EDA Techniques for the Efficient Analysis of Variability in Deeply-Scaled Transistors

Dimitrios Stamoulis

Department of Electrical and Computer Engineering
McGill University
Montreal, Canada

June 2015

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Master of Engineering.

Στη μνήμη της
Γεωργίας Σταμούλη

Acknowledgments

First and foremost, I am grateful to my supervisor, Professor Zeljko Zilic, for our collegial collaboration, which afforded me the freedom to explore different research ideas, and allowed me to acquire valuable academic experiences. I would like to sincerely thank him for his thoughtful mentorship and rigorous remarks, which always motivated me to push myself to work harder. I am grateful for his trust and support through his recommendation letters. I would also like to thank my thesis reviewer, Professor Gordon W. Roberts, for his constructive criticism and comments.

I must also preface the thesis with due acknowledgment of the institutions that supported the current work. I owe a great deal of gratitude to the Greek State Scholarship Foundation (IKY) for supporting my studies. Thanks to IKY's significant contribution, I was able to focus on my work and to develop a productive research momentum. I would also like to thank McGill University and the Hellenic Scholarships Foundation for the Graduate Excellence Fellowship and the Guy Ouellette – MNA Chomedey-Laval Scholarship respectively. I would like to acknowledge CMC Microsystems for the provision of products and services that also facilitated this work.

During my studies at McGill University, I was fortunate to collaborate and interact with Professor Brett H. Meyer. His insightful feedback on our paper drafts helped me to tremendously improve this current work. I am grateful for his support and the conversations regarding my applications and my admission to the Ph.D. program at Carnegie Mellon University. I would also like to thank Professor Haibo Zeng, for our collaboration during the aFSM project. His mentality and dedication to high-quality research will always be useful and inspiring experiences for me as a graduate student. Moreover, I am thankful for having worked with Ernest Wozniak. Our technical discussions were very helpful for my understanding of system-level concepts.

From NTUA, I would like to thank Professor Dimitrios Soudris for all the discussions we have had over the years. His perception of "strategic planning" have introduced me to powerful intellectual concepts and tools, which have had a dynamic impact on me in my most formative years as a young researcher. I would also like to express my appreciation to Dimitrios Rodopoulos for the highly productive collaboration we have had so far. His organizational skills and his ability to identify and to motivate novel research directions allowed me to form ideas that eventually found their way into this thesis. From IMEC, I

would like to express my sincere gratitude to my supervisors, Professor Francky Catthoor and Dr. Ben Kaczer, for their restless guidance during my internship. I would also like to thank Simone Corbetta, Pieter Weckx, and Peter Debacker for answering all my questions regarding system- and transistor-level modeling concepts.

Many thanks to my IML colleagues Ashraf Suyyagh, Steven Ding, Ben Nahill, Jason Tong, Ari Ramdial, Majid Janidarmian, Atena Roshan Fekr, and Omid Sarbishei for our fruitful cooperation. Thank you to Gabi Sarkis, Andrey Tolstikhin, Prokopis Prokopiou, and Giannis Bakagiannis for all the great experiences we have shared. I would like to thank Kishor Ramaswamy and Louis-Charles Trudeau for providing the French translation of the thesis abstract, and Steven Bielby for proofreading the final thesis draft. I would like to extend a special thanks to Bojan Mihajlovic and George Xereas, for being a great co-working companion over the course of writing all my conference papers and for contributing their thoughts at an important juncture in my life.

I owe a particular debt of gratitude to my friends George Pavlakos, Vagelis Nikoloudakis, Yannis Chatzimichos, Nikos Tsiamitros, and Katerina Mousteraki for their understanding and support during a time when I needed it the most. I find myself fortunate to have the guidance of Anastasios and Yannis Stamoulis. Their intellect has taught me how to be an honest, critical and respectful man. I am also grateful for the generous hospitality of my uncle and aunt, Savas and Vasso Tsolakidis, upon my arrival in Montreal.

But most importantly, my very special gratitude goes to my father Anastasios Stamoulis, my mother Anastasia Aleiferi and my brother Thomas Stamoulis for their unconditional love and their unrelenting support of my life goals and career ambitions. I am really blessed to have parents who dedicate their lives to the happiness of their children.

Abstract

As transistor dimensions scale down to the order of several atoms, digital systems are exhibiting alarming performance degradation and significantly reduced yield due to transistor variability. New Electronic Design Automation (EDA) tools have emerged across the spectrum of digital systems, aiming to capture and mitigate the effects of transistor degradation and to enable error-tolerant computing. This thesis provides a set of comprehensive EDA methodologies for efficiently capturing the impact of variability in deeply-scaled transistors. The developed tools enable complete reliability analysis of an entire Central Processing Unit (CPU) module over the design's lifetime.

The majority of transistor variability phenomena are related to threshold voltage fluctuations, resulting in timing failures due to an overall increase of circuit delay. This thesis delivers valuable contributions in several areas of variability-aware approaches for capturing the timing impact of V_{th} degradation: Firstly, it presents a linear regression technique that allows accurate prediction of the overall delay of a circuit. Based on the regression coefficients, the framework annotates the subset of transistors of the design that should be tracked for variability. Secondly, the thesis presents a path pruning algorithm that identifies the variation-critical paths of a design. Both the proposed approaches are capable of reducing the transistor inventory that needs to be monitored by the EDA solver. That way, the respective EDA flows achieve reduced execution times and memory usage for commercial Static Timing Analysis (STA) and open-source SPICE-based tools.

Prior art on reliability analysis under transistor variability exhibits a limited usability of novel atomistic models for more complex and realistic CPU modules. This thesis combines the accuracy of state-of-the-art (SotA) atomistic modeling with the efficiency of the proposed path pruning algorithm. For an entire CPU module, the EDA flow captures the evolution of reliability metrics (i.e. functional yield, maximum clock frequency) for three years of operation. It also provides useful hints for the degradation of the design under power management techniques and critical path re-ordering. To the best of our knowledge, this is the first study that employs complete reliability analysis for processor-wide reliability metrics, while exploiting the accuracy of atomistic variability modeling.

Abrégé

Présentement, la miniaturisation de la taille des transistors à l'échelle atomique entraîne une dégradation inquiétante de la performance des circuits numériques. De plus, la variabilité de fabrication exhibée entre les transistors engendre un rendement de production considérablement plus faible. De nouveaux outils de Conception Assistée par Ordinateur (CAO) ont été développés pour quantifier et réduire les effets de dégradation sur les transistors permettant ainsi l'élaboration de circuits numériques tolérant aux pannes. Cette thèse présente un ensemble de méthodologies CAO conçues pour évaluer efficacement l'impact de la variabilité pour les transistors de très petite taille. Les outils développés permettent de faire une analyse de fiabilité complète d'un Unité Centrale de Traitement (CPU) pour la durée de vie du circuit.

Les phénomènes de variabilité entre transistors sont principalement liés aux fluctuations de la tension de seuil, causant ainsi des erreurs de synchronisation par l'accumulation des délais dans le circuit. Cette thèse formule des contributions significatives aux techniques qui permettent de quantifier l'effet sur la synchronisation à partir de la dégradation de la tension de seuil (V_{th}). Premièrement, la thèse présente une technique de régression linéaire qui permet de prévoir avec précision le retard total du circuit. En se basant sur les coefficients de la régression, il est possible d'identifier un sous-ensemble de transistors dont la variabilité devrait être surveillée et analysée. Deuxièmement, la thèse présente un algorithme qui identifie les chemins fortement sensibles à la variabilité des transistors en épurant la liste complète des chemins. Les deux approches présentées permettent de réduire le nombre de transistors qui doivent être surveillés par les outils de CAO. Ceci résulte en une diminution du temps d'exécution et de l'espace mémoire nécessaire pour les produits commerciaux d'analyse temporelle statique (STA) et pour les outils d'analyse SPICE en code source libre.

Actuellement, les techniques qui considèrent la variabilité des transistors lors des analyses de fiabilité sont incapables d'appliquer les nouveaux modèles atomistiques pour évaluer des circuits numériques complexes tels que les CPU. Cette thèse combine la précision des modèles atomistiques avec l'efficacité de l'algorithme présenté qui épure la liste des chemins sensibles à la variabilité des transistors. Pour un CPU en entier, l'outil de CAO peut élaborer l'évolution des métriques de fiabilités (i.e. fréquence maximum de l'horloge, rendement de la production) durant trois années d'opération. De plus, la thèse identifie des

pistes de solution pour traiter la dégradation causée par l'utilisation de techniques de gestion de puissance et de réorganisation des chemins critiques. À notre connaissance, ceci est la première étude qui applique une analyse de fiabilité complète pour les métriques de fiabilités du processeur, tout en bénéficiant de la précision des modèles de variabilités atomistiques.

Contents

List of Figures

List of Tables

List of Acronyms

ALU	Arithmetic logic unit
AM	Add-Multiply
AVF	Architectural Vulnerability Factor
BFS	Breadth-First Search
BTI	Bias Temperature Instability
CHC	Channel Hot Carrier
CMOS	Complementary Metal Oxide Semiconductor
CPU	Central Processing Unit
DSP	Digital Signal Processing
EDA	Electronic Design Automation
EMC	Enhanced Monte Carlo
FA	Full Adder
FET	Field Effect Transistor
HDL	Hardware Description Language
IC	Integrated Circuit
ISA	Instruction Set Architecture
MB	Modified Booth
MC	Monte Carlo
MOSFET	Metal-Oxide Semiconductor Field-Effect Transistor

PSTA	Parameterized Static Timing Analysis
PTM	Predictive Technology Model
RD	Reaction-Diffusion
RTL	Register Transfer Level
RTN	Random Telegraph Noise
SPICE	Simulation Program with Integrated Circuit Emphasis
SRAM	Static Random Access Memory
STA	Static Timing Analysis
SSTA	Statistical Static Timing Analysis
VHDL	VHSIC Hardware Description Language

Chapter 1

Introduction

Transistor variability has been present as a valid concern for the reliability of Field-Effect Transistors (FETs) ever since the first Integrated Circuit (IC) was created [1]. The variations observed in older technology nodes were global, affecting the electrical characteristics of all transistors of a chip in the same way (i.e. inter-die variability). For many decades chip makers were able to reduce global variations by improving the manufacturing process control [2]. Moreover, the low complexity of integrated circuits allowed the development of accurate yet simple statistical approaches that capture such variations [3]. Nevertheless, as CMOS technology approaches nanometer scales, new threats for the transistor reliability have emerged and numerous sources of semiconductor variability have been identified.

A large portion of these degradation sources are related to local process variations due to the charge-level activity of dopant atoms in the gate stacks. These degradation mechanisms, which result in threshold voltage V_{th} variations, were initially observed more than 30 years ago [4]. Nonetheless, the impact of individual gate defects was insignificant since older transistor channels contained tens of thousands of dopant atoms. Recent advances in manufacturing tools during the past two decades enabled technology nodes with channels less than 90 nanometers [5]. At that scale, transistors accommodate only a few hundred of defects, and eventually the absence of a single atom has a much more severe impact on the transistor performance. That is, local process variability has a different effect on every transistor in any given chip (i.e. intra-die variability). As a consequence, "two transistors, fabricated a few dozen nanometers apart on the same piece of silicon, will not have the same electrical properties" [6].

As dimensions scale down to the order of several atoms, transistor variability attracts ever-increasing interest from both academia and industry. It has been described by the International Technology Roadmap for Semiconductors (ITRS) as the "red brick" problem [7]: *"one of a handful of important issues that lack any clear solution, forming a red brick wall that prevents forward progress"* on Moore's Law, the defining paradigm of the global semiconductor industry [8]. New EDA approaches have emerged across the spectrum of digital systems, aiming to capture the effects of process variations and to enable error-tolerant computing [9]. The latest edition of ITRS describes the variability-aware EDA methodologies as the key challenge to be addressed within the next five years [7]:

> *"Advanced numerical device simulation models and their efficient usage for predicting and reproducing statistical fluctuations of structure, dopant and material variations [are necessary] in order to assess the impact of variations on statistics of device performance, including non-Gaussian distributions."*

The impact of transistor variability propagates across the entire design abstraction hierarchy, reducing the yield to uneconomic levels and resulting in defective designs that "they might find no better destination than the junkyard. And the defect rate will only get worse as transistors continue to shrink" [6]. As a consequence, variability is becoming a threat not only for the semiconductor industry, with revenues that approached $350 billion in 2014, but for the entire $1.5 trillion global electronics industry [10]. Hence, chip makers now try to develop EDA methodologies to capture and eventually mitigate the impact of transistor variability. Such research challenge is well articulated in the following quote from Miguel Miranda, Qualcomm, CA [11]:

> *"But just because variability is here to stay doesn't mean we can't mitigate its effects. [..] Fortunately, a new family of design techniques [..] use statistical methods to make informed trade-offs between how fast the chips will run and how many good chips a given batch is likely to yield. Some makers of high-end microprocessors like IBM and foundries like the Taiwan Semiconductor Manufacturing Co. are already using some of these statistical techniques in their design flows. Although statistical tools are still far from being widely adopted, if we can push them along, these tools will help us make affordable chips that are as fast and efficient as those the semiconductor road map calls for–and perhaps then some."*

The aforementioned quotes from both industry and academia constitute the main motivation behind the current thesis. Our goal is to develop variability-aware EDA methodologies, in order to create comprehensive design flows for credible reliability analysis. Recent atomistic models have been proposed to accurately capture variability-induced V_{th} degradation, but they suffer from computational complexity. This thesis contributes to addressing some of the obstacles to the widespread adoption of those novel models for design-level simulations. We propose efficient variability-aware methods, such as regression techniques and path pruning algorithms, in order to reduce the transistor inventory that needs to be monitored by an EDA solver. The ultimate goal of our work is to provide a reliability engineer with tools to capture and mitigate the impact of transistor variability, in a reasonable amount of time and with reasonable accuracy.

1.1 Summary of Thesis Contributions

This thesis presents contributions in several areas of variability-aware EDA methodologies, including accurate regression models, path pruning algorithms and efficient processor-wide reliability analysis flows. A summary of contributions per Chapter is listed below.

Chapter 3

This work develops a linear regression model to accurately capture the variability-induced delay degradation. First, we show that such degradation is highly correlated with threshold voltage V_{th} variations. Based on the regression coefficients, we identify the subset of transistors of the IC that should be tracked in order to accurately predict delay values. We reduce the transistor inventory that needs to be monitored by a timing analysis tool and eventually we reduce the operations required by the Static Timing Analysis (STA) solver. That way, we achieve acceleration and reduced memory usage for commercial STA tools with negligible accuracy loss.

Chapter 4

We propose a framework for efficiently capturing the timing impact of transistor variability. Using commercial STA solvers as the core of our analysis, we integrate several novel ideas to further enhance conventional variability analysis flows. To name a few, our analysis

is not constrained inside the STA tool, but is fully decoupled from the STA kernel itself. More specifically, the additional modules are built on top of the STA solver, while adding negligible overhead compared to default timing analysis iterations. In addition, we propose a threshold pruning algorithm to identify the variation-critical paths that needs to be monitored by timing analysis and circuit simulation tools. That way, we achieve acceleration and reduced memory usage for both STA and SPICE-like solvers with negligible accuracy loss. To the best of our knowledge, no existing work attempts to tightly integrate all the aforementioned characteristics into a unified variability analysis flow.

Chapter 5

We propose a comprehensive EDA flow for efficient, processor-wide reliability analysis under transistor variability, i.e. Bias Temperature Instability (BTI). The proposed framework combines the efficiency of pseudo-transient atomistic BTI modeling with the accuracy of commercial STA tools. More specifically, we incorporate a novel pseudo-transient atomistic BTI methodology with a complete EDA flow, reflecting on the usability of atomistic BTI models for realistic circuits. Moreover, we use the path pruning technique developed in Chapter 4 to identify the variation-critical part of an entire CPU module. To the best of our knowledge, this is the *first* study that employs complete reliability analysis for processor-wide reliability metrics (i.e. yield or maximum clock frequency), while exploiting the accuracy of atomistic BTI models.

1.2 Self-Citations

This thesis is comprised of a body of work that has either been previously published, or is at various points in the process of being considered for future publication. Each paper comprising the thesis has been primarily authored by Dimitrios Stamoulis, who is listed as the first author of each of the publications seen below:

1. *Linear Regression Techniques for Efficient Analysis of Transistor Variability* [12] : Prior art on time-zero/-dependent variability shows its importance for digital system reliability throughout a typical integrated circuit (IC) lifetime. Timing analysis results could be questionable if the impact of such variations is not taken properly into consideration. Modern models can accurately capture transistor variability but

they suffer from prolonged execution times. In this paper, we employ linear regression analysis to accelerate transistor variability estimation. Compared to commercial transistor-level Static Timing Analysis (STA) tools, we achieve a 4.63x average speedup and a 3.56x average memory usage reduction for standard cells and ISCAS85 benchmark circuits, with negligible accuracy degradation.

2. *Efficient Reliability Analysis of Processor Datapath using Atomistic BTI Variability Models* (**Best Paper Nomination**) [13] : In this paper, we propose EDA methodologies for efficient, datapath-wide reliability analysis under Bias Temperature Instability (BTI). The proposed EDA flow combines the efficiency of atomistic, pseudo-transient BTI modeling with the accuracy of commercial Static Timing Analysis (STA) tools. In order to reduce the transistor inventory that needs to be tracked by the STA solver, we develop a threshold-pruning methodology to identify the variation-critical part of a design. That way, we accelerate variation-aware STA iterations, with a maximum speedup of 6.82x achieved for representative benchmark circuits. We substantiate the efficiency of the proposed framework for realistic designs. For a CPU datapath, our threshold-pruning technique outperforms built-in pruning commands of the STA solver by 16.87% in terms of runtime improvement. We demonstrate the impact of BTI after three years of operation, with clock frequency degradation up to 24% and functional yield reduction below 90% for higher frequencies.

1.3 Thesis Organization

In Chapter 2 the thesis reviews EDA methodologies and practices that exist for modeling transistor variability. Recent simulation techniques and reliability analysis methods are contrasted, in order to highlight their accuracy issues and their limited applicability. The Chapter also motivates and presents the objectives of the current work. Chapter 3 discusses the leveraging of linear regression methods to efficiently capture variability-induced delay degradation. The methodology to predict delay values is described and its efficiency is verified for representative test cases. Chapter 4 introduces a path pruning approach to identify the variation-critical part of a circuit. The applicability of the proposed technique to both timing analysis and SPICE-like solvers is assessed through benchmark circuits. The path pruning algorithm is further exploited as part of a comprehensive EDA flow for processor-

wide reliability analysis which is presented in Chapter 5. The Chapter demonstrates the usability of our methodology as a useful aid for reliability analysis of an entire CPU module. It also substantiates the impact of transistor variability to power-aware designs and critical path re-ordering. Finally, Chapter 6 summarizes the methodologies detailed in the thesis and speculates on the future of variability-aware EDA techniques.

Chapter 2

Background and Related Work

2.1 Introduction

This Chapter covers EDA methodologies and practices that exist for modeling transistor variability. First of all, it explores the main sources of variability in deeply-scaled transistors and the traditional simulation techniques that are employed to capture the impact of threshold voltage degradation. Considerations surrounding the use of existing EDA tools are explored in detail, while stressing the simulation challenges due to increased execution times and accuracy issues. Previous statistical regression and path pruning approaches that aim to tackle these challenges are also reviewed. Different reliability analysis methods are presented, while highlighting their limitations and the over-simplified test cases. Finally, the motivation and the objectives of the current thesis conclude this Chapter.

2.2 Transistor Variability

2.2.1 Sources of Threshold Voltage Degradation

Transistor variability has been studied for more than 30 years as a valid concern for the reliability of Field-Effect Transistors (FETs) [4]. A large portion of transistor degradation phenomena are related to threshold voltage V_{th} variations, due to the charge-level activity of gate-stack defects. Based on [14], a defect is defined as an unpaired electron of Si atoms at the SiO_2/Si interface that corresponds to a carrier recombination (and generation) site. At these sites, the minority carriers of the channel are trapped (and respectively emitted),

thus reducing the I_{DS} current during the device operation. Such reduction leads to an increase of the absolute value of the threshold voltage that is required for strong inversion of the transistor. While other factors could affect the transistor behavior [15], in this text we focus on V_{th} degradation phenomena due to the intermittent interaction of gate-stack defects with minority carriers, such as Bias Temperature Instability (BTI) and Random Telegraph Noise (RTN) [16, 17, 18].

The perception of the defect activity, and consequently the development of variation-aware EDA tools has undergone large changes over the years. Transistors of older technology nodes had sufficient size and a large number of defects, thus exhibiting a uniform degradation throughout their lifetime. That is, one transistor was representative of the way all transistors of the same technology node would behave. It was therefore a common practice to capture any variability phenomena as deterministic transistor values via worst case corner simulations. With knowledge of the specific characteristics of the technology node, the EDA engineer could include these design margins as parameters of the circuit solver. The low complexity of integrated circuits allowed simple variability models to be directly incorporated with EDA tools.

A modeling approach that has gathered significant momentum throughout the years is the Reaction-Diffusion (RD) model [19]. This approach advocates that V_{th} degradation comes as a result of the breaking (and annealing) of bonds between silicon and hydrogen. This model has been featured in many simulation methods [20], triggering extensive research on BTI/ RTN countermeasures [21]. Nonetheless, recent literature shows that the down-scaling trend observed in the semiconductor industry, has gradually been changing the perception of physics of individual gate oxide defects and the underlying modeling requirements [22]. As CMOS technology approaches nanometer scales, the number of gate-stack defects is reduced. As a consequence, the individual contribution of each defect amplifies the transistor variability. Moreover, the stochastic activity of the oxide traps with minority carriers (i.e. charge capture and emission events) drastically increases the variability among nominally identical transistors of the same technology node. Thus, pre-assigned deterministic worst-case values are no longer accurate and they inherently fail to capture variability phenomena at deca-nanometer technology nodes [22].

Previously, an atomistic approach has evolved, which concentrates on the charge-level activity of gate-stack defects, rather than their actual origin [22]. It attributes the manifestation of BTI/ RTN phenomena to the kinetics of a variable numbers of defects, each one

with a specific temporal behavior (modeled using time constants) and a certain contribution to overall V_{th} degradation. Minority carriers are trapped and emitted from these sites, leading to V_{th} fluctuations. It is also important that the atomistic model can accurately capture the variability of both pFETs (Negative BTI – NBTI) and nFETs (Positive BTI – PBTI), while the majority of RD models are limited to NBTI.

2.2.2 Time-Dependent Variability

Recent literature supports the "paradigm shift" from RD to defect-centric BTI modeling [22]. Novel defect-based EDA approaches inherently cover transistor aging, accounting for transistor variability at deeply scaled technology nodes [23, 24]. Nonetheless, the atomistic treatment of V_{th} fluctuations leads to a statistical and confidence-based perception of several reliability metrics (i.e. delay, functional yield etc) [25]. That is, the main parameters used to calibrate the atomistic model follow statistical distributions, as shown in Table 2.1. More specifically, the threshold voltage shift ΔV_{th} associated with each defect follows an exponential distribution around the mean value η, whereas the number of traps follows a Poisson distribution with an average λ based on the transistor dimensions [26, 27, 28, 29]. Consequently, the V_{th} fluctuations for nominally identical transistors of the same technology will be stochastically distributed in both time and magnitude (i.e. time- and workload-dependent variability respectively).

Table 2.1: Main parameters of the atomistic model and their distributions, based on experimental data [30], as similarly utilized in [13, 24, 31].

Parameter	Distribution
Defects per pFET	$N_p \sim \textbf{Poisson}\,(\lambda = 10^{11} \times \text{Area}[\text{cm}^2])$
Defects per nFET	$N_n \sim \textbf{Poisson}\,(\lambda = 6.7 \times 10^{10} \times \text{Area}[\text{cm}^2])$
Time Const. per Defect (s)	$\log_{10}\left\{\tau^*_{pV}\right\} \sim \textbf{Uniform}\,(a = -12, b = 12)$
ΔV_{th} per Defect (mV)	$\Delta V_{th} \sim \textbf{Exponential}\,(\eta = 5)$

* p: process, either capture (c) or emission (e)
V: voltage, either high (H) or low (L)

The stohastic nature of V_{th} variability is fully consistent with our previous observation that traditional EDA approaches which assume normal distributions fail to accurately account for transistor degradation [32, 33]. Novel methodologies capturing the non-normal

defect-centric V_{th} distributions are necessary. In other words, given the variable manifestation of BTI/RTN it is imperative to develop tools that employ statistical methods. Monte Carlo based techniques are therefore needed to capture time-dependent variability. This observation constitutes a main motivation behind the current thesis. Our goal is to *alleviate the increased CPU overhead of Monte-Carlo (MC) based variability-aware simulations, while maintaining the accuracy and the novel features of the atomistic model.*

2.3 Variability-Aware EDA tools

As transistor dimensions scale down to the order of several atoms, digital systems are exhibiting increasing amounts of performance degradation. As we discussed in the previous Section, more complex variability models have emerged across the spectrum of EDA techniques, aiming to simulate and mitigate the effects of threshold voltage variations. The increasing transistor variability and complexity of integrated circuits create demand for computationally viable, variability-aware EDA solutions.

Novel perception of physics of gate-stack defects has resulted in the respective change of the underlying modeling methodologies. Earlier approaches do not account for time-dependent variability, assuming either deterministic variations or parameter distributions expressed as normal or log-normal statistics. The use of these statistics originates from the practical requirements of low computational overhead during circuit simulations. Modeling the V_{th} variability as a normal distribution has been already used in the state-of-the art (SotA) techniques [34, 35]. Nonetheless, recent works show that the time-dependent component of the aging-induced V_{th} distributions deviates from a normal distributions [29, 36].

Novel simulation techniques based on the atomistic model are able to capture the non-normal defect-centric transistor variability. Therefore, more accurate variability-aware EDA tools are being advocated by several groups [23, 37]. A variety of related implementations exist, modeling BTI either over arbitrary circuit lifetime intervals [38] or in a transient way [39]. However, atomistic BTI modeling comes with increased processing overheads. Recent works have attempted to reconcile the increased accuracy with the computational feasibility of the atomistic BTI model through massive multi-threading [24, 40] or novel signal representations [31]. The atomistic modeling could be placed in different levels of abstraction, starting from transistor- up to system-level tools. In the following Subsections, we review the main EDA solutions per abstraction layer.

2.3.1 SPICE-based Solvers

One of the predominant methodologies for capturing transistor variability is to build transistor models on top of circuit solvers. For atomistic transient BTI/RTN simulation [23], a Simulation Program with Integrated Circuit Emphasis (SPICE) [41] is required. Catthoor et al. have integrated the atomistic model with a commercial SPICE solver [42]. Each transistor is initially populated with a number of traps and the model is evaluated at each simulation step, by calculating the probability for a capture or emission event of each trap. The probability evaluation is implemented on top of a Verilog-A code that functions as the transistor model. More recent works have further optimized the Verilog-A code [39], making the run-time model details fully transparent to the end-user.

These techniques are very accurate but they suffer from prolonged simulation times. Such temporal overhead has a significant impact on *time to market*, by increasing the complexity of reliability analysis during later design stages. Thus, it is important to efficiently facilitate the usage of a variation-aware SPICE tool during the variability analysis stage of a design process. Several approaches attempt to address the introduced overhead of transistor variability models. Rodopoulos et al. proposed a multi-threaded SPICE-based solver that scales across the nodes of a many-core system to handle massive simulations [24]. Nonetheless, an increase up to 24× in execution times is reported for atomistic BTI simulations, compared to default SPICE sessions, where the effect of transistor aging is not taken into consideration.

Another important observation is that transistor-level models focus on a particular physical degradation phenomenon. However, it has been shown that several degradation mechanisms exhibit an intrinsic interdependence [43]. Not capturing such correlation between different variability sources can lead to inaccurate results, with overall error up to 89.23% in temperature and V_{th} estimations [44]. Thus, we can easily observe that SPICE tools are not efficient for statistical Monte-Carlo reliability assessment, due to their complexity and accuracy issues.

2.3.2 Gate-level Timing Analysis Tools

In the previous Subsection, we showed that SPICE-like solutions tools are not viable for statistical time-dependent variability analysis. Thus, it is essential to develop techniques that accurately model V_{th} variability without the cost of prolonged execution times. Moving

to this direction, simulation-free timing analysis tools have been introduced. Different gate-level timing models have been proposed [45, 46]. Numerous transistor degradation phenomena result in timing failures due to an overall increase in circuit delay. Timing analysis results could be questionable if the impact of such variations is not taken properly into consideration. Representative works show that not considering such variability can result in 30% error in delay estimations during timing analysis [47].

State-of-the-art frameworks exhibit a trend towards approaches that allow incorporating stochastically distributed parameters encompassing their combined effect and correlations [48]. Representative examples are methods based on Statistical Static Timing Analysis (SSTA) [46, 49, 50] and enhanced Monte Carlo (EMC) [51, 52, 53] techniques. In Statistical STA (SSTA) methods [43, 44], the variational parameters are treated as random variables with probability distribution functions. Although SSTA and EMC could be more accurate compared to older deterministic simulations, the parameter distributions are usually limited by normal statistics. This underlying assumption of normally distributed variables is not always valid [48], resulting in inaccurate results. An interesting alternative for timing analysis employs Multi-Corner STA [54, 55], where the variational parameters are treated as unknown variables with known bounds. Circuit delay is predicted based on affine (linear) functions of these variables and their corner values. However, it has been shown the linear dependence of process, voltage and temperature (PVT) variables may not always hold, leading to questionable results [48].

2.3.3 Transistor-level STA Tools: Our reference for comparison

As we have already mentioned, our goal is to *"capture the timing impact of aging-induced variability at the transistor level"*. At this level of abstraction, netlists contain transistor properties (e.g. ΔV_{th} values) that can be used directly by transistor-level STA methods [56]. Recent works have shown that transistor-level STA tools have many advantages compared to other timing analysis solvers. More specifically, on-chip variations could be efficiently handled by using margins in transistor-level STA runs [48, 57]. Moreover, it has been shown that transistor-level STA models have a better defined relationship with physical parameters [47]. Finally, these STA approaches are more practical for multimillion gate STA iterations [58]. It is therefore critical to select transistor-level timing analysis tools for a credible analysis of transistor variability.

Throughout this work, we select a commercial STA solver as the reference for comparison, namely `Synopsys NanoTime` [59]. `NanoTime` is an industry-standard STA solution for deeply-scaled technology nodes [60]: "As designs go down to 90nm and below, [..] solutions including traditional static timing analysis with 3rd party variation-aware delay analysis do not provide the accuracy and productivity that is required." The selected reference has been used extensively to deliver timing verification of commercial products with an overall degree of accuracy up to 95% compared to actual delay data [61]. This accuracy obviously comes with an increased computational complexity for larger system-level designs. Nonetheless, in the context of our variability analysis we will focus on subsystem-level CPU modules and benchmarks. Hence, `NanoTime` is the most accurate yet efficient reference for comparison for our exploration.

2.4 Efficient Variability Analysis

Based on the assessment of variability-aware EDA approaches presented in the previous Section, it is evident that traditional STA methods remain the most accurate EDA solution for capturing the timing impact of transistor variability. This observation is another major motivation behind the current thesis: Our goal is to *accelerate Monte-Carlo based explorations, while maintaining the advantages and novel features of commercial transistor-level STA tools*. A key aspect for efficient timing analysis is the identification of the variation-critical part of the design under test. Several related methodologies have been proposed and applied to timing analysis tools, such as regression techniques and path pruning algorithms. In the following Subsections, we review existing works that aim to reduce the transistor inventory that needs to be monitored during timing analysis.

2.4.1 Regression Techniques

In order to predict delay degradation, several statistical techniques could be used. Wang et al. have used Chebyshev polynomial series to fit the gate delay degradation, achieving a maximum fitting error of 0.38% [62]. Nevertheless, time consuming SPICE simulations are necessary per fitting node for discrete delay values to be extracted. Alternatively, regression analysis has taken place at higher abstraction levels (i.e. architecture level) to estimate the architectural vulnerability factor (AVF) of the register file for out-of-order cores [63]. Hence,

it is an attractive option to explore linear regression methods for transistor variability analysis using STA tools.

Following this research direction, Ganapathy et al. have used regression methods to capture spatio-temporal variability at low computational cost [64]. However, significant accuracy degradation is reported in some cases (i.e. 14% average error). In addition, the delay estimation speedup is computed by comparing against SPICE simulators. Nonetheless, it has been shown that SPICE solvers are impractical for timing analysis purposes [65] and as a consequence they are not necessarily a suitable reference in order to evaluate the applied methodology. It is therefore essential to explore linear regression methods for transistor variability analysis, using STA tools as both the implementation base and the reference point.

2.4.2 Pruning Methods

Recent works aim to identify the variation-critical part of the circuit under test. In [66], the authors suggest a threshold pruning method that annotates paths as timing-critical and computes their delay values. Onaissi et al. have presented a path selection method that covers several corner cases [55]. This method was further optimized in [48], by the introduction of pre-determined small set of corners that reduces the required number of STA iterations. Onaissi et al. have also proposed a similar approach that uses branch and bound analysis to annotate the variation-critical paths [67]. Heloue et al. have introduced the notion of Parameterized STA (PSTA), where existing pruning algorithms are mathematically formulated and projected to a *parameter space* [68, 69, 70].

Nonetheless, all the aforementioned approaches are part of monolithic timing analysis solvers, isolated from the the underlying physical transistor degradation mechanisms. In other words, these techniques focus on reporting delay distributions in the context of a timing analysis solver, while failing to provide a generic exploration flow, in order to utilize the results in post-analysis or re-design stages. The motivation of the current thesis is the simple notion that, the results of timing analysis should be further useful, by providing guidance on which variability-sensitive parts the reliability engineer should focus during later development stages (e.g. during computationally heavy simulations for library characterization). Therefore, our intention is to develop a comprehensive framework where the pruning methodologies are built as generic, flexible modules on top of the default STA flow.

2.5 Processor-Wide Reliability Analysis

While the accurate modeling of V_{th} fluctuations at the transistor level remains a motivating challenge for the EDA community, its implications for variability-tolerant VLSI designs are even more important. Handling and mitigating transistor degradation at the system level has become one of the most significant design challenges for computer architects and embedded system engineers. It is therefore crucial to efficiently incorporate the accurate atomistic model with higher levels of abstraction, enabling processor-wide reliability assessment of state-of-the-art systems and applications. Nonetheless, all the existing atomistic modeling approaches limit their exploration to individual transistors, simple gates, SRAM cells and logic blocks.

Table 2.2: Assessment of SotA: Literature lacks an atomistic BTI flow for subsystem-wide reliability analysis, as also published in [13].

Approach	Case Study	Max No. FETs	Level of Abstraction
[23, 71]	CMOS Logic Gates	6	Gates
[38, 72]	6T SRAM cell	6	Gates
[31, 39]	subset of SRAM	244	Gates
[73]	Benchmark Circuits	68,000	RTL/ALU
[24]	Array Multiplier	229,376	RTL/ALU
[74]	Logic Subblocks*	N/A **	RTL/ALU
Current Thesis	**CPU module**	1830	Subsystem

* Adders, multipliers, mux-demux and shifter blocks
** Not explicitly reported

A summary of state-of-the-art (SotA) approaches on atomistic modeling is presented in Table 2.2. To fully assess prior art, we sort existing works by the level of abstraction in which BTI/RTN modeling has been placed, starting from lower to higher abstraction layers. By inspecting the atomistic-related literature, it is evident that it lacks a comprehensive atomistic modeling flow for subsystem-wide reliability analysis. Previous works have been mostly focusing either on simple CMOS logic gates [23, 71] and SRAM cells [31, 38, 39, 72] or on larger netlists of repetitive logic subblocks with reduced functional complexity [24, 73, 74]. Thus, we observe a limited usability of SotA atomistic models for more complex netlists and realistic CPU modules. This observation constitutes the main motivation

behind the current thesis. Our goal is to *incorporate an accurate yet efficient variability modeling methodology with a design flow for credible, subsystem-wide reliability analysis.*

2.6 Motivation and Mission Statement

The intention of this thesis is to provide meaningful contribution and to improve any insufficiencies observed in prior art. Therefore, in the previous Sections we presented an extensive assessment of the research landscape in order to identify such insufficiencies. After having properly motivated our work, we can now summarize the main objectives of our exploration:

1. *Modeling of transistor variability*: Prior art has shown that older modeling approaches fail to account for time-dependent variability at deca-nanometer technology nodes. Thus, we will instead use the more accurate atomistic model to capture V_{th} fluctuations throughout the thesis.

2. *Using efficient yet accurate EDA tools*: We will exploit the accuracy of STA tools to capture the timing impact of transistor variability. As we have already mentioned, we will use the commercial transistor-level `Synopsys NanoTime` tool [59] for the majority of the inspected test cases.

3. *Accelerating variability-aware STA iterations*: We will develop efficient EDA methodologies to accelerate STA iterations. Using regression techniques and path pruning algorithms we will identify the variation critical part of a design, in order to reduce the transistor inventory that needs to be tracked by the STA tool. The proposed methods will be extensively tested through representative cases studies (e.g. benchmark circuits, arithmetic modules etc).

4. *Enabling processor-wide reliability analysis*: After accelerating MC-based timing analysis iterations, we will incorporate the proposed EDA methodologies with a comprehensive design flow in order to enable processor-wide reliability analysis. The efficiency of our EDA flow will be fully verified for an entire CPU module.

2.7 Conclusion

In this Chapter we present the landscape of variability-aware EDA methodologies. The plethora of different variability models and EDA solutions yields a heterogeneous landscape to be explored. Consequently, we classify all the representative approaches into several levels of categorization. First, we present the main sources of V_{th} variations and the respective simulation techniques. Then we discuss the main simulation challenges of existing tools. Finally, we review related works that employ regression or pruning techniques and EDA flows that perform reliability analysis. That way, we are able to identify the insufficiencies of prior art and eventually the objectives of our work.

Chapter 3

Linear Regression Techniques

3.1 Introduction

In this Chapter, we employ linear regression analysis to model variability-induced delay degradation. First, we formulate the regression model. Based on our experimental setup, we show that the overall delay of a circuit is highly correlated with the V_{th} variations across the transistor inventory, which allows accurate prediction of delay values. Moreover, we explore which subset of transistors of the IC should be tracked in order to accurately predict these values. Finally, we verify our method through representative test cases, where acceleration and reduced memory usage are achieved for commercial STA tools with negligible accuracy loss.

3.2 Linear Regression Model

The general form of a univariate linear regression model is given by Equation 3.1, where $\mathbf{y}$ is the univariate response, $\mathbf{x}$ is the matrix that contains the predictor values, β contains the regression coefficients and $\mathbf{e}$ corresponds to the error terms.

$$\mathbf{y} = \mathbf{x} \cdot \beta + \mathbf{e} \Rightarrow \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} x_{11} & \cdots & x_{1n} \\ x_{21} & \cdots & x_{2n} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mn} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_m \end{bmatrix} \tag{3.1}$$

Each row of Equation 3.1 corresponds to an individual data point of the regression model. We could identify two sets of observations:

Definition 1. *Training set is the data set* $(y_i, x_{i1}, ..., x_{in})_{i=1}^{m}$ *of* m *observations used to fit the regression model.*

After having a sufficient number of observations, we fit the model using the *Training Set*. Using maximum likelihood estimation, we derive the regression coefficients (vector β) and the error terms (vector $\mathbf{e}$). We compute the coefficient of determination R^2. This metric will be used as an indicator of how well the generated function fits the observed data (i.e. 100% indicates perfect correlation).

Definition 2. *Test set is the data set* $(y_i, x_{i1}, ..., x_{in})_{i=1}^{k}$ *of* k *observations used to evaluate the regression coefficients* β_i *that were obtained by self-correlating the Training set.*

Using the predictor events of the Test set and the regression coefficients β_i, we compute the expected max delay value $\tilde{y_i}$. The prediction error is given by Equation 3.2.

$$e_i = y_i - \tilde{y_i} = y_i - (\beta_1 \cdot x_{i1} + ... + \beta_n \cdot x_{in}) \tag{3.2}$$

In the context of our exploration, we should properly select the predictors and the response of the model. Our intention is to capture the timing impact of threshold voltage variations on the IC's delay degradation. We will therefore use the maximum delay value $D_{\max,i}$ as the univariate response of the regression model. Moreover, given the equation-based nature of a transistor-level STA solver, several transistor parameters could be used to fit the regression model. Recent EDA approaches focus on the distribution of V_{th} values [31, 38], in order to capture the time-dependent variability. Thus, it is reasonable to use the threshold voltage values of all transistors as predictor variables. We can now derive the following linear regression model (Equation 3.3), where $V_{th(ij)}$ is the threshold voltage value of the $j-th$ transistor during the $i-th$ Monte-Carlo iteration:

$$\begin{bmatrix} D_{\max,1} \\ D_{\max,2} \\ \vdots \\ D_{\max,m} \end{bmatrix} = \begin{bmatrix} V_{th(11)} & \cdots & V_{th(1n)} \\ V_{th(21)} & \cdots & V_{th(2n)} \\ \vdots & \ddots & \vdots \\ V_{th(m1)} & \cdots & V_{th(mn)} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_m \end{bmatrix} \tag{3.3}$$

To obtain the aforementioned data sets, we perform Monte-Carlo transistor-level STA iterations. The delay values are derived using the `Synopsys NanoTime` [59] on an AMD® Opteron® server @2.61GHz. We utilize the 90 nm Predictive Technology Model [75]. Per MC iteration, we introduce a variation to the nominal threshold voltage V_{th0} values. For the introduced V_{th} degradation to be consistent with experimental results (i.e. stochastic properties of gate oxide defects [76]), the ΔV_{th0} variations follow the proper exponential distribution [30]. The circuit netlist with the transistors and the variational threshold voltage parameters is given as an input to the `NanoTime` tool. After performing a detailed transistor-level timing analysis, the tool returns the maximum delay across all paths of the circuit.

3.3 Experimental Results

Initially, we inspect a simple array multiplier as the circuit under test [77]. We focus our exploration on the main repetitive Full-Adder (FA*) module (i.e. 1×1 array multiplier). The schematic of this circuit is shown in Figure 3.1. The SPICE-like netlist that is given as an input to the STA tool consists of 56 transistors.

3.3.1 Delay Degradation Prediction

A *Training set* of 5000 MC iterations is given as an input to the linear regression model. The model is fitted using simple `MATLAB` commands (i.e `mvregress`). An overall coefficient of determination $R^2 = 93.56\%$ is achieved, indicating a good correlation between the maximum delay $D_{\max}$ and the V_{th} values of the circuit transistors. Next, we evaluate the accuracy of our analysis. Figure 3.2 shows the measured and predicted maximum delays using a *Test set* of 50 observations. We observe satisfactory prediction results with negligible deviations (i.e. 0.28% maximum error). In order to further inspect the accuracy of our

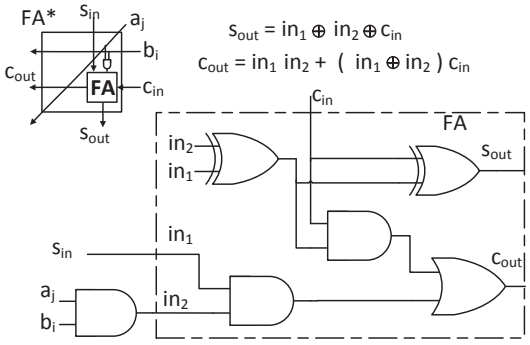


Fig. 3.1: A simple test-case: The FA* module.

regression model, we use a *Test set* of 2000 MC observations. An average error of 0.1125% and a maximum error of 0.3179% are reported.

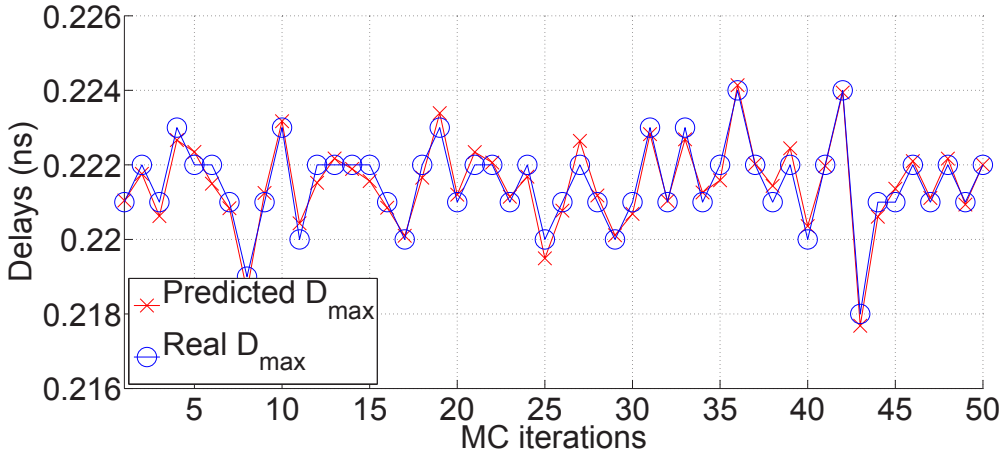


Fig. 3.2: Maximum circuit delay prediction: Predicted values, as compared to real delay data reported using the `Synopsys NanoTime` STA tool.

3.3.2 Tracking a subset of IC's transistors

We have shown that the regression model is able to predict the maximum circuit delay with reasonable accuracy. By further exploring the regression results for the FA* module, we notice that the regression coefficients β vary among the different MOSFET transistors of the circuit under test. Hence, we investigate how accurately we can predict the maximum

delay value using only a subset of the predictor events (i.e. V_{th} values of a subset of transistors). We sort the transistors by their regression coefficients β and we set a varying "threshold" value. That is, all regression coefficients smaller than this value are not taken into consideration while computing the maximum delay of the circuit. For all different subsets, we compute the correlation factor R^2. These values are shown in Figure 3.3. They indicate the overall accuracy while correlating the $D_{\max}$ acquired by tracking all transistors with the $D_{\max}$ acquired by tracking only a subset of n transistors. We highlight with an horizontal red dotted line the points where a correlation factor $R^2 \geq 0.9$ is achieved.

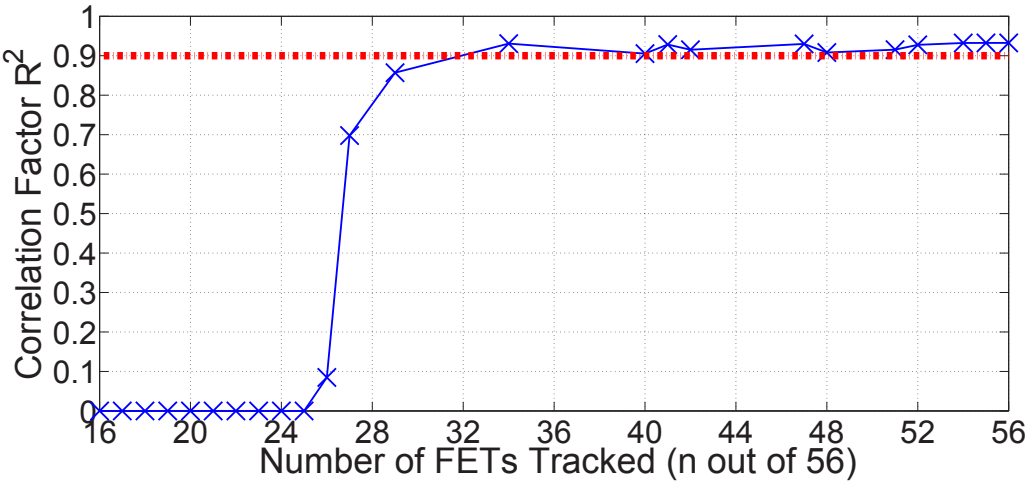


Fig. 3.3: Impact on the correlation factor of the number of transistors tracked.

Considering this as a level of reasonable accuracy, we can easily notice that we can exclude up to 22 out of 56 transistors. Another important observation could be made based on the topology of the FA* circuit itself. The critical path that was identified by the timing analysis tool consists of 24 transistors. One can see that reaching this number gradually leads to accuracy degradation, while keeping less than 24 MOSFETs leads to zero correlation. That is, by excluding transistors that significantly affect the $D_{\max}$ value (i.e. transistors of the critical path), the result becomes inaccurate.

Given the aforementioned observation, we fully investigate the accuracy degradation of timing analysis results while tracking the variability of a subset of transistors. That is, we set the threshold voltage parameters of certain MOSFETs to be variational inside the technology modelcard, while the rest of them maintain their nominal V_{th0} values. Once

again, we inspect the FA* module and we perform 200 pairs of MC STA iterations. For the second iteration per pair, the variability of a subset of MOSFETs is taken into consideration. Figure 3.4 shows the $D_{\max}$ values computed using the `NanoTime` tool for all pairs. To show that the same overall variability is introduced per pair, the mean value of all V_{th} values is presented in x axis, while their standard deviation $\sigma\{V_{th}\}$ is 7.87% of $\mu\{V_{th}\}$. We notice that the measured delays per pair are almost identical. More specifically, we observe an average error of 0.0135% and a maximum error of 0.4566%, while tracking the V_{th} variations of 40 out of 56 transistors.

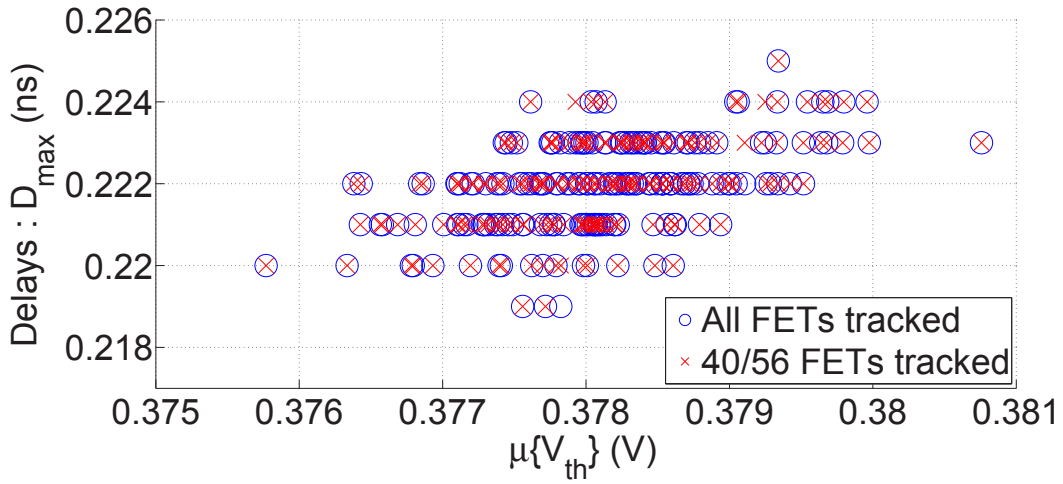


Fig. 3.4: Accuracy assessment of `NanoTime` results while tracking the variability of a subset of transistors.

3.4 Case Study: Accelerating STA tools

Up to this point, we have been able to track the V_{th} fluctuations only of a subset of transistors, with no significant accuracy degradation. Thus, the impact of the remaining transistors could not be taken into consideration during complex timing analysis. We exploit this observation in order to accelerate STA iterations of larger device inventories. More specifically, we apply our methodology to varying dimensions of the array multiplier and some ISCAS85 benchmarks [78]. Per circuit under test, we identify the MOSFETs that contribute more to the overall delay degradation due to V_{th} variations (i.e. greater regression coefficients β). That way, we reduce the transistor inventory that needs to be monitored in order to accurately model transistor variability, using a timing analysis tool.

Reduced execution times and memory usage are achieved compared to default, commercial STA sessions, where the variability of all transistors is tracked.

3.4.1 Test Case I: Array Multiplier

First, we inspect the circuit of a simple array multiplier [77], as shown in Figure 3.5. The device inventory of this circuit is manipulated by changing the bit length of its operands (assumed of equal length).

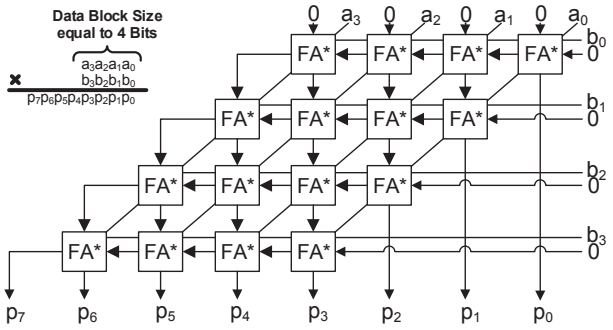


Fig. 3.5: Test Case I: Array Multiplier.

For varying array multiplier sizes and subsets of tracked transistors (i.e. respective speedup), we compute the maximum error of our analysis according to Equation 3.2 (Figure 3.6). We see an increase in the maximum error which is positively related to the logic depth of the circuit. We can easily notice that the reported error never exceeds 1.81%, with a mean error of 1.4%. Moreover, the experimental results illustrate the tradeoff between achieved speedup and accuracy degradation (i.e. less transistors tracked).

For representative subsets of tracked transistors (i.e. where $R^2 \geq 0.9$ based on Figure 3.3), we achieve reduced execution times and memory usage for STA iterations with more than 10,000 transistors. The acquired values are shown in Figure 3.7 and Figure 3.8 respectively. In general, a 2.25× maximum speedup and a 2.19× memory usage reduction are achieved, with an overall accuracy of $> 98\%$ always maintained.

3.4.2 Test Case II: ISCAS85 Benchmarks

We extend our analysis to representative ISCAS85 benchmarks [78]. Per benchmark, we identify the subset of transistors that contribute more to the overall delay degradation (i.e. greater regression coefficients β). We then perform pairs of STA iterations per benchmark

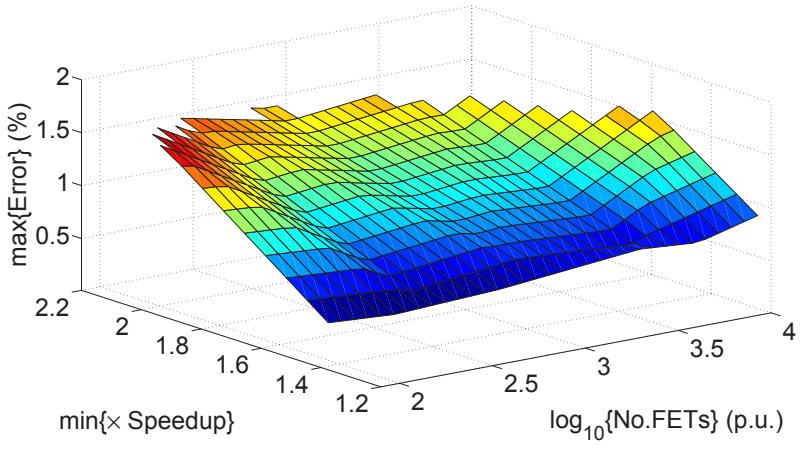


Fig. 3.6: Accuracy assessment of applied regression analysis for different netlist sizes and reported speedup of STA iterations.

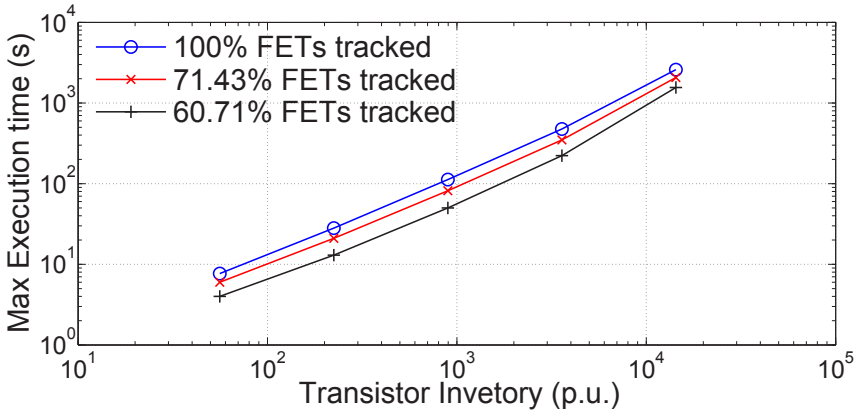


Fig. 3.7: STA tool performance: Maximum execution times for different netlist sizes and subsets of tracked transistors (i.e. respective speedup).

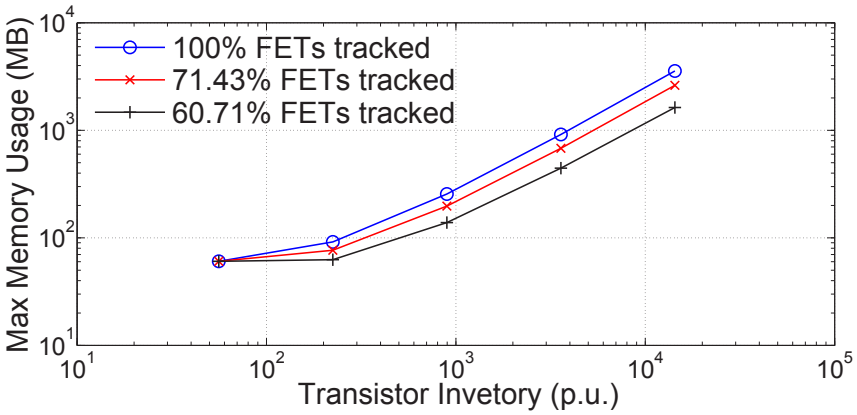


Fig. 3.8: STA tool performance: Maximum memory usage for different netlist sizes and subsets of tracked transistors (i.e. respective speedup).

Table 3.1: Runtime measurements: Evaluation of proposed methodology against default STA sessions for representative ISCAS85 benchmark circuits.

Benchmark circuit	**Default STA sessions**	**Proposed STA Methodology**	**Methodology Evaluation**		
	Max Runtime (s)	Max Runtime (s)	Max Error	Min Speedup	Avg Speedup
c432	123	99	0.09%	1.22×	1.24×
c499	282	42	2.77%	6.64×	6.84×
c880	230	56	1.21%	4.04×	4.09×
c1355	316	62	2.23%	5.00×	5.09×
c1908	492	131	1.01%	3.70×	3.76×
c3540	1016	178	2.10%	5.56×	5.67×
c5315	1493	262	0.52%	5.59×	5.71×
Average			1.42%	4.54×	4.63×

circuit as before. We select 100 iterations as an indicative number of MC samples [31]. Reduced execution times and memory usage are reported in Tables 3.1 and 3.2 respectively. More specifically, a 4.63× speedup and a 3.56× memory usage reduction are achieved on average, while reasonable accuracy is always maintained (i.e. 2.77% maximum error).

Table 3.2: Memory usage measurements: Evaluation of proposed methodology against default STA sessions for representative ISCAS85 benchmark circuits.

Benchmark circuit	**Default STA sessions**	**Proposed STA Methodology**	**Methodology Evaluation**	
	Max Memory Usage (MB)	Max Memory Usage (MB)	Min Memory Usage Reduction	Avg Memory Usage Reduction
c432	266.17	222.89	1.19×	1.19×
c499	584.86	135.15	4.32×	4.33×
c880	482.00	157.69	3.05×	3.06×
c1355	635.17	164.25	3.85×	3.87×
c1908	970.10	306.41	3.16×	3.17×
c3540	1913.98	402.25	4.75×	4.76×
c5315	2852.53	623.56	4.57×	4.57×
Average			3.56×	3.56×

3.5 Conclusion

In this Chapter, we propose a regression analysis method to capture the timing impact of threshold voltage variations. Based on the properly defined regression model, we analyze the data acquired by the commercial transistor-level STA `NanoTime` tool. First, we manage to accurately correlate the maximum delay degradation with the V_{th} fluctuations of circuit transistors. Furthermore, we identify the transistors which contribute more to the maximum circuit delay. By deriving the subset of transistors to be tracked for timing analysis purposes, we apply the methodology to standard cells and ISCAS85 benchmark circuits. For all the inspected cases, we achieve reduced execution times and memory usage in comparison to the reference, commercial STA tool, which employs no regression models. For different benchmark circuits, our analysis outperforms default STA sessions (4.63× speedup and 3.56× memory usage reduction on average), while exhibiting negligible accuracy degradation (2.77% maximum error).

Chapter 4

Variability-Aware Path Pruning

4.1 Introduction

In this Chapter, we propose a path pruning methodology that identifies the variation-critical part of a circuit. First, we present the EDA flow and we describe the individual components. Moreover, we evaluate the performance of our approach against prior art and in terms of introduced overhead. We also verify the efficiency of the proposed framework by applying the path pruning algorithm to representative benchmark circuits. Finally, we present experimental results where acceleration is achieved in comparison to timing analysis tools and SPICE-based solvers.

4.2 Proposed Framework

The proposed framework for capturing delay degradation under transistor variability is shown in Figure 4.1. Our methodology consists of three main parts: (i) the end-user configuration, (ii) the default STA kernel and (iii) our extensions, built on top of a conventional timing analysis flow. The *Netlist* and *Modelcard Generator* modules introduce variability to the netlist (top-level module and subcircuits) and the modelcard respectively. Detailed STA iterations are employed using the commercial, transistor-level STA tool, `Synopsys NanoTime` [59]. Given the STA results, the *Pathfinder* module identifies the variation-critical paths of the circuit.

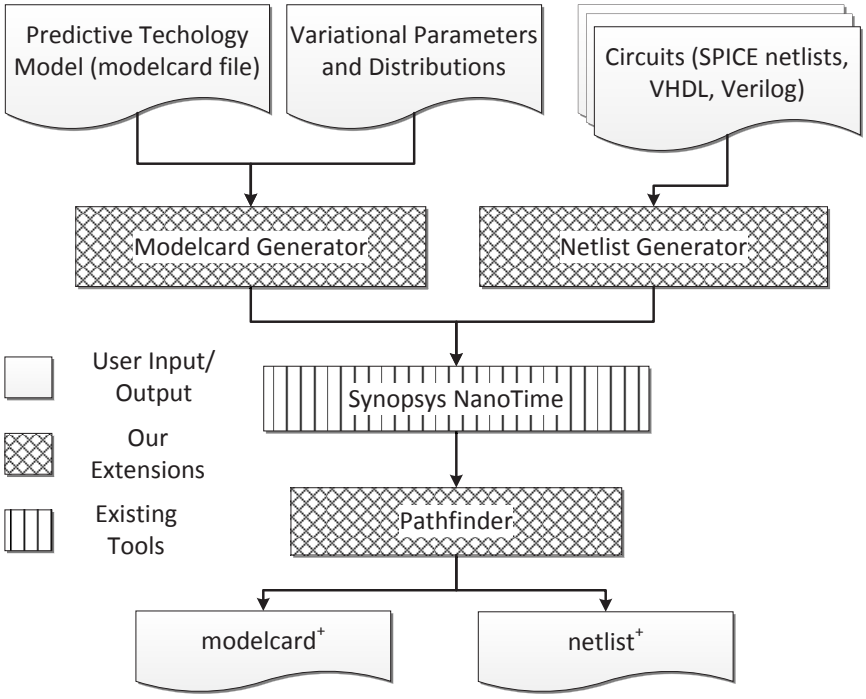


Fig. 4.1: Proposed framework for capturing timing impact of transistor variability.

4.2.1 Netlist Generator

The first step of our method is to capture the variability-induced degradation of the circuit under test. We would like to create a generic flow, well suited to other stages of a design process. That is, our flow is compatible with both SPICE-like netlists or RTL circuits (VHDL, Verilog etc), providing designers with the respective flexibility during previous design stages. We create the *Netlist Generator*. This module consists of a set of fully automated `Perl` scripts that parse the selected circuit file and generate the netlist to be given as input to the STA tool. The resulting netlist maintains all the characteristics of a standard transistor-level netlist (syntax, subcircuits, etc), thus being highly applicable to any transistor-level tool. That way, circuit descriptions of higher levels (e.g. RTL files from previous design stages) are ported to the transistor level, which is more accurate and suitable for directly capturing transistor degradation sources [47].

4.2.2 Modelcard Generator

To capture the timing impact of multiple variability sources, users can fully define the variational parameters and their distribution. To properly introduce transistor variability per STA iteration, we create the *Modelcard Generator* module. Given the distribution of the variational parameters, a set of `Perl` scripts generates the proper modelcard. It should be noted that the resulting modelcard maintains the syntax of the standard technology modelcard which is given as input by the user. The only difference is that the introduced variability varies among all transistors of the circuit (i.e. one model for each individual transistor) and for all individual STA iterations. That way, the analysis accounts for both the inter- and intra-die variations.

4.2.3 Pathfinder: Identifying the variation-critical part

The key idea behind our framework is to leverage accurate timing analysis results to identify the variation-critical part of a circuit. Intuitively, paths with small delay values for all samples do not exhibit significant delay degradation. Hence, we develop a threshold-pruning technique to identify the variation-critical paths under variability-induced V_{th} fluctuations. We create a set of `Python` and `Perl` scripts to implement such functionality. Given the netlist of our design, we first generate all the input-output paths $P_{i,o}$ as candidate search paths. Moreover, the user could select the variational parameters and explicitly define their distributions. Given the STA compatible netlist and the user-defined variability, we perform MC iterations to capture the timing impact of transistor degradation.

Having delay values across all paths of the circuit per MC iteration and the maximum delay D among all samples, we can apply a simple threshold-pruning condition. In [24], it is shown that a signal path that is not initially considered critical could dictate the total circuit delay when transistor variability is incorporated. To account for such uncertainty associated to the timing model, the authors in [66] propose a pruning procedure based on the following condition:

$$\max_{\Delta\lambda}[d_v^{\text{path}}(\Delta\lambda)] + \varepsilon_{\text{path}}^{\max} < D \tag{4.1}$$

where ε is the range that models uncertainty, D is the measured silicon delay and $d(\Delta\lambda)$ is the linear function of the process parameters. This function estimates the worst case delay given a parameter variation vector $\Delta\lambda$. We select this condition as the pruning operator for

its simplicity. We properly adjust Equation 4.1 in the context of our framework by replacing the estimated values with results acquired by the STA solver. The resulting condition is:

$$D_{P_{i,o}} + \varepsilon < D \tag{4.2}$$

where $D_{P_{i,o}}$ is the computed delay of the candidate path $P_{i,o}$, D is the maximum delay across all paths and for all MC samples and ε is the range that models uncertainty. It is reasonable to treat the uncertainty range as a deviation from the maximum expected value D, as in [66]. That is,

$$\varepsilon = \alpha \cdot D, \quad \alpha \in [0, 1] \tag{4.3}$$

Thus, the pruning condition can be written as:

$$\begin{aligned} D_{P_{i,o}} + \alpha \cdot D < D \Rightarrow \\ D_{P_{i,o}} < (1 - \alpha) \cdot D = \kappa D, \quad \kappa \in [0, 1] \end{aligned} \tag{4.4}$$

Instead of initially storing all candidate paths to be later pruned as non-critical, we apply the negation of Equation 4.4 to directly identify paths as critical. That way, we improve the memory efficiency of our pruning technique, as we store only the critical paths to the initially empty set $\mathbf{P}$. The resulting condition is:

$$D_{P_{i,o}} > \kappa \cdot D, \quad \kappa \in [0, 1] \tag{4.5}$$

In [66], it is shown that the threshold pruning strategy is effective, even for a small introduced range (i.e. $\varepsilon = \pm 1\% \cdot d(\Delta\lambda)$). Therefore, it is quite straightforward to select the κ value accordingly. For the remainder of this work, we select $\kappa = 99\%$. By applying the pruning operator, we identify the variation-critical paths (set $\mathbf{P}$) of the design under test. It is worth noting that this outcome is not restricted to a particular abstraction layer, is thus highly applicable to different EDA tools.

In the context of the current work, our goal is to apply our approach to accelerate *transistor-level* timing analysis and circuit simulation tools. Given that these solvers are not aware of paths, but only transistors, the transistors of the variation-critical paths should be therefore properly annotated for use in these tools. Such step is the last step of the proposed methodology. More specifically, fully-automated scripts annotate the variation-

critical part of the circuit under test inside the output files of our framework, by performing Breadth-First search (BFS) to traverse the circuit paths. Given the exponential complexity of the BFS algorithm for larger circuits, it is important to reduce the number of times BFS is executed. Thus, we perform BFS only to variation-critical paths $P_{i,o}$.

Algorithm 1 Proposed Pruning Algorithm.

Input: Gate-level HDL design
1: parse HDL
2: generate STA netlist & $P_{i,o}$ search paths
3: initialize maximum delay $D = 0$
4: set $\mathbf{P} \leftarrow \{\emptyset\}$
5: **for all** MC STA samples **do**
6: introduce variability (Table 2.1)
7: find delay per path $P_{i,o}$
8: **for all** delay values $D_{P_{i,o}}$ **do**
9: **if** $D_{P_{i,o}} >$ maximum delay D **then**
10: $D \leftarrow D_{P_{i,o}}$
11: **end if**
12: **end for**
13: **end for**
14: **for all** MC STA samples **do**
15: **for all** delay values $D_{P_{i,o}}$ **do**
16: **if** $D_{P_{i,o}} > \kappa \cdot D$ **then**
17: add $\mathbf{P} \leftarrow P_{i,o}$
18: **end if**
19: **end for**
20: **end for**
21: BFS: traverse $\mathbf{P}$ & generate variation-aware STA inputs
Output: Variation-Aware `netlist`$^+$ and `modelcard`$^+$

The aforementioned methodology is summarized in Algorithm 1. By executing the Algorithm, the *Pathfinder* module returns the variability-aware modelcard and netlist files, that are referred as `modelcard`$^+$ and `netlist`$^+$ respectively. Inside these files, only transistors that belong to variation-critical paths are annotated as variational (i.e. they are assigned parameters or new models), while the rest of them maintain their nominal values. The resulting files are returned to the user front-end and are used as input for successfully accelerating transistor-level tools, as we discuss in Section 4.4.

4.3 Methodology Evaluation

4.3.1 Pruning Performance

In order to assess the efficiency of our approach, we evaluate the performance of the proposed pruning algorithm against related state-of-the-art (SotA) results [66]. Such comparison is shown in Figure 4.2, where we report the percentage of paths that are pruned using our algorithm and the method presented in [66], for the same κ value. We observe that our proposed methodology is equally aggressive compared to prior art, in terms of number of paths pruned. The performance of both pruning methods is almost identical with only a 0.03% deviation between the average percentage of pruned paths. This is to be expected given the usage of the same pruning condition in both approaches.

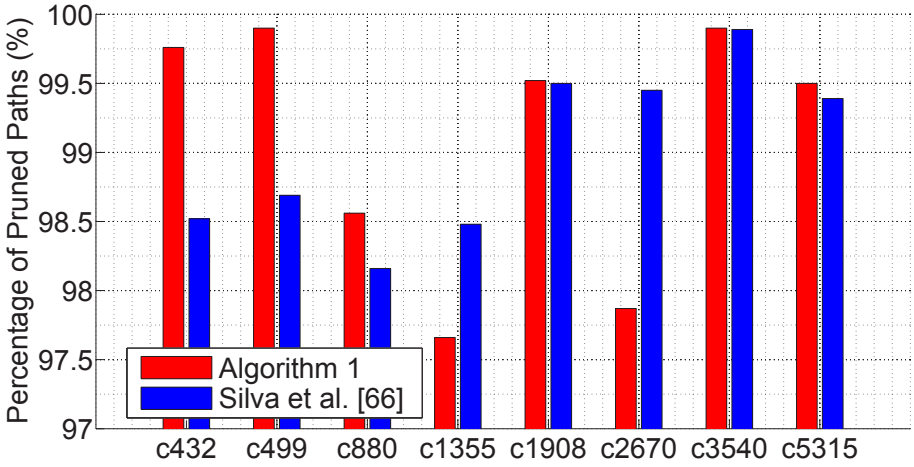


Fig. 4.2: Algorithm 1 is equally aggressive compared to prior art [66], in terms of number of paths pruned.

4.3.2 Performance Assessment

Moreover, we inspect the overhead introduced by our extensions, executed on top of a default Monte-Carlo (MC) based timing analysis flow. We compare the total runtime of our methodology against a commercial STA solver executed for 1,000 iterations. The total runtime values for both cases are presented in Table 4.1. We observe that the additional temporal overhead is negligible, with an average 0.533% increase in execution times.

Table 4.1: Temporal overhead compared to default STA iterations

Benchmark circuit	Runtime ($10^4\times$ s)		Temporal Overhead
	MC STA Iterations	Proposed Framework	
c432	0.2024	0.2030	0.330%
c499	0.4659	0.4675	0.332%
c880	0.3790	0.3803	0.334%
c1355	0.5194	0.5249	1.048%
c1908	0.8089	0.8122	0.413%
c2670	1.1480	1.1540	0.522%
c3540	1.6577	1.6652	0.452%
c5315	2.4683	2.4891	0.836%
Average	0.9562	0.9620	0.533%

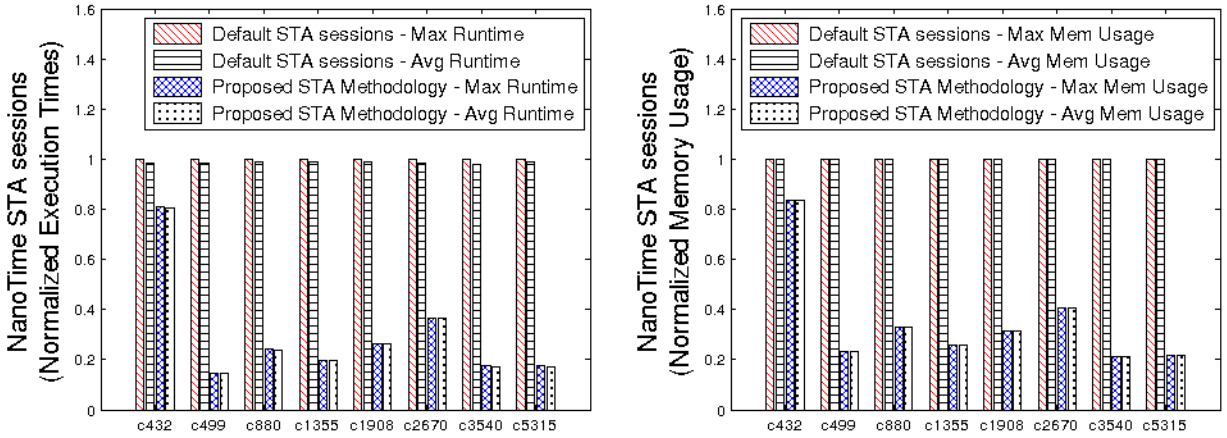
While the proposed pruning algorithm is an effective method for identifying the variation-critical paths, it could have some limitations. More specifically, the BFS used to traverse the circuit paths has an exponential complexity. This complexity might not be negligible for larger, system-level designs. Existing works have successfully addressed this limitation by substituting a brute-force search with more mathematically formulated methods [69, 70]. Nonetheless, fully formulating the proposed method is beyond the scope of the current work. It is important to note that in the context of our variability analysis we will focus on subsystem-level benchmarks and CPU modules in Chapters 4 and 5 respectively. Thus, the BFS overhead is insignificant for the purposes of our exploration.

4.4 Experimental Results

4.4.1 Case Study I: Accelerating STA tools

Given the computational feasibility of our framework, we can accelerate STA sessions by capturing the variability of the variation-critical part of a circuit. That is, we provide the `netlist`$^+$ and `modelcard`$^+$ files as inputs to the STA solver. Inside these files, transistors that belong to variation-critical paths are annotated as variational, while the rest of them maintain their nominal values. That way, we reduce the transistor inventory that needs to be monitored for variability by the transistor-level timing analysis tool.

We compare against 1,000 MC STA iterations where all paths are tracked for the same



(a) Normalized execution times (b) Normalized memory usage

Fig. 4.3: Comparison of the proposed methodology against default STA sessions.

Table 4.2: Evaluation of proposed methodology against default STA sessions for representative ISCAS85 benchmark circuits

Benchmark circuit	Min Speedup	Min Mem Usage Reduction	Max Error	Avg Error
c432	1.23×	1.19×	0.27%	0.03%
c499	6.82×	4.32×	3.99%	2.64%
c880	4.01×	3.05×	1.31%	0.77%
c1355	4.98×	3.86×	2.52%	1.60%
c1908	3.72×	3.16×	1.36%	0.70%
c2670	2.67×	2.45×	0.97%	0.48%
c3540	5.57×	4.75×	2.66%	1.60%
c5315	5.59×	4.57×	0.84%	0.45%
Average	4.32×	3.42×	1.74%	1.03%

introduced variability. We achieve reduced execution times (Figure 4.3a) and memory usage (Figure 4.3b) in comparison to the reference, commercial `Synopsys NanoTime` tool. Analyzing the results in Table 4.2, we can conclude that our approach achieves significant improvement for each circuit, except the c432. Such performance is highly dependent on the circuit topology, which dictates the number of variation-critical paths selected over the pruned paths. A 4.32× speedup and a 3.42× memory usage reduction are achieved on average, while reasonable accuracy is always maintained (i.e. 3.99% maximum error). It

should be noted that these reported values are the minimum acquired values per benchmark. That is, even for the most pessimistic results, our methodology always outperforms traditional STA iterations, while a maximum speedup up to 6.82× is achieved.

4.4.2 Case Study II: Accelerating SPICE tools

In terms of SPICE-level variability modeling, related works develop transistor models tightly integrated into circuit solvers [24, 39]. The execution flow of such EDA solvers is shown in Figure 4.4. It has been shown that 66% of transient run-time is spent inside the *SPICE Iteration* phase, to fully evaluate the transistor model code [79]. Thus, integrating degradation models inside the simulation flow introduces further temporal overhead.

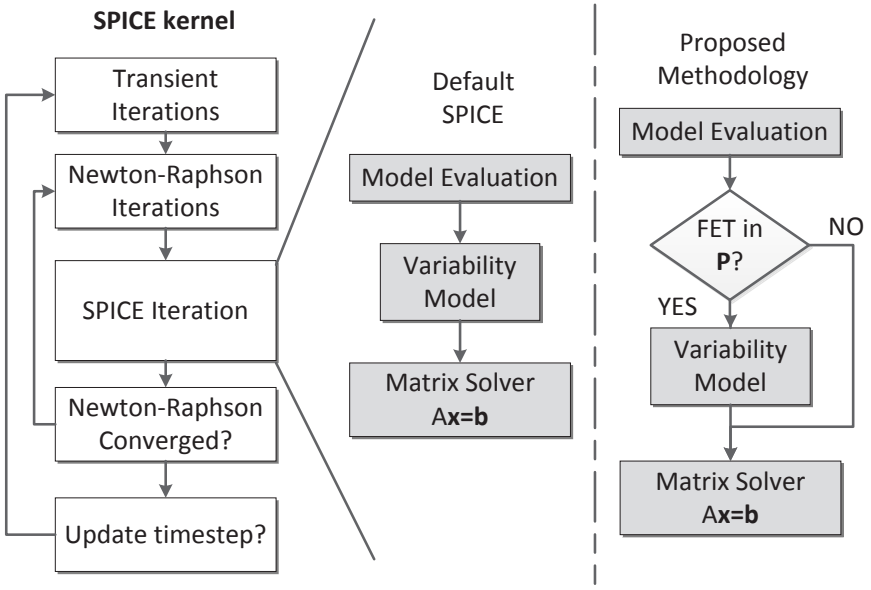


Fig. 4.4: Default and proposed SPICE simulation methodology.

We could accelerate this computationally expensive phase by reducing the operations required by the circuit solver. In other words, we can exploit the outcome of our analysis and apply it to SPICE-like solvers. Given the properly annotated `netlist`$^+$ and `modelcard`$^+$ files, we can optionally bypass the execution of the degradation model. We modify the traditional execution flow by adding the respective condition. At each time step of the simulator, we check if the transistor to be evaluated belongs to a variation-critical path $\mathbf{P}$. This modified flowchart is presented in the right half of Figure 4.4.

To evaluate our variation-critical methodology, we fully integrate the modified SPICE simulation flow into a SPICE kernel. We use the open source version of U.C. Berkeley's `Spice3f5` simulator, namely `ngspice` [80]. Full observability of the code base allows fine-grained alterations. We emulate a degradation model (*Variability Model* black-box) by using two parameters: (i) the introduced variability and its distribution and (ii) the timing impact of simulating the degradation mechanism. Both parameters are given as input to the modified SPICE kernel. It is important to point out that the fluctuations are introduced per transient analysis step. That is, our analysis can capture transient evolution of transistor variability, implemented as time-dependent degradation models.

Moreover, the temporal overhead is emulated per transient step as a "busy wait" of the circuit solver. Our goal is to provide scope for a broad exploration space, by selecting a distribution that encloses a reasonably wide overhead. As we have already mentioned, a defect-based model is *atomistic*, assigning different numbers of defects to each transistor. The degradation model is evaluated at each step of the simulation, for all the defects [39]. It is quite straightforward to assume that the atomistic model can be implemented as a *for loop*, that cycles through all the defects of all transistors. The complexity of such model per transistor is $O(n)$, where n is the number of defects of the MOSFET.

Given that n follows a Poisson distribution [24], it is reasonable to assume that the CPU overhead introduced due to the degradation model does the same. Hence, we perform SPICE simulations where the emulated overhead β per transistor follows distributions consistent with experimental results. More specifically, we have overheads β_p and β_n for pMOS and nMOS transistors respectively, as shown in Equation 4.6. Finally, it is worth noting that the proposed flow is fully decoupled from the content of the "black box" itself, making our analysis applicable to any underlying physical phenomenon and variation-aware transistor model [24, 39].

$$\begin{aligned} \beta_p &\sim \mathbf{Pois}\left(\lambda = 10^{11} \times W \times L\ [cm^2]\right) \\ \beta_n &\sim \mathbf{Pois}\left(\lambda = 6.7 \cdot 10^{10} \times W \times L\ [cm^2]\right) \end{aligned} \tag{4.6}$$

We apply the proposed SPICE simulation flow (Figure 4.4) to accelerate circuit solvers. We compare our methodology against default SPICE sessions, using identical input traces of $2.56\mu s$. For all the variation-critical paths (set $\mathbf{P}$), we measure delays based on the rise/fall times of the SPICE input/output signals [77]. Given these delays, we assess the accuracy

degradation while neglecting the variations of a subset of MOSFETs, in comparison to default SPICE simulations where the variations of all transistors are tracked. The computed error values and the respective speedup per case are presented in Table 4.3. A 4.48× speedup is observed on average, while reasonable accuracy is always maintained (i.e. 1.17% maximum error). Once again, it should be noted that the presented results correspond to the minimum speedup and maximum error. That is, even for the most pessimistic cases, our proposed methodology outperforms default SPICE simulations (Figure 4.5). In fact, a 7.49× maximum speedup is achieved, which is a significant improvement for SPICE simulations with execution times up to 10 hours.

Table 4.3: Evaluation of proposed methodology against default SPICE simulations for representative ISCAS85 benchmarks

Benchmark circuit	Min Speedup	Max Error	Avg Error
c432	1.19×	0.0351%	0.0094%
c499	7.49×	0.2000%	0.0592%
c880	3.97×	0.0102%	0.0053%
c1355	6.09×	0.0187%	0.0153%
c1908	3.67×	1.1700%	0.6850%
Average	4.48×	0.2868%	0.1548%

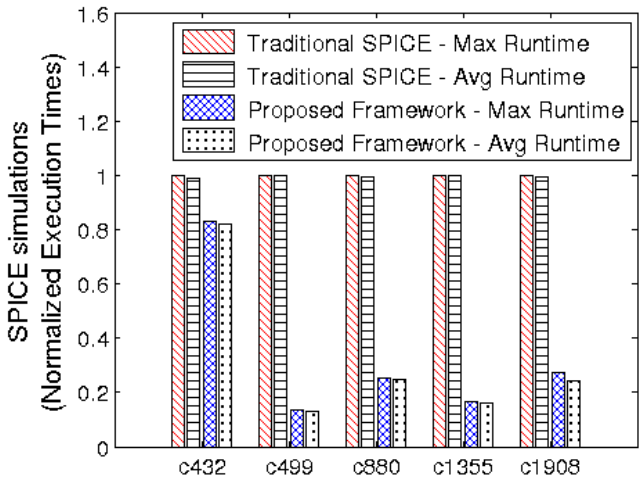


Fig. 4.5: Comparison of the proposed methodology against default SPICE simulations.

4.5 Conclusion

In this Chapter, we propose a framework that efficiently captures the timing impact of transistor variability. First, we present a pruning algorithm built on top of a timing analysis flow and we evaluate its performance by identifying the variation-critical part of representative benchmark circuits. We then show that the outcome of our framework is highly applicable to both timing analysis and SPICE-like tools. For ISCAS85 benchmarks, we achieve speedup and memory usage reduction compared to commercial STA tools. We also accelerate SPICE simulations with negligible accuracy degradation. That way, we alleviate the CPU complexity of time consuming STA and circuit simulation tools, while retaining variability-aware analysis, thus combining high speed and good accuracy.

Chapter 5

Processor-Wide Reliability Analysis

5.1 Introduction

In this Chapter, we propose a comprehensive EDA flow for efficient, processor-wide reliability analysis under transistor variability. More specifically, we capture the impact of Bias Temperature Instability (BTI) on reliability metrics, such as the functional yield and the maximum clock frequency of an entire CPU module. The proposed EDA methodologies combine the accuracy of atomistic BTI modeling with the efficiency of the path pruning method that was proposed in the previous Chapter. As we have already observed during the assessment of prior art, all the existing approaches limit their exploration to single transistors or to simple gates and logic blocks. To the best of our knowledge, this is the first study that employs complete reliability analysis for processor-wide reliability metrics, while exploiting the accuracy of atomistic BTI models.

First, we present the main principles of a novel, pseudo-transient atomistic BTI methodology [31], that is incorporated with our framework. We also describe the proposed EDA flow and its components. We then demonstrate the usability of our approach as a useful aid for reliability analysis through realistic testcases. For an entire CPU module, we capture the impact of BTI and we present the evolution of processor-wide reliability metrics for three years of operation. We also provide useful hints regarding power-aware designs. We investigate the design degradation under BTI for different power management techniques (i.e. voltage scaling). Finally, we substantiate the impact of BTI-induced variability to critical path re-ordering.

5.2 Transistor-Level V_{th} modeling

In previous work, a *pseudo-transient* simulation scheme has been devised, enabling fast yet accurate modeling of BTI [31]. We incorporate some of the basic principles of this prior art framework within our proposed EDA flow, in order to have an accurate transistor-level modeling of threshold voltage variations. In the context of our analysis, we will introduce two sources of V_{th} variability, (i) a *time-zero* ($\Delta V_{th,\mathrm{TZ}}$) and (ii) a *time-dependent* ($\Delta V_{th,\mathrm{TD}}(t)$) component, as summarized in Equation (5.1). The former source of variability is typically attributed to fluctuations during the manufacturing process. We will assume that this component is constant throughout the device lifetime and that it follows a Gaussian distribution, namely $\Delta V_{th,\mathrm{TZ}} \sim \mathbf{Norm}(0, 0.1 \times V_{th0})$, where V_{th0} is the threshold voltage at $V_{bs} = 0$ V, which is typically provided in the transistor modelcard. The latter variability component includes all the time-dependent mechanisms that affect V_{th}, which in our case includes BTI defect activity, simulated according to the atomistic model.

$$\Delta V_{th}(t) = \Delta V_{th,\mathrm{TZ}} + \Delta V_{th,\mathrm{TD}}(t) \tag{5.1}$$

In order to enable maximum compatibility with existing STA tools, we choose to express V_{th} variability with a single Gaussian distribution. This design choice also removes the complexity of verbose, circuit-wide, defect databases that were previously utilized [24, 31], since we are only interested in the aggregate V_{th} variability at the level of each transistor. As a result, our goal is to create a distribution formulated as $\Delta V_{th}(t) \sim \mathbf{Norm}\{\mu(t), \sigma(t)\}$. To achieve this goal, we perform a Monte Carlo session per transistor size, where each iteration follows the procedure illustrated in Figure 5.1, calibrated according to Table 5.1. Initially, time-zero variability is appended. Then, BTI is evaluated over four distinct intervals of transistor lifetime, covering a total of roughly three years (10^8 s). Strides over these intervals are performed according to a "pseudo-transient" simulation setup presented in previous work [31].

After 300 iterations of this process, one can derive estimates for $\mu(t)$ and $\sigma(t)$, by calculating the mean and standard deviation of the available V_{th} data at each point of transistor lifetime. In Figure 5.2, we present these simulation results, exploring representative FET transistor sizes, assuming the 90 nm Predictive Technology Model [75] and a 95% confidence interval [81]. Given that transistors of the same technology node are typically

Table 5.1: BTI model calibration [13], based on experimental data [30] and the 90 nm Predictive Technology Model [75].

Parameter	**Distribution**
Operating V_{dd}	$V_{dd} = 0.9$ V = const.
Operating Clock Freq.	$f = 1$ GHz = const.
Temp./Signal Activity	$T = 50^{\text{o}}$C = const. / $\alpha = 0.5$ = const.
Time-Zero pFET	$\Delta V_{th,\text{TZ}} \sim \textbf{Norm}\,(0\text{ V}, 0.0397\text{ V})$
Time-Zero nFET	$\Delta V_{th,\text{TZ}} \sim \textbf{Norm}\,(0\text{ V}, 0.0397\text{ V})$
Defects per pFET	$N_p \sim \textbf{Pois}\,(\lambda = 10^{11} \times \text{Area}[\text{cm}^2])$
Defects per nFET	$N_n \sim \textbf{Pois}\,(\lambda = 6.7 \times 10^{10} \times \text{Area}[\text{cm}^2])$
Time Const. per Defect (s)	$\log_{10}\left\{\tau^*_{pV}\right\} \sim \textbf{Unif}\,(a = -12, b = 12)$
$\Delta V_{th,\text{TD}}$ per Defect (mV)	$\Delta V_{th,\text{TD}} \sim \textbf{Exp}\,(\eta = 5)$

* p: process, either capture (c) or emission (e)
V: voltage, either high (H) or low (L)

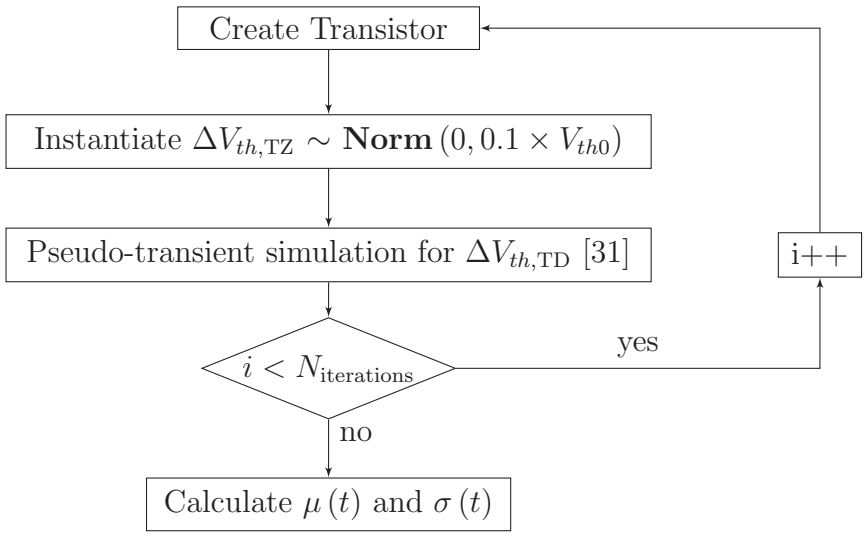


Fig. 5.1: MC iteration scheme to derive distributions for $\Delta V_{th}\,(t)$; we use $N_{\text{iterations}} = 300$ as the halting criterion.

instantiated at various widths (W), we explore different width values, assuming unit widths of $W = 180$ nm for an nFET and $W = 360$ nm for a pFET. Thus, for the 90 nm technology node the notation "$i \times$ pFET" refers to a pFET with area $A = i \times 360 \times 90$ nm^2 (similarly, for nFETs). This notation is used in the legends of Figure 5.2.

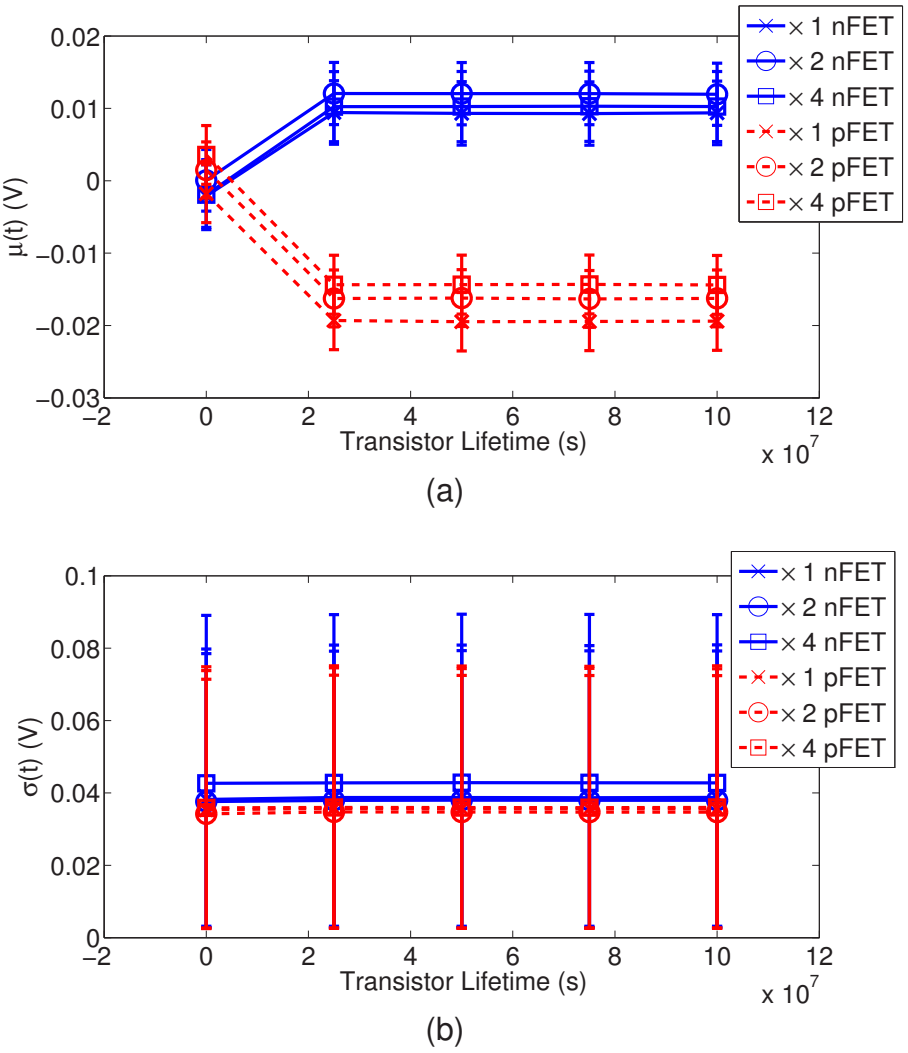


Fig. 5.2: Simulation data for $\Delta V_{th}(t) \sim \mathbf{Norm}\{\mu(t), \sigma(t)\}$, while combining both time-zero and time-dependent variability [13].

Based on the results of Figure 5.2, we can make the following observations: **(i)** The parameter estimation for the distribution of $\Delta V_{th}(t)$ shows a distinct shift after the time-zero instance and remains fairly constant for the rest of the transistor lifetime. This is to be expected given the uniform distribution of defect time constants across the logarithmic time axis. In other words, striking changes in threshold voltage are to be expected if transistor lifetime is inspected logarithmically [23]. This also resembles the power law that is expected by older BTI models, such as the Reaction-Diffusion model [82]. **(ii)** For the considered technology node, transistor area appears to have effectively no impact on BTI

impact. This is to be expected, if we consider that the mean V_{th} impact of a single defect is inversely proportional to transistor area, whereas the number of charged traps is positively related to A [83]. In other words, as the transistor area increases (by increasing W) more defects contribute to $\Delta V_{th,\mathrm{TD}}$, each of which has a decreased contribution. Thus, increase of W does not affect the aggregate V_{th} variability.

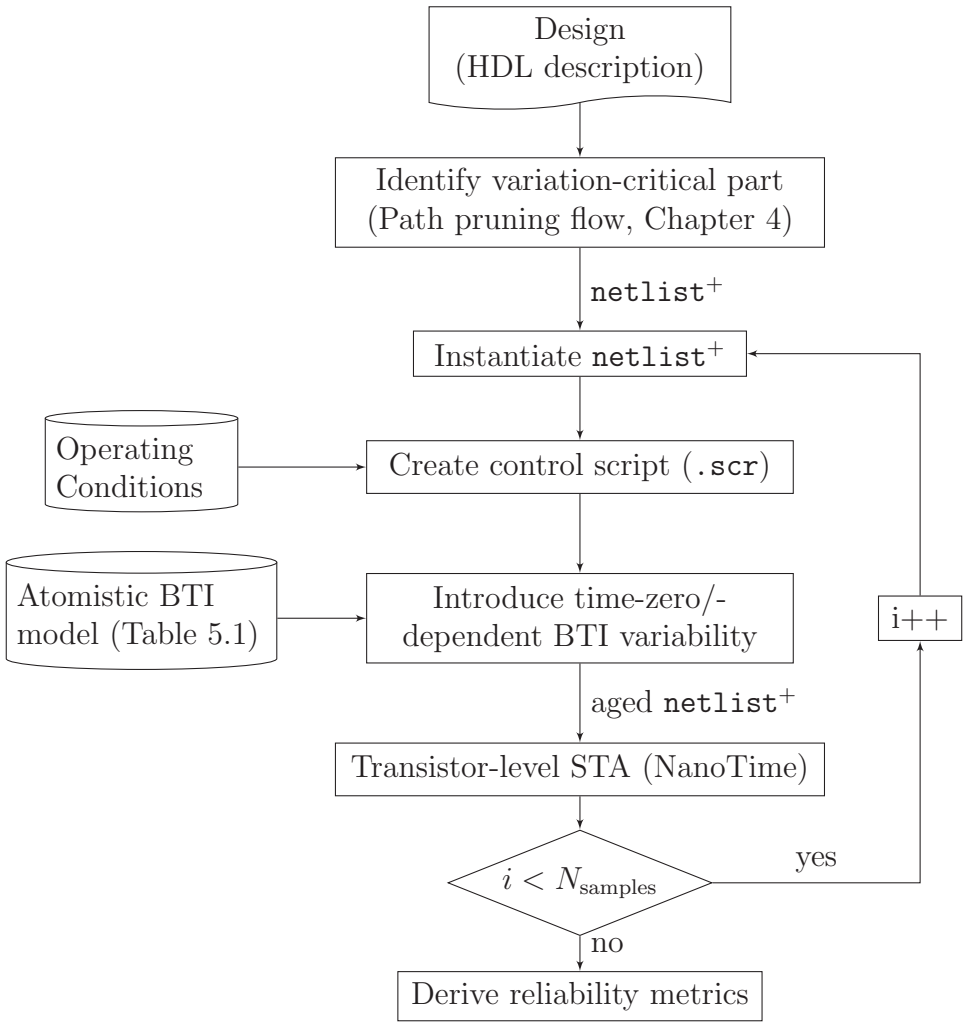


Fig. 5.3: Proposed reliability analysis flow.

5.3 Proposed EDA Flow

In order to create a comprehensive reliability analysis framework, we combine the modeling methodologies presented in the previous Section and the pruning algorithm introduced in the previous Chapter. All the individual components that have been already extensively tested are incorporated with a complete EDA flow that is shown in Figure 5.3. The first part of our analysis is the path pruning procedure, where the variation-critical part of the design is identified, using Algorithm 1 and the respective scripts. The variation-aware `netlist`$^+$ is given as an input to the remainder of the reliability analysis flow.

During the second part of the framework, we perform timing analysis iterations under varying operating conditions and different instances of transistor variability. Per iteration, we first generate the proper control script based on the operating conditions (i.e. f_{CLK}, V_{DD}). We then introduce BTI-induced variability to the transistor-level `netlist`$^+$, according to the distributions that accurately capture the time-dependent variability (Table 5.1). Hence, we can now employ accurate yet efficient transistor-level timing analysis using `Synopsys NanoTime`. Given the MC STA samples, we decide on the processor-wide reliability metrics (i.e. functional yield etc). We test our framework on an AMD® Opteron® server @2.61GHz.

5.4 Experimental Results

5.4.1 Case Study: Datapath

In order to investigate the usability of the proposed design flow for realistic circuits over their lifetime, we use a modern open source RISC architecture, namely `OpenRISC` [84]. We focus our exploration on an important CPU module, the datapath. The main I/O pins and a top-level schematic of the module are shown in Figure 5.4. More specifically, the inputs are presented in the left side of the module, whereas the outputs are shown in the right side. Based on the input buffers, we can observe that the CPU in general uses a 32-bit Instruction Set Architecture (ISA) and that the operand length is 16 bits. The module also consists of control logic that sets the proper control flags per instruction, in order to identify the ALU command to be executed, to reset the datapath values and to distinguish read or write register operations.

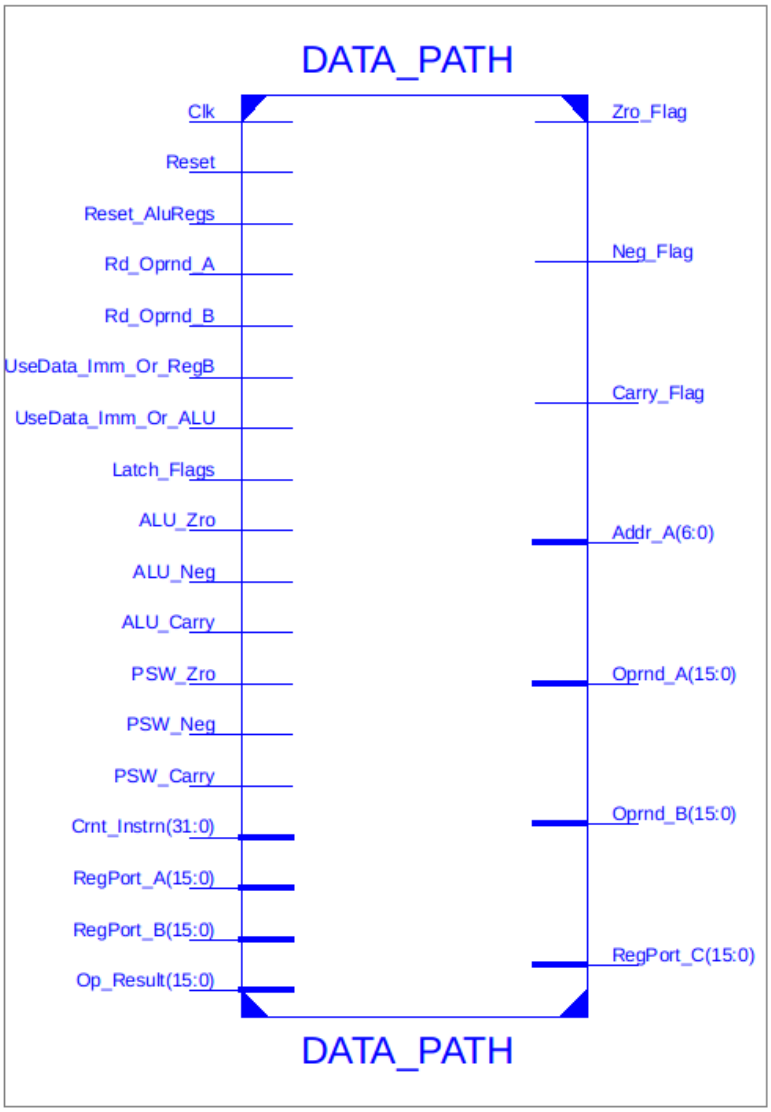


Fig. 5.4: A top-level schematic of the datapath module.

The next step is to port the exploration to the transistor level, starting from the RTL description. We utilize the 90 nm Predictive Technology Model [75], for which we have already calibrated the atomistic variability model in Section 5.2. Given the behavioral VHDL code of the datapath and the target technology node, we use the `Synopsys Design Compiler` [85] to generate the gate-level design. The synthesis is conducted considering the highest degree of optimization, in terms of area and speed. We should also identify the subset of the cell library to be used for the synthesis phase.

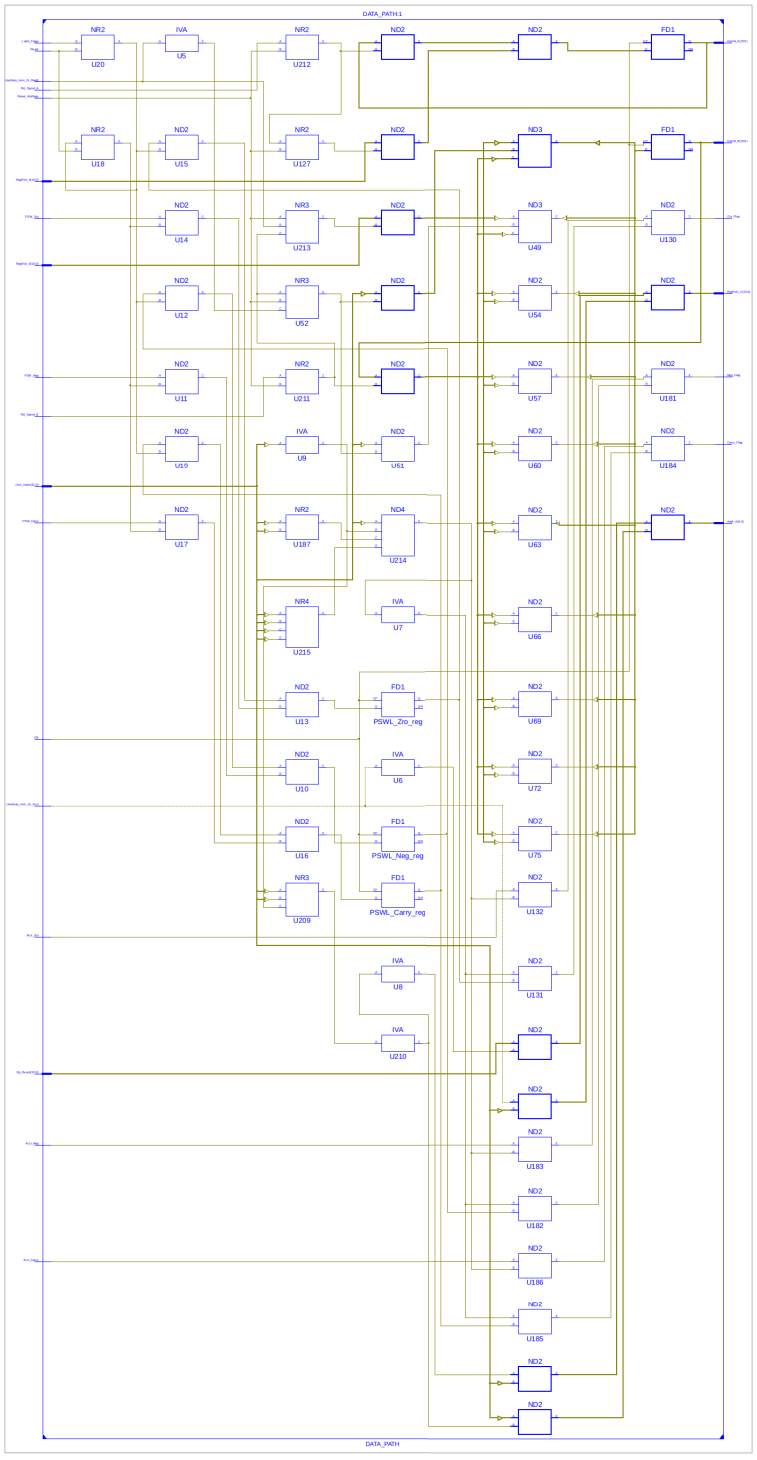


Fig. 5.5: A gate-level schematic of the datapath module.

For our exploration, we use simple CMOS logic, thus we utilize D Flip-Flops, Inverters, NAND and NOR gates. By issuing the `set_dont_use` command inside the compilation script, we identify the gates to be used from the library file. Such selection will further facilitate the timing analysis phase later on, given that `Nanotime` has powerful netlist parsing tools which automatically identify Flip-Flop topologies and basic gates (i.e. INVs, NANDs, NORs). The resulting netlist consists of 1830 transistors and its gate-level schematic is shown in Figure 5.5.

5.4.2 Generating the variation-aware `netlist`$^+$

We apply Algorithm 1 that was presented in the previous Chapter to the gate-level datapath netlist. We select 150 iterations as an adequate number of MC samples [31]. Compared against pruning-free STA sessions, we achieve 18.69% and 18.29% minimum runtime and memory usage reduction respectively, with 0.84% maximum error. We observe that the performance is highly dependent on the design topology. That is, a significant portion of the datapath corresponds to variation-critical registers, reducing the performance of Algorithm 1 compared to results that we acquired for the ISCAS85 benchmarks (Section 4.4.1). Nonetheless, it is worth observing that our approach outperforms the built-in pruning command of the STA solver, achieving up to 16.87% extra runtime improvement. That can be attributed to the phasing of the built-in pruning that is performed after the parsing phase of all varying parameters [59]. On the contrary, our methodology is decoupled from the STA solver itself. An already "pruned" (in terms of variability) design is generated, reducing the parameters to be captured by the STA solver beforehand.

5.4.3 Functional Yield Analysis

We now move towards performing a complete reliability analysis, using the flow presented in Figure 5.3. We estimate the *functional yield* of the module over three years of operation. A sample of this design is defined as *functionally correct* iff the worst path exhibits correct timing behavior (i.e. positive slack), under a particular state of BTI-induced variability. We estimate the functional yield of the target circuit, namely the percentage of samples that exhibit correct functionality at different instances of circuit lifetime. Our first step is to identify a representative exploration space. We perform STA MC iterations using `Synopsys NanoTime` for a set of different operating points (f, V_{DD}) with no variability introduced, as

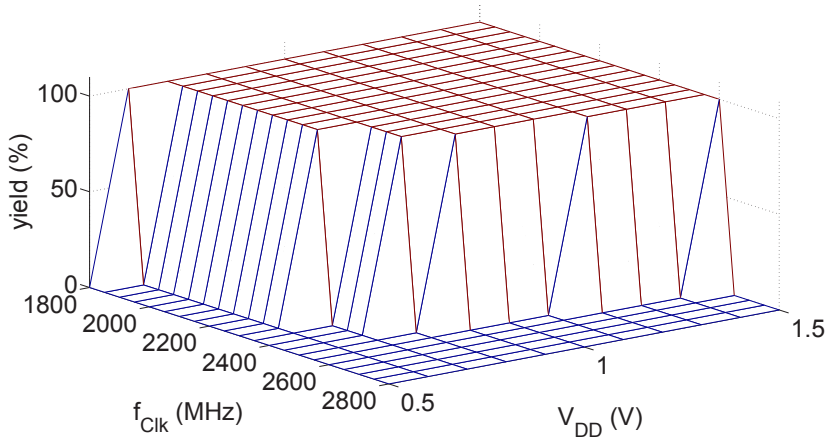


Fig. 5.6: Selecting the proper exploration space: "Optimal" behavior of our design with no variability introduced.

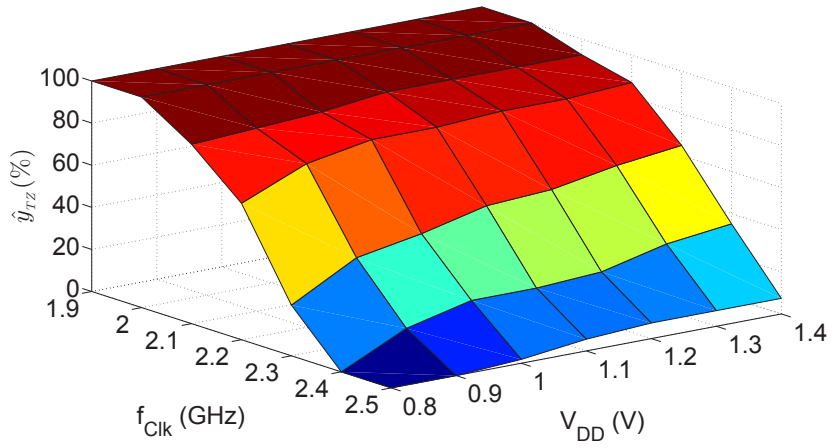
shown in Figure 5.6. Based on the "optimal" behavior of our design (i.e. 100% yield), we select the following 2D exploration space of (f, V_{DD}) pairs:

$$1.9 \text{ GHz} \leq f \leq 2.5 \text{ GHz}, \text{ step} = +0,1 \text{ GHz} \tag{5.2a}$$

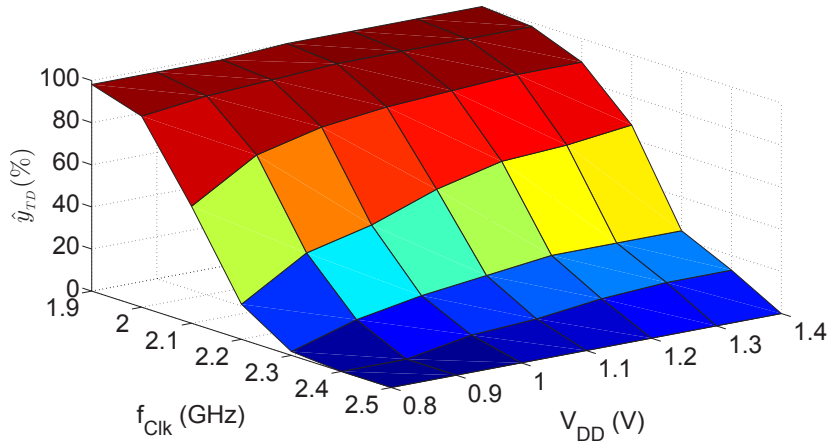
$$0.8 \text{ V} \leq V_{DD} \leq 1.4 \text{ V}, \text{ step} = +0,1 \text{ V} \tag{5.2b}$$

We introduce time-zero and time-dependent ΔV_{th} variations based on the estimates $\mu(t)$ and $\sigma(t)$ derived in Section 5.2. Again, we select 150 MC samples. As we have already mentioned, the ΔV_{th} distributions remain fairly constant after the time-zero instance. Thus, it is reasonable to select one more instance to capture BTI for the rest of the transistor lifetime. We estimate the functional yield at time-zero instance (Figure 5.7a) and after three years of operation (Figure 5.7b). We can observe that BTI significantly affects the proper functionality of our design.

More specifically, we notice the impact immediately after including the time-zero variability. We have non-functional samples starting from 2.1 GHz. That is, compared to the 2.5 GHz optimal frequency of the design without variability, we observe an $f_{\max}$ degradation by 16%. Such degradation is more intense after three years of operation, with the first non-functional samples appearing at 1.9 GHz, resulting in a greater frequency degradation of 24%. Another important observation is that the estimated yield is 0% for frequencies greater than 2.4 GHz in Figure 5.7b. In other words, the design loses its functionality com-



(a) Yield estimation $\hat{y}_{\mathrm{TZ}}$ at time-zero instance



(b) Yield estimation $\hat{y}_{\mathrm{TD}}$ after 3 years of operation

Fig. 5.7: Functional yield analysis : Applying the proposed EDA methodologies.

pletely for higher frequencies during its lifetime due to time-dependent variability. Thus, we stress the importance of properly accounting for BTI-induced variability, in order to achieve functional designs.

5.4.4 Power-Aware Reliability Analysis

We substantiate the usability of our flow as a useful aid for BTI-aware design techniques. A popular power management technique is *voltage scaling*. Figure 5.7 shows that the number of functional samples decreases as we reduce the supply voltage. As a consequence, decreasing V_{DD} to meet a power constraint could lead to an increased number of non-functional modules. Such observation is of great importance in terms of power-aware design. We define a sample as *functionally correct* for a specific clock frequency f, iff it can meet the timing constraints (i.e. positive slack), without exceeding a specific power constraint.

To select a representative constraint, we compute the power consumption of the data-path through SPICE simulation for each (f, V_{DD}) pair. We acquire power values varying from 2.07 W up to 9.17 W. We set a power constraint of $P_{\max} = 5$ W. For varying maximum clock frequencies, we estimate the functional yield (Figure 5.8), assuming a 95% confidence interval [81]. The yield estimations of this Figure provide useful hints regarding power-aware design under BTI variability. For higher frequencies, the functional yield is reduced below 90% under *voltage scaling*, after three years of operation. This observation is fully consistent with a recent, BTI-aware, mitigation technique that suggests progressively increasing supply voltage [31].

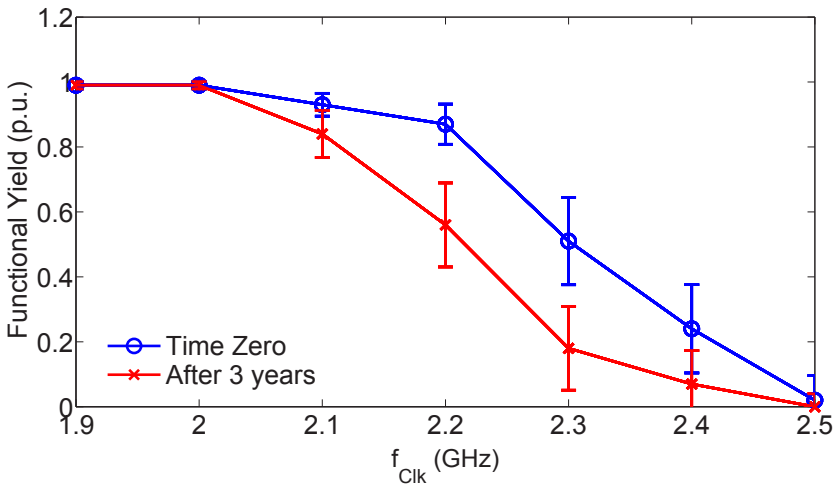


Fig. 5.8: Power-aware reliability analysis: Functional yield of datapath design for varying clock frequencies.

5.4.5 BTI-Induced Critical Path Re-ordering

Finally, prior art has already provided hints towards capturing the BTI-induced degradation of the critical path [24, 74]. Nonetheless, the experimental results are limited to simple case studies of logic subblocks (i.e. adders, multipliers etc). In the current Subsection, we fully investigate such observation for an entire CPU module with explicit timing analysis results. In Table 5.2, we sort paths based on delay values acquired using `Synopsys NanoTime` for one sample of our reliability analysis. For sake of brevity, we present the first few delay values. By highlighting the critical-path with no variability introduced, it is evident that paths that are not considered critical if BTI is ignored may finally dictate the total circuit delay when time-zero/-dependent activity is incorporated. Thus, we fully substantiate the impact of BTI-induced variability with respect to critical path re-ordering.

Table 5.2: Capturing critical path re-ordering while incorporating BTI-induced variability.

	No variability		**Time-zero**		**After 3 years**	
	Delay(ns)	**Path**	**Delay(ns)**	**Path**	**Delay(ns)**	**Path**
1	0.438	**RdB - X31**	0.496	RdB - X29	0.514	RstR - X24
2	0.436	RdB - X30	0.492	RdB - X24	0.507	RdB - X25
3	0.436	RdB - X29	0.492	RdB - X25	0.505	RdB - X25
4	0.436	RdB - X28	0.489	RdB - X26	0.499	RstR - X26
5	0.435	RdB - X27	0.487	RdA - X7	0.498	RdB - X7
6	0.435	RdB - X26	0.485	RstR - X29	0.493	RdB - X29
7	0.434	RdB - X25	0.478	**RdB - X31**	0.490	RdB - X31
8	0.433	RdB - X24	0.475	RstR - X24	0.490	RstR - X24
9	0.431	RdA - X15	0.475	RdB - X30	0.482	RdB - X30
	...	...	...	...	...	...
20	0.431	RdA - X4	0.469	RdB - X18	0.473	**RdB - X31**

5.5 Conclusion

In this Chapter, we propose a design flow for reliability analysis under BTI-induced variability. We exploit a novel, pseudo-transient BTI approach which alleviates the complexity of atomistic BTI models, while retaining time-zero/-dependent variability. We also take advantage of the pruning methodology presented in the previous Chapter, in order to efficiently capture the timing impact of transistor variability. We demonstrate the usability of

our framework as a useful aid for reliability analysis through realistic testcases. Until now and to the best of our knowledge a time- and workload-dependent yield analysis of an entire CPU module has never been performed with the atomistic BTI model. For a datapath design, we capture the impact of BTI and we present the evolution of the functional yield for three years of operation. We also provide useful hints regarding power-aware design under BTI variability. Finally, we substantiate the impact of BTI-induced variability to critical path re-ordering.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

As CMOS technology approaches nanometer scales, numerous sources of transistor variability have emerged. In order to mitigate transistor degradation at the system level and eventually meet lifetime reliability requirements of modern electronics, designers need accurate yet efficient EDA methodologies to capture the variability phenomena. A large portion of those reliability threats are related to threshold voltage variations, resulting to timing failures due to an overall increase of the IC's delay. Thus, it is important to develop EDA approaches that efficiently account for V_{th} fluctuations.

Recent literature supports a "paradigm shift" to workload- and time-dependent variability modeling. That is, the threshold voltage degradation for nominally identical transistors is stochastically distributed in both magnitude and time. Given this stochastic nature of V_{th} variability, it is imperative to develop tools that employ statistical methods. Efficient Monte Carlo based techniques are necessary to capture the timing impact of transistor degradation. It is therefore essential to alleviate the increased CPU overhead of Monte-Carlo (MC) based variability-aware timing analysis, while maintaining the accuracy and the novel features of the atomistic model.

To contribute to this research direction, we first propose a linear regression model in Chapter 3. By fitting our model to actual timing analysis results and V_{th} values, we achieve accurate prediction of delay values. We further exploit the results of the proposed model to accelerate STA sessions. Given the regression coefficients, we are able to identify the subset of transistors of the IC that should be tracked in order to accurately predict the

path delays. As a result, we reduce the transistor inventory that should be monitored by the STA solver. For representative test cases, we achieve acceleration and reduced memory usage for commercial STA tools with negligible accuracy loss.

The path pruning algorithm presented in Chapter 4 provides the reliability engineer with an alternative method to accelerate STA sessions. We develop a set of fully automated scripts to port the analysis to the transistor level and to introduce variability to STA MC iterations. The proposed EDA flow annotates the variation-critical part of a circuit, by identifying the paths (and eventually their transistors) that should be monitored for V_{th} fluctuations. It is worth observing that such result is highly applicable to different circuit-solvers, since the variability-aware `netlist`$^{+}$ could be given as an input to either transistor-level STA tools or SPICE-like kernels. For several ISCAS benchmarks, we verify the efficiency of our approach by achieving reduced execution times in comparison to the commercial `Synopsys NanoTime` and the open-source `ngspice` solver.

Given the achieved efficiency, the path pruning algorithm presented in Chapter 4 is suited to processor-wide reliability analysis. Hence, we combine the pruning method with the accuracy of a novel atomistic modeling approach, in order to capture the impact of transistor variability on reliability metrics, such as the functional yield and the maximum clock frequency of an entire CPU module. The resulting EDA flow is presented in Chapter 5. As we have already stressed in Chapter 2 and to the best of our knowledge, this is the first study that employs complete reliability analysis for an entire CPU module, while exploiting the accuracy of atomistic BTI models. Through an extensive exploration for a CPU datapath, we demonstrate the usability of our methodology as a useful aid for reliability analysis. First, we apply the path pruning algorithm to identify the variation-critical paths of the circuit under test. We then present the evolution of functional yield for three years of operation. We also investigate the design degradation under power management techniques (i.e voltage scaling) and critical path re-ordering.

In summary, the current thesis proposes EDA methods to accelerate timing analysis solvers. The purpose of our work is to develop efficient variability-aware EDA approaches to be incorporated with a comprehensive design flow for credible reliability analysis. The main differentiator in comparison to previous works is that we do not limit our exploration to simple CMOS gates and logic subblocks of reduced functional complexity, but we accelerate STA sessions of an entire CPU module. That way, we substantiate the usability of the proposed framework as a useful aid for variability-aware design techniques.

6.2 Future Work

There are many opportunities for future research in the EDA methodologies presented in the current thesis. While the approaches presented in both Chapters 3 and 4 are applied to MOSFETs, a flow that applies to other transistor types can be also be considered. Moreover, given the high applicability of the developed tool chains to several benchmarks, more test cases could be investigated. A summary of the main ideas that could either explored or they are currently under development is listed below.

Variability Analysis for Novel Transistor Architectures

A notable recommendation for future work is the employment of our methodologies to capture the timing impact of the variability observed in new transistor types, such as FinFETs and nanowire FETs. Recent literature shows that conventional transistor designs are being replaced by advanced 3D architectures. Even though these designs have emerged as important candidates for technology scaling within the following years [7], they exhibit increased transistor-to-transistor variations. According to the latest ITRS edition, the reliability issues of novel 3D architectures are listed among the most difficult challenges to be addressed within the next five years [7].

Recent works have successfully applied the time-dependent atomistic variability models to state-of-the-art transistor architectures [86, 87, 88]. These defect-centric modeling approaches inherently account for other V_{th} degradation mechanisms as well, such as the Random Telegraph Noise (RTN) [89] and the Channel Hot Carrier (CHC) [90]. The respective models capture stochastic trap behavior by using multiple statistical distributions, thus providing a unified perspective of several variability phenomena [91, 92].

The aforementioned observations lead to a straight-forward way to extend our proposed methodologies to FinFET-based netlists, while capturing multiple sources of V_{th} fluctuations. It is important to notice that all the developed tool chains receive the modelcard as a user-defined input. Thus, we could simply replace the currently used technology PTM files for MOSFETs with the respective PTM-MG models for FinFETs [75]. Moreover, the scripts that introduce variability to the transistor instances can be easily updated to include more than one source of variability (i.e. multiple distributions could be hard-coded at the proper portion of our code base). These additions will enable variation-aware timing analysis for FinFET netlists, while the current user-friendly flows will be fully maintained.

That way, the presented EDA approaches will be further enhanced as a meaningful option for reliability analysis.

Variability Analysis of VLSI DSP Modules

Another research direction that is currently under development is the employment of the regression models to accelerate variability analysis of Digital Signal Processing (DSP) designs. A growing number of on-chip co-processing components consists of modules dedicated to hardware acceleration [93, 94]. The reliability of these modules and their efficiency to carry out repetitive DSP operations can be consider especially important for the reliable performance of the entire design. Thus, it is essential to accelerate timing analysis of DSP circuits by exploiting the EDA methods proposed in Chapter 3.

The largest portion of several DSP applications (e.g. Fast-Fourier Transform, FIR filters) include Add-Multiply (AM) operations, which implement the $Z = X \cdot (A + B)$ computation. As a consequence, the design of the AM units of a co-processor is a challenging task that involves several power, critical delay and area trade-offs. Aiming to optimize the AM operation, prior art proposes fusion techniques based on the recoding of the $Y = A + B$ sum in its Modified Booth (MB) form [95, 96]. Nonetheless, the majority of these works focus on reporting the nominal behavior of their designs against previous implementations, without accounting for the presence of transistor variability. We can therefore use a representative design and evaluate its performance under V_{th} fluctuations.

A state-of-the-art DSP design that proposes a fused modulo $2^n - 1$ AM unit is presented in [97, 98]. Such module can be a perfect candidate to assess the applicability of our EDA flow to VLSI arithmetic topologies. We can first fit the regression model to delay and V_{th} values. By following the analysis described in Chapter 3, we can accelerate STA simulations of AM unit. That way, we can efficiently perform MC iterations and investigate the critical path sensitivity under transistor degradation.

Bibliography

[1] J. Forster and L. Miller, "The effect of surface treatments on point-contact transistor characteristics," *Bell System Technical Journal, The*, vol. 35, no. 4, pp. 767–811, Jul. 1956.

[2] F. Reynolds and J. Stevens, "Process-parameter variability in the manufacture of m.o.s. integrated circuits," *Proceedings of the Institution of Electrical Engineers*, vol. 124, no. 6, pp. 505–507, Jun. 1977.

[3] D. Kennedy, P. Murley, and R. O'Brien, "A Statistical Approach to the Design of Diffused Junction Transistors," *IBM Journal of Research and Development*, vol. 8, no. 5, pp. 482–495, Nov. 1964.

[4] K. O. Jeppson and C. M. Svensson, "Negative bias stress of MOS devices at high electric fields and degradation of MNOS devices," *Journal of Applied Physics*, vol. 48, no. 5, pp. 2004–2014, May 1977.

[5] L. Geppert, "The amazing vanishing transistor act," *IEEE Spectrum*, vol. 39, no. 10, pp. 28–33, Oct. 2002.

[6] M. Miranda, "When every atom counts," *IEEE Spectrum*, vol. 49, no. 7, pp. 32–32, Jul. 2012.

[7] ITRS, 2013., http://www.itrs.net/.

[8] E. Sperling, "Will 10nm Be The Last Big Node?" Semiconductor Engineering, http://semiengineering.com/will-10nm-be-the-last-big-node/.

[9] D. Lammers, "The Era of Error-Tolerant Computing," IEEE Spectrum, http://spectrum.ieee.org/semiconductors/processors/the-era-of-errortolerant-computing.

[10] T. Caulfield, "ASMC 2015 Keynote," GlobalFoundries, Santa Clara, SA, http://www.semi.org/en/node/54631.

[11] M. Miranda, "The Threat of Semiconductor Variability," IEEE Spectrum, http://spectrum.ieee.org/semiconductors/design/the-threat-of-semiconductor-variability.

[12] D. Stamoulis, D. Rodopoulos, B. H. Meyer, D. Soudris, and Z. Zilic, "Linear regression techniques for efficient analysis of transistor variability," in *2014 21st IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, Dec. 2014, pp. 267–270.

[13] D. Stamoulis, D. Rodopoulos, B. H. Meyer, D. Soudris, F. Catthoor, and Z. Zilic, "Efficient reliability analysis of processor datapath using atomistic bti variability models," in *2015 25th ACM Great Lakes VLSI Symposium (GLSVLSI)*, May 2015.

[14] D. K. Schroder and J. A. Babcock, "Negative bias temperature instability: Road to cross in deep submicron silicon semiconductor manufacturing," *Journal of Applied Physics*, vol. 94, no. 1, pp. 1–18, 2003.

[15] Y. Tsividis and C. McAndrew, *Operation and Modeling of the MOS Transistor*, ser. Oxford Series in Electrical and Computer Engineering. Oxford University Press, 2011.

[16] M. Toledano-Luque, B. Kaczer, E. Simoen, R. Degraeve, J. Franco, P. Roussel, T. Grasser, and G. Groeseneken, "Correlation of single trapping and detrapping effects in drain and gate currents of nanoscaled nFETs and pFETs," in *Reliability Physics Symposium (IRPS), 2012 IEEE International*, Apr. 2012, pp. XT.5.1–XT.5.6.

[17] B. Kaczer, T. Grasser, J. Martin-Martinez, E. Simoen, M. Aoulaiche, P. Roussel, and G. Groeseneken, "NBTI from the perspective of defect states with widely distributed time scales," in *Reliability Physics Symposium, 2009 IEEE International*, Apr. 2009, pp. 55–60.

[18] E. Cartier and A. Kerber, "Stress-induced leakage current and defect generation in nFETs with HfO2/TiN gate stacks during positive-bias temperature stress," in *Reliability Physics Symposium, 2009 IEEE International*, Apr. 2009, pp. 486–492.

[19] S. Ogawa and N. Shiono, "Generalized diffusion-reaction model for the low-field charge-buildup instability at the Si-SiO_2 interface," *Physical Review B*, vol. 51, no. 7, pp. 4218–4230, Feb. 1995.

[20] H. Kufluoglu and M. Alam, "A generalized reaction–diffusion model with explicit h–h_2 dynamics for negative-bias temperature-instability (nbti) degradation," *Electron Devices, IEEE Transactions on*, vol. 54, no. 5, pp. 1101–1107, May 2007.

[21] A. Calimera, E. Macii, and M. Poncino, "Nbti-aware clustered power gating," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 16, no. 1, pp. 3:1–3:25, Nov. 2010.

[22] T. Grasser, B. Kaczer, W. Goes, H. Reisinger, T. Aichinger, P. Hehenberger, P. J. Wagner, F. Schanovsky, J. Franco, M. Luque, and M. Nelhiebel, "The paradigm shift in understanding the bias temperature instability: From reaction–diffusion to switching

oxide traps," *IEEE Transactions on Electron Devices*, vol. 58, no. 11, pp. 3652–3666, Nov. 2011.

[23] B. Kaczer, S. Mahato, V. de Almeida Camargo, M. Toledano-Luque, P. Roussel, T. Grasser, F. Catthoor, P. Dobrovolny, P. Zuber, G. Wirth, and G. Groeseneken, "Atomistic approach to variability of bias-temperature instability in circuit simulations," in *Reliability Physics Symposium (IRPS), 2011 IEEE International*, April 2011, pp. XT.3.1–XT.3.5.

[24] D. Rodopoulos, D. Stamoulis, G. Lyras, D. Soudris, and F. Catthoor, "Understanding timing impact of BTI/RTN with massively threaded atomistic transient simulations," in *2014 IEEE International Conference on IC Design Technology (ICICDT)*, May 2014, pp. 1–4.

[25] P. Weckx, B. Kaczer, M. Toledano-Luque, P. Raghavan, J. Franco, P. Roussel, G. Groeseneken, and F. Catthoor, "Implications of BTI-Induced Time-Dependent Statistics on Yield Estimation of Digital Circuits," *IEEE Transactions on Electron Devices*, vol. 61, no. 3, pp. 666–673, Mar. 2014.

[26] V. Huard, C. Parthasarathy, C. Guerin, T. Valentin, E. Pion, M. Mammasse, N. Planes, and L. Camus, "NBTI degradation: From transistor to SRAM arrays," in *Reliability Physics Symposium, 2008. IRPS 2008. IEEE International*, Apr. 2008, pp. 289–300.

[27] J. Franco, B. Kaczer, M. Toledano-Luque, P. Roussel, J. Mitard, L.-A. Ragnarsson, L. Witters, T. Chiarella, M. Togo, N. Horiguchi, G. Groeseneken, M. Bukhori, T. Grasser, and A. Asenov, "Impact of single charged gate oxide defects on the performance and scaling of nanoscaled FETs," in *Reliability Physics Symposium (IRPS), 2012 IEEE International*, Apr. 2012, pp. 5A.4.1–5A.4.6.

[28] M. Toledano-Luque, B. Kaczer, J. Franco, P. Roussel, T. Grasser, T. Hoffmann, and G. Groeseneken, "From mean values to distributions of BTI lifetime of deeply scaled FETs through atomistic understanding of the degradation," in *2011 Symposium on VLSI Technology (VLSIT)*, Jun. 2011, pp. 152–153.

[29] B. Kaczer, T. Grasser, P. Roussel, J. Franco, R. Degraeve, L. A. Ragnarsson, E. Simoen, G. Groeseneken, and H. Reisinger, "Origin of NBTI variability in deeply scaled pFETs," in *Reliability Physics Symposium (IRPS), 2010 IEEE International*, May 2010, pp. 26–32.

[30] M. Toledano-Luque, B. Kaczer, P. Roussel, T. Grasser, G. Wirth, J. Franco, C. Vrancken, N. Horiguchi, and G. Groeseneken, "Response of a single trap to ac negative bias temperature stress," in *Reliability Physics Symposium (IRPS), 2011 IEEE International*, April 2011, pp. 4A.2.1–4A.2.8.

[31] D. Rodopoulos, P. Weckx, M. Noltsis, F. Catthoor, and D. Soudris, "Atomistic pseudo-transient BTI simulation with inherent workload memory," *IEEE Transactions on Device and Materials Reliability*, vol. 14, no. 2, pp. 704–714, Jun. 2014.

[32] S. Rauch, "Review and Re-examination of Reliability Effects Related to NBTI-Induced Statistical Variations," *IEEE Transactions on Device and Materials Reliability*, vol. 7, no. 4, pp. 524–530, Dec. 2007.

[33] M. Bukhori, T. Grasser, B. Kaczer, H. Reisinger, and A. Asenov, "Atomistic simulation of RTS amplitudes due to single and multiple charged defect states and their interactions," in *Integrated Reliability Workshop Final Report (IRW), 2010 IEEE International*, Oct. 2010, pp. 76–79.

[34] B. Vaidyanathan and A. Oates, "Technology Scaling Effect on the Relative Impact of NBTI and Process Variation on the Reliability of Digital Circuits," *IEEE Transactions on Device and Materials Reliability*, vol. 12, no. 2, pp. 428–436, Jun. 2012.

[35] V. Huard, R. Chevallier, C. Parthasarathy, A. Mishra, N. Ruiz-Amador, F. Persin, V. Robert, A. Chimeno, E. Pion, N. Planes, D. Ney, F. Cacho, N. Kapoor, V. Kulshrestha, S. Chopra, and N. Vialle, "Managing SRAM reliability from bitcell to library level," in *Reliability Physics Symposium (IRPS), 2010 IEEE International*, May 2010, pp. 655–664.

[36] A. Subirats, X. Garros, J. Mazurier, J. El Husseini, O. Rozeau, G. Reimbold, O. Faynot, and G. Ghibaudo, "Impact of dynamic variability on SRAM functionality and performance in nano-scaled CMOS technologies," in *Reliability Physics Symposium (IRPS), 2013 IEEE International*, Apr. 2013, pp. 4A.6.1–4A.6.5.

[37] K. Aadithya, A. Demir, S. Venugopalan, and J. Roychowdhury, "SAMURAI: An accurate method for modelling and simulating non-stationary Random Telegraph Noise in SRAMs," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2011*, Mar. 2011, pp. 1–6.

[38] P. Weckx, B. Kaczer, M. Toledano-Luque, T. Grasser, P. Roussel, H. Kukner, P. Raghavan, F. Catthoor, and G. Groeseneken, "Defect-based methodology for workload-dependent circuit lifetime projections - Application to SRAM," in *Reliability Physics Symposium (IRPS), 2013 IEEE International*, Apr. 2013, pp. 3A.4.1–3A.4.7.

[39] D. Rodopoulos, S. B. Mahato, V. de Almeida Camargo, B. Kaczer, F. Catthoor, S. Cosemans, G. Groeseneken, A. Papanikolaou, and D. Soudris, "Time and workload dependent device variability in circuit simulations," in *2011 IEEE International Conference on IC Design Technology (ICICDT)*, May 2011, pp. 1–4.

[40] G. Lyras, D. Rodopoulos, A. Papanikolaou, and D. Soudris, "Hypervised transient spice simulations of large netlists amp; workloads on multi-processor systems," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2013*, March 2013, pp. 655–658.

[41] L. W. Nagel, "Spice2: A computer program to simulate semiconductor circuits," Ph.D. dissertation, EECS – UC, Berkeley, 1975.

[42] F. Catthoor, B. Kaczer, D. Rodopoulos, V. de Almeida Camargo, and S. Mahato, "Time and workload dependent circuit simulation," EPO; Applicant: imec, BE, Tech. Rep. EP 2 509 011 A1, 2012.

[43] Q. Tang, A. Zjajo, M. Berkelaar, and N. van der Meijs, "Transistor-level gate model based statistical timing analysis considering correlations," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2012*, Mar. 2012, pp. 917–922.

[44] F. Firouzi, S. Kiamehr, and M. Tahoori, "Statistical analysis of BTI in the presence of process-induced voltage and temperature variations," in *Design Automation Conference (ASP-DAC), 2013 18th Asia and South Pacific*, Jan. 2013, pp. 594–600.

[45] W. Wang, V. Reddy, B. Yang, V. Balakrishnan, S. Krishnan, and Y. Cao, "Statistical prediction of circuit aging under process variations," in *IEEE Custom Integrated Circuits Conference, 2008. CICC 2008*, Sep. 2008, pp. 13–16.

[46] V. Camargo, B. Kaczer, G. Wirth, T. Grasser, and G. Groeseneken, "Use of SSTA Tools for Evaluating BTI Impact on Combinational Circuits," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 2, pp. 280–285, Feb. 2014.

[47] Q. Tang, J. Rodriguez, A. Zjajo, M. Berkelaar, and N. van der Meijs, "Statistical transistor-level timing analysis using a direct random differential equation solver," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 2, pp. 210–223, Feb. 2014.

[48] S. Onaissi, F. Taraporevala, J. Liu, and F. Najm, "A fast approach for static timing analysis covering all PVT corners," in *2011 48th ACM/EDAC/IEEE Design Automation Conference (DAC)*, Jun. 2011, pp. 777–782.

[49] D. Blaauw, K. Chopra, A. Srivastava, and L. Scheffer, "Statistical Timing Analysis: From Basic Principles to State of the Art," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 4, pp. 589–607, Apr. 2008.

[50] Y. Kanoria, S. Mitra, and A. Montanari, "Statistical static timing analysis using Markov chain Monte Carlo," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2010*, Mar. 2010, pp. 813–818.

[51] P. Zuber, V. Matvejev, P. Roussel, P. Dobrovoln, and M. Miranda, "Exponent monte carlo for quick statistical circuit simulation," in *Integrated Circuit and System Design. Power and Timing Modeling, Optimization and Simulation*, ser. Lecture Notes in Computer Science, J. Monteiro and R. van Leuken, Eds. Springer Berlin Heidelberg, 2010, vol. 5953, pp. 36–45.

[52] R. Kanj, R. Joshi, and S. Nassif, "Mixture importance sampling and its application to the analysis of SRAM designs in the presence of rare failure events," in *2006 43rd ACM/IEEE Design Automation Conference*, 2006, pp. 69–72.

[53] M. Qazi, M. Tikekar, L. Dolecek, D. Shah, and A. Chandrakasan, "Loop flattening & spherical sampling: Highly efficient model reduction techniques for SRAM yield analysis," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2010*, Mar. 2010, pp. 801–806.

[54] J.-J. Nian, S.-H. Tsai, and S.-L. Huang, "A unified multi-corner multi-mode static timing analysis engine," in *Design Automation Conference (ASP-DAC), 2010 15th Asia and South Pacific*, Jan. 2010, pp. 669–674.

[55] S. Onaissi and F. Najm, "A Linear-Time Approach for Static Timing Analysis Covering All Process Corners," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 7, pp. 1291–1304, Jul. 2008.

[56] J. Lee and P. Gupta, "Discrete circuit optimization," *Foundations and Trends in Electronic Design Automation*, vol. 6, no. 1, pp. 1–120, 2012.

[57] J. Bhasker and R. Chadha, *Static Timing Analysis for Nanometer Designs: A Practical Approach*. Springer US, Sep. 2011.

[58] S. Raja, F. Varadi, M. Becer, and J. Geada, "Transistor level gate modeling for accurate and fast timing, noise, and power analysis," in *45th ACM/IEEE Design Automation Conference, 2008. DAC 2008*, Jun. 2008, pp. 456–461.

[59] *NanoTime, User Guide*, Version G-2012.06, Synopsys, Inc., 2012.

[60] *NanoTime, Transistor-level Static Timing Analysis Solution for Custom Designs*, Datasheet, Synopsys, Inc., 2010.

[61] Synopsys, News Release, "Synopsys NanoTime Enables Full Chip Transistor Level Timing Analysis on Cavium Networks OCTEON II Internet Application Processor," Synopsys, Inc., http://news.synopsys.com/index.php?s=20295&item=123222.

[62] W. Wang, S. Yang, S. Bhardwaj, S. Vrudhula, F. Liu, and Y. Cao, "The Impact of NBTI Effect on Combinational Circuit: Modeling, Simulation, and Analysis," *IEEE*

Transactions on Very Large Scale Integration (VLSI) Systems, vol. 18, no. 2, pp. 173–183, Feb. 2010.

[63] J. Carretero, E. Herrero, M. Monchiero, T. Ramirez, and X. Vera, "Capturing vulnerability variations for register files," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2013*, Mar. 2013, pp. 1468–1473.

[64] S. Ganapathy, R. Canal, A. Gonzalez, and A. Rubio, "Circuit propagation delay estimation through multivariate regression-based modeling under spatio-temporal variability," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2010*, Mar. 2010, pp. 417–422.

[65] A. Devgan, "Accurate device modeling techniques for efficient timing simulation of integrated circuits," in *, 1995 IEEE International Conference on Computer Design: VLSI in Computers and Processors, 1995. ICCD '95. Proceedings*, Oct. 1995, pp. 138–143.

[66] L. Guerra e Silva, J. Phillips, and L. Miguel Silveira, "Speedpath analysis under parametric timing models," in *2010 47th ACM/IEEE Design Automation Conference (DAC)*, Jun. 2010, pp. 268–273.

[67] S. Onaissi, K. Heloue, and F. Najm, "PSTA-based branch and bound approach to the silicon speedpath isolation problem," in *IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers, 2009. ICCAD 2009*, Nov. 2009, pp. 217–224.

[68] K. Heloue, S. Onaissi, and F. Najm, "Efficient block-based parameterized timing analysis covering all potentially critical paths," in *IEEE/ACM International Conference on Computer-Aided Design, 2008. ICCAD 2008*, Nov. 2008, pp. 173–180.

[69] K. Heloue, C. Kashyap, and F. Najm, "Quantifying robustness metrics in parameterized static timing analysis," in *IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers, 2009. ICCAD 2009*, Nov. 2009, pp. 209–216.

[70] K. Heloue, S. Onaissi, and F. Najm, "Efficient block-based parameterized timing analysis covering all potentially critical paths," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 4, pp. 472–484, Apr. 2012.

[71] H. Kukner, S. Khan, P. Weckx, P. Raghavan, S. Hamdioui, B. Kaczer, F. Catthoor, L. Van der Perre, R. Lauwereins, and G. Groeseneken, "Comparison of reaction-diffusion and atomistic trap-based BTI models for logic gates," *IEEE Transactions on Device and Materials Reliability*, vol. 14, no. 1, pp. 182–193, Mar. 2014.

[72] P. Weckx, B. Kaczer, H. Kukner, J. Roussel, P. Raghavan, F. Catthoor, and G. Groeseneken, "Non-monte-carlo methodology for high-sigma simulations of circuits under workload-dependent BTI degradation – application to 6t SRAM," in *Reliability Physics Symposium, 2014 IEEE International*, Jun. 2014, pp. 5D.2.1–5D.2.6.

[73] J. Fang and S. Sapatnekar, "Understanding the impact of transistor-level BTI variability," in *Reliability Physics Symposium (IRPS), 2012 IEEE International*, Apr. 2012, pp. CR.2.1–CR.2.6.

[74] H. Kukner, M. Khatib, S. Morrison, P. Weckx, P. Raghavan, B. Kaczer, F. Catthoor, L. Van der Perre, R. Lauwereins, and G. Groeseneken, "Degradation analysis of datapath logic subblocks under NBTI aging in FinFET technology," in *2014 15th International Symposium on Quality Electronic Design (ISQED)*, Mar. 2014, pp. 473–479.

[75] Predictive Technology Model, http://ptm.asu.edu/.

[76] H. Reisinger, T. Grasser, W. Gustin, and C. Schlunder, "The statistical analysis of individual defects constituting NBTI and its implications for modeling DC- and AC-stress," in *Reliability Physics Symposium (IRPS), 2010 IEEE International*, May 2010, pp. 7–15.

[77] N. Weste and K. Eshraghian, *Principles of CMOS VLSI design: a systems perspective*, ser. VLSI systems series. Addison-Wesley, 1985.

[78] M. Hansen, H. Yalcin, and J. Hayes, "Unveiling the ISCAS-85 benchmarks: a case study in reverse engineering," *IEEE Design Test of Computers*, vol. 16, no. 3, pp. 72–80, 1999.

[79] R. E. Poore, "GPU-accelerated time-domain circuit simulation," in *IEEE Custom Integrated Circuits Conference, 2009. CICC '09*, Sep. 2009, pp. 629–632.

[80] P. Nenzi and H. Vogt, "Ngspice users manual," Tech. Rep. Version 26plus, Feb. 2014.

[81] T. McConaghy, K. Breen, J. Dyck, and A. Gupta, *Variation-Aware Design of Custom Integrated Circuits: A Hands-On Field Guide*. Springer, Oct. 2012.

[82] H. Kufluoglu and M. Alam, "Theory of interface-trap-induced nbti degradation for reduced cross section mosfets," *Electron Devices, IEEE Transactions on*, vol. 53, no. 5, pp. 1120–1130, May 2006.

[83] M. Toledano-Luque, B. Kaczer, J. Franco, P. Roussel, M. Bina, T. Grasser, M. Cho, P. Weckx, and G. Groeseneken, "Degradation of time dependent variability due to interface state generation," in *VLSI Technology (VLSIT), 2013 Symposium on*, June 2013, pp. T190–T191.

[84] OpenRISC project, http://opencores.org/or1k/.

[85] *Design Compiler, User Guide*, Version C-2009.06, Synopsys, Inc., 2009.

[86] H. Kukner, P. Weckx, J. Franco, M. Toledano-Luque, M. Cho, B. Kaczer, P. Raghavan, D. Jang, K. Miyaguchi, M. Bardon, F. Catthoor, L. Van der Perre, R. Lauwereins, and G. Groeseneken, "Scaling of BTI reliability in presence of time-zero variability," in *Reliability Physics Symposium, 2014 IEEE International*, Jun. 2014, pp. CA.5.1–CA.5.7.

[87] J. Franco, B. Kaczer, G. Eneman, P. Roussel, T. Grasser, J. Mitard, L. Ragnarsson, M. Cho, L. Witters, T. Chiarella, M. Togo, W.-E. Wang, A. Hikavyy, R. Loo, N. Horiguchi, and G. Groeseneken, "Superior NBTI reliability of SiGe channel pMOSFETs: Replacement gate, FinFETs, and impact of Body Bias," in *Electron Devices Meeting (IEDM), 2011 IEEE International*, Dec. 2011, pp. 18.5.1–18.5.4.

[88] H. Mertens, R. Ritzenthaler, A. Hikavyy, J. Franco, J. Lee, D. Brunco, G. Eneman, L. Witters, J. Mitard, S. Kubicek, K. Devriendt, D. Tsvetanova, A. Milenin, C. Vrancken, J. Geypen, H. Bender, G. Groeseneken, W. Vandervorst, K. Barla, N. Collaert, N. Horiguchi, and A.-Y. Thean, "Performance and reliability of high-mobility Si0.55ge0.45 p-channel FinFETs based on epitaxial cladding of Si Fins," in *2014 Symposium on VLSI Technology (VLSI-Technology): Digest of Technical Papers*, Jun. 2014, pp. 1–2.

[89] W. Goes, F. Schanovsky, T. Grasser, H. Reisinger, and B. Kaczer, "Advanced modeling of oxide defects for random telegraph noise," in *2011 21st International Conference on Noise and Fluctuations (ICNF)*, Jun. 2011, pp. 204–207.

[90] M. Cho, P. Roussel, B. Kaczer, R. Degraeve, J. Franco, M. Aoulaiche, T. Chiarella, T. Kauerauf, N. Horiguchi, and G. Groeseneken, "Channel Hot Carrier Degradation Mechanism in Long/Short Channel -FinFETs," *IEEE Transactions on Electron Devices*, vol. 60, no. 12, pp. 4002–4007, Dec. 2013.

[91] T. Grasser, K. Rott, H. Reisinger, M. Waltl, J. Franco, and B. Kaczer, "A unified perspective of RTN and BTI," in *Reliability Physics Symposium, 2014 IEEE International*, Jun. 2014, pp. 4A.5.1–4A.5.7.

[92] J. Franco, B. Kaczer, N. Waldron, J. Roussel, A. Alian, M. Pourghaderi, Z. Ji, T. Grasser, T. Kauerauf, S. Sioncke, N. Collaert, A. Thean, and G. Groeseneken, "RTN and PBTI-induced time-dependent variability of replacement metal-gate high-k InGaAs FinFETs," in *Electron Devices Meeting (IEDM), 2014 IEEE International*, Dec. 2014, pp. 20.2.1–20.2.4.

[93] Y. Pu, J. Pineda de Gyvez, H. Corporaal, and Y. Ha, "An Ultra-Low-Energy Multi-Standard JPEG Co-Processor in 65 nm CMOS With Sub/Near Threshold Supply Voltage," *IEEE Journal of Solid-State Circuits*, vol. 45, no. 3, pp. 668–680, Mar. 2010.

[94] N. Ickes, G. Gammie, M. Sinangil, R. Rithe, J. Gu, A. Wang, H. Mair, S. Datla, B. Rong, S. Honnavara-Prasad, L. Ho, G. Baldwin, D. Buss, A. Chandrakasan, and U. Ko, "A 28 nm 0.6 V Low Power DSP for Mobile Applications," *IEEE Journal of Solid-State Circuits*, vol. 47, no. 1, pp. 35–46, Jan. 2012.

[95] J. Bruguera and T. Lang, "Implementation of the FFT butterfly with redundant arithmetic," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 43, no. 10, pp. 717–723, Oct. 1996.

[96] M. Daumas and D. Matula, "A Booth multiplier accepting both a redundant or a non redundant input with no additional delay," in *IEEE International Conference on Application-Specific Systems, Architectures, and Processors, 2000. Proceedings*, 2000, pp. 205–214.

[97] K. Tsoumanis, S. Xydis, C. Efstathiou, N. Moschopoulos, and K. Pekmestzi, "An Optimized Modified Booth Recoder for Efficient Design of the Add-Multiply Operator," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 4, pp. 1133–1143, Apr. 2014.

[98] K. Tsoumanis, K. Pekmestzi, and C. Efstathiou, "Fused modulo $2^n - 1$ add-multiply unit," in *2014 21st IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, Dec. 2014, pp. 40–43.