.2: The implemented ARM Cortex M0 design specifications.

| | |
|---|---|
| **Target Frequency** | 200MHz |
| **Number of gates** | 8441 |
| **Number of flip-flops** | 841 |
| **Number of IOs** | 136 |
| **Power ($mW$)** | 2.333 |
| **Area ($\mu m^2$)** | 15646 |

### 3.5.2 *The Number of Insertion Points*

The proposed technique is verified for in-situ monitoring in an ARM Cortex M0 processor. The design specifications are provided in Table 3.2 for $t_{clock}$=5ns. The target speed of the design affects the topology of the circuit and its timing graph. Furthermore, the number of monitors varies depending on the specified $t_{mon}$. To investigate the effect of $f_{clock}$ and $t_{mon}$ on the number of monitors, $t_{mon}$ is varied from 50% to 100% of the clock period, for a design implemented with different target frequencies of 150MHz, 200MHz, and 250MHz. Fig. 3.8 illustrates the results of this experiment. A slower design is more relaxed, and the number of paths which are critical is less. Therefore, fewer monitors are required to monitor critical paths in a slower design. However, note that although a design with lower target speed generally requires fewer monitors, since the synthesis tool can synthesize the circuit differently, it can happen that for a specific $t_{mon}/t_{clock}$ ratio the number of required monitors is high with a lower target speed. This can be observed at $t_{mon} = 0.9 \times t_{clock}$ in Fig. 3.8. The results shown in Fig. 3.8 are obtained considering $Slk_{max} = 0.1 \times t_{clock}$. As can be observed in the figure, depending on the target design speed, there is a specific $t_{mon}$ for which the number of monitors is minimized. That is because changing the target design speed affects the circuit topology by changing the constraint for logic synthesis.

To show the efficiency of the proposed monitor insertion technique compared to endpoint monitoring, we swept $Slk_{max}$. The results for the 200MHz design are shown in Fig. 3.9. One can see that the number of in-situ monitors increases when $Slk_{max}$ increases. Thus, the effectiveness of the proposed method is higher when $Slk_{max}$ is smaller. With the proposed method, the number of monitors is reduced by up to ~23X compared to the traditional endpoint monitoring, as shown in Fig. 3.9.

### 3.5.3 *Adding the monitors to the design*

Based on the analysis discussed in the previous subsection, we added 64 monitors to cover $Slk_{max} = 0.05 \times t_{clock}$. To compare our technique with similar methods in terms

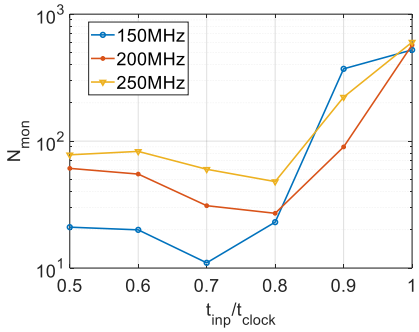



Figure 3.8: The number of monitors versus delay up to insertion point normalized to $t_{clock}$ for the ARM Cortex M0. The core was designed targeting different frequencies, considering $Slk_{max} = 0.1 \times t_{clock}$.

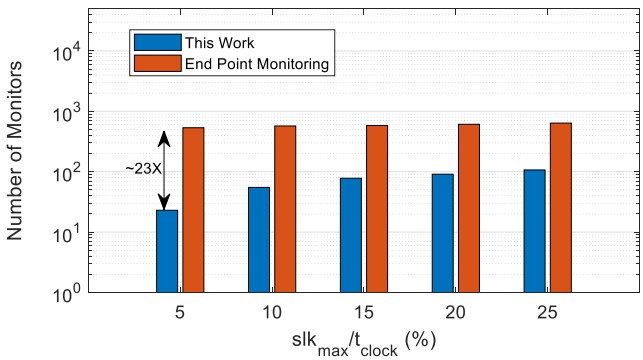



Figure 3.9: Comparison between the number of monitors with our technique ($t_{mon} = 0.7 \times t_{clock}$) and the number of monitored end-points for an ARM Cortex M0 processor considering different $Slk_{max}$ values normalized to the $t_{clock}$.

Table 3.3: Design overhead, monitored slack, and the number of monitored flip-flops with the proposed technique employed in an ARM Cortex M0 designed with a 200MHz speed target.

| **Techniques based on insertion points of monitors** | **Endpoint** | **SlackProbe** | **Proposed** |
|---|---|---|---|
| **Insertion point ($t_{inp}$)** | $t_{clock}$ | $0.9 \times t_{clock}$ | $0.6 \times t_{clock}$ |
| **Number of monitors** | 64 | 64 | 64 |
| **Power overhead (%)** | 10 | 16 | 2.9 |
| **Area overhead (%)** | 5.05 | 8.8 | 3.15 |
| **$Slk_{max}/t_{clock}$ (%)** | 2.8 | 5.2 | 8.5 |
| **Number of monitored flip-flops** | 64 | 457 | 543 |

of design overhead, we also implemented two alternative techniques. In one of the techniques, the monitors are added at the capturing flip-flops. In the other technique, the monitors are inserted almost at the end to reduce the number of monitors. In all cases, the number of monitors is fixed (64 monitors). The area and power overhead, as well as the covered slack and the number of monitored flip-flops, are shown in Table 3.3. The endpoint monitoring approach has the least coverage. According to the results, more monitors are required to cover the same list of critical paths. Besides, due to the additional delay elements and the shadow flip-flop, the design overhead is also too much for 64 monitors with endpoint monitoring, when compared to our technique. In the other technique, the monitors are inserted almost at the end of the timing paths ($t_{mon} = 0.9 \times t_{clock}$), which results in a reduced number of monitors. This approach allows achieving higher coverage of critical endpoints compared to the endpoint monitoring. Moreover, in addition to the monitor cells, more delay elements for the remainder of the paths are required. Therefore, the power and area of this technique are significantly more. Based on the observed results, with our technique, power and area overhead is about 5.5X and 2.8X less, respectively.

As was mentioned before, the monitors are truly inserted in the design when their inputs are connected to the identified insertion points. The slack of the monitors is determined after connecting the monitor inputs, and depends on $t_{mon}$ in the selection of insertion points. If $t_{mon} = 0.5 \times t_{clock}$, then the worst-case slack of the monitors is zero since the latches inside the monitors close at half of the clock cycle. If $t_{mon} > 0.5 \times t_{clock}$ the worst-case slack of the monitors is negative w.r.t $0.5 \times t_{clock}$, i.e., the monitors fail earlier than the main flip-flops in the presence of timing degradation. Therefore, the design is guard banded with the monitors if $t_{mon} > 0.5 \times t_{clock}$. In Fig. 3.10, the slack histograms of the monitors and the main flip-flops are illustrated, considering $t_{mon} = 0.6 \times t_{clock}$ and $t_{mon} = 0.7 \times t_{clock}$. The shown histograms verify that higher $t_{mon}$ results in lower slacks for the monitors. Note that the shown histogram is from a timing

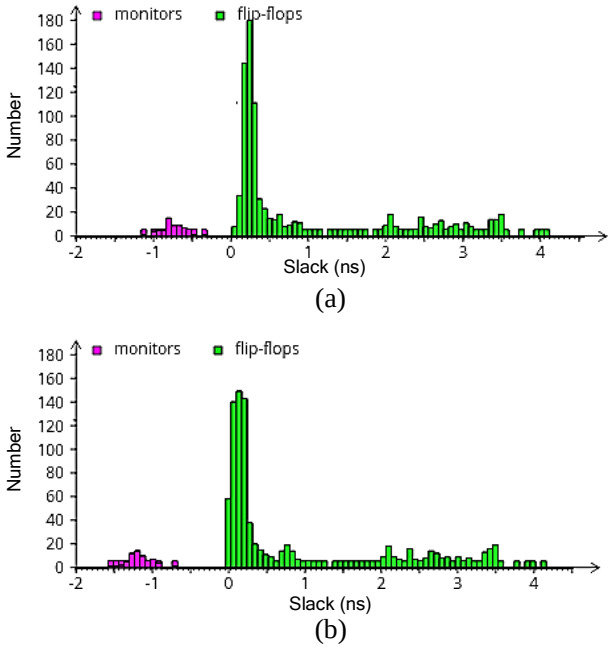



Figure 3.10: The histogram of slacks of monitors as well as the main flip-flops for ARM Cortex M0 design targeting 200MHz speed with 64 monitors considering (a) $t_{mon} = 0.6 \times t_{clock}$, and (b) $t_{mon} = 0.7 \times t_{clock}$.

analysis in the slow corner. In the typical corner, the bins shift to higher slack values The monitors must be inserted such that in regular operation (i.e., typical corner) the is slack is positive to avoid False-Positive error predictions. As the delays of the circuit components degrade, the slacks decrease. Since the slack of the monitors is smaller than the slack of the main flip-flops, the monitors flag the timing degradation before a timing violation occurs at the main flip-flop. With a higher $t_{mon}$, the monitors start flagging earlier because their slack is smaller, as shown in Fig. 3.10. Therefore, False-Positive error predictions are higher when a higher $t_{mon}$ is used for monitor insertion.

We performed a netlist simulation for 10K cycles and obtained the number of cycles in which at least one monitor flags a warning (i.e., error prediction signal). The simulation testbench runs image binarization, Finite Impulse Response filter, Infinite Impulse Response filter, and WHILE(1) applications on the processor. Timing annotation is performed at the typical corner, and delay scaling is applied to show the delay degradation effect. To examine the effect of $t_{mon}$ on the guard banding of monitors against variations, two cases of $t_{mon} = 0.6 \times t_{clock}$ and $t_{mon} = 0.7 \times t_{clock}$ were considered for monitor insertion. Fig. 3.11 shows the corresponding number of warnings. The ratio of the delay scaling factor for which a timing violation happens, to the smallest delay scaling

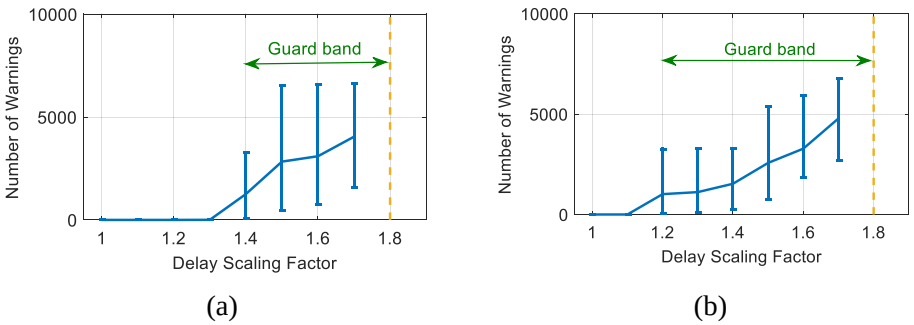



Figure 3.11: The number of cycles with warnings generated by the monitors versus the delay degradation factor. Results are of four applications obtained based on a netlist simulation for 10K cycles with timing annotation at the typical corner and scaling the delays. The insertion points of 64 monitors are identified considering (a) $t_{mon} = 0.6 \times t_{clock}$, and (b) $t_{mon} = 0.7 \times t_{clock}$.

factor for which the monitors generate a warning signal, reflects the guard banding of the monitors against delay variation. As shown in Fig. 3.11 (a), with $t_{mon} = 0.6 \times t_{clock}$ the design is guard banded by 1.38X. Similarly, it can be observed from Fig. 3.11 (b) that with $t_{mon} = 0.7 \times t_{clock}$ the design is guard banded by 1.5X. Therefore, with a larger $t_{mon}$ more guard banding is added with the monitors.

To assess the proposed methodology on a larger design, we applied it to an ARM Cortex M3 processor which has more than ~40K cells in 40nm technology (post place and route) targeting 200MHz speed. Similar to the results of Fig. 3.9, Fig. 3.12 displays the ones for the ARM Cortex M3. It can be observed that the proposed technique allows better monitoring compared to the endpoint monitoring technique. Assuming that $Slk_{max} = 0.05 \times t_{clock}$ is intended, with the endpoint monitoring technique ~2000 monitors are required while with our technique, the same slack coverage is obtained with less than 300 monitors when $t_{mon} = 0.7 \times t_{clock}$. Therefore, compared to the endpoint monitoring technique, more than 7X reduction in the number of monitors is achieved for the same path coverage ($Slk_{max} = 0.05 \times t_{clock}$). Furthermore, the power and area overhead of our in-situ monitoring technique are ~3% and ~1.9%, respectively.

### 3.5.4 *Comparison With Logic Built-In Self-Test for Delay Fault Detection*

In this subsection, the proposed technique is compared with *Logic Built-In Self-Test (LBIST)* technique for detection of timing degradation in a large industrial design [96]. The techniques are compared in terms of effectiveness and design overheads (i.e., power, area, and speed). LBIST is the standard method to test the ICs in-field by applying the

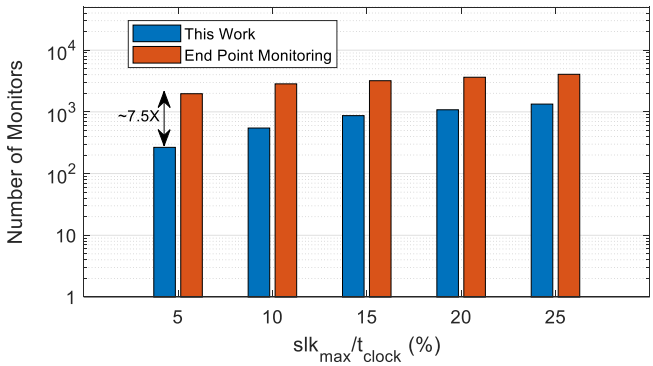



Figure 3.12: Comparison between the number of monitors with our technique ($t_{mon} = 0.7 \times t_{clock}$) and the number of monitored end-points for an ARM Cortex M3 processor considering different $Slk_{max}$ values normalized to the $t_{clock}$.

test patterns and analyzing the responses without the presence of an external tester (i.e., in-field testing). The test patterns are generated randomly and applied to some nodes in the circuit (control point), and the responses are collected from some nodes in the circuit (observation point) and compressed as a short signature to be compared with a reference signature that is stored in a memory. Although the primary usage of LBIST is detecting the breakdown of logical gates modelled as stuck-at fault (i.e., some nodes are constantly 0 or 1), it can also be used for at-speed testing. In at-speed testing, two test vectors are applied consecutively to excite the critical paths of the circuit. The resulting value in the flip-flops after applying the second vector must be captured at the nominal clock speed. If this response does not match the reference response, delay degradation is detected. This procedure must be repeated many times to make sure that most of (and ideally, all of) the critical paths are tested (i.e., the test coverage is maximized).

Both techniques are implemented in an industrial design in 40nm technology. The design has ~9 million gates, its size is ~800000 $\mu m^2$, and it can run with 125MHz clock frequency. The comparison of the techniques are summarized in Table 3.4. LBIST requires increasing the control points (i.e., additional AND, OR gates) and observation points (i.e., additional scan flip-flops) to achieve a good test coverage. Increasing the coverage also requires additional storage area to generate more test vectors and compare more responses. Overall, after adding ~10k control points, ~11k observation points, the control circuitry, and the required storage, LBIST increased the area and the total power of the circuit by 17% and 56%, respectively. On the other hand, our technique requires 2435 monitors to cover all paths whose slacks are less than 5% of the clock period. The area and the power overheads of our technique are 1.5% and 4.5%, respectively. Furthermore, with LBIST, replacing all flip-flops with scan flip-flops, and adding the control and ob-

Table 3.4: Comparing the proposed technique with LBIST technique for detection of timing degradation in a large industrial design.

| **Metric** | **LBIST technique** | **The proposed technique** |
|---|---|---|
| **Area overhead** | 17 % | 1.5 % |
| **Power overhead** | 56 % | 4.5 % |
| **Speed overhead** | high | low |
| **Detectable faults** | stuck-at fault, transition fault, path delay fault | path delay fault |
| **Path delay fault coverage ($Slk_{max}/t_{clock}$)** | 0.035 | 0.050 |
| **Online vs offline** | Offline | Online |
| **Integration in design flow** | good | good |

servation points, disturbs the timing of the circuit (specially the critical paths) to a large extent, while the speed overhead of our technique is due to the additional loading of the monitors which is marginal. As mentioned before, LBIST can also detect stuck-at fault. Besides, LBIST can detect the faults which cause large delay degradation in the circuit (known as transition faults). However, our technique can only detect delay degradation of the long paths provided that the degradation must be lower than half of the clock period such that it causes a transition in the inspection window of the monitors. On the other hands, LBIST can maximally cover critical paths whose slacks are less than 3.5% of the clock period while our technique can cover the critical paths whose slacks are less than 5% of the clock period. Note that this coverage can increase by adding more monitors. Since LBIST requires stopping the main circuit to apply the test vectors and collect the responses, it cannot be online. Our technique can run in parallel to the main circuit thereby it is an online technique. Finally, since adding LBIST and design for test infrastructure is nowadays a standard part of design flow, and we proposed a flow to integrate our technique in the design flow effectively, both techniques have good integrability in the design flow.

## 3.6 SUMMARY

A new in-situ chip-health monitoring technique is proposed in this chapter with monitor insertion at intermediate points along the critical timing paths. It is shown that with effective guard banding, the un-monitored part of the paths are protected such that false-negative detection is avoided. The guard banding is implemented without addi-

tional overhead per monitor by adequately selecting the insertion points. The proposed insertion technique is more than 2.8X faster compared to techniques that rely on path-based timing analysis. Moreover, the number of required monitors is up to 23X less compared to traditional endpoint monitoring for an ARM M0 processor. The power and area overheads of monitoring delay variation in the ARM Cortex M0 design is reduced by 5.5X and 2.8X, respectively. For designs with higher gate count, the number of monitors scales up, but according to our experience with ARM Cortex M3, the number of monitors with our technique is less compared to endpoint monitoring (~7.5X). Furthermore, the design overheads and effectiveness of out technique is compared with LBIST technique for detection of timing degradation in a large industrial design with ~9 million gates. The results show that our technique provides better coverage for detecting path delay degradation, while its design overheads are lower.

*"A single death is a tragedy; a million deaths is a statistic."*

— Joseph Stalin, *Tehran Conference*, 1943

# 4

# STATISTICAL IN-SITU DELAY MONITORING FOR EFFECTIVE CHIP HEALTH TRACKING

## 4.1 INTRODUCTION

In the nano-scale era, circuit components have become more unpredictable due to the variability and silicon aging effects. These uncertainties make the design of reliable electronic systems challenging. Moreover, the ever-increasing demand for more efficient designs (e.g., ultra low-power designs for battery-powered applications) adds to the design uncertainties by amplifying the uncertainty effects [23]. In the presence of aforementioned effects, tracking of timing reliability is crucial. The tracking technique must be such that the efficiency of the main design is not sacrificed due to the overheads of the technique. In this regard, improving the efficiency of in-situ delay monitoring is essential for an effective tracking of timing reliability. This efficiency increases if more information about the reliability status of the chip is extracted from the same monitoring hardware. In-situ delay monitoring is employed to sense the slack and flag a warning/error if the slack is critically low [97, 98]. That is done by checking if the critical path causes a transition during an inspection time window (we call this monitor excitation). Therefore, the output of the monitor is seen as a binary value indicating the presence of the problem. However, capturing the gradual effect of silicon aging on the circuit timing requires a fine-grain slack monitoring. For a more fine-grained slack measurement, *Time to Digital Converters (TDC)* are used [99]. The main challenge for TDC is the significant hardware overhead of this approach.

Conventional slack monitoring techniques intend to measure the worst-case static slack, i.e., the slack of longest timing path. In sharp contrast to the conventional techniques, a novel technique is proposed in this chapter that is based on *Monitor Excitation Rate (MER)*, that is defined as the probability of monitor excitation during a clock cycle. A monitor is excited if on of the paths that end at it is excited. As circuit delays degrade, the path delays increase. Therefore, excitation of more paths can cause excitation of a monitor when circuit delays degrade. Hence, the monitors are excited more

frequently when circuit delays degrade. We propose a novel chip health tracking technique in which a fine-grained signature of the delay degradation is extracted from the MER of monitors.

## 4.2 PRELIMINARIES AND RELATED WORKS

As defined in Chapter 2, a timing path is a sequence of timing arcs. A timing paths is excited if all of the timing arcs that constitute the path are excited consecutively. Consider two timing paths in the circuit, $L_1$ and $L_2$, that both of them start from the same origin. Assume $L_2$ is made up of the first $n_2$ timing arcs out of the total $n_1$ timing arcs of $L_1$, i.e., $n_2 < n_1$. We call $L_2$ the partial path of $L_1$. $L_1$ is excited only if the first $n_2$ constituting timing arcs (i.e., $L_2$) and also the rest of its $n_1 - n_2$ timing arcs are excited. Hence, the probability of excitation of $L_2$ is higher than or equal to the probability of excitation of $L_1$. By adding the in-situ delay monitors within timing paths as described in Chapter 3, new timing paths are added to the circuit which end to the latch of the monitors (see Fig. 3.2). The in-situ monitors can monitor a critical path $L_1$ based on the excitation of its partial path $L_2$. The observability of monitors is defined as the average number of monitored paths per cycle. Since the probability excitation of $L_2$ is higher than or equal to $L_1$ as mentioned before, the observability of within path inserted monitors are higher. Traditional in-situ delay monitoring techniques insert the monitors at the end point or close to the end point of timing paths [66, 72]. However, the observability is less when the monitors are inserted closer to the endpoints.

The conventional in-situ monitoring techniques only check whether or not the slack margins are tight (i.e., binary output) while it is essential to have a fine-grain slack monitoring to get a proper indication of aging effects. To get a fine-grain measurement of delay, TDC is employed in [62]. Although TDC is a reasonable choice for the replica path based or the RO based monitoring techniques, inserting TDC at each critical path increases hardware overhead substantially. Note that in reality multiple paths must be monitored because the most critical path can change considering manufacturing and environmental variations [98]. Besides, the sensitivity of TDC to variations limits its measurement accuracy [62]. The state-of-the-art static slack monitoring techniques do not use the underlying hardware in the most efficient way. The innovative technique of this chapter for in-situ delay monitoring enables fine-grain monitoring of delay degradation based on dynamic monitor excitation instead of measuring the static slack. With the proposed technique, MER is used as an indication of delay degradation. In the next section, the proposed chip-health tracking system is introduced.

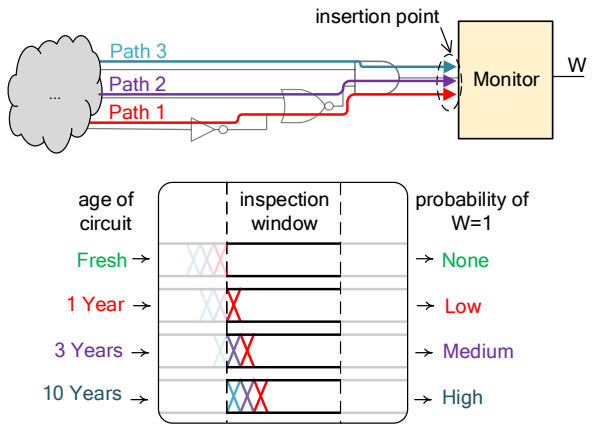



Figure 4.1: The proposed idea for chip-health tracking based on dynamic in-situ monitoring.

## 4.3 THE PROPOSED CHIP-HEALTH TRACKING SYSTEM

For each monitor $m$, the set of paths that cause monitor excitation $S_m$ are the paths whose slacks at the latch of the monitor are negative. Since during each cycle only one of these paths can be excited, the probability that the monitor $m$ is excited is

$$\text{MER} = \sum_{L \in S_m} p_L, \tag{4.1}$$

where $p_L$ is the probability of excitation of path $L$. As the circuit delays degrade, more paths become critical, i.e., the size of $S_m$ increases. Hence, based on (4.1), MER increases when the circuit delays degrade. This relation is illustrated in the example shown in Fig. 4.1. As shown in Fig. 4.1, suppose three paths with different delays are ending at the monitor. The monitor is excited when a transition occurs during its inspection window. When the monitor is excited, its output W becomes logic one. As path delays increase, more paths can cause transitions during the inspection window. Therefore, the MER increases when delays degrade, i.e., the probability of monitor's output W=1 increases. Hence, there is a strong positive correlation between delay degradation and MER. We employ this relation to generate a fine-grained signature of chip-health status with fewer hardware overheads.

The architecture of the proposed chip-health tracking system is shown in Fig. 4.2. This system is made up of the monitors and the signature extractor. As shown in the figure, the monitors are connected to internal circuit nodes along critical timing paths. Delay degradation is monitored by sensing data arrival time at the insertion points of monitors. The monitoring information is then scanned out periodically to be analyzed with the

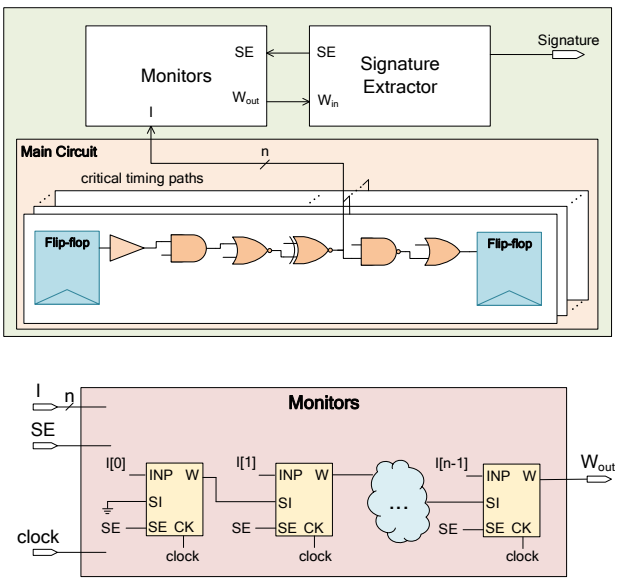



Figure 4.2: The proposed chip-health tracking system and the Monitors block which consists of the monitors connected in a scan chain.

signature extractor block. Accordingly, the monitors are either in the monitoring mode or in the scan mode and this is determined by SE, i.e., when SE is high (low), monitors are in scan (monitoring) mode. At the end of each scanning phase, all monitors are reset to zero for the next monitoring phase. In this way, a time sampling of monitoring excitation is implemented. The signature extractor continuously analyses the output of monitors and generates the reliability signature based on MER.

If MER reflects delay degradation which requires high monitoring observability, then the proposed system outputs a fine-grained health measurement. To increase the observability of monitoring, the monitors are inserted selectively at the intermediate points along timing paths using the method that was introduced in Chapter 3. In addition to that, since supply voltage scaling tightens the slack margins (i.e. increases the number of monitored paths) and it also magnifies the sensitivity of delays to device parameters, the supply voltage is lowered during chip-health tracking. This can be done safely during the operational mode of the circuit with the technique that is introduced in Chapter 5 of this thesis for timing resilience. Therefore, with the design time approach of finding the best monitor insertion points, and with the run-time approach of scaling the supply voltage during chip-health tracking, the observability of monitoring is increased. More detailed discussions of the monitors and the signature extractor are included in the following two sections, successively.

## 4.4 THE MONITORS

As shown in Fig. 4.2, the monitors are connected in a scan chain. The output of the last monitor in the chain is connected to $W_{out}$ and the input of the first monitor in the chain is tied low. During the scan phase, the stored values in the monitors are scanned out and logic zero propagates to all scan flip-flops of monitors from the input of the first monitor. Therefore, all monitors are reset to zero at the end of the scanning phase. Note that the monitoring data stream has no intrusion to the main data stream of the circuit. Furthermore, during the scan mode, there is no unnecessary toggling in the main circuit. Hence, the chip-health tracking is performed in parallel to the main functionality at the nominal speed. We take the design and the insertion flow that has been introduced in Chapter 3 for in-situ monitoring and modify the design to be used in the proposed chip-health tracking system. The design of the monitor is shown in Fig. 4.3. As mentioned before, the monitors are either in the monitoring mode (SE=0) or in the scan mode (SE=1). In the monitoring mode, the monitor detects late data arrival with a latch and an XOR gate. If the last transition at the insertion point occurs in the second half of the clock cycle, the output of the XOR gate becomes logic one. If a late arrival is detected, logic one is stored in the output scan flip-flop. With a NOR gate, the logic one stays in the scan flip-flop as long as the monitor is in the monitoring mode.

As discussed in Chapter 3, in-situ delay monitoring relies on path excitation. If the monitored path is not excited, the monitor fails to capture the degradation effect. Therefore, it is essential to monitor more than one path to increase the sensitivity of MER to delay degradation. Furthermore, in the presence of variation effects, the critical path of the circuit can change [100]. The paths with lower slack are more critical and have more priority for monitoring. All paths whose slack is less than a specific value are monitored and the monitors are inserted at intermediate points of those paths. The input activity of the monitor is illustrated in Fig. 4.3. The inspection window opens from $t = 0.5 \times t_{clock}$, and it closes at $t = t_{clock}$, where $t_{clock}$ is the clock period. Since multiple paths end at the monitor, activity at the insertion point can start from the beginning of the clock cycle, depending on the delay of the shortest path to the insertion point. The last transition during each clock cycle can happen at t=$t_{mon}$, where $t_{mon}$ is equal to the maximum delay to the insertion point. According to (4.1), MER as affected by the number of the paths that end into the monitor with negative slacks (i.e., the size of $S_m$) and the probability of excitation of each of those paths. Increasing $t_{mon}$ can change the size of $S_m$ and $p_L$ in (4.1) because the paths that end into the monitors change. Therefore, $t_{mon}$ affects MER by affecting the chance of transition during the inspection window. The optimal value of $t_{mon}$ depends on the circuit topology and the activity of circuit nodes.

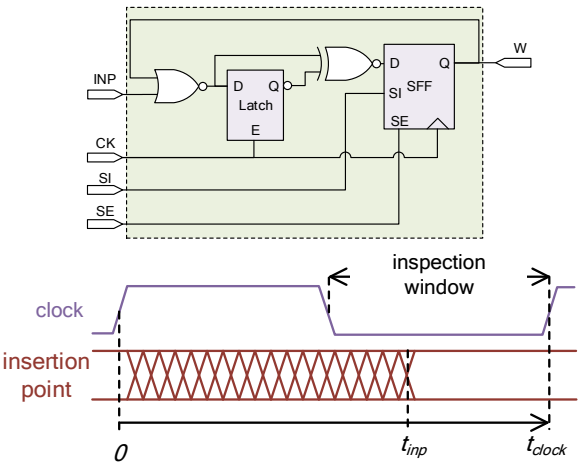



Figure 4.3: The design of monitor and activity of monitor insertion point.

## 4.5 THE SIGNATURE EXTRACTOR

As introduced before, the signature extractor analyzes the monitoring data to generate a reliability signature based on MER. To get MER, the monitors are periodically reset to zero after reading the monitoring data. In this way, a time-sampling of monitor excitation is implemented. In Fig. 4.4, the signature extractor is illustrated. SE controls the sampling period and it is generated with a counter which up-counts to a specified *period* and starts from zero again. When the counter value is less than $n_{mon}$, SE is logic one to scan out all monitoring data. The monitors are reset to zero after this by propagating logic zero in the scan chain. Then, SE becomes logic zero for (*period*-$n_{mon}$) cycles and during this time the monitors capture late data arrivals. Intuitively, when circuit delays increase, more monitors are excited, and each monitor is excited more frequently. To get an indication of MER, the overall number of excitation is counted continuously in the signature extractor, with an $m$-bit counter that is enabled with $W_{in}$. The counter up-counts until it reaches to its limit ($2^m - 1$) and starts from zero again. Hence, with higher MER the $m$-bit counter overflows faster. The overflow of the $m$-bit counter is flagged with a signal that is high for one clock cycle. Therefore, more frequent overflows indicates that the delay degradation effect is more severe. To measure this frequency, a timer is used to get the number of cycles between consecutive overflows of the $m$-bit counter. The output of the timer is sampled when $m$-bit counter overflows and the timer is reset to zero afterward. The sampled value of the timer can have too much variations due to the varying activities, and therefore varying path excitation, in the circuit. To have a

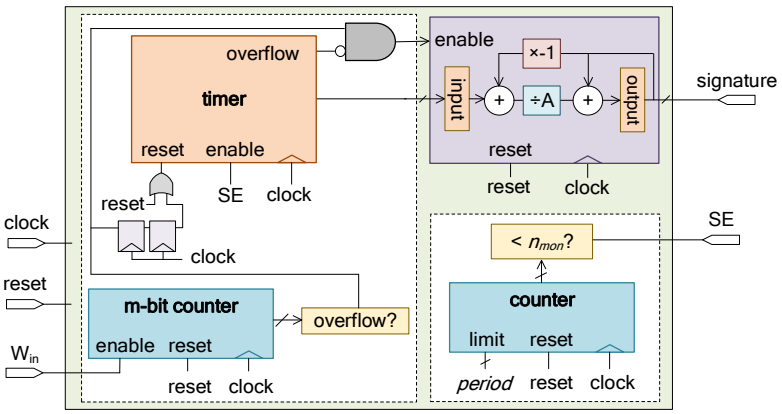



Figure 4.4: The design of signature extractor.

more stable output signature, low-pass filtering is applied to the sampled timer value. The final signature is thereby obtained as

$$S[i] = \frac{(A-1) \times S[i-1] + Timer[i]}{A}, \tag{4.2}$$

where $S$ is signature, $i$ is the sample number, $A$ is the weight factor, and $Timer$ is the sampled value of the timer. Since the $m$-bit counter is only enabled during scan mode ($W_{in}$ can be logic one only in this mode), the timer is disabled during monitoring mode by connecting enable port of timer to SE. As shown in Fig. 4.4, the overflow signal coming from the $m$-bit counter is combined with an overflow signal of the timer, to get the enable signal for sampling. That is because if $m$-bit counter does not overflow or it overflows very rarely, the timer overflows. In this case, the timer value is not valid and it should not be sampled. Therefore, the delay degradation for which signature generation starts is determined according to the bit-width of the timer. If the timer's bit-width is low, then it overflows faster and signature generation only starts if $m$-bit counter overflows more frequently, i.e. when delay degradation is more. On the other hand, if the timer's bit-width is high, it enables measuring the less frequent overflows of the $m$-bit counter, thereby the signature is generated for lower delay degradation factors. Therefore, the bit-width of timer determines the threshold of signature generation.

The design parameters of the proposed chip health tracking system are shown and described in Table 4.1. One can take an empirical approach to choose the optimal values for these parameters. That is because the optimal values of the parameters depend on the circuit topology and the activity of circuit nodes. In this work, we choose these parameters empirically for our benchmark circuit to illustrate that the proposed chip health tracking system is functional (the results are discussed in Section 4.6). Neverthe-

Table 4.1: The design parameters of the proposed chip health tracking technique.

| **parameter** | **description** |
|---|---|
| $n_{mon}$ | The number of monitors is determined based on the target coverage of the paths (discussed in Chapter~3). |
| $t_{mon}$ | The insertion points of the monitors are determined based on $t_{mon}$ which affects the number of monitors and guard-banding between monitors excitation and timing error in the circuit (discussed in Chapter~3) as well as maximizing the observability of the monitors (discussed in Section 4.4). |
| $A$ | The filtering coefficient of the sampled timer value in the signature extractor block. The selection of $A$ affects the smoothness of the output signature and the time that it takes to become stable. |
| $m$ | The bit-width of the counter which counts the number of monitors excitation (m-bit counter). A low value of $m$ causes more frequent over-flow of the counter and thereby the signature extraction starts when delay degradation effect is relatively low. When $m$ is larger the signature starts when delay degradation effect is relatively high. |
| *period* | The period of the pulse signal which drives scan enable inputs of monitors. The number of cycles in which SE is high is constant and equal to $n_{mon}$. The number of cycles in which this signal is high is equal to ($period - n_{mon}$). Therefore, *period* determines the duration in which the monitors are in the monitoring mode. |
| bit-width of *Timer* | The bit-width of timer in the signature extractor block determines the threshold of MER at which signature generation starts (i.e., signature>0). That is because timer is sampled when m-bit counter overflows. If the bit-width of timer is low, it overflows faster and no signature is generated until for relatively low MER. If the bit-width of timer is higher, then it can be sampled before overflow and for relatively low MER signature is generated. |

less, an analytical approach to find the optimal parameters is to be investigated in the future works. The objectives and constraints of the optimisations can be extracted from the descriptions of the parameters in Table 4.1.

## 4.6 EXPERIMENTAL RESULTS

The proposed chip-health tracking system is evaluated with an ARM Cortex M0 processor in 40nm technology. The target speed for the design is 200MHz. Cadence tools for front-end and back-end design, namely Genus and Innovus, are used for implementation. Timing analysis is performed with Cadence Tempus, and netlist simulation is performed with Cadence Incisive. The design corners are slow and fast for setup and hold analysis, respectively. To have a realistic sense of design margins, the typical corner is considered for netlist simulation with timing. We used the aging simulation that was introduced in Chapter 2, the BTI model from [24], and assumed that the threshold voltage of transistors degrades by 10% after 10 years of constant stress. To apply this degradation to the timing analysis, derating factors are specified for each of the gate delays according to the obtained delay degradation. For netlist simulation of the aged circuit, the SDF file is dumped from the timing analysis tool. Besides, the delay scaling factor is applied during timing annotation from the SDF file.

We performed netlist simulation for the ARM Cortex M0 with annotated timings to evaluate the functionality of our chip-health tracking system. The simulation testbench runs an FFT application on the processor for 100K cycles. 64 monitors are added to the design with $t_{mon} = 0.6 \times t_{clock}$. The period of SE is 128 cycles, which means that the monitors are in the monitoring mode for 64 cycles, as explained in the previous section. The output of the $m$-bit counter is shown in Fig. 4.5. For this simulation, three delay scaling factors are applied: 1.4X, 1.5X, and 1.6X. Note that based on our experiments, the delays can be scaled up to ~1.75X without having any timing error in the main design. As can be observed, with a higher delay scaling factor, the frequency of overflow increases. That is because MER is higher and this results in more countings with the $m$-bit counter. In this experiment, the bit-width of the $m$-bit counter is $m = 4$. With a higher (lower) $m$, the frequency of overflowing decreases (increases).

In Fig. 4.6, the output signature is shown considering different delay scaling factors. The bit-width of the timer is 11 and the weight factor in (4.2) is A=8. As can be observed, when the delay scaling factor is 1.5X, the signature is zero, i.e. the signature is not generated. That is because up to this delay scaling factor, the overflow frequency of $m$-bit counter is so low that the timer overflows always and the sampling is not enabled. When the delay scaling factor increases from 1.5X, the signature generation starts. As can be seen, with a higher delay scaling factor, the signature value is lower and that is because the overflow frequency of the $m$-bit counter increases when the delays are scaled up. The waveforms that are shown in Fig. 4.6 verify that the proposed chip-health tracking system reflects the delay degradation with distinct signatures for distinct delay scaling

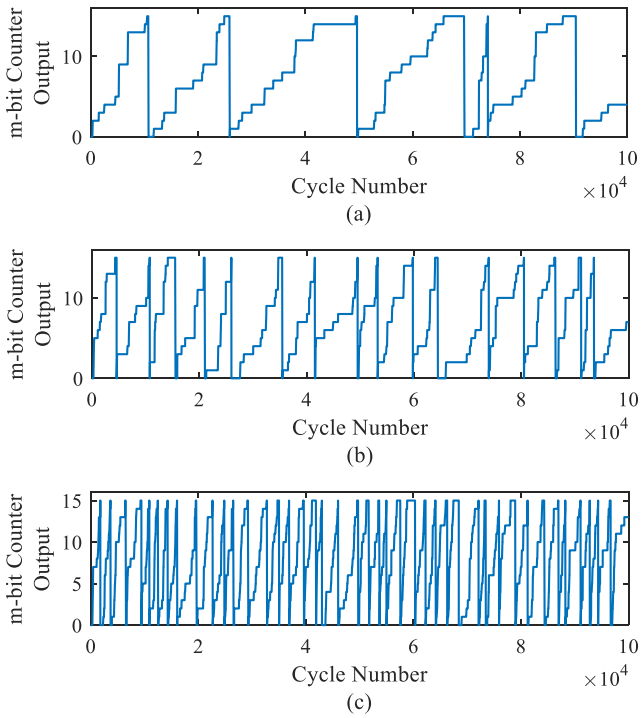



Figure 4.5: The output of the $m$-bit counter before overflow detector considering three delay scaling factors (a) 1.4X, (b) 1.5X, and (c) 1.6X.

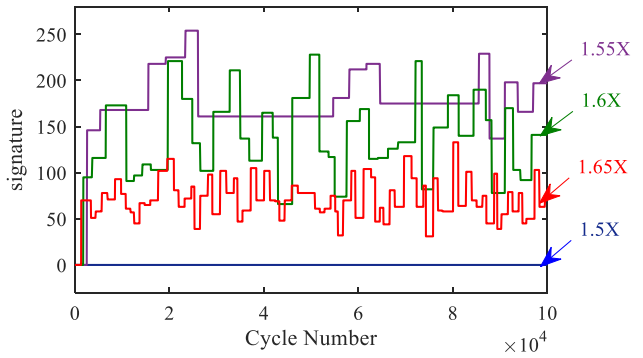



Figure 4.6: The signature considering different delay scaling factors.

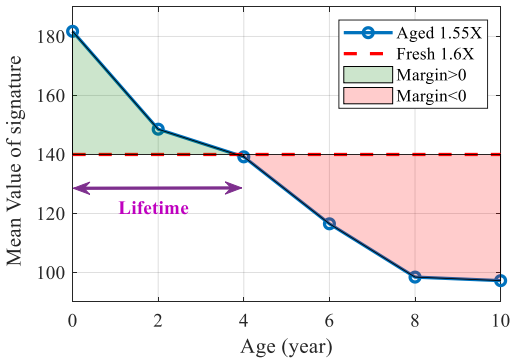



Figure 4.7: The averaged signature versus age with 1.55X delay scaling and the lifetime based on the signature with fresh 1.6X delay scaling.

factors. Nevertheless, to have a monotonic signature change with delay degradation, the signature output bits must be tied high when signature is not generated. For instance, in Fig. 4.6, signature value for 1.5X delay scaling should be higher than 1.55X. This can be done by setting the signature output register bits to one at the start, instead of resetting them to zero.

To get one number which reflects the age of the circuit, the average value of the signature is calculated. Note that the signature is a weighted average of timer samples and calculating its average requires less memory compared to calculating the average value of timer samples directly. In Fig. 4.7, the mean value of the signature over 50K cycles is shown versus the age of the circuit, considering 1.55X delay scaling. This scaling factor is used because based on the results that are shown in Fig. 4.6, the signature is more stable with 1.55X delay scaling. It can be observed that the age of the circuit is reflected in the mean value of the signature properly. Suppose, error-free functionality of the circuit is guaranteed when the voltage is scaled down to the point where the delays are 1.6X more than the nominal voltage. The mean value of the corresponding signature (SL) is shown with the dashed line in the figure. This value is stored when the circuit is fresh and during the lifetime of the circuit, the distance between the average signature and the threshold line is calculated as an indicator of available design margins. When the circuit is fresh, the distance is maximum, and as the circuit ages, the distance decreases, indicating that the slack margin is decreased. When the distance becomes negative, the margins are decreased to a level that the correct functionality of the main circuit cannot be guaranteed anymore. Therefore, the guaranteed lifetime of the circuit is ~4 years based on Fig. 4.7.

## 4.7 SUMMARY

In this chapter, a new chip-health tracking technique has been proposed which reflects the delay degradation in a fine-grained signature. The proposed technique is based on in-situ delay monitoring and it uses the underlying in-situ delay monitoring hardware more efficiently compared to the conventional static slack monitoring techniques. The idea is based on the fact that as the circuit delays degrade, the monitors are excited more frequently. A signature of chip-health status is then extracted from the excitation rate of the monitors, which indicates the degradation effects. The proposed idea is implemented for chip-health tracking of an ARM Cortex M0 processor. The results of netlist simulation with aging-induced delay degradation verified that the generated signature of chip-health status can reflect delay degradation clearly.

*"Delay is preferable to error."*

— Thomas Jefferson, *Letter to George Washington*, 1792

# 5

# WITHIN-CYCLE ERROR PREVENTION FOR TIMING RESILIENCE

## 5.1 INTRODUCTION

The traditional way of implementing a digital circuit with reliable timing considers the worst-case timing conditions at the design time. This static approach scarifies the circuit quality and results in large slack margins in most of the manufactured chip. *Timing speculation (TS)* systems intend to regain the sacrificed circuit quality by adding the slack margin dynamically, provided that the circuit should work reliably in the worst-case conditions. That is done by checking if a timing error has happened (going to happen), and masking (avoiding) the error accordingly. The chip health tracking techniques, which are based on in-situ delay monitoring, rely on the dynamic excitation of the monitors. Since the excitation of the monitors is based on the running workload, the observability of monitoring depends on the workload. For real-time chip health tracking, the system must remain operational while its reliability status is being tracked. As mentioned in Chapter 1, one of the requirement for chip health tracking is that it must run in parallel to the circuit's operation. To make sure that the circuit remains operational while it is being monitored, we propose a TS in this chapter to make the circuit resilient against timing degradation while it is being monitored.

A TS is presented in this chapter that is based on the low-overhead in-situ delay monitoring technique, which is described in Chapter 3. The insertion of monitors within timing paths enables timing error prevention within the same clock cycle. For timing error prevention, a global clock stretching unit extends the clock period temporarily. With the proposed error prevention technique, up to ~22% delay degradation is tolerated with a negligible energy overhead of less than ~1%. Furthermore, the proposed TS system is combined with supply voltage scaling to reduce the consumed energy by ~30%.

In what follows, first, the state-of-the-art TS systems are reviewed in section 2. The proposed TS system is explained in section 3. In section 4, the trade-off between the parameters of the proposed system is discussed. The experimental results of using the proposed TS system are provided in section 5. Finally, section 6 summarizes the chapter.

## 5.2 RELATED WORKS

TS systems are extensively investigated in the literature (see e.g., [67, 69, 71, 101, 102]). These systems are based on a monitoring mechanism and an error prevention/detection technique. Error prevention is better because the overheads, design intrusion, and complexity of error correction techniques are more. In many works, TS systems are based on in-situ monitoring due to advantages over other techniques (see Chapter 3 for a detailed discussion). In some works (see e.g. [91]), the error prevention based in situ monitoring is implemented by connecting the output of the monitor to the capture flip-flop of the monitored critical path to enable time borrowing. Time borrowing is thereby performed at the endpoint of each timing path to prevent timing error. Therefore, all capturing flip-flops must be redesigned to have a clock stretching circuitry inside. Besides, stretching the clock period locally at each flip-flop may cause hold time violations. This approach adds to the hold time margin of all flip-flops with clock stretching capability. Furthermore, clock stretching may accumulate along following pipeline stages depending upon consequent time borrowings, which eventually may cause a setup time violation at one of the flip-flops. Finally, in practice, multiple critical paths end at the same flip-flop in a normal design. Therefore, the clock stretching must be done when any of those paths are excited. Connecting the clock stretching enable signal to many monitors is not practical. Therefore, a local error prevention strategy may cause timing problems due to both setup and hold time violations in the next pipeline stage from skewing the clock between flip-flops. The authors of [94] propose a TS technique with within-cycle global clock stretching in which the occurrence of a timing error is predicted with a transition detector along the timing path. The insertion point of the monitor is selected such that the clock manipulation is performed before the end of the clock cycle. However, the in-situ monitoring technique in [94] does not take into account the variability of the unmonitored part of the path. Furthermore, it requires to generate an inspection window which adds to the overhead of the approach. The use of global clock stretching in combination with in-situ monitoring has been considered in other works too (see e.g. [103]). Nevertheless, the clock stretching unit in those works are PLL-based and normally it takes more than one clock cycle to take effect (see e.g. [104]). Recently, an all-digital clock stretching unit has been proposed in [105] which is based on a delay line to provide different clock phases.

In this chapter, a new TS is proposed to prevent timing error using within path inserted in-situ delay monitors and within-cycle clock pulse stretching. A new clock stretching unit is proposed that is more power and area efficient compared to the circuit proposed in [105] due to its efficient micro-architectural implementation. By using the proposed TS, up to ~25% delay degradation is tolerated with negligible energy overhead.

## 5.3 THE PROPOSED TIMING SPECULATION SYSTEM

The proposed TS system consists of in-situ monitors, an OR-tree, and a clock stretching unit, as shown in Fig. 5.1. The monitors are connected to a set of insertion points along critical timing paths. Since the insertion of monitors is at intermediate points along the paths, a (potential) delay degradation can be detected before the end of the clock cycle. The OR-tree collects the outputs of the monitors into one bit (W in Fig. 5.1). The clock stretching unit delays the start of the next clock cycle depending upon the output of the OR-tree to make sure that the delayed paths still fit into the current cycle. Therefore, the monitoring, the collection of monitor outputs with the OR-tree, and the error prevention should fit into one clock cycle.

The proposed clock stretching unit is illustrated in Fig. 5.2. In the clock stretching unit, the output clock is stretched by increasing the clock phase dynamically, i.e. the phase of the clock is increased by a fixed degree every time that the output of the OR-tree is high at the end of the cycle. This phase shifting is performed by circular selection from $n$ different clock phases, i.e. the clock phase can increase by $360°/n$. We considered four clock phases indicating that the clock phase can be shifted by 90°. The four phases are generated based on an input clock pulse whose frequency is 2X higher than the clock frequency of the main design, as shown in Fig. 5.2. The phase selection is performed using a MUX gate whose select signal is driven by a 2-bit counter. The 2-bit counter up-counts every time the output of the OR-tree is high at the end of the cycle. Hence, if at least one monitor predicts a timing violation, the clock stretching unit delays the upcoming clock positive edge by one-quarter of the clock period to avoid a setup time violation. One can consider a more fine-tuned clock stretching by having more clock phases. Note that clock stretching happens globally (i.e. at all flip-flops), thereby no risk of hold time violation. Since the output of the MUX is used for the counter, a narrow pulse will appear at the output clock (see n1 in Fig. 5.2). Glitch suppression is thereby required to have a clean clock signal at the output. This is done by generating the appropriate glitch suppression signal (see n3 in Fig. 5.2) and delaying the output of MUX (see n2 in Fig. 5.2).

Clock stretching has been employed in adaptive clocking systems. However, not all adaptive clocking systems are suitable for within-cycle error prevention because many of those require more than one cycle to take effect. In [106], clock gating is employed for clock manipulation. However, gating one clock cycle means shifting the clock edge by one cycle, which causes a performance loss. In other works, (see e.g. [105]) the clock is manipulated by shifting the edge for less than a cycle to reduce the performance loss when preventing timing errors. However, those techniques rely on a delay line which imposes higher hardware overhead and element-wise variation sensitivity compared to our clock stretching technique. We implemented the proposed clock stretching unit in 40nm technology. The proposed clock stretching unit has a power consumption of $1.126\mu$W at the nominal supply voltage (i.e. 1.1V) to generate a 200MHz clock pulse

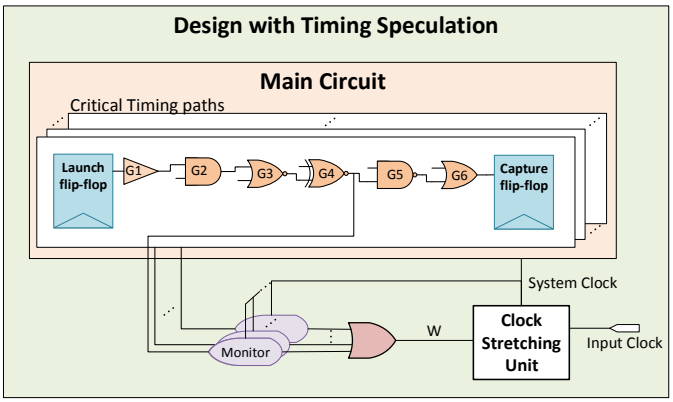



Figure 5.1: A high-level view to the proposed timing speculation technique.

at the output. Furthermore, the area of the proposed circuit is $96\mu m^2$. Therefore, the proposed circuit is much more power and area efficient compared to the circuit proposed in [105].

## 5.4 INSPECTION WINDOW AND OR-TREE DELAY

As introduced before, the inspection window of the monitor is the second half of the clock cycle. During the inspection window, the output of the monitor changes if there is any toggling at the insertion point after the negative clock edge. The insertion point is selected in the worst-case corner based on the maximum delay to the insertion point, which causes the last toggling at this point. Therefore, the maximum delay to the output of monitor $t_{mon}$ is the sum of maximum delay to the insertion point and the delay of buffers and XOR gate in the monitor (see Fig. 5.3). The monitoring data is thereby stable after $t_{mon}$ to be collected with the OR-tree into one bit, which triggers the clock stretching unit. Since the clock selection should be made before the end of the clock cycle,

$$t_{mon} + t_{OR} + t_{sel} < t_{clock}, \tag{5.1}$$

where $t_{OR}$ is the delay of OR-tree and $t_{sel}$ is the delay of clock selection (i.e., to generate enable signal of Counter in clock stretching unit). The OR-tree delay together with the clock period, thereby put a constraint on the maximum value for $t_{mon}$. We synthesized a 32, 64, and 128 inputs OR-tree with an industrial CMOS 40nm technology considering distinct maximum delays as the constraints. The power versus delay of the OR-tree with the different number of inputs is plotted in Fig. 5.3. The design point is selected such that the power of the OR-tree circuit is 10% more than the power when the delay

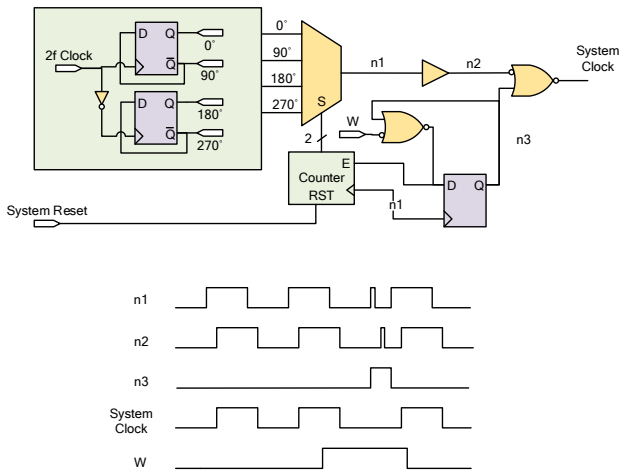



Figure 5.2: The design of clock stretching unit in the proposed timing speculation technique.

constraint is relatively long. Therefore, the delay of the OR-tree is in the range of 390ps to 550ps, depending on the number of inputs. Due to the design of monitors, $t_{mon}$ should be more than half of the clock period. Therefore, the OR-tree delay value indicates that according to (5.1), the proposed TS technique is suitable for the designs with the speed constraint of less than ~1GHz in 40nm technology.

## 5.5 EXPERIMENTAL RESULTS

In this section, experimental results are provided, and the effectiveness of the proposed TS system is evaluated. This evaluation is done considering the variation resilience achieved with the proposed system. The experiments were performed using an industrial CMOS 40nm technology. The proposed system is employed for timing resilience of an ARM Cortex M0 processor. Cadence Genus, Innovus, and Incisive were employed for the front-end design, back-end design, and netlist simulation, respectively.

### 5.5.1 *Robustness*

The proposed error prevention system is implemented by following the implementation flow, which has been introduced in Chapter 3. The OR-tree is also added to the design with monitors at the synthesis level, and proper timing constraints are set to avoid over-optimization. Besides, the output of monitors are connected to the inputs of OR-tree, and the output of the OR-tree, which collects monitor outputs, is connected to the

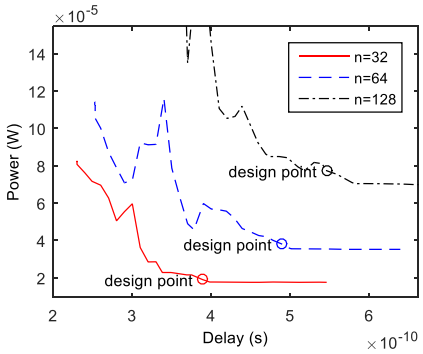



Figure 5.3: The power versus delay of the OR-tree considering the different number of inputs, n. The design point is selected considering 10% increase in the power compared to the power when the delay is relatively long.

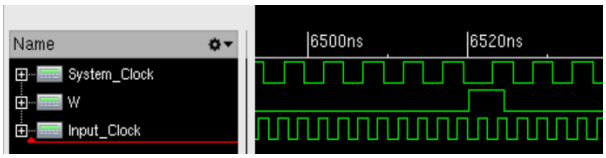



Figure 5.4: Netlist simulation waveforms showing System Clock being stretched by the clock stretching unit in response to a warning signal, W, generated by the monitors.

clock stretching unit in the same design stage. If a given monitor predicts an error, the next clock cycle starts with a delay (i.e., the clock edge is shifted) to avoid the timing error.

In Fig. 5.4, the simulation waveforms showing the output of the OR-tree (W) and the system clock are illustrated. The clock edge is shifted by 1.25ns (=$T/4$) if W is high at the end of the clock cycle. That means that the execution time of a task on the processor is delayed if error prevention is performed during its execution. On the other hand, the effective power consumption decreases with error prevention due to clock stretching as it decreases the effective frequency of the design. For instance, if at least one monitor flags a warning in each cycle, the clock is stretched for every cycle resulting in an effective clock period which is 25% longer.

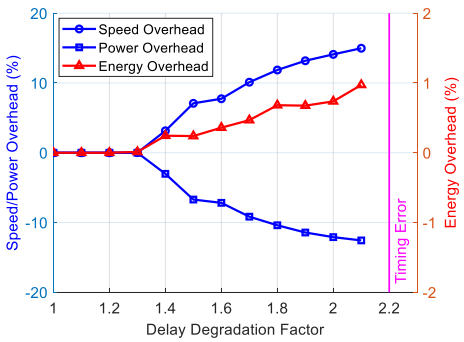
Therefore, the circuit works with a 20% lower clock frequency, and consequently, the power consumption decreases. Furthermore, putting less guard banding by having smaller $t_{mon}$ prevents unnecessary error prevention during error-free operation; thereby,

the performance penalty is less. In Fig. 5.5 (a) and (b) the speed, power, and energy overhead of the proposed error prevention system are illustrated with $t_{mon} = 0.6 \times T$ and $t_{mon} = 0.7 \times T$, respectively. With $t_{mon} = 0.6 \times T$, no clock stretching occurs up to 1.3X delay scaling. If the delay is scaled to higher values, the clock stretching occurs in some cycles, and the speed and energy overhead starts to increase. Note that the power overhead reduces due to the decreased effective clock frequency from stretching the clock.

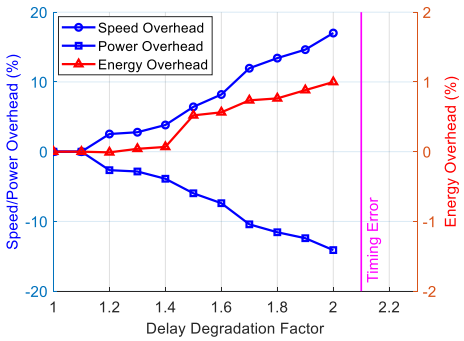
For $t_{mon} = 0.7 \times T$, the clock stretching starts from a lower delay scaling factor of 1.1X, because the slack of the monitors is less, i.e., the monitors are more pessimistic. It can be observed that with the proposed error prevention scheme, the system can tolerate more delay degradation compared to the original one. The resilience with $t_{mon} = 0.7 \times T$ is less compared to $t_{mon} = 0.6 \times T$, because according to (5.1) with higher $t_{mon}$ the output of the monitors becomes False-Negative for a lower delay scaling factor, i.e the monitor misses to capture the late transition at the insertion point. Without activating the technique, timing failure occurs when the delay scaling factor is 1.8X, while the technique allows delay scaling of up to 2.2X. Overall, the results show that with the proposed TS system, the resilience is increased by up to ~22% with a negligible energy cost of less than 1%.

### 5.5.2 *Better-Than-Worst-Case Design*

The proposed technique can be used in a better-than-worst-case design. A better-than-worst-case design can use overclocking or supply voltage scaling to convert the slack margin to power and speed improvements. Assuming that the IC instance is in the typical corner, the proposed TS system combined with supply voltage scaling is used to improve the power efficiency of the circuit. The LIB file (which contains power and timing information) is characterized for a typical case in different voltages from 0.9V to 1.1V. The characterized liberty files are used to see the effect of supply voltage scaling on the power and timing of the circuit. In order to assess the efficiency of the technique, the reference design is chosen to be an ARM Cortex M0 processor without the technique in the typical corner and at the nominal supply voltage (i.e. 1.1V). The supply voltage is scaled to lower than nominal values without timing failure. In Fig. 5.6, the speed, power, and energy overhead of technique versus the supply voltage are illustrated. In order to get this results, at each voltage, the circuit timing is analyzed, simulation with the annotated timing is performed, and the power is calculated with the obtained activity. The simulation test-bench runs *Fast Fourier Transform (FFT)* application for 10K cycles. As can be observed from the results, with supply voltage scaling, up to 10% speed overhead is caused due to clock stretching. On the other hand, power and energy are reduced by ~40% and ~30%, respectively. Therefore, the proposed TS system can be combined with supply voltage scaling to reduce the cost of over-engineering effectively.




(a)




(b)

Figure 5.5: Speed, power, and energy penalty due to clock stretching versus the delay degradation obtained from a netlist simulation of 10K cycles with timing annotation in the typical corner. The insertion points of 64 monitors are identified considering (a) $t_{mon} = 0.6 \times T$, and (b) $t_{mon} = 0.7 \times T$.

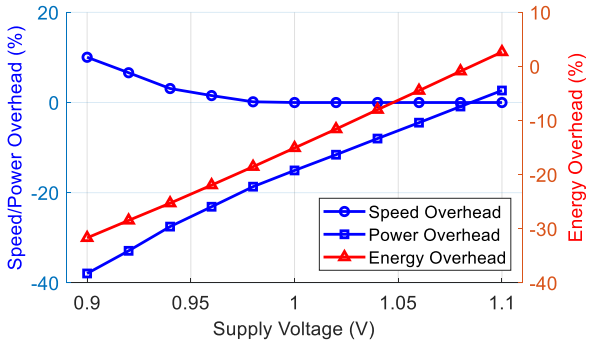



Figure 5.6: Speed, power, and energy overheads due to clock stretching versus supply voltage obtained from a netlist simulation of 10K cycles with timing annotation in the typical corner. The insertion points of 64 monitors are identified considering $t_{mon} = 0.6 \times T$

### 5.5.3 *Scalability*

Since in a larger design, the number of required monitors increases, the OR-tree should also scale up accordingly. According to (5.1), the sum of error prediction delay, OR-tree delay, and clock selection delay should fit into one clock cycle. Hence, the OR-tree delay can become a bottleneck when the design is too big, or the target speed is too high. In Fig. 5.7, the delay of the OR-tree is shown based on its power-delay characteristic considering different input counts. It can be seen that for 1000 inputs, the delay of the OR-tree is less than 1ns. According to (5.1) and assuming $t_{sel}$ is negligible, for $t_{OR}$=1ns and $t_{mon} \approx t_{mon} = 0.6 \times T$ the lower boundary for $T$ is 2.5ns, i.e. the maximum speed according to (5.1) is therefore 400MHz. On the other hand, with almost at the endpoint monitoring (e.g. $t_{mon} \approx t_{mon} = 0.9 \times T$), for the same OR-tree delay the maximum achievable speed is 100MHz (according to (5.1) and assuming negligible $t_{sel}$). Therefore, the proposed TS system allows a higher clock speed not only because it allows a reduction in the number of monitors (i.e. lower OR-tree delay) but also because it leaves more room for the OR-tree delay by inserting the monitors deeper inside the timing paths.

## 5.6 SUMMARY

A new TS has been proposed in which the monitors are inserted along timing paths, and timing errors are predicted within one clock cycle by checking if the delay of the first part of a path has increased such that an undesired transition happens during the second half the clock cycle. The variation resilience is improved by ~22% with the proposed error prevention technique. The proposed error prevention technique is based on a global clock stretching module connected to the output of an OR-tree which collects

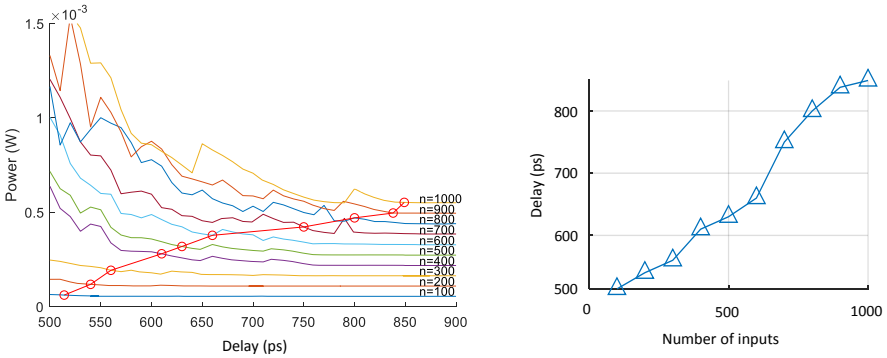



Figure 5.7: Delay of OR-tree versus the number of inputs based on the power-delay characteristic.

the monitor outputs. The energy overhead of resilience with the proposed error prevention technique is less than 1% which is negligible compared to the state of the art error recovery techniques. Furthermore, the proposed system is combined with supply voltage scaling to reduce the power and energy of an ARM Cortex M0 processor by ~40% and ~30%, respectively.

*"Science, my lad, is made up of mistakes, but they are mistakes which it is useful to make, because they lead little by little to the truth."*

— Jules Verne, *A Journey to the Center of the Earth*, 1864

# 6

# SILICON IMPLEMENTATION

## 6.1 INTRODUCTION

In this chapter, we will discuss the *Application Specific Integrated Circuit (ASIC)* implementation of the proposed ideas in two microprocessor platforms in 40nm technology. Due to some implementation issues, the expected functionalities are not fully achieved. The design issues are mainly related to the procedure of adding the monitors to the design (i.e.before synthesis or after place and route), enabling automatic clock gating in the design, and proper utilization of commands to report timing in the design tool. In spite of the implementation issues, the experimental results show that the proposed ideas are functional. In what follows, first, an ASIC implementation of the proposed within-path in-situ delay monitoring for chip health tracking is explained, and the design issues are discussed. Then, an ASIC implementation of the proposed in-situ delay monitoring and within-cycle timing error avoidance techniques are explained and discussed.

## 6.2 TAPE-OUT 1: CHIP-HEALTH TRACKING

A preliminary version of the proposed idea for chip health tracking (discussed in Chapters 3 and 4) was implemented in a microprocessor platform. The full chip contains two independent single-core microprocessor platforms which are physically separated. One of the platforms contains the chip health tracking technique, and the other platform serves as a reference to evaluate the design overheads of the technique. The output of the technique is a signature of reliability status based on scanning out the contents of within-path-inserted in-situ delay monitors. The design of the microprocessor platform is generated with CompSOC tool flow [107, 108]. CompSOC is a full stack of hardware from *Register Transfer Level (RTL)* to software for multi-core microprocessor platforms, that guarantee the composability and predictability of the running applications [108].

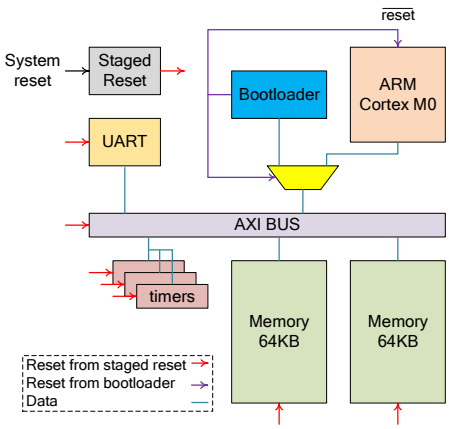



Figure 6.1: The architecture of the single-core microprocessor platform that is generated with CompSOC flow.

### 6.2.1 *The Architecture of Microprocessor Platform*

The architecture of the platform is shown in Fig. 6.1. As shown in the figure, the platform is made up of an ARM Cortex M0 processor, two 64KB *Static Random Access Memory (SRAM)* memory blocks, an *Advanced Extensible Interface (AXI)* bus, three timers, a *Universal Asynchronous Receiver-Transmitter (UART)* module, the stage reset block, and the boot-loader. The UART module has a programmable baud rate. Furthermore, the three timers enable having time stamps. The time stamps are needed to guarantee the composability and predictability of the running applications [108].

The boot procedure works as follows. When the platform is in reset, the stage reset block keeps the whole platform in the reset mode. When the reset input is released, the staged reset takes the platform out of reset after some cycles. The processor stays in reset until the boot-loader take it out of reset. The boot-loader waits for a 2-bit command through UART to perform one of the following actions:

- WRITE (`01`): write the UART data in a specified address of the memory,
- READ (`10`): read from a specified address of the memory and export it through UART, and
- START (`11`): take the processor out of reset, and give the control of the bus to it.

The content of memory is transferred through UART and using the write functionality of boot-loader. This procedure must be done when the chip is powered on because the SRAM memories are volatile (i.e., the contents of memories are lost when the chip is powered off).

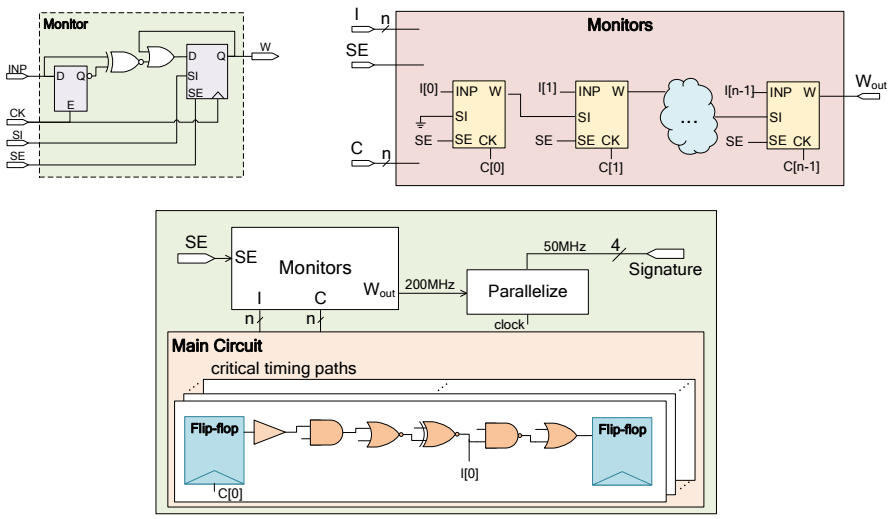



Figure 6.2: The design and insertion point of the in-situ monitor in tape-out 1.

### 6.2.2 *The Implemented Chip-Health Tracking Technique*

The chip health tracking technique is implemented in one of the microprocessor platforms when timing closure is achieved with 200MHz clock frequency. The implemented version of proposed technique is shown in Fig. 6.2. This design has similarities to the design which was introduced in Chapters 3 and 4 of this thesis (see Fig. 3.2, Fig. 4.2, and Fig. 4.3). The main differences are that in the implemented version:

- the gates of the monitors are added to the design after the place and route step instead of during the synthesis step,
- the clock pin of the monitors are connected to the clock pin of the launching flip-flop in the monitored critical path instead of being connected to the global clock,
- the delays to the monitor insertion points are not preserved, and
- the signature extractor block is not implemented (the raw monitoring data is scanned out).

In what follows, the implementation steps are explained.

As shown in Fig. 6.2, the monitors are inserted at the intermediate points of the critical paths. To have access to the monitoring data, the monitors are connected in a scan chain and they are either in the monitoring mode or in the scan mode. In the monitoring mode, any late data arrival to each monitor is captured by storing logic one in the monitor. In the scan mode, the stored monitoring data is scanned out at the nominal speed of the circuit (i.e. 200MHz). The modes are determined using the scan enable (SE) pin of the

monitors, available on the chip IOs. Due to the speed limitation of the IO cells, the monitoring data is parallelized into 4 bits to make sure that the signature bits do not toggle with more than 50MHz frequency during the scan mode.

As mentioned before, the implementation flow for this tape-out is different from the implementation flow that has been discussed in Chapter 3 of this thesis. In this tape-out, the logical gates of each monitor (i.e. one latch, one scan flip-flop, one XOR gate, and one OR gate) are added after the place and route step when timing closure is achieved. The clock input (CK), and the insertion point (INP) are identified for each monitor based on the timing report. In order to get the timing report, one critical timing path per flip-flop is reported after the place and route stage. For each reported critical path, the monitor gates are added first (using `addInstance` command). Then, the clock signal of the launching flip-flop of the path is connected (using `attachTerm` command) to the CK input of the monitor. Furthermore, after obtaining the data arrival to each internal point of the path in the timing report, the insertion point is selected based on the monitor insertion criteria (i.e., $t_{mon}$ as defined in Chapter 3). The selected insertion point is then connected (using `attachTerm` command) to the INP input of the corresponding monitor. The scan chain is made by connecting the scan input of the inserted monitor (SI) to the the output of the previously inserted monitor (using `attachTerm` command). For the first inserted monitor, SI is connected to logic zero with tie low cell. ECO commands (i.e., `ecoPlace` and `ecoRoute`) are used afterward to perform the physical placement and routing of the monitor gates in the design. 94 monitors have been added to design based on the timing reports to cover one critical path per flip-flop, for all flip-flops whose slacks are less than 15% of clock period (i.e., $Slk_{max} = 750ps$) with $t_{mon} = 0.6 \times T = 3ns$. Note that since only one critical path per flip-flop is covered, the number of monitors is much less than what is expected considering the design size. Finally, post-route optimization is performed to minimize the design intrusion of the monitors on the timing of the circuit, i.e., to compensate for the additional loading on the circuit nodes. This final optimizations are performed without preserving the maximum arrival time to monitors. Therefore, the design tool tries to optimize the design such that the latest arrival time to the insertion point is not larger than half of the clock period, because the latch of monitors closes at the negative clock edge. This affects the timing of the circuit to some extent and the monitors are not necessarily inserted at the intended point, i.e., $t_{mon} = 3ns$ does not necessarily hold after performing the optimization.

As discussed in Chapter 3, the monitors can predict a timing error if there is still enough slack at main flip-flops while the slacks of the monitors are negative. In Fig. 6.3, the histogram of the design slacks is shown in the slow corner, the typical corner, and the typical process corner with scaled supply voltage. As can be seen, the slacks of the monitors are negative in the slow corner, while the slacks of the main flip-flops are non-negative. In the typical corner (P=TT, $V_{DD}$=1.1V, T=27°C), all slacks increase as shown in Fig. 6.3-(b). Based on the shown histogram, the slacks of both the monitors and the main flip-flops are non-negative in the typical corner. Hence, there is no monitor

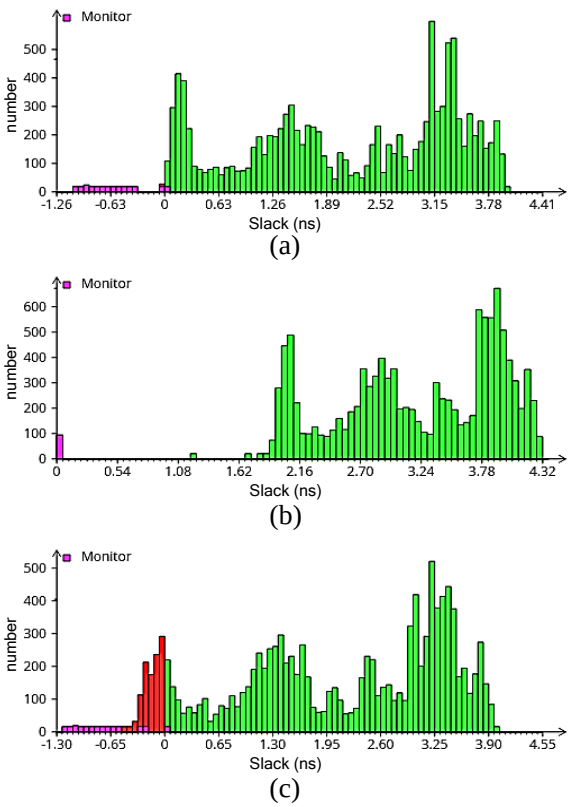



Figure 6.3: The slack histogram for (a) the slow corner, (b) the typical process corner at nominal voltage (1.1V), and (c) the typical process corner with lowered supply voltage to 0.9V.

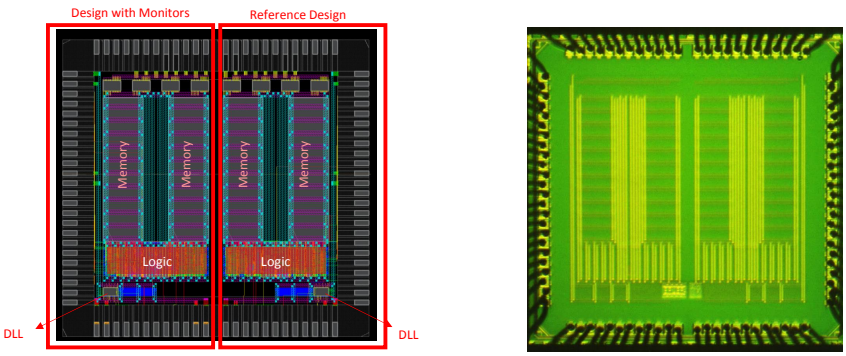



Figure 6.4: The layout and the die micro-graph of the first tape-out.

excitation in the typical corner. In Fig. 6.3-(c), the histogram of the slacks is shown in the typical corner with scaled supply voltage to $V_{DD}$=0.9V. As can be seen, the slack of the monitors, as well as some of the main flip-flops become negative in this corner. However, the slacks of the monitors are still less than the slacks of the main flip-flops. Therefore, the monitors are more pessimistic than the main flip-flops, and they are excited earlier than timing error in the main flip-flops.

### 6.2.3 *Physical Implementation*

Starting from the RTL code of the microprocessor platform, Cadence Genus and Innovus were used for synthesis and back-end design, respectively, with an industrial standard cell library in 40nm technology. In Fig. 6.4, the layout and the die micro-graph of the chip are shown. The die size is 1.85mm×1.79mm, and the total number of gates and IOs are ~128k and 96, respectively. As shown in the figure, the two design are physically separated and they can work independently. The monitors are added to the left platform and the left platform is used a reference. The floor-planning and IOs are mirrored to make sure that the designs are comparable in terms design costs. A *Delay Locked Loop (DLL)* is used to synthesize the internal 200MHz clock from an external 50MHz oscillator designed by Alejandro Rodriguez [109]. The DLL can multiply the input frequency by X1 (bypass DLL), X2, X4, and X8 factors. There are three voltage domains in each design: The IO voltage domain (2.5V), the fixed core voltage domain for DLL and the internal IO signals (1.1V), and the scalable core voltage domain for logic gates and memories ($\leq$ 1.1V). Having the scalable voltage domain allows us to increase the delays with supply voltage scaling. With increasing the delays, we can reduce the slack margins to mimic timing degradation and validate the functionality of the proposed chip-health tracking technique. For the scalable voltage domain, the analog type of voltage IO cells are used from the IO library. Since this type of IO cells do not have *electrostatic discharge (ESD)*

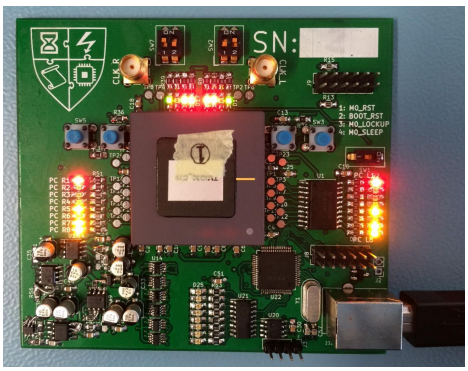


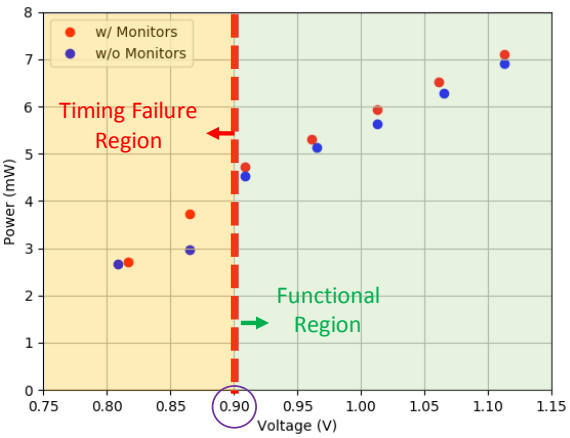

Figure 6.5: The evaluation board for tape-out 1.

protection, four ESD protection cells are added in each platform, as can be seen at the top of the layout shown in Fig. 6.4. For debugging purposes, 8 bits of the program counter register of CM0, the sleep pin of CM0, the lockup pin of CM0, and the reset pin of CM0, and the reset output of staged reset block are wired out on the chip IOs. Furthermore, a down-converted version of the internal clock signal is available on the chip IOs. For the monitors, the SI input and four bits of scanned out monitoring data are available on the chip IOs. In the reference design, the monitoring related output pins are tied low.

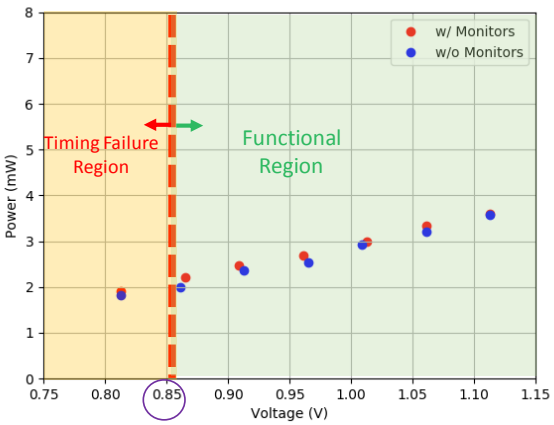
### 6.2.4 *Experiments*

The evaluation board is shown in Fig. 6.5. The board is powered through a USB B port. An FTDI chip is used as a USB to serial converter to communicate with the chip. The supply voltages of the chip (including the scalable one) are connected to an on-board programmable voltage regulator chip. Also, current and voltage measurements are performed on the board through some specific for this purpose. Programming the supply voltages and reading from the current and voltage sensors are also done through the FTDI chip.

In Fig. 6.6, the power consumption of both designs versus supply voltage are illustrated when the designs are running an application at 200MHz and 100MHz. Since the timing of the design is closed at the slow corner, for the measured chip, which is in a better-than-worst-case corner, the supply voltage can be scaled down without any timing error. When the supply voltage is lower than a specific value, a timing violation occurs in the design. The minimum functional supply voltage is less when the running frequency is lower. That is because the slack margin increases at a lower frequency. As can be observed in Fig. 6.6, the minimum supply voltage is ~50mV lower when the running frequency is reduced from 200MHz to 100MHz. The results of the power measurements are summarized in Table 6.1. Based on the results, the overall power overhead




(a)




(b)

Figure 6.6: The total power consumption of the monitored and reference platforms versus the supply voltage at (a) 200MHz, and (b) 100MHz.

Table 6.1: The power measurements at the nominal voltage and two running frequencies for the first tape-out.

| | **Monitored Platform** | | **Reference Platform** | |
|---|---|---|---|---|
| **Frequency** | 200MHz | 100MHz | 200MHz | 100MHz |
| **IO Current (mA)** | 10.08 | 8.39 | 9.98 | 8.37 |
| **Core Current (mA)** | 6.51 | 3.41 | 6.25 | 3.29 |
| **Core Power (mW)** | 7.225 | 3.750 | 6.904 | 3.668 |
| **Total Power (mW)** | 32.900 | 24.812 | 32.595 | 24.778 |

of the technique is ~0.93% and ~1.1% at 200MHz and 100MHz, respectively. Since the IO power is dominant, the core power (i.e. the power of logic gates and memories) gives a better sense of the power overhead of the technique. Considering the core power consumption, the power overhead of the monitoring technique is ~4.65% and ~2.24% at 200MHz and 100MHz, respectively.

To test the functionality of the chip-health tracking technique, the 4-bit monitoring data is probed with an oscilloscope. A periodic square wave signal with a proper duty-cycle is connected to SE input of the monitors. During the experiment, the platforms run an application in an infinite loop, and to excite the monitors, the supply voltage is scaled down. At the nominal supply voltage, unexpected toggling has been observed on the monitoring data output (i.e., the signature) at the beginning of the scan period. This observation is not expected because the process corner of the chip is typical and the slack of the monitors must be so large that they should not be excited. To make sure that the slack of the monitors are sufficiently large, the same experiment is repeated at lower frequencies (100MHz and 50MHz). However, we observed no change in the output of the monitors. Also, when the supply voltage was scaled down (to reduce the slacks), the output signature stayed the same at all voltage and frequency pairs. Changing the supply voltage and frequency should increase the MER, and the output signature must contain more toggling at lower supply voltages and/or higher frequencies. We also repeated the same experiments on different samples of the chip. Nevertheless, the output signatures were the same for all samples at different voltage and frequency pairs. Therefore, there must be a design issue which must be investigated.

### 6.2.5 *Discussions*

To find the design issue, we performed a netlist simulation with annotated timings in the the typical corner. Based on the histogram shown in Fig. 6.3-(b), toggling at the output signature is not expected because the slacks of the monitors are not negative in the typical corner. However, the simulation waveforms were consistent with the exper-

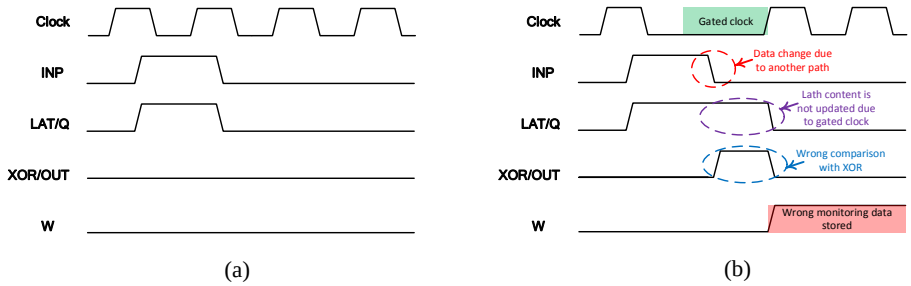



Figure 6.7: Example waveforms to illustrate the design issue with tape-out 1 due to automatic clock gating insertion during logic synthesis. (a) The expected functionality of the monitor without the gated clock signal, and (b) wrong information is stored in the monitor due to the gated clock signal.

imental results, i.e., the same toggling is observed on the output of the monitors. After investigating the source of the problem, the insertion of clock gating cells during logic synthesis is identified as the source of the problem.

During logic synthesis, the synthesis tool has automatically inserted clock gating cells in the design (automatic clock gating was enabled in the synthesis script). Automatic clock gating is a technique for power reduction in which unnecessary activity at the clock pin of flip-flops are avoided by gating the clock at some flip-flops based on the data path. More specifically, the tool inserts clock gating cells to gate clock pulse in the cycles that the data in the flip-flop stays the same. As mentioned before, the clock pin of the monitors (see Fig. 6.2) are connected to the clock pin of the launching flip-flops of the monitored critical paths. Connecting the clock pins of the monitors to the clock pins of flip-flops means that the clock signals of the monitors can be gated occasionally. However, since the data at the insertion point can change due to another timing path, wrong information can be stored in the monitors, as illustrated in Fig. 6.7 with some example waveforms. In Fig. 6.7-(a), the example waveforms of clock signal, the data at the insertion point (INP), the output of the latch inside monitor (LAT/Q), the output of the XOR gate inside monitor (XOR/OUT), and the output of the monitor (W) are illustrated for a chip without timing degradation. As expected, the content of monitor (W) stays zero when the clock signal is not gated. In Fig. 6.7-(b), a similar example is illustrated with gated clock signal at the monitor. INP can change with another timing path with a different clock signal which is not gated in the same cycle. However, the gated clock pulse keeps the latch opaque to its input, and thereby its content does not change with INP. This difference between INP and LAT/Q is flagged with XOR gate and the monitor stores a wrong value at the following clock positive edge. Therefore, with gated clock signal, the monitor mistakenly warns of timing degradation in the circuit.

The main design issue with this implementation is that the cells of monitors are added to the design after place and route and the clock pins of the monitors are connected to the launching flip-flops of the reported critical paths. Besides, the loading effect on the clock tree needs to be balanced during clock tree synthesis step, not after the place and route steps. This means that when the monitors are activated, the clock signal is disturbed, and this may cause functional failure in the main design.

Hence, due to an improper insertion of the monitors and with automatic clock gating insertion in the design, the taped out version of the proposed in-situ chip-health tracking technique is not functional. Note that this design issue is solved in the implementation flow that was introduced in Chapter 3 (see Fig. 3.5), in which the monitors are added at the synthesis stage with clock pin connected to main clock signal, and INP input connected after timing closure. In this way, we make sure that the clock signals of the monitors are not gated and their loading effects on the main clock tree of the design are taken into account properly.

## 6.3 TAPE-OUT 2: WITHIN-CYCLE ERROR PREVENTION

In the second tape-out, a multi-processor platform that is generated with CompSOC tool flow is implemented in TSMC 40nm technology. One of the three processor tiles in the platform is made resilient against timing degradation by in-situ monitoring and clock pulse stealing.

### 6.3.1 *The Architecture of Multi-processor Platform*

The architecture of the platform is shown in Fig. 6.8. The platform has three processor tiles. The processor tiles are similar to the single-core platform which was implemented in tape-out 1, with additional communication memories and network on chip interface. The complete platform also contains one memory tile, one communication tile, and a *Network on Chip (NoC)* [110]. Furthermore, one of the processor tiles (tile 3) is equipped with a *Coarse-Grained Re-configurable Architecture (CGRA)* accelerator [111]. For a more detailed view of the compSOC architecture is shown in Fig. B.1 in Appendix B. The proposed techniques for chip-health tracking and timing resilience are implemented in the processor tile 2.

### 6.3.2 *The Implemented Chip-Health Tracking and Resilience Techniques*

The implemented chip-health tracking and timing resilience techniques are shown in Fig. 6.8. Timing resilience is achieved by gating one clock pulse from the system clock. Note that with this approach the clock period is extended by one full clock period, while in the TS system which was introduced in Chapter 5, the clock period is extended by a quarter of original clock period. Therefore, the performance overhead of the imple-

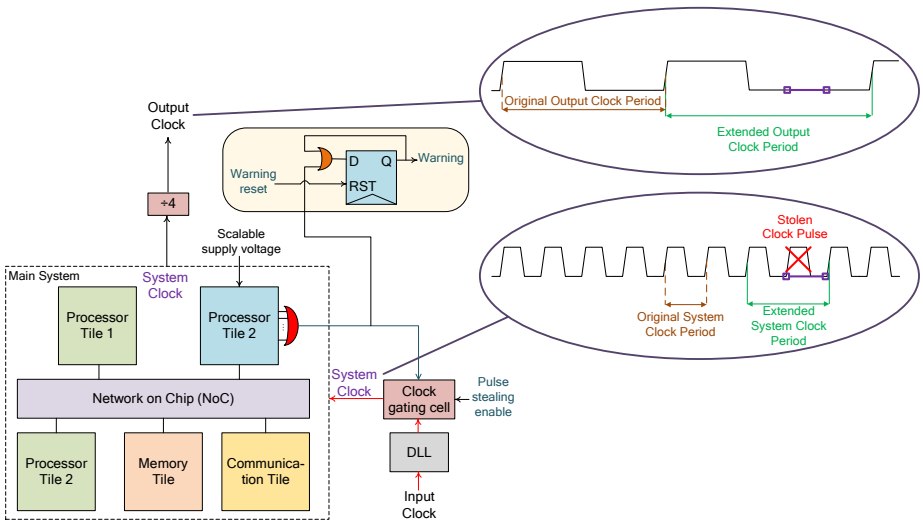



Figure 6.8: Architecture view to the taped out idea in a multi-processor platform generated from CompSOC flow.

mented technique is 4X higher than the one which is explained in Chapter 5. 256 in-situ delay monitors are inserted in the processor tile at the intermediate points along the paths which are reported as critical path based on their slacks. The design of the implemented in-situ delay monitor is similar to the previous designs, i.e., it is made up of a latch and an XOR gate. The outputs of the XOR gates of the monitors are collected with an OR tree into one bit, i.e., if at least one monitor is excited, the output of the OR tree goes high. The OR tree is synthesized separately with proper timing constraints considering the clock period and the criteria for monitor insertion points (i.e., $t_{mon}$ as introduced in Chapter 3). The delay of the OR tree is optimized such that its output goes high before the end of the clock cycle. During the implementation flow, the OR tree gates are preserved from optimization to make sure that the delay of the OR tree does not increase. As shown in Fig. 6.8, the output of the OR tree is connected to a clock gating cell which gates the upcoming clock edge. In this way, if any monitor is excited, the current clock cycle is doubled in order to avoid a timing error. The clock gating cell is inserted between the DLL and the internal clock input of the whole platform. Therefore, clock manipulation is applied to the clock signal of the complete platform, i.e., the gated clock goes in to the NoC, all processor tiles etc. Since the excitation of the monitors is triggered with the internal 200MHz system clock, and due to speed limitations of the IO cells, the output of the OR tree is stored if at least one monitor is excited in any cycle. In fact, if the output of the OR tree goes high, a logic one is stored in a flip-flop. The output of this flip-flop is wired out to the chip IO to be used as an indication of a timing

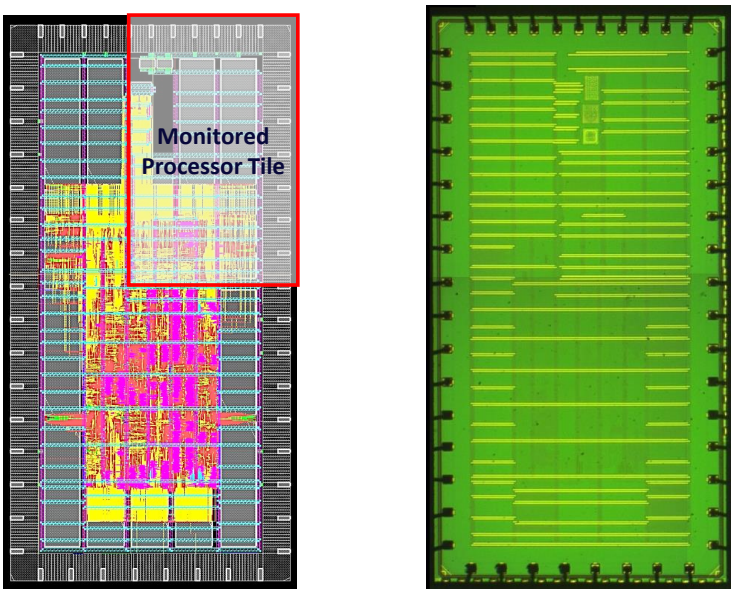



Figure 6.9: The layout and the die micro-graph of the second tape-out.

degradation occurrence. The reset pin of this flip-flop is also wired out to the chip IO to have control on its content. Similar to tape-out 1, the voltage of the monitored processor tile is made scalable. In this way, the circuit delays are increased by supply voltage scaling. Due to speed limitation of IOs, the system clock is down converted by X4 and wired out to chip IO (see "Output clock" in Fig. 6.8) to be able to observe the effect of the stolen clock pulses. The effect of clock pulse stealing on the output clock is that its period increases by 25% (or 50%, if two consecutive monitor excitations occur).

### 6.3.3 *Physical Implementation*

In Fig. 6.9, the layout and the die micro-graph of the second tape-out are shown. The design size is 1.9mm×3.8mm. The number of gates and IOs are ~815k and 51, respectively, and the target speed of the design is 200MHz. The DLL can multiply a 50MHz input clock frequency by X1 (bypass DLL), X2, X4, and X8. The monitored processor tile is physically separated from the rest of the platform (highlighted in Fig. 6.9), and its supply voltage is made scalable by separating its power rails from the rest of the platform and using the special analog type of voltage IO cells for it. ESD protection cells are inserted between these supply voltage IO cells and the main circuit.

In this tape-out, the implementation flow which is illustrated in Fig. 3.5 is followed for monitor insertion. First, the OR tree is synthesized with maximum delay given by

(5.1) in Chapter 3. Then, the gates of the 256 monitors (i.e., a latch and an XOR gate per monitor) as well as the gates of the synthesized OR tree are added to the RTL of the design. With preserving of the gates of the monitors and the OR tree, the design went through synthesis, floor-planning, and place and route. When timing closure is achieved, a long list of critical timing paths are reported, the insertion points are identified by parsing the timing report, and the monitors are connected to the identified insertion points. Based on the timing report, with 256 insertion points, 2772 paths (2317 unique end-points) are covered, and the maximum slack of the covered paths is 1.857ns ($Slk_{max} = 0.37 \times t_{clock}$). The insertion points are selected based on the arrival time criteria of 3ns, i.e., 0.5ns after the negative clock edge ($t_{mon} = 0.6 \times t_{clock}$). After connecting the monitors to the identified insertion points, the target slack of the monitors is set to -0.5ns, and design optimization is performed to minimize the loading effect of the monitors.

On the IO pins of the chip, there are three pins which are dedicated for the implemented technique:

- **DFS_E** enables the technique by enabling the corresponding clock gating cell ("Pulse stealing enable" in Fig. 6.8) and its internally inverted value connected to reset pin of the flip-flop which stores the output of the OR tree ("Warning reset" in Fig. 6.8),
- **DFS_O** is connected to the output of the flip-flop which stores the output of the OR tree ("Warning" in Fig. 6.8), and
- **CLK_G** is the down-converted internal clock signal of the system ("Output clock" in Fig. 6.8).

DFS_O pin goes high and stays high after monitor excitation until the capturing flip-flop is reset with the inverted DFS_E input pin.

### 6.3.4 *Experimental Results*

The evaluation board for this tape-out is shown in Fig. 6.10. Similar to the previous board, power and communication are through a USB B port. An FTDI chip emulates the UART communication and also allows access to the voltage regulators and current sensors. There are three current sensors in the design to measure the power of IO cells, the power of the processor tile with our implemented technique, and the power of the rest of the platform. In Table 6.2, the measured powers from each of the sensors are shown.

The monitored processor tile can work standalone by transferring the application and data to its memory through the communication pins (i.e., the UART pins). In our experiments, we program the monitored processor tile and degrade the delays to investigate the behavior of our technique. As mentioned before, the voltage of the monitored processor tile is scalable. The flow of running experiments is illustrated in Fig. 6.11. During the experiments, the test application with the input data are transferred to the platform

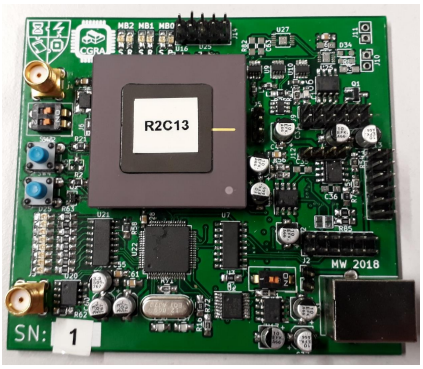



Figure 6.10: The evaluation board for tape-out 2.

Table 6.2: The power measurements for the second tape-out.

| | | |
|---|---|---|
| IO | 26mW | 15% |
| Core (Tile1, Tile3, DLL, memory and communication tiles) | 127mW | 73.4% |
| Tile 2 (processor tile with the timing resilience technique) | 20mW | 11.6% |
| Total | 174mW | 100% |

at the nominal voltage. The same application is executed on the same data on the PC to get the golden output data. Then, the supply voltage of the platform is scaled down to run the test application at the test voltage. To collect the output data from the platform, an application is transferred to the platform which reads memory content and sends the read data out through UART module to be read by PC for comparison. This application is transferred to the platform and runs at the nominal voltage. Finally, the output of the platform is compared with the golden output to check if timing error has occurred in the circuit at the test voltage.

Since the IC samples are in the typical corner, at the nominal voltage, no monitor excitation occurs, i.e., independent of DFS_E, the period of the output clock stays 20ns (i.e., 4X of internal clock period), and DFS_O stays zero. When the monitor excitation occurs, DFS_O goes high if the technique is activated by setting DFS_E to one. With monitor excitation the period of output clock also becomes 25ns (or 30ns if two internal clock period extensions happen consequently) occasionally. In Fig. 6.12, the captured waveforms of clock output and DFS_O are shown. To capture these waveforms, Digital IO inputs of an oscilloscope with proper signal acquisition settings are used. A periodic signal with a very long period (e.g. 250ms) is applied to DFS_E to activate the technique periodically and capture the effect delay degradation and monitor excitation on DFS_O and the output clock. To capture this effect, DFS_O is used as the triggering signal of

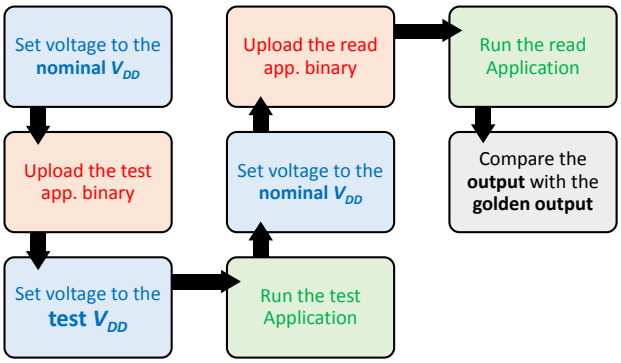



Figure 6.11: The flow of running experiments.

oscilloscope because its positive edge shows the moment when at least one monitor is excited and the output clock is extended. The illustrated waveforms in Fig. 6.12 are obtained at three different supply voltage values. As can be observed, lowering the supply voltage increases the number of times that the period of the output clock is extended (red circles in the figure). Therefore, the clock pulse stealing (i.e., monitor excitation) occurs when circuit delays degrade due to lowering the supply voltage value.

As the supply voltage is scaled down, circuit delays increase and at a specific voltage, $V_{warn}$, the clock pulse stealing starts (DFS_O=1) if the technique is activated (i.e., when DFS_E=1). Furthermore, without activating the technique (i.e., when DFS_E=0), the supply voltage can be scaled down to a specific voltage $V_{min}$ at which a timing violation occurs. Note that since the slacks of the monitors are less than the slacks of the main flip-flops, the clock pulse stealing starts at a voltage higher than the voltage which causes timing failure, i.e., $V_{warn} > V_{min}$. When the proposed technique is activated (DFS_E=1), the extensions of the clock period makes the circuit more resilient against timing degradation. Therefore, the supply voltage can be scaled down to a lower value $V_{min,R}$ when DFS_E=1, i.e., $V_{min,R} < V_{min}$ when DFS_E=1. In Fig. 6.13, $V_{min}$, $V_{min,R}$, and $V_{warn}$ are illustrated for five different samples of the design. As can be observed, $V_{warn} > V_{min} > V_{min,R}$ holds for all samples of the design. Due to the difference between the available slack margins of timing paths in different samples, the margins between $V_{min}$, $V_{min,R}$, and $V_{warn}$ vary. Note that the level noise on the supply voltage is less than 0.5mV.

The available slack margin can be converted to power reduction by scaling the supply voltage down to the minimum working voltage. In Fig. 6.14, the power consumption of the circuit is shown for the five samples of the IC at $V_{min}$, $V_{min,R}$, $V_{warn}$, and the nominal supply voltage value $V_{nom}$. Based on the results, compared to the nominal voltage, the power consumption of the circuit can be reduced by ~21% on average if the circuit works at $V_{warn}$. If the supply voltage is lowered to $V_{min}$ ($V_{min,R}$), the power saving increases to

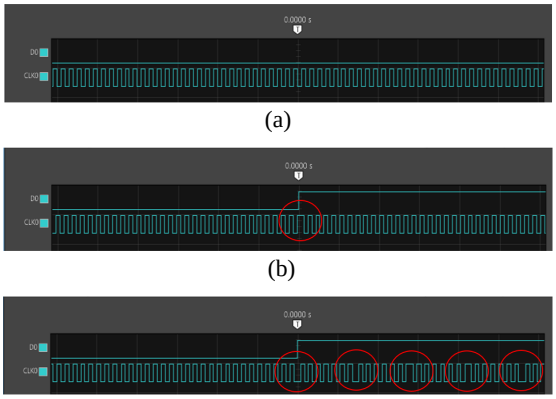
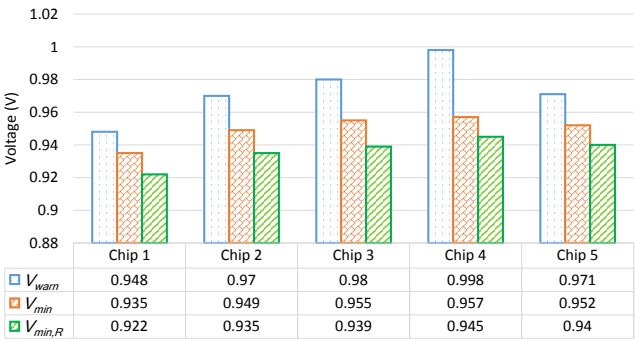



Figure 6.12: The effect of voltage scaling on the waveforms of the output clock of chip (with proper signal acquisition settings to digitize the waveforms): (a) at nominal voltage when no clock pulse stealing occurs, (b) at the voltage which the clock pulse stealing starts, and (c) at a lower voltage when clock pulses are stolen more often.




| | Chip 1 | Chip 2 | Chip 3 | Chip 4 | Chip 5 |
|---|---|---|---|---|---|
| $V_{warn}$ | 0.948 | 0.97 | 0.98 | 0.998 | 0.971 |
| $V_{min}$ | 0.935 | 0.949 | 0.955 | 0.957 | 0.952 |
| $V_{min,R}$ | 0.922 | 0.935 | 0.939 | 0.945 | 0.94 |

Figure 6.13: The effect of process variations on $V_{min}$, $V_{min,R}$, and $V_{warn}$.

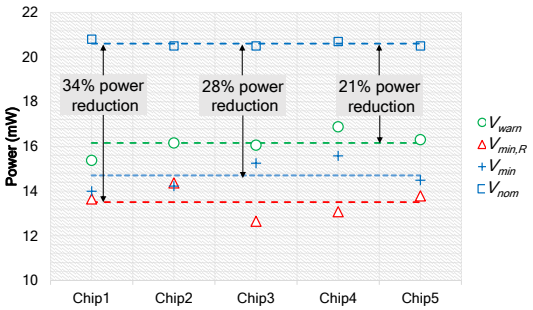



Figure 6.14: The effect of process variations on the power of the chip at $V_{min}$, $V_{min,R}$, $V_{warn}$, and $V_{nom}$.

~28% (~34%). However, it is safer to keep a margin between the operating voltage and the minimum supply voltage. A safe operating voltage for the circuit is $V_{warn}$ because at this voltage, the circuit works safely with a minimal distance with $V_{min}$. This safety margin is increased with the implemented TS system which lowers the minimum supply voltage further down to $V_{min,R}$. Note that the performance overhead of TS system is negligible when the circuit works at $V_{warn}$ and clock period is extended only if timing degradation happens.

In Fig. 6.15, the *Clock Pulse Stealing Rate (CPSR)* is plotted versus the supply voltage value for chip 1. The results are obtained by collecting multiple samples of output clock waveform at different supply voltage values and counting the number of times that the internal clock period is extended based on the waveform of the output clock. As expected, the CPSR is zero at higher voltages. It can be seen from the results that the CPSR starts to increase at 0.938 V, and it increases to ~25% at 0.935 V. Note that the circuit was operational during this experiment, i.e., timing errors are avoided by doubling the clock period dynamically. Therefore, based on Fig. 6.15, the performance overhead is up to 25% with the implemented technique when the supply voltage is scaled down. This relatively high rate of monitor excitation can be attributed to the fact that the monitors are pessimistically inserted in the design (i.e., their slacks less than the slacks of the main flip-flops), the wide inspection window of the monitors (i.e., the second half of clock cycle) which increases the chance of getting a transition which causes monitor excitation, and the insertion of monitors deep inside the paths which increases the rate of monitor excitation as discussed in Chapter 4.

Changing the application that is running on the platform can change the excited paths in the circuit. To investigate into this effect, $V_{min}$, $V_{min,R}$, $V_{warn}$, and the CPSR at $V_{DD} = V_{min}$ are measured considering four benchmark applications. The results of this experiment are shown in Table 6.3 for chip 1. The benchmark applications are *Infinite*

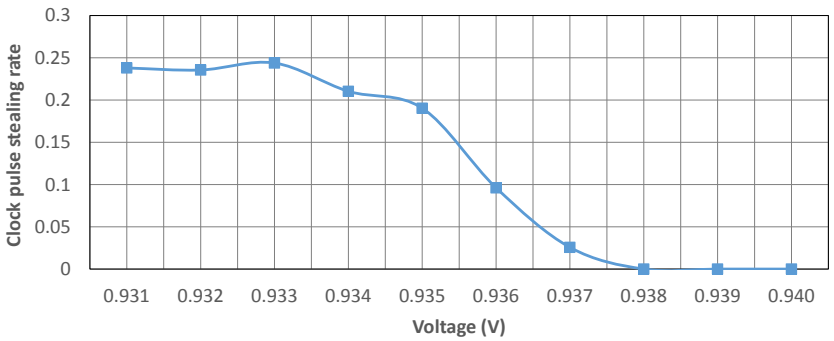



Figure 6.15: The CPSR of the first sample versus supply voltage value for chip 1.

Table 6.3: The effect of changing the running application on $V_{min}$, $V_{min,R}$, $V_{warn}$, and clock pulse stealing rate at $V_{DD} = V_{min}$ for chip 1.

| **Application** | **IIR filter** | **FIR filter** | **1D K-means** | **2D K-means** |
|---|---|---|---|---|
| $V_{min}$ (V) | 0.933 | 0.934 | 0.934 | 0.935 |
| $V_{min,R}$ (V) | 0.927 | 0.923 | 0.922 | 0.922 |
| $V_{warn}$ (V) | 0.948 | 0.948 | 0.948 | 0.948 |
| CPSR at $V_{DD} = V_{min}$ | 0.225 | 0.231 | 0.242 | 0.238 |

*Impulse Response (IIR)* filtering, *Finite Impulse Response (FIR)* filtering, 1 dimensional K-means classification, and 2 dimensional K-means classification. As mentioned before, the input data is generated randomly. Based on the result, there is variation on the measured voltages and the CPSR with changing the test application. Nevertheless, the functionality of the proposed technique is preserved independent of the running application.

As discussed in Chapter 1, temperature variation is a physical source of delay variation in the circuit. Hence, changing the temperature can change $V_{min}$, $V_{min,R}$, and $V_{warn}$ by affecting the slack margins in the circuit. To increase the temperature, we used the setup which is shown in Fig. 6.16. As shown in the figure, a Peltier element is placed on the chip to increase its temperature. The Peltier element can make a temperature difference between its two sides if enough current is applied. To set the temperature to a specific value, the applied current must be tuned accordingly. The temperature is sensed with a Thermocouple that is placed between the Peltier element and the chip. An Arduino platform reads the temperature and controls the current accordingly to reach to the target temperature. Since lowering the temperature is not possible with this setup, we used a climatic chamber for this purpose, as shown in Fig. 6.17. To investigate the effect of

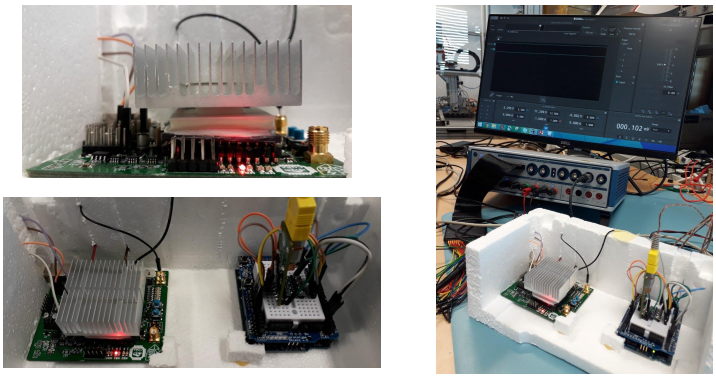


Figure 6.16: The setup to test the IC at high temperature.


Figure 6.17: The climatic chamber to decrease the temperature.

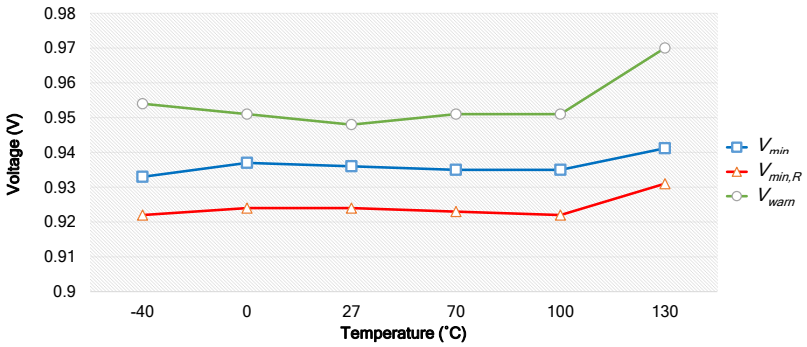



Figure 6.18: The effect of temperature on $V_{min}$, $V_{min,R}$, and $V_{warn}$ for chip 1.

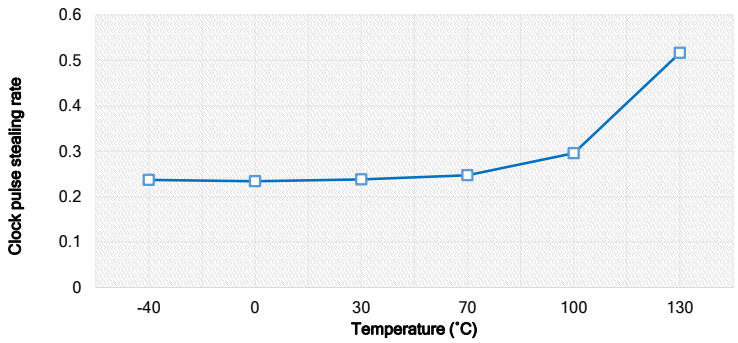



Figure 6.19: The effect of temperature on CPSR at $V_{min}$ as an indicator of timing degradation for chip 1.

temperature on the circuit timing, $V_{min}$, $V_{min,R}$, and $V_{warn}$ are measured at different temperatures, and the results are shown in Fig. 6.18. Based on the results, the delay variation due to temperature change is minimal when the temperature is lower than ~100°$C$. At 130°$C$, circuit delays degrade and the slack margins become lower. Consequently, $V_{min}$, $V_{min,R}$, and $V_{warn}$ increase at 130°$C$. As shown in the figure, the margin between $V_{min}$ and $V_{warn}$ increases when circuit delays increase at high temperature. That is because, as discussed in Chapter 4, MER increases when circuit delays degrade, i.e., more timing paths can excite each monitor. Since $V_{warn}$ is defined as the maximum voltage at which at least one monitor excitation occurs, by increasing the probability of monitor excitation, $V_{warn}$ increases more than $V_{min}$ and $V_{warn}$.

Similarly, the CPSR changes with temperature due the slack margin degradation. In Fig. 6.19, the CPSR is shown versus temperature for chip 1. As shown in the figure, the CPSR which indicates MER can reflect the temperature effect with a relatively high

Table 6.4: The effect of aging on $V_{min}$, $V_{min,R}$, $V_{warn}$ and clock pulse stealing rate for chip 1. The circuit is aged by working for 100ks (~28 hours) at $V_{DD} = 1.2V$ and $130°C$ temperature.

| | **Fresh** | **Aged** |
|---|---|---|
| $V_{min}$ | 0.935 | 0.937 |
| $V_{min,R}$ | 0.922 | 0.924 |
| $V_{warn}$ | 0.948 | 0.952 |
| CPSR at $V_{DD} = V_{min}$ | 0.238 | 0.267 |

sensitivity. The CPSR increases from 25% at the room temperature to ~30% at $100°C$. At $130°C$ the CPSR increases to more than 50% due to the significant delay degradation at this temperature. This effect is consistent with the results that are shown in Fig. 6.18. Therefore, the CPSR indicated the delay degradation with a high sensitivity.

To investigate the effect of silicon aging on the IC, accelerated aging test is performed by running an application at a high temperature and with a high supply voltage. As mentioned before, we can control the supply voltage value on-board. To control the temperature, we used the setup that is shown in Fig. 6.16. To age the circuit, the temperature is increased to $130°C$, the supply voltage is set to 1.2V, and all 4 applications run consequently in a while loop. The duration of this accelerated aging test was 100,000 seconds. Immediately after this test, $V_{min}$, $V_{min,R}$, $V_{warn}$, and CPSR at $V_{min}$ are measured. The results of this aging experiment are shown in Table 6.4 for chip 1. As show in the table, $V_{min}$ and $V_{min,R}$ increased by 2mV with aging and $V_{warn}$ increased by 3mV. Furthermore, due to slack degradation after this aging test, CPSR at $V_{min}$ increased from 23.8% to 26.7%. Although the effect of aging is small, but the monitoring technique can report it with a good sensitivity. Furthermore, the provided resiliency with clock pulse stealing avoids timing violation due to the aging effect.

### 6.3.5 *Discussions*

Although the chip health tracking technique which is implemented in the second tapeout is functional, the effectiveness of technique for timing resilience is low, i.e., the difference between $V_{min}$ and $V_{min,R}$ is marginal. This limited effectiveness is because some critical paths are excited while they are not monitored. Hence, the limited effectiveness must be related to the coverage of the implemented technique, and that depends on the identified set of insertion points. The insertion points of the monitors are selected based on the reported timing paths. All paths whose slacks are less than the target slack must be reported to make sure that there is no unmonitored path that can cause timing violation. The command "`report_timing`" in the timing analysis tool by default reports one critical timing path per endpoint. To include more paths per endpoint "`-nworst #`" should also be used to identify the number of paths which are reported per endpoint.

To make sure that all of the critical paths are reported, a large number must be specified for "`-nworst #`" which means path based timing analysis must be performed. Alternatively, the insertion algorithms that are introduced in Chapter 3 (not implemented in the tape-outs) guarantees that all critical paths are monitored. For this tape-out, one path per end point is protected because the "`-nworst #`" option of the command is not used. However, there are many more path per end point which must be protected. Note that although the missing paths are not the most critical path that end into their endpoints, but their slacks are still within the critical range. Since these paths are not monitored, clock pulse stealing does not happen if they are excited and timing error cannot be avoided. Therefore, it is essential for the insertion method to make sure that all paths within the target slack are monitored. Otherwise, a non-monitored path causes timing violation as its delay degradation is not masked with clock pulse stealing in the same cycle.

To summarize, the design issue with the second tape-out is that it does not cover all of the critical paths because of deficient usage of the command. This design issue has been solved in the proposed insertion flow in Chapter 3 (and in Code A.3 in Appendix A) to make sure that all paths whose slacks are within the intended range are monitored. This has been verified through simulations and the results (see Fig. 5.6), show that the proposed TS system is effective if the discussed design issue is solved.

## 6.4 SUMMARY

In this chapter, the silicon results of implementing the preliminary versions of the proposed ideas are discussed. In the first tape-out, within-path in-situ delay monitoring was implemented for chip health tracking in a single core microprocessor platform. However, the implemented technique was not functional. Connecting the clock pins of the monitors to the clock pins of the launching flip-flop was the design issue in the first tape-out. That is because when automatic clock gating is enabled during synthesis, the clock signal of monitors can be gated, and that causes wrong data to be stored in the monitors. In the second tape-out, this design issue was fixed. The implemented technique in the second tape-out was a preliminary version of the within-cycle error prevention technique. The technique was implemented in one of the three processor tiles of a multi-core microprocessor platform. The proposed technique was tested and it was functional. The monitors can warn if timing degradation occurs in the circuit. By using this warning signal and supply voltage scaling, ~21% power reduction could be achieved. A relatively limited range of timing resilience was also achieved. The effect of delay degradation on CPSR as an indication of MER was also investigated. Based on the experiments, we observed that MER can reflect the timing degradation in the circuit with a good sensitivity to delay degradation. Although the technique is functional, but its effectiveness for timing resilience is limited due to a design issue. The design issue was that not all the paths whose slacks are within the intended slack range are

monitored. This design issue has already been addressed in the insertion flow that was explained in Chapter 3.

# 7

# CONCLUSIONS AND FUTURE WORK

In this chapter, we first review the work that has been done in this thesis. Then, we present some prospective ideas to extend this work.

## 7.1 CONCLUSIONS

The use of electronic systems in safety-critical applications, as well as the challenges that we face in the advanced technology nodes, have made reliability a first-order concern in IC design. In a digital IC, reliable timing is vital as if timing error occurs, the functionality of the design is not preserved anymore. The goal of this thesis is to propose a reliable, cost-efficient, and predictive technique to monitor the circuit timing. The circuit timing can be monitored by sensing the physical source of timing unreliability such as process variations, voltage drop, temperature variations, silicon aging, etc. The effect of these physical sources on circuit timing is called delay fault and it can also be sensed by measuring the delays of logical gates and wires in the circuit. Alternatively, we can detect timing error, that is the propagation of erroneous data in the circuit due to delay fault, for chip health monitoring. After reviewing the state of the art, we found in-situ delay monitoring a technique that can be the base of a reliable, cost-efficient, and predictive chip health tracking. In this thesis, we proposed novel ideas to increase IC reliability with effective techniques for modeling, monitoring, and improving timing reliability.

In Chapter 2, first, the basic concepts of circuit timing and its unreliability were reviewed. A digital circuit usually implements a finite state machine with logical gates and memory elements. The memory elements capture and store the data in a circuit. The timing of the circuit is usually regulated with a clock signal. The clock signal goes to all memory elements and it determines the data capturing moments. Flip-flops are the most commonly used memory elements. The setup time and the hold time of flip-flops must not be violated to avoid timing error. To avoid setup (hold) time violation, the input data of a flip-flop must become stable long enough before (after) the capturing moment. Traditionally, setup time and hold time are characterized independently. In reality, setup time and hold time must be defined depending on the value of the counterpart constraint. We call this setup/hold-time interdependency. Taking this interdependence

into account increases the accuracy of timing analysis and in this regard, it is essential for the design of a reliable digital circuit. The setup/hold-time interdependency has already been observed and considered in other works. We proposed a new analytical model for the setup/hold-time interdependency that is more accurate, has less characterization effort and can be used in the design flow more effectively. The accuracy of the proposed model was ~10X higher, and the characterization effort of the model was ~2.5X less compared to the state of the art. We also proposed a new flow to employ the proposed model for more accurate timing analysis in IC design flow. The proposed flow was used in the design of an ARM Cortex M0 based microprocessor platform. With the proposed model, the pessimism of timing analysis was reduced that resulted in 1.26% power reduction. Another novel idea that was introduced in Chapter 2 was a new flow to capture the gradual timing degradation effects in the standard timing analysis tools. With the proposed flow, circuit-level stress factors were extracted and analytical models of aging were used to calculate the delay degradation factors of each gate in the circuit. Then, the calculated degradation factors were applied in the timing analysis by using derating factors. With the proposed flow, a timing report of the circuit that includes the effect of aging on the gate delays can be obtained. An ARM Cortex M0 processor was analyzed with the proposed flow. Based on this analysis, ~4.5% delay degradation was reported due to silicon aging effects.

Chapter 3 introduced a novel technique for in-situ delay monitoring. The new technique was based on monitoring the circuit delays at selective intermediate circuit nodes within critical timing paths. The proposed technique can be the base of a low-cost, in-situ, and predictive chip health tracking technique that has high coverage and can run in parallel to the main workload of the circuit. Furthermore, a new design flow was proposed to integrate the proposed technique in the circuit reliably. The design of each in-situ monitor was such that the cost per monitor was low. With inserting the monitors within critical timing paths, multiple paths were monitored with a single monitor. Therefore, for the same coverage of paths, the number of monitors was less with our technique compared to the state of the art. The monitor was inserted pessimistically by inserting them such that their slacks were less than the slacks of the main flip-flops of the design. In this way, the monitors become predictive of timing error, i.e., the monitors are excited when the slack margins are less but it is still enough for not having timing error in the main flip-flops. The correlation between monitoring information and the actual timing margin in the circuit was analyzed with *precision* and *recall* metrics. Provided that recall=1 (False-Negative is not acceptable), we showed that precision can be maximized with proposer selection of clock period and insertion point. Two techniques were proposed to find the insertion points reliably by guaranteeing the coverage of all critical paths. Since the insertion techniques do not rely on path-based timing analysis they were 3.4X to 152X faster, based on the results. We implemented our technique in an ARM Cortex M0 processor. Compared to end-point monitoring, our technique can reduce the number of required monitors by 23X. With 64 monitors, our technique achieves

better coverage and has lower power and area overheads compared to the other works that insert the monitors at the endpoints or close to the endpoints of the timing paths. Depending on the insertion point the guard band between monitor excitation and actual timing error can be controlled. We also implemented our technique in an ARM Cortex M3, which has a larger gate count. Based on the results, our technique can reduce the number of monitors by ~7.5X compared to the endpoint monitoring technique. Delay testing with LBIST is another approach for chip health tracking. Our technique as well as LBIST were implemented in two industrial designs. The results showed that with our technique the area, power and speed overheads were much lower while its coverage of critical paths was higher than LBIST. Besides, our technique can run in parallel to the main workload while LBIST has to stop the main workload to run the delay tests. Nevertheless, our technique can only be used for delay faults while LBIST is a more generic technique that can detect other faults too.

Chapter 4 proposed a probabilistic approach for in-situ delay monitoring. The conventional static in-situ delay monitoring techniques under-use the monitoring hardware while with our probabilistic approach, more monitoring data per in-situ delay monitor was extracted. The proposed approach was based on the fact that in general, more than one path is covered with each monitor and the slacks of these paths are normally very close to each other. If circuit delays degrade, more paths cause monitor excitation and the probability of monitor excitation (MER) increases. Based on this approach, we proposed a novel chip health tracking technique to get a fine-grained signature of chip health status. TDC techniques are the conventional approach to get such a fine-grained delay indication. However, monitoring all circuit slacks with TDC is very expensive, while with the proposed technique, we address this problem by extracting more information per monitor. The technique was made up of the within path inserted in-situ delay monitors that were connected in a scan chain and a signature extractor block that processes the monitoring information. The monitors are either in the monitoring phase or in the scan phase. During the monitoring phase, all late arrivals of data to the insertion points of the monitor are captured and stored in the monitors. During the scan phase, the stored monitoring information is scanned out. Then, MER is measured with the signature extractor block by analyzing the monitoring information. The proposed technique was implemented in an ARM Cortex M0 processor. With the proposed technique, a fine-grained signature is extracted that indicated the age of the circuit.

Chapter 5 presented a new TS system in which the clock signal was dynamically adjusted to avoid timing errors in the circuit. Since it is essential for the main workload to keep running while the chip health status is being monitored, the proposed TS system increases the timing resilience of the circuit to achieve this goal. The proposed TS system was based on within path inserted in-situ delay monitors to predict timing error within one clock cycle. The upcoming clock edge is then shifted to extend the clock period and avoid timing error due to the predicted late data arrival time. With this approach, the clock signal was manipulated globally to make sure that hold violation does not occur

due to unequal shifting of the clock edges at different flip-flops. By using the proposed TS system, up to 22% delay degradation was tolerated with a marginal energy overhead of less than 1% due to the additional run-time. The proposed TS system was also used to reduce the design costs by removing the unnecessary static margins during runtime while timing errors are avoided by adjusting the clock period dynamically. Supply voltage scaling was used to convert the static margin to power reduction during run-time. In this way, 40% power reduction was achieved compared to a design without our technique. Although the proposed TS system has performance overhead by extending the clock period, the achieved power saving with supply voltage scaling was so effective that overall 30% energy reduction can be achieved compared to a design without our technique.

Finally, Chapter 6 explained the results of implementing preliminary versions of the proposed chip health monitoring technique and TS system into real silicon. Two tape-outs have been done. In the first tape-out, a preliminary version of the chip health tracking system was implemented in a single-core ARM Cortex M0 based microprocessor platform. The platform was also replicated on the same chip without the technique to serve as a reference. The die size was 1.85mm×1.79mm, and the total number of gates and IOs were ~128k and 96, respectively, and the target speeds of the designs were 200MHz. The design of the monitors was similar to the one that was introduced in Chapter 4 while their insertion flow was different. Instead of adding the monitors at the synthesis level, the gates of the monitors were added to the design after the place and route step when timing closure was achieved. The clock pins of the monitors were connected to the clock pins of the launching flip-flops in the monitored timing paths. However, the automatic clock gating that was added to the circuit during synthesis was not compatible with this insertion flow. With automatic clock gating, the synthesis tool inserts clock gating cells in the clock path of some flip-flops of the design to reduce power consumption. Each clock gating cell is enabled based on the logical value of some specific circuit nodes. Therefore, the flip-flops have different clock signals when automatic clock gating is enabled. By connecting the clock pins of the monitors to the clock pins of the flip-flops, the latch inside the monitor remains opaque if the connected clock signal is gated. Since the data at the insertion of the monitor can change due to the activity of another timing path which is driven with a different clock signal that is not gated, the monitor stores a wrong value. After identifying this design issue proper modification was considered in the design methodology of the proposed chip health tracking technique. In the second tape-out, a multi-core microprocessor platform that includes 3 ARM Cortex M0 based processor tiles and a NoC was implemented. In one of the processor tiles, a preliminary version of the proposed TS system was implemented. The die size was 1.9mm×3.8mm. The number of gates and IOs were ~815k and 51, respectively, and the target speed of the design was 200MHz. In the implemented TS technique, the outputs of the within path inserted in-situ delay monitors were OR-ed to get a single bit that indicated the occurrence of timing degradation. This single bit was connected to

the enable input of a clock gating cell that is in the main clock path to control the clock signal of the whole chip. If any monitor is excited, the upcoming clock edge is gated to prevent timing error due to timing degradation, i.e., the clock period is doubled. Since the clock pulses were gated based on the output of the monitors, CPSR indicated the rate of monitor excitation, i.e., MER. The output of the OR-tree was also stored to be read from chip IOs as an indication of timing degradation. The monitors were inserted at the synthesis level together with a synthesized OR-tree with proper delay. After timing closure, the insertion points of the monitors were identified and connected to the monitors. Based on the timing report, with 256 monitors, 2772 paths (2317 unique endpoints) were monitored. Supply voltage scaling was employed to degrade circuit delays and test the functionality of the implemented techniques. The implemented techniques were tested and they were functional. As supply voltage scales down, at $V_{warn}$ the output of OR-tree goes as an indication of timing error. Without using the proposed clock pulse stealing technique, we can scale the supply voltage further down to $V_{min}$ without timing error. When the technique is activated, the supply voltage can be lowered further down to $V_{min,R}$ timing errors are avoided with extending the clock period. We measured $V_{warn}$, $V_{min}$, and $V_{min,R}$ for different samples of IC. On average, operating at $V_{warn}$, $V_{min}$, and $V_{min,R}$ results in 21%, 28%, and 34% power reduction compared to the nominal operating voltage, respectively. Operating at $V_{warn}$ is better because in addition to the power saving, enough slack margin is available to have timing resilience, and CPSR is minimal at this operating voltage (minimal performance overhead). As an indication of MER and performance overhead, CPSR was measured when the supply voltage was scaled down. Up to 25% performance overhead was observed when the voltage goes to lower than $V_{warn}$. The high sensitivity of CPSR in this range also shows that MER is a good measure of timing degradation. We also measured CPSR, $V_{warn}$, $V_{min}$, and $V_{min,R}$ when the running application on the platform changes. The results show that the variations of these parameters due to varying the applications were marginal. The effects of temperature variation on CPSR, $V_{warn}$, $V_{min}$, and $V_{min,R}$ were also measured. It was observed that at high temperatures, circuit delays degrade and this causes an increase in all of the mentioned parameters. Finally, we investigated the effect of aging on circuit delays by measuring CPSR, $V_{warn}$, $V_{min}$, and $V_{min,R}$ after running a test application on the circuit at high voltage and temperature. The results of this accelerated aging test showed that the marginal delay degradation that occurs can be tolerated and monitored well with the proposed TS system and chip health tracking technique, respectively. Although the implemented ideas were functional, the range of timing resiliency was limited. The design issue that causes this limited range was investigated. The issue is that only one path per endpoint was reported by the timing analysis tool during monitor insertion because of a deficient usage of the corresponding command. Therefore, all of the paths whose slacks were within the critical range were not covered. This design issue was addressed in the monitor insertion flow that was introduced in Chapter 3 of this thesis.

## 7.2 FUTURE WORK

Four concrete opportunities for extending the work that has been proposed in this thesis are explained in what follows.

**Combining The Proposed Idea with Delay Testing Technique**
As discussed in Chapter 3, LBIST is a technique with some advantages for online testing. However, the coverage of timing paths in LBIST is the main challenge for using that in delay testing. On the other hand, with the proposed within-path in-situ monitoring, the number of monitored paths is much higher while the second parts of the paths are not monitored. Not monitoring the second part of the paths limits the *precision* of monitoring, as discussed in Chapter 3. One can change the design of the monitors such that it can be used as test points for delay testing. In this way, the first part of the paths can be monitored with the in-situ monitors (high coverage), and the second part can be checked with LBIST approach. Since the depth of the second part of paths that must be tested with LBIST is much lower than the full path, the test effort (coverage) becomes much less (more). Therefore, by combining LBIST and the proposed within-path in-situ delay monitoring, both techniques can take advantage of each other to improve the coverage of chip health tracking.

**Transistor-Level Design for The In-Situ Monitors**
The design of the monitors can improve by using a more efficient transistor-level design for the same functionality. For instance, instead of the XOR gate and the latch inside the monitor, a transition detector can be used with a new transistor-level design. Overall, the number of transistors and their sizing can be optimized to achieve higher speed and lower power. An important challenge is characterizing the monitors such that they can be integrated into the design using the standard design flow. Defining the timing constraints, such as setup time, hold time, the minimum pulse width, and characterizing the power must be done to provide the design and analysis tools with proper views to the monitors. Also, having an HDL (e.g., Verilog or VHDL) model for the monitors is essential for functional simulation (required for verification) of the design with the monitors.

**Analytical Approach for Optimal Selection of Parameters in The Proposed Chip Health Tracking Technique**
In Chapter 3, we showed and discussed the effects of changing $t_{mon}$ on the coverage of critical paths, observability, *precision*, etc. However, we did not find the optimal value of $t_{mon}$, e.g., to maximize the coverage of critical paths, observability, and *precision*. Finding an analytical approach to optimize monitor insertion is a potential opportunity to improve the proposed within-path in-situ delay monitoring technique. Moreover, the design parameters of the chip health tracking technique that was proposed in Chapter 4 (see Table 4.1) were not selected optimally. We only discussed the effects of having different choices of these parameters on the output signature. An optimization tech-

nique that finds these design parameters optimally is another opportunity to extend this work.

**Signature Analysis with Machine Learning**

The chip health tracking technique that was introduced in Chapter 4 of this thesis outputs a signature of timing reliability. This signature can be further analyzed to extract more information about the effects of variations. Workload variations, process variations, environmental variations, and other variations effects can affect the signature. By using a machine learning technique, we can make the IC aware of its variability status such that it can adapt itself to the variation effects.

# BIBLIOGRAPHY


[1] J. Kauflin, "America's Top 50 Companies 1917-2017," https://www.forbes.com/sites/jeffkauflin/2017/09/19/americas-top-50-companies-1917-2017/, 2017, accessed: 2019-04-08. (Cited on pages xiii, 1, and 2.)

[2] V. Huard, S. Mhira, F. Cacho, and A. Bravaix, "Enabling robust automotive electronic components in advanced cmos nodes," *Microelectronics Reliability*, vol. 76-77, pp. 13 – 24, 2017. (Cited on pages xiii and 3.)

[3] E. Picardo, "Eight of the World's Top Companies are American," https://www.investopedia.com/articles/active-trading/111115/why-all-worlds-top-10-companies-are-american.asp, 2019, accessed: 2019-04-08. (Cited on page 1.)

[4] B. Lojek, *History of semiconductor engineering*. Springer, 2007. (Cited on page 1.)

[5] "SLT Circuits," https://www.ibm.com/ibm/history/exhibits/vintage/vintage_4506VV3081.html, accessed: 2019-04-08. (Cited on page 1.)

[6] "Intel Timeline: A History of Innovation," https://www.intel.com/content/www/us/en/history/historic-timeline.html, accessed: 2019-04-08. (Cited on page 1.)

[7] "Intel's First Microprocessor," https://www.intel.com/content/www/us/en/history/museum-story-of-intel-4004.html, accessed: 2019-04-08. (Cited on page 1.)

[8] R. Kurzweil, *The Singularity is Near: When Humans Transcend Biology*. Penguin, 2005. (Cited on page 1.)

[9] P. Teich, "Qualcomm Centriq Aims At Intel Xeon With Competitive Performance And Low Power," https://www.forbes.com/sites/tiriasresearch/2017/11/10/qualcomm-launches-into-server-market/, accessed: 2019-04-08. (Cited on page 1.)

[10] J. Hinshaw and P. Stearns, *Industrialization in the Modern World: From the Industrial Revolution to the Internet [2 volumes]: From the Industrial Revolution to the Internet*. ABC-CLIO, 2013. (Cited on page 2.)

[11] M. P. Groover, *Automation, Production Systems, and Computer-Integrated Manufacturing*, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2007. (Cited on page 2.)

[12] D. Harris and N. Weste, "CMOS VLSI Design," *Pearson Education, Inc*, 2010. (Cited on pages 2 and 6.)

[13] S. Kiamehr, "Cross-Layer Resiliency Modeling and Optimization: A Device to CircuitApproach," Ph.D. dissertation, Karlsruhe Institute of Technology, 2015. (Cited on page 2.)

[14] M. Stanisavljeví, A. Schmid, and Y. Leblebici, *Reliability of Nanoscale Circuits and Systems: Methodologies and Circuit Architectures.* Springer New York, 2011. (Cited on pages 2 and 4.)

[15] S. Asai, *VLSI Design and Test for Systems Dependability.* Springer Japan, 2018. (Cited on pages 2, 6, and 7.)

[16] L. H. Edmunds, "Advances in the heart-lung machine after john and mary gibbon," *The Annals of thoracic surgery*, vol. 76, no. 6, pp. S2220–S2223, 2003. (Cited on page 3.)

[17] J. B. Owen, L. R. Coia, and G. E. Hanks, "Recent patterns of growth in radiation therapy facilities in the united states: a patterns of care study report," *International Journal of Radiation Oncology* Biology* Physics*, vol. 24, no. 5, pp. 983–986, 1992. (Cited on page 3.)

[18] M. Lu, K. Wevers, and R. Van Der Heijden, "Technical feasibility of advanced driver assistance systems (adas) for road traffic safety," *Transportation Planning and Technology*, vol. 28, no. 3, pp. 167–187, 2005. (Cited on page 3.)

[19] R. Collinson, "Fly-by-wire flight control," *Computing & Control Engineering Journal*, vol. 10, no. 4, pp. 141–152, 1999. (Cited on page 3.)

[20] A. Mercha, W. Jeamsaksiri, J. Ramos, S. Jenei, S. Decoutere, D. Linten, and P. Wambacq, "Impact of scaling on analog/rf cmos performance," in *Proceedings. 7th International Conference on Solid-State and Integrated Circuits Technology, 2004.*, vol. 1, Oct 2004, pp. 147–152 vol.1. (Cited on page 4.)

[21] V. Castano and I. Schagaev, *Resilient Computer System Design.* Springer International Publishing, 2015. (Cited on page 4.)

[22] T. McConaghy, K. Breen, J. Dyck, and A. Gupta, *Variation-Aware Design of Custom Integrated Circuits: A Hands-on Field Guide.* Springer New York, 2012. (Cited on page 6.)

[23] H. A. Balef, M. Kamal, A. Afzali-Kusha, and M. Pedram, "All-region statistical model for delay variation based on log-skew-normal distribution," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 9, pp. 1503–1508, 2016. (Cited on pages 6 and 59.)

[24] R. Reis, G. Wirth, and Y. Cao, *Circuit design for reliability*. Springer New York, 1 2015. (Cited on pages 6, 7, 25, and 67.)

[25] E. Maricau and G. Gielen, *Analog IC Reliability in Nanometer CMOS*, ser. Analog Circuits and Signal Processing. Springer New York, 2013. (Cited on pages 7 and 23.)

[26] A. Bassi, A. Veggetti, L. Croce, and A. Bogliolo, "Measuring the effects of process variations on circuit performance by means of digitally-controllable ring oscillators," in *International Conference on Microelectronic Test Structures, 2003.* IEEE, 2003, pp. 214–217. (Cited on page 9.)

[27] S. Mitra, E. Volkerink, E. McCluskey, and S. Eichenberger, "Delay defect screening using process monitor structures," in *22nd IEEE VLSI Test Symposium, 2004. Proceedings.* IEEE, 2004, pp. 43–48. (Cited on page 9.)

[28] R. Rao, K. A. Jenkins, and J.-J. Kim, "A Completely Digital On-Chip Circuit for Local-Random-Variability Measurement," in *2008 IEEE International Solid-State Circuits Conference - Digest of Technical Papers.* IEEE, feb 2008, pp. 412–623. (Cited on page 9.)

[29] V. Melikyan, D. Mirzoyan, and G. Petrosyan, "A process variation detection method," in *2010 East-West Design & Test Symposium (EWDTS).* IEEE, sep 2010, pp. 30–33.

[30] A. Ghosh, R. M. Rao, J.-J. Kim, C.-T. Chuang, and R. B. Brown, "Slew-Rate Monitoring Circuit for On-Chip Process Variation Detection," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 9, pp. 1683–1692, sep 2013.

[31] T. J. Yamaguchi, J. S. Tandon, S. Komatsu, and K. Asada, "A novel test structure for measuring the threshold voltage variance in MOSFETs," in *2013 IEEE International Test Conference (ITC).* IEEE, sep 2013, pp. 1–8. (Cited on page 9.)

[32] P. Ituero, M. Lopez-Vallejo, H. Aparicio, and F. Garcia-Redondo, "Taxonomy of power supply monitors and integration challenges," in *2016 IEEE 21st International Mixed-Signal Testing Workshop (IMSTW).* IEEE, jul 2016, pp. 1–6. (Cited on page 9.)

[33] H. Aoki, M. Ikeda, and K. Asada, "On-chip voltage noise monitor for measuring voltage bounce in power supply lines using a digital tester," in *ICMTS 2000. Proceedings of the 2000 International Conference on Microelectronic Test Structures (Cat. No.00CH37095).* IEEE, 2000, pp. 112–117. (Cited on page 9.)

[34] H. Aparicio, P. Ituero, and M. Lopez-Vallejo, "2.64 pJ reference-free power supply monitor with a wide temperature range," in *2015 International Workshop on CMOS Variability (VARI).* IEEE, sep 2015, pp. 9–12. (Cited on page 9.)

[35] M. Sadi, Z. Conroy, B. Eklow, M. Kamm, N. Bidokhti, and M. M. Tehranipoor, "An All Digital Distributed Sensor Network Based Framework for Continuous Noise Monitoring and Timing Failure Analysis in SoCs," in *2014 IEEE 23rd Asian Test Symposium*. IEEE, nov 2014, pp. 269–274. (Cited on page 9.)

[36] X. Jiao, Y. Jiang, A. Rahimi, and R. K. Gupta, "WILD: A workload-based learning model to predict dynamic delay of functional units," in *2016 IEEE 34th International Conference on Computer Design (ICCD)*. IEEE, oct 2016, pp. 185–192. (Cited on page 9.)

[37] G. Quenot, N. Paris, and B. Zavidovique, "A temperature and voltage measurement cell for VLSI circuits," in *Euro ASIC '91*. IEEE Comput. Soc. Press, 1991, pp. 334–338. (Cited on page 10.)

[38] B. Datta and W. Burleson, "Low-power and robust on-chip thermal sensing using differential ring oscillators," in *2007 50th Midwest Symposium on Circuits and Systems*. IEEE, aug 2007, pp. 29–32.

[39] D. Ha, K. Woo, S. Meninger, T. Xanthopoulos, E. Crain, and D. Ham, "Time-Domain CMOS Temperature Sensors With Dual Delay-Locked Loops for Microprocessor Thermal Monitoring," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 9, pp. 1590–1601, sep 2012. (Cited on page 10.)

[40] V. Szekely, C. Marta, Z. Kohari, and M. Rencz, "CMOS sensors for on-line thermal monitoring of VLSI circuits," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 5, no. 3, pp. 270–276, sep 1997. (Cited on page 10.)

[41] B. Datta and W. Burleson, "A 12.4$\mu m^2$ 133.4$\mu$W 4.56mV/°C resolution digital on-chip thermal sensing circuit in 45nm CMOS utilizing sub-threshold operation," in *2011 12th International Symposium on Quality Electronic Design*. IEEE, mar 2011, pp. 1–7.

[42] M. Bashir, S. R. Patri, and K. S. R. Krishnaprasad, "On-chip CMOS temperature sensor with current calibrated accuracy of -1.1°C to +1.4°C (3$\sigma$) from -20°C to 150°C," in *2015 19th International Symposium on VLSI Design and Test*. IEEE, jun 2015, pp. 1–5. (Cited on page 10.)

[43] E. Mintarno, V. Chandra, D. Pietromonaco, R. Aitken, and R. W. Dutton, "Workload dependent NBTI and PBTI analysis for a sub-45nm commercial microprocessor," in *2013 IEEE International Reliability Physics Symposium (IRPS)*. IEEE, apr 2013, pp. 3A.1.1–3A.1.6. (Cited on page 10.)

[44] Abhishek Koneru, Arunkumar Vijayan, Krishnendu Chakrabarty and M. B. Tahoori, "Fine-Grained Aging Prediction Based on the Monitoring of Run-Time Stress Using DfT Infrastructure," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*. IEEE Press, 2015, pp. 51–58.

[45] A. Vijayan, A. Koneru, S. Kiamehr, K. Chakrabarty, and M. B. Tahoori, "Fine-Grained Aging-Induced Delay Prediction Based on the Monitoring of Run-Time Stress," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 5, pp. 1064–1075, may 2018.

[46] K. Huang, X. Zhang, and N. Karimi, "Real-Time Prediction for IC Aging Based on Machine Learning," *IEEE Transactions on Instrumentation and Measurement*, pp. 1–9, 2019. (Cited on page 10.)

[47] Y. Wang, M. Enachescu, S. Cotofana, and L. Fang, "Variation tolerant on-chip degradation sensors for dynamic reliability management systems," *Microelectronics Reliability*, vol. 52, no. 9-10, pp. 1787–1791, sep 2012. (Cited on page 10.)

[48] D. Sengupta and S. S. Sapatnekar, "Estimating Circuit Aging Due to BTI and HCI Using Ring-Oscillator-Based Sensors," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 10, pp. 1688–1701, oct 2017. (Cited on page 10.)

[49] M. Elgebaly and M. Sachdev, "Efficient adaptive voltage scaling system through on-chip critical path emulation," in *Proceedings of the 2004 International Symposium on Low Power Electronics and Design (IEEE Cat. No.04TH8758)*. Newport Beach, CA, USA: IEEE, 2004, pp. 375–380. (Cited on page 11.)

[50] A. Drake, R. Senger, H. Deogun, G. Carpenter, S. Ghiasi, T. Nguyen, N. James, M. Floyd, and V. Pokala, "A Distributed Critical-Path Timing Monitor for a 65nm High-Performance Microprocessor," in *2007 IEEE International Solid-State Circuits Conference. Digest of Technical Papers*. IEEE, feb 2007, pp. 398–399.

[51] James Tschanz, Keith 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 *2009 Symposium on VLSI Circuits*. Kyoto, Japan: IEEE, 2009, pp. 112–113. (Cited on page 11.)

[52] F. Firouzi, F. Ye, K. Chakrabarty, and M. B. Tahoori, "Representative critical-path selection for aging-induced delay monitoring," in *2013 IEEE International Test Conference (ITC)*. IEEE, sep 2013, pp. 1–10. (Cited on pages 11 and 12.)

[53] M. Bushnell and V. Agrawal, *Essentials of electronic testing for digital, memory and mixed-signal VLSI circuits*. Springer Science & Business Media, 2004, vol. 17. (Cited on page 11.)

[54] S. Pilarski and A. Pierzynska, "BIST and delay fault detection," in *Proceedings of IEEE International Test Conference - (ITC)*. IEEE, 1993, pp. 236–242. (Cited on page 11.)

[55] A. Vuksic and K. Fuchs, "A new BIST approach for delay fault testing," in *Proceedings of European Design and Test Conference EDAC-ETC-EUROASIC*. IEEE Comput. Soc. Press, 1994, pp. 284–288.

[56] B. Wurth and K. Fuchs, "A BIST approach to delay fault testing with reduced test length," in *Proceedings the European Design and Test Conference. ED&TC 1995*. IEEE Comput. Soc. Press, 1995, pp. 418–423.

[57] Chih-Ang Chen and S. Gupta, "Design of efficient BIST test pattern generators for delay testing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 12, pp. 1568–1575, 1996.

[58] Xiaowei Li and P. Cheung, "An effective BIST scheme for delay testing," in *ISCAS '98. Proceedings of the 1998 IEEE International Symposium on Circuits and Systems (Cat. No.98CH36187)*, vol. 2. IEEE, 2002, pp. 288–291.

[59] Lihong Tong, K. Suzuki, and H. Ito, "Optimal seed generation for delay fault detection BIST," in *Proceedings of the 11th Asian Test Symposium, 2002. (ATS '02)*. IEEE Comput. Soc, 2003, pp. 116–121.

[60] E. K. Moghaddam and S. Hessabi, "An On-Line BIST Technique for Delay Fault Detection in CMOS Circuits," in *16th Asian Test Symposium (ATS 2007)*. IEEE, oct 2007, pp. 73–78.

[61] 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 *2014 International Test Conference*. IEEE, oct 2014, pp. 1–8. (Cited on page 11.)

[62] M. Sadi, G. K. Contreras, J. Chen, L. Winemberg, and M. Tehranipoor, "Design of Reliable SoCs With BIST Hardware and Machine Learning," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 11, pp. 3237–3250, nov 2017. (Cited on pages 11, 12, and 60.)

[63] T. Sato and Y. Kunitake, "A simple flip-flop circuit for typical-case designs for dfm," in *8th International Symposium on Quality Electronic Design (ISQED'07)*. IEEE, 2007, pp. 539–544. (Cited on page 12.)

[64] T. Sato, M. Hashimoto, S. Tanakamaru, K. Takeuchi, Y. Sato, S. Kajihara, M. Yoshimoto, J. Jung, Y. Kimi, H. Kawaguchi *et al.*, "Time-dependent degradation in device characteristics and countermeasures by design," in *VLSI Design and Test for Systems Dependability*. Springer, 2019, pp. 203–243. (Cited on page 12.)

[65] Y. Sato and S. Kajihara, "A stochastic model for nbti-induced lsi degradation in field," in *2013 22nd Asian Test Symposium*. IEEE, 2013, pp. 183–188. (Cited on page 12.)

[66] M. Nicolaidis, "Time redundancy based soft-error tolerance to rescue nanometer technologies," in *Proceedings 17th IEEE VLSI Test Symposium (Cat. No.PR00146)*. IEEE Comput. Soc, 1999, pp. 86–94. (Cited on pages 12, 40, and 60.)

[67] D. Ernst, N. Sung Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, and T. Mudge, *Razor: A Low-Power Pipeline Based on Circuit-Level Timing Speculation Dan.* San Diego, CA, USA: IEEE Computer Society, 2003. (Cited on pages 12 and 72.)

[68] K. A. Bowman, J. W. Tschanz, Nam Sung Kim, J. C. Lee, C. B. Wilkerson, S.-L. L. Lu, T. Karnik, and V. K. De, "Energy-efficient and metastability-immune timing-error detection and recovery circuits for dynamic variation tolerance," in *2008 IEEE International Conference on Integrated Circuit Design and Technology and Tutorial*. IEEE, jun 2008, pp. 155–158. (Cited on page 13.)

[69] S. Das, C. Tokunaga, S. Pant, W. H. Ma, S. Kalaiselvan, K. Lai, D. M. Bull, and D. T. Blaauw, "RazorII: In situ error detection and correction for PVT and ser tolerance," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 1, pp. 32–48, 2009. (Cited on pages 40 and 72.)

[70] M. Choudhury, V. Chandra, K. Mohanram, and R. Aitken, "TIMBER: Time borrowing and error relaying for online timing error resilience," in *2010 Design, Automation & Test in Europe Conference & Exhibition (DATE 2010)*. IEEE, mar 2010, pp. 1554–1559.

[71] I. Kwon, S. Kim, D. Fick, M. Kim, Y.-P. Chen, and D. Sylvester, "Razor-Lite: A Light-Weight Register for Error Detection by Observing Virtual Supply Rails," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 9, pp. 2054–2066, sep 2014. (Cited on page 72.)

[72] Liangzhen Lai, V. Chandra, R. C. Aitken, and P. Gupta, "SlackProbe: A Flexible and Efficient In Situ Timing Slack Monitoring Methodology," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 8, pp. 1168–1179, aug 2014. (Cited on pages 40, 45, and 60.)

[73] Y. Zhang, M. Khayatzadeh, K. Yang, M. Saligane, N. Pinckney, M. Alioto, D. Blaauw, and D. Sylvester, "iRazor: Current-Based Error Detection and Correction Scheme for PVT Variation in 40-nm ARM Cortex-R4 Processor," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 2, pp. 619–631, feb 2018. (Cited on page 13.)

[74] S. Srivastava and J. Roychowdhury, "Independent and interdependent latch setup/hold time characterization via Newton-Raphson solution and Euler curve tracking of state-transition equations," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 5, pp. 817–830, 2008. (Cited on pages 20, 26, and 35.)

[75] E. Salman and E. Friedman, "Utilizing interdependent timing constraints to enhance robustness in synchronous circuits," *Microelectronics Journal*, vol. 43, no. 2, pp. 119–127, 2012. (Cited on pages 26 and 35.)

[76] A. B. Kahng and H. Lee, "Timing Margin Recovery With Flexible Flip-Flop Timing Model," *15th International Symposium on Quality Electronic Design*, no. i, pp. 496 – 503, 2014. (Cited on pages 20, 26, and 27.)

[77] M. A. Alam, "A critical examination of the mechanics of dynamic nbti for pmosfets," in *IEEE International Electron Devices Meeting 2003.* IEEE, 2003, pp. 14–4. (Cited on page 22.)

[78] J. Fang and S. S. Sapatnekar, "The impact of hot carriers on timing in large circuits," *Proceedings of the Asia and South Pacific Design Automation Conference, ASP-DAC*, pp. 591–596, 2012. (Cited on page 22.)

[79] J. B. Velamala, K. B. Sutaria, T. Sato, and Y. Cao, "Aging statistics based on trapping/detrapping: Silicon evidence, modeling and long-term prediction," in *2012 IEEE International Reliability Physics Symposium (IRPS).* IEEE, 2012, pp. 2F–2. (Cited on page 22.)

[80] W. McMahon, A. Haggag, and K. Hess, "Reliability scaling issues for nanoscale devices," *IEEE Transactions on Nanotechnology*, vol. 2, no. 1, pp. 33–38, 2003. (Cited on page 23.)

[81] T. Sakurai and A. R. Newton, "Alpha-power law mosfet model and its applications to cmos inverter delay and other formulas," *IEEE Journal of solid-state circuits*, vol. 25, no. 2, pp. 584–594, 1990. (Cited on page 25.)

[82] S. Keller, D. M. Harris, and A. J. Martin, "A compact transregional model for digital cmos circuits operating near threshold," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 10, pp. 2041–2053, 2014. (Cited on page 25.)

[83] A. Dasdan, S. Kolay, and M. Yazgan, "Derating for static timing analysis: Theory and practice," in *2009 10th International Symposium on Quality Electronic Design.* IEEE, 2009, pp. 719–727. (Cited on page 25.)

[84] E. Salman, E. G. Friedman, A. Dasdan, F. Taraporevala, and K. Kucukcakar, "Pessimism reduction in static timing analysis using interdependent setup and hold times," in *Proceedings of the 7th International Symposium on Quality Electronic Design.* IEEE Computer Society, 2006, pp. 159–164. (Cited on pages 26 and 29.)

[85] N. Chen, B. Li, and U. Schlichtmann, "Iterative timing analysis based on nonlinear and interdependent flipflop modelling," *IET Circuits, Devices & Systems*, vol. 6, no. 5, pp. 330–337, 2012. (Cited on page 26.)

[86] H. Seo, J. Heo, and T. Kim, "Clock skew optimization for maximizing time margin by utilizing flexible flip-flop timing," in *Sixteenth International Symposium on Quality Electronic Design*. IEEE, 2015, pp. 35–39. (Cited on page 26.)

[87] S. Hatami, H. Abrishami, and M. Pedram, "Statistical timing analysis of flip-flops considering codependent setup and hold times," in *Proceedings of the 18th ACM Great Lakes symposium on VLSI*. ACM, 2008, pp. 101–106. (Cited on pages 26 and 35.)

[88] S. V. Kumar, C. V. Kashyap, and S. S. Sapatnekar, "A framework for block-based timing sensitivity analysis," in *2008 45th ACM/IEEE Design Automation Conference*. IEEE, 2008, pp. 688–693. (Cited on page 40.)

[89] D. Fick, N. Liu, Z. Foo, M. Fojtik, J.-s. Seo, D. Sylvester, and D. Blaauw, "In situ delay-slack monitor for high-performance processors using an all-digital self-calibrating 5ps resolution time-to-digital converter," in *2010 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE, 2010, pp. 188–189. (Cited on page 40.)

[90] V. Mahalingam, N. Ranganathan, and R. Hyman Jr, "Dynamic clock stretching for variation compensation in vlsi circuit design," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 8, no. 3, p. 16, 2012. (Cited on page 41.)

[91] M. Nejat, B. Alizadeh, and A. Afzali-Kusha, "Dynamic flip-flop conversion: A time-borrowing method for performance improvement of low-power digital circuits prone to variations," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 11, pp. 2724–2727, 2014. (Cited on pages 41 and 72.)

[92] H. Cherupalli and J. Sartori, "Graph-based dynamic analysis: Efficient characterization of dynamic timing and activity distributions," in *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2015, pp. 729–735. (Cited on page 41.)

[93] *Tempus User Guide*, Cadence Design Systems, 2019. (Cited on pages 43 and 44.)

[94] S. M. Londoño and J. P. de Gyvez, "A better-than-worst-case circuit design methodology using timing-error speculation and frequency adaptation," in *2012 IEEE International SOC Conference*. IEEE, 2012, pp. 15–20. (Cited on pages 45 and 72.)

[95] N. Japkowicz and M. Shah, *Evaluating learning algorithms: a classification perspective*. Cambridge University Press, 2011. (Cited on page 48.)

[96] A. Halim, "Logic Built-In Self-Test and Midway Delay Fault Sensor for In-field Digital Circuit Wear-out Faults Detection," Master's thesis, Eindhoven University of Technology, 2019. (Cited on page 55.)

[97] L. Lai, V. Chandra, R. Aitken, and P. Gupta, "Slackprobe: A low overhead in situ on-line timing slack monitoring methodology," in *Proceedings of the Conference on Design, Automation and Test in Europe*. EDA Consortium, 2013, pp. 282–287. (Cited on page 59.)

[98] H. Ahmadi Balef, H. Fatemi, K. Goossens, and J. Pineda de Gyvez, "Effective in-situ chip health monitoring with selective monitor insertion along timing paths," in *Proceedings of the 2018 on Great Lakes Symposium on VLSI*. ACM, 2018, pp. 213–218. (Cited on pages 59 and 60.)

[99] M. Sadi, L. Winemberg, and M. Tehranipoor, "A robust digital sensor ip and sensor insertion flow for in-situ path timing slack monitoring in socs," in *2015 IEEE 33rd VLSI Test Symposium (VTS)*. IEEE, 2015, pp. 1–6. (Cited on page 59.)

[100] F. Cacho, A. Benhassain, R. Shah, S. Mhira, V. Huard, and L. Anghel, "Investigation of critical path selection for in-situ monitors insertion," in *2017 IEEE 23rd International Symposium on On-Line Testing and Robust System Design (IOLTS)*. IEEE, 2017, pp. 247–252. (Cited on page 63.)

[101] L. Wan and D. Chen, "Dynatune: circuit-level optimization for timing speculation considering dynamic path behavior," in *Proceedings of the 2009 International Conference on Computer-Aided Design*. ACM, 2009, pp. 172–179. (Cited on page 72.)

[102] M. Fojtik, D. Fick, Y. Kim, N. Pinckney, D. Harris, D. Blaauw, and D. Sylvester, "Bubble razor: An architecture-independent approach to timing-error detection and correction," in *2012 IEEE International Solid-State Circuits Conference*. IEEE, 2012, pp. 488–490. (Cited on page 72.)

[103] K. Chae and S. Mukhopadhyay, "A dynamic timing error prevention technique in pipelines with time borrowing and clock stretching," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 1, pp. 74–83, 2013. (Cited on page 72.)

[104] S. Guntur, G. Al-Kadi, R. I. M. P. Meijer, J. Hoogerbrugge, and H. Fatemi, "Clock selection circuit and method," Sep. 16 2014, uS Patent 8,836,379. (Cited on page 72.)

[105] W. Shan, L. Wan, X. Liu, X. Shang, W. Dai, S. Shao, J. Yang, and L. Shi, "A low overhead, within-a-cycle adaptive clock stretching circuit with wide operating range in 40-nm cmos," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 65, no. 11, pp. 1718–1722, 2018. (Cited on pages 72, 73, and 74.)

[106] J. Zhou, X. Liu, Y.-H. Lam, C. Wang, K.-H. Chang, J. Lan, and M. Je, "Hepp: A new in-situ timing-error prediction and prevention technique for variation-tolerant ultra-low-voltage designs," in *2013 IEEE Asian Solid-State Circuits Conference (A-SSCC)*. IEEE, 2013, pp. 129–132. (Cited on page 73.)

[107] S. Goossens, B. Akesson, M. Koedam, A. B. Nejad, A. Nelson, and K. Goossens, "The compsoc design flow for virtual execution platforms," in *Proceedings of the 10th FPGAworld conference*. ACM, 2013, p. 7. (Cited on page 81.)

[108] K. Goossens, M. Koedam, A. Nelson, S. Sinha, S. Goossens, Y. Li, G. Breaban, R. van Kampenhout, R. Tavakoli, J. Valencia *et al.*, "Noc-based multiprocessor architecture for mixed-time-criticality applications," *Handbook of hardware/software codesign*, pp. 1–40, 2017. (Cited on pages 81 and 82.)

[109] O. A. R. Rosas, "Multi-bit Pulse-based Latches for Low Power Design," Master's thesis, Eindhoven University of Technology, 2016. (Cited on page 86.)

[110] R. A. Stefan, A. Molnos, and K. Goossens, "daelite: A tdm noc supporting qos, multicast, and fast connection set-up," *IEEE Transactions on Computers*, vol. 63, no. 3, pp. 583–594, 2012. (Cited on page 91.)

[111] M. Wijtvliet, J. Huisken, L. Waeijen, and H. Corporaal, "Blocks: Redesigning coarse grained reconfigurable architectures for energy efficiency," in *Proceedings of 29th International Conference on Field Programmable Logic and Applications (FPL19)*. Springer, 2019, p. 7. (Cited on page 91.)

# A

# CODE LISTINGS

Listing A.1: Convert TCF to Aging Derating Factors (Python)

```
import numpy as np

lib_v_file = './lib.v' # Verilog model of library cells
v_file = './postMON.v' # Verilog netlist of the circuit
tcf_file = './cortexm0ds.tcf' # The activity file

# Make a dictionary of the input pins of each library cell
cell_input_dict = dict()
in_module = False
with open(lib_v_file,'r') as f:
  for line in f:
    ls = line.strip().split()
    if len(ls)>1:
      if ls[0]=='module':
        module_name = ls[1]
        cell_input_dict[module_name] = []
        in_module = True
      elif in_module and ls[0]=='input':
        input_pins_str = line.strip().split('input')[1]
        input_pins = [inp.replace(';','').strip() for inp in input_pins_str.split(',
            ')]
        cell_input_dict[module_name] = cell_input_dict[module_name]+input_pins

# Make dictionary of the input pins of each cell instance in the circuit
inst_input_dict = dict()
with open(v_file,'r') as f:
  for line in f:
    ls = line.split()
    if len(ls)>1:
      if ls[0] in cell_input_dict.keys():
        inst_input_dict[ls[1]]= cell_input_dict[ls[0]]

# Make a dictionary of the activity of instance inputs
inst_pob1_dict = dict()
with open(tcf_file,'r') as f:
  for line in f:
    ls = line.strip().split()
    if '"' in ls[0]:
      ls2 = line.strip().split('"')
      instance_pin = ls2[1].split('/')
      if len(instance_pin)>1:
        instance = instance_pin[-2]
        pin = instance_pin[-1]
        pob1 = float(ls2[3].split()[0])
```

```
        if instance in inst_input_dict.keys():
          if pin in inst_input_dict[instance]:
            if not(instance in inst_pob1_dict.keys()):
              inst_pob1_dict[instance] = [pob1]
            else:
              inst_pob1_dict[instance].append(pob1)

# consider the average of input activities for each instance
for k in inst_pob1_dict.keys():
  inst_pob1_dict[k] = sum(inst_pob1_dict[k])*1.0/len(inst_pob1_dict[k])

# calculate the derating factor based on the activity
def calc_del_deg(alpha,dvt_perc):
  vth = 0.4
  K = 150e-12
  VDD = 1.1
  D0 = K*VDD/np.power(VDD-vth,1.5)
  dvth_au = 0.5-np.power(1.5*(alpha-0.5),3) # From Fig. 6.15 of "Circuit Design for
       Reliability"
  dvth_max = dvt_perc*vth/100;
  vth_aged = vth+dvth_max*dvth_au
  D_aged = K*VDD/np.power(VDD-vth_aged,1.5)
  deg_fac = D_aged/D0;
  return deg_fac


# Degradation over 10 years of constant stress
ages = range(11)
dvt_perc = [0, 3.7583, 5.3753, 6.4202, 7.1937, 7.8081, 8.3178, 8.7533, 9.1336,
     9.4710, 9.7743]

# Calculate degradation for each cell after each year
#(ten different derating application files each for a specific year)
for age in ages:
  inst_degfac_dict = dict()
  for k in inst_pob1_dict.keys():
    inst_degfac_dict[k] = calc_del_deg(inst_pob1_dict[k], dvt_perc[age])
  age_tcl_file = './sta/age_derate_'+str(age)+'.tcl'
  with open(age_tcl_file,'w+') as f:
    for k in inst_degfac_dict.keys():
      f.write('set_timing_derate %f [dbGet top.insts.name *%s*] -late\n' % (
           inst_degfac_dict[k],k))
```

Listing A.2: Find Insertion Points of Monitors (TCL)

```
# A procedure to find the net to connect the monitor
# based on the reported timing path and the maximum
# arrival time to the monitor insertion point
proc findInp { rpt tmon } {
  set path [lindex [lindex [lindex [lindex  $rpt 1] 1] 7] 1]
  if {$path!=""} then {
    set path_len [llength $path]
    set idx 1
    set  arrival_prv 0
    while {$idx<$path_len} {
      set timing_row [lindex [lindex $path $idx] 1]
      set timing_pnt [lindex [lindex $timing_row 0] 0]
      set arrival [lindex [lindex $timing_row 1] 0]
      if { $idx==1} then {
        set clk_arrival $arrival
```

```
      } else {
        set arrival [expr $arrival - $clk_arrival]
      }
      if { $arrival>=$tmon && $arrival_prv<$tmon } then {
        set inp $timing_pnt
        return [list $inp $arrival]
      }
      set  arrival_prv $arrival
      incr idx
    }
  }
}
```

Listing A.3: Monitor Insertion (TCL)

```
# The inputs of this script are the clock period,
# slk_max, and the intended tmon
set cp $clk_period
set sr $slack_max_ratio
set tm_l $tmon_low
set tmon [expr $tm_l*$cp]

#Restore design
setMultiCpuUsage -localCpu 24 -cpuPerRemoteHost 1 -remoteHost 0 -keepLicense true
setDistributeHost -local
restoreDesign ./DBS/postRoute.enc.dat CORTEXM0DS
timeDesign -postRoute
set_interactive_constraint_modes [all_constraint_modes -active]

# define empty sets and get the number of added
# monitors to the design
set inp_set {}
set arv_set {}
set num_mon [llength [lsort -unique [dbGet top.insts.hInst.name DFS*]]]

# Find the monitor insertion points
set cnt 1
while { $cnt<=$num_mon } {
  if { $inp_set == {} } {
    set rpt [report_timing -format { timing_point arrival} -to [all_registers -flops
        ] -tcl_list -from [all_registers -flops]]
  } else {
    set rpt [report_timing -format { timing_point arrival} -to [all_registers -flops
        ] -tcl_list -not_through $inp_set -from [all_registers -flops]]
  }
  # Find the insrtion point with above procedure
  set inp_arv [findInp $rpt $tmon]
  set inp [lindex $inp_arv 0]
  set arv [lindex $inp_arv 1]
  lappend inp_set $inp
  lappend arv_set $arv
  incr cnt
}

# Connect the monitors to the insertion points
set cnt 1
while { $cnt<=$num_mon }  {
  set i [expr $cnt-1]
  set inp [lindex $inp_set $i]
  set arv [lindex $arv_set $i]
```

```
  # A procedure which connects a specific monitor to a
  # specific insertion point
  connect_to_inp $cnt $inp $arv
  incr cnt
}

# Physical implementation and optimization
ecoPlace
ecoRoute
optDesign -postRoute -setup

# Save design
rm -rf ./DBS/postMON.*
saveDesign ./DBS/postMON.enc
saveNetlist ./postMON.v
rcOut -spef ./postMON.spef -rc_corner rcworst

# Verification: calculate the covered slack
set rpt [report_timing -to [all_registers -flops] -collection -not_through $inp_set]
set slk_max [get_property $rpt slack]
puts "The covered slack is [expr 100*$slk_max/$cp]% of the clock period"

exit
```

# B

## COMPSOC ARCHITECTURE

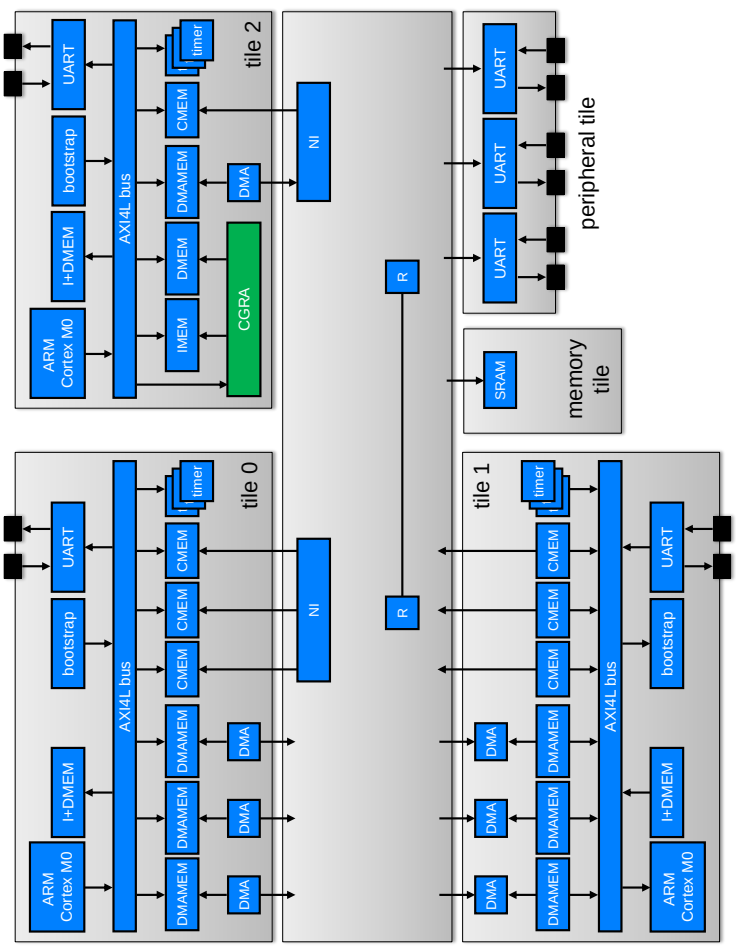



Figure B.1: The architecture if taped out multi-core microprocessor platform generated with CompSOC flow.

# C

## LIST OF ACRONYMS

| | | |
|---|---|---|
| ADAS | Advanced Driver Assistance Systems | 3 |
| ASIC | Application Specific Integrated Circuit | 81 |
| AXI | Advanced Extensible Interface | 82 |
| BIST | Built-In-Self-Test | 11 |
| BTI | Bias Temperature Instability | 7 |
| CGRA | Coarse-Grained Re-configurable Architecture | 91 |
| CHC | Channel Hot Carrier | 23 |
| CPSR | Clock Pulse Stealing Rate | 98 |
| DAHC | Drain Avalanche Hot Carrier Generation | 23 |
| DfT | Design for Test | 8 |
| DLL | Delay Locked Loop | 86 |
| DMR | Double Modular Redundancy | 12 |
| ECO | Engineering Change Order | 45 |
| EDS | Error Detection Sequential | 13 |
| EM | Electromigration | 7 |
| TDDB | Time-Dependent Dielectric Breakdown | 7 |
| ESD | electrostatic discharge | 86 |
| FBW | Fly-By-Wire | 3 |
| FFT | Fast Fourier Transform | 77 |
| FIR | Finite Impulse Response | 99 |
| FSM | Finite State Machine | 18 |
| HCI | Hot Carrier Injection | 7 |
| IC | Integrated Circuit | 1 |
| IIR | Infinite Impulse Response | 98 |
| LBIST | Logic Built-In Self-Test | 55 |
| LER | Line Edge Roughness | 6 |
| LIB | Liberty | 33 |

# D

## LIST OF SYMBOLS

| | | |
|---|---|---|
| $\alpha$ | The carrier velocity saturation power factor in alpha-power-law MOS model for the ON current of transistors. | 25 |
| $\lambda$ | The failure rate of an electronic system | 4 |
| $\mu_{full}$ | The average value of full path delay variation | 48 |
| $\sigma_{full}$ | The standard deviation of full path delay variation | 48 |
| $\tau_f$ | The timing constant of falling | 28 |
| $\tau_{hs,L}$ | The lower estimation of the hold skew time constant | 31 |
| $\tau_{hs}$ | The hold skew time constant | 29 |
| $\tau_r$ | The timing constant of rising | 28 |
| $\tau_{ss,L}$ | The lower estimation of the setup skew time constant | 31 |
| $\tau_{ss,prev}$ | The previous value of setup skew time constant | 30 |
| $\tau_{ss}$ | The setup skew time constant | 29 |
| $C_{load}$ | load capacitance in delay model | 25 |
| $d$ | delay | 25 |
| $d_{CKI}$ | The delay from the clock capturing edge until when the slave latch becomes transparent | 27 |
| $d_{Dm,f}$ | The falling delay from the input data pin to the input data pin of slave latch in a flip-flop | 28 |
| $d_{Dm,r}$ | The rising delay from the input data pin to the input data pin of slave latch in a flip-flop | 28 |
| $d_{ccq}$ | Clock to Q contamination delay of a flip-flop | 20 |
| $d_{cq}$ | Clock to Q delay of a flip-flop | 20 |
| $d_{hs0}$ | the lower bound of hold time | 29 |
| $d_{hs}$ | Hold skew | 19 |
| $d_{ht,L}$ | The lower bound of hold time | 29 |
| $d_{ht,minW}$ | The corresponding hold time for the minimum input capturing pulse width | 29 |
| $d_{ht}$ | Hold time | 20 |

| | | |
|---|---|---|
| $d_{lp}$ | Delay of the longest path | 19 |
| $d_{mQ}$ | The delay of the slave latch from the start of transparency to the switching of the flip-flop output | 27 |
| $d_{pcq}$ | Clock to Q propagation delay of flip-flop | 20 |
| $d_{sp}$ | Delay of the shortest path | 19 |
| $d_{ss0}$ | The lower bound of setup time | 29 |
| $d_{ss}$ | Setup skew | 19 |
| $d_{st,L}$ | The lower bound of setup time | 29 |
| $d_{st,minW}$ | The corresponding setup time for the minimum input capturing pulse width | 29 |
| $d_{st}$ | Setup time | 19 |
| $D$ | The vector which contains the maximum arrival time to all circuit nodes | 43 |
| $D_{ht}$ | The vector which includes characterized hold time values | 30 |
| $DS_{ht}$ | The vector which includes sorted characterized hold time values | 30 |
| $D_{st}$ | The vector which includes characterized setup time values | 30 |
| $DS_{st}$ | The vector which includes sorted characterized setup time values | 30 |
| $e_{ij}$ | Timing arc from vertex i to vertex j in circuit graph | 43 |
| $e_{tau}$ | The error of calculated time constant | 30 |
| $E(G)$ | The set of all timing arcs in circuit graph | 43 |
| $f_{clock}$ | Clock frequency | 51 |
| $I_{on}$ | The ON current of transistor | 25 |
| $k_0$ | A constant for considering the non-modeled parameters in delay model | 25 |
| $L_{eff}$ | The effective channel length | 25 |
| $n_{mon}$ | The number of monitors | 45 |
| $N_{FN}$ | The number of false negative samples | 48 |
| $N_{FP}$ | The number of false positive samples | 48 |
| $N_{IT}$ | Number of interface traps | 22 |
| $N_{TP}$ | The number of true positive samples | 48 |
| $R(t)$ | The reliability of an electronic system | 4 |
| $SLK$ | The vector which contains the minimum slacks of the paths that include each vertex $v_i \in V(G)$. | 43 |
| $S_{bnd}$ | The set of circuit nodes which are on the boundary that is defined based on maximum arrival time criteria | 44 |

| | | |
|---|---|---|
| $S_{inp}$ | The set of insertion points | 44 |
| $Slk_h$ | Hold slack | 20 |
| $Slk_i$ | The slack of the $i$th critical path | 44 |
| $Slk_{max}$ | The maximum covered slack for in-situ delay monitoring | 43 |
| $Slk_s$ | Setup slack | 20 |
| $V(G)$ | The set of vertices in the timing graph of a circuit | 43 |
| $t_{CK}$ | The capturing moment of the clock signal at a flip-flop | 28 |
| $t_{CKI}$ | The time at which the slave latch becomes open | 28 |
| $t_{OR}$ | The delay of OR-tree | 74 |
| $t_{clock}$ | Clock period | 19 |
| $t_{full}$ | The delay of full path | 47 |
| $t_{hs}$ | The time at which the data changes after the clock capturing moment at a flip-flop | 28 |
| $t_{mon}$ | The maximum arrival time to the insertion point of monitors | 45 |
| $t_{sel}$ | The clock selection delay | 74 |
| $t_{ss}$ | The time at which the data changes before the clock capturing moment at a flip-flop | 28 |
| $tr_{hs}$ | The time at which the data changes after the clock capturing moment at a flip-flop for capturing logic one | 31 |
| $tr_{ss}$ | The time at which the data changes before the clock capturing moment at a flip-flop for capturing logic one | 31 |
| $v_i$ | The $i$th vertex in the timing graph of a circuit | 43 |
| $V_{DD}$ | Supply voltage value | 25 |
| $V_{m,eff}$ | The effective voltage driving the slave latch at the onset of transparency | 27 |
| $V_{min,R}$ | The minimum supply voltage of the circuit when the timing resilience technique is activated | 96 |
| $V_{min}$ | The minimum supply voltage of the circuit | 96 |
| $V_m$ | The voltage at the intermediate data node between the master and the slave latches in a flip-flop | 27 |
| $V_{nom}$ | The nominal supply voltage value | 96 |
| $V_{th}$ | The threshold voltage of a transistor | 22 |
| $V_{th,eff}$ | The effective threshold voltage value in the gate delay model | 27 |
| $V_{warn}$ | The maximum supply voltage at which monitor excitation occurs | 96 |
| $W$ | The width of a transistor | 25 |

# E

## CURRICULUM VITAE

Hadi Ahmadi Balef was born on 12-07-1989 in Ray, Iran. After finishing Bachelor's degree in 2011 at Amirkabir University of Technology in Tehran, Iran, he studied Electronics at University of Tehran in Tehran, Iran. In 2015 he graduated within the School of Electrical & Computer Engineering on Circuits & Systems. From June 2015 he started a PhD project at Eindhoven University of Technology at Eindhoven, The Netherlands, of which the results are presented in this dissertation. Since August 2019 he is employed at NXP Semiconductors.

# LIST OF PUBLICATIONS


[1] H. Ahmadi Balef, H. Fatemi, K. Goossens, and J.P. de Gyvez. "Timing Speculation with Optimal In-Situ Monitoring Placement and WithinCycle Error Prevention", *Very Large Scale Integration Systems, IEEE Transactions on*, vol. 27, no. 5, pp. 1206–1217, May 2019.

[2] H. Ahmadi Balef, K. Goossens, and J.P. de Gyvez. "Chip Health Tracking Using Dynamic In-Situ Delay Monitoring", In *Design, Automation & Test in Europe Conf. & Exhibition (DATE 2019)*, Florence, Italy, 2019, pp. 304–307.

[3] H. Ahmadi Balef, H. Fatemi, K. Goossens, and J.P. de Gyvez. "Effective In-Situ Chip Health Monitoring with Selective Monitor Insertion Along Timing Paths", In *Great Lakes Symposium on VLSI (GLSVLSI 2018)*, Chicago, IL, 2018, pp. 213–218.

[4] H. Ahmadi Balef, H. Jiao, K. Goossens, and J.P. de Gyvez. "An analytical model for interdependent setup/hold-time characterization of flipflops", In *International Symposium on Quality Electronic Design (ISQED 2017)*, Santa Clara, CA, 2017, pp. 209–214.